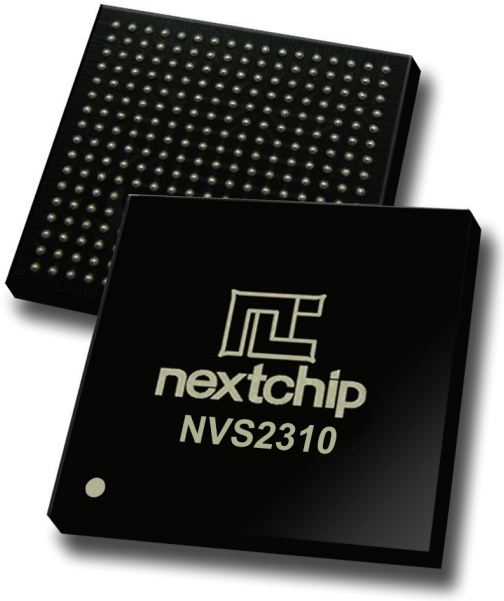
NVS2310 Datasheet



www.nextchip.com
Jun 01, 2011
Rev. 1.02

[주]넥스트칩 nextchip

The material in this data sheet is for information only. NEXTCHIP Co., Ltd. assumes no responsibility for errors or omissions and reserves the right to change, without notice, product specifications, operating characteristics, packaging, etc.
NEXTCHIP Co., Ltd. assumes no liability for damage resulting from the use of information contained in this document.

Important Notice

This product may include technology owned by Microsoft Corporation and in this case it cannot be used or distributed without a license from Microsoft Licensing, GP.

For customers who use licensed Codec ICs and/or licensed codec firmware of mp3:
"Supply of this product does not convey a license nor imply any right to distribute content created with this product in revenue-generating broadcast systems (terrestrial. Satellite, cable and/or other distribution channels), streaming applications(via internet, intranets and/or other networks), other content distribution systems(pay-audio or audio-on-demand applications and the like) or on physical media(compact discs, digital versatile discs, semiconductor chips, hard drives, memory cards and the like). An independent license for such use is required. For details, please visit http://mp3licensing.com".

For customers who use other firmware of mp3:
"Supply of this product does not convey a license under the relevant intellectual property of Thomson and/or Fraunhofer Gesellschaft nor imply any right to use this product in any finished end user or ready-to-use final product. An independent license for such use is required. For details, please visit http://mp3licensing.com".

For customers who use Digital Wave DRA solution:
"Supply of this implementation of DRA technology does not convey a license nor imply any right to this implementation in any finished end-user or ready-to-use terminal product. An independent license for such use is required."

For customers who use DTS technology:
"Supply of this implementation of DTS technology does not convey a license, exhaust DTS' rights in the implementation, or imply a right under any patent, or any other industrial or intellectual property right of DTS to use, offer for sale, sell, or import such implementation in any finished end-user or ready-to-use final product. Notice is hereby provided that a license from DTS is required prior to such use."
"This product made under license to U.S. Patents 5,451,942; 5,956,674; 5,974,380; 5,978,762; 6,487,535; 6,226,616 and/or foreign counterparts."
"© 1996 – 2010 DTS, Inc."

For customers who use Dolby technology:
"Supply of this Implementation of Dolby technology does not convey a license nor imply a right under any patent, or any other industrial or intellectual property right of Dolby Laboratories, to use this Implementation in any finished end-user or ready-to-use final product. It is hereby notified that a license for such use is required from Dolby Laboratories."

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial release
2011-04-22	1.01	∗ The revision number of “CHIPSPEC” was changed to 1.01. ∗The revision number of “PART 1. OVERVIEW” was changed to 1.01. ∗The revision number of “PART 2. SMU & PMU” was changed to 1.01. ∗The revision number of “PART 3. GRAPHIC BUS” was changed to 1.01. ∗The revision number of “PART 4. MEMORY BUS” was changed to 1.01. ∗The revision number of “PART 5. IO BUS” was changed to 1.01. ∗The revision number of “PART 6. HSIO BUS” was changed to 1.01. ∗The revision number of “PART 7. DISPLAY BUS” was changed to 1.01. ∗The revision number of “PART 8. VIDEO BUS” was changed to 1.01. ∗The revision number of “PART 9. CAMERA BUS” was changed to 1.01.
2011-06-01	1.02	∗The revision number of “PART 2. SMU & PMU” was changed to 1.02. ∗The revision number of “PART 5. IO BUS” was changed to 1.02. ∗The revision number of “PART 7. DISPLAY BUS” was changed to 1.02. ∗The revision number of “PART 9. CAMERA BUS” was changed to 1.02.

CHIP SPECIFICATION

NVS2310

Rev. 1.01

Apr 22, 2011

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial Release
2011-04-22	1.01	∗ Correct the register table and document format. ∗ Update min/max values for Recommended Operating Conditions/Recommended Operating Frequency ∗ Update min/max values for Electrical characteristics for Power supply, SD/MMC, camera interface, General I/O, HDMI PHY, ADC and GPSB ∗ Update NVS2310 I/O Type ∗ Correct Figure 5.25 & Figure 5.26 and Table 5.30.

TABLE OF CONTENTS

Contents

Figures

Tables

1 Introduction

For new, innovative user experience by PC-like web browsing, Full HD video, intuitive user interfaces, location based services, and next generation social networking applications, Nextchip presents the NVS2310 multimedia application processor. The NVS2310 is built and optimized for IP Camera.

NVS2310 integrates the ARM® 1176JZF-S microprocessor core with performance up to 800MHz (600MHz at 1.2V), and hardwired VPU/GPU/ISP to maximize multimedia experience at its peak level. The hardwired VPU enables 1080p full HD video encoding and decoding, and the GPU offers up to 20M polygon 3D graphic with OpenGL ES2.0 and Open VG 1.1. The ISP provides AWB, AE Control, Auto Focus, Lens Shade Corrector, Bad Pixel Correction, WDR, and etc. It can support various bases of operation systems such as Linux®, Android, and Windows® CE.

The NVS2310 is a great processor for networked devices which require the ease of connectivity functions: The USB OTG and USB OTG Host controllers will enable the data transmission between the multimedia device and storage devices. The Ethernet MAC controller provides internet connection, compliant to IEEE802.3 standard.

This innovative and rich feature set will expand the categories of applications yet to be imagined.

1.1 NVS2310 Features

Table 1.1 NVS2310 Features

Category	Description
PROCESSORS (ARM1176JZF-S)	• High-speed AMBA AXI Bus Interface • High performance integer processor - 8 stage pipelines - Separate load-store and arithmetic pipelines - Branch prediction with return stack • Instruction and Data MMU(Memory Management Units) - Micro TLB structures backed by a unified Main TLB. • Virtually indexed and physically addressed caches • 16KBs Instruction/Data Caches - Including a non-blocking data cache with Hit-Under-Miss(HUM) • Vector-Floating-Point(VFP) coprocessor support • ARM Jazelle technology for efficient embedded Java execution • JTAG interface for code debugging
MEMORY ORGANIZATION	• **Internal(On-Chip) Memory** - 24KB Boot-ROM (EHI, NAND, USB Boot with security and etc.) - 128KB Internal SRAM (IRAM0) - 64KB Internal SRAM (IRAM1) - 16KB Internal SRAM (Backup RAM) • **External(Off-Chip) Memory**[1] - LPDDR SDRAM : up to 200MHz(400Mbps) - LPDDR2 SDRAM : up to 400MHz(800Mbps) - DDR2 SDRAM : up to 400MHz(800Mbps) - DDR3 SDRAM : up to 400MHz(800Mbps) - LPDDR/LPDDR2/DDR2 SDRAM : Support 16/32 bit data bus - DDR3 SDRAM : Support 32 bit data bus
VIDEO CODEC	▪ **Decompressor**[2] **(Decoder) – up to 30fps @ Full-HD (1920x1080)** • H.263 - Up to Baseline Profile + AnnexI,J,K(RS=0 and ASO =0),T - Including Sorenson Spark - Max. bitrate : up to 30Mbps • MPEG 1/2 - Up to Main Profile @ High Level - Max. bitrate : up to 80Mbps • MPEG4-ASP - Up to Advanced Simple Profile Including DivX 3.x/4.x/5.x/6.x - Max. bitrate : up to 35Mbps • MPEG4-AVC(H.264) - Up to High Profile @ Level 5.1 - Max. bitrate : up to 40Mbps • VC-1 - Up to Advanced Profile @ Level 3.0 - Max. bitrate : up to 45Mbps • AVS - Jizhun Profile @ L6.2 - Max. bitrate : up to 40Mbps • MJPEG/JPEG - Up to 32M pixel/s - Max Image Size : 8192 x 8192 ▪ **Compressor**[3] **(Encoder) – up to 24fps ~ 30 fps @FHD(1920x1080)**

[1] The maximum memory clock is dependent on the operation voltage. Refer to Electrical specification
[2] The performance of the video decoding can be limited by the overall system bus traffic

	• H.263 : up to 30fps @ FHD(1920x1080p) • MPEG4-ASP : up to 30fps @ FHD(1920x1080p) • H.264 : up to 24fps ~ 30 fps @ FHD(1920x1080p) • MJPEG/JPEG : up to 32M pixel/s (Max 4096x4096)
GRAPHIC ENGINE	▪ **2D/3D Graphic** • High Geometry and Pixel Processing • Up to 20M polygon[4] • Full OpenVG v1.1 Support - Lines, Squares, Triangles, Points - Vector Graphics - ROP 3/4 - Arbitrary Rotation / Scaling - Alpha Blending - Multitexture BItBLT • Full OpenGL ES v2.0, v1.x Support - 4X /16X FSAA - Flat/Gouraud Shading - Perspective Correct Texturing - Point Sampling/Bilinear/Trilinear Filtering - Mipmapping - Multi Texturing - Dot3 Bump Mapping - Alpha Blending - Stencil Buffering (4-bit) - JSR 184 - Point Sprites - 2 bit per pixel Texture Compressing (FLXTC) - 4 bit per pixel Texture Compressing (ETC) ▪ **Overlay Mixer** − 8bpp (RGB332) − RGB (444, 454, 555, 565, 666, 888) − Alpha-RGB (444, 454, 555, 666, 888) − Sequential YUV (444, 422) − Separated YUV (444, 440, 422, 420, 411, 410) − Interleaved YUV (422, 420) − BitBLT (16 Raster Operations) − 3 Channel Source Mirror/Flip/90°, 180°, 270° Rotate − 1 Channel Destination Mirror/Flip/90°, 180°, 270° Rotate − 3 Channel Arithmetic Operation − 3 Channel YCbCr-to-RGB Color Space Converting − Overlay and Alphablending (2 overlay, 256-level alphablending) − Color LUT − Dithering
IMAGE ENHANCEMENT	• **Histogram Measurement** - Analyze the Luminance Components - Multi-frames Average Mode - User-defined Pixel Segments Support • **Contrast Enhancement** - User-defined Scaling Segments - Multi-frames Average Mode • **De-Interlacer** - Motion-adaptive and Pixel-based Processing - Film-mode Detection - Simple Edge-oriented Mode - Advanced Spatial-Temporal Mode • **Noise Reduction**

[3] The performance of the video encoding can be limited by the overall system bus traffic
[4] The performance of the polygon processing can be limited by the overall system bus traffic

	- Directional-Smoothing Filter - Temporal-Recursive Filter - Noise Estimation • **Sharpness** - Spatial High-pass Filter
DISPLAY INTERFACE	▪ **Display Controller** • 2 Display Controllers - Controller 0 has 3 image channels - Controller 1 has 3 image channels - Progressive or Interlaced Digital Video Output - 3 Channel Overlay / Chroma-Keying / Alpha-blending - Gamma Correction - Look-up table for Indexed or RGB Color - Contrast, Brightness, Hue Function Supported. ▪ **Supported Output Media** - TFT-LCD - HDMI Output : up to 1920x1080p - Composite TV-Out (NTSC/PAL) : NTSC(720x480), PAL(720x576) ▪ **Dual-Display Supported**[5] - Two types of supported media - CPU Type Main/Sub LCD : Time Shared
CAMERA INTERFACE	•*** Camera I/F 0 (parallel)** - CCIR-601/656 Interface - Camera Input Supported - 1 Channel Overlay / Chroma-Keying - Image Effect(Gray/Negative/Sepia/Emboss/Sketch & ETC.) - Up-scaling (x4), Down-scaling(x1/1024: X/32, Y/32)
TOUCH ADC	• **Touch Screen Interface** - 10/12 bits 16CH ADC - Dedicated 4 ch analog input for touch screen I/F - Shared 12 Channel analog input for general purpose[6]
AUDIO	• **Dual-I2S Master & Slave Interface** - Simultaneous 7.1 Channel /Stereo channel supported • **SPDIF Transmitter/Receiver** - 5.1 Channel Supported - Support Tx 2 Channel or Tx 1 Channel and SPDIF Rx 1Channel • **CD I/F** - I2S Slave Interface - Up to 2 Channel
STORAGE INTERFACE	• **Dual USB 2.0 OTG Interface** - Simultaneous Dual USB2.0 OTG I/F supported • **NAND Flash Interface** - 8 Bits / 16 Bits / 32 bits - Support 4 CS I/F - • **SD/MMC Controller** - SD, MMC/eMMC, SDIO

[5] The maximum resolution and combination of image channels can be determined by the the overall system bus traffic
[6] The usage of 12 ch analog inputs is restricted by the port configuration because the analog inputs are shared with digital signals

	- Up to 4 Channels (4 independent SD/MMC controller) • **Memory Stick Pro/Pro-HG Supported**
HOST INTERFACE	• **EHI(External Host Interface)** - 8, 16bits, 18bits - 2 Channels - Bypass to LCD Port (CPU Type)
STREAMING INTERFACE	• **TS Interface** - 2 Channel TS serial/parallel interfaces (up to 8bit)
NETWORK INTERFACE	• **Ethernet MAC controller** - IEEE 802.3-compliant MII/GMII - Support RGMII specification form HP/Marvell
SECURITY	• **AES** - key length 128/192/256 bit • **DES** - single/double DES - triple DES (2 key/3 key) • **MULTI2**
PERIPHERALS	• **UART** - Up to 6 Channels • **I2C** - Up to 6 Channel for I2C compatible interface - Configurable master/slave • **GPSB(General Purpose Serial-Bus – Master/Slave)** - Up to 6 Channels • **Infra-Red Remote Receiver** • **PWM generator** **-** Up to 4 Channel • **Timers** - Four 16-bit timers, two 20-bit timers - One 32-bit timer • **General DMA controller** - 12 Channels
PMU	• **RTC** - Power-Down Mode & Auto-wakeup • **Power Gating** - Internal power island for saving the current consumption.
PROCESS	• 45nm Low-Power CMOS

1.2 Block Diagram

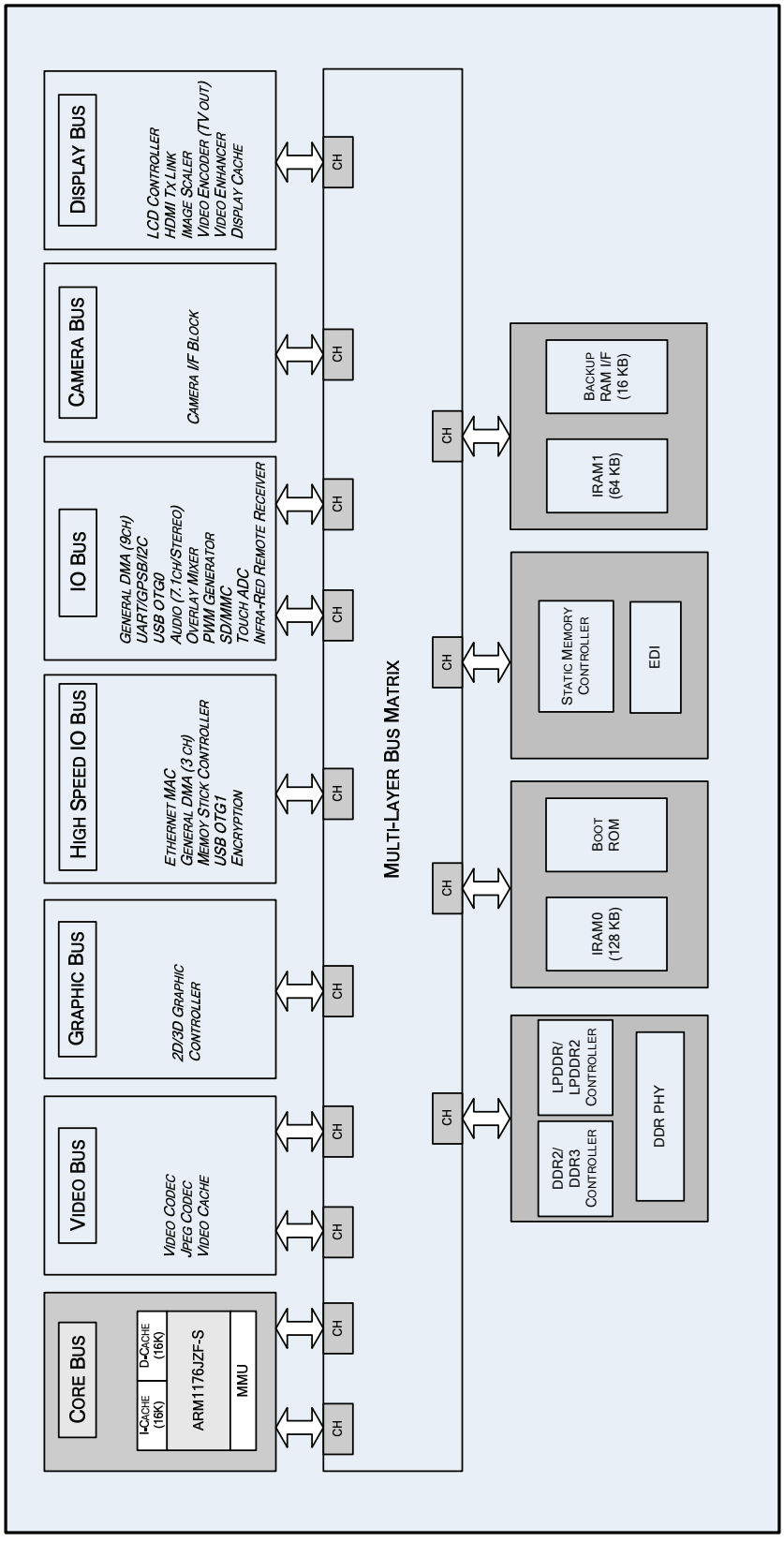


Figure 1.1 NVS2310 Functional Block Diagram

2 Hardware Features

Table 2.1 ARM1176JZFS Processor

ARM1176JZFS[7]	Key Features
Cache Organizations	• 16KBs/16KBs I/D Caches • I/D MMU Supported • Java Accelerator
Debug Interface	• JTAG Synchronized Ports with RTCK

Table 2.2 Memory Organization

Memory Map	Description
0x00000000	• This region is remapped to as follows. 1) If Remap is 000b, On-chip Boot-ROM 2) If Remap is 001b, On-chip memory0 3) If Remap is 010b, Off-chip DRAM 4) If Remap is 011b, Off-chip NOR . 5) If Remap is 100b, On-chip Backup RAM 6) If Remap is 101b, On-chip memory1 7) others is reserved
0x10000000	Internal SRAM0 (128 KB)
0x20000000	External Static Memory CS0
0x30000000	External Static Memory CS1
0x40000000 ~ 0x7FFFFFF	External DDR SDRAM
0x80000000	External Static Memory CS2
0x90000000	External Static Memory CS3
0xB0000000	Assigned to on-chip peripherals
0xC0000000	MMU Virtual Table (Do not use this area)
0xD0000000 ~ 0xD00FFFFF	Internal SRAM1 (64 KB)
0xD8000000 ~ 0xDFFFFFFF	PMU Backup RAM (16KB)
0xE0000000	Internal Boot ROM (24 KB)

Table 2.3 Video Controller

Video Codec	Key Features
Encoder	• **H.264 Encoding** - 24 ~ 30 fps@ Full-HD Resolution (1920x1280p) • **MPEG-4-ASP Encoding** - 30fps @ Full-HD Resolution (1920x1280p) • **H.263 Encoding** - 30fps @ Full-HD Resolution (1920x1280p)
Decoder	• **H.264 Decoding** - 30fps @ Full-HD Resolution • **MPEG4-ASP Decoding** - 30fps @ Full-HD Resolution • **H.263 Decoding** - 30fps @ Full-HD Resolution • **VC-1 Decoding** - 30fps @ Full-HD Resolution

7 If more detailed information is required, refer to the ARM1176JZFS Technical Reference Manual on ARM site.

Table 2.4 Camera Interface 0

CAMERA I/F 0	Key Features
Various Input Formats	• CCIR601/656 4:2:2 • Down Scaling for Preview Display : up to X/32, Y/32 • Change the Image size and windowing. • Support the master clock for camera module.
Camera Processing Functions	• Reconfigurable Packing the Pixel Data • Dispatching the Pixel Data into Y/Cb/Cr • Horizontal and Vertical Window Clipping • Overlaying the Background Frame for Still or Moving Pictures - Chroma-Keying - Alpha-blending (0%, 25%, 50%, 75%, 100%, XOR) • Support the Master Clock for Camera Module → w/o External Oscillator
Maximum Resolutions	• up to 120MHz[8] for Still Image[9]

8 The maximum frequency can be limited by the timing specification of the camera sensor or external device.
9 The maximum resolution can be limited by the system configuration.

Table 2.5 Camera Interface 1

CAMERA I/F 1	Key Features
Various Input Formats	▪ Parallel interface - ITU-R BT 601 compliant video input supporting YCbCr and RGB Bayer - ITU-R BT 656 compliant video input supporting YCbCr - Support up to 12 bit camera interface • MIPI interface - D-PHY spec v0.90 compliant - Support up to 4 data lane - Support up to 1G bps per lane - YUV420 8bit/10bit, cosited/non-cosited - YUV422 8bit/10bit - RGB 444/555/565/666/888 - RAW 6/7/8/10/12 bit
Camera Processing Functions	• Image processing pipeline - Black level compensation - Bad pixel detection and correction (support cluster correction) - Lens shade correction (Vignetting) - Auto focus measurement - Auto exposure support by brightness measurement - Auto white balance support - Enhanced color interpolation (RGB Bayer demosaicing) - Advanced de-noise filter (pre/post) - Sharpening/Blurring - Chromatic aberration correction - Histogram calculation - Color correction matrix (cross-talk matrix) - Gamma correction - WDR (Wide Dynamic Range) control - Contrast/saturation/brightness/hue - Mechanical shutter/flash control • Image Effect - Emboss/sketch/sepia/gray scale/selection/negative etc • Superimpose - Support 1 image overlay channel - Chroma-Keying mode • Image Scaler - 2 independent image scalers for capture/preview path each - • Image Rotate - 90/180/270 rotation • JPEG Encoding - Support up to 12M pixel capture - Support up to 15 fps 12 M pixel still image capture - JFIF1.02 stream generator and programmable quantization and Huffman tables
Maximum Resolutions	• up to 12M pixel still capture • up to Full-HD (1920x1280p) video camcording

Table 2.6 Video Output Interface

Video Output Interface	Key Features
Color TFT LCD	• Various type image sources - RGB565, RGB555, RGB666, RGB24, YCbCr4:2:0, YCbCr4:2:2 • Various type YCbCr4:2:0 and YCbCr4:2:2 to RGB converter • Parallel 24bits and 18bits pixel data output • 6(R):6(G):6(B)bits and 8(R):8(G):8(B)bits pixel data output
Mono/Color STN LCD	• Mono: 1, 2, 4bpp image source • Color: 8(332), 12(444), 555, 565 bpp image source • 4 and 8-bit pixel data interface
NTSC/PAL Encoder Interface	• CCIR601/656 interlace/non-interlace • RGB to YCbCr4:2:2 converter
NTSC/PAL composite output	• Supports all NTSC and PAL formats (NTSC-M/4.43, PAL-B/D/G/H/I/M/N/Combination N)
HDMI Output[10]	• The supported formats are - 1920x1080p @ 60Hz - 1920x1080i @ 30Hz - 1280x720p @ 30Hz - 720x480i @ 60Hz - 720x480p @ 60Hz - etc.
MIPI Output	• Compliant to MIPI DSI Standard Specification V1.01r11 • Maximum resolution range : up to XGA I(1024x768) • Support 1/2/3/4 data lines • Support pixel format : 16/18/24 bpp
Image Processing	• OSD/Overlay: can mix up to 3 image sources. - Channel 0 /Channel 1 has the 3 overlay channels. - Chroma-keying - 256 level Alpha-blending - Contrast/Brightness/Hue Control - Simple Gamma Correction Supported - LUT for each image channels • Virtual Window: Panning / Sliding the Window • Subsampling: 1/2, 1/3, 1/4, 1/8 • Duplication: x2, x3, x4, x8

10 The maximum resolution can be limited by the system configuration.

Table 2.7 DAI/CDIF Controller

I2S (DAI/CDIF)	Key Features
DAI (Digital Audio Interface)	• System clock: 256fs, 384fs, 512fs. • Maximum 7.1 channel supported • 7.1/Stereo dual-channel supported • Support of Master/Slave Mode with Reconfigurable Clock Polarity • Wide Range of Sampling Frequency in Audio application : 8kHz, 16kHz, 11.05kHz, 24kHz, 32kHz 44.1kHz, 48kHz • Supports the I2S (MSB Justified Mode) • Controls the Digital Audio Volume over the range 0dB to -90dB • Using 2 Double Buffers for Audio I/O Data
CDIF (CD Interface)	• CD Interface for Feasible Implementation of CD Application • Slave Mode • I2S (LSB Justified Mode)

Table 2.8 SPDIF Controller

SPDIF	Key Features
General Features	• Transmitter/Receiver Included • Bit Rate is 64 times the sampling frequency • Configurable 16/24 Bits Mode
Maximum Operating Frequency	• 24MHz Output Data-Rate - SPDIF Clock = 12.288MHz, Ratio = 1 - 3.072Mbps / 48kHz (Data Rate)

Table 2.9 External Device Interface

External Device Interface	Key Features
Static Memory Controller	• Support of 4 Types Static Memory (NAND/IDE/ROM/SRAM) • Controllable Setup / Pulse Width / Hold Time • 8/16 Bits Width

Table 2.10 USB 2.0 OTG(On-The-GO)

USB 2.0 OTG	Key Features
General Feature	• Compliant USB2.0 Specification • Support Interrupt, Bulk Transfer • Support FS/HS dual mode operation • 16bit interface • FIFO size configuration
USB DMA	• 3 Channel Scattered DMA (EP1,EP2,EP3) • Support 16/32bit MCU interface • Single / Fly mode • 4x32 FIFO for Each Endpoint
PHY Interface	• On-Chip UTMI PHY Parallel Interface
Maximum Operating Frequency	• 12 External Oscillator (Main Oscillator) • 30MHz with 16bits parallel interface

Table 2.11 nano PHY for USB2.0 OTG

UTMI PHY	Key Features
Supported Specification	• Compliant with USB 2.0 Transceiver Macrocell Interface Spec. Ver-1.04
General Features	• 480Mbps High Speed / 12Mbps Full Speed, FS Only, 1.5Mbps Low Speed • Separate 8/16 bit Unidirectional Parallel Interface • Dual-Mode Device Support (HS/FS) • Data and Clock Recovery from Serial Data on the USB Connector • SYNC/End-Of-Packet Generation and Checking • Bit Stuffing and unstuffing, Bit-stuffing Error Detection • NRZI Encoding/Decoding • Support of Suspend, Resume, Remote Wakeup Operations • Integrated HS and FS Termination and Signaling Switching • On-Chip PLL for 480Mbps • Low Power Dissipation while Active, Idle, or on Standby
System Features	• 45-ohm Termination / 1.5k Pull-up 15k Pull-down Integrated • Minimal External Components – Single Resistor
Maximum Operating Frequency	• up to 480MHz

Table 2.12 External Host Interface

EHI	Key Features
SRAM Type Interface	• 68/80 Series Interface with 8/16 Bit Width • Burst Transfer and Address Auto-Increment • Internal Interrupt Generation by an External Host Device • Semaphore Register for Improving Data Transfer Efficiency • READY can be Checked via Status Register and Pin. • LOCK MODE: External Host Device can Occupy System bus without any Handover.
Host Booting Procedure	• Configures 8/16 Bits Host Booting Mode • Host Downloads the Program into On-Chip SRAM or Off-Chip Memory • Restarts with Downloaded Program Code
Peak Access Bandwidth	• 8 Bits Configuration : 20MB/s • 16 Bits Configuration : 40MB/s

Table 2.13 SD/MMC Controller

SD/MMC	Key Features
Supported Specification	• SD ver.3.0 • SDIO ver.2.0 • eMMC ver.4.4 (Support booting from eMMC 4.4 Card)
General Features	• Automatic CRC Generation & Checking the Data/Command • Data transmit/receive FIFO (32bits x 8) • Supported SD/MMC Mode - 1 Bit Serial or 4 Bit Parallel SD - 1 Bit Serial for MMC - 4/8 Bits Parallel Transfer • External DMA Handshaking for Burst & Fast Transfer
Maximum Frequency	• up to 50MHz[11] • Clock generation block Inside

Table 2.14 Memory Stick Controller

Memory Stick	Key Features
Supported Specifications	• Memory Stick Ver.1.x • Memory Stick Pro • Memory Stick Pro-HG
General Features	• Data transmit/receive FIFO (64bits x 4) • External DMA Handshaking for Burst & Fast Transfer
Maximum Operating Frequency	• Memory Stick serial clock (Serial : 20MHz, Parallel : 40MHz)

11 The maximum operating frequency for storage devices can be limited by the system configuration and corresponding interface ports.

Table 2.15 Nand Flash Controller

NAND I/F	Key Features
NAND I/F	• Automatic Detection of External READY Signal • Configurable Cycle Times based-on Bus Frequency • 8bit, 16bit, 32bit Interface to Buffer Memory • 16x32bits FIFO Included • External DMA Handshaking for Burst and Fast Transfer • Dedicated DMA for Burst and Fast Transfer • Dedicated MEMORY(2048Bytes) for system related data and ECC
External Configuration	• 1 NAND - Single 8 bit NAND / Single 16bit NAND • 2 NAND - Double Series 8 bit NAND / Double Series 16 Bit NAND • 4 NAND - Double Series 16 bit NAND & Quad Series 8 Bit NAND
SLC	• 2 Bit Error Detection & 1 Bit Error Correction per 256 bytes.
MLC	• 4/6/12/16/24 Bit Error Detection/Correction Based on BCH Algorithm • 8x32bits FIFO

Table 2.16 UART Interface

UART	Key Features
General Features	• 16 bytes TX/RX FIFO • Support of Hardware Flow Control (CTS/RTS) • 16 bits clock divider • 16C550 Compatible Core • Smart Card Interface
Maximum Operating Frequency	• Baud Rate Clock (3MHz ← 48MHz / 16)

Table 2.17 GPSB Interface

GPSB	Key Features
General Feature	• MSB / LSB Selection mode • Support Variable transfer (1~16bit) • Clock frequency/ Polarity selection mode • Support configurable Frame signal mode
Maximum Operating Frequency	• 60MHz for slave only • 30MHz for master mode

Table 2.18 General DMA Controller

General DMA	Key Features
General Features	• 12-Channel DMA • Dedicated Bus Interface for Various Storage Interface Controllers • Support of Byte/Half-word/Word Transfer • Support of Circular Buffer Interface - Masking of the Source/Destination Address Bits • 1/2/4/8 Burst Transfers • Byte Swapping Function • Support of Single/Continuous/Burst Mode • 8x32bits FIFO Included
DMA Request/Acknowledge	• External DMA Request/Acknowledge • Interfacing the On-chip Storage Controllers
Inter-channel Arbitration	• Configurable Priority for Each Channel • Round-robin Arbitration / Fixed Priority Arbitration

Table 2.19 Vectored Interrupt Controller

VPIC	Key Features
General Features	• 64 Individual Interrupt Sources • FIQ/IRQ Configurable • Priority Reconfigurable for Each Interrupt Sources • Polarity Controllable • Edge/Level Sensitivity Controllable • Dual/Single Edge Controllable when Edge Sensitivity Selected
Vectored Interrupt Handler	• Vector ID Returned for Fast Handling • Vector ID is one of 0 ~ 63 • 64x32 Vector Table Needed for Vector Handler on On-Chip Memory

Table 2.20 Timer

TIMER & WDT	Key Features
Timer Counters	• Four 16-bit timers with PWM output/counters, two 20-bit timers, and one 32-bit timer • External Event Counter • Stop Mode / Free Running Mode • Various Clock Sources (PLL outputs ~ Divided Sub-Clock) • PWM Functions → TREFn, TMREFn
Watchdog Timers	• Watchdog Timer Interrupt / Reset

Table 2.21 ADC

TSADC	Key Features
General Features	• 12bit Resolution • 0 ~ 3.3V Input Range (In case of 3.3V AVDD)
Operating Frequency	• 1MSPS / 5MHz
Touch Screen	• X/Y Position • Up/Down Wake-up

Table 2.22 Real Time Clock

RTC	Key Features
General Features	• Sub Oscillator Included • Clock and Calendar Function (BCD Display) - Sec/Min/Hour/Date/Day-of-Week/Month/Year • Leap Year Generation • Wakeup Signal Generation from the Deep Power-down Mode
Interrupt & Round Reset	• Alarm Interrupt in Normal Operation Mode • Cyclic interrupts 1/256, 1/64, 1/16, 1/4, 1/2, 1 second interrupts • Round-reset function 30-, 40-, 50- second
Wakeup Function	• Dedicated Wake-up Port

3 PIN Description

3.1 NVS2310 Pin Description

Table 3.1 Power/Ground Information

Group	# of Balls	Ball#	MIN(V)	TYP(V)	MAX(V)	Description
VSS_CORE	53	E5, F5, G5, H5, J5, K5, N5, P5, R5, T5, U5, V5, W5, E6, W6, E7, J7, M7, P7, T7, W7, E8, G8, E9, G9, G11, E16, W16, E17, G17, H17, J17, K17, L17, M17, N17, R17, W17, W18, E19, F19, G19, H19, J19, K19, L19, N19, P19, R19, T19, U19, V19, W19	-	-	-	Internal Core Ground
VDD_CORE	18	F15, F16, F17, F18, G18, H18, J18, K18, L18, M18, N18, P18, R18, T18, U18, V18, V17, V16	1.14	1.20	1.26	Digital Internal Core Power
VSS_IO_x (x = A ~ G, MD)	15	G7, H7, N7, R7, U7, U8, E10, G10, G15, U15, G16, U16, T17, U17, E18	-	-	-	I/O Ground
VDD_IO_A	1	K6	1.71 2.38 3.14	1.8 2.5 3.3	1.89 2.62 3.46	GPIOA Group I/O Power
VDD_IO_B	1	F6				GPIOB Group I/O Power
VDD_IO_C	1	F7				GPIOC Group I/O Power
VDD_IO_D	1	J6				GPIOD Group I/O Power
VDD_IO_E	1	L5				GPIOE Group I/O Power
VDD_IO_F	1	H6				GPIOF Group I/O Power
VDD_IO_G	1	L6				GPIOG Group I/O Power
VDD_IO_MD	1	G6				CORE I/O Power
VDDQ	11	N6, P6, R6, T6, U6, V6, V7, V8, V9, V10, V11	1.14 1.425 1.71	1.2 1.5 1.8	1.26 1.575 1.89	DDR/mDDR I/O Power
VDDQ_ZQ	1	W11				DDR2 ZQ Calibration Power
VDDQ_CKE	1	F14				
VSSQ_ZQ	1	P17				SDR/DDR/mDDR/DDR2 I/F Ground
VSSQ_CKE	1	M19				
VDD_OSC	1	M5	1.71	1.8	1.89	Oscillator Power
VSS_OSC	1	V12	-	-	-	Oscillator Ground
VDD12_USB0	1	V14	1.14	1.20	1.26	UTMI0 Digital Core Power
VDD33_USB0	1	F9	3.135	3.3	3.465	UTMI0 Analog Core Power
VSS12_USB0	1	W12	-	-	-	UTMI0 Ground
VSS33_USB0	2	U13, U14,				UTMI0 Ground
VDD12_USB1	1	V15	1.14	1.20	1.;26	UTMI1 Digital Core Power
VDD33_USB1	1	F10	3.135	3.3	3.465	UTMI1 Analog Core Power
VSS12_USB1	1	U12				UTMI1 Ground
VSS33_USB1	2	W14, W15				UTMI1 Ground
VSSAC_USB0	1	R12				UTMI0 Analog Ground
VSSAC_USB1	1	R13				UTMI1 Analog Ground
VDD_HDMI	1	E13	1.14	1.20	1.26	HDMI Core Power
VSS_HDMI	4	E12, F12, G12, G13				HDMI Core Ground
VDDPLL_HDMI	1	F13	1.14	1.20	1.26	HDMI PLL Power 1
VDDOSC_HDMI	1	E11	3.0	3.3	3.6	HDMI Oscillator Power
VSSOSC_HDMI	1	E14				HDMI Oscillator Ground
VDD_ADC	1	F8	3.0	3.3	3.6	ADC Power
VSS_ADC	1	L7	-	-	-	ADC Ground
VDD_DAC	1	F11	3.0	3.3	3.6	DAC Power
VSS_DAC	2	G14, E15	-	-	-	DAC Ground
VDD0_PLL	1	V13	1.14	1.20	1.26	PLL Power
VSS0_PLL	3	W8, W9, W10	-	-	-	PLL Ground
VDD1_PLL	1	W13	1.14	1.20	1.26	PLL Power
VSS1_PLL	3	U9, U10, U11	-	-	-	PLL Ground
VDD_RTC	1	M6	1.5	3.0	3.3	RTC Core & I/O Power

Table 3.2 GPIOA(VDD_IO_A) Group I/O Pin Description

NAME	BALL	I/O[12]	INIT[13]	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
GPIOA[0]	Y17	B	I	GPIO_A[0]	I2C0(SCL0)					
GPIOA[1]	Y14	B	I	GPIO_A[1]	I2C0(SDA0)					
GPIOA[2]	AC18	B	I	GPIO_A[2]	CLK_OUT0	I2C1(SCL0)				
GPIOA[3]	Y15	B	I	GPIO_A[3]	CLK_OUT1	I2C1(SDA0)				
GPIOA[4]	AA16	B	I	GPIO_A[4]		TCO0	I2C1(SCL1)			
GPIOA[5]	AB18	B	I	GPIO_A[5]	IRDI	TCO1	I2C1(SDA1)			
GPIOA[6]	Y16	B	I	GPIO_A[6]	HDMI_CEC	TCO2	I2C1(SCL0)			EDIXA[19]
GPIOA[7]	T15	B	I	GPIO_A[7]	UTM1_DRVVBUS	TCO3	I2C1(SDA0)			EDIXA[20]
GPIOA[8]	AA17	B	I	GPIO_A[8]	I2C0(SCL1)	EDIXA[19]				
GPIOA[9]	AA18	B	I	GPIO_A[9]	I2C0(SDA1)	EDIXA[20]				
GPIOA[10]	P13	B	I	GPIO_A[10]	I2C1(SCL1)	CDATA(1)				
GPIOA[11]	N13	B	I	GPIO_A[11]	I2C1(SDA1)	CBCLK(1)				
GPIOA[12]	R14	B	I	GPIO_A[12]	TCO0	CLRCK(1)				
GPIOA[13]	M13	B	I	GPIO_A[13]	EXTCLKI	SPDIF_TX(0)				
GPIOA[14]	Y18	B	I	GPIO_A[14]	HDMI_HPD	TCO4				
GPIOA[15]	P14	B	I	GPIO_A[15]	UTM0_DRVVBUS	TCO5				
GPIOA[16]	R16	B	I	GPIO_A[16]	PDM_OUT[0]	SPDIF_TX(1)	I2C2(SCL0)			
GPIOA[17]	N14	B	I	GPIO_A[17]	PDM_OUT[1]	CDATA(0)	I2C2(SDA0)			
GPIOA[18]	T16	B	I	GPIO_A[18]	PDM_OUT[2]	CBCLK(0)	I2C2(SCL1)			
GPIOA[19]	R15	B	I	GPIO_A[19]	PDM_OUT[3]	CLRCK(0)	I2C2(SDA1)			

Table 3.3 GPIOB(VDD_IO_B) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
GPIOB[0]	C16	B	I	GPIO_B[0]	EDIXD0		MS_D0(5)			
GPIOB[1]	A22	B	I	GPIO_B[1]	EDIXD1		MS_D1(5)			
GPIOB[2]	D16	B	I	GPIO_B[2]	EDIXD2		MS_D2(5)			
GPIOB[3]	A23	B	I	GPIO_B[3]	EDIXD3		MS_D3(5)			
GPIOB[4]	A4	B	I	GPIO_B[4]	EDIXD4		MS_D4(5)			
GPIOB[5]	C9	B	I	GPIO_B[5]	EDIXD5		MS_D5(5)			
GPIOB[6]	B4	B	I	GPIO_B[6]	EDIXD6		MS_D6(5)			
GPIOB[7]	C7	B	I	GPIO_B[7]	EDIXD7		MS_D7(5)			
GPIOB[8]	B5	B	I	GPIO_B[8]	EDIXD8			SFRM(1)		
GPIOB[9]	C8	B	I	GPIO_B[9]	EDIXD9			SCLK(1)		
GPIOB[10]	A5	B	I	GPIO_B[10]	EDIXD10			SDI(1)		
GPIOB[11]	B21	B	I	GPIO_B[11]	EDIXD11			SDO(1)		
GPIOB[12]	D7	B	I	GPIO_B[12]	EDIXD12		MS_BUS(5)			
GPIOB[13]	B22	B	I	GPIO_B[13]	EDIXD13		MS_CLK(5)			
GPIOB[14]	D8	B	I	GPIO_B[14]	EDIXD14					
GPIOB[15]	B23	B	I	GPIO_B[15]	EDIXD15					
GPIOB[16]	H16	B	I	GPIO_B[16]	EDIWEN0					EDIRDY4
GPIOB[17]	D9	B	I	GPIO_B[17]	EDIWEN1			SFRM(0)		
GPIOB[18]	J16	B	I	GPIO_B[18]	EDIOEN0					EDIRDY5
GPIOB[19]	D10	B	I	GPIO_B[19]	EDIOEN1			SCLK(0)		
GPIOB[20]	C10	B	I	GPIO_B[20]	EDIXA[0]					
GPIOB[21]	D15	B	I	GPIO_B[21]	EDIXA[1]		MS_D4(6)			
GPIOB[22]	J15	B	I	GPIO_B[22]	EDIXA[2]		MS_D5(6)			
GPIOB[23]	J14	B	I	GPIO_B[23]	EDICSN0		MS_D6(6)			EDIRDY2
GPIOB[24]	J13	B	I	GPIO_B[24]	EDICSN1		MS_D7(6)			EDIRDY3
GPIOB[25]	D6	B	I	GPIO_B[25]	EDICSN2		MS_D0(6)			
GPIOB[26]	J12	B	I	GPIO_B[26]	EDICSN3	I2C2(SCL0)	MS_D1(6)			
GPIOB[27]	D5	B	I	GPIO_B[27]	EDICSN4	I2C2(SDA0)	MS_D2(6)			EDIXA[23]
GPIOB[28]	C15	B	I	GPIO_B[28]	EDIRDY0		MS_D3(6)			
GPIOB[29]	C5	B	I	GPIO_B[29]	EDIRDY1		MS_BUS(6)			
GPIOB[30]	D14	B	I	GPIO_B[30]	EDICSN5	I2C2(SCL1)	MS_CLK(6)	SDI(0)		EDIXA[22]
GPIOB[31]	C6	B	I	GPIO_B[31]	EDICSN6	I2C2(SDA1)		SDO(0)		EDIXA[21]

[12] I/O type. B = bi-direction, I = Input, O = Output, A = Analog
[13] INIT column represents the corresponding I/O state while the reset signal is asserted.
OL = output low, OH =output high, O=output unknown, IU = input/pull-up, ID = input /pull-down, I = input floating

Table 3.4 GPIOC(VDD_IO_C) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
GPIOC[0]	H4	B	I	GPIO_C[0]	LXD[0]	L0_LPD[0]	TSD5(3)		L1_LPD[0]	GPIOF[0]
GPIOC[1]	C4	B	I	GPIO_C[1]	LXD[1]	L0_LPD[1]	TSD6(3)		L1_LPD[1]	GPIOF[1]
GPIOC[2]	J2	B	I	GPIO_C[2]	LXD[2]	L0_LPD[2]	TSD7(3)		L1_LPD[2]	GPIOF[2]
GPIOC[3]	A3	B	I	GPIO_C[3]	LXD[3]	L0_LPD[3]			L1_LPD[3]	GPIOF[3]
GPIOC[4]	J1	B	I	GPIO_C[4]	LXD[4]	L0_LPD[4]			L1_LPD[4]	GPIOF[4]
GPIOC[5]	B3	B	I	GPIO_C[5]	LXD[5]	L0_LPD[5]			L1_LPD[5]	GPIOF[5]
GPIOC[6]	H3	B	I	GPIO_C[6]	LXD[6]	L0_LPD[6]			L1_LPD[6]	GPIOF[6]
GPIOC[7]	C3	B	I	GPIO_C[7]	LXD[7]	L0_LPD[7]			L1_LPD[7]	GPIOF[7]
GPIOC[8]	E3	B	I	GPIO_C[8]	LXD[8]	L0_LPD[8]			L1_LPD[8]	GPIOF[8]
GPIOC[9]	A2	B	I	GPIO_C[9]	LXD[9]	L0_LPD[9]			L1_LPD[9]	GPIOF[9]
GPIOC[10]	E4	B	I	GPIO_C[10]	LXD[10]	L0_LPD[10]	SDO(2)		L1_LPD[10]	GPIOF[10]
GPIOC[11]	B2	B	I	GPIO_C[11]	LXD[11]	L0_LPD[11]	SDI(2)		L1_LPD[11]	GPIOF[11]
GPIOC[12]	F1	B	I	GPIO_C[12]	LXD[12]	L0_LPD[12]	SCLK(2)		L1_LPD[12]	GPIOF[12]
GPIOC[13]	A1	B	I	GPIO_C[13]	LXD[13]	L0_LPD[13]	SFRM(2)		L1_LPD[13]	GPIOF[13]
GPIOC[14]	F2	B	I	GPIO_C[14]	LXD[14]	L0_LPD[14]		MS_D7(0)	L1_LPD[14]	GPIOF[14]
GPIOC[15]	C2	B	I	GPIO_C[15]	LXD[15]	L0_LPD[15]		MS_D6(0)	L1_LPD[15]	GPIOF[15]
GPIOC[16]	F3	B	I	GPIO_C[16]	LXD[16]	L0_LPD[16]		MS_D5(0)	L1_LPD[16]	GPIOF[16]
GPIOC[17]	B1	B	I	GPIO_C[17]	LXD[17]	L0_LPD[17]		MS_D4(0)	L1_LPD[17]	GPIOF[17]
GPIOC[18]	D2	B	I	GPIO_C[18]	LXD[18]	L0_LPD[18]		MS_D3(0)	L1_LPD[18]	
GPIOC[19]	G3	B	I	GPIO_C[19]	LXD[19]	L0_LPD[19]		MS_D2(0)	L1_LPD[19]	
GPIOC[20]	C1	B	I	GPIO_C[20]	LXD[20]	L0_LPD[20]		MS_D1(0)	L1_LPD[20]	
GPIOC[21]	H2	B	I	GPIO_C[21]	LXD[21]	L0_LPD[21]		MS_D0(0)	L1_LPD[21]	
GPIOC[22]	D1	B	I	GPIO_C[22]	LXD[22]	L0_LPD[22]		MS_CLK(0)	L1_LPD[22]	
GPIOC[23]	G4	B	I	GPIO_C[23]	LXD[23]	L0_LPD[23]		MS_BUS(0)	L1_LPD[23]	
GPIOC[24]	F4	B	I	GPIO_C[24]	LWEN	L0_LDE	TSD4(3)		L1_LDE	GPIOF[19]
GPIOC[25]	D4	B	I	GPIO_C[25]	LOEN	L0_LCK	TSD3(3)		L1_LCK	GPIOF[18]
GPIOC[26]	G1	B	I	GPIO_C[26]	LXA[0]	L0_LHS	TSD2(3)		L1_LHS	GPIOF[22]
GPIOC[27]	D3	B	I	GPIO_C[27]	LCSN0	L0_LVS	TSD1(3)		L1_LVS	GPIOF[20]
GPIOC[28]	G2	B	I	GPIO_C[28]	LCSN1	SDO(10)	TSVALID(3)			GPIOF[21]
GPIOC[29]	E1	B	I	GPIO_C[29]		SDI(10)	TSCLK(3)			
GPIOC[30]	H1	B	I	GPIO_C[30]	EXTLVS1(0)	SCLK(10)	TSD0(3)			
GPIOC[31]	E2	B	I	GPIO_C[31]	EXTLVS0(0)	SFRM(10)	TSSYNC(3)			

Table 3.5 GPIOD(VDD_IO_D) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
GPIOD[0]	AB19	B	I	GPIO_D[0]	DAI0_BCLK					
GPIOD[1]	AC19	B	I	GPIO_D[1]	DAI0_LRCLK					
GPIOD[2]	AA19	B	I	GPIO_D[2]	DAI0_MCLK					
GPIOD[3]	AC20	B	I	GPIO_D[3]	DAI0_SDO[0]					
GPIOD[4]	Y19	B	I	GPIO_D[4]	DAI0_SDI[0]					
GPIOD[5]	AC21	B	I	GPIO_D[5]	DAI0_SDO[1]	SFRM(11)				
GPIOD[6]	AB22	B	I	GPIO_D[6]	DAI0_SDI[1]	SCLK(11)				
GPIOD[7]	AC22	B	I	GPIO_D[7]	DAI0_SDO[2]	SDI(11)				
GPIOD[8]	AB21	B	I	GPIO_D[8]	DAI0_SDI[2]	SDO(11)				
GPIOD[9]	AC23	B	I	GPIO_D[9]	DAI0_SDO[3]	SFRM(6)	TSD7(2)			
GPIOD[10]	L14	B	I	GPIO_D[10]	DAI0_SDI[3]	SCLK(6)	TSD6(2)			
GPIOD[11]	AB23	B	I	GPIO_D[11]	SPDIF_TX(0)	SDI(6)				
GPIOD[12]	M15	B	I	GPIO_D[12]	SPDIF_RX	SDO(6)	TSSYNC(2)			
GPIOD[13]	P15	B	I	GPIO_D[13]	UTXD(4)	I2C1(SCL1)	TSD5(2)			
GPIOD[14]	K15	B	I	GPIO_D[14]	URXD(4)	I2C1(SDA1)	TSD4(2)			
GPIOD[15]	L13	B	I	GPIO_D[15]	UCTS(4)	SFRM(12)	TSVALID(2)			
GPIOD[16]	L15	B	I	GPIO_D[16]	URTS(4)	SCLK(12)	TSCLK(2)			
GPIOD[17]	K13	B	I	GPIO_D[17]	UTXD(5)	I2C2(SCL0)	TSD3(2)			
GPIOD[18]	N16	B	I	GPIO_D[18]	URXD(5)	I2C2(SDA0)	TSD2(2)	I2C0(SCL1)		
GPIOD[19]	M14	B	I	GPIO_D[19]	UCTS(5)	SDI(12)	TSD1(2)	I2C0(SDA1)		
GPIOD[20]	K16	B	I	GPIO_D[20]	URTS(5)	SDO(12)	TSD0(2)	I2C0(SCL0)		
GPIOD[21]	N15	B	I	GPIO_D[21]	DAI1_BCLK			I2C0(SDA0)		
GPIOD[22]	L16	B	I	GPIO_D[22]	DAI1_LRCLK			I2C1(SCL0)		
GPIOD[23]	P16	B	I	GPIO_D[23]	DAI1_MCLK			I2C1(SDA0)		
GPIOD[24]	M16	B	I		DAI1_SDO[0]					
GPIOD[25]	K14	B	I		DAI1_SDI[0]					

Table 3.6 GPIOE(VDD_IO_E) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
GPIOE[0]	AB7	B	I	GPIO_E[0]	UTXD(0)				SD3_XD[0]	
GPIOE[1]	AB4	B	I	GPIO_E[1]	URXD(0)				SD3_XD[1]	
GPIOE[2]	AC7	B	I	GPIO_E[2]	UCTS(0)	SFRM(5)			SD3_XD[2]	
GPIOE[3]	AC4	B	I	GPIO_E[3]	URTS(0)	SCLK(5)			SD3_XD[3]	
GPIOE[4]	Y8	B	I	GPIO_E[4]	UTXD(1)				SD3_XD[4]	
GPIOE[5]	Y5	B	I	GPIO_E[5]	URXD(1)				SD3_XD[5]	
GPIOE[6]	AC10	B	I	GPIO_E[6]	UCTS(1)	SDI(5)		MS_CLK(4)	SD3_CLK	
GPIOE[7]	AA5	B	I	GPIO_E[7]	URTS(1)	SDO(5)		MS_BUS(4)	SD3_CMD	
GPIOE[8]	AB10	B	I	GPIO_E[8]	UTXD(2)	SFRM(4)	CPD[8]	MS_D0(4)	SD3_XD[6]	
GPIOE[9]	AB5	B	I	GPIO_E[9]	URXD(2)	SCLK(4)	CPD[9]	MS_D1(4)	SD3_XD[7]	
GPIOE[10]	AC9	B	I	GPIO_E[10]	UTXD(3)	SDI(4)	CPD[10]	MS_D2(4)		
GPIOE[11]	AC5	B	I	GPIO_E[11]	URXD(3)	SDO(4)	CPD[11]	MS_D3(4)		
GPIOE[12]	AC8	B	I	GPIO_E[12]	CPD[0]		TSD0(1)	MS_D0(2)		
GPIOE[13]	Y6	B	I	GPIO_E[13]	CPD[1]		TSD1(1)	MS_D1(2)		
GPIOE[14]	AB9	B	I	GPIO_E[14]	CPD[2]		TSD2(1)	MS_D2(2)		
GPIOE[15]	AA6	B	I	GPIO_E[15]	CPD[3]		TSD3(1)	MS_D3(2)		
GPIOE[16]	AA9	B	I	GPIO_E[16]	CPD[4]		TSD4(1)	MS_D4(2)		
GPIOE[17]	AB6	B	I	GPIO_E[17]	CPD[5]		TSD5(1)	MS_D5(2)		
GPIOE[18]	Y9	B	I	GPIO_E[18]	CPD[6]		TSD6(1)	MS_D6(2)		
GPIOE[19]	AC6	B	I	GPIO_E[19]	CPD[7]		TSD7(1)	MS_D7(2)		
GPIOE[20]	AB8	B	I	GPIO_E[20]	CCKI		TSCLK(1)	MS_CLK(2)		
GPIOE[21]	AA7	B	I	GPIO_E[21]	CVS		TSSYNC(1)	MS_BUS(2)		
GPIOE[22]	AA8	B	I	GPIO_E[22]	CHS		TSVALID(1)			
GPIOE[23]	Y7	B	I	GPIO_E[23]	CCKO	CFIELD				
GPIOE[24]	R11	B	I	GPIO_E[24]	SENSOR_PWDN					
GPIOE[25]	M12	B	I	GPIO_E[25]	FL_TRIG					
GPIOE[26]	P12	B	I	GPIO_E[26]	FLASH_TRIG		MS_BUS(7)			
GPIOE[27]	N12	B	I	GPIO_E[27]	PRELIGHT_TRIG		MS_CLK(7)			
GPIOE[28]	M11	B	I	GPIO_E[28]	SHUTTER_TRIG	CFIELD	MS_D0(7)	TS_AIN[0]		
GPIOE[29]	R10	B	I	GPIO_E[29]	SHUTTER_OPEN		MS_D1(7)	TS_AIN[1]		
GPIOE[30]	L11	B	I	GPIO_E[30]	I2C0(SCL1)		MS_D2(7)	TS_AIN[2]		
GPIOE[31]	R9	B	I	GPIO_E[31]	I2C0(SDA1)		MS_D3(7)	TS_AIN[3]		

Table 3.7 GPIOF(VDD_IO_F) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6	FUNC7
GPIOF[0]	U2	B	I	GPIO_F[0]	HPXD[0]		GMAC_RXD[4]	TSD0(0)	L1_LPD[0]	EDIXA[3]	
GPIOF[1]	U1	B	I	GPIO_F[1]	HPXD[1]		GMAC_RXD[5]	TSD1(0)	L1_LPD[1]	EDIXA[4]	
GPIOF[2]	V1	B	I	GPIO_F[2]	HPXD[2]		GMAC_RXD[6]	TSD2(0)	L1_LPD[2]	EDIXA[5]	
GPIOF[3]	J3	B	I	GPIO_F[3]	HPXD[3]		GMAC_RXD[7]	TSD3(0)	L1_LPD[3]	EDIXA[6]	
GPIOF[4]	V2	B	I	GPIO_F[4]	HPXD[4]		GMAC_TXD[4]	TSD4(0)	L1_LPD[4]	EDIXA[7]	
GPIOF[5]	K1	B	I	GPIO_F[5]	HPXD[5]		GMAC_TXD[5]	TSD5(0)	L1_LPD[5]	EDIXA[8]	
GPIOF[6]	W1	B	I	GPIO_F[6]	HPXD[6]		GMAC_TXD[6]	TSD6(0)	L1_LPD[6]	EDIXA[9]	
GPIOF[7]	K2	B	I	GPIO_F[7]	HPXD[7]		GMAC_TXD[7]	TSD7(0)	L1_LPD[7]	EDIXA[10]	
GPIOF[8]	W2	B	I	GPIO_F[8]	HPXD[8]		GMAC_CLK_RX	TSVALID(0)	L1_LPD[8]	EDIXA[11]	
GPIOF[9]	K3	B	I	GPIO_F[9]	HPXD[9]		GMAC_RXDV	TSCLK(0)	L1_LPD[9]	EDIXA[12]	
GPIOF[10]	N3	B	I	GPIO_F[10]	HPXD[10]	SDO(7)	GMAC_RXER	TSSYNC(0)	L1_LPD[10]	EDIXA[13]	
GPIOF[11]	K4	B	I	GPIO_F[11]	HPXD[11]	SDI(7)	GMAC_RXD[0]		L1_LPD[11]	EDIXA[14]	
GPIOF[12]	P3	B	I	GPIO_F[12]	HPXD[12]	SCLK(7)	GMAC_RXD[1]		L1_LPD[12]	EDIXA[15]	
GPIOF[13]	J4	B	I	GPIO_F[13]	HPXD[13]	SFRM(7)	GMAC_RXD[2]		L1_LPD[13]	EDIXA[16]	
GPIOF[14]	R3	B	I	GPIO_F[14]	HPXD[14]	SDO(8)	GMAC_RXD[3]		L1_LPD[14]	EDIXA[17]	EDIXD16
GPIOF[15]	N1	B	I	GPIO_F[15]	HPXD[15]	SDI(8)	GMAC_CLK_TX		L1_LPD[15]	EDIXA[18]	EDIXD17
GPIOF[16]	T3	B	I	GPIO_F[16]	HPXD[16]	SCLK(8)	GMAC_TXEN		L1_LPD[16]	EDIXA[19]	EDIXD18
GPIOF[17]	P1	B	I	GPIO_F[17]	HPXD[17]	SFRM(8)	GMAC_TXER		L1_LPD[17]	EDIXA[20]	EDIXD19
GPIOF[18]	Y1	B	I	GPIO_F[18]	HPRDN		GMAC_TXD[0]	MS_D3(1)	L1_LPD[18]	EDIXA[21]	EDIXD20
GPIOF[19]	R1	B	I	GPIO_F[19]	HPWRN		GMAC_TXD[1]	MS_D2(1)	L1_LPD[19]	EDIXA[22]	EDIXD21
GPIOF[20]	Y2	B	IU	GPIO_F[20]	HPCSN0		GMAC_TXD[2]	MS_D1(1)	L1_LPD[20]		EDIXD22
GPIOF[21]	T1	B	IU	GPIO_F[21]	HPCSN1		GMAC_TXD[3]	MS_D0(1)	L1_LPD[21]		EDIXD23
GPIOF[22]	U3	B	I	GPIO_F[22]	HPXA0		GMAC_COL	MS_BUS(1)	L1_LPD[22]		EDIXD24
GPIOF[23]	N2	B	I	GPIO_F[23]	HPINT0		GMAC_CRS	MS_CLK(1)	L1_LPD[23]		EDIXD25
GPIOF[24]	V3	B	I	GPIO_F[24]	HPINT1	SDO(9)	GMAC_MDC		L1_LDE		EDIXD26
GPIOF[25]	P2	B	I	GPIO_F[25]	HPXA1	SDI(9)	GMAC_MDIO		L1_LCK		EDIXD27
GPIOF[26]	W3	B	I	GPIO_F[26]	EXTLVS1(1)	SCLK(9)	GMAC_CLK_GTX		L1_LHS		EDIXD28
GPIOF[27]	R2	B	I	GPIO_F[27]	EXTLVS0(1)	SFRM(9)			L1_LVS		EDIXD29
GPIOF[28]	Y3	B	IU	GPIO_F[28]	HPCSN2						EDIXD30
GPIOF[29]	T2	B	I	GPIO_F[29]							EDIXD31

Table 3.8 GPIOG(VDD_IO_G) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
GPIOG[0]	AB2	B	I	GPIO_G[0]	SD0_XD[0]		MS_D0(8)			
GPIOG[1]	W4	B	I	GPIO_G[1]	SD0_XD[1]		MS_D1(8)			
GPIOG[2]	AB3	B	I	GPIO_G[2]	SD0_XD[2]		MS_D2(8)			
GPIOG[3]	Y4	B	I	GPIO_G[3]	SD0_XD[3]		MS_D3(8)			
GPIOG[4]	AC1	B	I	GPIO_G[4]	SD0_XD[4]	UCTS(2)	MS_D4(8)	I2C2(SCL0)		
GPIOG[5]	AA4	B	I	GPIO_G[5]	SD0_XD[5]	URTS(2)	MS_D5(8)	I2C2(SDA0)		
GPIOG[6]	AC2	B	I	GPIO_G[6]	SD0_XD[6]	URXD(2)	MS_D6(8)	I2C2(SCL1)		
GPIOG[7]	AA3	B	I	GPIO_G[7]	SD0_XD[7]	UTXD(2)	MS_D7(8)	I2C2(SDA1)		
GPIOG[8]	AC3	B	I	GPIO_G[8]	SD0_CLK		MS_CLK(8)			
GPIOG[9]	AA2	B	I	GPIO_G[9]	SD0_CMD		MS_BUS(8)			
GPIOG[10]	M10	B	I	GPIO_G[10]	SD1_XD[0]					
GPIOG[11]	P9	B	I	GPIO_G[11]	SD1_XD[1]	URXD(3)		I2C1(SCL0)		
GPIOG[12]	L10	B	I	GPIO_G[12]	SD1_XD[2]	UTXD(3)		I2C1(SDA0)		
GPIOG[13]	P10	B	I	GPIO_G[13]	SD1_XD[3]					
GPIOG[14]	K10	B	I	GPIO_G[14]	SD1_CLK					
GPIOG[15]	K11	B	I	GPIO_G[15]	SD1_CMD					
GPIOG[16]	J10	B	I	GPIO_G[16]	SD2_XD[0]		MS_D0(9)			
GPIOG[17]	J11	B	I	GPIO_G[17]	SD2_XD[1]	URXD(4)	MS_D1(9)	I2C1(SCL1)		
GPIOG[18]	P11	B	I	GPIO_G[18]	SD2_XD[2]	UTXD(4)	MS_D2(9)	I2C1(SDA1)		
GPIOG[19]	L12	B	I	GPIO_G[19]	SD2_XD[3]		MS_D3(9)			
GPIOG[20]	N11	B	I	GPIO_G[20]	SD2_CLK		MS_CLK(9)			
GPIOG[21]	K12	B	I	GPIO_G[21]	SD2_CMD		MS_BUS(9)			
GPIOG[22]	M9	B	I	GPIO_G[22]	TS_AIN[4]	UCTS(0)	SDO(3)			
GPIOG[23]	L9	B	I	GPIO_G[23]	TS_AIN[5]	URTS(0)	SDI(3)			
GPIOG[24]	N10	B	I	GPIO_G[24]	TS_AIN[10]	URXD(0)	SCLK(3)			
GPIOG[25]	AA1	B	I	GPIO_G[25]	TS_AIN[11]	UTXD(0)	SFRM(3)			
GPIOG[26]	N9	B	I	GPIO_G[26]	TS_AIN[12]	UCTS(1)	SD0_BUS_POW	I2C2(SCL0)		
GPIOG[27]	U4	B	I	GPIO_G[27]	TS_AIN[13]	URTS(1)	SD1_BUS_POW	I2C2(SDA0)		
GPIOG[28]	AB1	B	I	GPIO_G[28]	TS_AIN[14]	URXD(1)	SD2_BUS_POW	I2C2(SCL1)		
GPIOG[29]	V4	B	I	GPIO_G[29]	TS_AIN[15]	UTXD(1)	SD3_BUS_POW	I2C2(SDA1)		

Table 3.9 ADC (VDD_ADC) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
AIN[0]	N4	A		AIN[0]						
AIN[1]	R4	A		AIN[1]						
AIN[2]	P4	A		AIN[2]						
AIN[3]	T4	A		AIN[3]						
AGND_ADC	K7	A		AGND_ADC						

Table 3.10 ETC (VDD_IO_MD) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
NBPEN	J9	I		Normal Bypass En						
MBPEN	K9	I		MIPI Bypass En						
BM[0]	C18	I		BM[0]						
BM[1]	D17	I		BM[1]						
BM[2]	B20	I		BM[2]						
BM[3]	C17	I		BM[3]						
TDO	B16	O		TDO						
TEST	A20	I		TEST						
NSYSRST	B19	O		RSTN						
NTRST	A19	I		NTRST						
TMS	B18	I		TMS						
TCK	B17	I		TCK						
TDI	A17	I		TDI						
RTCK	A16	O		RTCK						
NPOR	A18	I		NPOR						

Table 3.11 OSC (VDD_OSC) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
XI	Y10	I		XI						
XO	AA10	O		XO						

Table 3.12 USB0 (VDD33_USB0) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
USB0_VBUS	AB14	A		USB0_VBUS						
USB0_DP	AC14	A		USB0_DP						
USB0_DM	AC15	A		USB0_DM						
USB0_TXRTUNE	AA14	A		USB0_TXRTUNE						
USB0_ID	AB15	A		USB0_ID						

Table 3.13 USB1 (VDD33_USB1) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0	FUNC1	FUNC2	FUNC3	FUNC4	FUNC5	FUNC6
USB1_VBUS	AA15	A		USB1_VBUS						
USB1_DP	AC16	A		USB1_DP						
USB1_DM	AC17	A		USB1_DM						
USB1_TXRTUNE	AB16	A		USB1_TXRTUNE						
USB1_ID	AB17	A		USB1_ID						

Table 3.14 RTC(VDD_RTC) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0						
XTI	C19	I		RTC_XTI						
XTO	D19	O		RTC_XTO						
PMWKUP	A21	O	OH	RTC_PMWKUP						
RTC_RSTN	D18	I	IL	RTC_RSTN						

Table 3.15 HDMI OSC (VDDOSC_HDMI) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0						
HDMI_XI	A10	I		HDMI_XI						
HDMI_XO	B10	O		HDMI_XO						

Table 3.16 HDMI(VDD_HDMI) Group I/O Pin Description

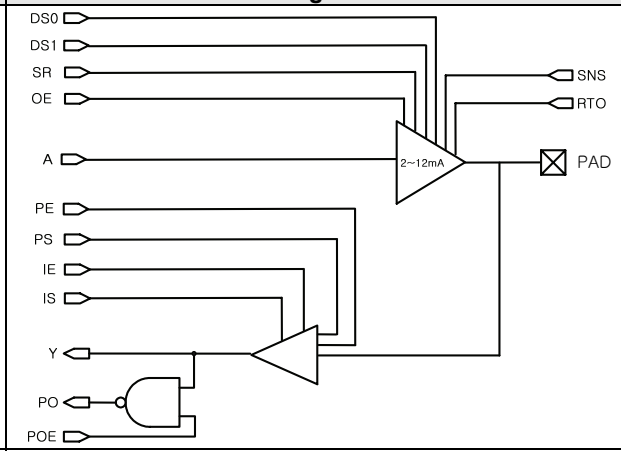
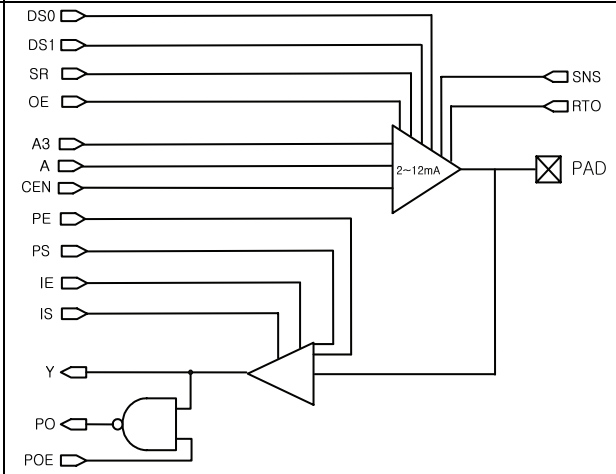
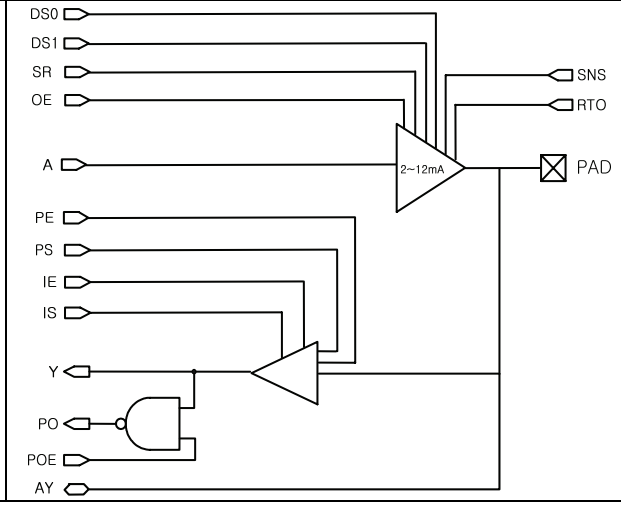
NAME	BALL	I/O	INIT	FUNC0						
HDMI_TXCP	A9	A		HDMI_TXCP						
HDMI_TXCN	B9	A		HDMI_TXCN						
HDMI_TX0P	A8	A		HDMI_TX0P						
HDMI_TX0N	B8	A		HDMI_TX0N						
HDMI_TX1P	A7	A		HDMI_TX1P						
HDMI_TX1N	B7	A		HDMI_TX1N						
HDMI_TX2P	A6	A		HDMI_TX2P						
HDMI_TX2N	B6	A		HDMI_TX2N						
HDMI_REXT	C14	A		HDMI_REXT						

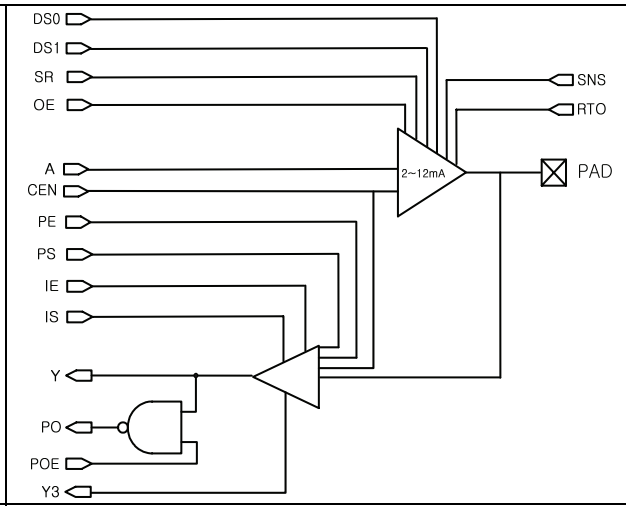
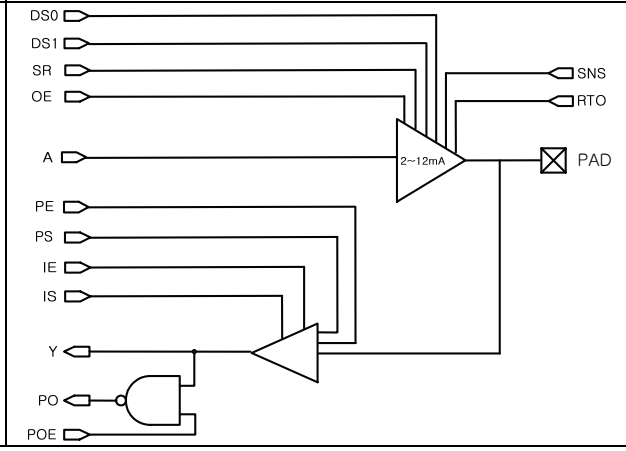
Table 3.17 DAC(VDD_DAC) Group I/O Pin Description

NAME	BALL	I/O	INIT	FUNC0						
DAC_DOUT	A15	A		DAC_DOUT						
DAC_COMP	B14	A		DAC_COMP						
DAC_IREF	B15	A		DAC_IREF						
DAC_VREF	A14	A		DAC_VREF						

3.2 NVS2310 I/O Type

Table 3.18 NVS2310 I/O Type

TYPE	Diagram	DESCRIPTION	PAD Name
A		Wide-range I/O supply, programmable bi-direction I/O with cmos / schmitt trigger input, input disable, pull-up/down and 4-step (7mA~28mA) strength output	GPIOA Group GPIOB Group GPIOD Group GPIOE[27:00] GPIOG[21:00]
B		Wide-range I/O supply, programmable bi-direction I/O with cmos / schmitt trigger input, input disable, pull-up/down and 4-step (2mA~12mA) strength output *Bypass output mode can be supported	GPIOC Group
C		Wide-range I/O supply, programmable bi-direction I/O with cmos / schmitt trigger input, input disable, pull-up/down and 4-step (2mA~12mA) strength output *analog mode can be supported *analog path has 50ohm resistor	GPIOE[31:28] GPIOG[29:22]

D		Wide-range I/O supply, programmable bi-direction I/O with cmos / schmitt trigger input, input disable, pull-up/down and 4-step (2mA~12mA) strength output *Bypass input mode can be supported	GPIOF Group
E		Wide-range I/O supply, programmable bi-direction I/O with cmos / schmitt trigger input, input disable, pull-up/down and 4-step (2mA~12mA) strength output	NSYSRST TDO RTCK MBPEN BM[0]~BM[2] TEST NPOR NTRST TCK TDI TMS

4 Package Information

4.1 Dimension

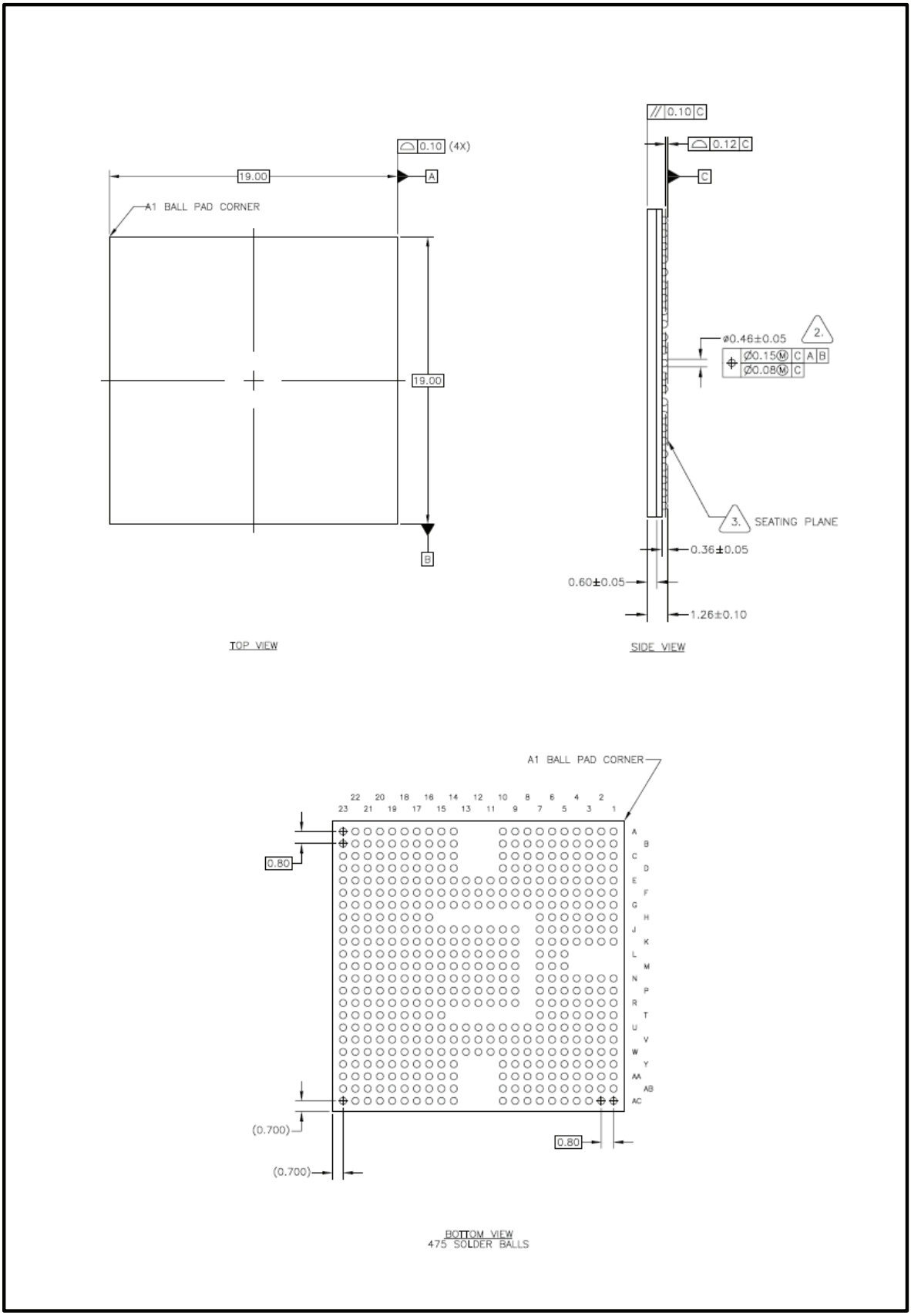


Figure 4.1 NVS2310 Package Dimension

4.2 Ball Map

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
A	GPIOC13	GPIOC9	GPIOC3	GPIOB4	GPIOB10	HDMI_TX2P	HDMI_TX1P	HDMI_TX0P	HDMI_TXCP	HDMI_XI				DAC_VREF	DAC_DOUT	RTCK	TDI	NPOR	NTRST	TEST	PMWKUP	GPIOB1	GPIOB3
B	GPIOC17	GPIOC11	GPIOC5	GPIOB6	GPIOB8	HDMI_TX2N	HDMI_TX1N	HDMI_TX0N	HDMI_TXCN	HDMI_XO				DAC_COMP	DAC_IREF	TDO	TCK	TMS	NSYSRST	BM2	GPIOB11	GPIOB13	GPIOB15
C	GPIOC20	GPIOC15	GPIOC7	GPIOC1	GPIOB29	GPIOB31	GPIOB7	GPIOB9	GPIOB5	GPIOB20				HDMI_REXT	GPIOB28	GPIOB0	BM3	BM0	XTI	DQM0	DQSB1	DQS1	DQSB0
D	GPIOC22	GPIOC18	GPIOC27	GPIOC28	GPIOB27	GPIOB25	GPIOB12	GPIOB14	GPIOB17	GPIOB19				GPIOB30	GPIOB21	GPIOB2	BM1	RTC_RSTN	XTO	XD14	XD12	DQS0	XD15
E	GPIOC29	GPIOC31	GPIOC8	GPIOC10	VSS_CORE	VSS_CORE	VSS_CORE	VSS_CORE	VSS_CORE	VSS_IO_C	VDDOSC_HDI	VSSA_HDMI	VDDA_HDMI	VSSOSC_HDMI	VSSA_DAC	VSS_CORE	VSS_CORE	VSS_IO_MEI	VSS_CORE	XD8	XD10	XD9	XD13
F	GPIOC12	GPIOC14	GPIOC16	GPIOC24	VSS_CORE	VDD_IO_B	VDD_IO_C	VDD_ADC	VDD33_USB	VDD33_USB	VDDDAC	VSSA_HDMI	VDDPLL_HDMI	VDDQ_CKE	VDD_CORE	VDD_CORE	VDD_CORE	VDD_CORE	VSS_CORE	XD1	XD3	XD11	DQM1
G	GPIOC26	GPIOC28	GPIOC19	GPIOC23	VSS_CORE	VDD_IO_M	VSS_IO_F	VSS_CORE	VSS_CORE	VSS_IO_C	VSS_CORE	VSSA_HDMI	VSSA_HDMI	VSSD_DAC	VSS_IO_B	VSS_IO_B	VSS_CORE	VDD_CORE	VSS_CORE	XD7	XD5	XD0	XD2
H	GPIOC30	GPIOC21	GPIOC6	GPIOC0	VSS_CORE	VDD_IO_F	VSS_IO_F									GPIOB16	VSS_CORE	VDD_CORE	VSS_CORE	CASN	WEN	XD6	XD4
J	GPIOC4	GPIOC2	GPIOF3	GPIOF13	VSS_CORE	VDD_IO_D	VSS_CORE		NBPEN	GPIOG16	GPIOG17	GPIOB26	GPIOB24	GPIOB23	GPIOB22	GPIOB18	VSS_CORE	VDD_CORE	VSS_CORE	RASN	XA13	XA0	XA2
K	GPIOF5	GPIOF7	GPIOF9	GPIOF11	VSS_CORE	VDD_IO_A	AGND_ADC		MBPEN	GPIOG14	GPIOG15	GPIOG21	GPIOD17	GPIOD25	GPIOD14	GPIOD20	VSS_CORE	VDD_CORE	VSS_CORE	XA3	XA5	XA7	XA9
L			VDD_IO_E	VDD_IO_G	VSS_ADC				GPIOG23	GPIOG12	GPIOE30	GPIOG19	GPIOD15	GPIOD10	GPIOD16	GPIOD22	VSS_CORE	VDD_CORE	VSS_CORE	VREF1	XA14	XA10	XA4
M			VDD_OSC	VDD_RTC	VSS_CORE				GPIOG22	GPIOG10	GPIOE28	GPIOE25	GPIOA13	GPIOD19	GPIOD12	GPIOD24	VSS_CORE	VDD_CORE	VSSQ_CKE	VREF0	XA6	XA8	CLK
N	GPIOF15	GPIOF23	GPIOF10	AIN0	VSS_CORE	VDDQ	VSS_IO_G		GPIOG26	GPIOG24	GPIOG20	GPIOE27	GPIOA11	GPIOA17	GPIOD21	GPIOD18	VSS_CORE	VDD_CORE	VSS_CORE	XA12	XA1	XA11	CLKB
P	GPIOF17	GPIOF25	GPIOF12	AIN2	VSS_CORE	VDDQ	VSS_CORE		GPIOG11	GPIOG13	GPIOG18	GPIOE26	GPIOA10	GPIOA15	GPIOD13	GPIOD23	VSSQ_ZQ	VDD_CORE	VSS_CORE	BA1	CKE	GATEO	GATEI
R	GPIOF19	GPIOF27	GPIOF14	AIN1	VSS_CORE	VDDQ	VSS_IO_G		GPIOE31	GPIOE29	GPIOE24	VSSAC_USB	VSSAC_USB	GPIOA12	GPIOA19	GPIOA16	VSS_CORE	VDD_CORE	VSS_CORE	CSN0	ODT0	ODT1	BA0
T	GPIOF21	GPIOF29	GPIOF16	AIN3	VSS_CORE	VDDQ	VSS_CORE								GPIOA7	GPIOA18	VSS_IO_D	VDD_CORE	VSS_CORE	ZQ	CSN1	XD23	BA2
U	GPIOF1	GPIOF0	GPIOF22	GPIOG27	VSS_CORE	VDDQ	VSS_IO_E	VSS_IO_E	VSS1_PLL	VSS1_PLL	VSS1_PLL	VSS12_USB	VSS33_USB	VSS33_USB0	VSS_IO_A	VSS_IO_A	VSS_IO_D	VDD_CORE	VSS_CORE	XD20	XD18	XD17	XD21
V	GPIOF2	GPIOF4	GPIOF24	GPIOG29	VSS_CORE	VDDQ	VDDQ	VDDQ	VDDQ	VDDQ	VDDQ	VSS_OSC	VDD0_PLL	VDD12_USB0	VDD12_USB	VDD_CORE	VDD_CORE	VDD_CORE	VSS_CORE	XD22	XD27	XD24	XD19
W	GPIOF6	GPIOF8	GPIOF26	GPIOG31	VSS_CORE	VSS_CORE	VSS_CORE	VSS0_PLL	VSS0_PLL	VSS0_PLL	VDDQ_ZQ	VSS12_USB	VDD1_PLL	VSS33_USB1	VSS33_USB	VSS_CORE	VSS_CORE	VSS_CORE	VSS_CORE	XD29	XD25	XD30	XD26
Y	GPIOF18	GPIOF20	GPIOF28	GPIOG3	GPIOE5	GPIOE13	GPIOE23	GPIOE4	GPIOE18	XI				GPIOA1	GPIOA3	GPIOA6	GPIOA0	GPIOA14	GPIOD4	XD31	XD16	XD28	DQM2
AA	GPIOG25	GPIOG9	GPIOG7	GPIOC5	GPIOE7	GPIOE15	GPIOE21	GPIOE22	GPIOE16	XO				USB0_TXRTUN	USB1_VBUS	GPIOA4	GPIOA8	GPIOA9	GPIOD2	DQM3	DQSB2	DQS3	DQSB3
AB	GPIOG28	GPIOG0	GPIOG2	GPIOE1	GPIOE9	GPIOE17	GPIOE0	GPIOE20	GPIOE14	GPIOE8				USB0_VBUS	USB0_ID	USB1_TXRTUN	USB1_ID	GPIOA5	GPIOD0	DQS2	GPIOD8	GPIOD6	GPIOD11
AC	GPIOG4	GPIOG6	GPIOG8	GPIOE3	GPIOE11	GPIOE19	GPIOE2	GPIOE12	GPIOE10	GPIOE6				USB0_DP	USB0_DM	USB1_DP	USB1_DM	GPIOA2	GPIOD1	GPIOD3	GPIOD5	GPIOD7	GPIOD9

Figure 4.2 NVS2310 Ball Map

5 Electrical Specification

5.1 Absolute Maximum Ratings

Parameter	Symbol	Rating	Unit
DC Supply Voltage for I/O (Various I/O power excepts for analog – ADC, DAC, PLL, USB)	V_{DDIO}	4.6	V
DC Supply Voltage for Internal Digital Logic (VDD_CORE)	V_{DDI}	1.8	V
DC Supply Voltage for Analog Part of ADC (VDD_ADC)	V_{DDADC}	4.6	V
DC Supply Voltage for PLL (VDD_PLL)	V_{DDPLL}	1.8	V
DC Supply Voltage for USB2.0 (VDD33_USB0/1)	V_{DDUSB}	4.6	V
DC Supply Voltage for RTC (VDD_RTC)	V_{DDRTC}	4.6	V
Digital Input Voltage for Input Buffer	V_{IN}	4.6	V
Digital Input Voltage for OTG_VBUS	V_{OTG_VBUS}	6.0	V
Digital Output Voltage for Output Buffer	V_{OUT}	4.6	V
In/Out Current for Digital I/O	$I_{I/O}$	±20	mA
Analog Input Voltage for ADC	V_{IN_ADC}	0 ~ V_{DDADC}	V
Storage Temperature	T_{STG}	-55 to 150	°C

Note:
Absolute maximum ratings specify the values beyond which the device may be damaged permanently. Exposure to absolute maximum rating conditions for extended periods may affect reliability. Each condition value is applied with the other values kept within the following operating conditions and functional operation under any of these conditions is not implied.

(1) All voltages are measured with respect to VSS unless otherwise specified.
(2) V_{DDI} must always be less than V_{DDIO}
(3) The voltage difference between analog and digital grounds must always be within 0.3V.

5.2 Recommended Operating Conditions

Table 5.1 Recommended Operating Conditions[14]

Parameter	Symbol	MIN	TYP	MAX	Unit
Output Load Resistance for DAC [±1% tolerance]	R_{LOAD}	-	37.5	-	Ω
Core (VDD_CORE) and PLL (VDD_PLL) and USB(VDD12_USB) supply Voltage	V_{DDI} V_{DDPLL} $V_{DDUSB12}$	0.95	1.0	1.05	V
		1.05	1.1	1.15	
		1.14	1.2	1.26	
		1.23	1.3	1.37	
		1.33	1.4	1.47	
		1.43	1.5	1.57	
Operating Ambient Temperature [Extended]	T_{OPER}	-30		85	°C
Operating Ambient Temperature [Industrial]	T_{OPER}	-40	-	85	°C

14 The recommended operating conditions for power/ground are described on the Power/Ground Information in the Pin Descriptions.

5.3 Recommended Operating Frequency

Table 5.2 Recommended Operating Frequency

Parameter	Condition	Symbol	MIN	TYP	MAX	Unit
XIN Oscillator	@ 1.0V(TYP)	F_{XIN}	12	12	12	MHz
	@ 1.1V(TYP)					
	@ 1.2V(TYP)					
	@ 1.3V(TYP)					
	@ 1.4V(TYP)					
	@ 1.5V(TYP)					
XTIN Oscillator	@ 1.2V(TYP)	F_{XTIN}	32.768		32.768*128	KHz
PLL0/1/2 VCO Range	@ 1.0V(TYP)	F_{PLL012_VCO}	1000		1200	MHz
	@ 1.1V(TYP)		1000		1600	
	@ 1.2V(TYP)		**1000**		**2000**	
	@ 1.3V(TYP)		1000		2400	
	@ 1.4V(TYP)		1000		2800	
	@ 1.5V(TYP)		1000		3200	
PLL3/4/5 VCO Range	@ 1.0V(TYP)	F_{PLL345_VCO}	330		396	MHz
	@ 1.1V(TYP)		330		528	
	@ 1.2V(TYP)		**330**		**660**	
	@ 1.3V(TYP)		330		792	
	@ 1.4V(TYP)		330		924	
	@ 1.5V(TYP)		330		1056	
Input Clock of Bus Clock Generator (1 ~ 10)[15]	@ 1.0V(TYP)	$F_{BCLKGEN}$			500	MHz
	@ 1.1V(TYP)				750	
	@ 1.2V(TYP)				**1000**	
	@ 1.3V(TYP)				1250	
	@ 1.4V(TYP)				1500	
	@ 1.5V(TYP)				1750	
Input Clock of CPU Clock Generator[16]	@ 1.0V(TYP)	F_{CPUGEN}			500	MHz
	@ 1.1V(TYP)				750	
	@ 1.2V(TYP)				**1000**	
	@ 1.3V(TYP)				1250	
	@ 1.4V(TYP)				1500	
	@ 1.5V(TYP)				1750	
Input Clock of PLL Divider	@ 1.0V(TYP)	$F_{PLLDIVIN}$			500	MHz
	@ 1.1V(TYP)				750	
	@ 1.2V(TYP)				**1000**	
	@ 1.3V(TYP)				1250	
	@ 1.4V(TYP)				1500	
	@ 1.5V(TYP)				1750	
Input Clock of I/O Clock Generator[17]	@ 1.0V(TYP)	$F_{IOCLKGEN}$			500	MHz
	@ 1.1V(TYP)				750	
	@ 1.2V(TYP)				**1000**	
	@ 1.3V(TYP)				1250	
	@ 1.4V(TYP)				1500	
	@ 1.5V(TYP)				1750	
ARM1176JZF-S Core Clock	@ 1.0V(TYP)	F_{CPU}			300	MHz
	@ 1.1V(TYP)				450	
	@ 1.2V(TYP)				**600**	
	@ 1.3V(TYP)				750	
	@ 1.4V(TYP)				900	
	@ 1.5V(TYP)				1050	
Bus Clock of DDI Bus	@ 1.0V(TYP)	F_{BUS_DDI}			150	MHz
	@ 1.1V(TYP)				225	
	@ 1.2V(TYP)				**300**	
	@ 1.3V(TYP)				375	
	@ 1.4V(TYP)				450	
	@ 1.5V(TYP)				525	
Bus Clock of Display Sub Bus	@ 1.0V(TYP)				120	MHz
	@ 1.1V(TYP)				180	
	@ 1.2V(TYP)				**240**	
	@ 1.3V(TYP)				300	
	@ 1.4V(TYP)				360	
	@ 1.5V(TYP)				420	
Bus Clock of Graphic Bus	@ 1.0V(TYP)	F_{BUS_GRP}			120	MHz

15 The related clocks are F_{BUS_DDI}, F_{BUS_GRP}, F_{BUS_MEM}, F_{BUS_VBUS}, F_{BUS_VCODEC},F_{BUS_SMU}, F_{BUS_IOB}.
16 The related clock is F_{CPU}.
17 The prefix of related clocks is F_{IO} .

	@ 1.1V(TYP)				140	
	@ 1.2V(TYP)				**180**	
	@ 1.3V(TYP)				220	
	@ 1.4V(TYP)				260	
	@ 1.5V(TYP)				300	
Bus Clock of I/O Bus	@ 1.0V(TYP)	F_{BUS_IOB}			100	MHz
	@ 1.1V(TYP)				150	
	@ 1.2V(TYP)				**200**	
	@ 1.3V(TYP)				250	
	@ 1.4V(TYP)				300	
	@ 1.5V(TYP)				350	
Bus Clock of High Speed I/O Bus	@ 1.0V(TYP)	F_{BUS_HSIOB}			120	MHz
	@ 1.1V(TYP)				180	
	@ 1.2V(TYP)				**240**	
	@ 1.3V(TYP)				300	
	@ 1.4V(TYP)				360	
	@ 1.5V(TYP)				420	
Bus Clock of SMU Controller	@ 1.0V(TYP)	F_{BUS_SMU}			120	MHz
	@ 1.1V(TYP)				140	
	@ 1.2V(TYP)				**180**	
	@ 1.3V(TYP)				220	
	@ 1.4V(TYP)				260	
	@ 1.5V(TYP)				300	
Bus Clock of Video Bus	@ 1.0V(TYP)	F_{BUS_VBUS}			146	MHz
	@ 1.1V(TYP)				218	
	@ 1.2V(TYP)				**290**	
	@ 1.3V(TYP)				362	
	@ 1.4V(TYP)				434	
	@ 1.5V(TYP)				506	
Bus Clock of Camera Bus	@ 1.0V(TYP)	F_{BUS_CAMBUS}			146	MHz
	@ 1.1V(TYP)				218	
	@ 1.2V(TYP)				**290**	
	@ 1.3V(TYP)				362	
	@ 1.4V(TYP)				434	
	@ 1.5V(TYP)				506	
Core Clock of Video Codec	@ 1.0V(TYP)	F_{BUS_VCODEC}			136	MHz
	@ 1.1V(TYP)				203	
	@ 1.2V(TYP)				**270**	
	@ 1.3V(TYP)				337	
	@ 1.4V(TYP)				404	
	@ 1.5V(TYP)				471	
Bus Clock of Memory Interface (LPDDR2/DDR2/DDR3)	@ 1.0V(TYP)	F_{BUS_MEM}			140	MHz
	@ 1.1V(TYP)				220	
	@ 1.2V(TYP)				**300**	
	@ 1.3V(TYP)				380	
	@ 1.4V(TYP)				460	
	@ 1.5V(TYP)				540	
Bus Clock of Memory Interface (LPDDR)	@ 1.0V(TYP)	F_{BUS_MEM}			100	MHz
	@ 1.1V(TYP)				150	
	@ 1.2V(TYP)				**200**	
	@ 1.3V(TYP)				200	
	@ 1.4V(TYP)				200	
	@ 1.5V(TYP)				200	
Operating Clock of Camera Interface 0[18]	@ 1.0V(TYP)	F_{IO_CIF}			100	MHz
	@ 1.1V(TYP)				140	
	@ 1.2V(TYP)				**180**	
	@ 1.3V(TYP)				220	
	@ 1.4V(TYP)				260	
	@ 1.5V(TYP)				300	
Operating Clock of EHI[18]	@ 1.0V(TYP)	F_{IO_EHI}			100	MHz
	@ 1.1V(TYP)				140	
	@ 1.2V(TYP)				**180**	
	@ 1.3V(TYP)				220	
	@ 1.4V(TYP)				260	
	@ 1.5V(TYP)				300	
Operating Clock of GPSB Controller[18]	@ 1.0V(TYP)	F_{IO_GPSB}			80	MHz
	@ 1.1V(TYP)				120	
	@ 1.2V(TYP)				**160**	
	@ 1.3V(TYP)				200	

18 The operating frequencies of external interface are not same as this. More detailed information is described on the timing characteristics of I/O interface. Refer to the corresponding timing information.

	@ 1.4V(TYP)				240	
	@ 1.5V(TYP)				280	
Operating Clock of Memory Stick Controller[18]	@ 1.0V(TYP)	F_{IO_MSTICK}			70	MHz
	@ 1.1V(TYP)				105	
	@ 1.2V(TYP)				**140**	
	@ 1.3V(TYP)				175	
	@ 1.4V(TYP)				210	
	@ 1.5V(TYP)				245	
Operating Clock of SD/MMC Controller[18]	@ 1.0V(TYP)	F_{IO_SDMMC}			100	MHz
	@ 1.1V(TYP)				150	
	@ 1.2V(TYP)				**200**	
	@ 1.3V(TYP)				250	
	@ 1.4V(TYP)				300	
	@ 1.5V(TYP)				350	
Operating Clock of UART Controller[18]	@ 1.0V(TYP)	F_{IO_UART}			100	MHz
	@ 1.1V(TYP)				140	
	@ 1.2V(TYP)				**180**	
	@ 1.3V(TYP)				220	
	@ 1.4V(TYP)				260	
	@ 1.5V(TYP)				300	
Operating Clock of LCD Controller[18]	@ 1.0V(TYP)	F_{IO_LCD}			100	MHz
	@ 1.1V(TYP)				140	
	@ 1.2V(TYP)				**180**	
	@ 1.3V(TYP)				220	
	@ 1.4V(TYP)				260	
	@ 1.5V(TYP)				300	
Operating Clock of PMU	@ 1.0V(TYP)	F_{IO_PMU}	12	12	12	MHz
	@ 1.1V(TYP)					
	@ 1.2V(TYP)					
	@ 1.3V(TYP)					
	@ 1.4V(TYP)					
	@ 1.5V(TYP)					
Operating Clock of Timer	@ 1.0V(TYP)	F_{IO_TIMER}			100	MHz
	@ 1.1V(TYP)				140	
	@ 1.2V(TYP)				**180**	
	@ 1.3V(TYP)				220	
	@ 1.4V(TYP)				260	
	@ 1.5V(TYP)				300	
Operating Clock of TSIF(Not GPSB)	@ 1.0V(TYP)	F_{IO_TSIF}			70	MHz
	@ 1.1V(TYP)				105	
	@ 1.2V(TYP)				**140**	
	@ 1.3V(TYP)				175	
	@ 1.4V(TYP)				210	
	@ 1.5V(TYP)				245	
Operating Clock of SPDIF Transmitter/Receiver[18]	@ 1.0V(TYP)	F_{IO_SPDIF}			70	MHz
	@ 1.1V(TYP)				105	
	@ 1.2V(TYP)				**140**	
	@ 1.3V(TYP)				175	
	@ 1.4V(TYP)				210	
	@ 1.5V(TYP)				245	
Operating Clock of Audio (ADMA/DAI/CDIF) [18]	@ 1.0V(TYP)	F_{IO_AUDIO}			70	MHz
	@ 1.1V(TYP)				105	
	@ 1.2V(TYP)				**140**	
	@ 1.3V(TYP)				175	
	@ 1.4V(TYP)				210	
	@ 1.5V(TYP)				245	
Operating Clock of I2C[18]	@ 1.0V(TYP)	F_{IO_I2C}			100	MHz
	@ 1.1V(TYP)				140	
	@ 1.2V(TYP)				**180**	
	@ 1.3V(TYP)				220	
	@ 1.4V(TYP)				260	
	@ 1.5V(TYP)				300	

➔The maximum operating frequency can be changed without any notice until approved for mass production.

5.4 Electrical Characteristics for Power Supply

Table 5.3 Peak Power Consumption

Parameter	Power	Condition	MIN	TYP	MAX	Unit
Internal Core Power	VDD_CORE	@ 1.2V			1500	mA
GPIO Power	VDD_IO_n, (n=A,B,C,D,E,F,G,MD)	@ 1.8V @ 2.7V @ 3.3V			100	mA
Memory I/O Power	VDDQ, VDDQ_ZQ VDDQ_CKE	@ 1.2V @ 1.5V @ 1.8V			100	mA
Oscillator Power (XI/XO)	VDD_OSC	@ 1.8V			50	mA
USB 1.2 Power of nanoPHY	VDD12_USB0 VDD12_USB1	@ 1.2V			10	mA
USB 3.3 Power of nanoPHY	VDD33_USB0 VDD33_USB1	@ 3.3V			100	mA
RTC Power	VDD_RTC	@ 2.7V			1	mA

➔The rests of the power which are not described in the above table are shown in the corresponding sub-section.

➔***The value in the above table does not mean the average power***. Refer to this at the designing of the power circuit.

5.5 Electrical Characteristics for General I/O

Table 5.4 DC Electrical Specification for General I/O

Parameter	Symbol	Test Condition	MIN	TYP	MAX	Unit
High Level Input Voltage	V_{IH}		$0.7V_{DDIO}$	-	V_{DDIO}+0.3	V
Low Level Input Voltage	V_{IL}		-0.3	-	$0.3V_{DDIO}$	V
Hysteresis Voltage	△V		$0.1V_{DDIO}$	-		V
High Level Input Current	I_{IH}	VIN = VDDIO, pull-down disabled	-10	-	10	μA
		VIN = VDDIO, pull-down enabled(3.3V)	70	-	130	μA
		VIN = VDDIO, pull-down enabled(2.5V)	40	-	80	μA
		VIN = VDDIO, pull-down enabled(1.8V)	20	-	40	μA
Low Level Input Current	I_{IL}	VIN = VSSIO, pull-up disabled	-10	-	10	μA
		VIN = VSSIO, pull-up enabled(3.3V)	70	-	130	μA
		VIN = VSSIO, pull-up enabled(2.5V)	40	-	80	μA
		VIN = VSSIO, pull-up enabled(1.8V)	20	-	40	μA
High Level Output Voltage	V_{OH}	IOH = -100uA	V_{DDIO}–0.2	-		V
Low Level Output Voltage	V_{OL}	IOL = 100uA			0.2	V
Tri-state Output Leakage Current	I_{OZ}	VOUT = VSSIO or VDDIO	-10		10	μA
Input capacitance	C_{IN}	Any input and Bidirectional buffers			5	pF
Output capacitance	C_{OUT}	Any output buffer			5	pF
XI/XO Frequency	F_{OSC1}		-	12	-	MHz
XTIN/XTOUT Frequency	F_{OSC2}	Normal High Drive = Normal * 128	-	32.768 4194.304	-	kHz

Ta = 25oC, VSS = 0.0V unless otherwise specified.

Note:

(1) 12MHz is recommended for XI/XO frequency.
(2) PLL Output Frequencies are determined by XI/XO frequency.

5.6 Electrical Characteristics for PLL

Table 5.5 DC Electrical Characteristics for PLL0/1/2

Parameter	Symbol	Test Condition	MIN	TYP	MAX	Unit
Power Down Current	I_{PD}	V_{DDPLL} = 1.2V			120	μA
Power Consumption	P_{DD}	V_{DDPLL} = 1.2V			2.4	mW

Ta = 25oC unless otherwise specified.

Table 5.6 AC Electrical Characteristics for PLL0/1/2

Parameter	Symbol	Test Condition	MIN	TYP	MAX	Unit
Input Frequency	Fin	V_{DDPLL} = 1.2V		12		MHz
VCO output frequency	Fvco	V_{DDPLL} = 1.2V	1000	-	2000	MHz
Output Frequency	Fout	V_{DDPLL} = 1.2V	32		2000	MHz
Locking Time	T_{LT}	V_{DDPLL} = 1.2V			400	cycle

Ta = 25oC unless otherwise specified.

Table 5.7 DC Electrical Characteristics for PLL3/4/5

Parameter	Symbol	Test Condition	MIN	TYP	MAX	Unit
Power Down Current	I_{PD}	V_{DDPLL} = 1.2V			120	μA
Power Consumption	P_{DD}	V_{DDPLL} = 1.2V			1.2	mW

Ta = 25oC unless otherwise specified.

Table 5.8 AC Electrical Characteristics for PLL3/4/5

Parameter	Symbol	Test Condition	MIN	TYP	MAX	Unit
Input Frequency	Fin	V_{DDPLL} = 1.2V		12		MHz
VCO output frequency	Fvco	V_{DDPLL} = 1.2V	330	-	660	MHz
Output Frequency	Fout	V_{DDPLL} = 1.2V	12		660	MHz
Locking Time	T_{LT}	V_{DDPLL} = 1.2V			400	cycle

Ta = 25oC unless otherwise specified.

5.7 Electrical Characteristics for Video DAC

Table 5.9 DC Electrical Characteristics for DAC

Parameter	Symbol	Test Condition	MIN	TYP	MAX	Unit
Resolution	Bit		-	-	10	bits
Conversion Rate	F_{CLK}		-	-	27	MHz
Differential Non-Linearity	DNL		-	-	±1	LSB
Integral Non-Linearity	INL		-	-	±2	LSB
Full Scale Output Voltage	V_O		1.17	1.3	1.43	V
Output Load	R_{LOAD}	±1% tolerance		37.5		Ω
External Reference Voltage	V_{REF}		-	1.26	-	V

(VDDDAC =3.0V, VSSDAC =0V, Power Down = OFF, Top=30° C,
R(IREF) =1.2kΩ, Load Resistance=37.5Ω unless otherwise specified.)

5.8 Electrical Characteristics for ADC(for Touch Screen)

Table 5.10 DC Electrical Characteristics for ADC

Parameter	Symbol	Test Condition	MIN	TYP	MAX	Unit
Resolution	Bit		-	-	12	bits
Differential Non-Linearity	DNL	VREF=3.3V, GND=0V	-	±0.5	±1	LSB
Integral Non-Linearity	INL	VREF=3.3V, GND=0V	-	±2	±4	LSB
Offset Voltage	TOPOFF BOTOFF	VREF=3.3V, GND=0V	-	-	20	LSB

(Converter Specifications: VDDADC= 3.3V, VSSADC= 0V, Top=25°C, VREF=3.3V, GND=0.0V unless otherwise specified)

Table 5.11 AC Electrical Characteristics for ADC

Parameter	Symbol	Test Condition	Min	Typ	Max	Unit
Maximum conversion rate	fc	f_{CKIN} = 5MHz	–	–	1	MSPS
Standby supply current	–	STBY = VDD	–	-	10	uA
Dynamic supply current	IVDD	f_{CKIN} = 5MHz (without system load)	–	4	6	mA
Total harmonic distortion	THD	f_{CKIN} = 5MHz f_{AIN} = 100kHz	65	77	-	dB
Signal-to-noise & distortion ratio	SNDR	f_{CKIN} = 5MHz f_{AIN} = 100kHz	54	60	–	dB

(Converter Specifications: VDDADC =3.3V, VSSADC=0V, Top=25°C, VREF=3.3V, GND=0.0V unless otherwise specified) VDDIO = 3.3V±0.3V

5.9 Electrical Characteristics for HDMI PHY

Table 5.12 DC Electrical Characteristics for HDMI PHY

Parameter	Symbol	Test Condition	MIN	TYP	MAX	Unit
Normal Mode Operating Power	P_{CC}	720p @60Hz	-	36	-	mW
Power-Down Mode Power	P_{PD}	-	-	0.5	-	mW

Table 5.13 AC Electrical Characteristics for HDMI Oscillator

Parameter	Symbol	Test Condition	Min	Typ	Max	Unit
HDMI Oscillator Frequency	F_R	-	–	27		MHz
Frequency Tolerance	F_{TOL}		-100		100	ppm
Duty Cycle	D_C		40		60	%
Jitter	J_{CLKI}	Peak-to-Peak Jitter RMS Jitter			50 3.5	ps ps

5.10 Electrical Characteristics for LCD Interface

The following figure shows the timing diagram for TFT-LCD with RGB interface. All the timing parameters can be configured by LHTIME1, LHTIME2, LVTIME1 ~ 4 registers.

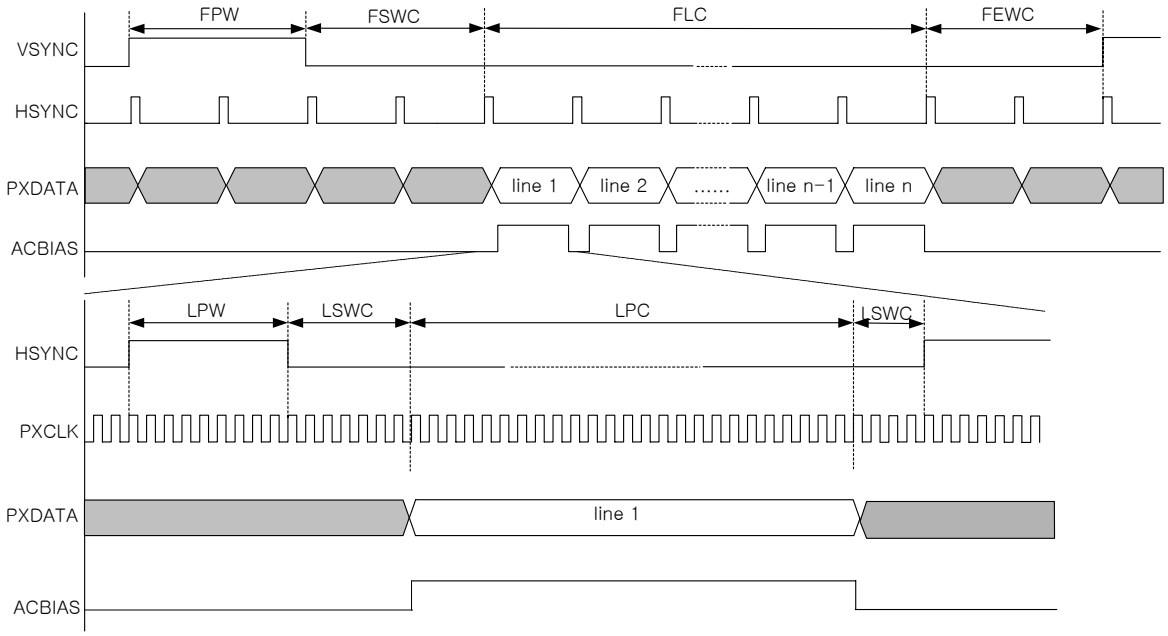


Figure 5.1 Timing Diagram for LCD Controller

The LHS (HSYNC), LVS (VSYNC), LBIAS (ACBIAS, Data Enable) and LPD[17:0] (PXDATA[17:0]) signals are referenced by LCK (PXCLK). Each min and max timing for the output delay are shown in the following figure.

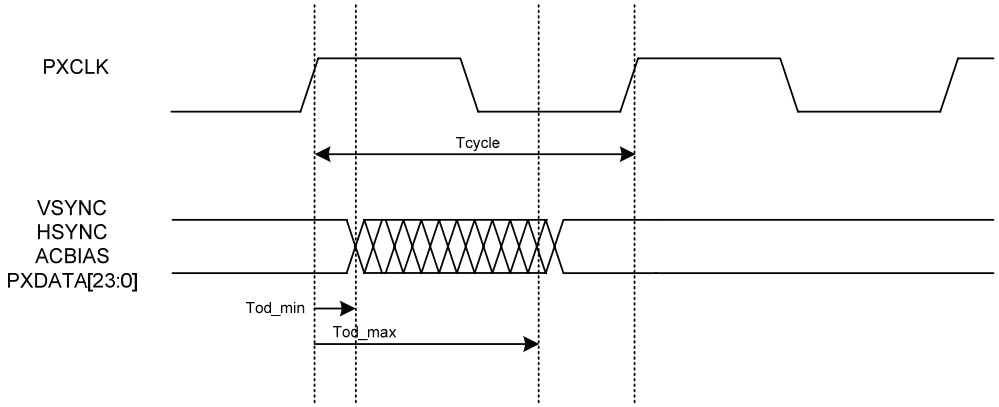


Figure 5.2 Timing Diagram Data Output Referenced to PXCLK

Table 5.14 Timing Parameters for Each Symbol

Parameter	Symbol	Min	Max	Unit	Remark
Clock Cycle	T_{CYCLE}	10	-	ns	
Output Delay	T_{OD}	0	7	ns	

Table 5.15 I/O Function Name for Corresponding Signal Name

Signal Name	I/O Function Name
VSYNC	L0_LVS, L1_LVS
HSYNC	L0_LHS, L1_LHS
ACBIAS	L0_LDE, L1_LDE
PXDATA[23:0]	L0_LPD[23:0], L1_LPD[23:0]
PXCLK	L0_LCK, L1_LCK

The following figure shows the timing diagram of bus interface to CPU I/F LCD device. The reference clock is used internally and the cycle time is defined as the register value written by software.

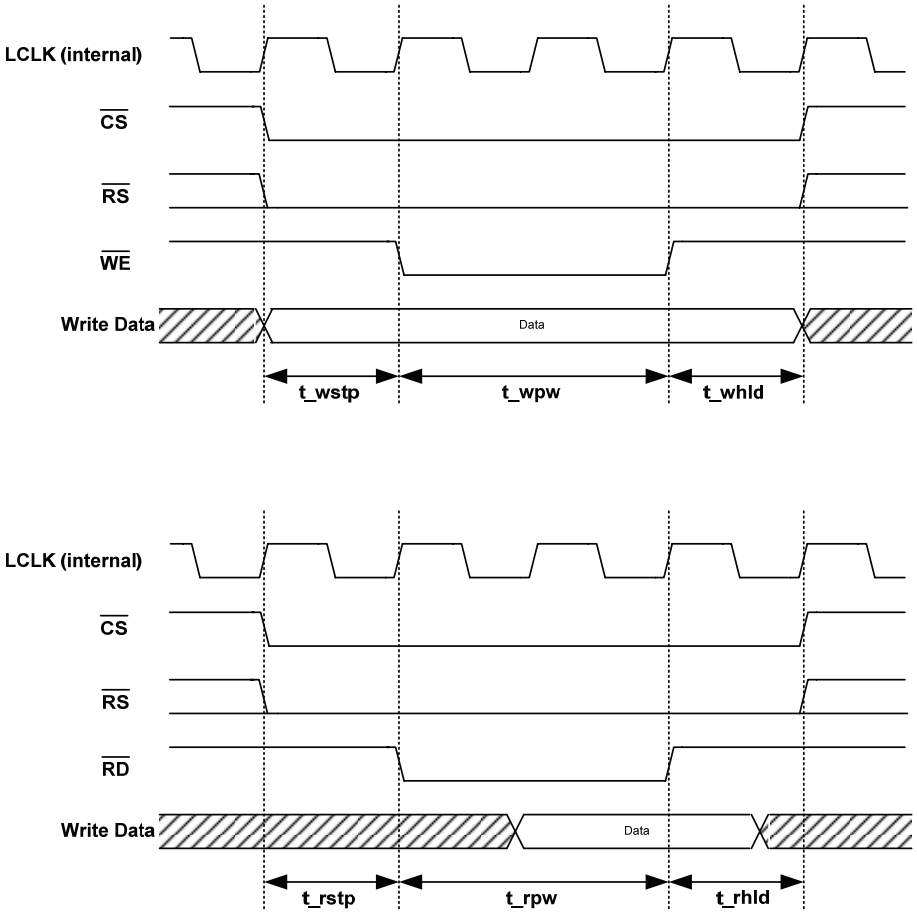


Figure 5.3 Timing Diagram Data Output Referenced to LCDSI

Table 5.16 Timing Parameters for Each Symbols

Parameter	Symbol	Min	Max	Unit	Remark
LCLK Clock Period (LCDSI Clock)	tCLK	18		ns	
RD/WE Setup Time Referenced to LCLK	T_rstp	0 * tCLK	7 * tCLK	ns	
RD/WE Pulse Width Referenced to LCLK	T_rpw	1 * tCLK	256 * tCLK	ns	
RD/WE Hold Time Referenced to LCLK	T_rhld	0 * tCLK	7 * tCLK	ns	

Signal Name	I/O Function Name
#CS	LCSN0, LCSN1
#RS	LXA[0]
#RD	LOEN
#WE	LWEN
Write Data[17:0]	LXD[17:0]

5.11 Electrical Characteristics for Camera Interface

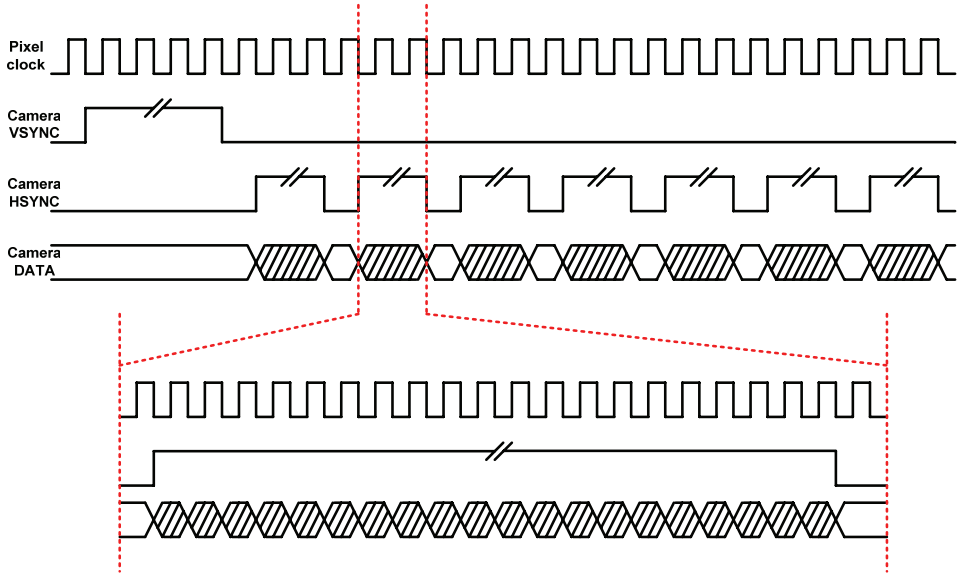


Figure 5.4 Timing Diagram for Camera Interface

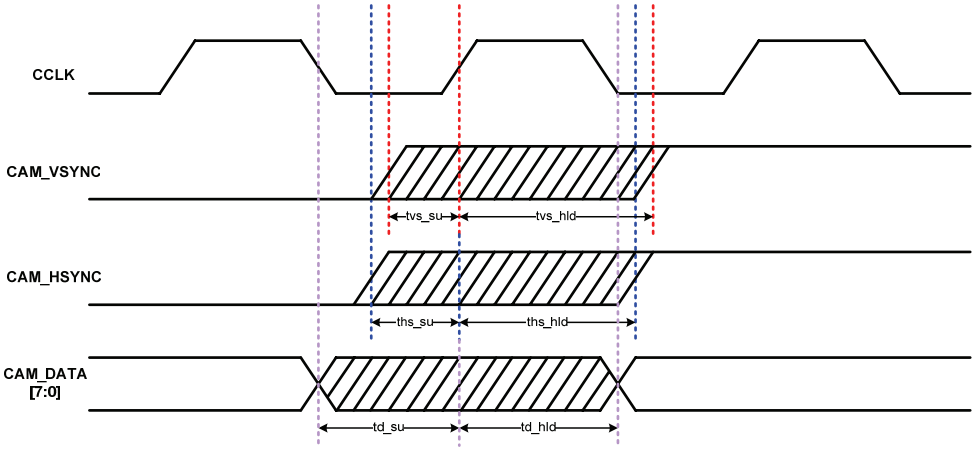


Figure 5.5 Timing Diagram Data Output Referenced to CCLK

Table 5.17 Timing Parameters for Each Symbol

Parameter	Symbol	Min	Max	Unit	Remark
Clock Frequency	TCCK		150	MHz	
Setup time for CVS(CAM_VSYNC)	Tvs_su	2		ns	
Hold time for CVS(CAM_VSYNC)	Tvs_hld	2		ns	
Setup time for CHS(CAM_HSYNC)	Ths_su	2		ns	
Hold time for CHS(CAM_HSYNC)	Ths_hld	2		ns	
Setup time for CPD[7:0](CAM_DATA)	Td_su	2		ns	
Hold time for CPD[7:0](CAM_DATA)	Td_hld	2		ns	

Table 5.18 I/O Function Name for Corresponding Signal Name

Signal Name	I/O Function Name
CCLK	CCKI
CAM_VSYNC	CVS
CAM_HSYNC	CHS
CAM_DATA[7:0]	CPD[7:0]

5.12 Electrical Characteristics for External Host Interface (EHI)

The EHI has two clock inputs; one is HCLK, which is for the on-chip system bus, the other is ECLK, which is for interface with the external host device. Therefore, interface timing with the external host is only related with ECLK.

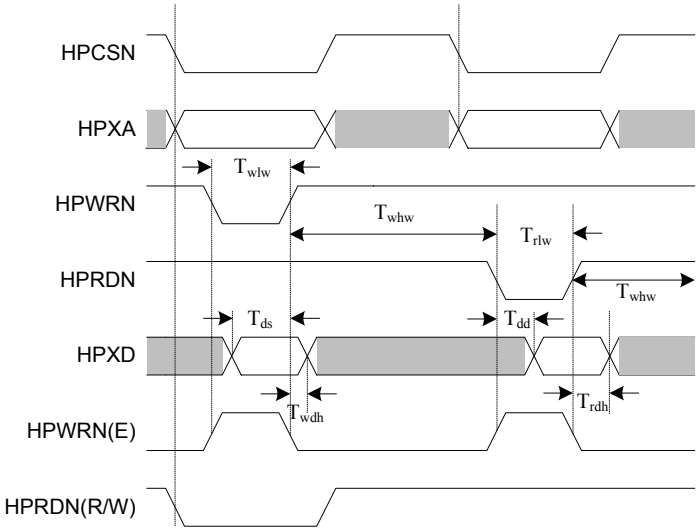


Figure 5.6 EHI Timing Diagram

Symbol	Description	Min.	Max.	Unit
Twlw	Write low width	2TP[19]	-	ns
Twhw	Write high width	3TP	-	ns
Trlw	Read low width	4TP	-	ns
Trhw	Read high width	3TP	-	ns
Tds	Data setup time	10	-	ns
Twdh	Write data hold	5	25ns	ns
Tdd	Data delay time	-	3TP +10ns	-
Trdh	Read data hold	0		ns

19 TP = ECLK period (ns)

5.13 Electrical Characteristics for SD/MMC Controller

The SD/MMC host controller is designed to supports high-speed mode (SD rev.1.10, up to 50 MHz Clock) as well as default speed mode (SD rev.1.01, up to 25 MHz Clock). A user doesn't need to set differently our SD/MMC host controller for mode change between default mode and high speed mode. If you want to change mode to high-speed mode from default mode and vice versa, by using switch-function command (CMD6), the SD/MMC cards are set to such mode. The timing diagram shows the input/output timing criterion referenced to SD/MMC clock.

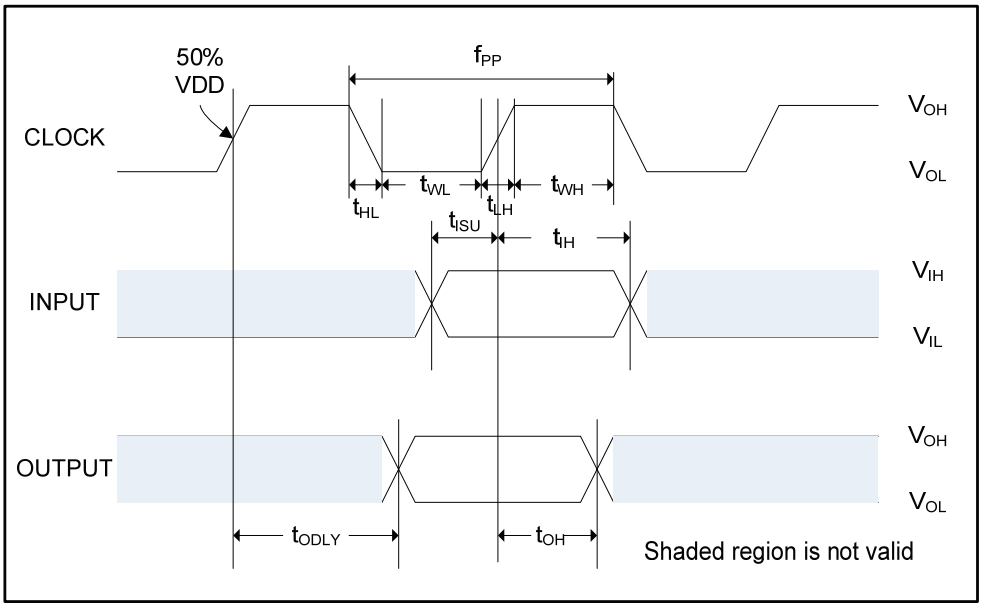


Figure 5.7 Timing Diagram for SD/MMC Controller

Table 5.19 Timing Parameters for Each Symbol

Parameter	Symbol	Min	Max	Unit	Remark
Clock frequency Data Transfer Mode	fPP	0	50	MHz	Ccard ≤ 30pF
Clock low time	tWL	7		ns	Ccard ≤ 30pF
Clock high time	tWH	7		ns	Ccard ≤ 30pF
Clock rise time	tLH		3	ns	Ccard ≤ 30pF
Clock fall time	tHL		3	ns	Ccard ≤ 30pF
Input set-up time	tISU	6		ns	Ccard ≤ 30pF
Input hold time	tIH	2.5		ns	Ccard ≤ 30pF
Output delay time	tODLY	10		ns	Ccard ≤ 30pF
Output hold time	tOH	2		ns	Ccard ≤ 30pF

5.14 Electrical Characteristics for I2C Controller

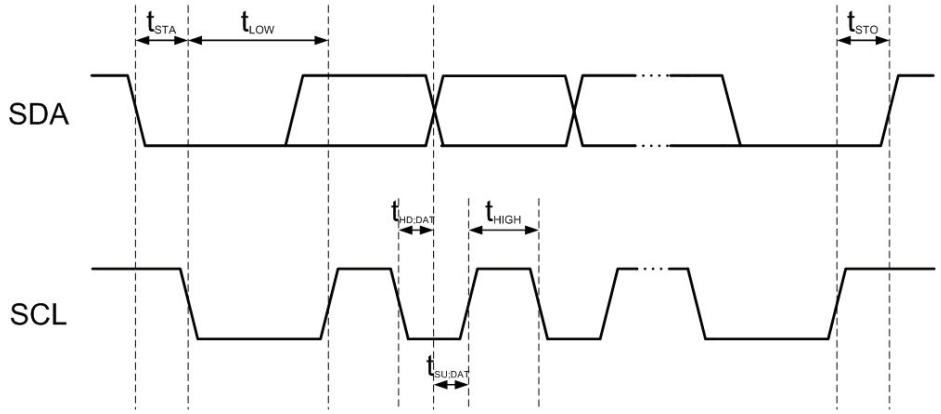


Figure 5.8 Timing Diagram for I2C Controller

Table 5.20 Timing Parameters for Each Symbol

Parameter	Symbol	Min	Max	Unit	Remark
SCL clock frequency		0	400	KHz	
Hold time(repeated) START condition	tSTA	0.95	-	us	
Data hold time	tHD:DAT	0.9		us	
Data setup time	tSU:DAT	0.4	-	us	
HIGH period of the SCL clock	tHIGH	0.96	-	us	
LOW period of the SCL clock	tLOW	1.4	-	us	
Setup time for STOP condition	tSTO	1.0	-	us	

5.15 Electrical Characteristics for SPDI/F Transmitter

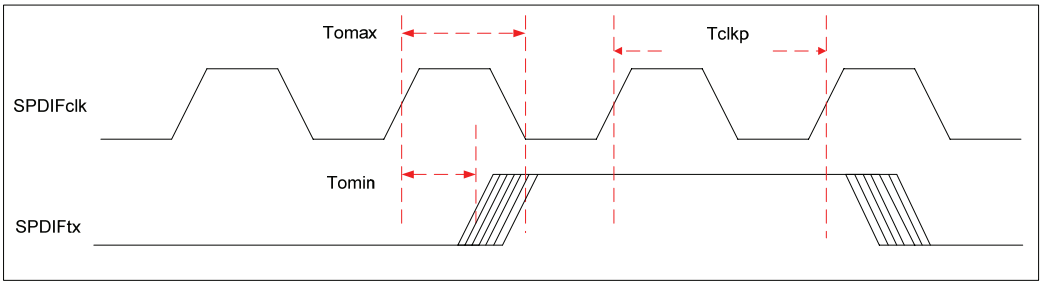


Figure 5.9 Timing Diagram for SPDI/F Transmitter

Table 5.21 Timing Parameters for Each Symbol

Parameter	Symbol	Min	Max	Unit	Remark
SPDIFclk Clock Cycle Time	Tclkp	110		ns	
SPDIFclk Data Output Time Referenced to SPDIFclk	Tomin/Tomax	1	10	ns	CL = 50pF

5.16 Electrical Characteristics for DAI(I2S)

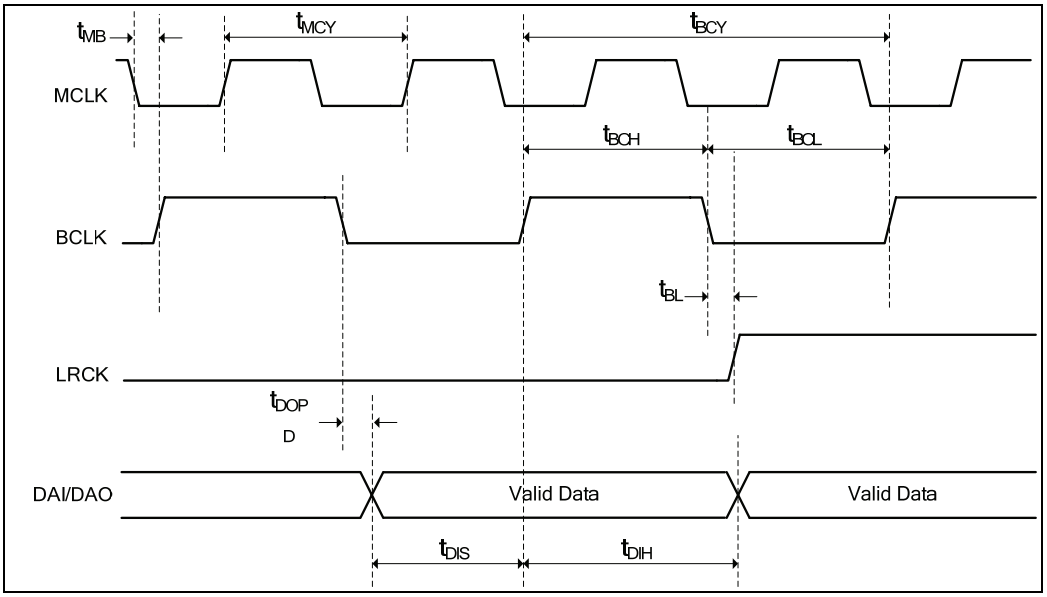


Figure 5.10 Timing Diagram for DAI (receiver)

Test Conditions
MODE = I2S, fs = 48KHz, MCLK = 256fs, BCLK = 64fs

Table 5.22 Timing for DAI (receiver)

Parameter	Symbol	Min	Typ	Max	Unit	Remark
MCLK cycle time	tMCY	19.40	81.40		ns	
BCLK cycle time	tBCY	4 * tMCY	4 * tMCY		ns	
BCLK pulse width high	tBCH	39	163		ns	
BCLK pulse width low	tBCL	38	162		ns	
MCLK to BCLK	tMB	0.18	0.18		ns	
BCLK to LRCK	tBL	14.77	14.77		ns	
DAI setup time to BCLK rising edge	tDIS	1	1		ns	
DAI hold time from BCLK rising edge	tDIH	1	1		ns	
DAO Output Timing Referenced to BCLK	tDOPD			1	ns	

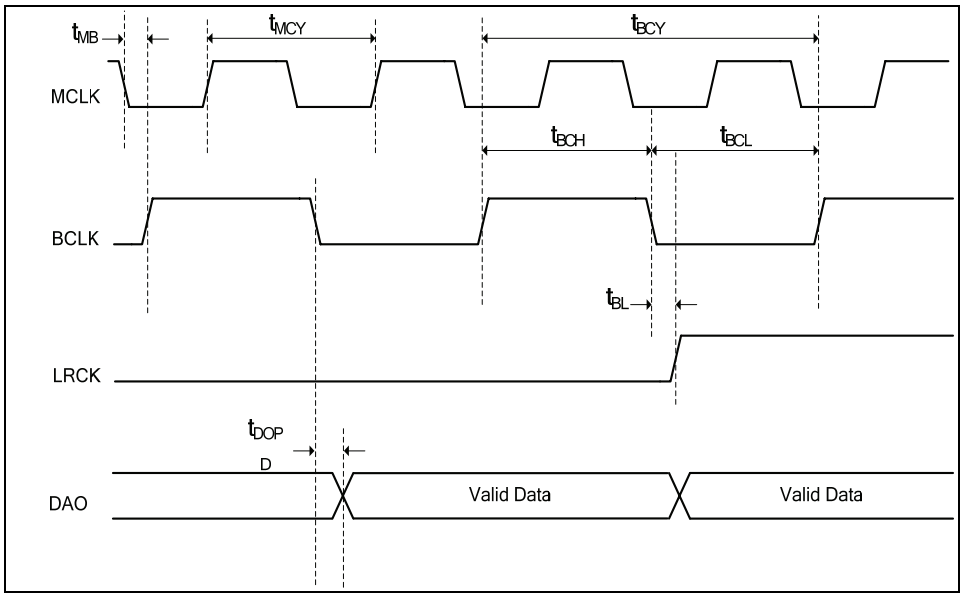


Figure 5.11 Timing Diagram for DAI Transmitter

Table 5.23 Timing Parameters for Each Symbols

Parameter	Symbol	Min	Max	Unit	Remark
MCLK cycle time	tMCY	36		ns	
BCLK cycle time	tBCY	16 * tMCY		ns	
BCLK pulse width high	tBCH	8 * tMCY		ns	
BCLK pulse width low	tBCL	8 * tMCY		ns	
MCLK to BCLK	tMB	19		ns	
BCLK to LRCK	tBL	13		ns	
DAO Output Timing Referenced to BCLK	tDOPD	1		ns	

5.17 Electrical Characteristics for Nand Flash Controller

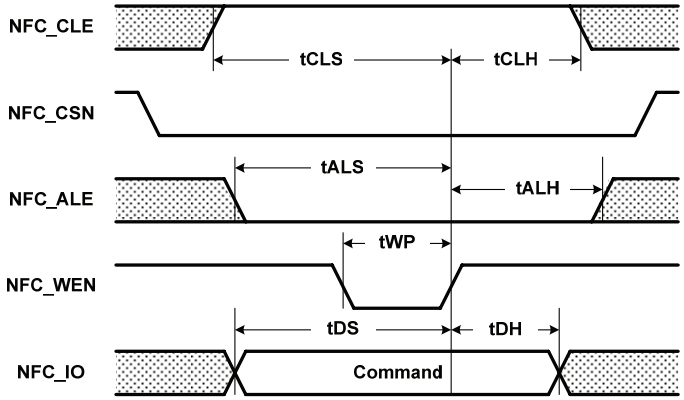


Figure 5.12 Timing Diagram for Command Latch Enable Cycle

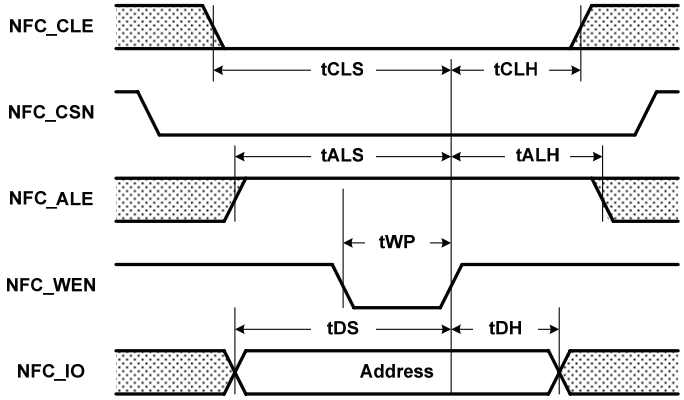


Figure 5.13 Timing Diagram for Single Address Latch Cycle

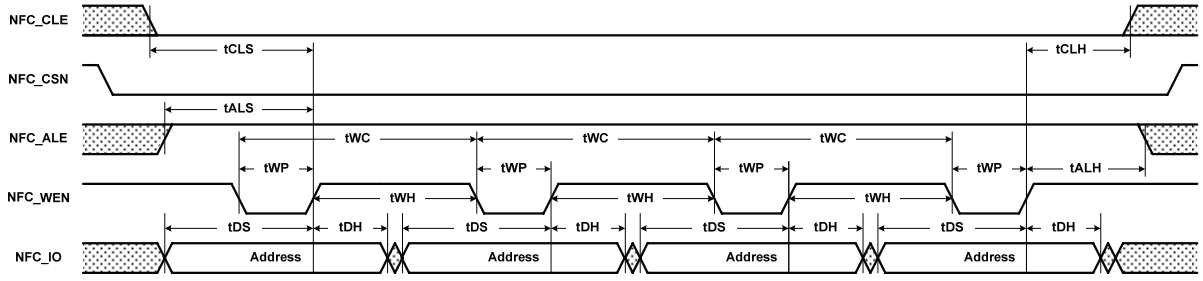


Figure 5.14 Timing Diagram for Linear Address Latch Cycle

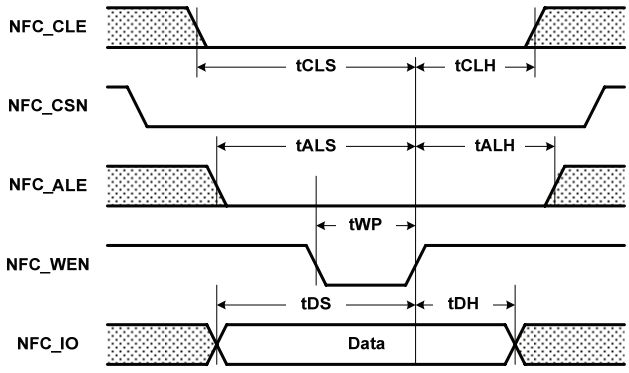


Figure 5.15 Timing Diagram for Single Data Write Cycle

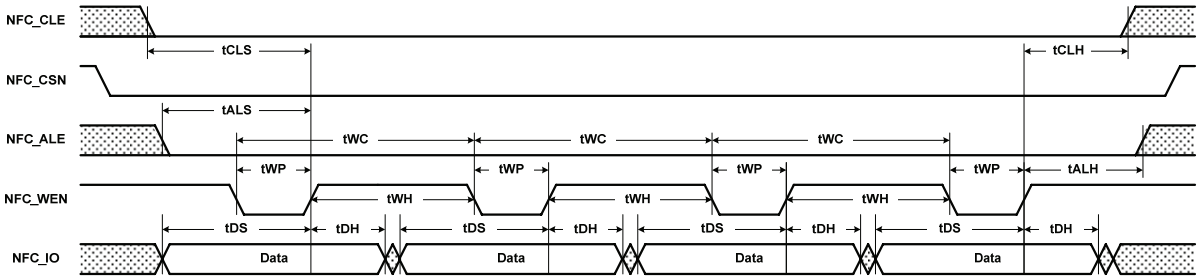


Figure 5.16 Timing Diagram for Linear Data Write Cycle

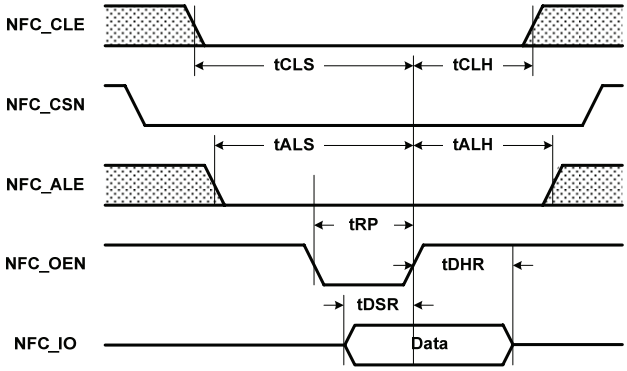


Figure 5.17 Timing Diagram for Single Data Read Cycle

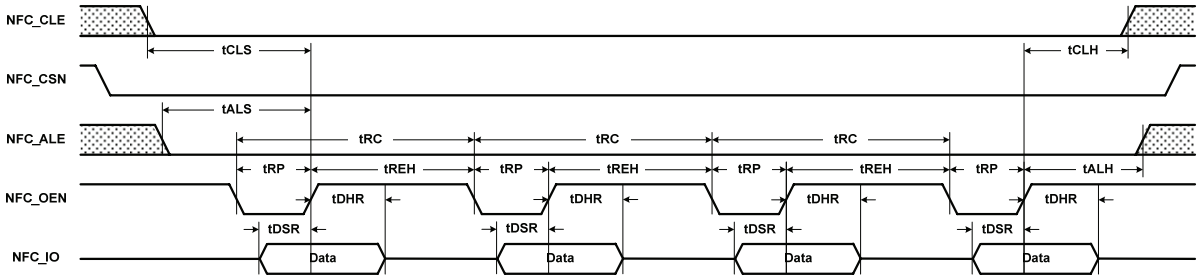


Figure 5.18 Timing Diagram for Linear Data Read Cycle

Table 5.24 Timing Parameters for Each Symbol

Parameter	Symbol	Min	Max	Unit
Clock Period	tHCLK	6		ns
CLE Set-Up Time	tCLS	(STP + PW) x tHCLK + 1.0	(STP + PW) x tHCLK + 2	ns
CLE Hold Time	tCLH	HLD x tHCLK – 2.0	HLD x tHCLK – 1.0	ns
WEN Pulse Width	tWP	PW x tHCLK	PW x tHCLK	ns
WEN High Hold Time	tWH	(STP + HLD) x tHCLK	(STP + HLD) x tHCLK	ns
Write Cycle Time	tWC	(STP + PW + HLD) x tHCLK	(STP + PW + HLD) x tHCLK	ns
OEN Pulse Width	tRP	PW x tHCLK	PW x tHCLK	ns
OEN High Hold Time	tREH	(STP + HLD) x tHCLK	(STP + HLD) x tHCLK	ns
Read Cycle Time	tRC	(STP + PW + HLD) x tHCLK	(STP + PW + HLD) x tHCLK	ns
ALE Set-Up Time	tALS	(STP + PW) x tHCLK – 1.00	(STP + PW) x tHCLK + 2.00	ns
ALE Hold Time	tALH	HLD x tHCLK – 2.00	HLD x tHCLK + 1.00	ns
Data Set-Up Time	tDS	(STP + PW) x tHCLK – 7.00	(STP + PW) x tHCLK – 1.00	ns
Data Hold Time	tDH	HLD x tHCLK – 1.00	HLD x tHCLK + 1.00	ns
Data Set-Up Time in READ	tDSR	5.00	15.0	ns
Data Hold Time in READ	tDHR	0	0	ns

Table 5.25 I/O Function Name for Corresponding Signal Name

Signal Name	I/O Function Name
NFC_CSN	NAND_CSN0, NAND_CSN1
NFC_ALE	NAND_ALE
NFC_CLE	NAND_CLE
NFC_OEN	NAND_OEN
NFC_WEN	NAND_WEN
NFC_IO[15:0]	NANDXD[15:0]

5.18 Electrical Characteristics for UART Controller

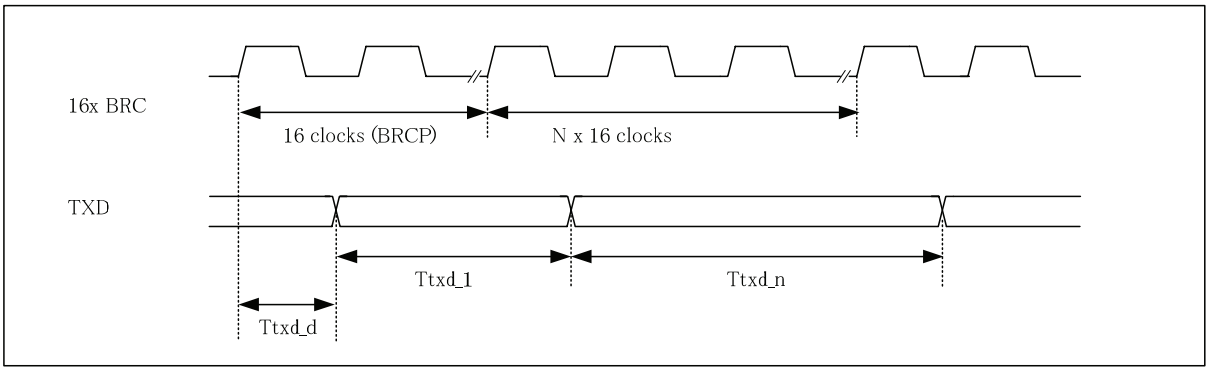


Figure 5.19 Timing Diagram for TXD

Table 5.26 Timing Parameters for Each Symbol

Parameter	Symbol	Min	Max	Unit	Remark
Pulse duration of 1bit TXD	Ttxd_1	BRCP -15	BRCP +15	ns	3.3V
Pulse duration of nbit TXD	Ttxd_n	N x BRCP -15	N x BRCP + 15	ns	3.3V
TXD output delay time	Ttxd_d	0.5	15	ns	3.3V

- BRC : Baud-Rate Clock
- BRCP : Baud-Rate Clock Period
- CL : 71pF

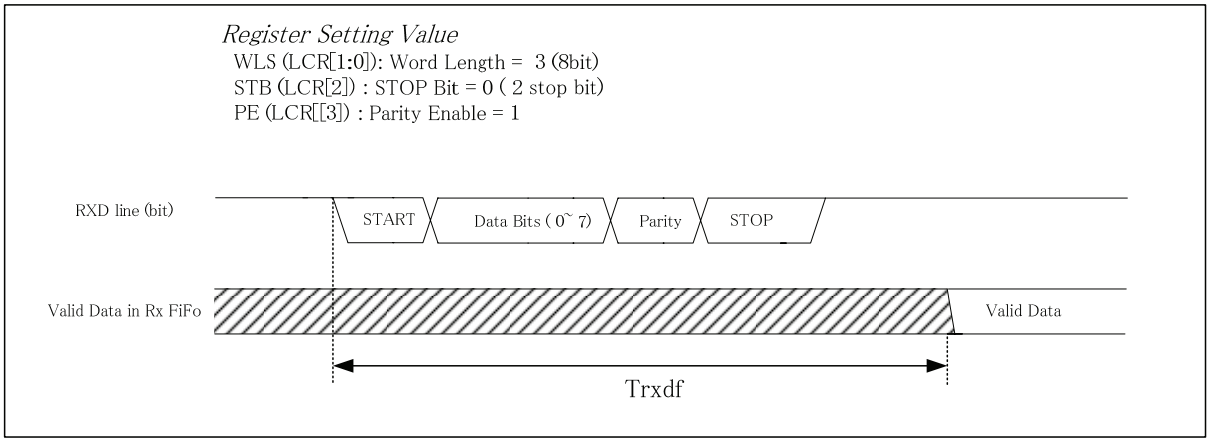


Figure 5.20 Timing Diagram for RXD

Table 5.27 Timing Parameters for Each Symbol

Parameter	Symbol	Min	Max	Unit	Remark
RXD Start to Rx FiFo.	Trxdf	10.5 x BRCP	11 x BRCP	ns	3.3V

- BRC : Baud-Rate Clock
- BRCP : Baud-Rate Clock Period
- CL : 71pF

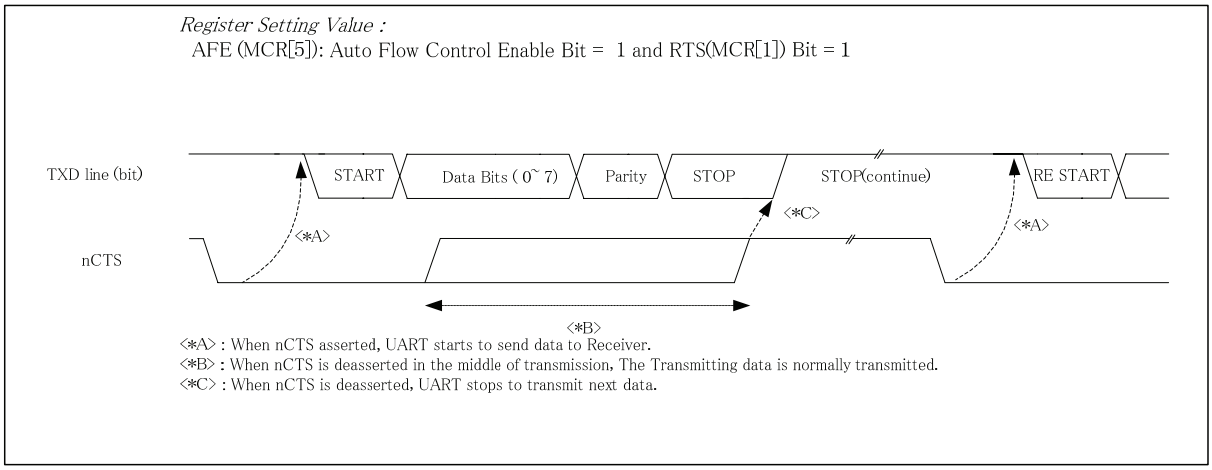


Figure 5.21 Timing Diagram for TX Operation with H/W Flow Control

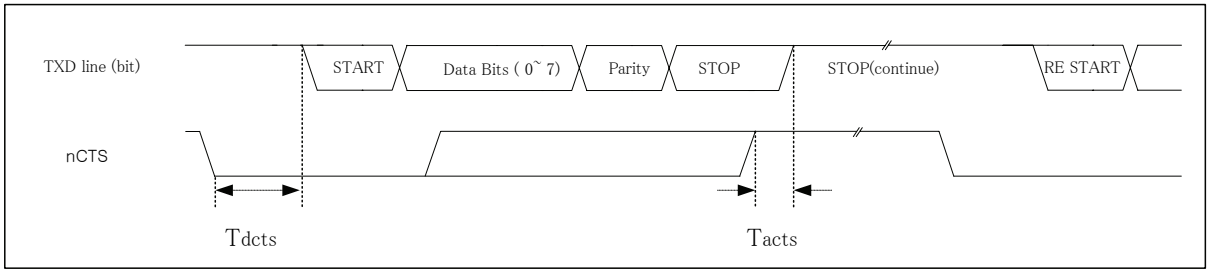


Figure 5.22 Timing Diagram for nCTS Timing Diagram

Table 5.28 Timing Parameters for Each Symbol

Parameter	Symbol	Min	Max	Unit	Remark
Deasserted nCTS to Tx Start	Tdcts	-	BRCP	ns	3.3V
Deasserted nCTS to Tx Stop :to stop next transmission(setup time)	Tacts	4 x BRCP/16	-	ns	3.3V

- BRC : Baud-Rate Clock , BRCP : Baud-Rate Clock Period , CL : 71pF

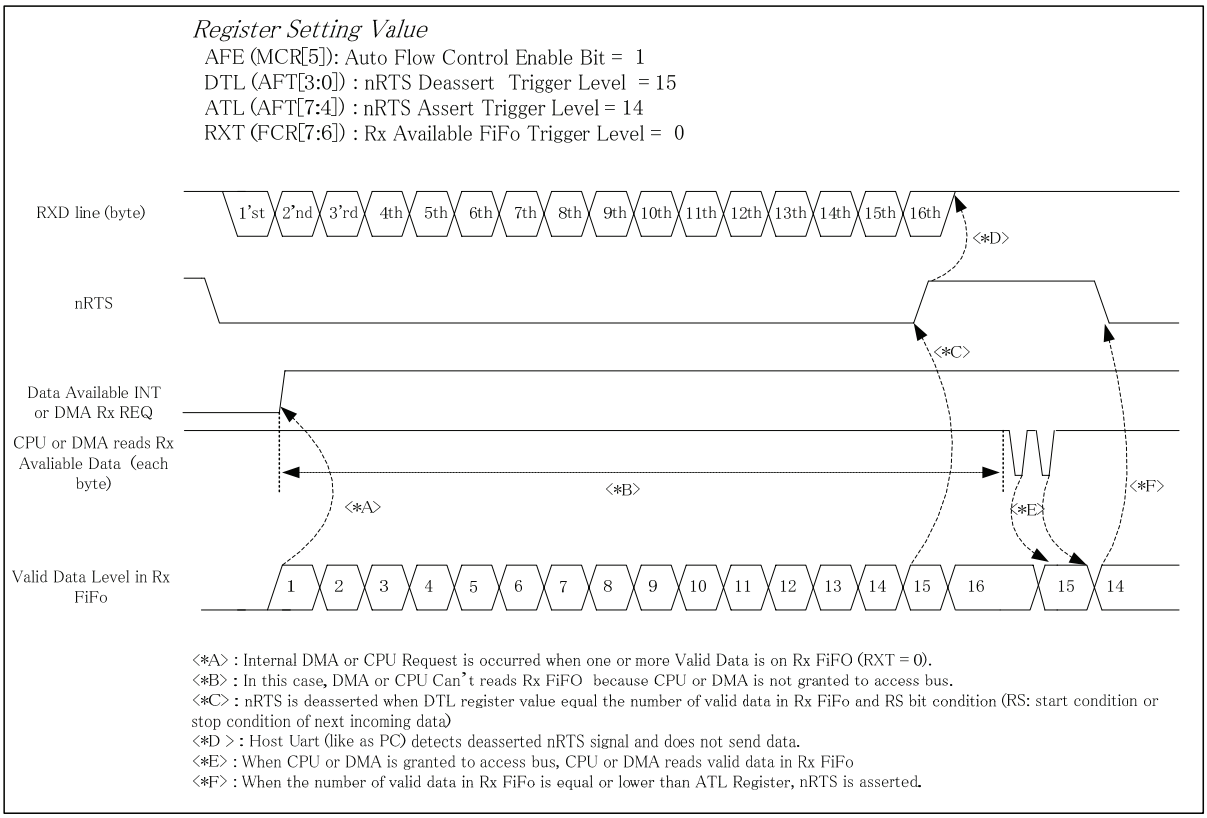


Figure 5.23 Timing Diagram for RX Operation with H/W Flow Control

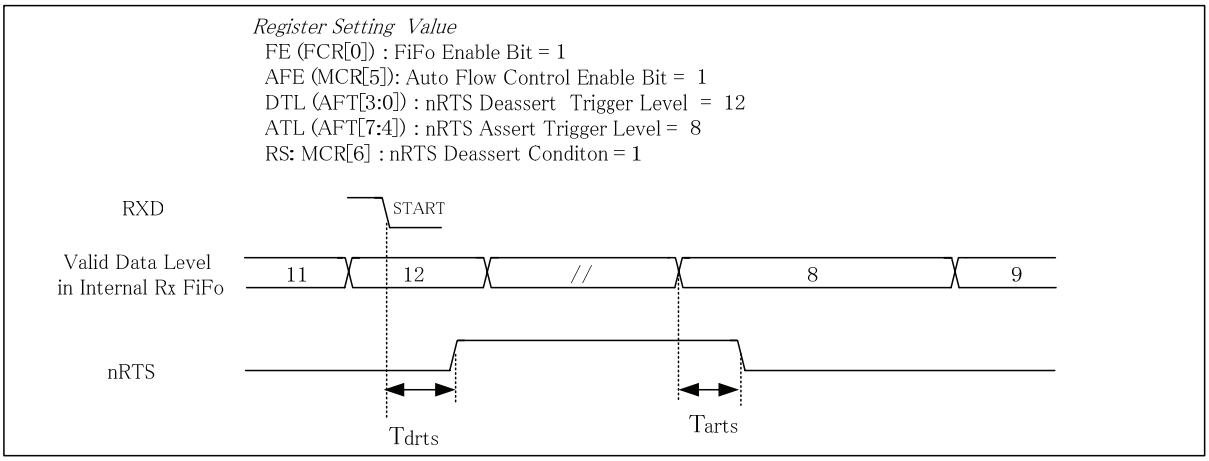


Figure 5.24 Timing Diagram for nRTS Timing Diagram

Table 5.29 Timing Parameters for Each Symbols

Parameter	Symbol	Min	Max	Unit	Remark
DTL(and RXD start condition) to deasserted nRTS	Tdrts	-	BRCP	ns	3.3V
ATL to asserted nRTS	Tarts	-	BRCP/16 + 8	ns	3.3V

- BRC : Baud-Rate Clock , BRCP : Baud-Rate Clock Period , CL : 71pF

5.19 Electrical Characteristics for DDR

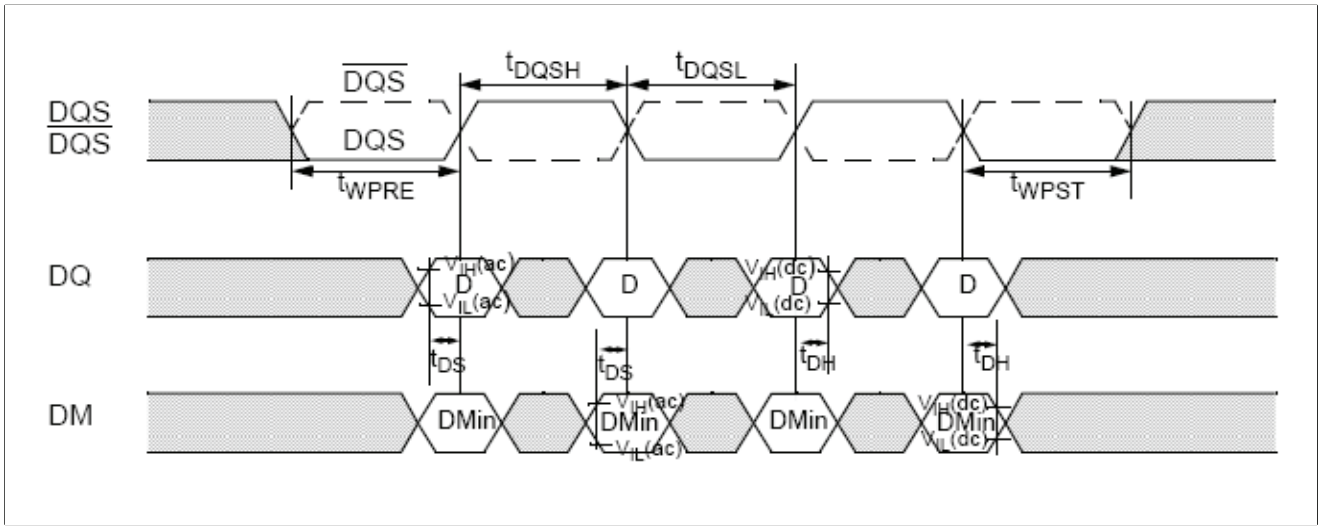


Figure 5.25 Write Cycle Timing

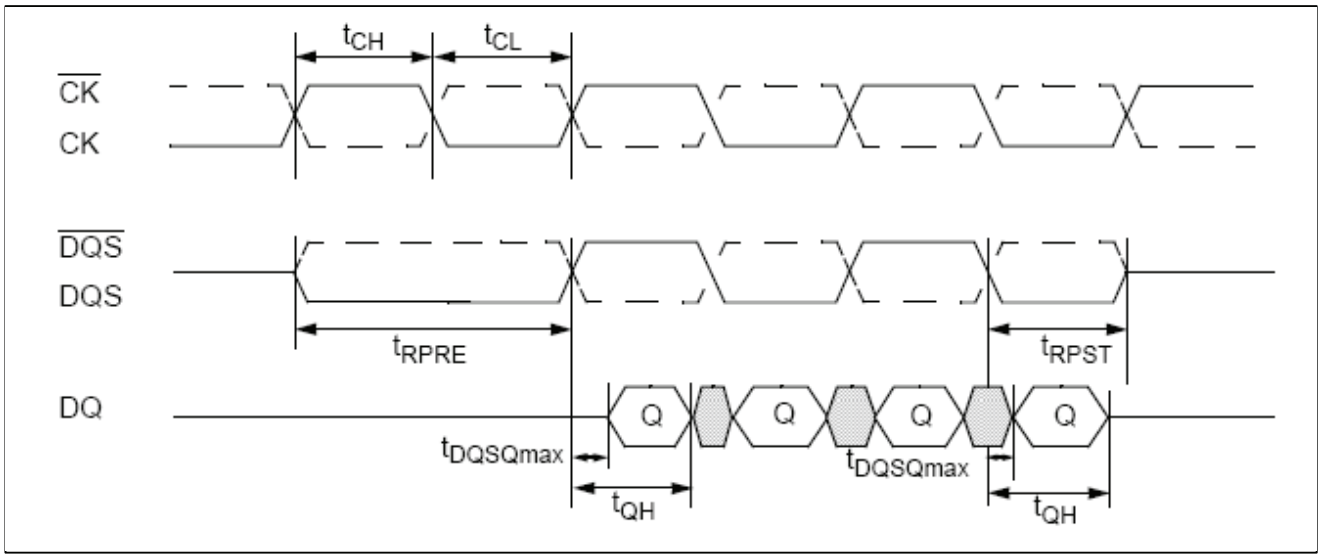


Figure 5.26 Read Cycle Timing

Table 5.30 DDR2-800 Interface Timing Parameters

S.No	Parameter	Notation	Min (ps)	Max (ps)
1	CK HIGH pulse width	tCH	480	520
2	CK LOW pulse width	tCL	480	520
3	Address and control output hold time	tIH	250	-
4	Address and control output setup time	tIS	175	-
5	DQ and DM output hold time	tDH	125	-
6	DQ and DM output setup time	tDS	50	-
7	DQS-DQ skew for DQS and associated DQ signals	tDQSQ	-	200

5.20 Electrical Characteristics for GPSB

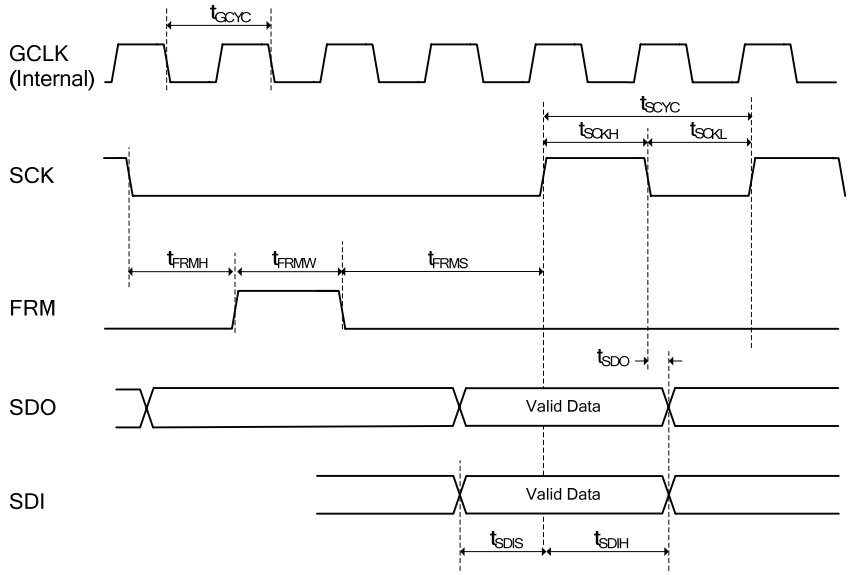


Figure 5.27 GPSB Interface Timing

Table 5.31 GPSB Interface Timing Parameters for SPI

Parameter	Symbol	Min	Max	Unit
GCLK cycle time	t_{GCYC}	20	-	ns
SCK cycle time	t_{SCYC}	2 * t_{GCYC}	-	ns
SCK pulse width high	t_{SCKH}	t_{GCYC} - t_{SCKR}	-	ns
SCK pulse width low	t_{SCKL}	t_{GCYC} - t_{SCKF}	-	ns
SCK rise time	t_{SCKR}	-	10	ns
SCK fall time	t_{SCKF}	-	10	ns
Master Mode				
FRM output low to SCK rising edge	t_{FRMS}	t_{SCYC}	-	ns
FRM output high from SCK falling edge	t_{FRMH}	t_{SCYC}	-	ns
FRM output high pulse width	t_{FRMW}	t_{SCYC}	-	ns
SDO output delay from SCK falling edge	t_{SDO}	-	10	ns
SDI setup time to SCK rising edge	t_{SDIS}	10	-	ns
SDI hold time from SCK rising edge	t_{SDIH}	10	-	ns
Slave Mode				
FRM input low setup time to SCK rising edge	t_{FRMS}	25	-	ns
FRM input low hold time from SCK falling edge	t_{FRMH}	10	-	ns
FRM input high pulse width	t_{FRMW}	20	-	ns
SDO output delay from SCK falling edge	t_{SDO}	2	15	ns
SDI setup time to SCK rising edge	t_{SDIS}	10	-	ns
SDI hold time from SCK rising edge	t_{SDIH}	10	-	ns

Note:

1. CL = 30pF
2. GCLK is an internal signal
3. Parameters are for SPI Timing 0

PART1 - OVERVIEW
NVS2310

Rev. 1.01

Apr 22, 2011

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial Release
2011-04-22	1.01	∗ Correct the register table and document format.

TABLE OF CONTENTS

Contents

Tables

1 Address and Register Map

1.1 Address Map

The NVS2310 has fixed address maps for on-chip resources and off-chip resources. The following table represents overall address space of system.

Table 1.1 Remap & Address Map of NVS2310

Address Space	Region	Device Name
0x00000000 – 0x0FFFFFFF	R.0	This region is remapped to as follows. 1) If Remap is 000b, On-chip boot-ROM for region R.E 1) If Remap is 001b, On-chip memory0 for region R.1 2) If Remap is 010b, Off-chip DRAM for region R.4 3) If Remap is 011b, Off-chip NOR for region R.9. 4) If Remap is 100b, On-chip Backup RAM R.D1. 5) If Remap is 101b, On-chip memory1 R.D0 6) others is reserved
0x10000000 – 0x1001FFFF	R.1	Assigned to on-chip 128 kB memory (IRAM0)Region
0x20000000 – 0x2FFFFFFF	R.2	Assigned to chip select 0 Initially the configuration register is set to SRAM
0x30000000 – 0x3FFFFFFF	R.3	Assigned to chip select 1 Initially the configuration register is set to IDE type device
0x40000000 – 0x4FFFFFFF	R.4	Assigned to Off-chip DRAM chip
0x50000000 – 0x5FFFFFFF	R.5	Assigned to Off-chip DRAM chip
0x60000000 – 0x6FFFFFFF	R.6	Assigned to Off-chip DRAM chip
0x70000000 – 0x7FFFFFFF	R.7	Assigned to Off-chip DRAM chip
0x80000000 – 0x8FFFFFFF	R.8	Assigned to chip select 2 Initially the configuration register is set to NAND flash
0x90000000 – 0x9FFFFFFF	R.9	Assigned to chip select 3 Initially the configuration register is set to ROM
0xA0000000 – 0xAFFFFFFF	R.A	Reserved
0xB0000000 – 0xBFFFFFFF	R.B	Assigned to on-chip peripherals
0xC0000000 – 0xCFFFFFFF	R.C	MMU Virtual Table
0xD0000000 – 0xD00FFFFF	R.D0	Assigned to on-chip 64kB memory (IRAM1) Region
0xD8000000 – 0xDFFFFFFF	R.D1	PMU Backup RAM (16 kB)
0xE0000000 – 0xE0003FFF	R.E	Assigned to internal boot ROM (24 kB)
0xF0000000 – 0xFFFFFFFF	R.F	Reserved

The address space (0x00000000 ~ 0x0FFFFFFF) is initially allocated to internal or external memory for booting procedure, and a special flag exists in system controller unit for remapping this space to other type of memories. Refer to the description of system controller for detailed operation.

The NVS2310 has various peripherals for specific interface controllers or on-chip hardware components. These peripherals can be configured appropriately by its own registers that can be accessed through specially allocated address. These address maps are represented in the following table. In case of memory controller, its space is separated for preventing illegal accessing.

Refer to corresponding sections for detail information of each peripheral.

Table 1.2 Base Address of Functional Bus

Base Address	Peripherals
0xB0000000	IO Bus AHB Components Registers
0xB0100000	IO Bus APB Components Registers
0xB0200000	Camera Bus Components Registers
0xB0300000	Memory Bus Components Registers
0xB0400000	Reserved
0xB0500000	System Bus Registers (SMU)
0xB0600000	VPIC
0xB0700000	Graphic Bus Components Registers
0xB0800000	High-speed IO Components Registers
0xB0900000	Video Bus Components Registers
0xB0A00000	Display Bus Components Registers

Table 1.3 IO AHB Component (Base Address = 0xB0000000)

Base Address	Peripherals
0x00000 ~ 0x0FFFF	EHI0
0x10000 ~ 0x1FFFF	EHI1
0x20000 ~ 0x201FF	SD/MMC0
0x20200 ~ 0x203FF	SD/MMC1
0x20400 ~ 0x205FF	SD/MMC2
0x20600 ~ 0x207FF	SD/MMC3
0x30000 ~ 0x300FF	GDMA0
0x30100 ~ 0x301FF	GDMA1
0x30200 ~ 0x302FF	GDMA2
0x30300 ~ 0x303FF	GDMA0/1/2 IRQ
0x40000 ~ 0x40FFF	USB OTG0
0x50000 ~ 0x50FFF	NFC/ECC
0x60000 ~ 0x60FFF	PWM generator
0x70000 ~ 0x70FFF	Overlay/Mixer
0x80000 ~ 0x80FFF	Peripheral Bus CFG

Table 1.4 IO APB Component (Base Address = 0xB0100000)

Base Address	Peripherals
0x0000 ~ 0x0FFF	TS ADC Controller
0x1000 ~ 0x1FFF	Remote Control Interface
0x2000 ~ 0x20FF	UART0
0x2100 ~ 0x21FF	UART1
0x2200 ~ 0x22FF	UART2
0x2300 ~ 0x23FF	UART3
0x2400 ~ 0x24FF	UART4
0x2500 ~ 0x25FF	UART5
0x3000 ~ 0x3FFF	Audio DMA0 (7.1 ch)
0x4000 ~ 0x4FFF	DAI0 (7.1ch)
0x5000 ~ 0x5FFF	Audio DMA1 (Stereo)
0x6000 ~ 0x6FFF	DAI1 (Stereo)
0x7000 ~ 0x70FF	GPSB0 with DMA
0x7100 ~ 0x71FF	GPSB1 with DMA
0x7200 ~ 0x72FF	GPSB2 with DMA
0x7300 ~ 0x73FF	GPSB3
0x7400 ~ 0x74FF	GPSB4
0x7500 ~ 0x75FF	GPSB5
0x8000 ~ 0x80FF	TSIF0
0x8100 ~ 0x81FF	TSIF1
0x9000 ~ 0x90FF	I2C0
0x9100 ~ 0x91FF	I2C1
0x9200 ~ 0x92FF	I2C2
0x9300 ~ 0x93FF	I2C IRQ
0x9400 ~ 0x94FF	I2C CFG
0xA000 ~ 0xA1FF	GPIO
0xB000 ~ 0xB1FF	SPDIF0
0xC000 ~ 0xC1FF	Reserved
0xD000 ~ 0xD1FF	SPDIF1

Table 1.5 Camera Bus Component (Base Address = 0xB0200000)

Base Address	Peripherals
0x00000 ~ 0x0FFFF	CIF
0x10000 ~ 0x1FFFF	ISP
0x20000~ 0x2FFFF	Camera Bus Bus Matrix
0x30000 ~ 0x3FFFF	Camera Bus CFG

Table 1.6 Memory Bus Component (Base Address = 0xB0300000)

Base Address	Peripherals
0x0000 ~ 0x0FFF	Memory Bus Bus Matrix
0x1000 ~ 0x1FFF	DDR SDRAM Controller 0 (LPDDR)
0x2000 ~ 0x20FF	DDR SDRAM Controller 1 (DDR2)
0x3000 ~ 0x3FFF	DMC CFG
0x4000 ~ 0x4FFF	DDR PHY CFG
0x5000 ~ 0x5FFF	DDR SDRAM Controller 2 (LPDDR/LPDDR2)
0x6000 ~ 0x6FFF	Static Memory Controller
0x7000 ~ 0x7FFF	IMC CFG
0x8000 ~ 0x8FFF	EDI
0x9000 ~ 0x9FFF	Reserved
0xA000 ~ 0xAFFF	Reserved
0xB000 ~ 0xBFFF	MBUS CFG
0xC000 ~ 0xCFFF	DDR SDRAM Controller 3 (DDR2/DDR3)

Table 1.7 System Bus Component (Base Address = 0xB0500000)

Base Address	Peripherals
0x0000 ~ 0x0FFF	CKC
0x1000 ~ 0x1FFF	Timer
0x2000 ~ 0x20FF	RTC
0x3000 ~ 0x3FFF	PMUIF
0x4000 ~ 0x4FFF	System Bus CFG
0x5000 ~ 0x5FFF	SMU I2C

Table 1.8 Graphic Bus Component (Base Address = 0xB0700000)

Base Address	Peripherals
0x0000 ~ 0x3FFF	2D/3D Graphic Controller
0x4000 ~ 0x4FFF	Graphic Bus CFG

Table 1.9 High Speed IO Bus Component (Base Address = 0xB0800000)

Base Address	Peripherals
0x00000~0x0FFFF	USB_OTG1
0x10000~0x1FFFF	Reserved
0x20000~0x2FFFF	Ethernet MAC Controller
0x30000~0x3FFFF	GDMA3
0x40000~0x4FFFF	Memory Stick Controller
0x50000~0x5FFFF	Reserved
0x60000~0x6FFFF	Reserved
0x70000~0x700FF	Cipher Controller
0x70100~0x701FF	Memory Stick Controller CFG
0x70200~0x702FF	High Speed IO Bus Bus Matrix
0x80000~0x8FFFF	High Speed IO Bus CFG

Table 1.10 Video Bus Component (Base Address = 0xB0900000)

Base Address	Peripherals
0x00000 ~ 0x0FFFF	Video CODEC
0x10000 ~ 0x1FFFF	Video Cache Controller
0x20000 ~ 0x2FFFF	Video Bus CFG
0x80000 ~ 0x8FFFF	JPEG Encoder

Table 1.11 Display Bus Component (Base Address = 0xB0A00000)

Base Address	Peripherals
0x00000 ~ 0x03FFF	LCD Controller 0
0x04000 ~ 0x07FFF	LCD Controller 1
0x08000 ~ 0x0BFFF	LCDSI0
0x0C000 ~ 0x0FFFF	LCDSI1
0x10000 ~ 0x1FFFF	MSCL0 (Memory Scaler 0)
0x20000 ~ 0x2FFFF	MSCL1 (Memory Scaler 1)
0x30000 ~ 0x3FFFF	Reserved
0x40000 ~ 0x4FFFF	TV Encoder
0x50000 ~ 0x50FFF	Display Bus Prefetch Buffer (DDIC)
0x51000 ~ 0x51FFF	Display Bus CFG
0x52000 ~ 0x52FFF	Video Enhancer
0x53000 ~ 0x53FFF	Video Enhancer (Lookup Table)
0x54000 ~ 0x54FFF	HDMI0 (Control)
0x55000 ~ 0x55FFF	HDMI1 (Core)
0x56000 ~ 0x56FFF	HDMI2 (AES)
0x57000 ~ 0x57FFF	HDMI3 (SPD)
0x58000 ~ 0x58FFF	HDMI4 (I2S)
0x59000 ~ 0x59FFF	HDMI5 (CEC)
0x5A000 ~ 0x5AFFF	Display Bus Bus Matrix
0x5B000 ~ 0x5BFFF	MSCL0 (HPF)

PART2 – SMU & PMU
NVS2310

Rev. 1.02

Jun 01, 2011

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial Release
2011-04-22	1.01	∗ Correct the register table and document format. ∗ Correct the error in watchdog register.
2011-06-01	1.02	∗ Correct the mis-spelled chip name.

TABLE OF CONTENTS

Contents

Figures

Tables

1 CKC

1.1 Overview

The CKC block generates all the necessary bus and peripheral clocks used in NVS2310. Various clock sources are provided for each bus/peripheral clock genetrators and Flexible clock frequency can be generated using independent clock divier for each clock generator.
The CKC block also generates software reset signal for each functional bus.

1.2 Bus Clock Generation

The bus clock generator has 16 primary clock sources from 6 PLLs, XIN, XTIN and divided 6 PLLs, divided XIN, divided XTIN. Fig. 1.1 show the structure of bus clock generators. The output clock of safe clock changer, which generates glitch-free clock from 16 independent clock sources, can be used for making each operating bus clocks.

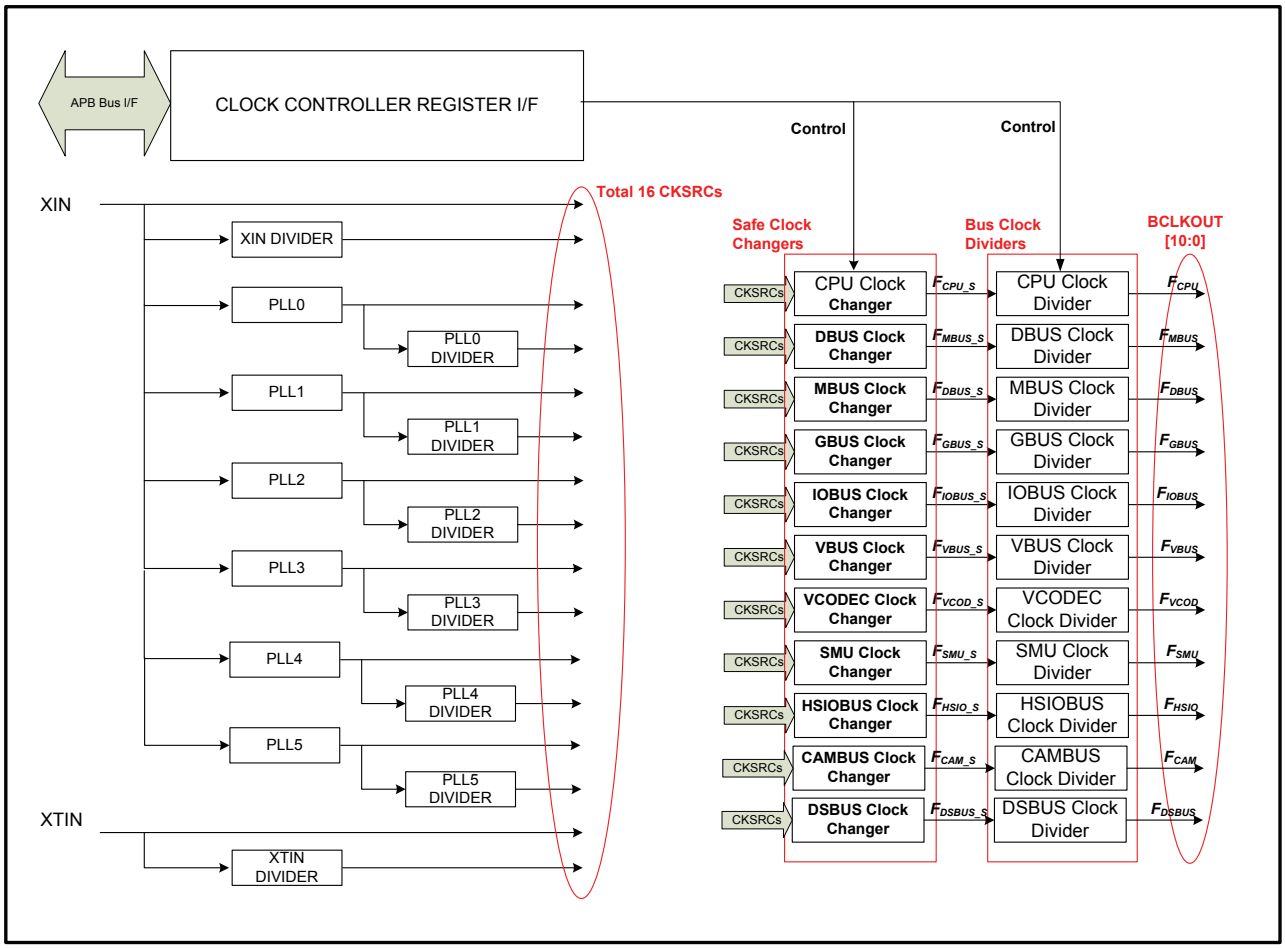


Figure 1.1 CKC Functional Bus Clock Generator

Table 1.1 Bus Clocks and Description

Name	Description
F_{CPU}	CPU Clock Frequency
F_{DBUS}	Display Bus Clock (Video In/Out + Display Related Hardwares)
F_{MBUS}	Asynchronous Memory Bus Clock
F_{GBUS}	Graphic Bus Clock
F_{VBUS}	Video Bus Clock
F_{VCOD}	Video Bus Core Clock
F_{IOBUS}	IO Bus Clock
F_{SMU}	System Management Unit Bus Clock
F_{HSIO}	High Speed IO Bus Clock
F_{CAM}	Camera Bus Clock
F_{DSBUS}	Display Sub Bus Clock

1.3 Peripheral Clock Generation

The peripheral clock generator makes the corresponding hardware peripheral clock using 5 primary clock sources and 19 generated clocks. Up to 24 selectable clock sources enable to generate various clock frequency for each peripherals.

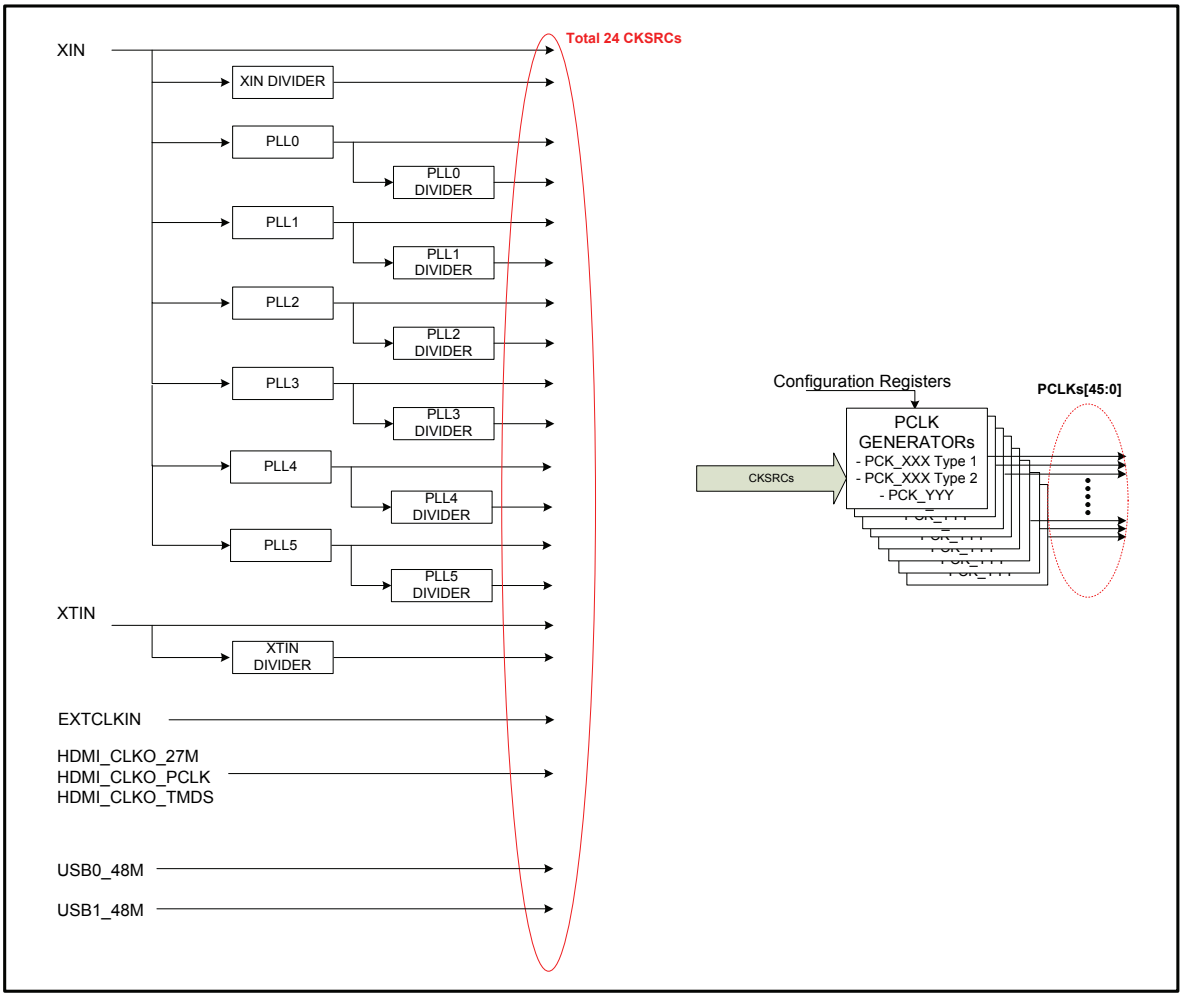


Figure 1.2 Peripheral Clock Generator

1.4 Bus Software Reset Generation

The software reset generation for each functional bus is controlled by program code through defined register setting and release

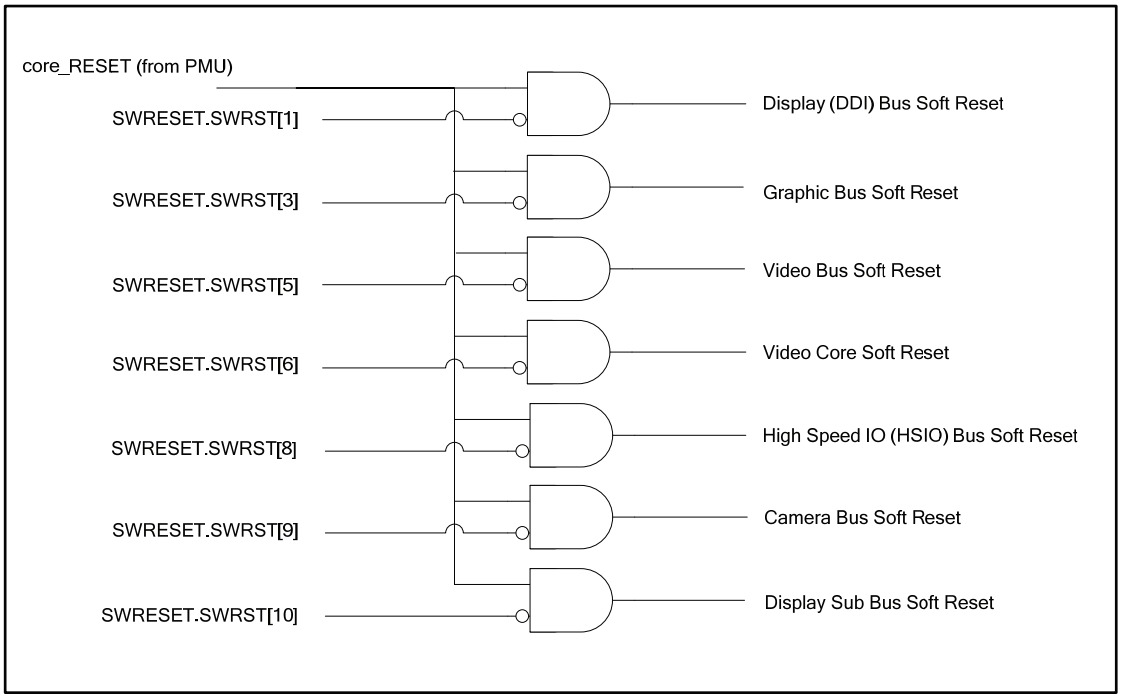


Figure 1.3 Software Reset Generation

1.5 Signal Descriptions

1.5.1 Global Signals

Table 1.2 Global Signals

Name	Direction	Descriptions	Related Blocks
PRESETn	Input	Reset Signal	

1.5.2 Clock Source Signals

Table 1.3 Clock Source Signals

Name	Direction	Descriptions	Related Blocks
XIN	Input	Clock Signal from PMU (12MHz Fixed)	PMU
XTIN	Input	Clock Signal from RTC	RTC
EXTCLKIN	Input	Clock Signal from External GPIO	GPIOA[13]
HDMI_CLKO_27M	Input	Clock Signal from HDMI Oscillator (HDMI_XI)	HDMI Oscillator
HDMI_CLKO_PCLK	Input	Clock Signal from HDMI Phy (Video Clock)	HDMI Phy
HDMI_CLKO_TMDS	Input	Clock Signal from HDMI Phy (TMDS Clock)	HDMI Phy
USB0_48M	Input	Clock Signal from USB OTG0 PHY (48 MHz)	USB OTG0 PHY
USB1_48M	Input	Clock Signal from USB OTG1 PHY (48 MHz)	USB OTG1 PHY

1.6 Register Descriptions

Table 1.4 CKC Register Map (Base Address = 0xB0500000)

Name	Address	Type	Reset	Description
CLKCTRL0	0x000	R/W	0x002FF014	CPU Clock Control Register
CLKCTRL1	0x004	R/W	0x002FF014	Bus Clock Control Register for Display Bus
CLKCTRL2	0x008	R/W	0x000FF014	Bus Clock Control Register for Memory Bus
CLKCTRL3	0x00C	R/W	0x000FF014	Bus Clock Control Register for Graphic Bus
CLKCTRL4	0x010	R/W	0x000FF014	Bus Clock Control Register for Peripheral Bus
CLKCTRL5	0x014	R/W	0x000FF014	Bus Clock Control Register for Video Bus
CLKCTRL6	0x018	R/W	0x002FF014	Core Clock Control Register for Video CODEC
CLKCTRL7	0x01C	R/W	0x002FF014	Bus Clock Control Register for SMU
CLKCTRL8	0x020	R/W	0x000FF014	Bus Clock Control Register for HSIOBus
CLKCTRL9	0x024	R/W	0x000FF014	Bus Clock Control Register for Camera Bus
CLKCTRL10	0x028	R/W	0x000FF014	Bus Clock Control Register for Display Sub Bus
MBUSCTRL	0x02C	R/W	0x07FFFC04	MBUS Clock Controller
PLL0CFG	0x030	R/W	0x00004B03	PLL0 Configuration Register
PLL1CFG	0x034	R/W	0x00004B03	PLL1 Configuration Register
PLL2CFG	0x038	R/W	0x00004B03	PLL2 Configuration Register
PLL3CFG	0x03C	R/W	0x00082943	PLL3 Configuration Register
PLL4CFG	0x040	R/W	0x00082943	PLL4 Configuration Register
PLL5CFG	0x044	R/W	0x00082943	PLL5 Configuration Register
CLKDIVC0	0x048	R/W	0x81818181	PLL0/1/2/3 Divider Configuration Register
CLKDIVC1	0x04C	R/W	0x00008181	XIN/XTIN Divider Configuration Register
CLKDIVC2	0x050	R/W	0x81810000	PLL4/5 Divider Configuration Register
-	0x054	R/W	-	Reserved
-	0x058	R/W	-	Reserved
SWRESET	0x05C	R/W	0x00000000	Functional Bus Software Reset Control Register
HCLKMASK_CFG	0x060	R/W	0x00000000	Functional Bus CFG Block Clock Mask Control Register
SWRESET_CFG	0x064	R/W	0x00000000	Functional Bus CFG Block Software Reset Control Register
				Reserved
PCLK_TCX	0x080	R/W	0x14000000	Control Register for Timer Counter X Clock
PCLK_TCT	0x084	R/W	0x14000000	Control Register for Timer Counter T Clock
PCLK_TCZ	0x088	R/W	0x14000000	Control Register for Timer Counter Z Clock
PCLK_LCD0	0x08C	R/W	0x14000000	Control Register for LCD Channel 0
PCLK_LCD1	0x090	R/W	0x14000000	Control Register for LCD Channel 1
PCLK_LCDSI0	0x094	R/W	0x14000000	Control Register for LCD System Interface 0(CPU I/F)
PCLK_CIFMC	0x098	R/W	0x14000000	Control Register for CIF Interfnal Camera
PCLK_CIFSC	0x09C	R/W	0x14000000	Control Register for CIF Scaler Camera
PCLK_OUT0	0x0A0	R/W	0x14000000	Control Register for External Clock Output 0
PCLK_OUT1	0x0A4	R/W	0x14000000	Control Register for External Clock Output 1
PCLK_HDMI	0x0A8	R/W	0x14000000	Control Register for HDMI PHY
PCLK_SDMMC2	0x0AC	R/W	0x14000000	Control Register for SD/MMC Channel 2
PCLK_SDMMC0	0x0B0	R/W	0x14000000	Control Register for SD/MMC Channel 0
PCLK_MSTICK	0x0B4	R/W	0x14000000	Control Register for Memory Stick
PCLK_I2C0	0x0B8	R/W	0x14000000	Control Register for I2C0 External Device Interface
PCLK_UART0	0x0BC	R/W	0x14000000	Control Register for UART Channel 0
PCLK_UART1	0x0C0	R/W	0x14000000	Control Register for UART Channel 1
PCLK_UART2	0x0C4	R/W	0x14000000	Control Register for UART Channel 2
PCLK_UART3	0x0C8	R/W	0x14000000	Control Register for UART Channel 3
PCLK_UART4	0x0CC	R/W	0x14000000	Control Register for EHI Channel 0
PCLK_UART5	0x0D0	R/W	0x14000000	Control Register for EHI Channel 1
PCLK_GPSB0	0x0D4	R/W	0x14000000	Control Register for GPSB Channel 0
PCLK_GPSB1	0x0D8	R/W	0x14000000	Control Register for GPSB Channel 1
PCLK_GPSB2	0x0DC	R/W	0x14000000	Control Register for GPSB Channel 2
PCLK_GPSB3	0x0E0	R/W	0x14000000	Control Register for GPSB Channel 3
PCLK_GPSB4	0x0E4	R/W	0x14000000	Control Register for Audio 0 (5.1ch)
PCLK_GPSB5	0x0E8	R/W	0x14000000	Control Register for Audio 1 (Stereo)
PCLK_ADC	0x0EC	R/W	0x14000000	Control Register for ADC (Touch Screen)
PCLK_SPDIF0	0x0F0	R/W	0x14000000	Control Register for SPDIF
PCLK_EHI0	0x0F4	R/W	0x14000000	Control Register for EHI0
PCLK_EHI1	0x0F8	R/W	0x14000000	Control Register for EHI1
PCLK_AUD0	0x0FC	R/W	0x14000000	Control Register for Audio 0 (7.1 ch)
PCLK_PDM	0x100	R/W	0x14000000	Control Register for PMD/PWM Generator

PCLK_LCDSI1	0x104	R/W	0x14000000	Control Register for LCD System Interface 1(CPU I/F)
PCLK_SDMMC1	0x108	R/W	0x14000000	Control Register for SD/MMC Channel 1
PCLK_I2C1	0x10C	R/W	0x14000000	Control Register for I2C1 External Device Interface
PCLK_AUD1	0x110	R/W	0x14000000	Control Register for Audio 1 (stereo)
PCLK_I2C2	0x114	R/W	0x14000000	Control Register for I2C2 External Device Interface
PCLK_ISPS	0x118	R/W	0x14000000	Control Register for ISP System
PCLK_ISPJ	0x11C	R/W	0x14000000	Control Register for ISP JPEG
PCLK_NFC	0x120	R/W	0x14000000	Control Register for NFC
PCLK_GMAC	0x124	R/W	0x14000000	Control Register for GMAC
PCLK_SD3	0x12C	R/W	0x14000000	Control Register for SDMMC3
PCLK_SPDIF1	0x130	R/W	0x14000000	Control Register for SPDIF1

CLKCTRL0 (CPU CLK) **0xB0500000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
										EN		CONFIG			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CONFIG												SEL			

Field	Name	RW	Reset	Description
21	EN	R/W	0	Clock Controller Enable Register '0' for Disable, '1' for Enable
19-4	CONFIG	R/W	1	The CONFIG[0] ~ [15] means clock output enable for the X'th cycle which makes 16 clock cycle repeatedly. For example, the CONFIG[15:0] = 1100101010110101b, the 1st, 2nd, 5th, 7th, 9th, 11th, 12th, 14th, 16th cycle are valid clock output. ***In synchronous clock mode, this config field should be CONFIG[15:0] = 1111111111111111b***
3-0	SEL	R/W	4	Clock source selection register for bus or CPU clocks 0000b : Direct output from PLL0 0001b : Direct output from PLL1 0010b : Direct output from PLL2 0011b : Direct output from PLL3 0100b : Direct output from XIN 0101b : Divided output from PLL0 0110b : Divided output from PLL1 0111b : Direct output from XTIN 1000b : Direct output from PLL4 1001b : Direct output from PLL5 1010b : Divided output from PLL2 1011b : Divided output from PLL3 1100b : Divided output from PLL4 1101b : Divided output from PLL5 1110b : Divided output from XIN 1111b : Divided output from XTIN

**** In synchronous clock mode, memory bus clock is dependent on CPU clock generated using this CPU clock control register (CLKCTRL0).***

CLKCTRL1 (Display Bus CLK) **0xB0500004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
										EN					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								CONFIG				SEL			

Field	Name	RW	Reset	Description
21	EN	R/W	0	Clock Controller Enable Register '0' for Disable, '1' for Enable
7-4	CONFIG	R/W	1	Bus configuration register CONFIG[3:0] : Divisor : 1/(DIVISOR+1) ***The value should not be 'ZERO'***
3-0	SEL	W	4	Clock source selection register for bus or CPU clocks 0000b : Direct output from PLL0 0001b : Direct output from PLL1 0010b : Direct output from PLL2 0011b : Direct output from PLL3 0100b : Direct output from XIN 0101b : Divided output from PLL0 0110b : Divided output from PLL1 0111b : Direct output from XTIN 1000b : Direct output from PLL4 1001b : Direct output from PLL5 1010b : Divided output from PLL2 1011b : Divided output from PLL3 1100b : Divided output from PLL4 1101b : Divided output from PLL5 1110b : Divided output from XIN 1111b : Divided output from XTIN

CLKCTRL2 (MBUS_CLK) **0xB0500008**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
										EN					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								CONFIG				SEL			

Field	Name	RW	Reset	Description
21	EN	R/W	0	Clock Controller Enable Register '0' for Disable, '1' for Enable
7-4	CONFIG	R/W	1	Bus configuration register CONFIG[3:0] : Divisor : 1/(DIVISOR+1) ***The value should not be 'ZERO'***
3-0	SEL	R/W	4	Clock source selection register for bus or CPU clocks 0000b : Direct output from PLL0 0001b : Direct output from PLL1 0010b : Direct output from PLL2 0011b : Direct output from PLL3 0100b : Direct output from XIN 0101b : Divided output from PLL0 0110b : Divided output from PLL1 0111b : Direct output from XTIN 1000b : Direct output from PLL4 1001b : Direct output from PLL5 1010b : Divided output from PLL2 1011b : Divided output from PLL3 1100b : Divided output from PLL4 1101b : Divided output from PLL5 1110b : Divided output from XIN 1111b : Divided output from XTIN

**** In synchronous clock mode, memory bus clock is dependent on CPU clock generated using the CPU clock control register(CLKCTRL0). In asynchronous clock mode memory bus clock is provided using this async memory bus clock control register (CLKCTRL1)***

CLKCTRL3 ~ 10 **0xB050000C ~ 0xB0500028**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
										EN					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								CONFIG				SEL			

Field	Name	RW	Reset	Description
21	EN	R/W	0	Clock Controller Enable Register '0' for Disable, '1' for Enable
7-4	CONFIG	R/W	1	Bus configuration register CONFIG[3:0] : Divisor : 1/(DIVISOR+1) ***The value should not be 'ZERO'***
3-0	SEL	R/W	4	Clock source selection register for bus or CPU clocks 0000b : Direct output from PLL0 0001b : Direct output from PLL1 0010b : Direct output from PLL2 0011b : Direct output from PLL3 0100b : Direct output from XIN 0101b : Divided output from PLL0 0110b : Divided output from PLL1 0111b : Direct output from XTIN 1000b : Direct output from PLL4 1001b : Direct output from PLL5 1010b : Divided output from PLL2 1011b : Divided output from PLL3 1100b : Divided output from PLL4 1101b : Divided output from PLL5 1110b : Divided output from XIN 1111b : Divided output from XTIN

MBUSCTRL **0xB050002C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMECOUNT[11:0]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BCLKDIV[7:0]															ASYNC

Field	Name	RW	Reset	Description
31-20	TIMECOUNT	R/W	0xFFF	Clock mode transition counter TIMECOUNT field controls transition time between synchronous and asynchronous clock mode ***Recommended TIMECOUNT value is 0x080***
15-8	BCLKDIV	R/W	0x01	$F_{MBUS} = F_{CPU} / (BCLKDIV + 1)$ BCLKDIV must be >= 1 and an odd number for clock duty cycle
0	ASYNC	R/W	0	Asynchronous clock mode enable '0' for disable asynchronous clock mode '1' for enable asynchronous clock mode

**** MBUSCTRL register is used for generating memory bus clock in synchronous clock mode.***
**** BCLKDIV should be an odd number so that memory bus clock maintains 50:50 clock duty cycle for DDR SDRAM type memory access***

PLL0CFG/PLL1CFG/PLL2CFG Register **0xB0500030~0xB0500038**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN	VSEL				S					LOCK	LEN			M	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
M								0		P					

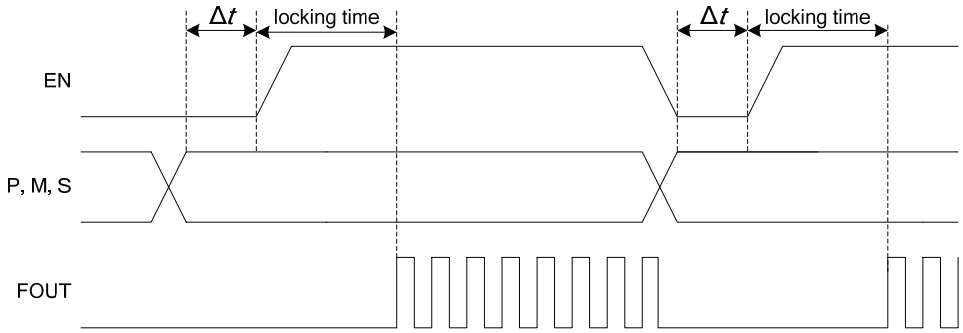
Field	Name	RW	Reset	Description
31	PD	R/W	0x0	PLL Enable Register 0 : PLL Disable 1 : PLL Enable
30	VSEL	R/W	0x00	VCO Range Selection 0 : Lower VCO range (1000 ~ 1400 MHz) 1 : Higher VCO range (1400 ~ 2000 MHz)
26-24	S	R/W	0x01	PLL S value (s = S) 0 ≤ S ≤ 5
21	LOCK	R	-	Lock Status 0 : PLL Unlocked 1 : PLL Locked
20	LEN	R/W	0x0	Lock Enable 0 : Lock disable 1 : Lock enable
17-8	M	R/W	0x12C	PLL M value (m = M) 16 ≤ M ≤1023
5-0	P	R/W	0x03	PLL P value (p = P) 1 ≤ P ≤ 63

The PLL0/PLL1/PLL2 has the following constraints.

F_{VCO} = (m * F_{IN}) / (p) ***: 1000 MHz ~ 2000 MHz (F_{IN} is XIN oscillator)***

$F_{PLL} = F_{VCO} / (2 ^ s)$: F_{PLL} should be less than 1GHz.

Whenever P, M, and S value are changed, the EN bit needs to be 0 for Δt (>1us) and to be 1 again to restart the PLL with new setting values.



Δt > 1us

locking time > 400 cycle

1 cycle = $1/F_{REF} = 1/ (F_{IN}/p)$

PLL3CFG/PLL4CFG/PLL5CFG Register 0xB050003C~0xB0500044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN	VSEL				S					LOCK	LEN				M
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
M								0		P					

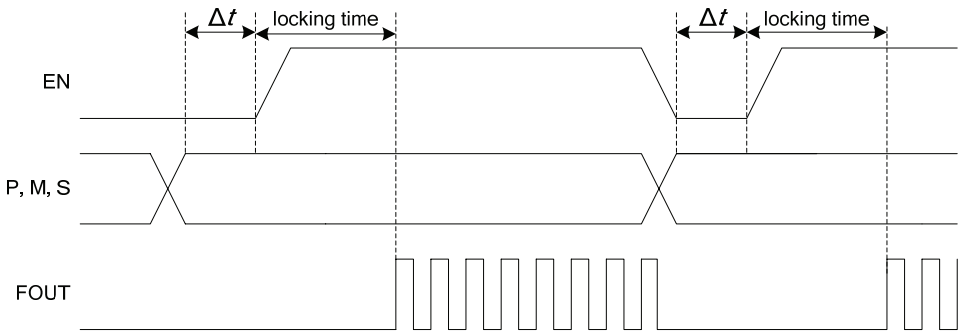
Field	Name	RW	Reset	Description
31	PD	R/W	0x0	PLL Enable Register 0 : PLL Disable 1 : PLL Enable
30	VSEL	R/W	0x00	VCO Range Selection 0 : Lower VCO range (330 ~ 460 MHz) 1 : Higher VCO range (460 ~ 660 MHz)
26-24	S	R/W	0x01	PLL S value (s = S) 0 ≤ S ≤ 5
21	LOCK	R	-	Lock Status 0 : PLL Unlocked 1 : PLL Locked
20	LEN	R/W	0x0	Lock Enable 0 : Lock disable 1 : Lock enable
16-8	M	R/W	0x12C	PLL M value (m = M) 16 ≤ M ≤511
5-0	P	R/W	0x03	PLL P value (p = P) 1 ≤ P ≤ 63

The PLL3/PLL4/PLL5 has the following constraints.

F_{VCO} = (m * F_{IN}) / (p) ***: 330 MHz ~ 660 MHz (F_{IN} is XIN oscillator)***

$F_{PLL} = F_{VCO} / (2 \hat{} s)$: Max F_{PLL} is 330 MHz

Whenever P, M, and S value are changed, the EN bit needs to be 0 for Δt (>1us) and to be 1 again to restart the PLL with new setting values.



Δt > 1us
locking time > 400 cycle
1 cycle = $1/F_{REF} = 1/ (F_{IN}/p)$

CLKDIVC0 Register 0xB0500048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
P0E	-	P0DIV						P1E	-	P1DIV					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
P2E	-	P2DIV						P3TE	-	P3DIV					

Field	Name	RW	Reset	Description
31	P0E	R/W	0x1	PLL0 divider enable register
30	-	-	-	Undefined
29-24	P0DIV	R/W	0x1	PLL0 divisor value $F_{P0DIV} = F_{PLL0} / (P0DIV + 1)$ P0DIV should not be zero
23	P1E	R/W	0x1	PLL1 divider enable register
22	-	-	-	Undefined
21-16	P1DIV	R/W	0x2	PLL1 divisor value $F_{P1DIV} = F_{PLL1} / (P1DIV + 1)$ P1DIV should not be zero
15	P2E	R/W	0x1	PLL2 divider enable register
14	-	-	-	Undefined
13-8	P2DIV	R/W	0x3	PLL2 divisor value $F_{P2DIV} = F_{PLL2} / (P2DIV + 1)$ P2DIV should not be zero
7	P3E	R/W	0x1	PLL3 divider enable register
6	-	-	-	Undefined
5-0	P3DIV	R/W	0x3	PLL3 divisor value $F_{P3DIV} = F_{PLL3} / (P3DIV + 1)$ P3DIV should not be zero

CLKDIVC1 Register 0xB050004C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
XE	-	XDIV						XTE	-	XTDIV					

Field	Name	RW	Reset	Description
31-16	-	-	-	Undefined
15	XE	R/W	0x0	XIN divider enable register
14	-	-	-	Undefined
13-8	XDIV	R/W	0x0	XIN divisor value $F_{XDIV} = F_{XIN} / (XDIV + 1)$ XDIV should not be zero
7	XTE	R/W	0x0	XTIN divider enable register
6	-	-	-	Undefined
5-0	XTDIV	R/W	0x0	XTIN divisor value $F_{XTDIV} = F_{XTIN} / (XTDIV + 1)$ XTDIV should not be zero

CLKDIVC2 Register 0xB0500050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
P4E	-	P4DIV						P5E	-	P5DIV					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-															

Field	Name	RW	Reset	Description
31	P4E	R/W	0x1	PLL4 divider enable register
30	-	-	-	Undefined
29-24	P4DIV	R/W	0x1	PLL4 divisor value $F_{P4DIV} = F_{PLL4} / (P4DIV + 1)$ P4DIV should not be zero
23	P5E	R/W	0x1	PLL5 divider enable register
22	-	-	-	Undefined
21-16	P5DIV	R/W	0x2	PLL5 divisor value $F_{P5DIV} = F_{PLL5} / (P5DIV + 1)$ P5DIV should not be zero
15-0	-	-	-	Undefined

SWRESET Register 0xB050005C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					SWRST										

Field	Name	RW	Reset	Description
10	SWRST[10]	R/W	0	Software Reset for Display Sub Bus 0 for Not-reset, ‘1’ for Reset
9	SWRST[9]	R/W	0	Software Reset for Camera Bus 0 for Not-reset, ‘1’ for Reset
8	SWRST[8]	R/W	0	Software Reset for High Speed IO Bus 0 for Not-reset, ‘1’ for Reset
7	SWRST[7]	R/W	0	*Reserved*
6	SWRST[6]	R/W	0	Software Reset for Video Core 0 for Not-reset, ‘1’ for Reset
5	SWRST[5]	R/W	0	Software Reset for Video Bus 0 for Not-reset, ‘1’ for Reset
4	SWRST[4]	R/W	0	*Reserved*
3	SWRST[3]	R/W	0	Software Reset for Graphic Bus 0 for Not-reset, ‘1’ for Reset
2	SWRST[2]	R/W	0	*Reserved*
1	SWRST[1]	R/W	0	Software Reset for Display(DDI) Bus 0 for Not-reset, ‘1’ for Reset
0	SWRST[0]	R/W	0	*Reserved*

HCLKMASK_CFG 0xB0500060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					CKMASK_CFG[10:0]										

Field	Name	RW	Reset	Description
10	CKMASK_CFG [10]	R/W	0	Clock Mask for Display Sub Bus CFG block 0 for Clock Enable, ‘1’ for Clock Mask
9	CKMASK_CFGT[9]	R/W	0	Clock Mask for Camera Bus CFG block 0 for Clock Enable, ‘1’ for Clock Mask
8	CKMASK_CFG [8]	R/W	0	Clock Mask for High Speed IO Bus CFG block 0 for Clock Enable, ‘1’ for Clock Mask
7	CKMASK_CFG [7]	R/W	0	Clock Mask for SMU CFG block 0 for Clock Enable, ‘1’ for Clock Mask
6	CKMASK_CFG [6]	R/W	0	Clock Mask for Video Core CFG block 0 for Clock Enable, ‘1’ for Clock Mask
5	CKMASK_CFG [5]	R/W	0	Clock Mask for Video Bus CFG block 0 for Clock Enable, ‘1’ for Clock Mask
4	CKMASK_CFG [4]	R/W	0	Clock Mask for IO Bus CFG block 0 for Clock Enable, ‘1’ for Clock Mask
3	CKMASK_CFG [3]	R/W	0	Software Reset for Graphic Bus CFG block 0 for Clock Enable, ‘1’ for Clock Mask
2	CKMASK_CFG [2]	R/W	0	Clock Mask for Memory Bus CFG block 0 for Clock Enable, ‘1’ for Clock Mask
1	CKMASK_CFG[1]	R/W	0	Clock Mask for DDI Bus CFG block 0 for Clock Enable, ‘1’ for Clock Mask
0	CKMASK_CFG[0]	R/W	0	*Reserved*

SWRESET_CFG 0xB0500064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					SWRST_CFG[10:0]										

Field	Name	RW	Reset	Description
10	SWRST _CFG [10]	R/W	0	Software Reset for Display Sub Bus CFG block 0 for Not-reset, ‘1’ for Reset
9	SWRST _CFGT[9]	R/W	0	Software Reset for Camera Bus CFG block 0 for Not-reset, ‘1’ for Reset
8	SWRST _CFG [8]	R/W	0	Software Reset for High Speed IO Bus CFG block 0 for Not-reset, ‘1’ for Reset
7	SWRST _CFG [7]	R/W	0	Software Reset for SMU CFG block 0 for Not-reset, ‘1’ for Reset
6	SWRST _CFG [6]	R/W	0	Software Reset for Video Core CFG block 0 for Not-reset, ‘1’ for Reset
5	SWRST _CFG [5]	R/W	0	Software Reset for Video Bus CFG block 0 for Not-reset, ‘1’ for Reset
4	SWRST _CFG [4]	R/W	0	Software Reset for IO Bus CFG block 0 for Not-reset, ‘1’ for Reset
3	SWRST _CFG [3]	R/W	0	Software Reset for Graphic Bus CFG block 0 for Not-reset, ‘1’ for Reset
2	SWRST _CFG [2]	R/W	0	Software Reset for Memory Bus CFG block 0 for Not-reset, ‘1’ for Reset
1	SWRST _CFG[1]	R/W	0	Software Reset for DDI Bus CFG block 0 for Not-reset, ‘1’ for Reset
0	SWRST_CFG[0]	R/W	0	*Reserved*

PCK_XXX Register **0xB0500080 + 4 * n (n=0~45, n != 27,28,31,36)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		EN	SEL					-							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-				DIV											

Field	Name	RW	Reset	Description
31-30	-	-	-	Undefined
29	EN	RW	0x1	Clock divider enable register **** If you want to change, the value should be same level over 2 clock cycles for F_{CKS}.***
28-24	SEL	RW	0x4	Source clock selection register Refer to Clock Source Table
23-12	-	-	-	Undefined
11-0	DIV	RW	0x0	Clock divisor for each hardware ***$F_{PCK\_xxx} = F_{CKS} / (DIV + 1)$*** F_{CKS} is selected clock by SEL In case of DIV being 0, the $F_{PCK\_xxx} = F_{CKS}$, F_{CKS} should be under 500MHz

Clock Source Table			
0	PLL0 direct output	1	PLL1 direct output
2	PLL2 direct output	3	PLL3 direct output
4	XIN direct	5	PLL0 divider output
6	PLL1 divider output	7	PLL2 divider output
8	PLL3 divider output	9	XTIN direct output
10	External clock[1]	11	HDMI_TMDS
12	HDMI_PCLK	13	HDMI oscillator clock (27MHz)
14	-	15	USB0 PHY clock output (48MHz)
16	XIN divider output	17	XTIN divider output
18	PLL4 direct output	19	PLL5 direct output
20	PLL4 divider output	21	PLL5 divider output
22	USB1 PHY clock output (48MHz)	23	-

[1] GPIOA[13] can be configured as the external clock function.

PCK_YYY Register **0xB0500080 + 4 * n, n = 27,31,36**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MD		EN	SEL					-							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIV															

Field	Name	RW	Reset	Description
31	MD	R/W	0x0	Mode selection register 1 : DIVIDER mode 0 : DCO mode
30	-	-	-	Undefined
29	EN	R/W	0x1	Clock divider enable register **** If you want to change, the value should be same level over 2 clock cycles for F_{CKS}.***
28-24	SEL	R/W	0xA	Source clock selection register Refer to Clock Source Table
23-16	-	-	-	Undefined
15-0	DIV	R/W	0x0	Clock divisor for each hardware In Divider Mode, $F_{PCK\_YYY} = F_{CKS} / (DIV + 1)$ In DCO Mode, ***DIV > 32768*** $F_{PCK\_DAI} = F_{CKS} * ((65536-DIV) / (65536))$ ***DIV <= 32768*** $F_{PCK\_DAI} = F_{CKS} * (DIV / (65536))$ FCKS is selected clock by SEL DIV should not be “ZERO”

* If you want the peripheral clock generator to be disabled, you should select the lowest frequency clock source for reducing the power consumption.

Clock Source Table			
0	PLL0 direct output	1	PLL1 direct output
2	PLL2 direct output	3	PLL3 direct output
4	XIN direct	5	PLL0 divider output
6	PLL1 divider output	7	PLL2 divider output
8	PLL3 divider output	9	XTIN direct output
10	External clock[2]	11	HDMI_TMDS
12	HDMI_PCLK	13	HDMI oscillator clock (27MHz)
14	-	15	USB0 PHY clock output (48MHz)
16	XIN divider output	17	XTIN divider output
18	PLL4 direct output	19	PLL5 direct output
20	PLL4 divider output	21	PLL5 divider output
22	USB1 PHY clock output (48MHz)	23	-

[2] GPIOA[13] can be configured as the external clock function.

1.7 Operation & Timing Diagram

1.7.1 Clock Change Operation (Safe Clock Changer)

An example of the timing diagram for changing the clock is shown below.

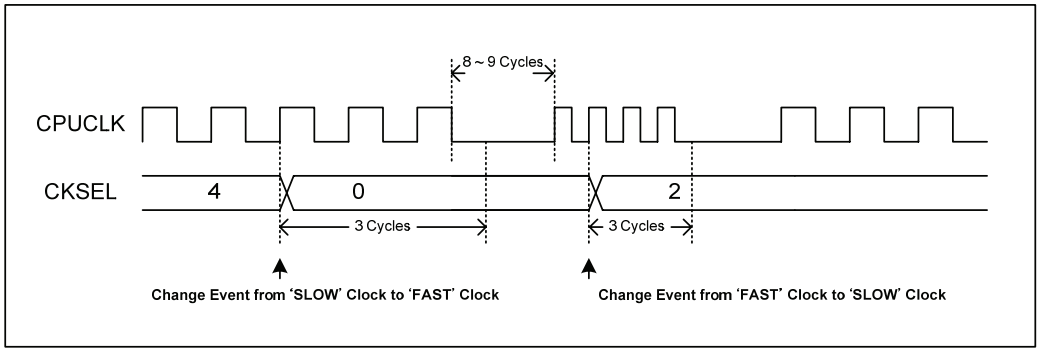


Figure 1.4 Clock Change Timing Diagram

An example shows two changing sequences. The first sequence is for changing from slower to faster frequency.

When the CPU write the 'SEL' register bits in the CLKCTRL0 ~ CLKCTRL10 with specified value ('0' in this example), the glitch-free circuit in the CKC hardware first stops the current clock after 3 clock cycles. And then in the 3 clock period, the clock multiplexor changes from the current clock to the next clock. Finally, the wake-up circuits enables the clock output. The changing sequence from 'faster' to 'slower' clocks is same procedure.

1.7.2 How to Generate "F_{CPU} " through BCLKOUT[0]

The following figure shows the clock chain from source to target which is BCLKOUT[0].
The clock selected by safe clock changer is "XIN" at the reset state.
Before changing the clock, the source and target clock should be alived. For example, assuming that the current clock is PLL1 and target clock is PLL0, the PLL1 should not be disabled and the PLL0 should be enabled before changing.

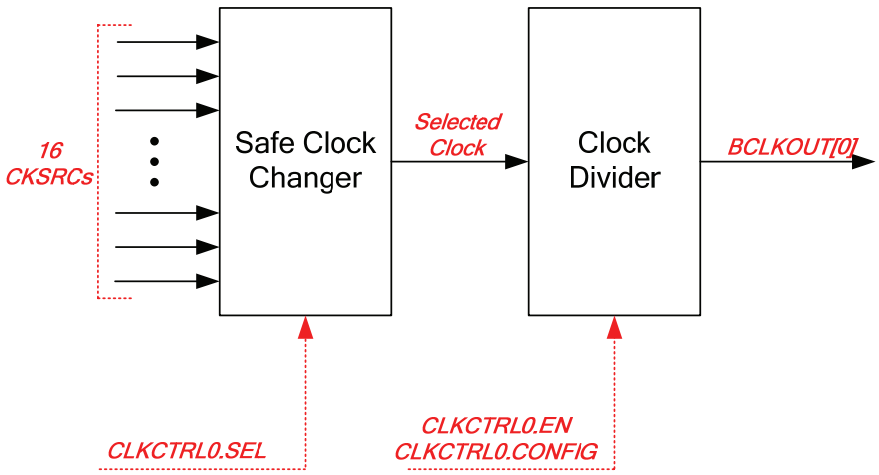


Figure 1.5 Clock Chain for BCLKOUT[0]

After selection of the source clock, you can change the divider configuration "CLKCTRL0.CONFIG" to make fast or slow for the target clock.

The following figure shows the waveform of the BCLKOUT[0] configured by CLKCTRL0.CONFIG.

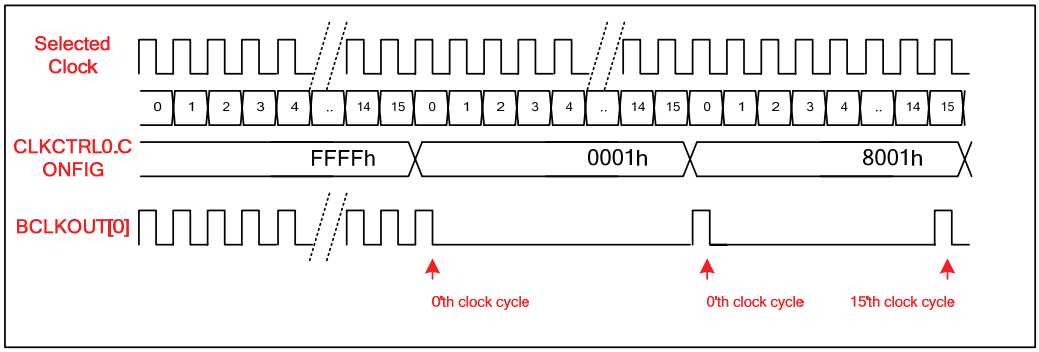


Figure 1.6 Clock Change Timing Diagram for BCLKOUT[0]

If the CLKCTRL0.CONFIG value is “FFFFh”, the output clock frequency is same as the input clock. But, in case of “0001h”, the valid clock cycle is only one which is first clock cycle and makes the frequency is 16 times slower than source clock. In the case of “8001h”, the output clock frequency is 8 times slower than source clock and the valid clocks are first and last clock cycle. ***the CLKCTRL0.CONFIG value should be “FFFFh” in synchrmode clock mode.***

1.7.3 How to Generate BCLKOUT[1] ~ BCLKOUT[10]

The following figure shows the clock chain from source to target which is BCLKOUT[1] ~ BCLKOUT[10].
The clock selected by safe clock changer is “XIN” at the reset state.
Before changing the clock, the source and target clock should be alived. For example, assuming that the current clock is PLL1 and target clock is PLL0, the PLL1 should not be disabled and the PLL0 should be enabled before changing.

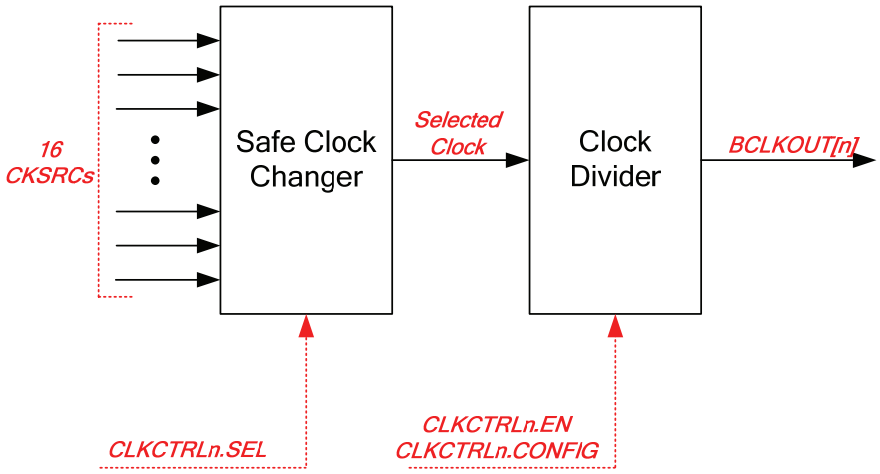


Figure 1.7 Clock Chain for BCLKOUT[10:1]

After selection of the source clock, you can change the divider configuration “CLKCTRLn.CONFIG” to make fast or slow for the target clock.
The following figure shows the waveform of the BCLKOUT[n] configured by CLKCTRLn.CONFIG.

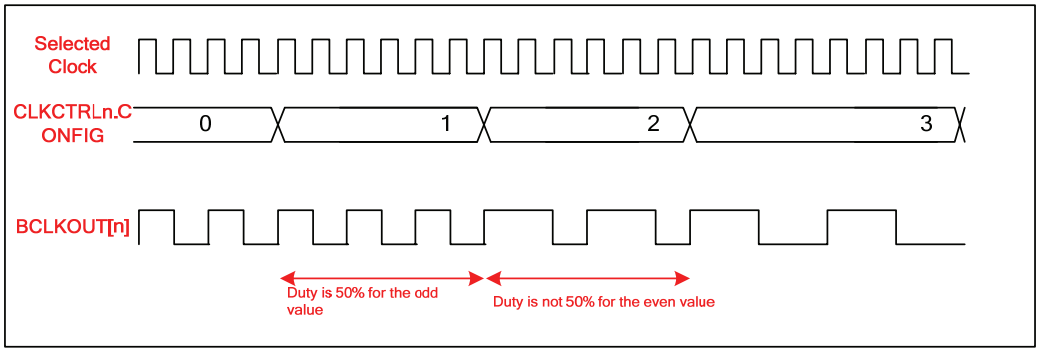


Figure 1.8 Clock Change Timing Diagram for BCLKOUT[7:1]

The divisor is the “config value + 1”. And for the odd value(even divisor), the duty of the output clock is 50%. But, the even value(odd divisor) does not guarantee the 50% duty. The larger divisor, the closer to 50% duty. The duty of the BCLKOUT[2] for memory interface should be 50%, so the value for the CLKCTRL2.CONFIG should be odd which means even divisor such as 1/2, 1/4, 1/8 and etc.

1.7.4 How to Generate PCLKs[45:0]

The following figure shows the clock chain from source to target which is PCLKs[0] ~ PCLKs[36].
The input clock sources are 16 which are different from PCK_XXX type1, PCK_XXX type2 and PCK_YYY.
The clock selected by safe clock changer is “XIN” at the reset state.

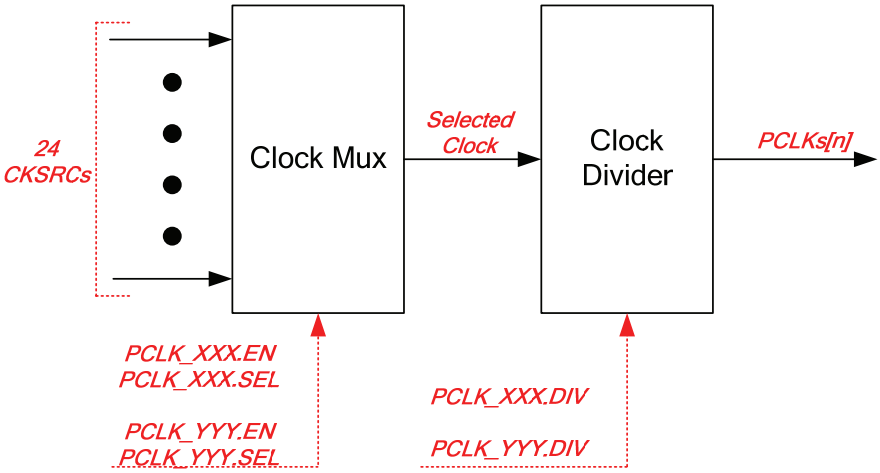


Figure 1.9 Clock Chain for PCLKs[45:0]

After selection of the source clock, you can change the divider configuration with “PCLK_XXX.DIV” or “PCLK_YYY.DIV” to make fast or slow for the target clock.

The following figure shows the waveform of the PCLKs[n] configured.

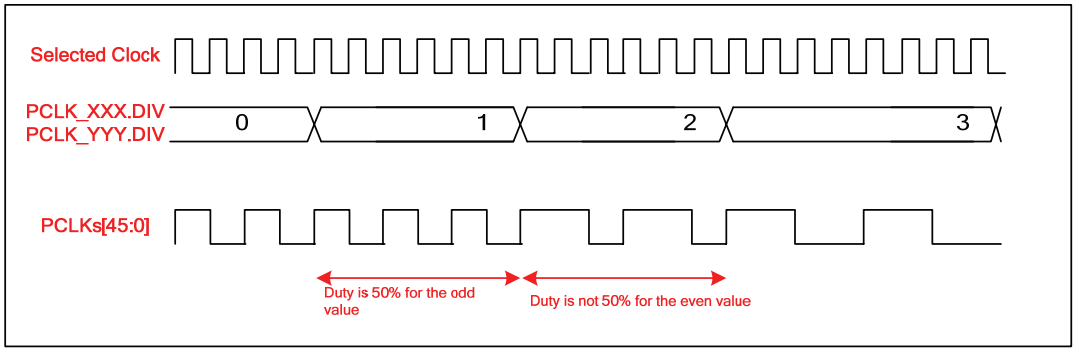


Figure 1.10 Clock Change Timing Diagram for PCLKs[45:0]

The divisor is the “config value + 1”. And for the odd value(even divisor), the duty of the output clock is 50%. But, the even value(odd divisor) does not guarantee the 50% duty. The larger divisor, the closer to 50% duty.

1.7.5 Procedure for Configuration of Peripheral Clocks

The Figure 1.11 shows the configuration procedure for peripheral clock.
The programmer should take care of some cautions in configuring the peripheral clock for corresponding hardware. (ex, MS, DAI, etc.)
The controller should be in reset state by setting the SWRESET bit before configuring the hardware clock and configuring the corresponding ports. If you configure the clock and ports in not-reset state, which is not-initialized state, the unexpected operations can occur.
The hardware starts operating by clearing the SWRESET bit after the configuration for the clock and ports of corresponding controller.
If you want to close the hardware operation, you should make the controller into reset-state by SWRESET bit and stop the clock and release the port.
If the controller is in reset-state (SWRESET='1'), the program can't access the control register for corresponding hardware.

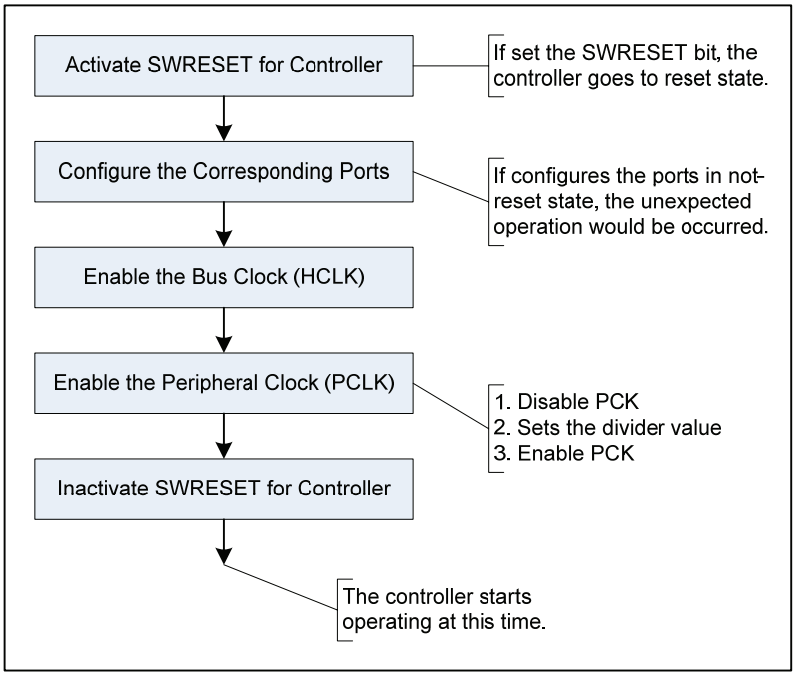


Figure 1.11 Peripheral Clock Configuration Procedure

1.7.6 Synchronous/Asynchronous Memory Bus Clock Configuration

1.7.6.1 Overview

CPU clock and memory bus clock of NVS2310 can be configured programmably as synchronous or asynchrous relation each other. In synchronous mode, memory bus clock frequency F_{MBUS} is always F_{CPU} /k (k is a programmable even number) and memory access latency from CPU is minimized. In asynchronous mode, memory bus clock frequency F_{MBUS} is not related with CPU clock frequency F_{CPU} and memory access latency from CPU is increased compared to one in synchronous mode. However, because there is no frequency dependency between CPU clock and memory bus clock in asynchronous mode, more flexible clock management is possible as an example for low power application.
Therefore, if the maximum CPU performance is required (especially when heavy external memory access exists), synchronous clock mode is recommended. On the other hand, if low power operation or asymmetrical clock frequency relationship between CPU and memory bus clock is required, asynchronous clock mode is recommended.
Clock mode transition between synchronous and asynchronous clock mode is possible by software control through pre-defined transition procesure. Default clock mode of NVS2310 at reset state is the synchronous mode.

1.7.6.2 Procedure for Clock Mode Transition

Clock mode transition between the synchronous and asynchronous clock mode is conducted by a simple software control using an interrupt and the defined register settings.

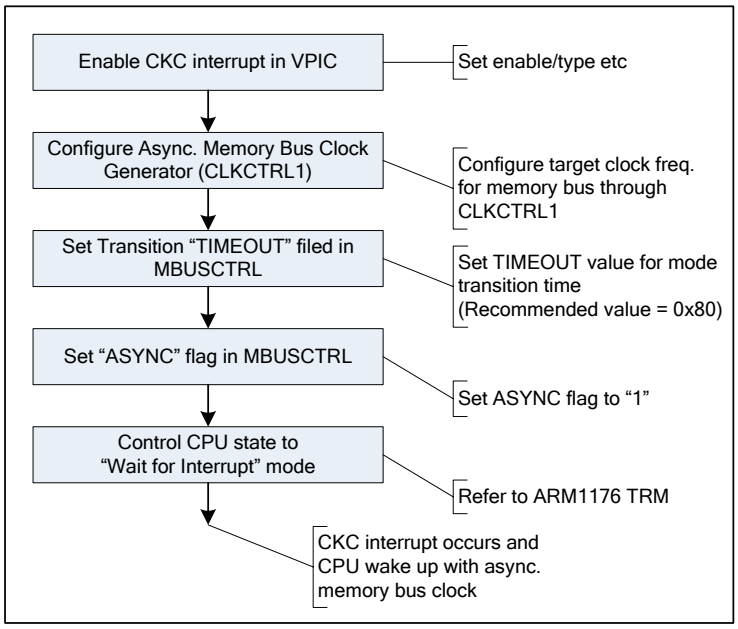


Figure 1.12 Clock Mode Transition Procedure (Sync. Mode to Async Mode)

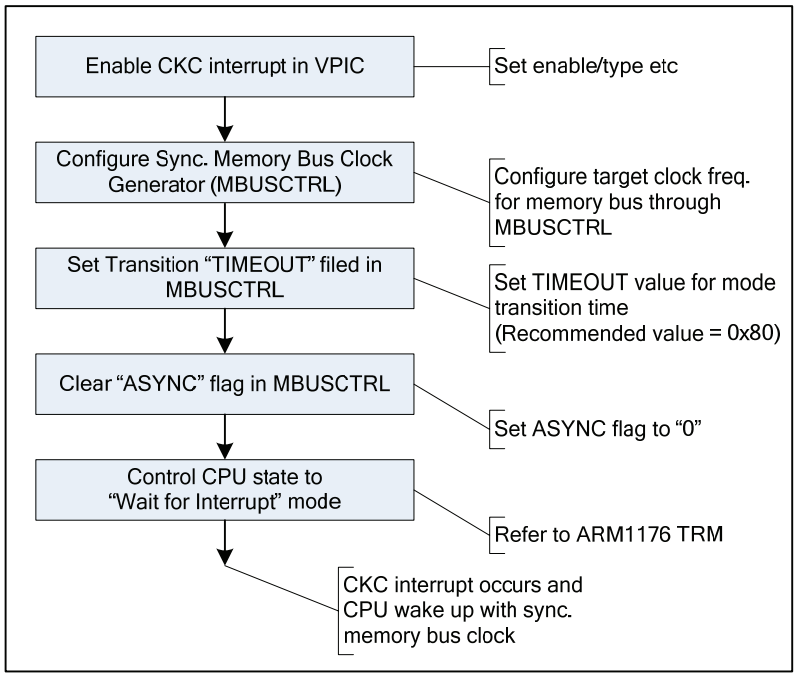
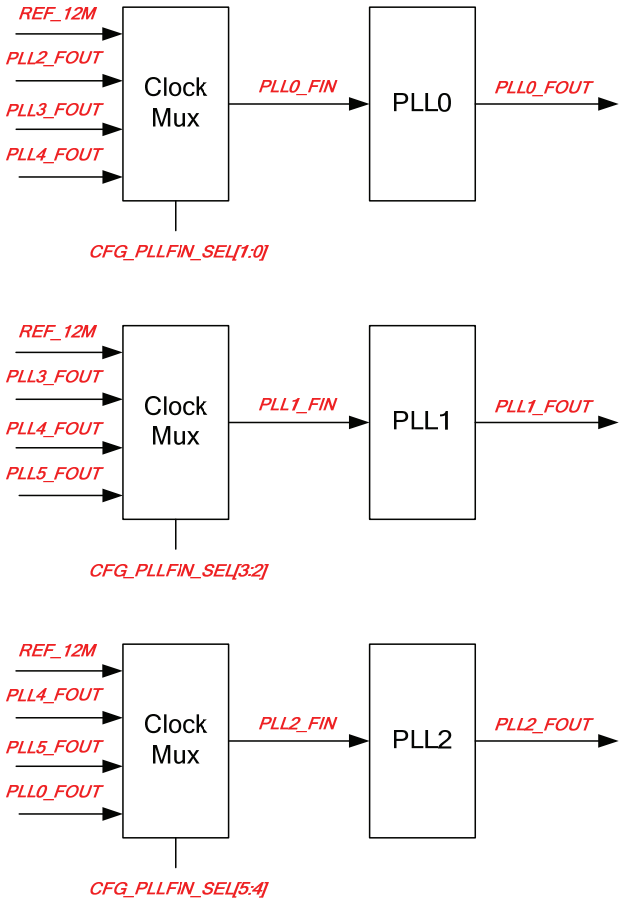


Figure 1.13 Clock Mode Transition Procedure (ASync. Mode to Sync Mode)

1.7.7 PLL Input Source Selection

The input reference clock of PLL0/PLL1/PLL2/PLL3/PLL4 can be selectable from 4 different clock sources as below figure. The input clock of PLL5 is fixed to REF_12M.



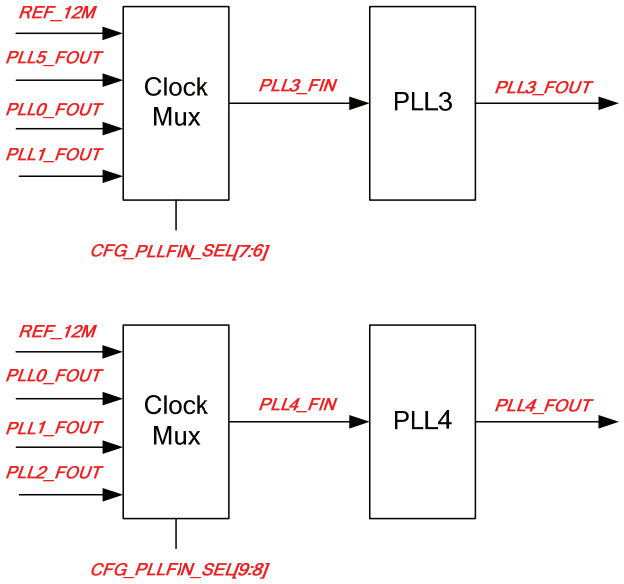


Figure 1.14 Input Clock Selection of PLL (CFG_PLLFIN_SEL[9:0] is from SMU CFG module)

1.7.8 Reference PMS Table for Target Frequency

PLLs embedded in the NVS2310 have the following constraints.

FVCO = M * (FIN / P), FPLL = FVCO / (2 ^ S)

Where FIN is the input(XIN) frequency of oscillator, FPLL is final output of PLL.

About FVCO, it is restriced to 1000MHz ~ 2000MHz in PLL0/1/2 and 330MHz ~ 660MHz in PLL3/4/5. As a result, FPLL is restriced to 32MHz ~ 2000MHz in PLL0/1/2 and 12MHz ~ 660MHz in PLL3/4/5.

You must choose the proper divider values from the following table. If you want to use other divider value than that in following, consult with nextchip. To set the selected divider value, refer to PLLnCFG.

Table 1.5 PLL0/1/2 Configuration Examples

FIN (MHz)	P(1~63)	M	S(0~5)	VSEL	FPLL(MHz)
12	3	280	4	0	70
12	3	288	4	0	72
12	3	344	4	0	86
12	3	404	4	1	101
12	3	264	3	0	132
12	3	292	3	0	146
12	3	360	3	1	180
12	3	368	3	1	184
12	3	380	3	1	190
12	3	384	3	1	192
12	3	408	3	1	204
12	3	416	3	1	208
12	3	420	3	1	210
12	3	428	3	1	214
12	3	432	3	1	216
12	3	444	3	1	222
12	3	452	3	1	226
12	3	456	3	1	228
12	3	460	3	1	230

FIN (MHz)	P(1~63)	M	S(0~5)	VSEL	FPLL(MHz)
12	3	468	3	1	234
12	3	472	3	1	236
12	3	488	3	1	244
12	3	492	3	1	246
12	3	496	3	1	248
12	3	252	2	0	252
12	3	260	2	0	260
12	3	262	2	0	262
12	3	264	2	0	264
12	3	266	2	0	266
12	3	276	2	0	276
12	3	282	2	0	282
12	3	288	2	0	288
12	3	290	2	0	290
12	3	294	2	0	294
12	3	304	2	0	304
12	3	308	2	0	308
12	3	312	2	0	312
12	3	316	2	0	316
12	3	320	2	0	320
12	3	324	2	0	324
12	3	312	2	0	312
12	3	316	2	0	316
12	3	320	2	0	320
12	3	324	2	0	324
12	3	328	2	0	328
12	3	330	2	0	330
12	3	332	2	0	332
12	3	336	2	0	336
12	3	340	2	0	340
12	3	348	2	0	348
12	3	352	2	1	352
12	3	356	2	1	356
12	3	360	2	1	360
12	3	362	2	1	362
12	3	364	2	1	364
12	3	369	2	1	369
12	3	370	2	1	370
12	3	372	2	1	372
12	3	380	2	1	380
12	3	381	2	1	381
12	3	384	2	1	384
12	3	388	2	1	388
12	3	390	2	1	390
12	3	392	2	1	392
12	3	393	2	1	393
12	3	396	2	1	396
12	3	399	2	1	399
12	3	400	2	1	400
12	3	404	2	1	404
12	4	541	2	1	405.75

FIN (MHz)	P(1~63)	M	S(0~5)	VSEL	FPLL(MHz)
12	3	406	2	1	406
12	3	408	2	1	408
12	3	410	2	1	410
12	3	412	2	1	412
12	3	416	2	1	416
12	4	557	2	1	417.75
12	3	420	2	1	420
12	3	424	2	1	424
12	3	428	2	1	428
12	3	429	2	1	429
12	3	430	2	1	430
12	3	432	2	1	432
12	3	436	2	1	436
12	3	440	2	1	440
12	3	442	2	1	442
12	3	444	2	1	444
12	3	448	2	1	448
12	3	450	2	1	450
12	3	452	2	1	452
12	3	454	2	1	454
12	3	456	2	1	456
12	3	460	2	1	460
12	3	463	2	1	463
12	3	464	2	1	464
12	3	466	2	1	466
12	3	468	2	1	468
12	3	470	2	1	470
12	3	472	2	1	472
12	3	474	2	1	474
12	3	476	2	1	476
12	3	480	2	1	480
12	3	484	2	1	484
12	3	486	2	1	486
12	3	488	2	1	488
12	3	490	2	1	490
12	3	492	2	1	492
12	3	496	2	1	496
12	3	497	2	1	497
12	3	498	2	1	498
12	3	500	2	1	500
12	3	252	1	0	504
12	3	254	1	0	508
12	3	256	1	0	512
12	3	258	1	0	516
12	3	260	1	0	520
12	3	262	1	0	524
12	3	264	1	0	528
12	3	266	1	0	532
12	3	268	1	0	536
12	3	270	1	0	540
12	3	272	1	0	544
12	3	274	1	0	548

FIN (MHz)	P(1~63)	M	S(0~5)	VSEL	FPLL(MHz)
12	3	276	1	0	552
12	3	278	1	0	556
12	3	280	1	0	560
12	3	282	1	0	564
12	3	284	1	0	568
12	3	286	1	0	572
12	3	288	1	0	576
12	3	290	1	0	580
12	3	292	1	0	584
12	3	294	1	0	588
12	3	296	1	0	592
12	3	298	1	0	596
12	3	300	1	0	600
12	3	302	1	0	604
12	3	304	1	0	608
12	3	306	1	0	612
12	3	308	1	0	616
12	3	310	1	0	620
12	3	312	1	0	624
12	3	314	1	0	628
12	3	316	1	0	632
12	3	318	1	0	636
12	3	320	1	0	640
12	3	322	1	0	644
12	3	324	1	0	648
12	3	326	1	0	652
12	3	328	1	0	656
12	3	330	1	0	660
12	3	332	1	0	664
12	3	334	1	0	668
12	3	336	1	0	672
12	3	338	1	0	676
12	3	340	1	0	680
12	3	342	1	0	684
12	3	344	1	0	688
12	3	346	1	0	692
12	3	348	1	0	696
12	3	350	1	0	700
12	3	352	1	1	704
12	3	354	1	1	708
12	3	356	1	1	712
12	3	358	1	1	716
12	3	360	1	1	720
12	3	362	1	1	724
12	3	364	1	1	728
12	3	366	1	1	732
12	3	368	1	1	736
12	3	370	1	1	740
12	3	372	1	1	744
12	3	374	1	1	748
12	3	376	1	1	752

FIN (MHz)	P(1~63)	M	S(0~5)	VSEL	FPLL(MHz)
12	3	378	1	1	756
12	3	380	1	1	760
12	3	382	1	1	764
12	3	384	1	1	768
12	3	386	1	1	772
12	3	388	1	1	776
12	3	390	1	1	780
12	3	392	1	1	784
12	3	394	1	1	788
12	3	396	1	1	792
12	3	398	1	1	796
12	3	400	1	1	800
12	3	402	1	1	804
12	3	404	1	1	808
12	3	406	1	1	812
12	3	408	1	1	816
12	3	410	1	1	820
12	3	412	1	1	824
12	3	414	1	1	828
12	3	416	1	1	832
12	3	418	1	1	836
12	3	420	1	1	840
12	3	422	1	1	844
12	3	424	1	1	848
12	3	426	1	1	852
12	3	428	1	1	856
12	3	430	1	1	860
12	3	432	1	1	864
12	3	434	1	1	868
12	3	436	1	1	872
12	3	438	1	1	876
12	3	440	1	1	880
12	3	442	1	1	884
12	3	444	1	1	888
12	3	446	1	1	892
12	3	448	1	1	896
12	3	450	1	1	900
12	3	454	1	1	908
12	3	458	1	1	916
12	3	462	1	1	924
12	3	466	1	1	932
12	3	470	1	1	940
12	3	474	1	1	948
12	3	478	1	1	956
12	3	482	1	1	964
12	3	486	1	1	972
12	3	490	1	1	980
12	3	494	1	1	988
12	3	498	1	1	996
12	3	251	0	0	1004
12	3	253	0	0	1012
12	3	255	0	0	1020

FIN (MHz)	P(1~63)	M	S(0~5)	VSEL	FPLL(MHz)
12	3	257	0	0	1028
12	3	259	0	0	1036
12	3	261	0	0	1044
12	3	263	0	0	1052
12	3	265	0	0	1060
12	3	267	0	0	1068
12	3	269	0	0	1076
12	3	271	0	0	1084
12	3	273	0	0	1092
12	3	275	0	0	1100
12	3	277	0	0	1108
12	3	279	0	0	1116
12	3	281	0	0	1124
12	3	283	0	0	1132
12	3	285	0	0	1140
12	3	287	0	0	1148
12	3	289	0	0	1156
12	3	291	0	0	1164
12	3	293	0	0	1172
12	3	295	0	0	1180
12	3	297	0	0	1188
12	3	299	0	0	1196
12	3	300	0	0	1200
12	3	301	0	0	1204
12	3	303	0	0	1212
12	3	305	0	0	1220
12	3	307	0	0	1228
12	3	309	0	0	1236
12	3	311	0	0	1244
12	3	313	0	0	1252
12	3	315	0	0	1260
12	3	317	0	0	1268
12	3	319	0	0	1276
12	3	321	0	0	1284
12	3	323	0	0	1292
12	3	325	0	0	1300
12	3	327	0	0	1308
12	3	329	0	0	1316
12	3	331	0	0	1324
12	3	333	0	0	1332
12	3	335	0	0	1340
12	3	337	0	0	1348
12	3	339	0	0	1356
12	3	341	0	0	1364
12	3	343	0	0	1372
12	3	345	0	0	1380
12	3	347	0	0	1388
12	3	349	0	0	1396
12	3	351	0	1	1404
12	3	353	0	1	1412
12	3	355	0	1	1420

FIN (MHz)	P(1~63)	M	S(0~5)	VSEL	FPLL(MHz)
12	3	357	0	1	1428
12	3	359	0	1	1436
12	3	361	0	1	1444
12	3	363	0	1	1452
12	3	365	0	1	1460
12	3	367	0	1	1468
12	3	369	0	1	1476
12	3	371	0	1	1484
12	3	373	0	1	1492
12	3	375	0	1	1500
12	3	377	0	1	1508
12	3	379	0	1	1516
12	3	381	0	1	1524
12	3	383	0	1	1532
12	3	385	0	1	1540
12	3	387	0	1	1548
12	3	389	0	1	1556
12	3	391	0	1	1564
12	3	393	0	1	1572
12	3	395	0	1	1580
12	3	397	0	1	1588
12	3	399	0	1	1596
12	3	401	0	1	1604
12	3	403	0	1	1612
12	3	405	0	1	1620
12	3	407	0	1	1628
12	3	409	0	1	1636
12	3	411	0	1	1644
12	3	413	0	1	1652
12	3	415	0	1	1660
12	3	417	0	1	1668
12	3	419	0	1	1676
12	3	421	0	1	1684
12	3	423	0	1	1692
12	3	425	0	1	1700
12	3	427	0	1	1708
12	3	429	0	1	1716
12	3	431	0	1	1724
12	3	433	0	1	1732
12	3	435	0	1	1740
12	3	437	0	1	1748
12	3	439	0	1	1756
12	3	441	0	1	1764
12	3	443	0	1	1772
12	3	445	0	1	1780
12	3	447	0	1	1788
12	3	449	0	1	1796
12	3	451	0	1	1804
12	3	453	0	1	1812
12	3	455	0	1	1820
12	3	457	0	1	1828
12	3	459	0	1	1836

FIN (MHz)	P(1~63)	M	S(0~5)	VSEL	FPLL(MHz)
12	3	461	0	1	1844
12	3	463	0	1	1852
12	3	465	0	1	1860
12	3	467	0	1	1868
12	3	469	0	1	1876
12	3	471	0	1	1884
12	3	473	0	1	1892
12	3	475	0	1	1900
12	3	477	0	1	1908
12	3	479	0	1	1916
12	3	481	0	1	1924
12	3	483	0	1	1932
12	3	485	0	1	1940
12	3	487	0	1	1948
12	3	489	0	1	1956
12	3	491	0	1	1964
12	3	493	0	1	1972
12	3	495	0	1	1980
12	3	497	0	1	1988
12	3	499	0	1	1996
12	3	500	0	1	2000

Table 1.6 PLL3/4/5 Configuration Examples

FIN (MHz)	P(1~63)	M	S(0~5)	FVCO(MHz)	FPLL(MHz)
12	3	128	5	1	16
12	3	99	4	0	24.75
12	3	140	3	1	70
12	3	144	3	1	72
12	3	86	2	0	86
12	3	101	2	0	101
12	3	132	2	1	132
12	3	146	2	1	146
12	3	90	1	0	180
12	3	92	1	0	184
12	3	95	1	0	190
12	3	96	1	0	192
12	3	102	1	0	204
12	3	104	1	0	208
12	3	105	1	0	210
12	3	107	1	0	214
12	3	108	1	0	216
12	3	111	1	0	222
12	3	113	1	0	226
12	3	114	1	0	228
12	3	115	1	0	230
12	3	117	1	1	234
12	3	118	1	1	236
12	3	122	1	1	244
12	3	123	1	1	246
12	3	124	1	1	248
12	3	126	1	1	252

FIN (MHz)	P(1~63)	M	S(0~5)	FVCO(MHz)	FPLL(MHz)
12	3	130	1	1	260
12	3	131	1	1	262
12	3	132	1	1	264
12	3	133	1	1	266
12	3	138	1	1	276
12	3	141	1	1	282
12	3	144	1	1	288
12	3	145	1	1	290
12	3	147	1	1	294
12	3	152	1	1	304
12	3	154	1	1	308
12	3	156	1	1	312
12	3	158	1	1	316
12	3	160	1	1	320
12	3	162	1	1	324
12	3	156	1	1	312
12	3	158	1	1	316
12	3	160	1	1	320
12	3	162	1	1	324
12	3	164	1	1	328
12	3	165	1	1	330
12	3	83	0	0	332
12	3	84	0	0	336
12	3	85	0	0	340
12	3	87	0	0	348
12	3	88	0	0	352
12	3	89	0	0	356
12	3	90	0	0	360
12	6	181	0	0	362
12	3	91	0	0	364
12	4	123	0	0	369
12	6	185	0	0	370
12	3	93	0	0	372
12	3	95	0	0	380
12	4	127	0	0	381
12	3	96	0	0	384
12	3	97	0	0	388
12	4	130	0	0	390
12	3	98	0	0	392
12	4	131	0	0	393
12	3	99	0	0	396
12	4	133	0	0	399
12	3	100	0	0	400
12	3	101	0	0	404
12	5	169	0	0	405.6
12	6	203	0	0	406
12	3	102	0	0	408
12	6	205	0	0	410
12	3	103	0	0	412
12	3	104	0	0	416
12	5	174	0	0	417.6
12	3	105	0	0	420

FIN (MHz)	P(1~63)	M	S(0~5)	FVCO(MHz)	FPLL(MHz)
12	3	106	0	0	424
12	3	107	0	0	428
12	4	143	0	0	429
12	6	215	0	0	430
12	3	108	0	0	432
12	3	109	0	0	436
12	3	110	0	0	440
12	6	221	0	0	442
12	3	111	0	0	444
12	3	112	0	0	448
12	4	150	0	0	450
12	3	113	0	0	452
12	6	227	0	0	454
12	3	114	0	0	456
12	3	115	0	0	460
12	5	193	0	1	463.2
12	3	116	0	1	464
12	6	233	0	1	466
12	3	117	0	1	468
12	6	235	0	1	470
12	3	118	0	1	472
12	4	158	0	1	474
12	3	119	0	1	476
12	3	120	0	1	480
12	3	121	0	1	484
12	4	162	0	1	486
12	3	122	0	1	488
12	6	245	0	1	490
12	3	123	0	1	492
12	3	124	0	1	496
12	5	207	0	1	496.8
12	4	166	0	1	498
12	3	125	0	1	500
12	3	126	0	1	504
12	3	127	0	1	508
12	3	128	0	1	512
12	3	129	0	1	516
12	3	130	0	1	520
12	3	131	0	1	524
12	3	132	0	1	528
12	3	133	0	1	532
12	3	134	0	1	536
12	3	135	0	1	540
12	3	136	0	1	544
12	3	137	0	1	548
12	3	138	0	1	552
12	3	139	0	1	556
12	3	140	0	1	560
12	3	141	0	1	564
12	3	142	0	1	568
12	3	143	0	1	572

FIN (MHz)	P(1~63)	M	S(0~5)	FVCO(MHz)	FPLL(MHz)
12	3	144	0	1	576
12	3	145	0	1	580
12	3	146	0	1	584
12	3	147	0	1	588
12	3	148	0	1	592
12	3	149	0	1	596
12	3	150	0	1	600
12	3	151	0	1	604
12	3	152	0	1	608
12	3	153	0	1	612
12	3	154	0	1	616
12	3	155	0	1	620
12	3	156	0	1	624
12	3	157	0	1	628
12	3	158	0	1	632
12	3	159	0	1	636
12	3	160	0	1	640
12	3	161	0	1	644
12	3	162	0	1	648
12	3	163	0	1	652
12	3	164	0	1	656
12	3	165	0	1	660

2 Timer / Counter

2.1 Overview

The NVS2310 has four 16-bits, two 20-bits, and one 32-bits timers/counters. 16-bits and 20-bits timer/counters have three registers for basic operation modes. Refer to register description table for details. When operating in counter modes, External interrupt pin is used as counting clock for that counter.

The main clock frequency of timer counter is determined by TCLK, XCLK, ZCLK frequency. With the 12bit internal basic counter, the timer counter can generate various intervals from microseconds to seconds unit.

The TCLK is for 16/20bits timer/couter named T-Timer, the XCLK is for watchdog timer named X-Timer, and finally ZCLK is for 32bits timer named Z-Timer. The following figure shows the overall timer/counter block diagram.

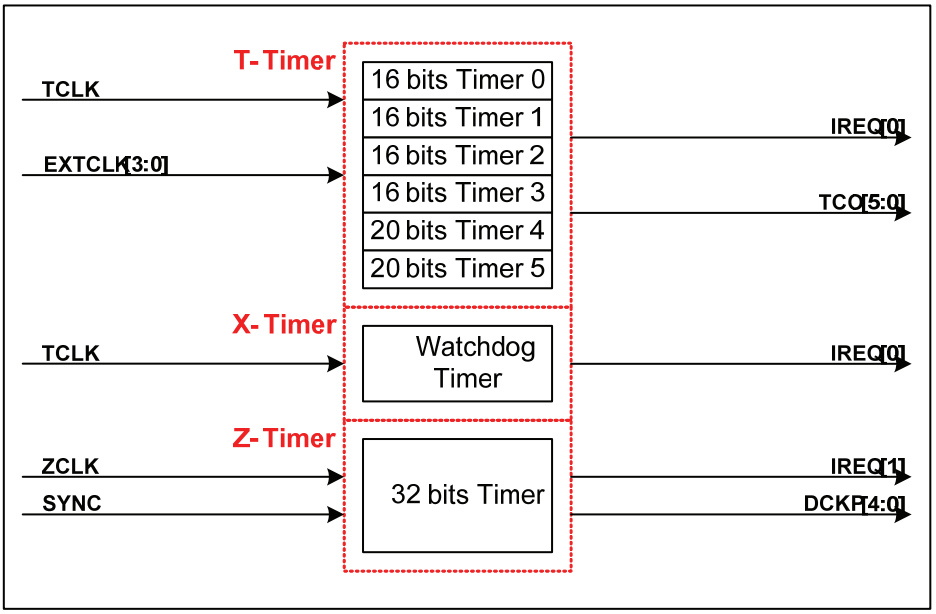


Figure 2.1 Overall Timer/Counter Block Diagram

The following figure represents the block diagram of T-Timer.

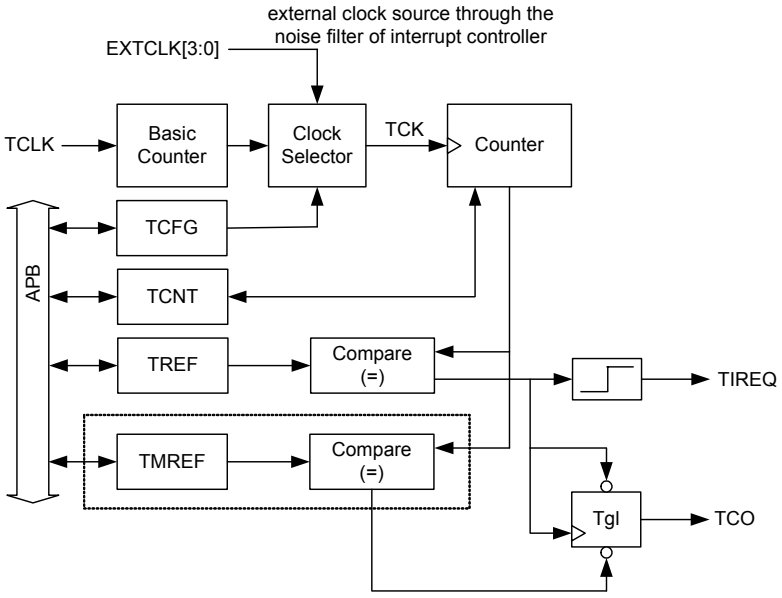


Figure 2.2 16-bit and 20bit Timer/Counter Block Diagram

The following figure shows block diagram of the watchdog timer.

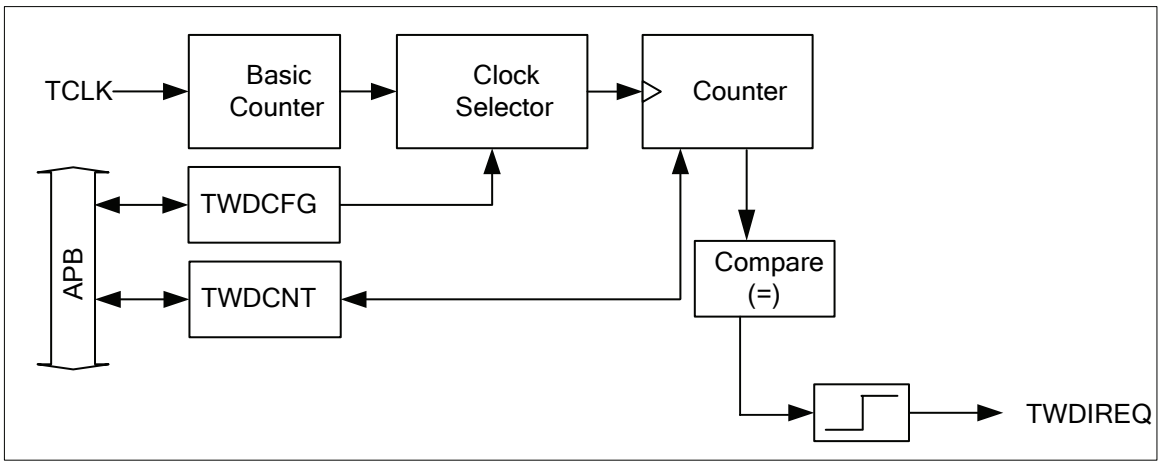


Figure 2.3 Watchdog Timer Block Diagram

As illustrated in the figure below, TC32 consists of a pre-scale counter, main counter and two comparators. The pre-scale counter is a simple 24-bit up-counter which always counts from zero to PRESCALE value programmed in TC32EN register. The 32-bit main counter is incremented only when the prescale counter reaches PRESCALE value.

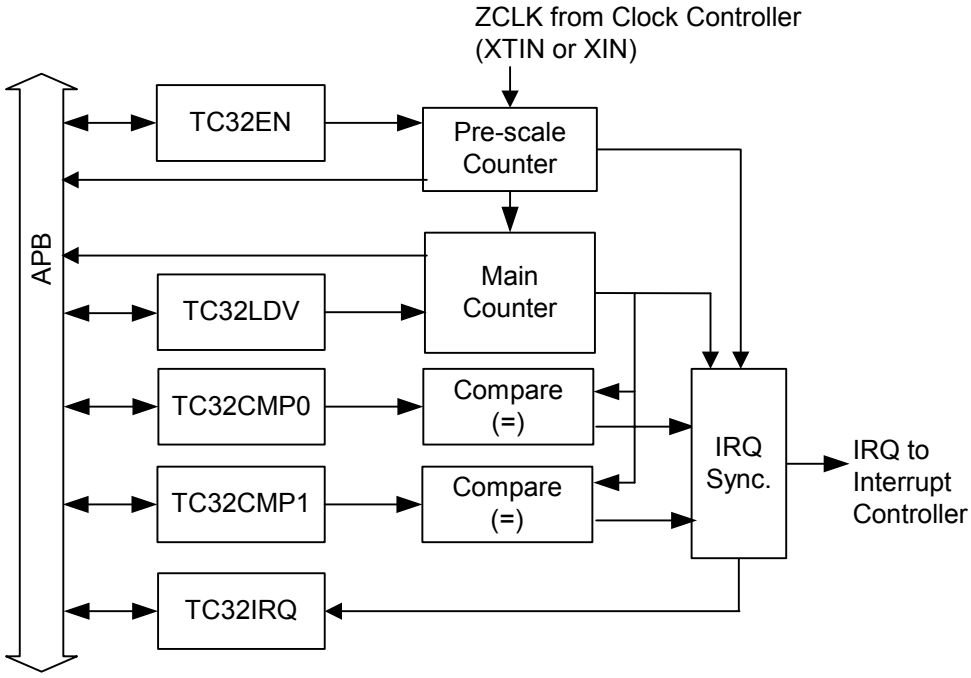


Figure 2.4 32-bit Counter Block Diagram

2.2 Signal Descriptions

2.2.1 Global Signals

Table 2.1 Global Signals

Name	Direction	Descriptions	Related Blocks
PCLK	Input	Timer Bus Clock (SMU Bus Clock)	CKC
PRESETn	Input	Reset Signal	CKC

2.2.2 T-Timer/Counter(16/20bits Timer/Counter) Related Signals

Table 2.2 T-Timer/Counter(16/20bits Timer/Counter) Related Signals

Name	Direction	Descriptions	Related Blocks
TCLK	Input	Timer Operating Clock	PCLKs[1] of CKC
EXTCLK[3:0]	Input	External Counter Clock Sources	IRQP[3:0] of VPIC
TCO[5:0]	Output	Timer Counter Outputs [0] : 16bits Timer 0 [1] : 16bits Timer 1 [2] : 16bits Timer 2 [3] : 16bits Timer 3 [4] : 20bits Timer 4 [5] : 20bits Timer 5	GPIO
IREQ[0]	Output	T-Timer Interrupt Output	IRQI[0] of VPIC

2.2.3 X-Timer (Watchdog Timer)

Table 2.3 X-Timer (32bits Timer)

Name	Direction	Descriptions	Related Blocks
TCLK	Input	Timer Operating Clock	PCLKs[0] of CKC
WDTRST	Output	Watchdog Reset but **Not Used**	-
IREQ[0]	Output	T-Timer Interrupt Output	IRQI[0] of VPIC

2.2.4 Z-Timer (32bits Timer)

Table 2.4 Interrupt Output Signals (to ARM)

Name	Direction	Descriptions	Related Blocks
ZCLK	Input	Timer Operating Clock	PCLKs[2] of CKC
DCKP[4:0]	Output	Pulse Outputs but **Not used**	-
SYNC	Input	Stuck at HIGH	-
IREQ[1]	Output	Z-Timer Interrupt Output	IRQI[1] of VPIC

2.3 Register Description

The following table explains the registers of each timer counter. The address of each timer counter is 16bytes aligned. The base address of timer counter is 0xB0501000.

The number n represents for each timer/counter. In case of timer/counter 4, 5 (that is n = 4 or 5) the TREF, TCNT register has 20bit resolution. It can be used for generating long time events.

Table 2.5 Timer/Counter Register Map (Base Address = 0xB0501000)

Name	Address	Type	Reset	Description
TCFG0	0x00	R/W	0x00	Timer/Counter 0 Configuration Register
TCNT0	0x04	R/W	0x0000	Timer/Counter 0 Counter Register
TREF0	0x08	R/W	0xFFFF	Timer/Counter 0 Reference Register
TMREF0	0x0C	R/W	0x0000	Timer/Counter 0 Middle Reference Register
TCFG1	0x10	R/W	0x00	Timer/Counter 1 Configuration Register
TCNT1	0x14	R/W	0x0000	Timer/Counter 1 Counter Register
TREF1	0x18	R/W	0xFFFF	Timer/Counter 1 Reference Register
TMREF1	0x1C	R/W	0x0000	Timer/Counter 1 Middle Reference Register
TCFG2	0x20	R/W	0x00	Timer/Counter 2 Configuration Register
TCNT2	0x24	R/W	0x0000	Timer/Counter 2 Counter Register
TREF2	0x28	R/W	0xFFFF	Timer/Counter 2 Reference Register
TMREF2	0x2C	R/W	0x0000	Timer/Counter 2 Middle Reference Register
TCFG3	0x30	R/W	0x00	Timer/Counter 3 Configuration Register
TCNT3	0x34	R/W	0x0000	Timer/Counter 3 Counter Register
TREF3	0x38	R/W	0xFFFF	Timer/Counter 3 Reference Register
TMREF3	0x3C	R/W	0x0000	Timer/Counter 3 Middle Reference Register
TCFG4	0x40	R/W	0x00	Timer/Counter 4 Configuration Register
TCNT4	0x44	R/W	0x00000	Timer/Counter 4 Counter Register
TREF4	0x48	R/W	0xFFFFF	Timer/Counter 4 Reference Register
TCFG5	0x50	R/W	0x00	Timer/Counter 5 Configuration Register
TCNT5	0x54	R/W	0x00000	Timer/Counter 5 Counter Register
TREF5	0x58	R/W	0xFFFFF	Timer/Counter 5 Reference Register
TIREQ	0x60	R/W	0x0000	Timer/Counter n Interrupt Request Register
TWDCFG	0x70	R/W	0x0000	Watchdog Timer Configuration Register
TWDCLR	0x74	W	-	Watchdog Timer Clear Register
TWDCNT	0x78	R/W	0xFFE0	Watchdog Timer Counter Register
TC32EN	0x80	R/W	0x00007FFF	32-bit Counter Enable / Pre-scale Value
TC32LDV	0x84	R/W	0x00000000	32-bit Counter Load Value
TC32CMP0	0x88	R/W	0x00000000	32-bit Counter Match Value 0
TC32CMP1	0x8C	R/W	0x00000000	32-bit Counter Match Value 1
TC32PCNT	0x90	R/W	-	32-bit Counter Current Value (pre-scale counter)
TC32MCNT	0x94	R/W	-	32-bit Counter Current Value (main counter)
TC32IRQ	0x98	R/W	0x0000----	32-bit Counter Interrupt Control

The registers prefixed by “TC32” are for the Z-Timer and the registers prefixed by “TWD” are for the X-Timer. Others are for T-Timers.

Timer/Counter n Configuration Register (TCFG*n*) 0xB05010*n*0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						STOP	CC	POL	TCKSEL			IEN	PWM	CON	EN

Field	Name	RW	Reset	Description
9	STOP	R/W	0x00	0 : Continuous counting mode. 1 : If TCNTn is equal to the TREFn, the TCNTn counter stop to increment.
8	CC	R/W	0x00	0 : TCNTn is not cleared. 1 : TCNTn is cleared to zero.
7	POL	R/W	0x00	0 : TCNTn is incremented at rising edge of the selected counting clock 1 : TCNTn is incremented at falling edge of the selected counting clock
6-4	TCKSEL	R/W	0x00	k = 0 ~ 4 : TCK is internally generated from divider circuit. It is driven by TCLK, and this value determines the division factor of this circuit. Division factor is $2^{(k+1)}$. k = 5, 6 : TCK is internally generated from divider circuit. It is driven by TCLK, and this value determines the division factor of this circuit. Division factor is 2^{2k} k = 7 : TCK is the external pin shared by external interrupt signal. In NVS2310, there are 4 external pins for this purpose, so this configuration is valid only for timer/counter 3 ~ 0. (not for timer/counter 5, 4)
3	IEN	R/W	0x00	0 : Disable Timer/Counter interrupt 1 : Enable Timer/Counter interrupt
2	PWM	R/W	0x00	0 : Disable PWM mode Timer/Counter output can be changed only when the TCNTn is equal to TREFn. It can be used to generate a rectangular pulse of variable frequency. 1 : Enable PWM mode Timer/Counter output is changed at every time the TCNTn is equal to TREFn and TMREFn value. It can be used to generate PWM waveform, by changing TMREFn while fixing TREFn. (where, TREFn > TMREFn)
1	CON	R/W	0x00	0 : When the TCNTn is reached to TREFn, TCNTn restarts counting from 0 at the next pulse of selected clock source. 1 : The TCNTn continues counting from the TREFn.
0	EN	R/W	0x00	0 : Timer counter is disabled. 1 : Timer counter is enabled.

Timer/Counter n Counting Register (TCNTn) 0xB05010n4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												TCNTn[19:16]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TCNTn[15:0]															

Field	Name	RW	Reset	Description
19-0	TCNTn	R/W	0x0000	TCNTn is increased by 1 at every pulse of selected clock source.

TCNTn can be set to any value by writing to this register. In case of timer 4 and timer 5, it has 20 bits, otherwise it has 16 bits.

Timer/Counter n Counting Reference Register (TREFn) 0xB05010n8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												TREFn[19:16]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TREFn[15:0]															

Field	Name	RW	Reset	Description
19-0	TREFn	R/W	0xFFFF	When TCNTn is reached at TREFn and the CON flag of TCFGn register is set to 1, the TCNTn is cleared to 0 at the next pulse of selected clock source. According to the TCFGn settings, various kinds of operations may be done. In case of timer 4 and timer 5, it has 20 bit, otherwise it has 16 bit.

Timer/Counter n Middle Reference Register (TMREFn) 0xB05010nC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMREFn[15:0]															

Field	Name	RW	Reset	Description
15-0	TMREFn	R/W	0x0000	When TCNTn is reached at TMREFn and the PWM flag of TCFGn register is set to 1, the timer output of TCOn is cleared to 0 at the negative edge of that pulse of selected clock source. The TCOn is set to 1 when the TCNTn is reached at TREFn. (refer TREFn). So you can generate PWM signal by modifying TMREFn(n=0~3) between 0 ~ (TREFn-1).

Timer/Counter Interrupt Request Register (TIREQ) 0xB0501060

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	TWF	TF5	TF4	TF3	TF2	TF1	TF0	0	TWI	TI5	TI4	TI3	TI2	TI1	TI0

Field	Name	RW	Reset	Description
14	TWF	R/W	0	Watchdog timer has reached to its reference value.
13	TF5	R/W	0	Timer/counter 5 has reached to its reference value.
12	TF4	R/W	0	Timer/counter 4 has reached to its reference value.
11	TF3	R/W	0	Timer/counter 3 has reached to its reference value.
10	TF2	R/W	0	Timer/counter 2 has reached to its reference value.
9	TF1	R/W	0	Timer/counter 1 has reached to its reference value.
8	TF0	R/W	0	Timer/counter 0 has reached to its reference value.
6	TWI	R/W	0	Read : Watchdog timer has generated its interrupt. Write : Watchdog timer interrupt is cleared.
5	TI5	R/W	0	Read :Timer/counter 5 has generated its interrupt Write : Timer/counter 5 interrupt flag is cleared.
4	TI4	R/W	0	Read :Timer/counter 4 has generated its interrupt Write : Timer/counter 4 interrupt flag is cleared.
3	TI3	R/W	0	Read :Timer/counter 3 has generated its interrupt Write : Timer/counter 3 interrupt flag is cleared.
2	TI2	R/W	0	Read :Timer/counter 2 has generated its interrupt Write : Timer/counter 2 interrupt flag is cleared.
1	TI1	R/W	0	Read :Timer/counter 1 has generated its interrupt Write : Timer/counter 1 interrupt flag is cleared.
0	TI0	R/W	0	Read :Timer/counter 0 has generated its interrupt Write : Timer/counter 0 interrupt flag is cleared.

If a timer n has reached its reference value, the TFn is set. (bit n represents for Timer n). If its interrupt request is enabled by set bit 3 of TCFGn register, then the TIn is set. If the TC bit of IEN register is set, the timer interrupt is really generated and this TIREQ register can be used to determine which timer has requested the interrupt. After checking these flags, user can clear these TFn and TIn field by writing “1” to corresponding TFn or TIn bit field.

Watchdog Timer Configuration Register (TWDCFG) 0xB0501070

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									TCLKSEL			IEN	0		EN

Field	Name	RW	Reset	Description
6-4	TCLKSEL	R/W	0x0000	k = 4: WDTCLK is internally generated from divider circuit. It is driven by TCLK, and this value determines the division factor of this circuit. Division factor is 25. k = 5, 6: WDTCLK is internally generated from divider circuit. It is driven by TCLK, and this value determines the division factor of this circuit. Division factor is 22. k = 0~3,7: Undefined. Should not be used.
3	IEN	R/W	0x0000	1 : Watchdog Timer Interrupt is enabled. This field is valid only if RST field is set to 0.
0	EN	R/W	0x0000	1 : Watchdog timer is enabled. If the watchdog timer is disabled, its counter goes to 0xE0, so when it is first enabled, user must clear the counter by writing to TWDCLR register.

Watchdog timer is used for the system not to be stuck by generating a reset pulse automatically when the watchdog timer counter overflows to zero. It has 8bit counter and when this counter overflows from 0xFF to 0x00, the reset or interrupt is generated.
The programmer must clear the watchdog counter before it overflows by writing any value to TWDCLR register. The duration can be chosen by selecting TCKSEL field appropriately.

Watchdog Timer Clear Register (TWDCLR) 0xB0501074

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Any Value															

Field	Name	RW	Reset	Description
15-0	TWDCLR	W	0x0000	The watchdog timer counter can be cleared to 0 by writing any value to this register.

The watchdog timer counter can be cleared to 0 by writing any value to this register. If it is not cleared before it overflows, the watchdog timer generate reset signal to the entire component of chip.

Watchdog Timer Counting Register (TWDCNT) 0xB0501078

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TWDCNT [15:0]															

Field	Name	RW	Reset	Description
15-0	TWDCNT	R/W	0xFFE0	TWDCNT is increased by 1 at every pulse of selected clock source.

TWDCNT can be set to any value by writing to this register.

TC32 Enable / Pre-scale Value Register (TC32EN) 0xB0501080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-		LDM1	LDM0	-	STOP MODE	LOAD ZERO	ENABLE	PRESCALE							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRESCALE															

Field	Name	RW	Reset	Description
31-30	Reserved	0	R	
29	LDM1	0	R/W	Re-load counter when the counter value is matched with CMP1. LOADZERO bit below selects the counter load(start) value.
28	LDM0	0	R/W	Re-load counter when the counter value is matched with CMP0 LOADZERO bit below selects the counter load(start) value.
27	Reserved	0	R	
26	STOPMODE	0	R/W	0 = Free Running Mode, 1 = Stop Mode.
25	LOADZERO	0	R/W	By default, counter starts from LOADVAL. When this bit is enabled (1), the counter is forced to count from “0” to “LOADVAL – 1”.
24	ENABLE	0	R/W	Counter Enable 0 = Disable, 1 = Enable
23-0	PRESCALE	0x007FFF	R/W	Pre-scale counter load value. The pre-scale counter always runs from “0” up to PRESCALE. The default value is for 1Hz counter when ZCLK = XTIN (32.768kHz).

Possible counter modes are described in the table below.

Table 2.6 TC32 Count Mode

Mode	TC32EN Register Bits			Main Counter Operation	
	LOADZERO	LDM1	LDM0	Start Count Value	End Count Value
0	0	0	0	LOADVAL	0xFFFFFFFF
1	0	0	1	LOADVAL	CMP0 (if LOADVAL < CMP0)
2	0	1	0	LOADVAL	CMP1 (if LOADVAL < CMP1)
3	0	1	1	LOADVAL	CMP0 (if LOADVAL < CMP0 ≤ CMP1) or CMP1 (if LOADVAL < CMP1 ≤ CMP0)
4	1	0	0	0	LOADVAL – 1
5	1	0	1	0	CMP0 (if LOADVAL > CMP0)
6	1	1	0	0	CMP1 (if LOADVAL > CMP1)
7	1	1	1	0	CMP0 (if LOADVAL > CMP1 ≥ CMP0) or CMP1 (if LOADVAL > CMP0 ≥ CMP1)

Refer to register descriptions below for CMP0, CMP1 and LOADVAL.
Mode0 can be used as 1Hz counter mode, if PRESCALE = 0x007FFF, STOPMODE = 0, ZCLK = XTIN (32.768kHz)

TC32 Load Value Register (TC32LDV) 0xB0501084

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOADVAL															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOADVAL															

Field	Name	RW	Reset	Description
31-0	LOADVAL	0x00000000	R/W	Counter Load Value.

The counter is restarted whenever one of the TC32En and TC32LDV is written.

TC32 Match Value 0 Register (TC32CMP0) 0xB0501088

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMP0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP0															

Field	Name	RW	Reset	Description
31-0	CMP0	0x00000000	R/W	Counter Match Value

TC32 Match Value 1 Register (TC32CMP1) 0xB050108C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMP1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP1															

Field	Name	RW	Reset	Description
31-0	CMP1	0x00000000	R/W	Counter Match Value

TC32 Pre-scale Counter Current Value Register (TC32PCNT) 0xB0501090

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-								PCNT							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCNT															

Field	Name	RW	Reset	Description
31-24	Reserved	0x00	R	
23-0	PCNT	0x000000	R	Pre-scale counter current value. The SMU BUS clock must be three times faster than the frequency of ZCLK to read valid value.

TC32 Main Counter Current Value Register (TC32MCNT) 0xB0501094

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MCNT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCNT															

Field	Name	RW	Reset	Description
31-0	MCNT	0x00000000	R	Main counter current value. When RSYNC is enabled, the AHB system clock must be faster than the frequency calculated below. (ZCLK frequency) / (PRESCALE + 1) * 3

TC32 Interrupt Control Register (TC32IRQ) 0xB0501098

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IRQCLR	RSYNC	BITSEL						-			IRQEN[4]	IRQEN[3]	IRQEN[2]	IRQEN[1]	IRQEN[0]
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-			IRQRSTAT					-			IRQMSTAT				

Field	Name	Reset	R/W	Description
31	IRQCLR	0	R/W	Interrupt Clear Control. When this bit is 0, interrupt raw status bits (IRQRSTAT) are cleared by reading this register. When this bit is set, IRQSTAT bits are cleared only if written with non-zero value.
30	RSYNC	0	R/W	Synchronization control for Counter Current Value Registers (TC32PCNT and TC32MCNT). 0 = Enable, 1 = Disable.
29-24	BITSEL	0x00	R/W	Counter bit selection value for interrupt generation. Any one of the counter bits {MCNT[31:0], PCNT[23:0]} selected by BITSEL is used to generate an interrupt. 0x00 ~ 0x17 : PCNT[0] ~ PCNT[23] 0x18 ~ 0x38: MCNT[0] ~ MCNT[31]
23-21	Reserved	0	R/W	
20	IRQEN[4]	0	R/W	Enable Interrupt at the rising edge of a counter bit selected by BITSEL 0 = Disable, 1 = Enable
19	IRQEN[3]	0	R/W	Enable Interrupt at the end of pre-scale count 0 = Disable, 1 = Enable
18	IRQEN[2]	0	R/W	Enable Interrupt at the end of count 0 = Disable, 1 = Enable
17	IRQEN[1]	0	R/W	Enable Interrupt when the counter value matched with CMP1 0 = Disable, 1 = Enable
16	IRQEN[0]	0	R/W	Enable Interrupt when the counter value matched with CMP0 0 = Disable, 1 = Enable
15-13	Reserved	0	R/W	
12-8	IRQRSTAT	0x00	R/W	Interrupt Raw Status. Refer to the description for IRQEN above. 0 = Disable, 1 = Enable
7-5	Reserved	0	R/W	
4-0	IRQMSTAT	0x00	R/W	Masked Interrupt Status = IRQRSTAT & IRQEN

The IRQEN[n] is active high, ‘1’ means enabled and ‘0’ means disabled. And the IRQRSTAT[n] is active high, ‘1’ for valid and ‘0’ for invalid.

2.4 Operation & Timing Diagram

2.4.1 How to Run 16bits Timer

The Timer/Counter 0 ~ 3 are belongs to 16-bits timer group. The 16bits timer uses not only the bus clock but also TCLK. The TCO configuration is required to indicate the end of timer counting after clock configuration. After the CFG, CNT, REF registers are configured, you have only to set enable bit in the CFG to start counting. The MREF or PWM bit can be configured if required.

The TCK in the following figure starts to working if the enable bit in the CFG register. As shown in the following figure, the TCO can be inverted at the time in which TCNT and TREF are same and makes the interrupt to CPU.

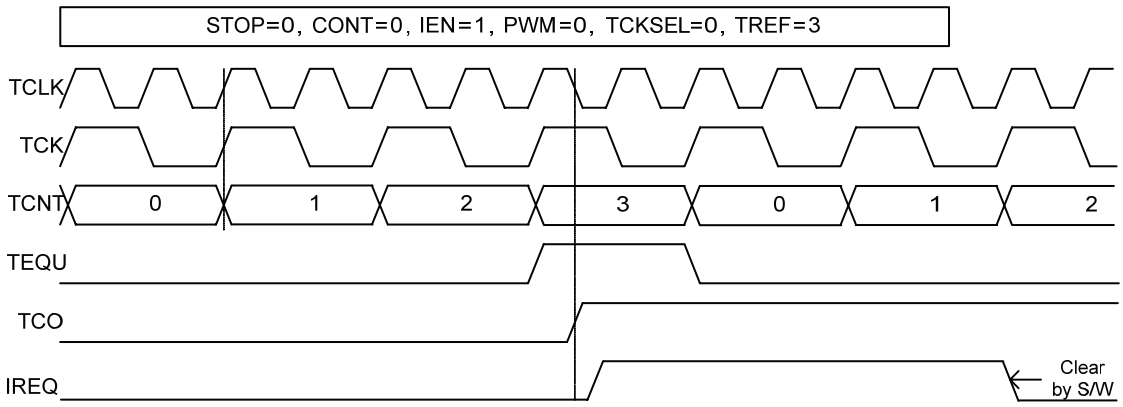


Figure 2.5 The Basic Timing Diagram of the Timer

You should take the cautions in using TCO into consideration in the case that the STOP bit is '1'. The TCO will be inverted in the TEQU being '1' in the following figure. So, if the STOP was '1', the TCNT will be stopped after it is same as the TREF and it makes the TCO will be toggled because the TEQU is always '1'. To avoid this toggling of the TCO, the enable bit of the CFG register should be cleared.

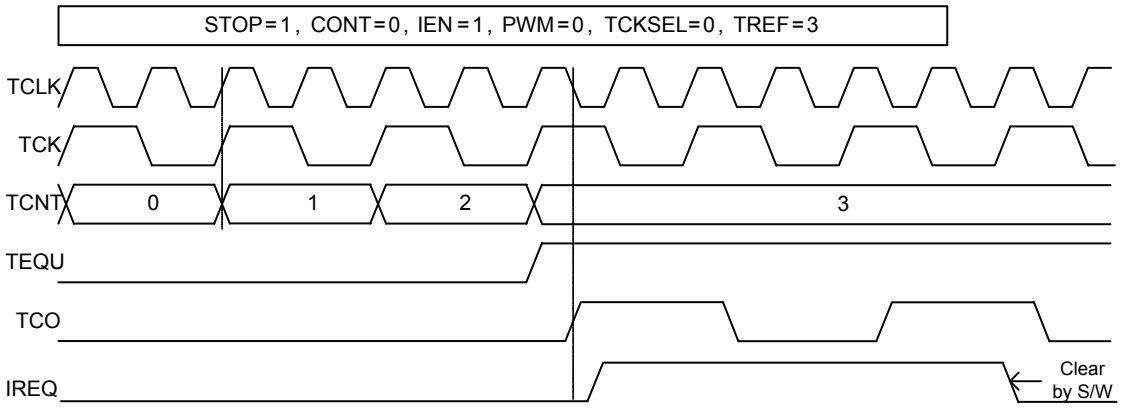


Figure 2.6 The Waveform in Case of STOP being '1'

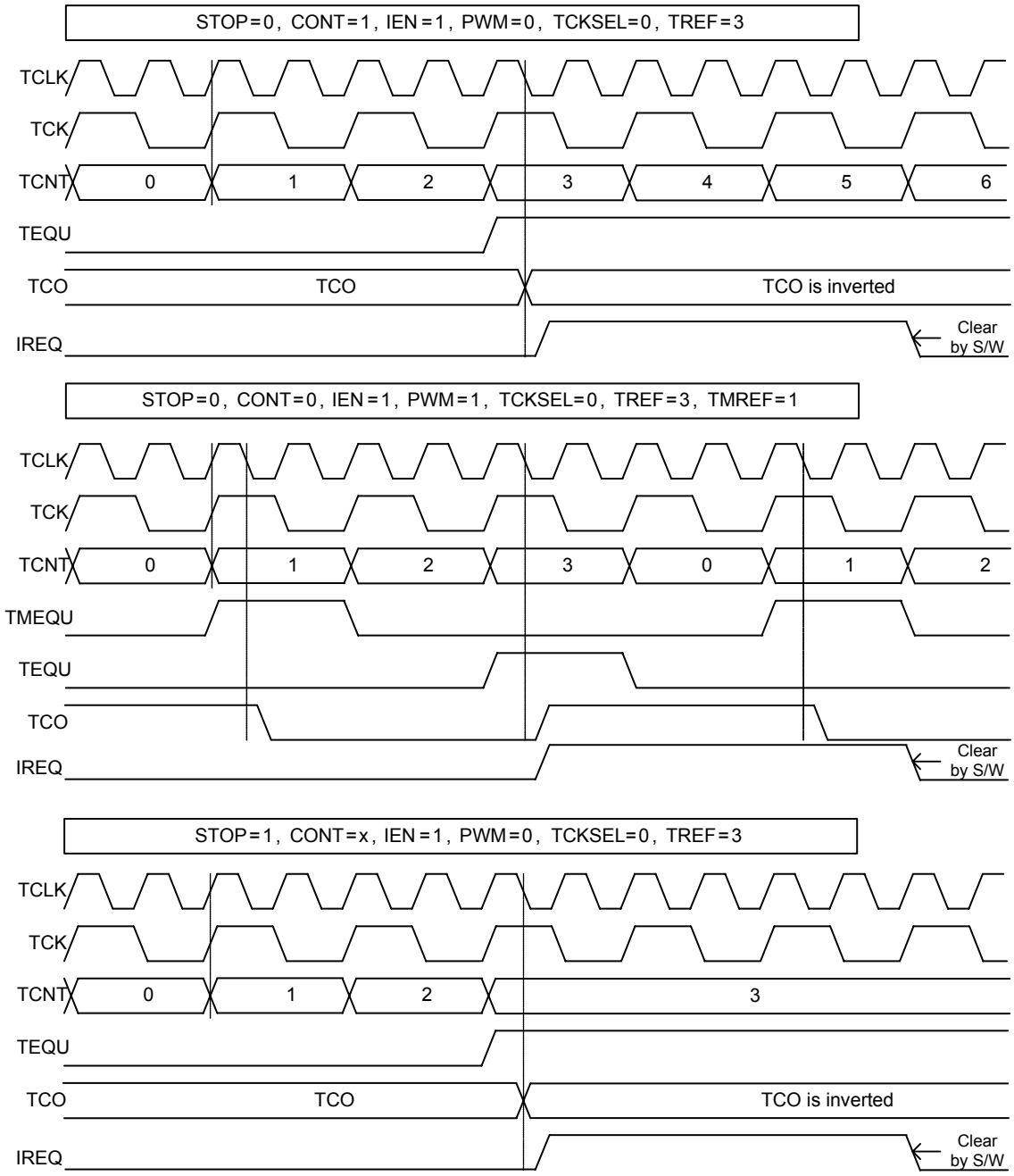


Figure 2.7 Timing Diagram of Timer/Counter

2.4.2 How to Run 20bits Timer

The Timer/Counter 4 and 5 are belongs to 20bits timer. Basically, the 20bits timer operation is same as the 16bits, but the 20bits timer does not have the MREF register and bitwidth of the CNT and REF counter is 20bits. The TCO output is same as the 16bits timer.

2.4.3 How to Run External Clock Counter

The external clocks can be used as the input clock to the 16bits timer. If the TCLK field of the CFG register for the 16bits timer is ‘7’, the 16bits timer uses the external clock as the operating clock.

2.4.4 How to Run Watchdog Timer

The watchdog timer uses the bus clock and TCLK. The bus clock is used to interface control or configuration registers and TCLK is used as the main operationg clock. The watchdog timer does not have the TCO output, so only interrupt can be used. The TCK selection bits should be one of the 4,5,6. In the another case, the clock could not be generated and does not

work properly.

2.4.5 How to Run 32bits Timer

The 32bits timer uses the bus clock and ZCLK. Same as the watchdog timer, there is no TCO output and only interrupt can indicate the end of the timer operation. The basic operation is same as the other timers, but there are many counting modes which can make more flexible to use in the system. Refer to 3.6 Table

2.4.6 Interrupt Structure for IREQ[0] – 16/20bits/Watchdog Timer/Counter,

The following figure shows the interrupt structure of the 16bits, 20bits, watchdog timer and how to be transferred to the main interrupt. There is no timing relation between various interrupt sources, the IREQ[0] should be used as the level-sensitive interrupt type and each interrupt status should be cleared in the handler routine as soon as possible.

2.4.7 Interrupt Structure for IREQ[1] – 32bits Timer/Counter

The IREQ[1] is dedicated to the 32bits timer.

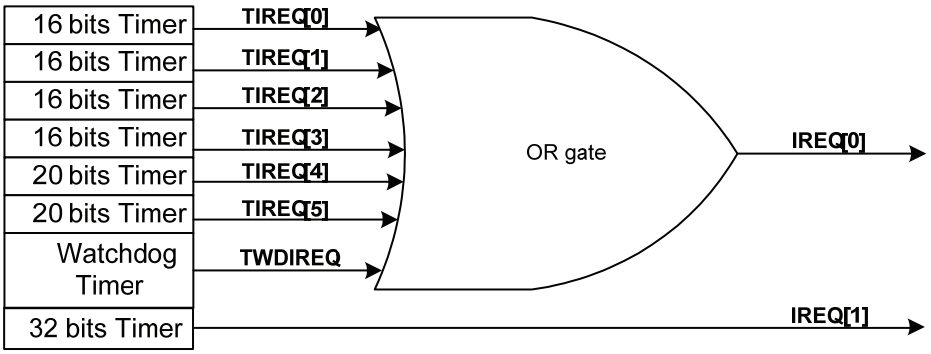


Figure 2.8 IREQ

3 PMU (power management unit)

3.1 Overview

The block diagram of PMU is shown in the following figure.

The PMU operated by XI/XO makes the signals to turn on or to turn-off the sub blocks or overall system.

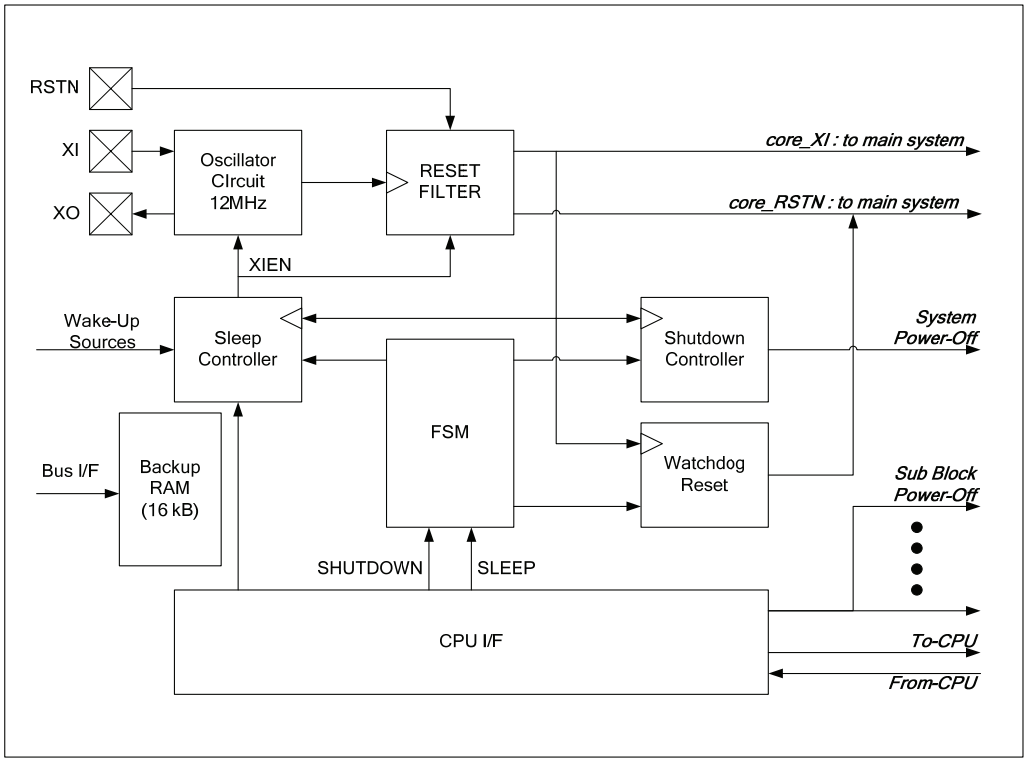


Figure 3.1 PMU Block Diagram

The PMU of the NVS2310 also can support wake-up procedure from the shutdown or sleep mode through the pre-defined wake-up sources of external or internal devices.. Watchdog reset controller generates the watchdog reset to internal system when the watchdog timeout counter is expired. A 16 kB backup RAM is integrated in PMU for fast system reboot from the shutdown mode.

3.2 Register Overview

Table 3.1 PMU Register Map (Base Address = 0xB0503000)

Name	Address	Type	Reset	Description
CONTROL	0x00	R/W		PMU Control Register
WKUPEN0	0x04	R/W		Wakeup Enable Configuration0 Register
WKUPPOL0	0x08	R/W		Wakeup Polarity Configuration0 Register
WATCHDOG	0x0C	R/W		Watchdog Control Register
CONFIG0	0x10	R/W		Boot Configuration Register
CONFIG1	0x14	R/W		Remap & Status Register
PWROFF	0x18	R/W		Power-Off Control Register
PWRCFG	0x1C	R/W		PMU configuration
PWRCNT	0x20	R/W		Power Up/Down Sequence Counter
WKUPSTS0	0x24	R		Wakeup Status0
WKUPSTS1	0x28	R		Wakeup Status1
WKUPEN1	0x2C	R/W		Wakeup Enable Configuration1 Register
WKUPPOL1	0x30	R/W		Wakeup Polarity Configuration1 Register
WKUPSTS_ALL0	0x34	R		Wakeup Status All0
WKUPSTS_ALL1	0x38	R		Wakeup Status All1
MEM_PDN	0x3C	R/W		Memory Power Down Control Register
MEM_SLN	0x40	R/W		Memory Sleep Control Register

3.3 Register Descriptions

CONTROL Register **0xB0503000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IOR	FWKU	FPO	IOR_M										AISO	ASTM	APEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
							WCLR			XI_E0			PDN	POFF	XIEN

Field	Name	RW	Reset	Description
31	IOR	R/W	0x0	I/O Retention Enable Register 0 : Not-Retentsion 1 : Retention **** Refer to the power-up/down sequence***
30	FWKU	R/W	0x0	Fast Wakeup Enable Register 0 : Normal 1 : Fast **** This is only for test mode.***
29	FPO	R/W	0x0	Fast Power-Off 0 : Normal 1 : Fast **** This is only for test mode***
28	IOR_M	R/W	0x0	DDR Memory I/O Retention Enable Register 0 : Not-Retention 1 : Retention **** Refer to the power-up/down sequence***
18	AISO	R/W	0x0	Touch Screen ADC Isolation Enable Register 0 : Not-Isolated 1 : Isolated
17	ASTM	R/W	0x0	Touch Screen ADC Stop Mode Register 0 : Normal 1 : Stop Mode
16	APEN	R/W	0x0	Touch Screen ADC Power Enable Register 0 : Disable 1 : Enable
8	WKCLR	R/W	0x0	Wakeup Status Clear 0 : Nothing 1 : Wakeup status clear **** When Set tio "1", WKUPSTS0/WKUPSTS1 is cleared***
5	XI_E0	R/W	0x1	Main Oscillator Mode Control Register 0 : Oscillator mode 1 : Crystal mode **** Do not write '0'***
2	SHTDN	R/W	0x0	Shutdown Mode Register 0 : Normal mode 1 : Shutdown mode **** Refer to enter/exit power-down sequence***
1	SLEEP	R/W	0x0	Sleep Mode Register 0 : Normal mode 1 : Sleep mode **** Should check the power-up/down sequence***
0	XIEN	R/W	0x1	Main Oscillator Enable Register 0 : Disable 1 : Enable **** Do not write '0'***

WKUPEN0 Register 0xB0503004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRCS[31:0]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SRCS[31:0]															

Field	Name	RW	Reset	Description
31-0	SRCS[31:0]	R/W	0x0	Wakeup Enable Register for Each Wakeup Sources SRCS[0] : RTCWKUP[3] SRCS[1] : GPIOC[28] SRCS[2] : GPIOC[29] SRCS[3] : GPIOC[30] SRCS[4] : GPIOC[31] SRCS[5] : GPIOF[27] SRCS[6] : GPIOF[26] SRCS[7] : GPIOF[28] SRCS[8] : GPIOF[16] SRCS[9] : GPIOF[17] SRCS[10] : TSC_WKU[4] SRCS[11] : GPIOD[18] SRCS[12] : TSC_STOP_WKU[5] SRCS[13] : TSC_UPDOWN[6] SRCS[14] : GPIOA[2] SRCS[15] : GPIOA[3] SRCS[16] : GPIOA[4] SRCS[17] : GPIOA[5] SRCS[18] : GPIOA[6] SRCS[19] : GPIOA[7] SRCS[20] : GPIOA[16] SRCS[21] : GPIOA[17] SRCS[22] : GPIOA[18] SRCS[23] : GPIOA[13] SRCS[24] : GPIOA[14] SRCS[25] : GPIOA[19] SRCS[26] : GPIOB[30] SRCS[27] : GPIOB[31] SRCS[28] : GPIOE[04] SRCS[29] : GPIOE[05] SRCS[30] : GPIOE[24] SRCS[31] : GPIOE[25]

[3] RTC Wakeup Output Signal
[4] Touch Screen Wake-Up Signal
[5] Touch Screen Stop Wakeup Signal
[6] Touch Screen Up/Down Signal

WKUPPOL0 Register **0xB0503008**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRCS[31:0]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SRCS[31:0]															

Field	Name	RW	Reset	Description
31-0	SRCS[31:0]	RW	0x0	Wakeup Enable Register for Each Wakeup Sources 0 : Active High 1 : Active Low SRCS[0] : RTCWKUP[7] SRCS[1] : GPIOC[28] SRCS[2] : GPIOC[29] SRCS[3] : GPIOC[30] SRCS[4] : GPIOC[31] SRCS[5] : GPIOF[27] SRCS[6] : GPIOF[26] SRCS[7] : GPIOF[28] SRCS[8] : GPIOF[16] SRCS[9] : GPIOF[17] SRCS[10] : TSC_WKU[8] SRCS[11] : GPIOD[18] SRCS[12] : TSC_STOP_WKU[9] SRCS[13] : TSC_UPDOWN[10] SRCS[14] : GPIOA[2] SRCS[15] : GPIOA[3] SRCS[16] : GPIOA[4] SRCS[17] : GPIOA[5] SRCS[18] : GPIOA[6] SRCS[19] : GPIOA[7] SRCS[20] : GPIOA[16] SRCS[21] : GPIOA[17] SRCS[22] : GPIOA[18] SRCS[23] : GPIOA[13] SRCS[24] : GPIOA[14] SRCS[25] : GPIOA[19] SRCS[26] : GPIOB[30] SRCS[27] : GPIOB[31] SRCS[28] : GPIOE[04] SRCS[29] : GPIOE[05] SRCS[30] : GPIOE[24] SRCS[31] : GPIOE[25]

[7] RTC Wakeup Output Signal
[8] Touch Screen Wake-Up Signal
[9] Touch Screen Stop Wakeup Signal
[10] Touch Screen Up/Down Signal

WATCHDOG Register 0xB050300C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN	CLR							STR							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					CNT										

Field	Name	RW	Reset	Description
31	EN	R/W	0x0	Watchdog Enable Register 0 : Disable 1 : Enable If watchdog reset activated, this will be cleared.
30	CLR	R/W	0x0	Watchdog Clear Register If write '1', the watchdog counter restarts. This is cleared automatically.
23-16	STR	R/W	0x0	User Status Register for Software This is not used by H/W. This is initialized by external RSTN. **** If watchdog reset activated, STR field is set to "0xF0" and cleared to "0x00" when external RTSN is asserted***
15-11	Reserved	Reserved for future use	0x1F	Reserved for future use
10-0	CNT	R/W	0x7FF	Watchdog Count Register The watchdog period is (CNT * 65536 / 12) usec. This is initialized by external RSTN.

CONFIG0 Register 0xB0503010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
							TM		PKG			BM			

Field	Name	RW	Reset	Description
8	TM	R/W	0x0	Test Mode Pin **** Don't write***
6-4	PKG	R/W	-	Package Port Status **** Don't write***
3-0	BM	R/W	-	Boot Mode Port Status **** Don't write***

CONFIG1 Register **0xB0503014**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			BBEN		REMAP			USTS							

Field	Name	RW	Reset	Description
12	BBEN	R/W	0x0	Backup RAM Boot Enable Register 0 : Disable 1 : Enable **** Refer to the Backup RAM boot procedure***
10-8	REMAP	R/W	0x0	System Remap Register 000 : Internal ROM 001 : Internal IRAM0 (128 KB) 010 : External DDR SDRAM 011 : External NOR 100 : Backup RAM (16 KB) 101 : Internal IRAM1 (64 KB)
7-0	USTS	R/W	0x0	User Status Register This register is initialized by RSTN port

PWROFF Register **0xB0503018**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					-	MB[2:0]			HSB[2:0]			CB[2:0]			GB[2:0]
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GB[2:0]		VB[2:0]			DSB[2:0]			DB[2:0]			-	UP0	UP1	HD	DAC

Field	Name	RW	Reset	Description
25-23	MB[2:0]	R/W	0x1	DDR Controller & PHY Power-Off Control Register MB[0] : DDR Controller & PHY Bus Isolation Enable 0 : Enable 1 : Disable MB[1] : DDR Controller & PHY Bus Pre Power Off Enable 0 : Disable 1 : Enable MB[2] : DDR Controller & PHY Bus Power Off Enable 0 : Disable 1 : Enable **** Refer to "Power On/Off Sequence for DDR Controller & PHY"***
22-20	HSB[2:0]	R/W	0x1	High Speed IO Bus Power-Off Control Register HSB[0] : High Speed IO Bus Isolation Enable 0 : Enable 1 : Disable HSB[1] : High Speed IO Bus Pre Power Off Enable 0 : Disable 1 : Enable HSB[2] : High Speed IO Bus Power Off Enable 0 : Disable 1 : Enable **** Refer to "Power On/Off Sequence for High Speed IO Bus"***
19-17	CB[2:0]	R/W	0x1	Camera Bus Power-Off Control Register CB[0] : Camera Bus Isolation Enable 0 : Enable 1 : Disable CB[1] : Camera Bus Pre Power Off Enable 0 : Disable 1 : Enable CB[2] : Camera Bus Power Off Enable 0 : Disable 1 : Enable **** Refer to "Power On/Off Sequence for Camera Bus"***
16-14	GB[2:0]	R/W	0x1	Graphic Bus Power-Off Control Register GB[0] : Graphic Bus Isolation Enable 0 : Enable 1 : Disable GB[1] : Graphic Bus Pre Power Off Enable 0 : Disable 1 : Enable GB[2] : Graphic Bus Power Off Enable 0 : Disable 1 : Enable **** Refer to "Power On/Off Sequence for Graphic Bus"***
13-11	VB[2:0]	R/W	0x1	Video Bus Power-Off Control Register VB[0] : Video Bus Isolation Enable 0 : Enable 1 : Disable VB[1] : Video Bus Pre Power Off Enable 0 : Disable 1 : Enable VB[2] : Video Bus Power Off Enable 0 : Disable 1 : Enable **** Refer to "Power On/Off Sequence for Video Bus"***
10-8	DSB[2:0]	R/W	0x1	Display Sub Bus Power-Off Control Register DSB[0] : Display Sub Bus Isolation Enable 0 : Enable 1 : Disable DSB[1] : Display Sub Bus Pre Power Off Enable 0 : Disable 1 : Enable DSB[2] : Display Sub Bus Power Off Enable 0 : Disable 1 : Enable

				**** Refer to "Power On/Off Sequence for Display Sub Bus"***
7-5	DB[2:0]	R/W	0x1	Display Bus Power-Off Control Register DB[0] : Display Bus Isolation Enable 0 : Enable 1 : Disable DB[1] : Display Bus Pre Power Off Enable 0 : Disable 1 : Enable DB[2] : Display Bus Power Off Enable 0 : Disable 1 : Enable **** Refer to "Power On/Off Sequence for Display Bus"***
3	UP0	R/W	0x0	USB0 Nano Phy Power-Off Control Register 0: Power-Off 1 : Power-On **** This is only used to isolate the core interface signals. If you want to reduce power consumption for the USB 2.0 phy, the external USB 2.0 power should be turned-off. The USB2.0 power is composed of VDD12_USB0, VDD33_USB0.***
2	UP1	R/W	0x0	USB1 Nano Phy Power-Off Control Register 0: Power-Off 1 : Power-On **** This is only used to isolate the core interface signals. If you want to reduce power consumption for the USB 2.0 phy, the external USB 2.0 power should be turned-off. The USB2.0 power is composed of VDD12_USB1, VDD33_USB1.***
1	HD	R/W	0x0	HDMI Phy Power-Off Control Register 0: Power-Off 1 : Power-On **** This is only used to isolate the core interface signals. If you want to reduce power consumption for the HDMI phy, the external HDMI power should be turned-off.***
0	DAC	R/W	0x0	Video DAC Power-Off Control Register 0: Power-Off 1 : Power-On **** This is only used to isolate the core interface signals. The DAC power should not be turned-off for leakage path between internal core power and DAC power.***

PWRCFG Register 0xB050301C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						BPDN	BSLN				SRST			BSRC	

Field	Name	RW	Reset	Description
9	BPDN	R/W	0x1	Backup RAM Power Down Control 0 : Power Down Enable 1 : Normal Operation ***For Normal Operation the Both of BSLN and BPDN should be "1"***
8	BSLN	R/W	0x1	Backup RAM Sleep Control 0 : Sleep Enable 1 : Normal Operation ***For Sleep Mode BPDN should be "1"***
4	SRST	R/W	0x1	System Reset Port Register 0 : "SYSRST" port outputs "0" 1 : "SYSRST" port outputs "1" ***When POR is asserted, it is set to "1" and "SYSRST" port outputs "high" state***
1-0	BSRC	R/W	0x0	Re-Boot from Shutdown Mode Base Address Selection 00 : Re-boot from Internal ROM 01 : Re-boot from Backup RAM 10 : Reserved 11 : Reserved

PWRCNT Register 0xB0503020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								WCNT							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCNT1								PCNT0							

Field	Name	RW	Reset	Description
23-16	WCNT	R/W	0x00	Wakeup Sequence Counter ***Recommended value is "TBD"***
15-8	PCNT1	R/W	0x1F	Power On/Off Sequence Counter 0 Global Logic Power On/Off Counter ***Recommended value is "TBD"***
7-0	PCNT0	R/W	0x0F	Power On/Off Sequence Counter 0 Corebus Power On/Off Counter ***Recommended value is "TBD"***

WKUPSTS0 Register 0xB0503024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSTS[31:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSTS[31:0]															

Field	Name	RW	Reset	Description
31-0	WSTS[31:0]	R/W	0x0	Wakeup Status Register0 ***In WKUPSTS0/WKUPSTS1 register, Only the first occured wakeup source during the shutdown or sleep mode state among the enabled wakeup sources is recorded***

WKUPSTS1 Register 0xB0503028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSTS[63:32]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSTS[63:32]															

Field	Name	RW	Reset	Description
31-0	WSTS[63:32]	R/W	0x0	Wakeup Status Register1 ***In WKUPSTS0/WKUPSTS1 register, Only the first occured wakeup source during the shutdown or sleep mode state among the enabled wakeup sources is recorded***

WKUPEN1 Register 0xB050302C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRCS[63:32]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRCS[63:32]															

Field	Name	RW	Reset	Description
31-0	SRCS[63:32]	R/W	0x0	Wakeup Enable Register for Each Wakeup Sources SRCS[32] : GPIOB[17] SRCS[33] : GPIOB[19] SRCS[34] : GPIOB[26] SRCS[35] : GPIOB[27] SRCS[36] : GPIOC[20] SRCS[37] : GPIOC[21] SRCS[38] : GPIOC[22] SRCS[39] : GPIOC[23] SRCS[40] : GPIOD[15] SRCS[41] : GPIOD[16] SRCS[42] : GPIOD[17] SRCS[43] : GPIOD[19] SRCS[44] : GPIOD[20] SRCS[45] : GPIOE[26] SRCS[46] : GPIOE[27] SRCS[47] : GPIOE[28] SRCS[48] : GPIOE[29] SRCS[49] : GPIOE[30] SRCS[50] : GPIOE[31] SRCS[51] : GPIOG[04] SRCS[52] : GPIOG[05] SRCS[53] : GPIOG[06] SRCS[54] : GPIOG[07] SRCS[55] : GPIOG[11] SRCS[56] : GPIOG[12] SRCS[57] : GPIOG[17] SRCS[58] : GPIOG[26] SRCS[59] : GPIOG[27] SRCS[60] : GPIOG[28] SRCS[61] : GPIOG[29] SRCS[62] : Reserved SRCS[63] : RMWKUP[11]

[11] Wakeup Signal from Remote Control Logic

WKUPPOL1 Register 0xB0503030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRCS[63:32]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRCS[63:32]															

Field	Name	RW	Reset	Description
31-0	SRCS[63:32]	R/W	0x0	Wakeup Enable Register for Each Wakeup Sources 0 : Active High 1 : Active Low Wakeup Enable Register for Each Wakeup Sources SRCS[32] : GPIOB[17] SRCS[33] : GPIOB[19] SRCS[34] : GPIOB[26] SRCS[35] : GPIOB[27] SRCS[36] : GPIOC[20] SRCS[37] : GPIOC[21] SRCS[38] : GPIOC[22] SRCS[39] : GPIOC[23] SRCS[40] : GPIOD[15] SRCS[41] : GPIOD[16] SRCS[42] : GPIOD[17] SRCS[43] : GPIOD[19] SRCS[44] : GPIOD[20] SRCS[45] : GPIOE[26] SRCS[46] : GPIOE[27] SRCS[47] : GPIOE[28] SRCS[48] : GPIOE[29] SRCS[49] : GPIOE[30] SRCS[50] : GPIOE[31] SRCS[51] : GPIOG[04] SRCS[52] : GPIOG[05] SRCS[53] : GPIOG[06] SRCS[54] : GPIOG[07] SRCS[55] : GPIOG[11] SRCS[56] : GPIOG[12] SRCS[57] : GPIOG[17] SRCS[58] : GPIOG[26] SRCS[59] : GPIOG[27] SRCS[60] : GPIOG[28] SRCS[61] : GPIOG[29] SRCS[62] : Reserved SRCS[63] : RMWKUP[12]

WKUPSTS_ALL0 Register 0xB0503034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSTS[31:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSTS[31:0]															

Field	Name	RW	Reset	Description
31-0	WSTS[31:0]	R/W	0x0	Wakeup Status Register0 ***In WKUPSTS_ALL0/WKUPSTS_ALL1 register, All the occured wakeup source during the shutdown or sleep mode state among the enabled wakeup sources are recorded***

[12] Wakeup Signal from Remote Control Logic

WKUPSTS_ALL1 Register 0xB0503038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WSTS[63:32]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WSTS[63:32]															

Field	Name	RW	Reset	Description
31-0	WSTS[63:32]	R/W	0x0	Wakeup Status Register1 ***In WKUPSTS_ALL0/WKUPSTS_ALL1 register, All the occured wakeup source during the shutdown or sleep mode state among the enabled wakeup sources are recorded***

MEM_PDN Register 0xB050303C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									RC2_PD	RC1_PD	RC0_PD	IRMA1_PD		IRAM0_PD	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NFC_PD	GE_PD							USB_PD		EHI1_PD	EHI0_PD	SD3_PD	SD2_PD	SD1_PD	SD0_PD

Field	Name	RW	Reset	Description
22	RC2_PD	R/W	0x1	High Speed IO Bus Prefetch Buffer Power Down 0 : Power Down 1 : Normal
21	RC1_PD	R/W	0x1	IO Bus Prefetch Buffer Power Down 0 : Power Down 1 : Normal
20	RC0_PD	R/W	0x1	CPU Prefetch Buffer Power Down 0 : Power Down 1 : Normal
19-18	IRAM1_PD	R/W	0x0	IRAM1 Power Down 00 : Normal 11 : Power Down
17-16	IRAM0_PD	R/W	0x0	IRAM0 Power Down 00 : Normal 11 : Power Down
15	NFC_PD	R/W	0x1	NFC Memory Power Down 0 : Power Down 1 : Normal
14-8	GE_PD	R/W	0x3F	Overlay Mixer Memory Power Down 0x00 : Power Down 0x3F : Normal
7-6	USB_PD	R/W	0x3	USB0 Memory Power Down 00 : Power Down 11 : Normal
5	EHI1_PD	R/W	0x1	EHI0 Memory Power Down 0 : Power Down 1 : Normal
4	EHI0_PD	R/W	0x1	EHI0 Memory Power Down 0 : Power Down 1 : Normal
3	SD3_PD	R/W	0x0	SD3 Memory Power Down 0 : Normal 1 : Power Down
2	SD2_PD	R/W	0x0	SD2 Memory Power Down 0 : Normal 1 : Power Down
1	SD1_PD	R/W	0x0	SD1 Memory Power Down 0 : Normal 1 : Power Down
0	SD0_PD	R/W	0x0	SD0 Memory Power Down 0 : Normal 1 : Power Down

MEM_SLN Register 0xB0503040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									RC2_SL	RC1_SL	RC0_SL	IRMA1_SL		IRAM0_SL	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NFC_SL	GE_SL							USB_SL		EHI1_SL	EHI0_SL	SD3_SL	SD2_SL	SD1_SL	SD0_SL

Field	Name	RW	Reset	Description
22	RC2_SL	R/W	0x1	High Speed IO Bus Prefetch Buffer Sleep 0 : Sleep 1 : Normal
21	RC1_SL	R/W	0x1	IO Bus Prefetch Buffer Sleep 0 : Sleep 1 : Normal
20	RC0_SL	R/W	0x1	CPU Prefetch Buffer Sleep 0 : Sleep 1 : Normal
19-18	IRAM1_SL	R/W	0x0	IRAM1 Sleep 00 : Normal 11 : Sleep
17-16	IRAM0_SL	R/W	0x0	IRAM0 Sleep 00 : Normal 11 : Sleep
15	NFC_SL	R/W	0x1	NFC Memory Sleep 0 : Sleep 1 : Normal
14-8	GE_SL	R/W	0x3F	Overlay Mixer Memory Sleep 0x00 : Sleep 0x3F : Normal
7-6	USB_SL	R/W	0x3	USB0 Memory Sleep 00 : Sleep 11 : Normal
5	EHI1_SL	R/W	0x1	EHI0 Memory Sleep 0 : Sleep 1 : Normal
4	EHI0_SL	R/W	0x1	EHI0 Memory Sleep 0 : Sleep 1 : Normal
3	SD3_SL	R/W	0x0	SD3 Memory Sleep 0 : Normal 1 : Sleep
2	SD2_SL	R/W	0x0	SD2 Memory Sleep 0 : Normal 1 : Sleep
1	SD1_SL	R/W	0x0	SD1 Memory Sleep 0 : Normal 1 : Sleep
0	SD0_SL	R/W	0x0	SD0 Memory Sleep 0 : Normal 1 : Sleep

3.4 Operation & Timing Diagram

3.4.1 Power Management Scheme

The following figure shows the overall block diagram for power management of the NVS2310.

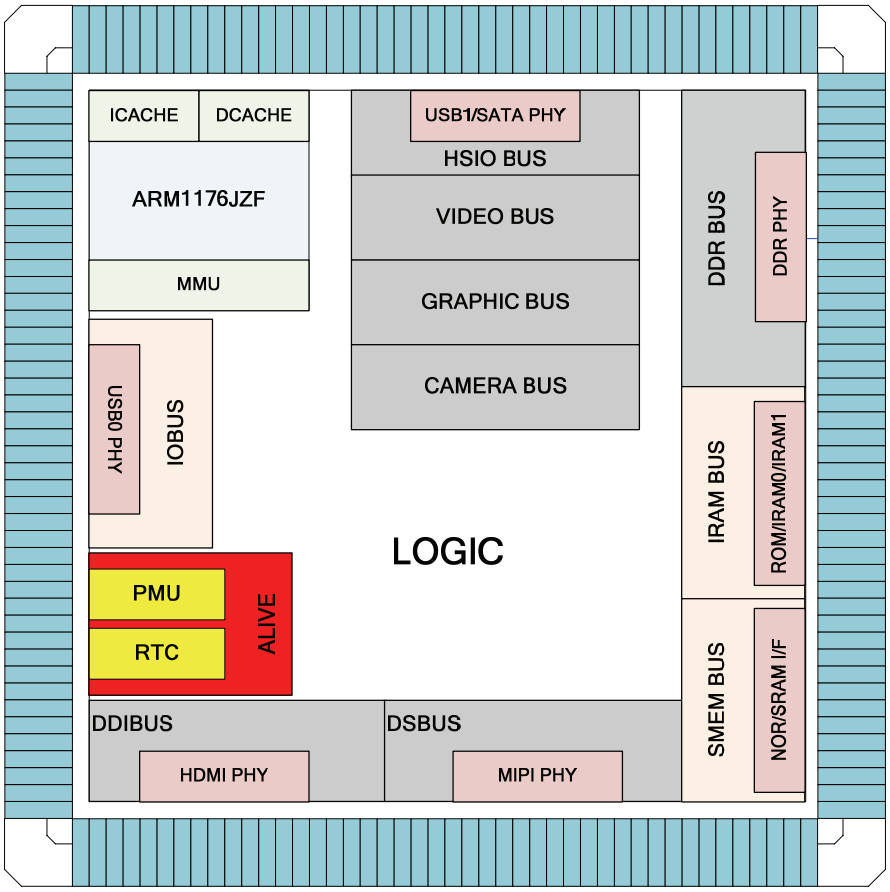


Figure 3.2 Overall Block Diagram

Basically, The NVS2310 has the 9 controllable power groups for power management. The PMU can control the power state of each block. But, the RTC block is always alive if VDD_RTC is supplied.

the GPIOs in the NVS2310 can keep the state such as input or output direction and output value in the output mode. It means that the register values for GPIOs before turning-off the internal power should reside in the Backup memories or external memory to restore the state of the GPIOs.

3.4.2 Operating Mode Definitions

3.4.2.1 Overview

The global operation modes of NVS2310 is shown in the following figure.

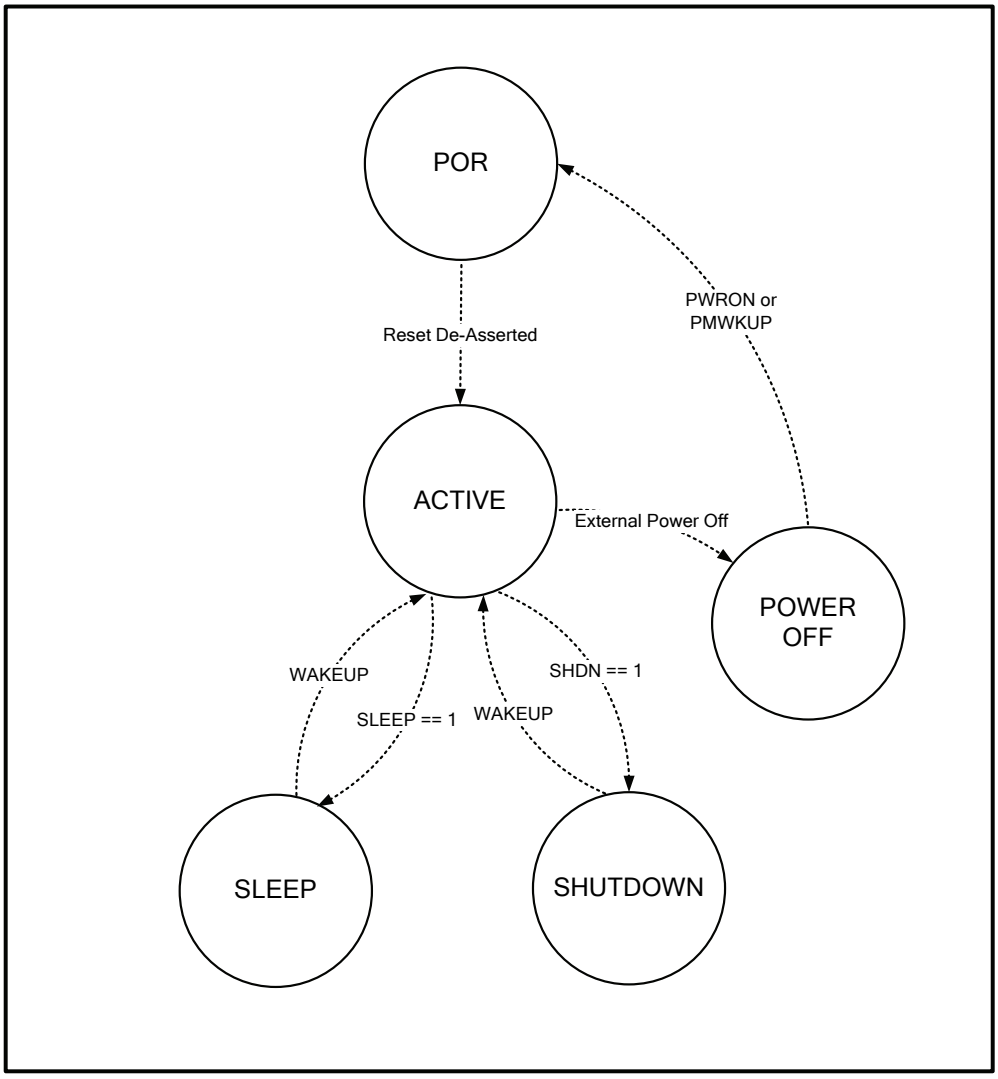


Figure 3.3 Operating Modes

3.4.2.2 POWER OFF Mode

This mode is defined that all the powers except for the RTC are turned-off which means the power consumption is lowest. The only way to wake up is to be turned on by the power circuit through external events such as key or the RTC “PMWKUP”.

3.4.2.3 POR Mode

This mode is defined that the core, IO and all the necessary powers of NVS2310 are supplied and the reset signal is asserted through “NPOR” input port of NVS2310 by the external Power-On-Reset circuits. When the power-on reset goes to “low” to “high”, NVS2310 starts pre-defined procedure for the system boot.

3.4.2.4 ACTIVE Mode

This mode is defined that NVS2310 completes the initial system boot procedures and conduct normal operation. In ACTIVE mode, The internal power supply of the below 7 sub bus groups (shaded bus in Fig4-2) can be turned on or off internally by CPU for low power operation.

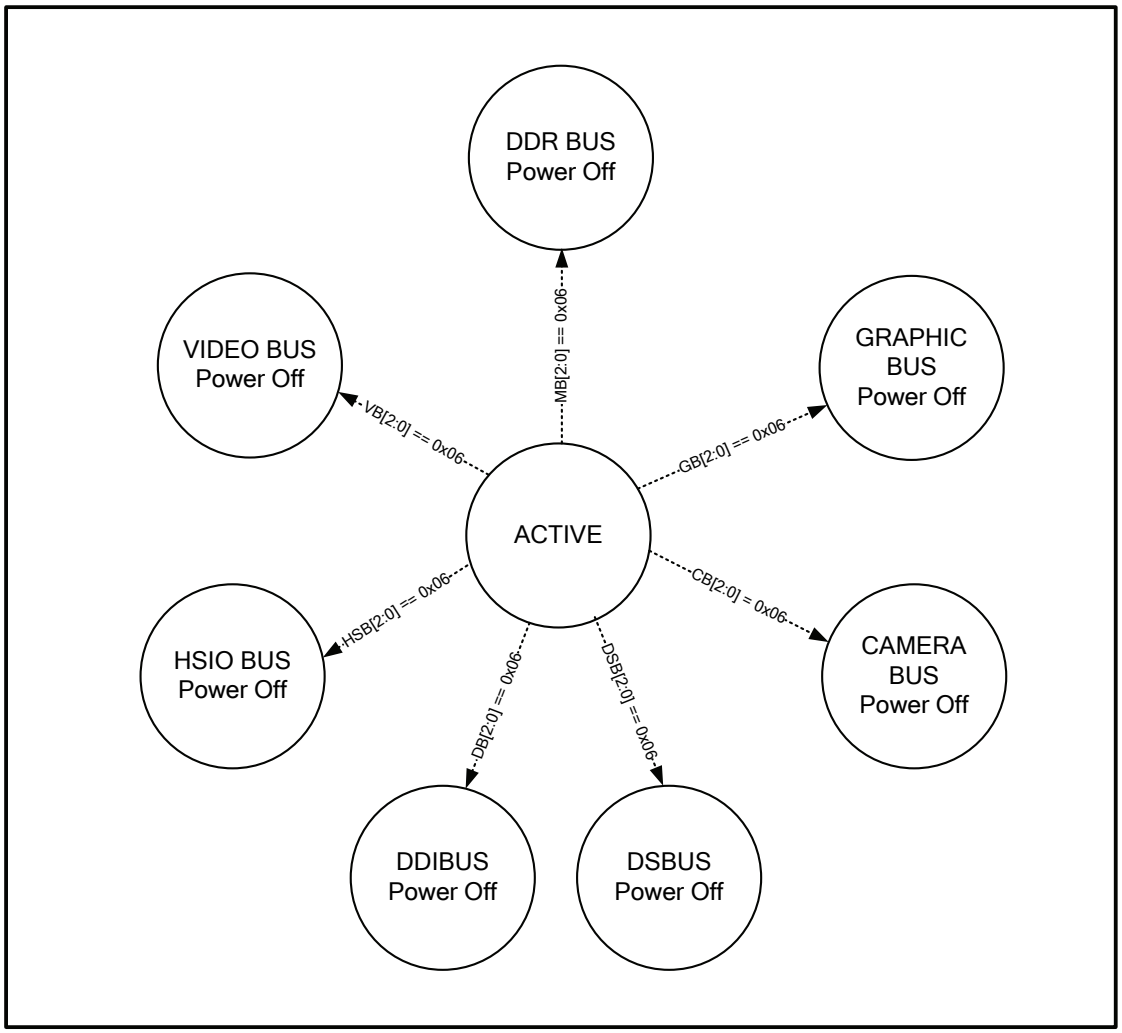


Figure 3.4 Sub Bus Power On/Off Control in ACTIVE Mode

3.4.2.5 SLEEP Mode

This mode is defined that internal core, I/O and all the necessary powers are alive but all the clock sources of NVS2310 are disabled including the main oscillator. Because all the necessary supply powers are alive, all the necessary states of NVS2310 are preserved. Restarting the main oscillator is done by the pre-defined wake-up events which can be activated externally or internally. For example, the touch screen event is belongs to the internally generated event such as UP/DOWN event described in the TSC(Touch Screen Controller). The sleep mode is required for the low-power operation with fast wake-up time.

3.4.2.6 SHUTDOWN Mode

This mode is defined that all the powers are internally “power-off” and the only alive blocks (PMU, RTC) remain “power-on”. In this mode, the main oscillator is also disabled and because all internal power except for the alive blocks is “power-off”, all the states of NVS2310 do not preserved. Therefore, the wake-up procedure from the SHUTDOWN mode requires the system boot operation. NVS2310 provides 16 KB backup RAM which can be alive during the SHUTDOWN mode for system boot operation from SHUTDOWN mode. In addition to the backup RAM, IRAM0(128 KB)/IRAM1(64 KB) also can preserve their contents during the SHUTDOWM mode by software control. The wake-up procedure from the SHUTDOWM mode is done by the pre-defined wake-up events which can be activated externally or internally.

Modes	Ext. I/O Power	Ext. Core Power	Int. Core Power	Int. I/O Retension	ENTER /WAKEUP	ALIVE BLOCKS
POWER-OFF	OFF	OFF	OFF	X	KEY/ PMWKUP	RTC
POR	ON	ON	ON	O/X[13]	CPU/ WAKEUP	-
ACTIVE	ON	ON	OFF/ON[14]	O/X[13]	CPU/ WAKEUP	-
SLEEP	ON	ON	OFF/ON[15]	O/X[13]	CPU/ WAKEUP	-
SHUTDOWN	ON	ON	OFF[16]	O/X[13]	-	RTC/PMU

[13] The I/O retension function can be controlled by CPU(Program) before entering the corresponding mode.
[14] 7 sub bus can be power-on or power-off by software control in this mode
[15] 7 sub bus can be power-on or power-off by software control in this mode
[16] Except for alive block (PMU)

3.4.3 Basic Power-Up Sequence

An example of the timing diagram for power-up sequence is shown below figure

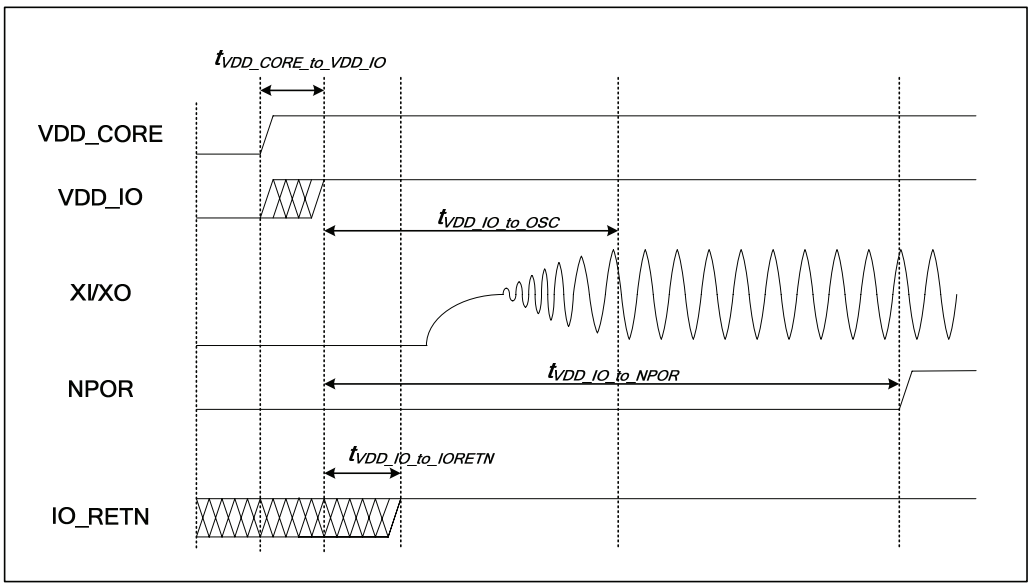


Figure 3.5 Power-Up Sequence

- $t_{VDD\_CORE\_to\_VDD\_IO}$: VDD_IO turn-on time after VDD_CORE asserted.
  0 =< $t_{VDD\_CORE\_TO\_VDD\_IO}$
  ($t_{VDD\_CORE\_TO\_VDD\_IO}$ is recommened as mimimum as possible)
- $t_{VDD\_IO\_to\_OSC}$: The main oscillator stable time after VDD_IO(OSC) asserted.
  $t_{VDD\_IO\_TO\_OSC}$ < 10ms
- $t_{VDD\_IO\_to\_NPOR}$: The NPOR time after VDD_IO asserted.
  20ms < $t_{PWRIO\_TO\_RSTN}$
- $t_{VDD\_IO\_to\_IORETN}$: The I/O release time after VDD_IO asserted when NPOR LOW.
  $t_{VDD\_IO\_TO\_IORETN}$ < 1us
- *Power State* : HIGH : VDD * 0.9, LOW : VDD * 0.1

The function of I/O retension is defined that the I/O direction control signal and output level were kept during the VDD_CORE off. In the NVS2310, the internal core power can be controlled by internal power management circuit.

3.4.4 Basic Power-Off Sequence – Turn-Off

The sequence in this section is invalid for the pre-defined power-down modes such as POWER OFF, SHUTDOWN, SLEEP, and etc. An example of the timing diagram for power-off sequence is shown below figure.

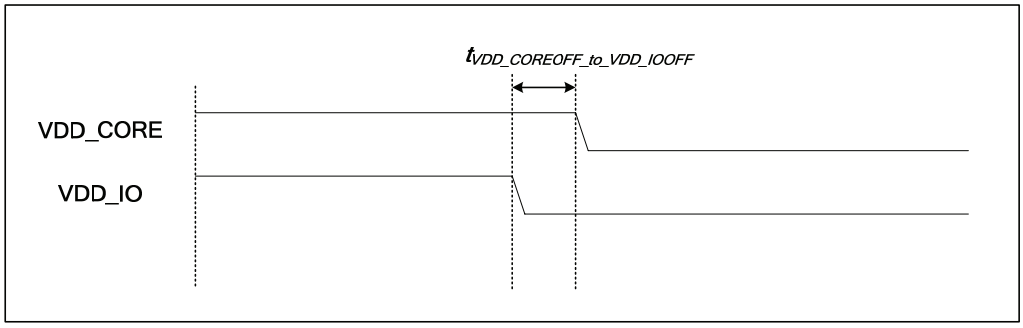


Figure 3.6 Power-Off Sequence

- $t_{VDD\_COREOFF\_to\_VDD\_IOOFF}$: VDD_CORE turn-off time after VDD_IO de-asserted.
  $0 < t_{VDD\_COREOFF\_TO\_VDD\_IOOFF}$
  ($t_{VDD\_COREOFF\_TO\_VDD\_IOOFF}$ is recommened as mimimum as possible)
- *Power State* : HIGH : VDD * 0.9, LOW : VDD * 0.1

3.4.5 Enter POR State from Initial Boot-Up or RTC PMWKUP

If there is rising transition on NPOR port, the NVS2310 starts to initialize the PMU and enter the POR state. In this section, the timing diagram from the NPOR rising transition to POR state will be described.

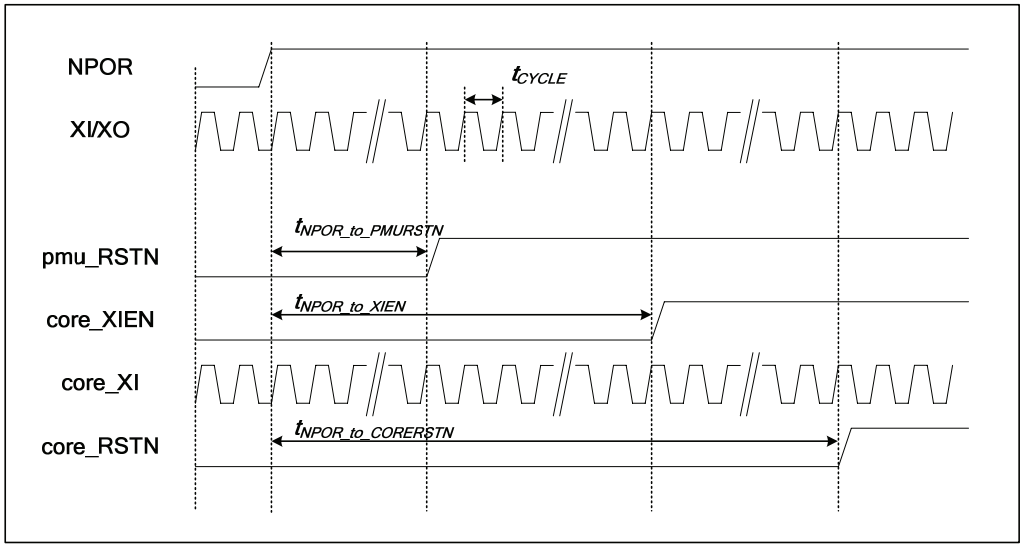


Figure 3.7 Timing Diagram for Entering POR State

- t_{CYCLE} : Clock cycle period, about 84ns for 12MHz.
- $t_{NPOR\_to\_PMURSTN}$: The cycle period from external NPOR to internal RSTN for PMU.
  $t_{NPOR\_to\_PMURSTN}$ is about 11ms.
- $t_{NPOR\_to\_XIEN}$: The XI/XO stable indicator.
  $t_{NPOR\_to\_XIEN}$ is about 12ms.
- $t_{NPOR\_to\_CORERSTN}$: The cycle period from external NPOR to internal RSTN for core.
  $t_{NPOR\_to\_XIEN}$ is about 16ms.

3.4.6 Enter SLEEP Mode from ACTIVE Mode

In the SLEEP mode, the XI/XO oscillator is disabled, which requires the following restrictions before entering this mode.

- All the clock sources using PLL s should be changed to external XI source .
- The wakeup event should be assigned to the corresponding wakeup source.

The SLEEP mode can be entered by writing '1' to SLEEP in the CONTROL register. After that, the SLEEP sequence is as following figure.

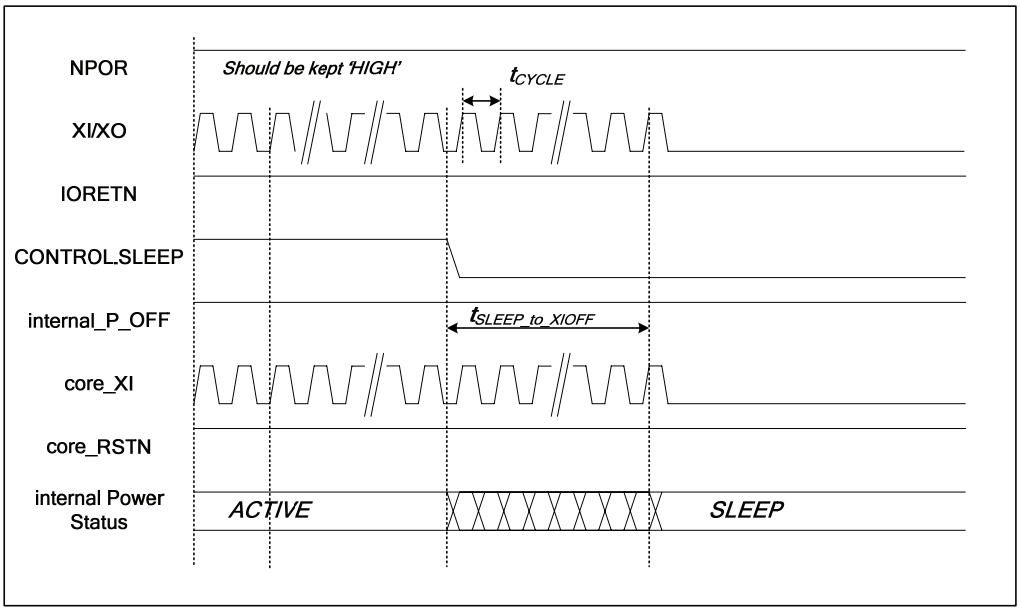


Figure 3.8 Timing Diagram for Entering SLEEP from ACTIVE Mode

- t_{CYCLE} : Clock cycle period, about 84ns for 12MHz.
- $t_{SLEEP\_to\_XIOFF}$: The cycle period from SLEEP(register) to main oscillator disable.
  $t_{SLEEP\_to\_XIOFF}$ is controllable from programmable register

3.4.7 Enter SHUTDOWN Mode from ACTIVE

In the SHUTDOWN mode, the XI/XO oscillator is disabled, which requires the following restrictions before entering this mode.

- All the clocks using PLL should be changed to XI or disabled.
- The wakeup event should be assigned to the corresponding wakeup source.
- The I/O retension function should be enabled.

The SHUTDOWN mode can be entered by writing '1' to SHTDN in the CONTROL register. After that, the SHUTDOWN sequence is as following figure.

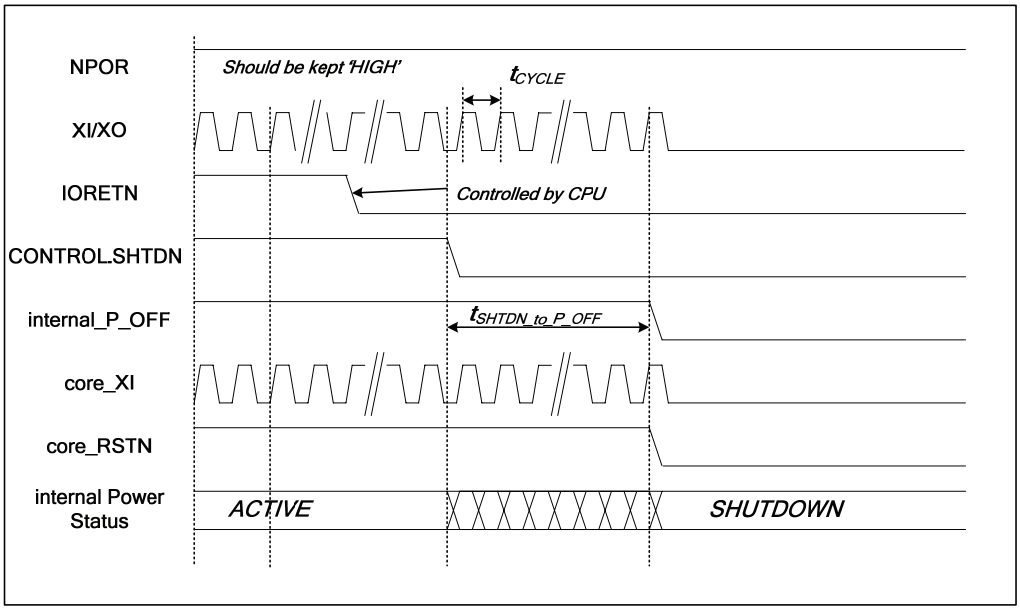


Figure 3.9 Timing Diagram for Entering SHUTDOWN from ACTIVE Mode

- t_{CYCLE} : Clock cycle period, about 84ns for 12MHz.
- $t_{POFF\_to\_IPOFF}$: The cycle period from SHTDN(register) to internal power-off signal. $t_{SHTDN\_to\_IPOFF}$ is controllable from programmable register

3.4.8 WAKE-UP Event Configuration

The WAKE-UP event can be made by the various sources which are defined in the register description. The following figure shows how to make WAKE-UP event with polarity control register and enable register.

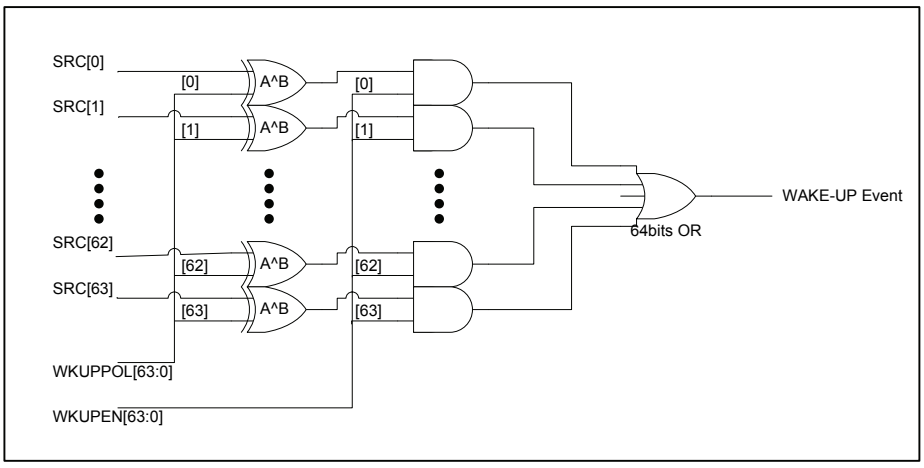


Figure 3.10 Timing Diagram for Exiting SLEEP and SHUTDOWN Mode

3.4.9 Exit SLEEP Mode

To wake-up from the SLEEP mode, the wake-up sources should be configured.

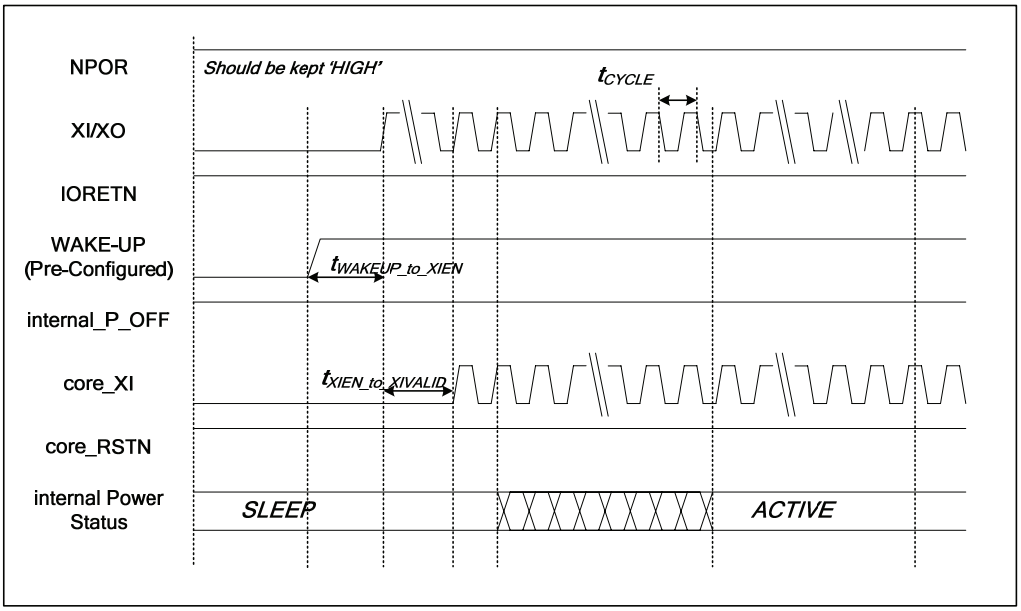


Figure 3.11 Timing Diagram for Exiting SLEEP Mode

- t_{CYCLE} : Clock cycle period, about 84ns for 12MHz.
- $t_{WAKEUP\_to\_XIEN}$: The maximum delay from WAKEUP to XIEN.
  $t_{WAKEUP\_to\_XIEN}$ < 1us
- $T_{XIEN\_to\_XIVALID}$: The maximum delay from XIEN to valid clock output.
  $t_{XIEN\_to\_XIVALID}$ is about 11ms.

3.4.10 Exit SHUTDOWN Mode

To wake-up from the SHUTDOWN mode, the wake-up sources should be configured.

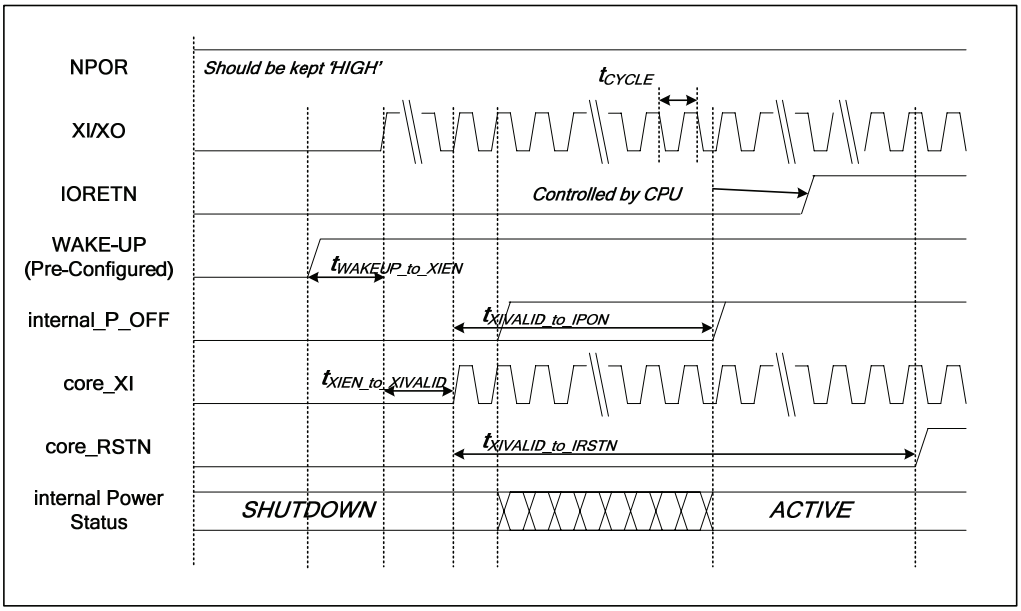


Figure 3.12 Timing Diagram for Exiting SHUTDOWN Mode

- t_{CYCLE} : Clock cycle period, about 84ns for 12MHz.
- $t_{WAKEUP\_to\_XIEN}$: The maximum delay from WAKEUP to XIEN.
  $t_{WAKEUP\_to\_XIEN}$ < 1us
- $T_{XIEN\_to\_XIVALID}$: The maximum delay from XIEN to valid clock output.
  $t_{XIEN\_to\_XIVALID}$ is about TBD.
- $t_{XIVALID\_to\_IPON}$: The clock cycles from XIVALID to internal Power-On.
  $t_{XIVALID\_to\_IPON}$ is about TBD
- $t_{XIVALID\_to\_IRSTN}$: The clock cycles from XIVALID to internal reset.
  $t_{WAKEUP\_to\_XIEN}$ is about TBD.

3.4.11 Enter and Exit SHUTDOWN Mode with Backup RAM Booting

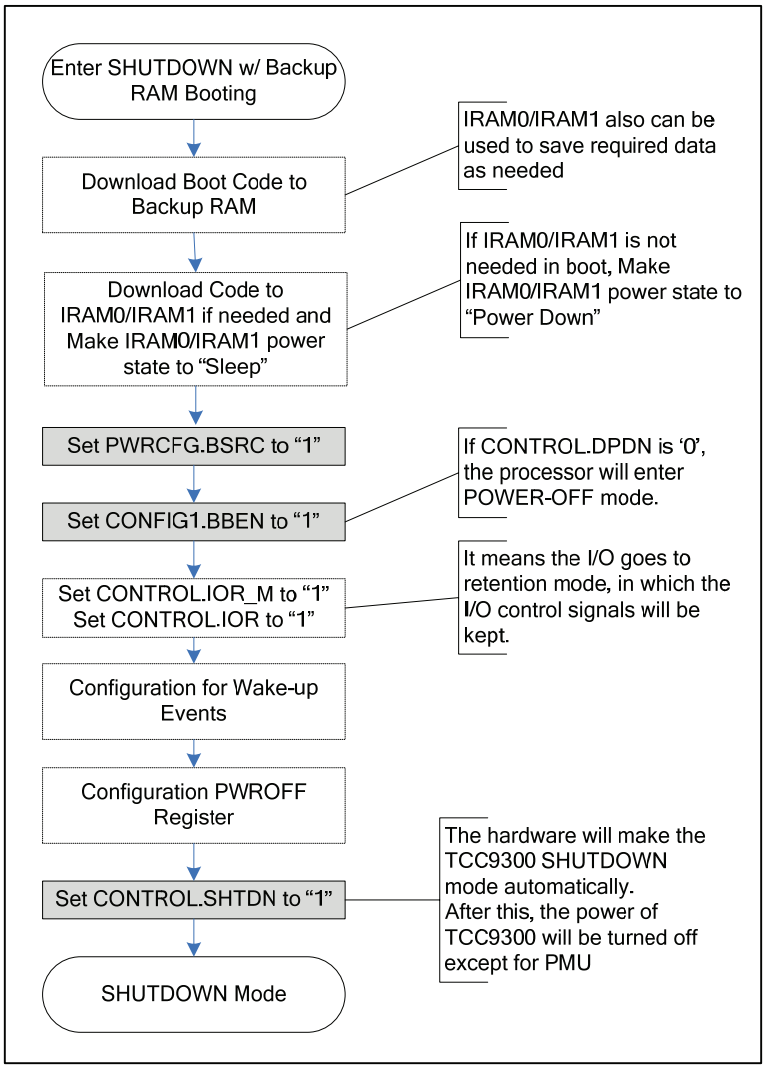


Figure 3.13 Example Flow Chart to Enter SHUTDOWN mode with Backup RAM Boot

The above figure shows an example flow chart to enter SHUTDOWN mode. the "PWRCFG.BSRC" register field selects the location of initial boot code between the internal boot ROM and the backup RAM. If the internal boot ROM is selected, NVS2310 start the boot procedure with the boot code in the internal boot ROM (this is the same boot procedure from POWER OFF mode), If the backup RAM is selected NVS2310 start the boot procedure with the boot code in the backup RAM. When the PMU enters SHUTDOWN mode with "CONTROL.SHTDN", Just after "CONTROL.SHTDN" register being '1', the main oscillator will be disabled and the power of the ARM and other peripherals will be turned off consequently.

The "CONTROL.IOR_M" and "CONTROL.IOR" signals which control the DDR IO and the General IO retention respectively make the I/Os to retend their status including direction, driver strengths, pull-up/down control signals and during the "CONTROL.IOR_M" and "CONTROL.IOR" being '1', the I/O can't be controlled by internal signals.

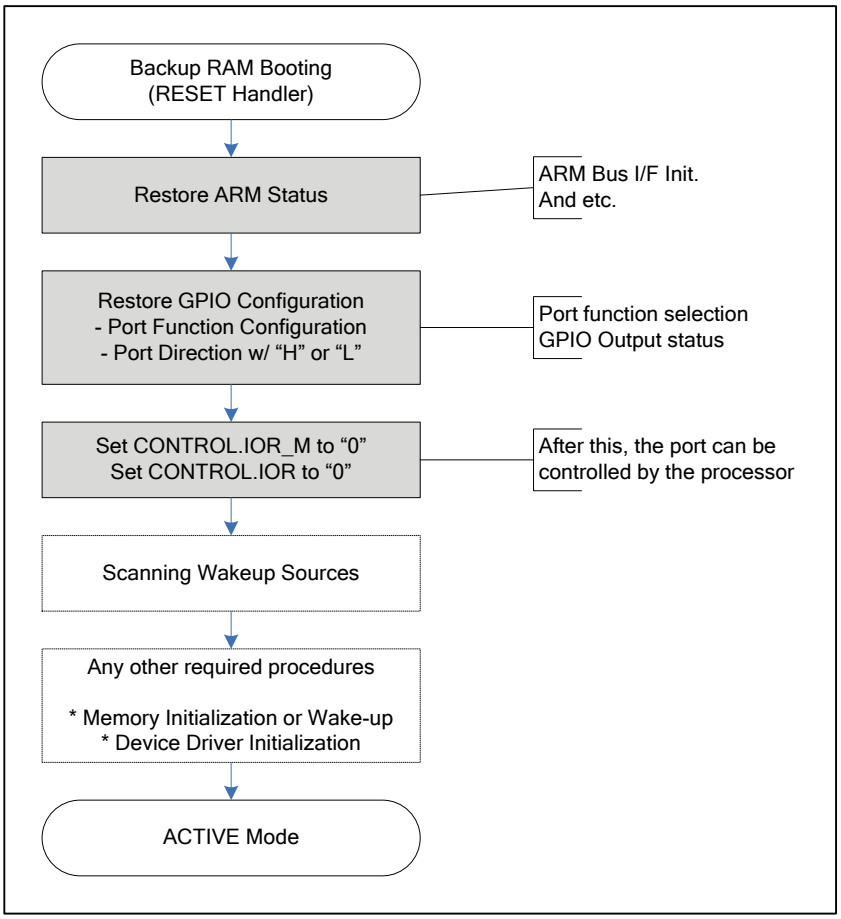


Figure 3.14 Example Flow Chart to Exit SHUTDOWN Mode with Backup RAM Boot

The above figure shows a example flow chart to exit SHUTDOWN mode. When the wake-up events occurred, the PMU makes clock enable and de-assert the RESET to internal core and ARM cpu and then the ARM can start booting procedure. At this time, if the the "PWRCFG.BSRC" register is "0x01" and "CONFIG1.BBEN" is "1", the ARM cpu will fetch the start instruction from the backup RAM.

After ARM booted, the ARM status should be restored by software such as peripheral configuration and etc.

the port status should be restored before the "CONTROL.IOR" being '0' for that the I/O status may be in the unstable(unknown) status which can make the glitch on the external port.

[Caution]

The operation frequency of the backup RAM is restricted to the half of memory bus frequency.
Therefore, If the maximum memory bus freuency at specific operating voltage is F_{MBUS}, Maximum operation frequency of the backup RAM at that specific operation voltage is restricted to F_{MBUS}/2. (This means that during the backup RAM access, memory bus frequency should be maintained =< F_{MBUS}/2)

3.4.12 Power On/Off Sequence for Sub Bus

3.4.12.1 Power On/Off Sequence for the Video Bus

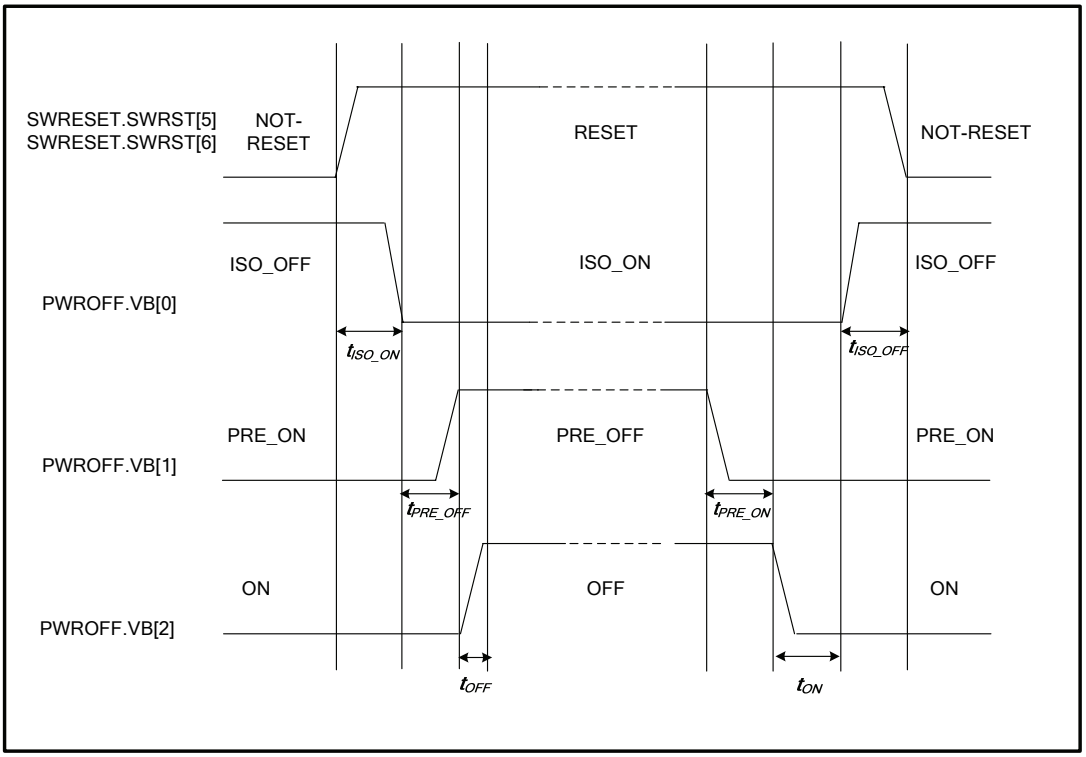


Figure 3.15 Power-On/Off Sequence for the Video Bus

The Video Bus of NVS2310 can be turned off to reducing the power consumption. The above figure shows the power on/off sequence controlled by the software. The SWRESET.SWRST[5] and [6] are described in the CKC.

Before entering to or exiting from the power-off state, the video bus should be in “RESET” state. If the video bus is in the non RESET state during the power-off state, the system can be disturbed by signals from the video bus.

PWROFF.VB[0] is “active-low” signal and PWROFF.VB[1:0] are “active_high” signals.

Table 3.2 Power Timing Parameter for Video Bus

Timing Parameter	Recommeded Value			Unit
	Min	Typ	Max	
t_{ISO_ON}	50			ns
t_{PRE_OFF}		TBD		
t_{OFF}		TBD		
t_{ON}		TBD		
t_{PRE_ON}		TBD		
t_{ISO_OFF}	50			ns

3.4.12.2 Power On/Off Sequence for the Graphic Bus

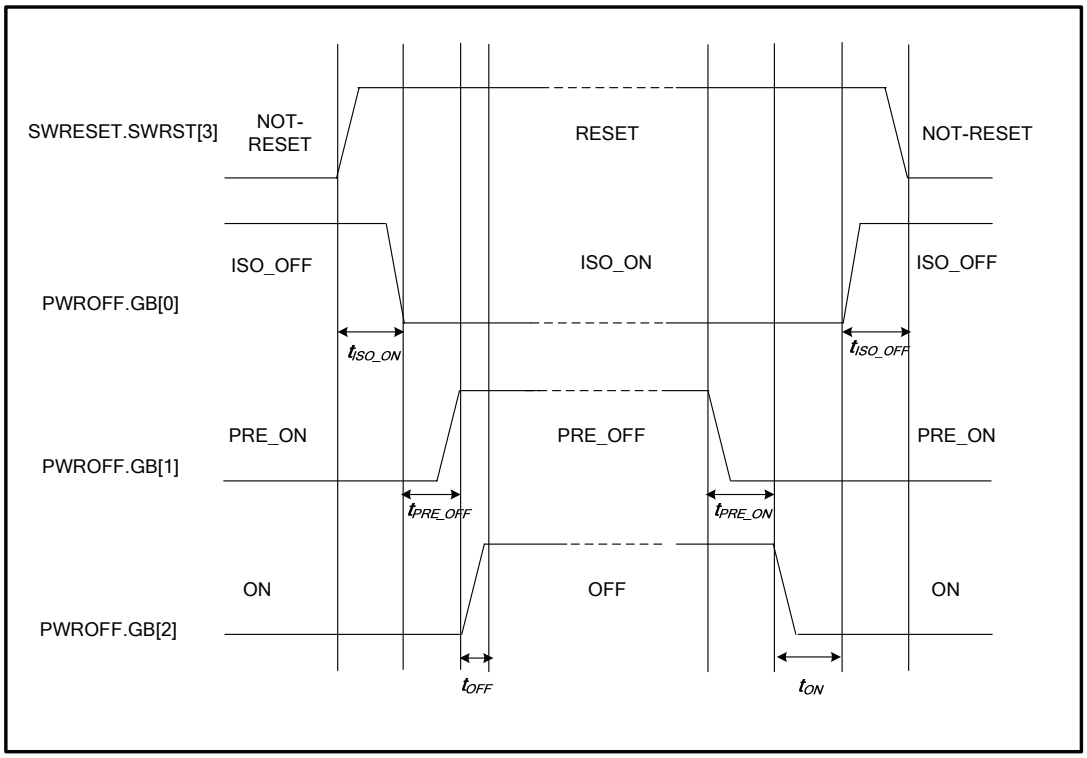


Figure 3.16 Power-On/Off Sequency for the Graphic Bus

The Graphic Bus inside the processor can be turned off to reducing the power consumption. The above figure shows the power on/off sequence controlled by the software. The SWRESET.SWRST[3] is described in the CKC.

Before entering to or exiting from the power-off state, the video bus should be in “RESET” state. If the video bus is in the non RESET state during the power-off state, the system can be disturbed by signals from the video bus.

PWROFF.GB[0] is “active-low” signal and PWROFF.GB[1:0] are “active_high” signals.

Table 3.3 Power Timing Parameter for Graphic Bus

Timing Parameter	Recommeded Value			Unit
	Min	Typ	Max	
t_{ISO_ON}	50			ns
t_{PRE_OFF}		TBD		
t_{OFF}		TBD		
t_{ON}		TBD		
t_{PRE_ON}		TBD		
t_{ISO_OFF}	50			ns

3.4.12.3 Power On/Off Sequence for the DDI (Display) Bus

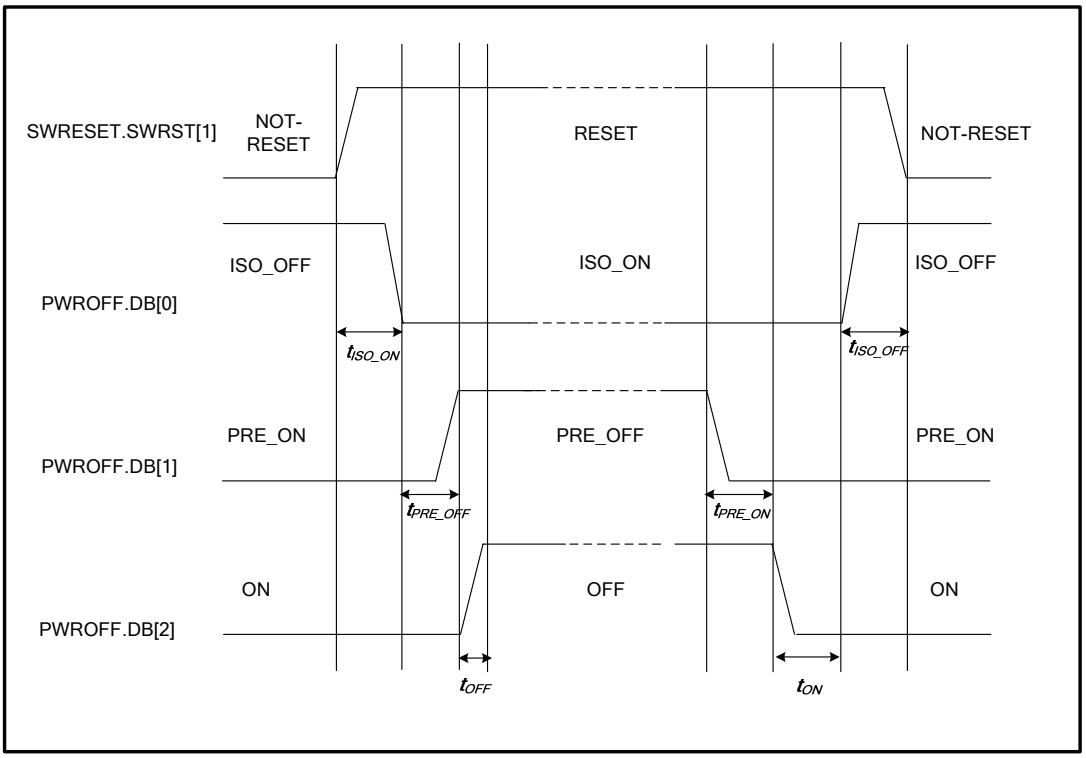


Figure 3.17 Power-On/Off Sequency for the DDI Bus

The DDI Bus inside the processor can be turned off to reducing the power consumption. The above figure shows the power on/off sequence controlled by the software. The SWRESET.SWRST[1] is described in the CKC.

Before entering to or exiting from the power-off state, the video bus should be in “RESET” state. If the video bus is in the non RESET state during the power-off state, the system can be disturbed by signals from the video bus.

PWROFF.DB[0] is “active-low” signal and PWROFF.DB[1:0] are “active_high” signals.

Table 3.4 Power Timing Parameter for DDI Bus

Timing Parameter	Recommeded Value			Unit
	Min	Typ	Max	
t_{ISO_ON}	50			ns
t_{PRE_OFF}		TBD		
t_{OFF}		TBD		
t_{ON}		TBD		
t_{PRE_ON}		TBD		
t_{ISO_OFF}	50			ns

3.4.12.4 Power On/Off Sequence for the DS (Display Sub) Bus

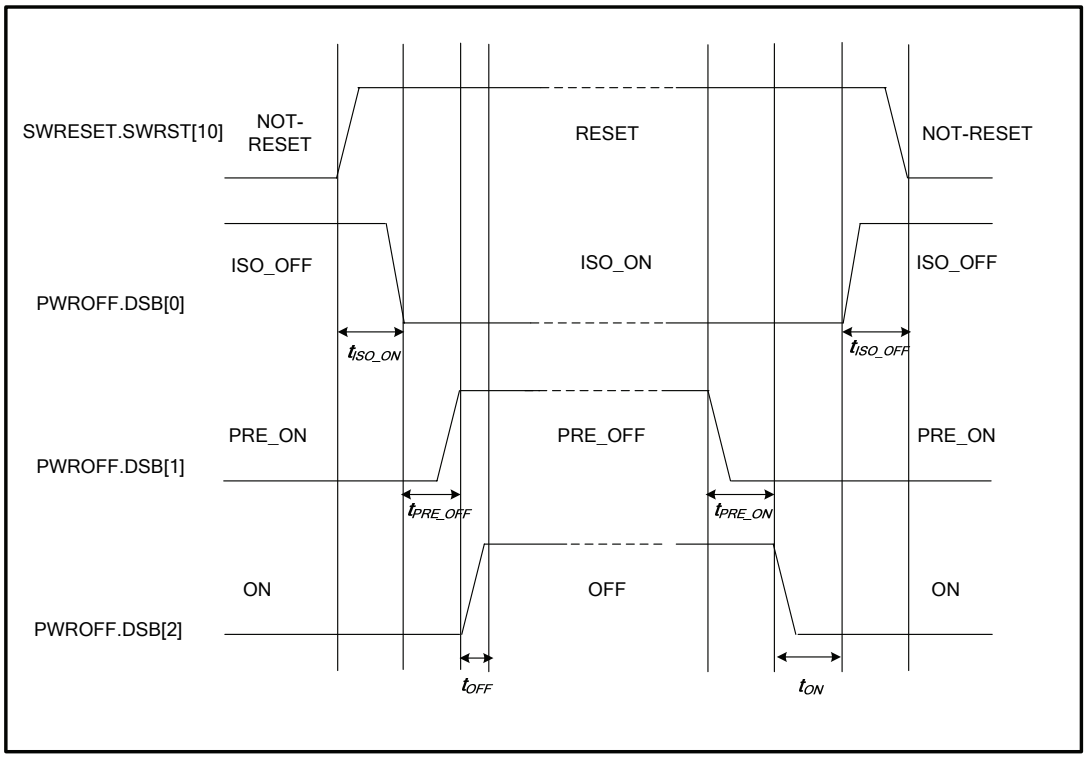


Figure 3.18 Power-On/Off Sequency for the DS Bus

The DS Bus inside the processor can be turned off to reducing the power consumption. The above figure shows the power on/off sequence controlled by the software. The SWRESET.SWRST[10] is described in the CKC.

Before entering to or exiting from the power-off state, the video bus should be in “RESET” state. If the video bus is in the non RESET state during the power-off state, the system can be disturbed by signals from the video bus.

PWROFF.DSB[0] is “active-low” signal and PWROFF.DSB[1:0] are “active_high” signals.

Table 3.5 Power Timing Parameter for DS Bus

Timing Parameter	Recommeded Value			Unit
	Min	Typ	Max	
t_{ISO_ON}	50			ns
t_{PRE_OFF}		TBD		
t_{OFF}		TBD		
t_{ON}		TBD		
t_{PRE_ON}		TBD		
t_{ISO_OFF}	50			ns

3.4.12.5 Power On/Off Sequence for the HSIO (High Speed IO) Bus

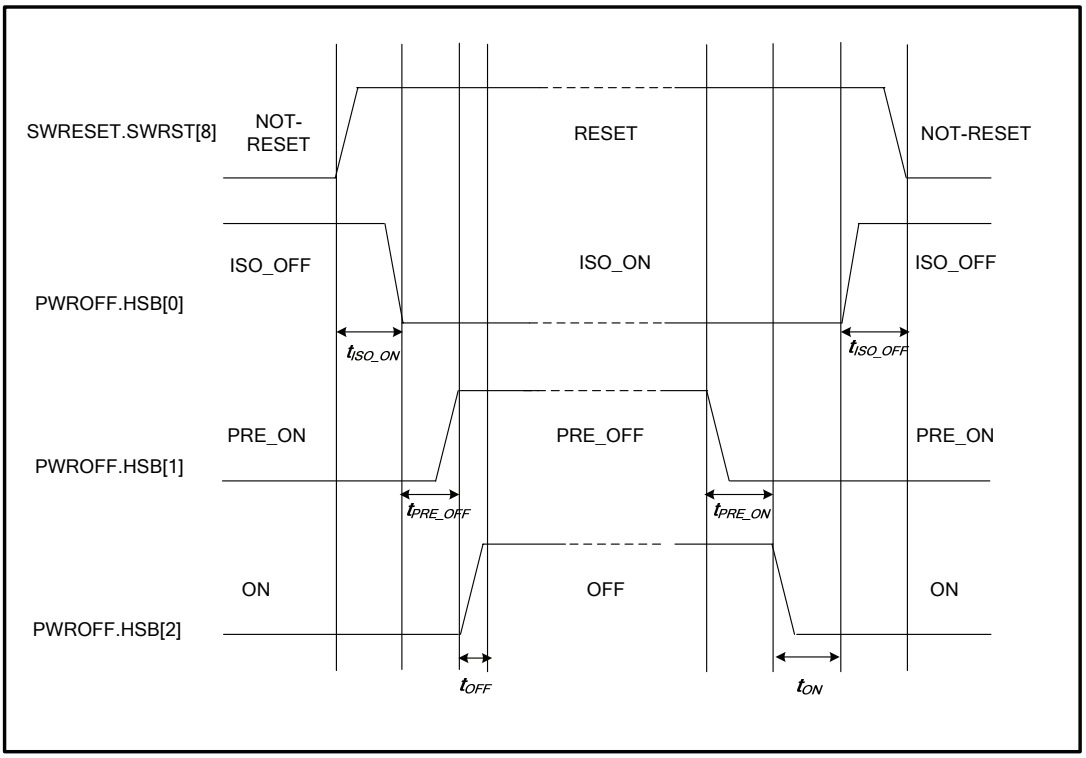


Figure 3.19 Power-On/Off Sequency for the HSIO Bus

The HSIO Bus inside the processor can be turned off to reducing the power consumption. The above figure shows the power on/off sequence controlled by the software. The SWRESET.SWRST[8] is described in the CKC.

Before entering to or exiting from the power-off state, the video bus should be in “RESET” state. If the video bus is in the non RESET state during the power-off state, the system can be disturbed by signals from the video bus.

PWROFF.HSB[0] is “active-low” signal and PWROFF.HSB[1:0] are “active_high” signals.

Table 3.6 Power Timing Parameter for HSIO Bus

Timing Parameter	Recommeded Value			Unit
	Min	Typ	Max	
t_{ISO_ON}	50			ns
t_{PRE_OFF}		TBD		
t_{OFF}		TBD		
t_{ON}		TBD		
t_{PRE_ON}		TBD		
t_{ISO_OFF}	50			ns

3.4.12.6 Power On/Off Sequence for the Camera Bus

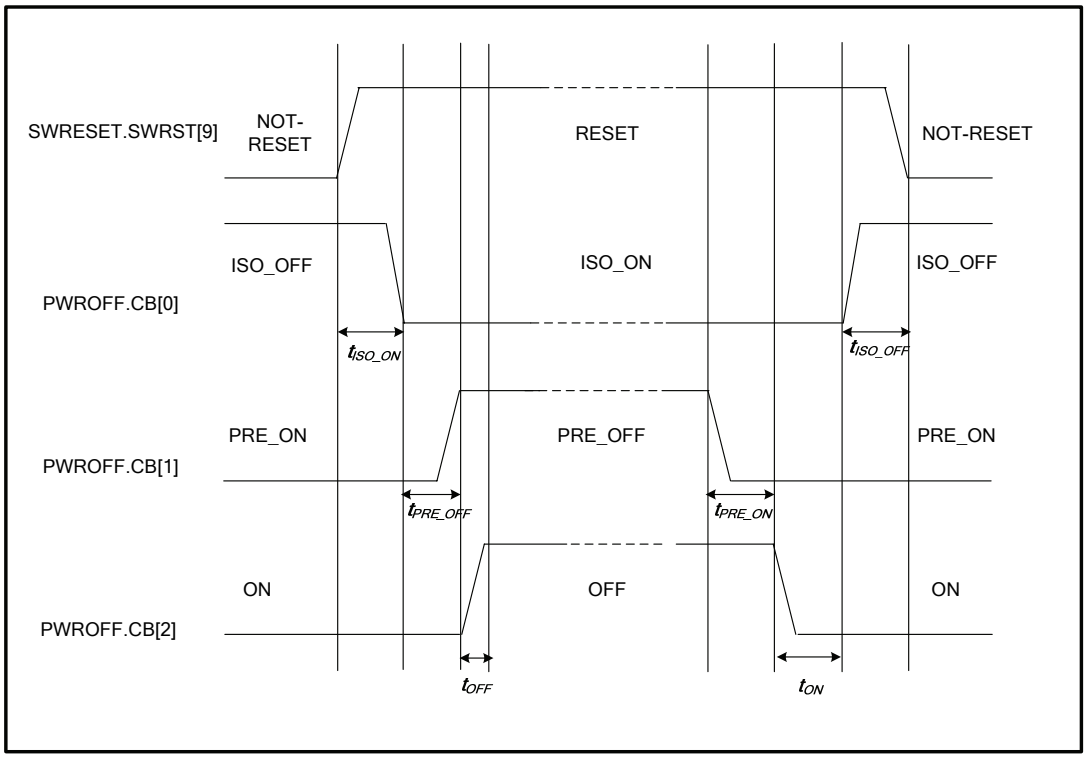


Figure 3.20 Power-On/Off Sequency for the Camera Bus

The Camera Bus inside the processor can be turned off to reducing the power consumption. The above figure shows the power on/off sequence controlled by the software. The SWRESET.SWRST[9] is described in the CKC.

Before entering to or exiting from the power-off state, the video bus should be in “RESET” state. If the video bus is in the non RESET state during the power-off state, the system can be disturbed by signals from the video bus.

PWROFF.CB[0] is “active-low” signal and PWROFF.CB[1:0] are “active_high” signals.

Table 3.7 Power Timing Parameter for Camera Bus

Timing Parameter	Recommeded Value			Unit
	Min	Typ	Max	
t_{ISO_ON}	50			ns
t_{PRE_OFF}		TBD		
t_{OFF}		TBD		
t_{ON}		TBD		
t_{PRE_ON}		TBD		
t_{ISO_OFF}	50			ns

3.4.12.7 Power On/Off Sequence for the DDR Bus

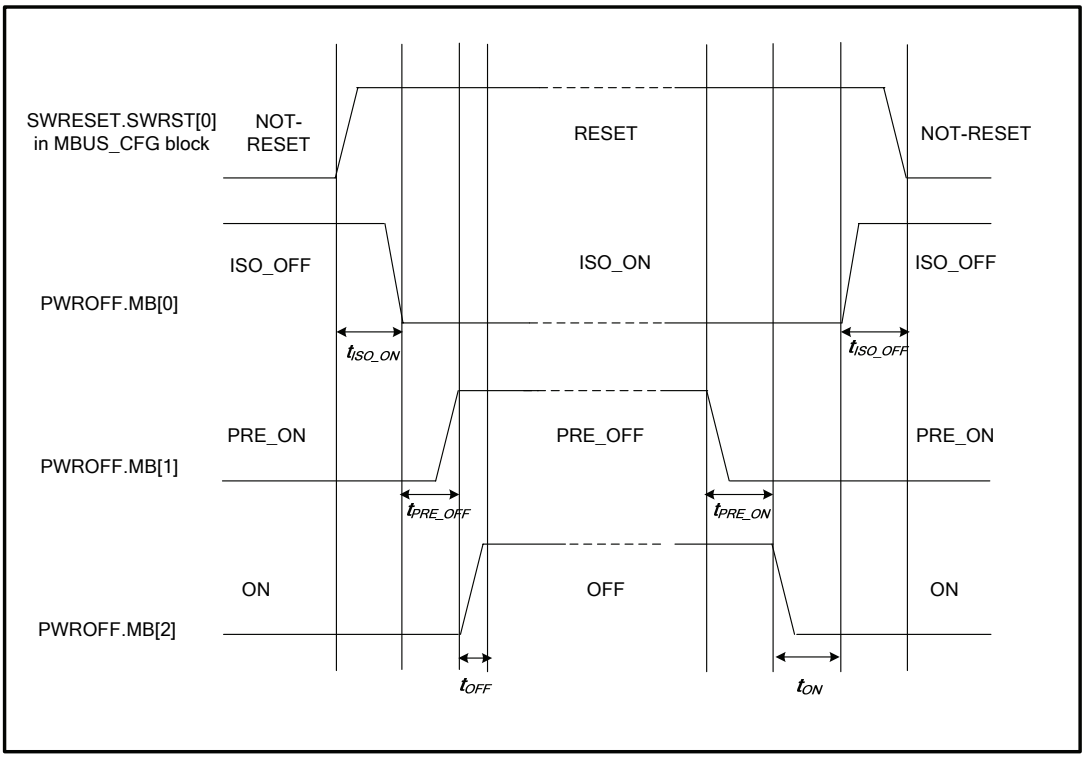


Figure 3.21 Power-On/Off Sequency for the DDR Bus

The DDR Bus inside the processor can be turned off to reducing the power consumption. The above figure shows the power on/off sequence controlled by the software. The SWRESET.SWRST[0] is described in the **MBUS CFG** block.

Before entering to or exiting from the power-off state, the video bus should be in “RESET” state. If the video bus is in the non RESET state during the power-off state, the system can be disturbed by signals from the video bus.

PWROFF.MB[0] is “active-low” signal and PWROFF.MB[1:0] are “active_high” signals.

Table 3.8 Power Timing Parameter for DDR Bus

Timing Parameter	Recommeded Value			Unit
	Min	Typ	Max	
t_{ISO_ON}	50			ns
t_{PRE_OFF}		TBD		
t_{OFF}		TBD		
t_{ON}		TBD		
t_{PRE_ON}		TBD		
t_{ISO_OFF}	50			ns

[Caution]

Software Reset Control register for DDR bus is not located at CKC block but MBUS_CFG block.

3.4.12.8 Power On/Off Sequence for the HDMI PHY

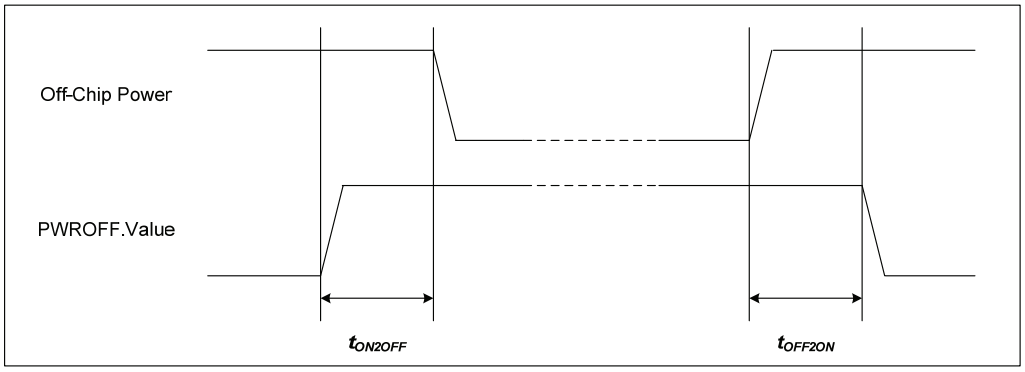


Figure 3.22 Power-On/Off Sequency for the HDMI PHY

The above figure shows the power on/off sequence for the HDMI PHY. The “Off-Chip Power” means the power group for the HDMI PHY.

Before turning off the power of the “Off-Chip Power”, the “PWROFF.HD” field should be high and should be low after turning on the power of the “Off-Chip Power”.

The “t_{ON2OFF}” and “t_{OFF2ON}” fields should be greater than 100us. And the “Off-Chip Power” can be controlled by the GPIO and by the external device.

3.4.12.9 Power On/Off Sequence for the USB0/USB1 PHY

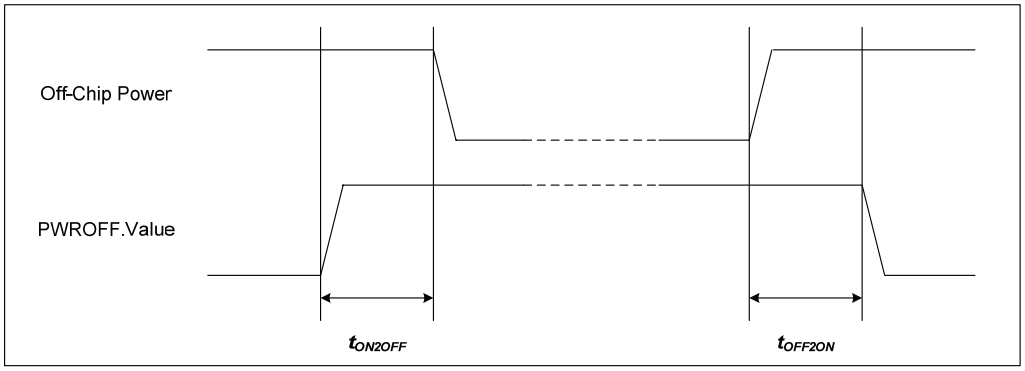


Figure 3.23 Power-On/Off Sequency for the USB0/USB1 PHY

The above figure shows the power on/off sequence for the USB0/USB1 PHY. The “Off-Chip Power” means the power group for the USB0/USB1 phy .

Before turning off the power of the “Off-Chip Power”, the “PWROFF.UP0” or “PWROFF.UP1” field should be high and should be low after turning on the power of the “Off-Chip Power”.

The “t_{ON2OFF}” and “t_{OFF2ON}” fields should be greater than 100us. And the “Off-Chip Power” can be controlled by the GPIO and by the external device.

4 SMU_I2C

4.1 Overview

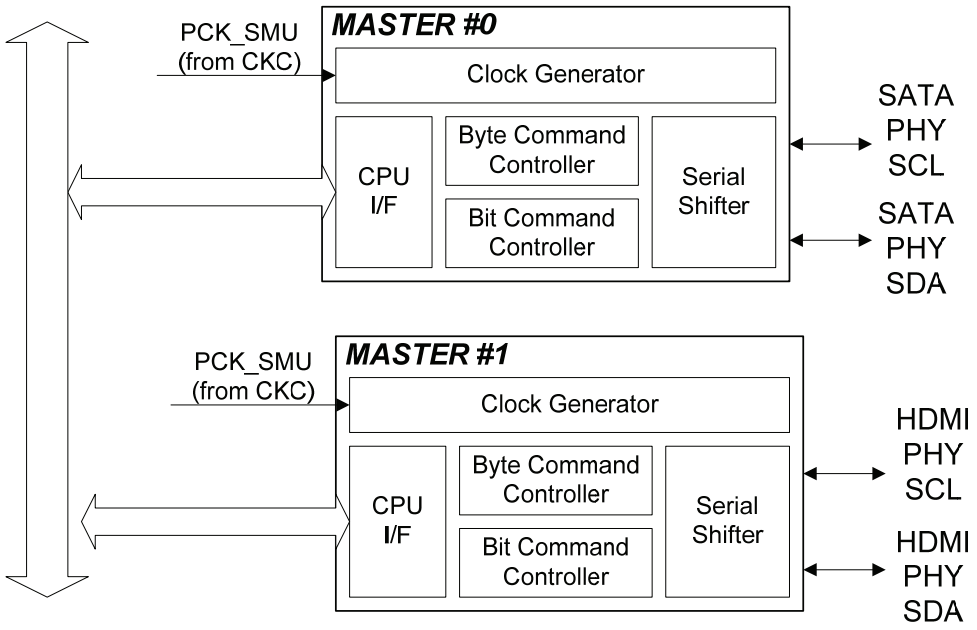


Figure 4.1 I2C Block Diagram

The SMU_I2C controller in the NVS2310 has two masters controller. The each master can control the dedicated I2C device, master 1 for HDMI PHY.

4.2 Register Descriptions

Table 4.1 SMU_I2C Register Map (Base Address = 0xB0505000)

Ch	Name	Addr. Offset	Type	Reset	Description
Master 0	PRES	0x00	R/W	0xFFFF	Clock Prescale register
	CTRL	0x04	R/W	0x0000	Control Register
	TXR	0x08	W	0x0000	Transmit Register
	CMD	0x0C	W	0x0000	Command Register
	RXR	0x10	R	0x0000	Receive Register
	SR	0x14	R	0x0000	Status Register
	TIME	0x18	R/W	0x0000	Timing Control Register
Master 1	PRES	0x40	R/W	0xFFFF	Clock Prescale register
	CTRL	0x44	R/W	0x0000	Control Register
	TXR	0x48	W	0x0000	Transmit Register
	CMD	0x4C	W	0x0000	Command Register
	RXR	0x50	R	0x0000	Receive Register
	SR	0x54	R	0x0000	Status Register
	TIME	0x58	R/W	0x0000	Timing Control Register
Common	ICLK	0x80	R/W	0x0010	I2C_SCL divider Register

Prescale Register (PRES) — 0xB0505000, 0xB0505040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Clock Prescale data															

Field	Name	RW	Reset	Description
15-0	Pres	R/w	R/W	Prescaler Value

This register is used to prescale the SCL clock line. Due to the structure of the I2C interface, the core uses a 5*SCL clock internally. The prescale register must be programmed to this 5*SCL frequency (minus 1). Change the value of the prescale register only when 'EN' bit is cleared.

Example :
CLK Input frequency = 8MHz , Desired SCL frequency = 100KHz
Prescale = (8MHz / 100KHz) – 1 = 15

Control Register (CTRL) — 0xB0505004, 0xB0505044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								EN	IEN	MOD	RESERVED				

Field	Name	RW	Reset	Description
7	EN	R/W	0x0000	0 : Disable 1 : Enable
6	IEN	R/W	0x0000	0 : Disable 1 : Enable
5	MOD	R/W	0x0000	0 : 8bit Mode 1 : 16bit Mode

Transmit Register (TXR) — 0xB0505008, 0xB0505048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Transmit Data															

When CTRL[5] is set, in case of 16Bit Mode is selected, Transmit Data bit width become 16 bit. Default mode is 8bit mode.

Field	Name	RW	Reset	Description
15-0	Transmit Data	R/W	0x0	Transmit Data

Command Register (CMD) 0xB050500C, 0xB050504C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								STA	STO	RD	WR	ACK	RESERVE		IACK

Field	Name	RW	Reset	Description
7	STA	W	0x0000	0 : Disable 1 : Enable
6	STO	W	0x0000	0 : Disable 1 : Enable
5	RD	W	0x0000	0 : Disable 1 : Enable
4	WR	W	0x0000	0 : Disable 1 : Enable
3	ACK	W	0x0000	0 : Disable 1 : Enable
0	IACK	W	0x0000	0 : - 1 : Clear a pending interrupt

Receive Register (RXR) 0xB0505010, 0xB0505050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Receive Data															

Field	Name	RW	Reset	Description
15-0	Receive Data	R	-	Received Data

When CTRL[5] is set, in case of 16Bit Mode is selected, Transmit Data bit width become 16 bit. Default mode is 8bit mode.

Status Register (SR) 0xB0505014, 0xB0505054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RxACK	BUSY	AL				TIP	IF

Field	Name	RW	Reset	Description
7	RxACK	R	0x0000	0 : Acknowledge received 1 : No Acknowledge received
6	BUSY	R	0x0000	0 : ‘0’ after STOP signal detected 1 : ‘1’ after START signal detected
5	AL	R	0x0000	0 : The core does not lose arbitration 1 : The core loses arbitration
1	TIP	R	0x0000	0 : Transfer Complete 1 : Transferring Data
0	IF	R	0x0000	0 : - 1 : Interrupt is pending

Timing Register (TIME)

0xB0505018, 0xB0505058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RC								-		CKSEL	FC				

Field	Name	RW	Reset	Description
15-8	RC	R/W	0x0000	Value : “0” disables recovery time counter. The recovery time counter is enabled and loaded with RC[7:0] whenever a STOP condition is issued by the core. Execution of a new command written to CMD register is delayed until the counter is expired. Actual wait time = (I2CCLK period) * PRES[15:0] * 5 * RC[7:0]
5	CKSEL	R/W	0x0000	0 : I2CCLK from Clock controller 1 : PCLK (HCLK) divided by 2
4-0	FC	R/W	0x0000	Value : “0” disables noise filter. SCL and SDA inputs are checked for stability until the counter is expired.

I2C_SCL divider Register (ICLK)

0xB0505080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT_EN	0														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER															

Field	Name	RW	Reset	Description
31	CNT_EN	R/W	0x0010	I2C clock divider counter enable to generate I2C_SCL
15-0	DIVIDER	R/W	0x0010	I2C clock divider value to generate I2C_SCL

5 RTC (Real-Time Clock)

5.1 Overview

The Real Time Clock (RTC) unit can be operated by the backup battery if the system power is turned off. The RTC provides time keeping data to CPU as BCD (binary coded decimal) values. The data includes second, minute, hour, date, day of the week, month, and year. The RTC unit also can perform the alarm function. The block diagram is shown in Figure 16.1.

Features

- Clock and calendar functions (BCD display): seconds, minutes, hours, date, day of week, month, year
- Leap year generator
- Wake-up (PMWKUP) signal generation: support on the power down mode (PWDN)
- Alarm interrupt (ALMINT) in normal operation mode
- Supports 32.768kHz XTAL input
- Power Supply Voltage: System power supply(3.0V, 1.2V), Backup battery (3.0V)

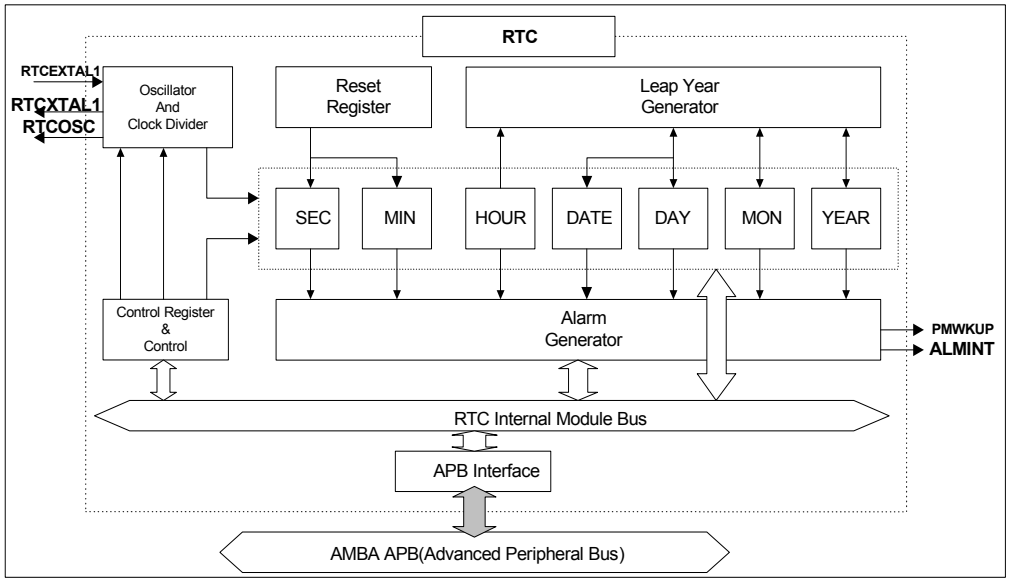


Figure 5.1 RTC Block Diagram

5.2 Function Description

5.2.1 System Clock Frequency Control

The leap year generator calculates whether the last date of each month is 28, 29, 30 or 31 based on data from BCDDAY, BCDMON, and BCDYEAR. This also considers leap years in deciding the last date. A 16 bit counter can just represent four BCD digits, so it can decide whether any year is a leap year or not.

5.2.2 System Power Operation

It is required to set bit 1 of the RTCCON register for interfacing between CPU and RTC logic. An one second error can occur when the CPU reads or writes data into BCD counters and this can cause the change of the higher time units. When the CPU reads/writes data to/from the BCD counters, another time unit may be changed if BCDSEC register is overflowed. To avoid this problem, the CPU should reset BCDSEC register to 00h. The reading sequence of the BCD counters is BCDYEAR, BCDMON, BCDDATE, BCDDAY, BCDHOUR, BCDMIN, and BCDSEC. It is required to read it again from BCDYEAR to BCDSEC if BCDSEC is zero.

5.2.3 Backup Battery Operation

The RTC logic is driven by backup battery if the system power turns off. The interfaces of the CPU and RTC logic are blocked and the battery drives the oscillation circuit, clock divider (DIVIDER), and BCD counters (RTCFCN) to minimize power dissipation.

5.2.4 Alarm Function

The RTC generates alarm signal at specified time in the power down mode or normal operation mode. In normal operation mode, the alarm interrupt (ALMINT) is activated and in the power down mode the power management wake up (PMWKUP) signal is activated. The RTC alarm register, RTCALM, determines the alarm enable and the condition of the alarm time setting.

5.3 RTC Operation

5.3.1 Boot-Up Sequence

The figure below shows how to initialize register of RTCINT block (RTCALM, RTCIM, RTCPEND).

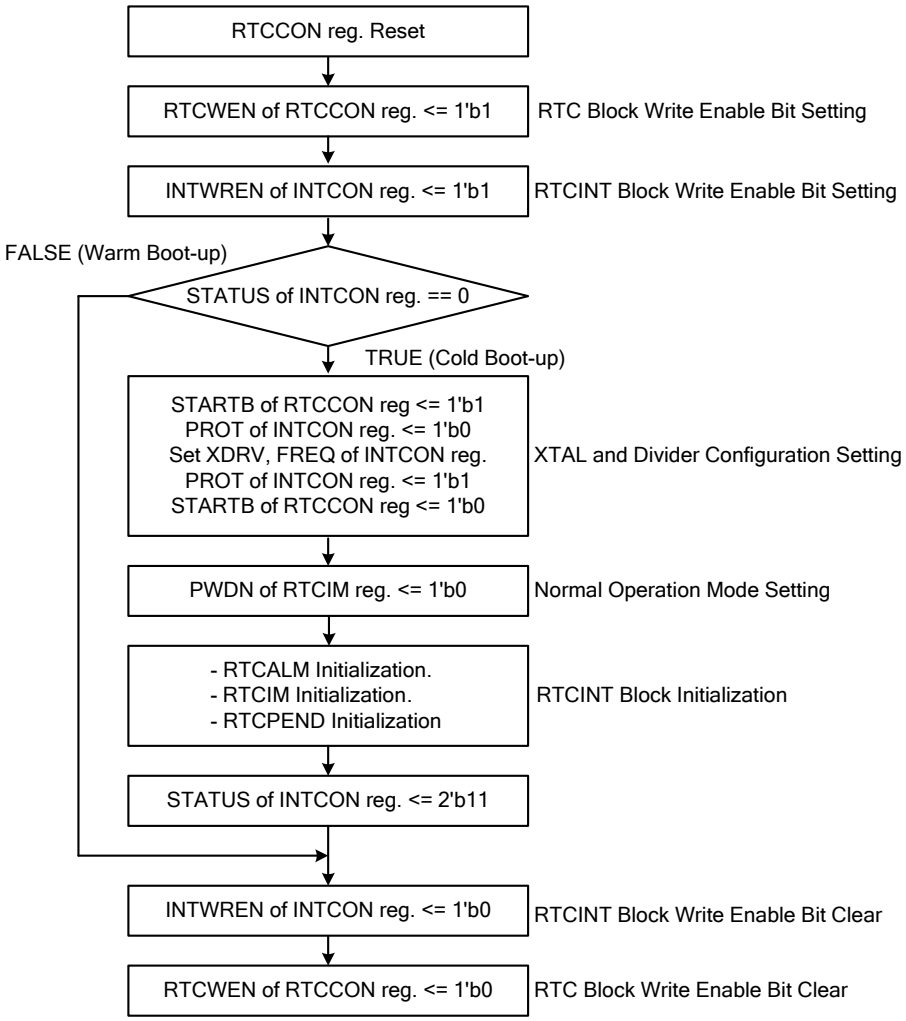


Figure 5.2 Boot-Up Sequence

5.3.2 RTC Time Setting

The following figure shows how to set the time when clock is stopped. This works when the entire calendar or clock is to be set.

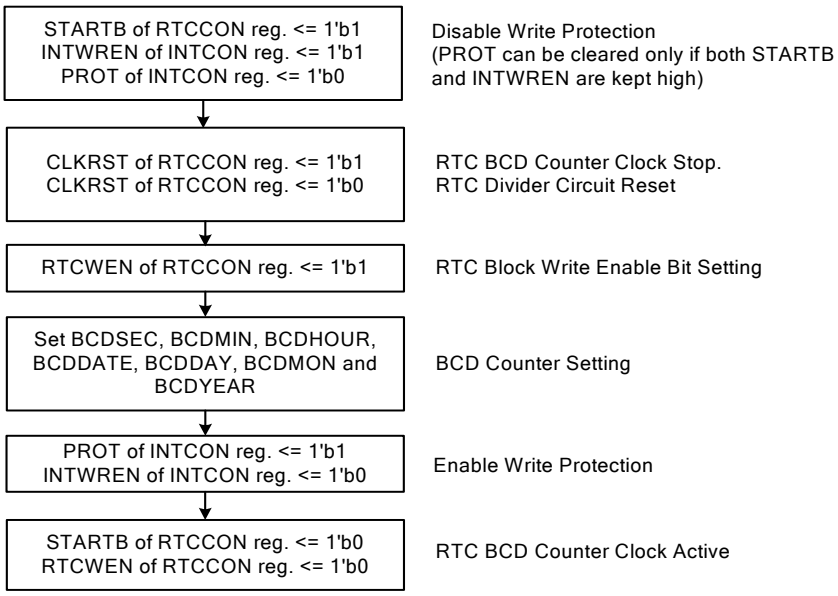


Figure 5.3 The RTC Time Setting Sequence

5.3.3 RTC Alarm Time Setting

The following figure shows how to use the alarm function. Alarms can be generated using the seconds, minutes, hours, days of week, date, month, year or any combination of these. Set the ALMEN bit (bit 7) in the register on which the alarm is placed to "1", and then set the alarm time. Clear the ALMEN bit in the register on which the alarm is placed to "0".

When the INTMODE bit of RTCIM register is high, and the clock and alarm times match, "1" is set in the PEND bit of RTCPEND register. The detection of alarm can be checked with reading the PEND bit.

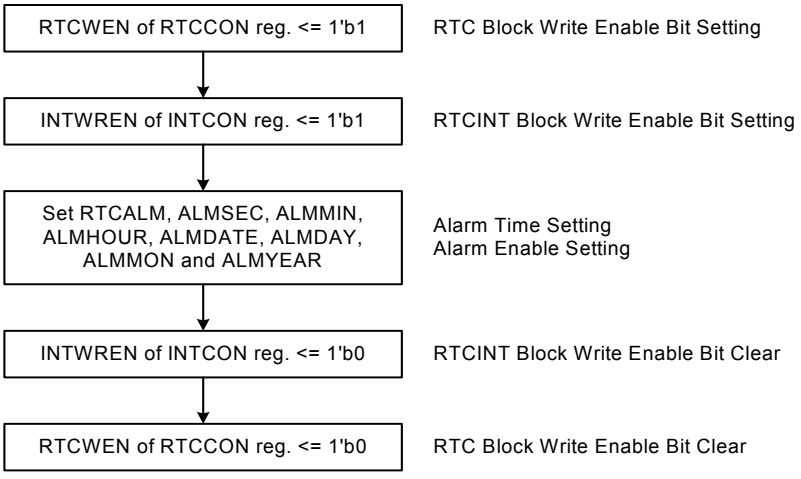


Figure 5.4 RTC Alarm Time Setting Sequence

Note: To generate the alarm interrupt, you must set INTWREN (INTCON[0])

5.3.4 RTCPEND Clear

The following figure shows how to Clear RTC interrupt

There are two types of interrupt.

- Alarm Interrupt (ALMINT).
- Wake-Up Interrupt (WKUPINT).

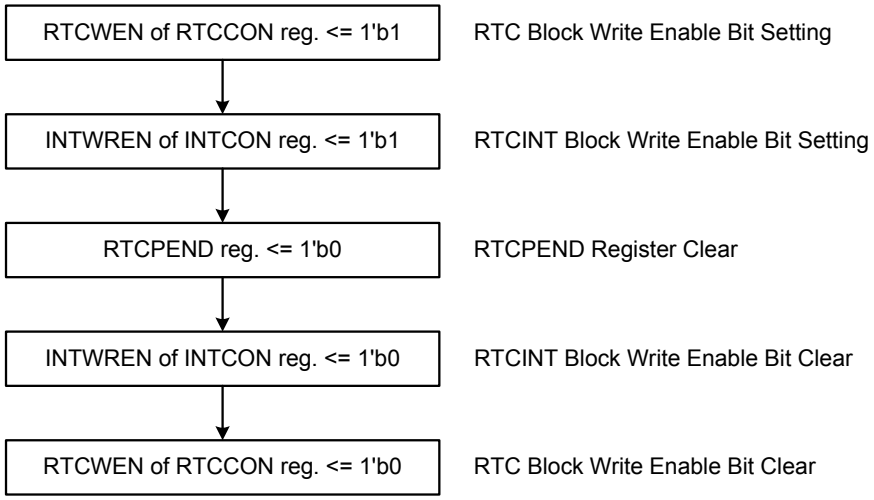


Figure 5.5 PEND Clear Sequence

5.3.5 RTC Operation

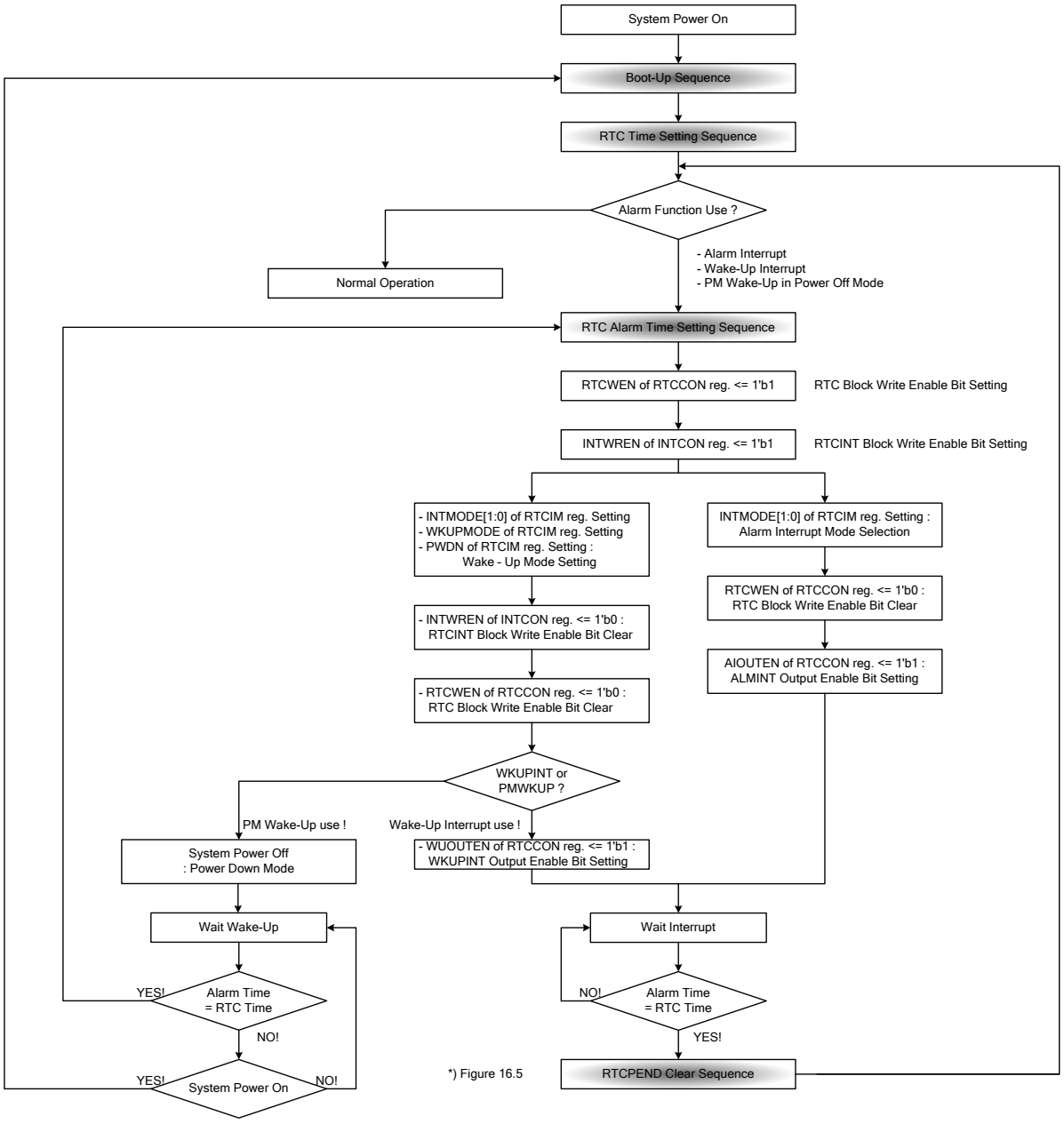


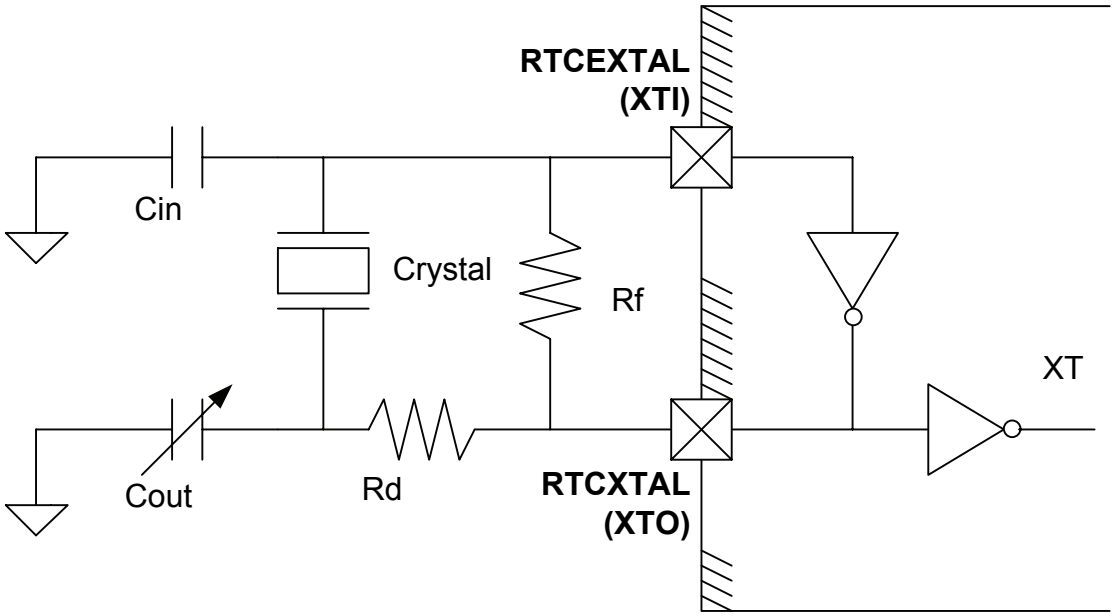
Figure 5.6 RTC Operation Process Flow Chart.

5.3.6 Crystal Oscillator Circuit

Crystal oscillator circuit constants (recommended values) are shown in the following table and the RTC crystal oscillator circuit in the following figure.

Table 5.1 Recommended Oscillator Circuit Constants

fosc	C1	C2	Rf
32.768 kHz	from 10 pF to 22 pF	from 10 pF to 22 pF	> 5MΩ



requirements such as frequency range, degree of stability, etc.

Notes:
1. Select either the Cin or Cout side for frequency adjustment variable capacitor according to
2. Built-in resistance value Rf(Typ value) = 5 ~10 Mega Ohm
3. Cin and Cout values include floating capacitance due to writing. Take care when using a ground plane.
4. The crystal oscillation setting time depends on the mounted circuit constants, floating capacitance, etc., and should be decided after consultation with the crystal resonator manufacturer.
5. Place the crystal resonator and load capacitance Cin and Cout as close as possible to the chip. (Correct oscillation may not be possible if there is externally included noise in RTCEXTAL1 and RTCXTAL1 pins.)
6. Ensure that the crystal resonator connection pin (RTCEXTAL1, RTCXTAL1) wiring is routed as far away as possible from other power lines (except GND) and signal lines.
7. Rd is an optional output resistance to improve power consumption and performance.

Figure 5.7 Example of Crystal Oscillator Circuit Connection

5.4 Programmer's Model

5.4.1 Register Memory Map

Table 5.2 RTC Register Map (Base Address = 0xB0502000)

Register	Address	R/W	Reset value	Description
RTCCON	0x00	R/W	0x00	RTC Control Register
INTCON	0x04	R/W	-	RTC Interrupt Control Register
RTCALM	0x08	R/W	-	RTC Alarm Control Register
ALMSEC	0x0C	R/W	-	Alarm Second Data Register
ALMMIN	0x10	R/W	-	Alarm Minute Data Register
ALMHOUR	0x14	R/W	-	Alarm Hour Data Register
ALMDATE	0x18	R/W	-	Alarm Date Data Register
ALMDAY	0x1C	R/W	-	Alarm Day of Week Data Register
ALMMON	0x20	R/W	-	Alarm Month Data Register
ALMYEAR	0x24	R/W	-	Alarm Year Data Register
BCDSEC	0x28	R/W	-	BCD Second Register
BCDMIN	0x2C	R/W	-	BCD Minute Register
BCDHOUR	0x30	R/W	-	BCD Hour Register
BCDDATE	0x34	R/W	-	BCD Date Register
BCDDAY	0x38	R/W	-	BCD Day of Week Register
BCDMON	0x3C	R/W	-	BCD Month Register
BCDYEAR	0x40	R/W	-	BCD Year Register
RTCIM	0x44	R/W	-	RTC Interrupt Mode Register
RTCPEND	0x48	R/W	-	RTC Interrupt Pending Register
RTCSTR	0x4C	R/W	0x00	RTC Interrupt Status Register

5.4.2 Register Description

RTC Control Register (RTCCON) **0xB0502000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								WUOUTEN	AIOUTEN	OSCEN	CLKRST	CNTSEL	CLKSEL	RTCWEN	STARTB

RTCCON register consists of 8-bits such as STARTB that controls of running the normal counters, RTCWEN that controls the read/write enable of the BCD registers, CLKSEL, CNTSEL, and CLKRST for BCD counters testing.

RTCWEN bit controls all interfaces between the CPU and the RTC, so it should be set to '1' in an initialization routine to enable data transfer after a system reset. Instead of working BCD with 1Hz, CLKSEL bit enables the operation of BCD counters with an external clock which is applied through the pin EXTAL1 to the test BCD counters. CNTSEL bit converts the dependent

The operation of BCD counters into independent counters for the test. CLKRST resets the frequency divided logic in the RTC.

OSCEN bit controls the path from input of Crystal to the output of divider logic. If this bit is high, the output of divider is 1 Hz clock. This bit is implemented to test the oscillator circuit and divider block.

INTWREN bit controls the path from the RTCIF Block to the RTCINT Block.

NOTE: CLKRST, CNTSEL and CLKSEL are affected by PROT and INTWERN of INTCON register. They are valid only if (~PROT & INTWREN & STARTB)

Field	Name	RW	Reset	Description
7	WUOUTEN	R/W	0	Wake Up Interrupt Output Enable to the CPU 0 Disable 1 Enable if PWDN == 1
6	AIOUTEN	R/W	0	Alarm Interrupt Output Enable to the CPU 0 Disable 1 Enable if PWDN == 1
5	OSCEN	R/W	0	Oscillator and Divider Circuit Test Enable 0 Disable 1 Enable
4	CLKRST	R/W	0	RTC Clock Count(Divider) Reset 0 No reset (Normal operation) 1 Enable Reset if (~PROT & INTWREN & STARTB) This bit resets the frequency divider logic in the RTC.
3	CNTSEL	R/W	0	BCD Count Test Type Select 0 Test Disable (Normal operation) 1 Test Enable if (~PROT & INTWREN & STARTB) This bit forces BCD counters into test mode.
2	CLKSEL	R/W	0	BCD Counter Test Clock Select 0 1Hz clock (Normal operation) 1 XTI divided by 2 if (~PROT & INTWREN & STARTB)
1	RTCWEN	R/W	0	RTC Write Enable 0 Disable 1 Enable This bit controls all interfaces between the CPU and the RTC, so it should be set to '1' to enable data transfer.
0	STARTB	R/W	0	RTC Start Bit 0 Run (Normal operation) 1 Halt Note : If PROT bit of INTCON register is set as “1”, the BCD counters start normal operation with 1Hz clock regardless of STARTB status.

Note: If PROT bit of INTCON register is set as “1”, the BCD counters start normal operation with 1Hz clock regardless of STARTB status.

RTC Interrupt Control Register (INTCON) 0xB0502004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROT	0	Reserved		0	FSEL			0					STATUS		INTWREN

Field	Name	RW	Reset	Description
15	PROT	R/W	0	Protection Enable Bit 0 Disable 1 Enable. If enabled, the following fields are affected. - FSEL of INTCON register are write protected - All the BCD Data Registers are write protected - CLKRST, CNTSEL and CLKSEL bits of RTCCON reigister are made non-functional. To clear this bit, RTCWEN, STARTB and INTWREN must be set first. To enable this bit, RTCWEN and STARTB must be set first.
13-12	Reserved	-	-	-
10-8	FSEL	R/W	Undef.	Divider Output Select 3'b000 32.768kHz XTAL others reserved
2-1	STATUS	R/W	0	Reserved
0	INTWREN	R/W	0	Interrupt Block Write Enable Bit 0 Disable 1 Enable

*)To write INTCON register, you must set RTCWEN (RTCCON[1]).

FSEL bits can be written only if (RTCWEN & STARTB & INTWREN & ~PROT)

To enable PROT bit, RTCWEN and STARTB must be set first.

To clear PROT bit, RTCWEN, STARTB and INTWREN must be set first.

RTC Alarm Control Register (RTCALM) 0xB0502008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ALMEN	YEAREN	MONEN	DAYEN	DATEEN	HOUREN	MINEN	SECEN

RTCALM register determines the alarm enable and the condition of the alarm time setting. Note that RTCALM register generates the alarm signal through ALMINT in normal operation mode.

Alarms can be generated using the seconds, minutes, hours, day of week, date, month, year or any combination of these. Set the ALMEN bit (bit 7) in the register on which the alarm is placed to "1", and then set the alarm time. Clear the ALMEN bit in the register on which the alarm is placed to "0".

Field	Name	RW	Reset	Description
7	ALMEN	R/W	0	Alarm Global Enable Bit 0 Disable 1 Enable
6	YEAREN	R/W	0	Year Alarm Enable Bit 0 Disable 1 Enable
5	MONEN	R/W	0	Month Alarm Enable Bit 0 Disable 1 Enable
4	DAYEN	R/W	0	Day of week Alarm Enable Bit 0 Disable 1 Enable
3	DATEEN	R/W	0	Date Alarm Enable Bit 0 Disable 1 Enable
2	HOUREN	R/W	0	Hour Alarm Enable Bit 0 Disable 1 Enable
1	MINEN	R/W	0	Minute Alarm Enable Bit 0 Disable 1 Enable
0	SECEN	R/W	0	Second Alarm Enable Bit 0 Disable 1 Enable

Note: To Write RTCALM register, you must set RTCWEN (RTCCON[1]) and INTWREN (INTCON[0])

To Read RTCALM register, you must set INTWREN (INTCON[0]).

Alarm Second Data Register (ALMSEC) 0xB050200C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									SECDATA						

Field	Name	RW	Reset	Description
6-4	SECDATA	R/W	Undef.	Alarm Value for Second Bit Alarm BCD value from 0 to 5
3-0				Alarm Value for Second Bit Alarm BCD value from 0 to 9

Note: To Write ALMSEC register, you must set RTCWEN (RTCCON[1]) and INTWREN (INTCON[0])

To Read ALMSEC register, you must set INTWREN (INTCON[0]).

Alarm Minute Data Register (ALMMIN) 0xB0502010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									MINDATA						

Field	Name	RW	Reset	Description
6-4	MINDATA	R/W	Undef.	Alarm Value for Minute Bit Alarm BCD value from 0 to 5
3-0				Alarm Value for Minute Bit Alarm BCD value from 0 to 9

Note: To Write ALMMIN register, you must set RTCWEN (RTCCON[1]) and INTWREN (INTCON[0])

To Read ALMMIN register, you must set INTWREN (INTCON[0]).

Alarm Hour Data Register (ALMHOUR) 0xB0502014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										HOURDATA					

Field	Name	RW	Reset	Description
6-4	HOURDATA	R/W	Undef.	Alarm Value for Hour Bit Alarm BCD value from 0 to 2
3-0				Alarm Value for Hour Bit Alarm BCD value from 0 to 9

Note: To Write ALMHOUR register, you must set RTCWEN (RTCCON[1]) and INTWREN (INTCON[0])

To Read ALMHOUR register, you must set INTWREN (INTCON[0]).

Alarm Date Data Register (ALMDATE) 0xB0502018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										DATEDATA					

Field	Name	RW	Reset	Description
6-4	DATEDATA	R/W	Undef.	Alarm Value for Date Bit Alarm BCD value from 0 to 3
3-0				Alarm Value for Date Bit Alarm BCD value from 0 to 9

Note: To Write ALMDATE register, you must set RTCWEN (RTCCON[1]) and INTWREN (INTCON[0])

To Read ALMDATE register, you must set INTWREN (INTCON[0]).

Alarm Day of Week Data Register (ALMDAY) 0xB050201C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													DAYDATA		

Field	Name	RW	Reset	Description
2-0	DAYDATA	R/W	Undef.	Alarm Value for Day Bit 0 Sunday 1 Monday 2 Tuesday 3 Wednesday 4 Thursday 5 Friday 6 Saturday

Note: To Write ALMDAY register, you must set RTCWEN (RTCCON[1]) and INTWREN (INTCON[0])

To Read ALMDAY register, you must set INTWREN (INTCON[0]).

Alarm Month Data Register (ALMMON) 0xB0502020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											MONDATA				

Field	Name	RW	Reset	Description
4-0	MONDATA	R/W	Undef.	Alarm Value for Month Bits Alarm BCD value from 0 to 1

Note: To Write ALMMON register, you must set RTCWEN (RTCCON[1]) and INTWREN (INTCON[0])

To Read ALMMON register, you must set INTWREN (INTCON[0]).

Alarm Year Data Register (ALMYEAR) 0xB0502024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
YEARDATA															

Field	Name	RW	Reset	Description
15-0	YEARDATA	R/W	Undef.	Alarm Value for Year Bits Alarm BCD value from 0 to 99

Note: To Write ALMYEAR register, you must set RTCWEN (RTCCON[1]) and INTWREN (INTCON[0])

To Read ALMYEAR register, you must set INTWREN (INTCON[0]).

BCD Second Data Register (BCDSEC) 0xB0502028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									SECDATA						

Field	Name	RW	Reset	Description
6-4	SECDATA	R/W	Undef.	BCD Value for Second Bit BCD value from 0 to 5
3-0				Alarm Value for Second Bit BCD value from 0 to 9

Note: To Write BCDSEC register, you must set STARTB (RTCCON[0]) and RTCWEN (RTCCON[1]) and clear PROT(INTCON[15]).

To Read BCDSEC register, you must set INTWREN (INTCON[0]).

BCD Minute Data Register (BCDMIN) 0xB050202C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									MINDATA						

Field	Name	RW	Reset	Description
6-4	MINDATA	R/W	Undef.	BCD Value for Minute Bit BCD value from 0 to 5
3-0				BCD Value for Minute Bit BCD value from 0 to 9

Note: To Write BCDMIN register, you must set STARTB (RTCCON[0]) and RTCWEN (RTCCON[1]) and clear PROT(INTCON[15]).

To Read BCDMIN register, you must set INTWREN (INTCON[0]).

BCD Hour Data Register (BCDHOUR) 0xB0502030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										**HOURDATA**					

Field	Name	RW	Reset	Description
6-4	HOURDATA	R/W	Undef.	BCD Value for Hour Bit BCD value from 0 to 2
3-0				BCD Value for Hour Bit BCD value from 0 to 9

Note: To Write BCDHOUR register, you must set STARTB (RTCCON[0]) and RTCWEN (RTCCON[1]) and clear PROT(INTCON[15]).

To Read BCDHOUR register, you must set INTWREN (INTCON[0]).

BCD Date Data Register (BCDDATE) 0xB0502034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										DATEDATA					

Field	Name	RW	Reset	Description
6-4	DATEDATA	R/W	Undef.	BCD Value for Date Bit BCD value from 0 to 3
3-0				BCD Value for Date Bit BCD value from 0 to 9

Note: To Write BCDDATE register, you must set STARTB (RTCCON[0]) and RTCWEN (RTCCON[1]) and clear PROT(INTCON[15]).

To Read BCDDATE register, you must set INTWREN (INTCON[0]).

BCD Day of Week Data Register (BCDDAY) 0xB0502038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													DAYDATA		

Field	Name	RW	Reset	Description
2-0	DATEDAY	R/W	Undef.	BCD Value for Day Bit 0 Sunday 1 Monday 2 Tuesday 3 Wednesday 4 Thursday 5 Friday 6 Saturday

Note: To Write BCDDAY register, you must set STARTB (RTCCON[0]) and RTCWEN (RTCCON[1]) and clear PROT(INTCON[15]).

To Read BCDDAY register, you must set INTWREN (INTCON[0]).

BCD Month Data Register (BCDMON) 0xB050203C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											MONDATA				

Field	Name	RW	Reset	Description
4	MONDATA	R/W	Undef.	BCD Value for Month Bits BCD value from 0 to 1
3-0				BCD Value for Month Bits BCD value from 0 to 9

Note: To Write BCDMON register, you must set STARTB (RTCCON[0]) and RTCWEN (RTCCON[1]) and clear PROT(INTCON[15]).

To Read BCDMON register, you must set INTWREN (INTCON[0]).

BCD Year Data Register (BCDYEAR) 0xB0502040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
YEARDATA															

Field	Name	RW	Reset	Description
15-8	YEARDATA	R/W	Undef.	BCD Value for Year Bits BCD BCD value from 0 to 99
7-0				BCD Value for Year Bits BCD BCD value from 0 to 99

Note: To Write BCDYEAR register, you must set STARTB (RTCCON[0]) and RTCWEN (RTCCON[1]) and clear PROT(INTCON[15]).

To Read BCDYEAR register, you must set INTWREN (INTCON[0]).

RTC Interrupt Mode Register (RTCIM) 0xB0502044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												PWDN	WKUPMD	INTMODE	

Field	Name	RW	Reset	Description
3	PWDN	R/W	Undef.	Operation Mode Selection Bit 0 Normal operation mode 1 Power down mode (PMWKUP output enabled)
2	WKUPMODE	R/W	Undef.	Wakeup Mode Selection Bit 0 PMWKUP active low 1 PMWKUP active high
1-0	INTMODE	R/W	Undef.	Interrupt Mode Selection Bit 2'bx0 Disable alarm interrupt mode 2'bx1 Enable alarm interrupt mode 2'b01 Supports on the edge alarm interrupt 2'b11 Supports on the level alarm interrupt

When the INTMODE bit of RTCIM register is high, and the clock and alarm times match, "1" is set in the PEND bit of RTCPEND register. The detection of alarm can be checked with reading the PEND bit.

Note: To Write RTCIM register, you must set RTCWEN (RTCCON[1]) and INTWREN (INTCON[0])

To Read RTCIM register, you must set INTWREN (INTCON[0]).

To generate the alarm interrupt, you must be set INTMODE (RTCIM[0] = 1), PWDN (RTCIM [3] = 0) and RTCALM register should be set properly. To generate PMWKUP signal, PWDN register should be set to power down mode.

RTC Interrupt Pending Register (RTCPEND) 0xB0502048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															PEND

Field	Name	RW	Reset	Description
0	PEND	R/W	Undef.	Interrupt Pending Enable Bit 0 PEND bit is cleared 1 PEND bit is pending

Note: To Clear RTCPEND register, you must set RTCWEN (RTCCON[1]) and INTWREN (INTCON[0])

To Read RTCPEND register, you must set INTWREN (INTCON[0]).

RTC Interrupt Status Register (RTCSTR) 0xB050204C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								PRI	ALM	0		CNT			

Field	Name	RW	Reset	Description
7	PRI	R/W	0	RTC Wake-up interrupt pending 1 RTC wake-up interrupt was generated To clear this bit, write '1'
6	ALM	R/W	0	RTC Alarm interrupt pending 1 : RTC Alarm interrupt was generated To clear this bit, write '1'
3-0	CNT	R/W	0	Count value of alarm/wake-up interrupt

NOTE: RTC Interrupt = ALM | PRI

6 SMU Config

6.1 Register Description

Name	Address	Type	Reset	Description
HCLKMASK	0x00	R/W	0	Module Clock Mask Register
SWRESET	0x04	R/W	0	Module Software Reset Register
CFG_PLLFIN_SEL	0x08	R/W	0	PLL FIN Source Selection Register
CFG_RTC_WAIT	0x0C	R/W	0	RTC Bus Interface Wait Control Register

HCLKMASK **0xB0504000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											HCLKMASK0[4:0]				

Field	Name	RW	Reset	Description
4-0	HCLKMASK	R/W	0	Enable signals of HCLK clock gating logic AHB HCLKs, supplied to each SMU block, are controlled by clock enable signal. 0 : enable clock 1: disable clock The bit position indicates each sub-block which is controlled by. BIT 0: CKC BIT 1: TIMER BIT 2: PMU BIT 3: RTC BIT 4: I2C

SWRESET **0xB0504004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											SWRESET[4:0]				

Field	Name	RW	Reset	Description
4-0	SWRESET	R/W	0	Enable signals of SWRESET logic AHB SWRESETs, supplied to each peripheral block, are controlled by SWRESET signal. 0 : disable SWRESET 1: enable SWRESET The bit position indicates each sub-block which is controlled by. BIT 0: CKC BIT 1: TIMER BIT 2: PMU BIT 3: RTC BIT 4: I2C

CFG_PLLFIN_SEL 0xB0504008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						PLL4_FIN		PLL3_FIN		PLL2_FIN		PLL1_FIN		PLL0_FIN	

Field	Name	RW	Reset	Description
9-8	PLL4_FIN	R/W	0	PLL4 FIN source selection 00 : XIN (12 MHz) 01 : PLL0 FOUT 10 : PLL1 FOUT 11 : PLL2 FOUT
7-6	PLL3_FIN	R/W	0	PLL3 FIN source selection 00 : XIN (12 MHz) 01 : PLL5 FOUT 10 : PLL0 FOUT 11 : PLL1 FOUT
5-4	PLL2_FIN	R/W	0	PLL2 FIN source selection 00 : XIN (12 MHz) 01 : PLL4 FOUT 10 : PLL5 FOUT 11 : PLL0 FOUT
3-2	PLL1_FIN	R/W	0	PLL1 FIN source selection 00 : XIN (12 MHz) 01 : PLL3 FOUT 10 : PLL4 FOUT 11 : PLL5 FOUT
1-0	PLL0_FIN	R/W	0	PLL0 FIN source selection 00 : XIN (12 MHz) 01 : PLL2 FOUT 10 : PLL3 FOUT 11 : PLL4 FOUT

CFG_RTC_WAIT 0xB050400C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													RTC_WAIT		

Field	Name	RW	Reset	Description
2-0	RTC_WAIT	RW	0	RTC APB I/F wait cycle control This wait cycle control is needed because RTC bus I/F is operated slower than the overall bus system. ***Recommended setting value is "TBD"***

7 Boot Procedure

7.1 Power Up/Down Sequence for Core and I/O Power

The recommended power-up sequence for core and I/O power is described in the following figure

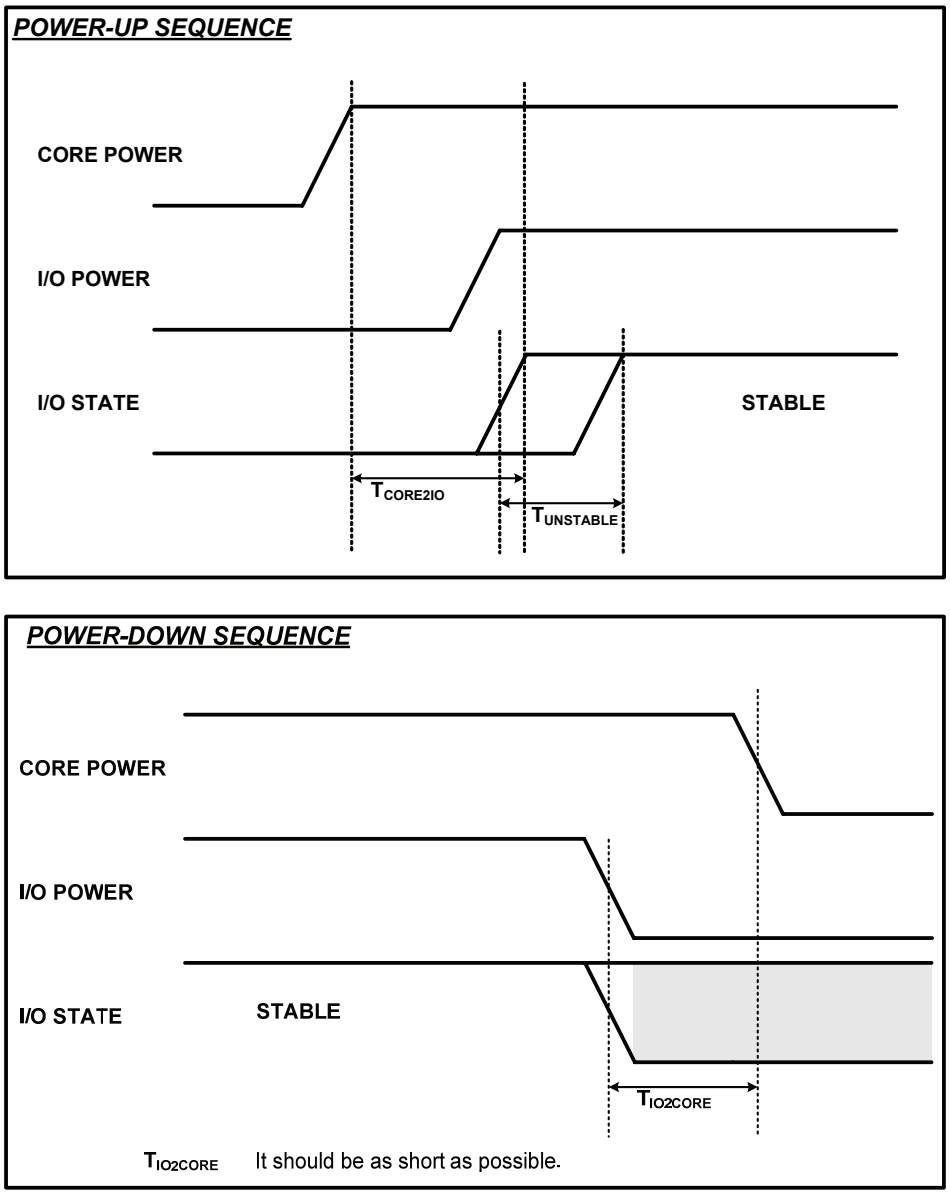


Figure 7.1 Power Up/Down Sequence for Core and I/O Power

Timing parameters in the above figure are as follows.

Name	Description	Min	Typ	Max
$T_{CORE2IO}$	It should be short as possible, namely the value should be greater or equal to 0ns.	0ns		
$T_{UNSTABLE}$	It is dependent on the application circuit. It can cause the meta-stability to CMOS I/O and GPIO connections. Sufficient power supply can make it shorter.	8ns		

* The minimum value of the $T_{UNSTABLE}$ is from the result of simulation, in case the supplied I/O power is ideal at 1.65 volt.

7.2 Boot Mode

The NVS2310 has the 5 pins for booting configuration with BM[3:0], GPIOG[27], GPIOE[25].

Table 7.1 Configuration Value

BM[3:0]	DESCRIPTION	COMMENTS
0000b	NAND Flash Boot Mode {GPIOG[27], GPIOG[25]} = 00b : NAND 8/16/32bits scan mode {GPIOG[27], GPIOG[25]} = 01b : NAND 8bits only mode {GPIOG[27], GPIOG[25]} = 10b : NAND 16/32bits scan mode {GPIOG[27], GPIOG[25]} = 11b : NAND 32bits scan mode	f_{CPU} : 288 MHz f_{IOBUS} : 144 MHz
0001b	I2C Master Boot (EEPROM Boot) Mode {GPIOG[27]} = 0 : I2C channel0 & port0 boot {GPIOG[27]} = 1 : I2C channel0 & port1 boot {GPIOG[25]} = 0 : If I2C boot failed, go to UART channel 0 boot {GPIOG[25]} = 1 : If I2C boot failed, go to UART channel 1 boot	f_{CPU} : 288 MHz f_{IOBUS} : 144 MHz f_{I2C} : 3 MHz f_{SCK} : 300 KHz
0010b	Serial Flash Boot (Serial EEPROM Boot) Mode {GPIOG[27]} = 0 : If boot failed, go to UART boot. {GPIOG[27]} = 1 : If boot failed, go to USB boot. {GPIOG[25]} = 0 : In UART boot, the channel is 0. {GPIOG[25]} = 1 : In UART boot, the channel is 1.	f_{CPU} : 288 MHz f_{IOBUS} : 144 MHz f_{SF} : 96 MHz
0011b	NOR Flash Boot Mode {GPIOG[27], GPIOG[25]} = 00b : 8bits NOR Flash {GPIOG[27], GPIOG[25]} = 01b : 16bits NOR Flash {GPIOG[27], GPIOG[25]} = 1xb : 32bits NOR Flash	f_{CPU} : 288 MHz f_{IOBUS} : 144 MHz
0100b	SPI Slave Boot Mode {GPIOG[27]} = 0 : CMD polarity is active low {GPIOG[27]} = 1 : CMD polarity is active high {GPIOG[25]} = 0 : PCK polarity is rising edge {GPIOG[25]} = 1 : PCK polarity is falling edge	f_{CPU} : 288 MHz f_{IOBUS} : 144 MHz
0101b	SDMMC Boot Mode {GPIOG[27]} = 0 : SDMMC channel 0 {GPIOG[27]} = 1 : SDMMC channel 1 {GPIOG[25]} = 0 : SD Boot mode. {GPIOG[25]} = 1 : eMMC Boot mode.	f_{CPU} : 288 MHz f_{IOBUS} : 144 MHz f_{SDCLK_MAX} : 48 MHz
0110b	UART Boot Mode {GPIOG[25]} = 0 : UART channel 0 {GPIOG[25]} = 1 : UART channel 1	f_{CPU} : 288 MHz f_{IOBUS} : 144 MHz
0111b	EHI 80 Boot Mode	f_{CPU} : 288 MHz f_{IOBUS} : 144 MHz f_{EHI} : 144 MHz
1xxxb	USB Function Boot Mode (USB port0)	f_{CPU} : 288 MHz f_{IOBUS} : 144 MHz

In the NVS2310, there is an internal boot ROM for system initialization process. It contains the fundamental routines for system initialization or boot procedure through various device or interface such as EHI, I2C master, SPI, USB device, parallel NOR flash, serial EEPROM with SPI or I2C protocol, NAND flash, SD/MMC.
The NVS2310 uses 3 clock domains in system boot operation. They are fCPU , fIOBUS , fMBUS (CPU, IOBUS, Memory bus)). The fCPU, fIOBUS and fMBUS are fixed to 288, 144 MHz and 144MHz..

There are 9 modes for booting procedure. The Figure 7.2 illustrates the timing of reset sequence at power-up.

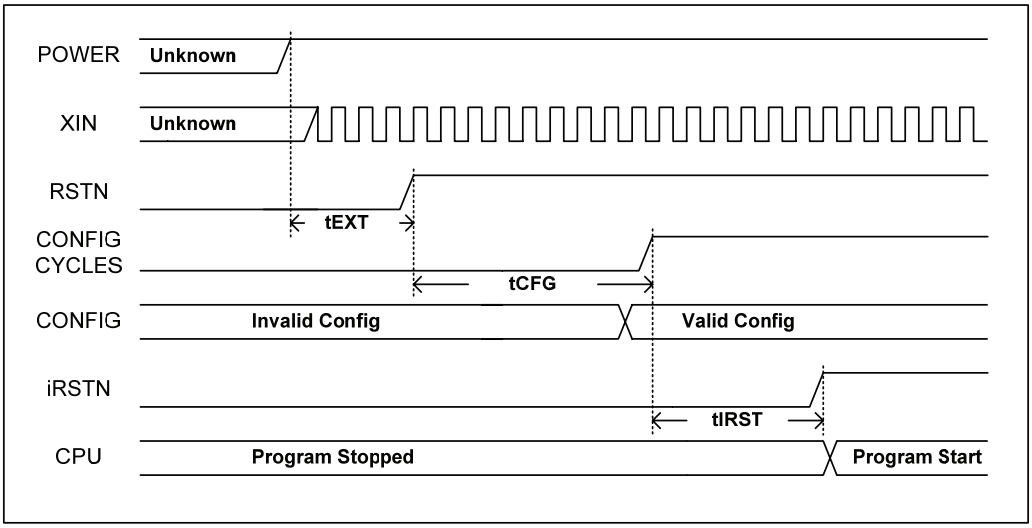


Figure 7.2 Reset Sequence

* The XIN and RSTN are the external pins and other signals are internal signals.
* The BM[3:0] are in input state during the “tEXT” and “tCFG”.
In the above figure, the ‘tEXT’ time is determined by external POR reset circuit or component and ‘tIRST’ is fixed to 64 clock cycles. But the ‘tCFG’ is about 1,048,576 clock cycles.

7.3 Overall Procedure

The following figure shows the overall flowchart of boot code.

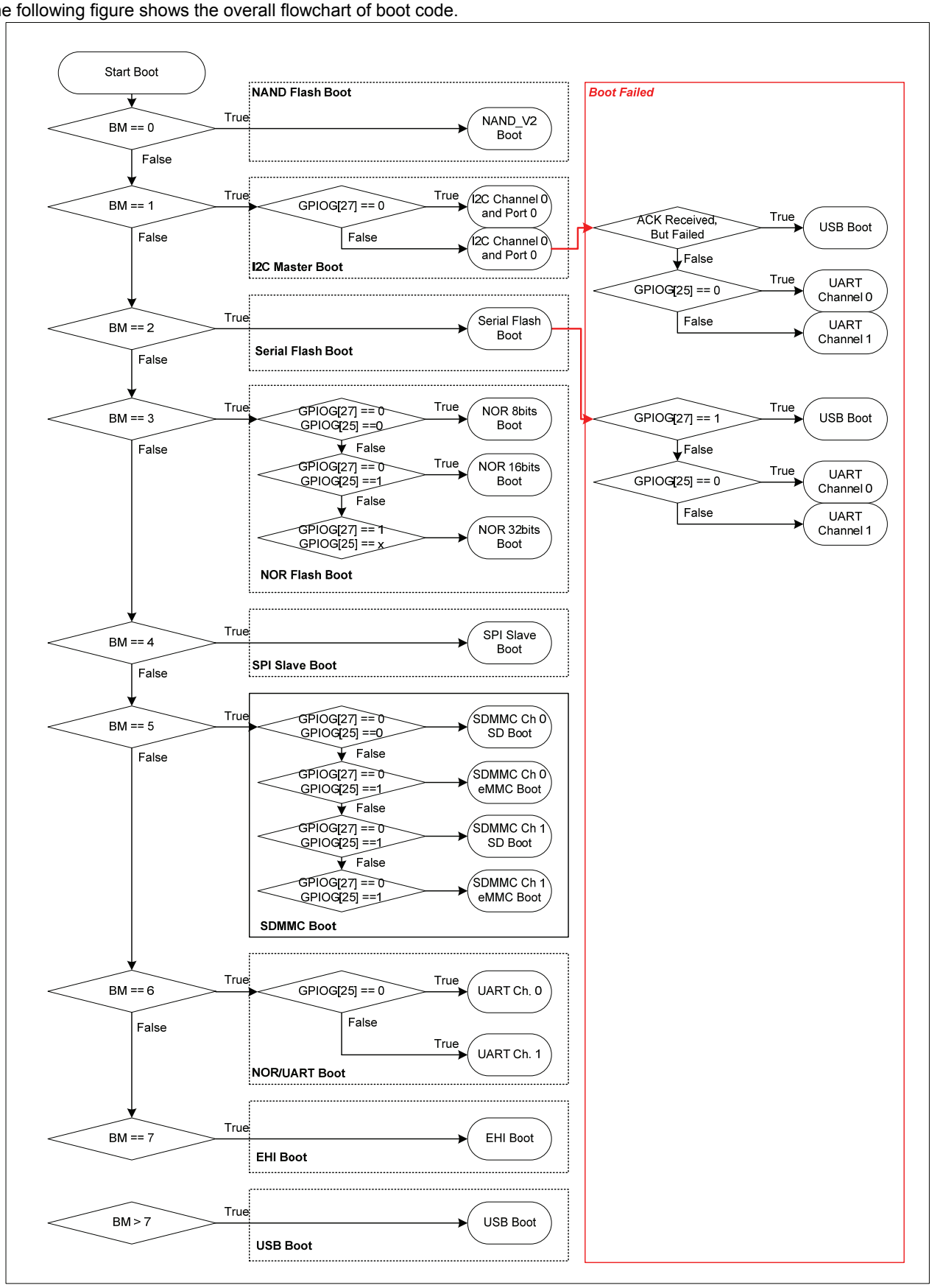


Figure 7.3 Overall Flowchart of Boot Code

7.4 EHI Boot (BM[3:0] = 0111b)

In this mode, the on-chip CPU enables EHI module and sets HPINT to high. In this time, operating clock of EHI block is 144MHz. At the same time, the external host must wait until HPINT is high. After HPINT is asserted, the external host sets the ST field of EHST to INITCFG which indicates the interface bus width, the validity for this packet, and the total size of the BIP(Boot Information Packet).
After then, the on-chip CPU re-initializes the EHI with the bus width information in INITCFG.

7	6	5	4	3	2	1	0
1	0	0	0	0	BIPEN	VAL	BW

NOTE) BIIPEN = 0 (recommend)
VAL = should be 1 for valid INITCFG
BW = 0 (8bits), 1 (16bits)

Figure 7.4 INITCFG Bit Field

For handshaking between this LSI and the external host, the ST field of EHST register is used. The following is EHI booting procedure and refer to Figure 7.7.

After boot codes reinitialize this LSI using INITCFG, the EHST.ST is set to 0x55 by the on-chip CPU (On-chip CPU).
The external host uploads the data to this LSI. When uploading is completed, the external host set EHD register to start address of uploaded data and set EHST.ST to 0xAA. (External host).
When upload is completed, EHST.ST is not equal to 0x55. The on-chip CPU determines that type of the uploaded data is the BIP or the firmware program using the BIPEN field of INITCFG.(On-chip CPU)
If INITCFG. BIPEN is zero, the on-chip CPU jumps to start address in EHD register and EHI boot is completed. Otherwise, the uploaded data is the BIP and start address is BIP address. The BIP has much information about booting from EHI(Figure 7.5). After receiving the BIP from the external host and processing it, the on-chip CPU set the EHST.ST to 0x55. After then, the external host can upload program.(On-chip CPU)
The external host fills buffer0 that it is indicated by BIP. If it is full, the external host sets EHST.ST to 0x2C. After then, the external host fills buffer1 when it is full. And the external host sets EHST.ST to 0x25(Figure 7.6). This procedure is repeated by the external host until all of data is uploaded.(External host)

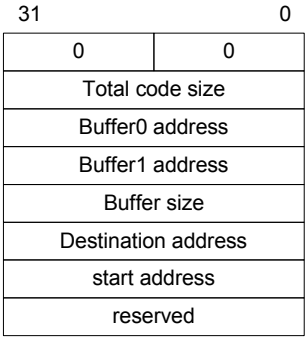


Figure 7.5 BIP Data Structure

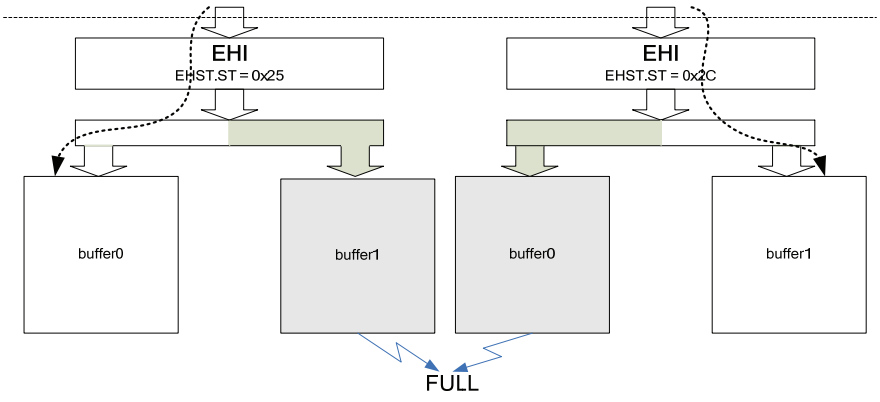


Figure 7.6 External Host Uploads Program when BIPEN = 1

The EHI can manipulate either of 80 interfacing. The bus-width of EHI is determined by the BW field of INITCFG. During the procedure for booting from EHI, the external host can access all the address space in the NVS2310. Therefore, if you want to read from or write to any register in NVS2310, just do it as you want

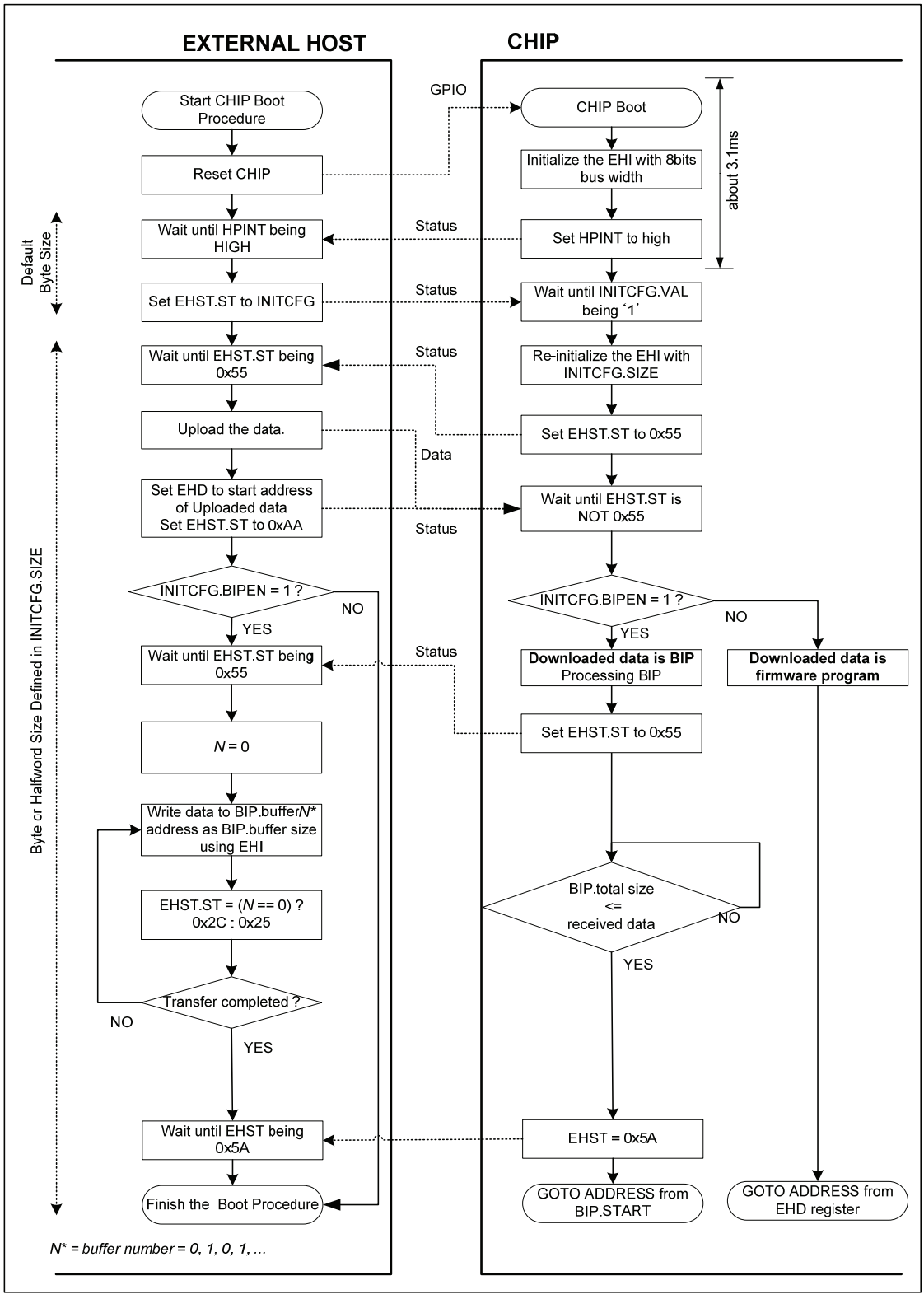


Figure 7.7 EHI Boot Procedure

7.5 USB Boot (BM == 1xxxb)

This mode is mainly for firmware upgrade mode. In this mode, user can download a program into the user defined area When the failure occurs in some other boot modes, it may progress this boot sequence also.
The procedure of this mode is as follows.

The NVS2310 makes internal SRAM area starts from zero, and copies USB interrupt service routine to internal SRAM area.
It waits until USB connection is established.
Once it is connected, host transfers first the parameter for USB loader routine including start address, destination address and the amount of data to be transferred (with a unit of packet).
The NVS2310 starts communicating between a host PC with fixed amount of data which is called as packet. The packet size of NVS2310 is 512 bytes.
At every successful reception of packet, it copies them where the destination address pointed, and after all amount of data has been copied, it starts program where the start address pointed.

Normally, the program downloaded is for writing user system firmware to non-volatile memory like NAND flash.
The following figure illustrates the sequence of USB boot mode described above.

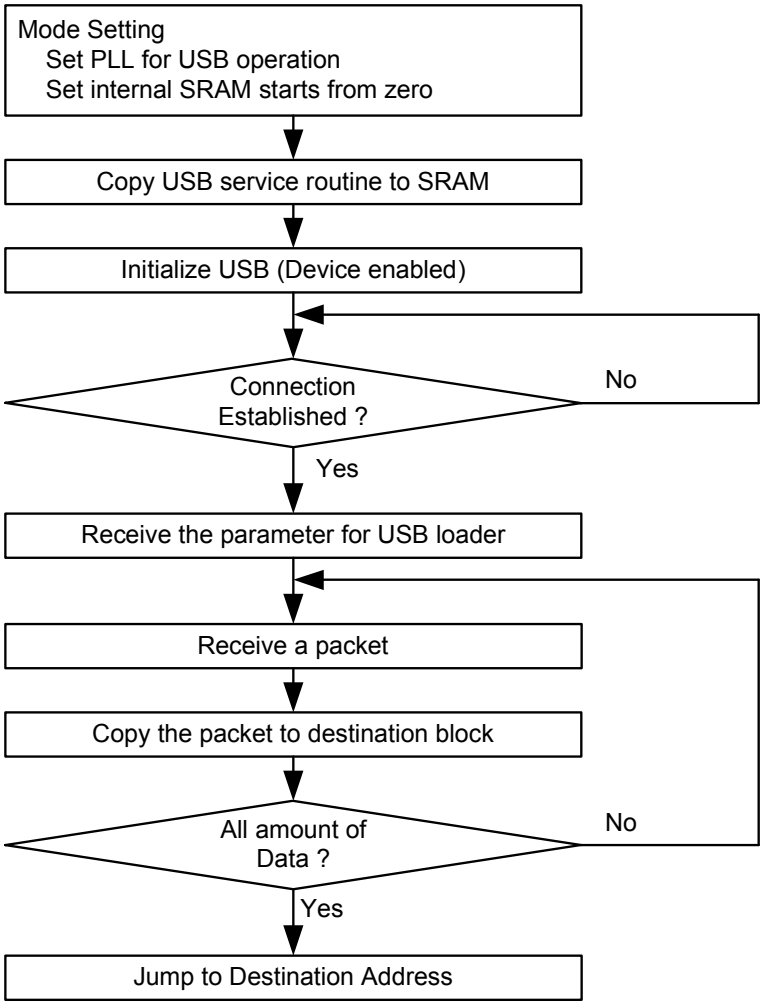


Figure 7.8 USB Boot Procedure

7.6 External NOR Boot (BM == 0011b)

In this mode, the contents of NOR flash can be examined by checking CRC. If it is OK, it boots like a normal mode. If it is not OK, the NVS2310 automatically changes to USB boot mode so user can fix NOR contents via USB interface. NOR flash must be attached to CSN_NOR(GPIOB[26]) pin. Supportable band width of NOR Flash is 8bits, 16bits or 32bits. GPIOG[27] and GPIOG[25] are used to decide a band width of attached NOR Flash. If GPIOG[27] is pulled up, 32bits NOR Flash woulde be attated as boot devic. In another case, if GPIOG[25] is pulled up, 16bits NOR Flash would be attached as boot device. Otherwise 8bits NOR Flash would be attached.

The detailed procedure of NOR Flash boot is as follows.

If 1st or 2nd word is 0xFFFFFFFF, NVS2310 goes to USB boot mode for F/W downloading.
If the 31'th bit of 5'th word in NOR flash is zero, the CRC checking procedure will be skipped.
The NVS2310 do the C2 (Dual CRC Checking) process on the image of external NOR flash.
The C2 process is described at the Dual CRC Checking section.
After CRC process has finished and if it returns "OK", NVS2310 finishes booting procedure by jumping to the destination address.

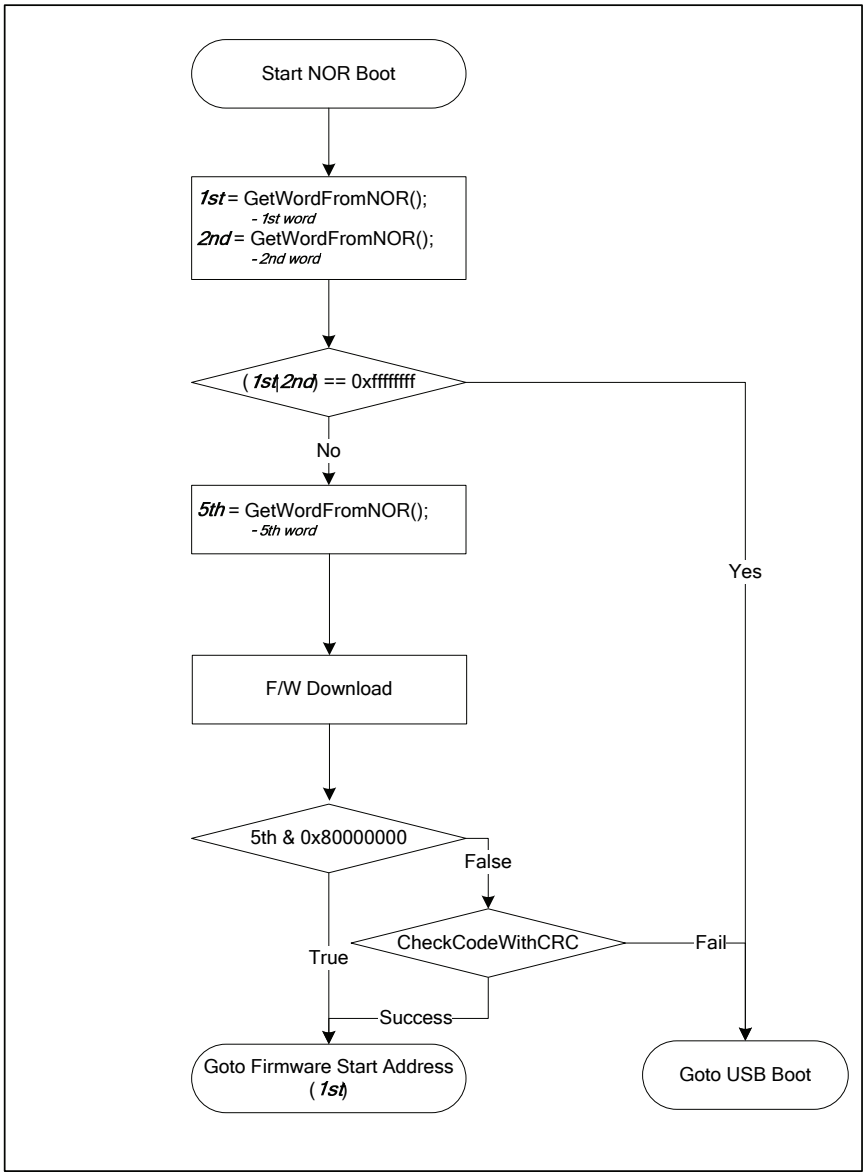


Figure 7.9 External NOR Boot Procedure

7.7 NFC NAND V2 Boot (BM == 0000b)

In this mode, the F/W code is read from NAND flash attached. To make use of this mode, the predefined structure such as Master Block and Master Cluster should be fused in the NAND flash.
In NVS2310, 24 bit ECC based on BCH algorithm is used for error correction of read data. It can guantee high robustness for all kinds of NAND flash(mass production).

The supportable configuration of NAND flash is as follows.

Case1) single NAND flash of 8-bit bus-width. (8bit I/O BUS cofiguration)
Case2) single NAND flash of 16-bit bus-width. (16bit I/O BUS configuration)
Case3) double NAND flashes of 8-bit bus-width. (16bit I/O BUS configuration) with the same chip enable signals
Case4) double NAND flashes of 16-bit bus-width. (32bit I/O BUS configuration) with the same chip enable signals
Case5) quad NAND flashes of 8-bit bus-width. (32bit I/O BUS configuration) with the same chip enable signals

NAND flash I/O connection of NVS2310 is the following

1. 8 bits I/O : The configuration of CASE1 can be possible
   - nWE : GPIOB[17]
   - nOE : GPIOB[19]
   - nCS : GPIOB[30]
   - ALE, CLE : GPIOB[21], GPIOB[20]
   - Data I/O[7:0] : GPIOB[7:0]
2. 16 bits I/O : The configuration of Case2 or Case3 can be possible
   - nWE : GPOB[17]
   - nOE : GPIOB[19]
   - nCS : GPIOB[30]
   - ALE, CLE : GPIOB[21], GPIOB[20]
   - Data I/O[15:0] : GPIOB[15:0]
3. 32 bits I/O : The configuration of Case4 or Case5 can be possible
   - nWE : GPIOB[17]
   - nOE : GPIOB[19]
   - nCS : GPIOB[30]
   - ALE, CLE : GPIOB[21], GPIOB[20]
   - Data I/O[31:0] : {GPIOF[29:14], GPIOB[15:0]}

To read the F/W code from NAND flash, first of all, should know the type of attached NAND flash such as I/O band width, connection type(single/double/quad) and so on.
In NVS2310, NAND flash boot search the type by scanning from the type of Case1 to the type of Case5 during golden block reading. There are optional conditions to support fast searching and boot as a configuration of GPIOG[27] and GPIOG[25].

The optionals conditions are the follows

Table 7.2 The configuration of scanning mode

GPIOG[27]	GPIOG[25]	Scnning mode	The range of scnning
0	0	Normal scan mode	Case1 ~ Case5
0	1	NAND 8bits I/O scan mode	Case1
1	0	NAND 16bits I/O scan mode	Case2 ~ Case3
1	1	NAND 32bits I/O scan mode	Case4 ~ Case5

The boot sequence of this mode is as the follows. If there exists any problem hard to recover during this sequence, it goes to USB boot mode automatically. It assumes the contents is stored as little endian format.

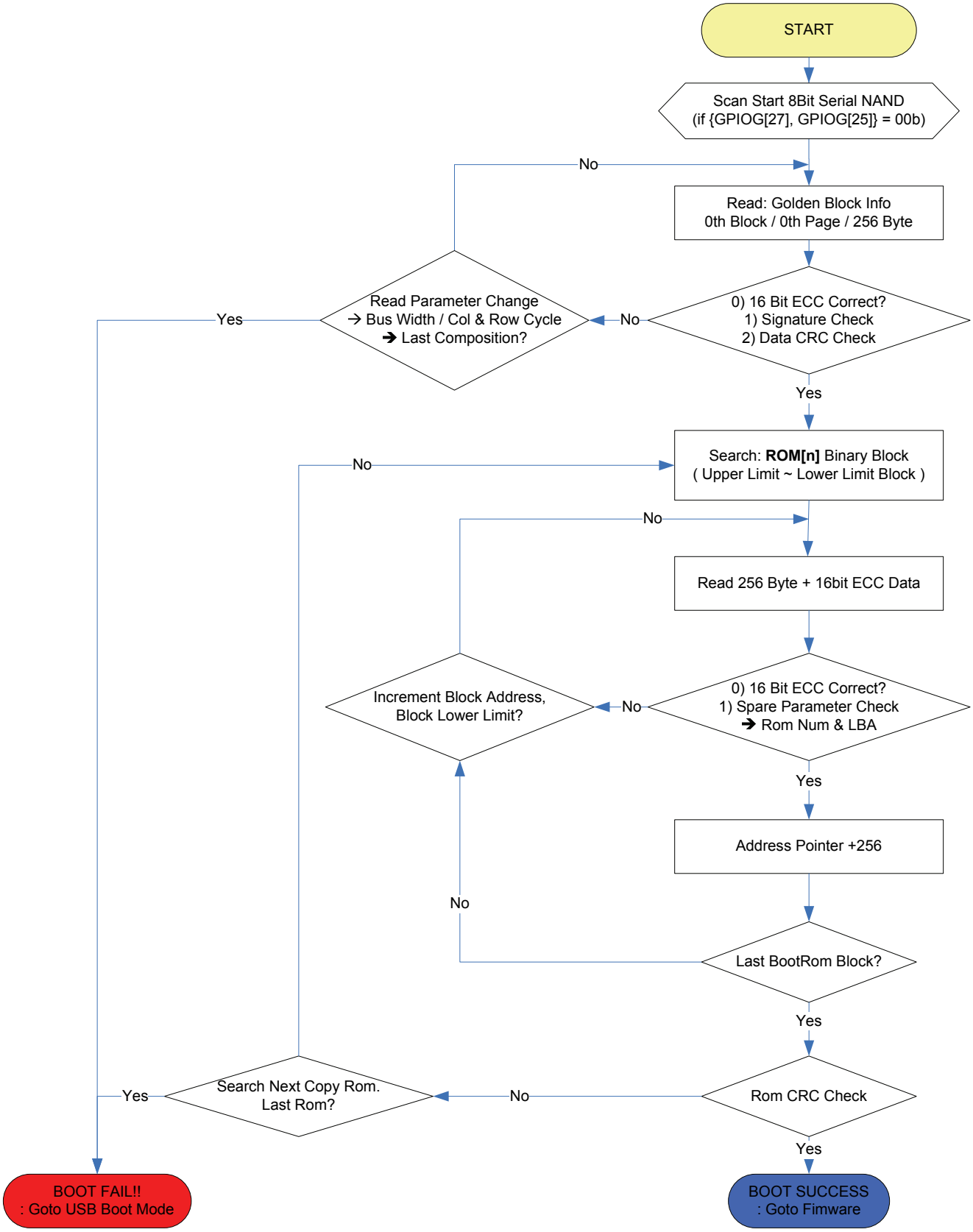


Figure 7.10 NAND_V2 Boot Procedure

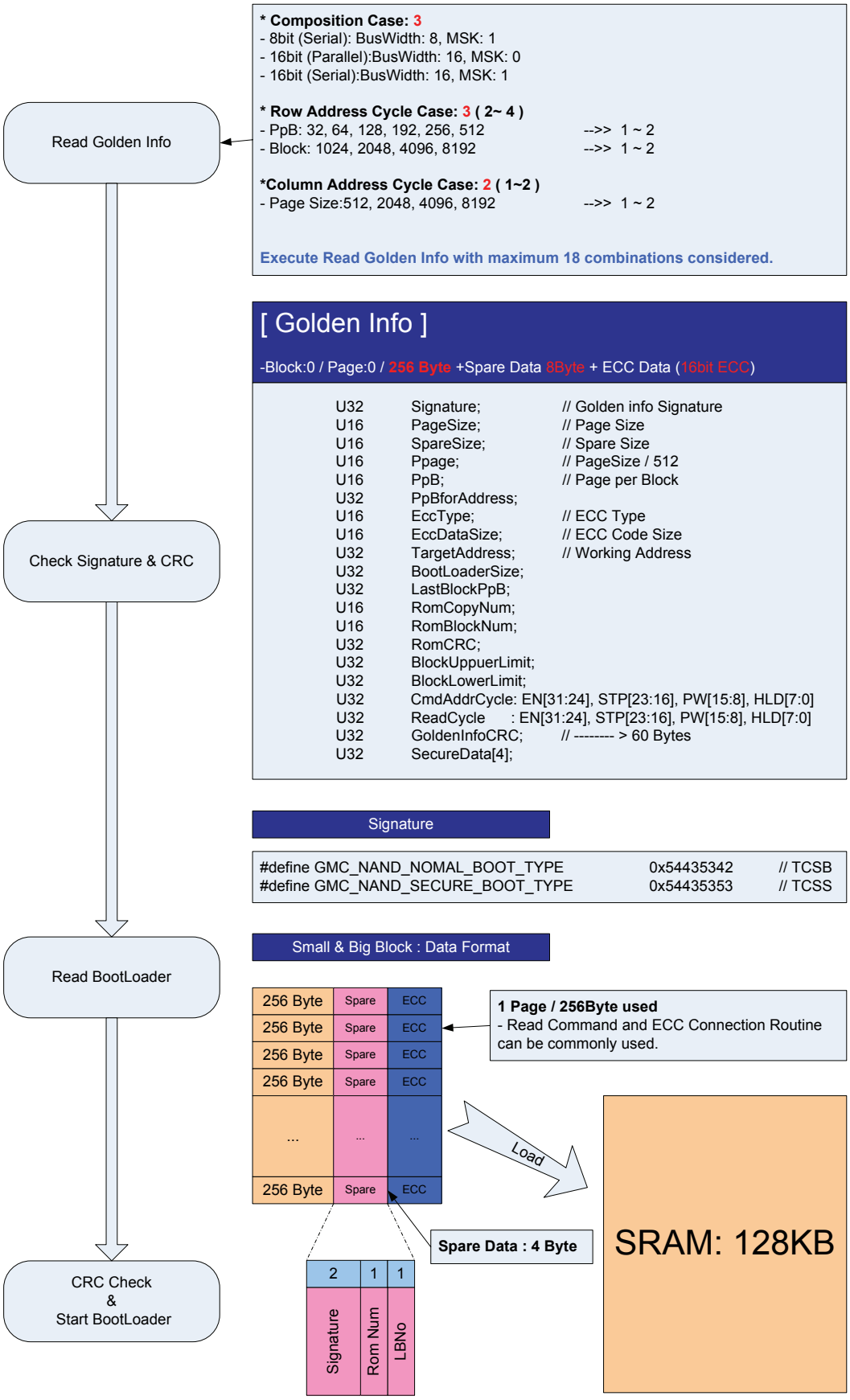


Figure 7.11 NAND_V2 Golden Info. Structure

7.8 I2C Master Boot – EEPROM Boot (BM == 0001b)

In this mode, the F/W code would be read from serial EEPROM attached to the I2C ports. It interfaces with standard I2C protocol. If serial EEPROM is not attached, the NVS2310 changes to UART boot mode.
The procedure checks if there exist EEPROM first. If there exist an EEPROM, the NVS2310 do the following procedure. If certain problem which can not be solved has occurred, it goes to USB boot mode automatically.
The GPIOG[25] port determines the boot channel for I2C. If pulled-up, the channel 1 port would be used and otherwise the channel 0 port used.

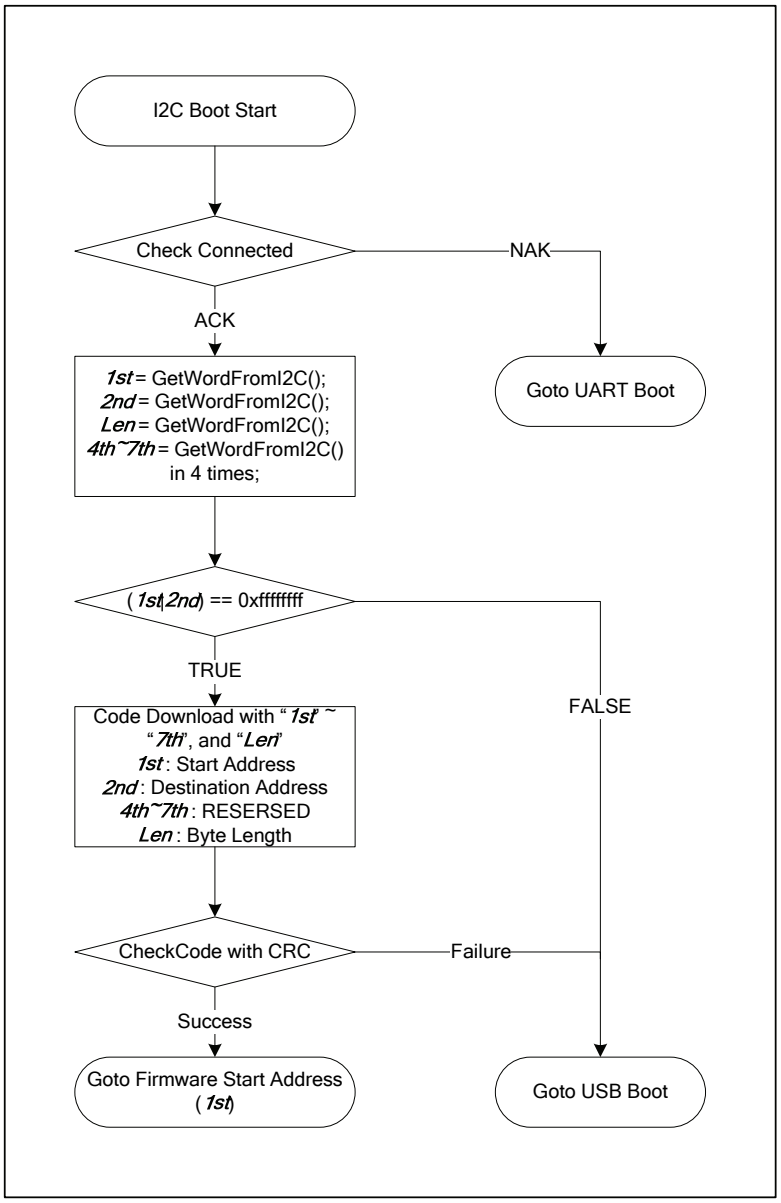


Figure 7.12 I2C Boot Procedure

7.9 Serial Flash Boot (BM==0010b)

The serial flash boot mode supports the serial NOR flash with SPI protocol. The GPIOB[11:8] are used for interfacing the serial EEPROM. The GPIOG[27] pin determines which mode to branch after serial EEPROM booting failed. If GPIOG[27] is pulled-up, the boot mode would be changed to USB boot after failed and otherwise to the UART boot mode. In the case of UART boot mode, the GPIOG[25] determines that which port can be used in UART boot mode. If GPIOG[25] is pulled-up, the UART channel 1 will be used and otherwise UART channel 0 used.

```
typedef       struct {
  unsigned short       MID;                    // Unused for booting
  unsigned short       DID;                    // Unused for booting
  unsigned             SectorSize;             // Unused for booting
  unsigned             BlockSize;              // Unused for booting
  unsigned             BlockNum;               // Unused for booting
  unsigned             uStart;                 // Used for booting
  unsigned             uDest;                  // Used for booting
  unsigned             uLength;                // Used for booting
  unsigned             uDLDV;                  // Used for booting
  unsigned char        Name[20];               // Unused for booting
  unsigned char        Rsv[256-56];            // Unused for booting
              // Rsv[100]              // Used for booting
                  //      :               // Used for booting
                  //  Rsv[115] // MCFG0~3   // Used for booting
  unsigned             uCRC;                   // Used for booting
} sSFBootHeader;
```

Figure 7.13 Data Structure for Header Information Stored In Serial Flash

In the above figure, the data structure required for booting from serial flash is described. The variables named as uStart, uDest, uLength, and Rsv[100] to Rsv[115] are used in the boot code. If the uDest is not equal to uStart, the uDLDV is used to DLDV, which determines the clock frequency of GPSB(General Purpose Serial Bus) controller.
The uLength field is total number of bytes to stored in the serial flash and should be aligned to word(4bytes).
The overall procedure for serial flash boot is shown in the following figure.

```
Void IO_GPSBM_GetWordData (unsigned *data, unsigned nwords);
// Function to read from serial flash
// “data” is destination pointer
// “nwords” is the total number of words to read

Void StartSFlashBoot ()
{
   Unsigned char mem_header[16];

    IO_GPSB_PortConfig ();      // Port Configuration for Serial Flash Interface
    IO_GPSBM_Init ();  // Initialization for GPSB controller
    IO_GPSB_FlushBuffer ();     // Flushing Buffer for Initial Loading
    IO_GPSBM_GetWordData ((unsigned *)pHeader,0,sizeof(sSFBootHeader)/4);

    if ( IsEmptyFlash () ) return (-1); // Booting Failed

    uData = IO_UTIL_CalcCRC32(0,(unsigned *)pHeader,256-4,0);
    if ( uData != pHeader->uCRC ) return (-1);// Booting Failed

for (i=0;i<16;i++)
  mem_header[i] = pHeader->Rsv[i+100];

    IO_UTIL_SetMEM (pHeader->uDest,mem_header);
    // Initialization for Memory Controller

    if ( (pHeader->uDest&0xf0000000) != (pHeader->uStart&0xf0000000) ) {
      HwGPSB0->uMOD.bMOD.DLDV = pHeader->uDLDV;
      uExecStart = pHeader->uDest;
    }

    IO_GPSB_FlushBuffer ();
    IO_GPSBM_GetWordData
    ((unsigned *)pHeader->uDest, SFLASH_PAGE_DATA_SIZE, (pHeader->uLength>>2));
    // SFLASH_PAGE_DATA_SIZE : 256

    if (IO_UTIL_CheckCODE((unsigned *)pHeader->uDest, pHeader->uLength, 0))
      return 0;        /* to usb boot */

    goto_firmware (pHeader->uStart);
}
```

Figure 7.14 Pseudo Code for Serial Flash Booting Procedure

7.10 SPI Slave Boot (BM==0100b)

The SPI slave boot uses the ports named GPIOB[11:8]. And if boot failed, the boot mode would jump to the USB boot – USB firmware download mode.
Accroding to input configuration of GPIOG[27] and GPIOG[25], the polarities of CMD(GPIOB[8] and PCK(GPIOB[9]) are determined, If GPIOG[27] is pulled down to low, the CMD operates as active low signal and if pulled up to high, it operates as active high signal. Similiarly, if GPIOG[25] is pulled down to low, the rising edge of PCK is used and if GPIOG[25] is pulled up to high, the falling edge of it is used to check its data.

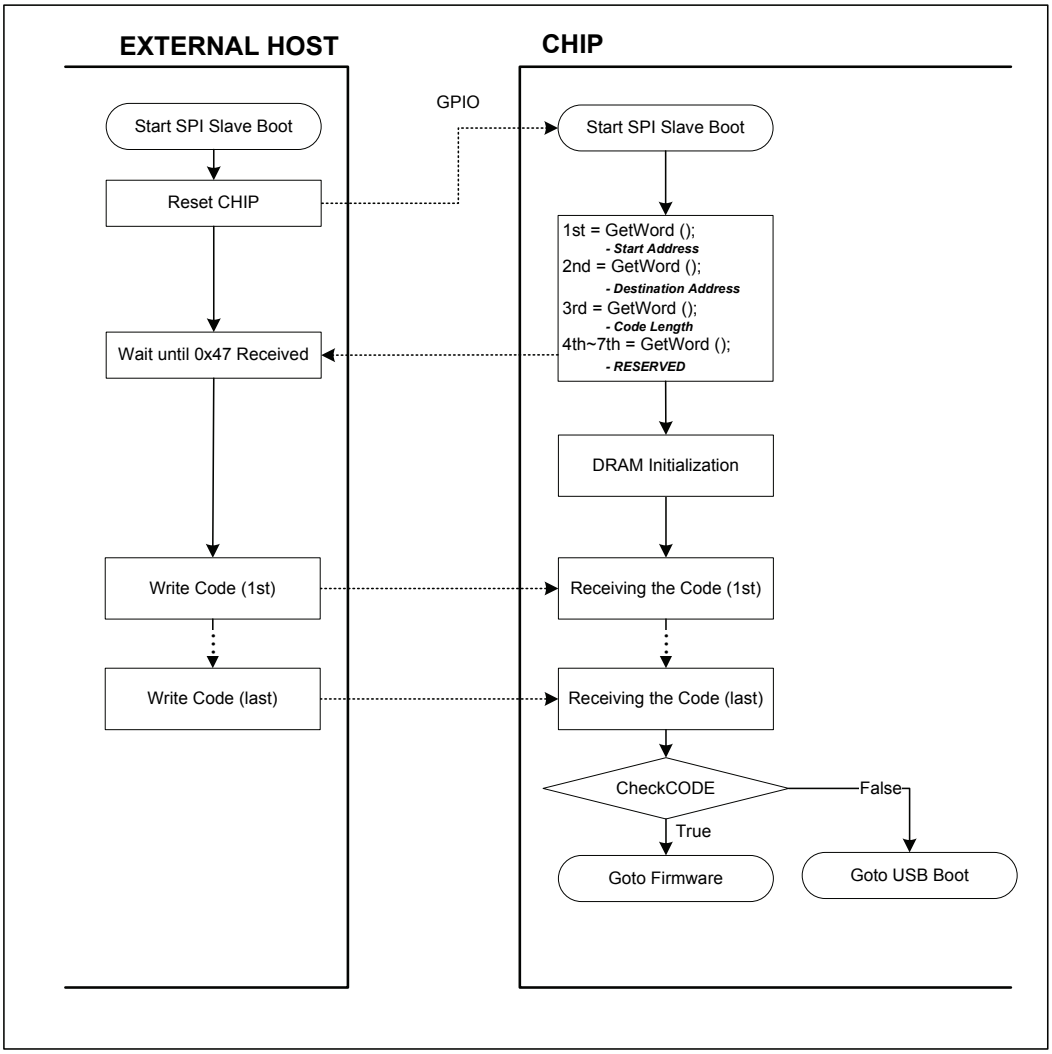


Figure 7.15 SPI Slave Boot Procedure

7.11 UART Boot (BM==0110b)

The UART boot mode can be changed from I2C boot mode or serial flash boot mode. If the I2C eeprom boot mode was failed, the UART boot would be started with the configuration of the GPIOG[25] pin. If GPIOG[25] is pulled-up, the UTXD0(GPIOG[25]) will be used for booting and if GPIOG[25] is pulled-down, the UTXD1(GPIOG[29]) and URXD1(GPIOG[28]) will be used.
The detailed procedure is as follows. It uses little endian for word manipulation. So, the LSB byte is received first, and MSB byte is received last.

Enable UART I/F. (fUART=24MHz, 115.2kbps, Data 8bit, Non-parity, 1 Stop bit)
Send ACK (0x52, ASCII code of 'R').
Wait until ACK is received. ('H' or 'R')
Send ACK
Receive the Start Address.
Send ACK
Receive the Destination Address.
Send ACK
Receive the size of code in byte unit.
Send ACK
Receive the 4 words RESERVED data.
(This value is not used in rev.ax chip. But always received 4 word value.)
Send ACK and go to routine (11) if memory parameter is not RESERVED3.
If ACK in (3) is 'H', new divisor value (DLM, DLL) is received (fUART = 24MHz) and baud rate is reconfigured.
Load the code from HOST to destination address
Do the CRC checking process on the downloaded image.
After CRC checking process has finished, and it is OK, NVS2310 finishes booting procedure by jumping to the destination address. Or, it restarts UART boot from the first step.

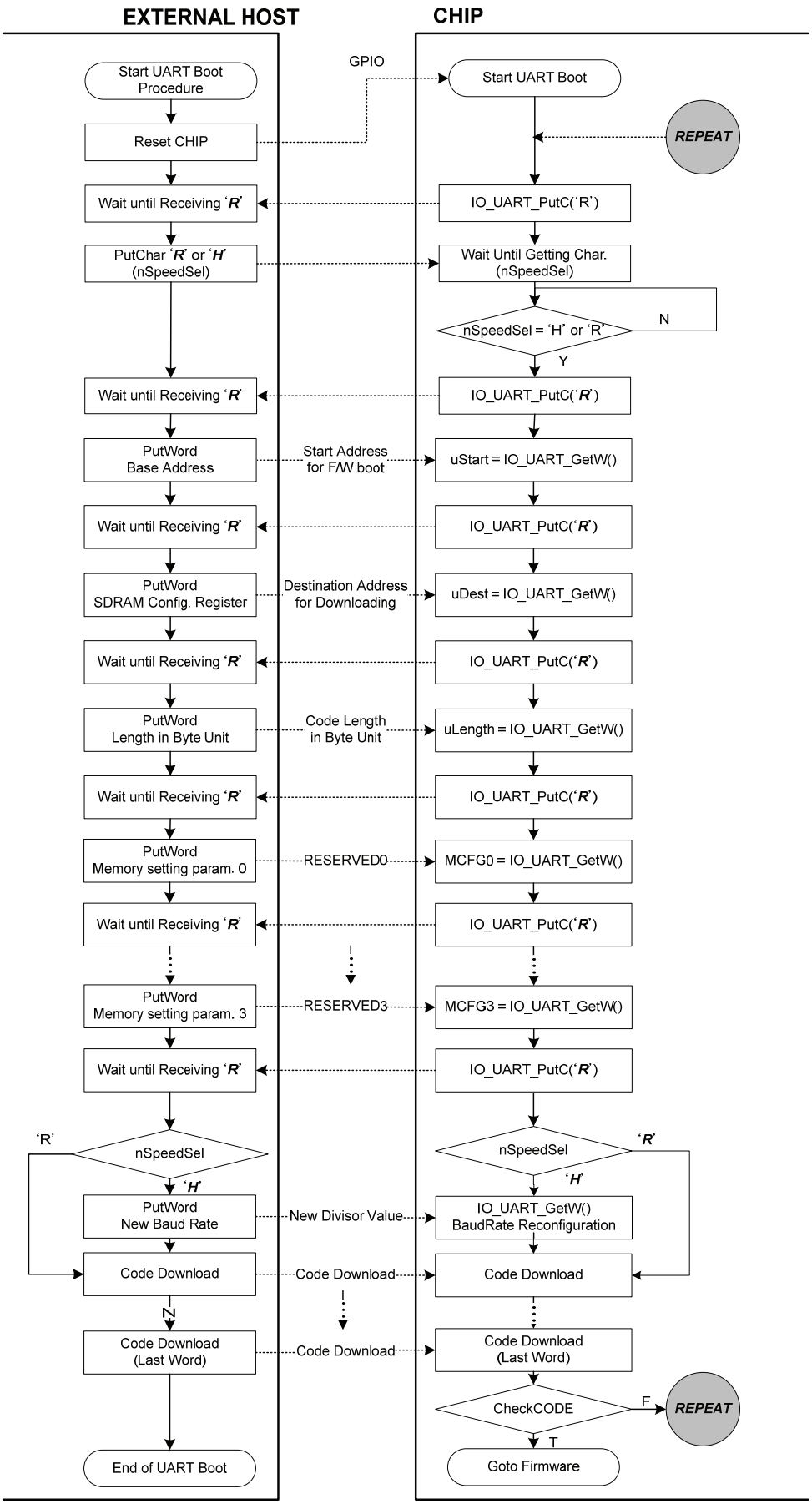

EXTERNAL HOST
CHIP
GPIO
Start UART Boot Procedure
Reset CHIP
Wait until Receiving 'R'
PutChar 'R' or 'H' (nSpeedSel)
Wait until Receiving 'R'
PutWord Base Address
Start Address for F/W boot
Wait until Receiving 'R'
PutWord SDRAM Config. Register
Destination Address for Downloading
Wait until Receiving 'R'
PutWord Length in Byte Unit
Code Length in Byte Unit
Wait until Receiving 'R'
PutWord Memory setting param. 0
RESERVED0
Wait until Receiving 'R'
PutWord Memory setting param. 3
RESERVED3
Wait until Receiving 'R'
'R'
nSpeedSel
'H'
PutWord New Baud Rate
New Divisor Value
Code Download
Code Download
N
Code Download (Last Word)
Code Download
End of UART Boot
Start UART Boot
REPEAT
IO_UART_PutC('R')
Wait Until Getting Char. (nSpeedSel)
nSpeedSel = 'H' or 'R'
N
Y
IO_UART_PutC('R')
uStart = IO_UART_GetW()
IO_UART_PutC('R')
uDest = IO_UART_GetW()
IO_UART_PutC('R')
uLength = IO_UART_GetW()
IO_UART_PutC('R')
MCFG0 = IO_UART_GetW()
IO_UART_PutC('R')
MCFG3 = IO_UART_GetW()
IO_UART_PutC('R')
nSpeedSel
'R'
'H'
IO_UART_GetW() BaudRate Reconfiguration
Code Download
Code Download (Last Word)
CheckCODE
F
REPEAT
T
Goto Firmware

Figure 7.16 UART Boot Procedure

7.12 SD/MMC and eMMC Boot (BM==101b)

The NVS2310 supports SD/MMC boot mode using SD, MMC, MMC+, SDHC and SDXC cards. eMMC boot mode is also available for eMMC 4.3 or higher compatible cards. The GPIOG[25] indicates which mode of SD/MMC and eMMC is selected, if it is pulled down to low, the chip will be under SD boot mode and if pulled up to high, it will run under eMMC boot mode. The GPIOG[27] is related to channel section. If it is pulled down to low, Host Controller Channel0 will be activated and GPIOG[9:0] will be dedicated for Channel0. In the case of pull up, Channel1 will be activated and GPIOG[15:10] will be dedicated for that.
Both of SD/MMC and eMMC boot mode have the same booting procedure. But how to read rom code and parameters and where those are located is different.
The detailed procedure is as follows. It uses little endian for word manipulation. So, the LSB byte is received first, and MSB byte is received last.

Enable SD/MMC I/F and initialization. (fSDMMC=48MHz, SDCARD_CLK limit= 48MHz)
Card Reset.
Scan Card for checking SD/MMC/MMC+/SDHC (SD_CLK = 170 KHz)
Get response for identification.(OCR,CID,RCA,CSD)
Set SD_CLK for transfer mode (using SDCARD information CLK, it's limitation CLK is 48MHz)
Read the first single block (512byte) for setting boot parameter
(start address, destination address, code length)
Read user-defined code if it is needed
If user-defined parameter (SD port delay parameter) is enable, SD port delay parameters are change to user - defined parameter.
Calculating ROM code address and size refer to boot parameter.
Read firmware data from card.
Do the CRC checking process on the downloaded image.
After CRC checking process has finished, and it is OK, NVS2310 finishes booting procedure by jumping to the start address. Or, it restarts USB boot from the first step.

The SD/MMC boot parameters are shown in following figure. These parameters are stored in the last block of hidden area in SD or MMC cards. Destination is the start address of memory to wirte ROM code. User Parameter is reserved for Special Purpose. ROM Code Size means the size of ROM code. Config Code Size is needed to indicate there exists Configuration Code Area. If it is not available, it may be '0'. Maximum ROM Size stands for totoal ROM size including boot parameter sector and ROM code. Hidden Default Page Size is reseved for Application Software. CRC Value is the result of CRC calculation of boot paramter sector.
If Config Code Size is non-zero, user-defined configuration code will precede boot parameter sector. It can be accessable by the block(512 bytes).

OFFSET	
0x00	Destination
0x04	User Parameter
0x08	ROM Code Size
0x0C	Config Code Size
0x24	Reserved for Special Purpose
0x44	Reserved for Special Purpose
0x80	Maximum ROM Size
0x84	Hidden Default Page Size
0x1F8	
0x1FC	CRC Value

Figure 7.17 SD/MMC Boot Parameter

The eMMC boot parameter is shown at the Figure 7.18 . The structure of eMMC boot parameter is similar to that of SD/MMC boot mode. The difference between SD/MMC and eMMC boot mode is where paramters are loacated. The access sequence of the latter is the reverse of the former.

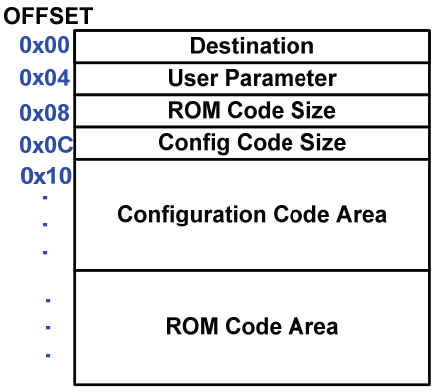


Figure 7.18 eMMC Boot Parameter

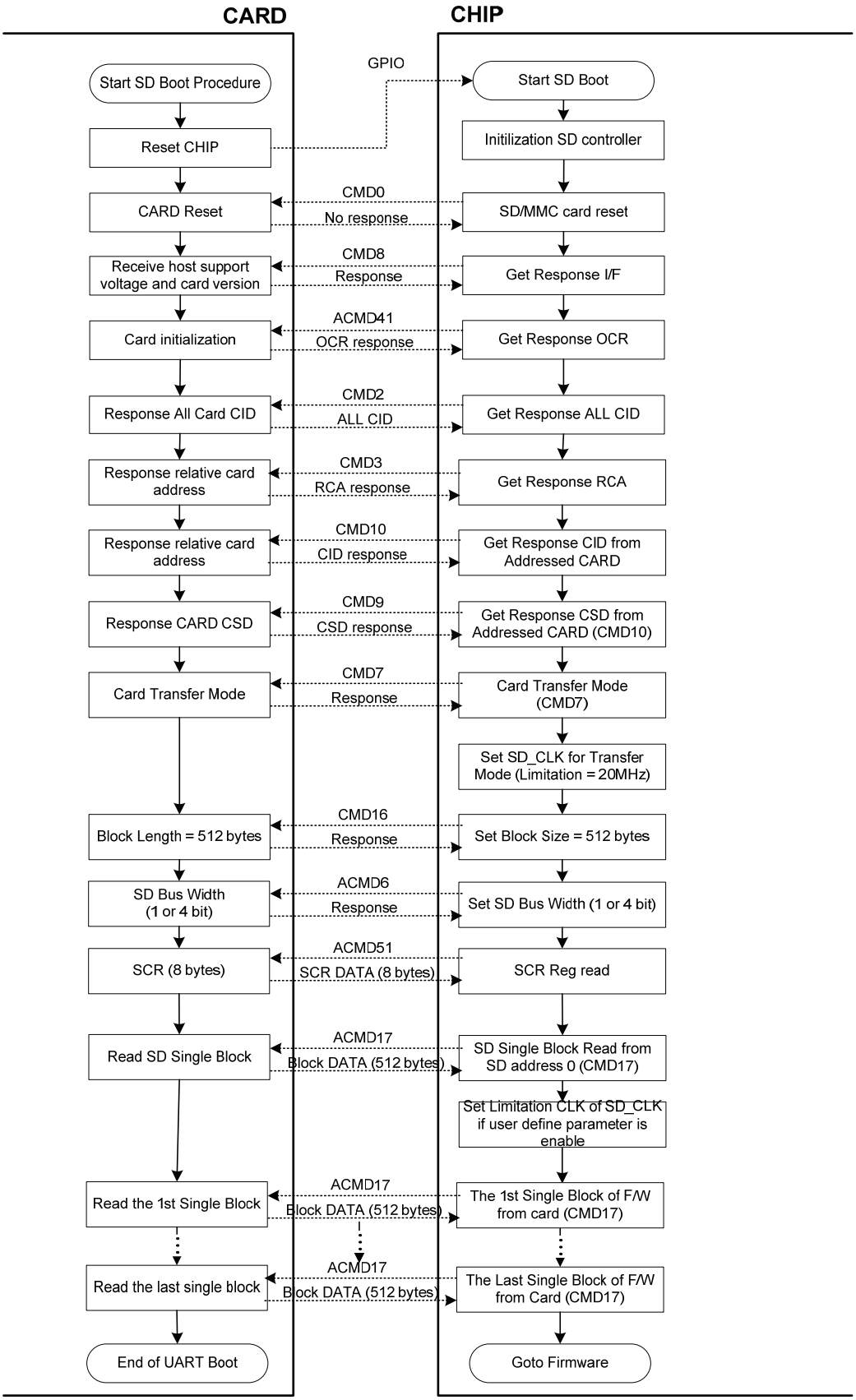


Figure 7.19 SD/MMC Boot Procedure

7.13 Dual CRC Checking

The NVS2310 can check the CRC for the downloaded image according to the dedicated algorithm. This is described as follows.
Calculate CRC on the downloaded image up to 64Kbytes. In calculation, the word at the offset of 0x10 and 0x14 is skipped, because the good CRC code is contained at the offset of 0x10, and the word at the offset of 0x14 means the address of user-defined CRC routine.
After the first CRC calculation has been done, and it is same as the value at the offset address of 0x10, NVS2310 check if the address of user-defined CRC routine is not 0. Or, NVS2310 goes to pre-defined boot mode (USB or UART boot) automatically.
If there exist user-defined CRC routine, call that routine.
The user-defined CRC routine checks its own CRC code and return with Equal condition if it is OK or return Not Equal condition if it fails.
If there are failures on the user-defined CRC checking, NVS2310 goes to pre-defined boot mode (USB or UART boot) automatically.
The following code shows same CRC generation algorithm used in the first CRC routine.

```
word calc_crc (word *base, word length, word *crctable)
{   // contents of crctable is acquired by gen_crc () function
    word       crcout  = 0;
    word       cnt, i, code, tmp;

    length     >>= 2;           // convert into word unit.
    for (cnt = 0; cnt < length; cnt ++) {
      if (cnt == 0x04 or cnt == 0x05)  // skip offset of 0x10 and 0x14
      continue;
      code     = base[cnt];
      for (i = 0; i < 4; i ++) {
      tmp      = code ^ crcout;
      crcout   = (crcout >> 8) ^ crctable[tmp & 0xFF];
      code     = code >> 8;
      }
    }
    return     crcout;
}

void gen_crc (word *crctable)
{   // Polynomial = x^32 + x^31 + x^16 + x^15 + x^4 + x^3 + x + 1
    word       crc, cnt, i;

    for (cnt = 0; cnt < 0x100; cnt ++) {
      crc      = cnt;
      for (i = 0; i < 8; i ++) {
      if (crc & 1)     crc      = (crc >> 1) ^ 0xD8018001;
      else             crc      = (crc >> 1);
      }
      crctable[cnt]    = crc;
    }
}
```

PART3 – GRAPHIC BUS
NVS2310

Rev. 1.01

Apr 22, 2011

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial Release
2011-04-22	1.01	∗ Correct the register table and document format.

TABLE OF CONTENTS

Contents

Figures

Tables

1 Introduction

NVS2310 GRPBUS provides connections between the internal graphics bus masters and the other buses. Also on chip peripherals are located in it. The graphics bus masters supported is as follows : 2D/3D graphics rendering, various operations for raster graphics.

[Features]

- 2D/3D GPU (Graphics Processing Unit)
  - Pixel Processor
    - Programmable fragment shader
    - Access to framebuffer from fragment shaders
    - Alpha blending
    - Arbitrary memory reads and writes
    - Complete non-power-of-2 texture support
    - Cube mapping
    - Dynamic recursion
    - Fast dynamic branching
    - Fast trigonometric functions, including arctangent
    - Full floating-point arithmetic
    - Framebuffer blend with destination Alpha
    - High Dynamic Range (HDR) textures and framebuffers
    - Indexable texture samplers
    - Line, quad, triangle and point sprites
    - Multiple render targets
    - No limit on program length
    - Perspective Anisotropic Filtering (AF)
    - Perspective correct texturing
    - Point sampling, bilinear, and trilinear filtering
    - Programmable mipmap level-of-detail biasing and replacement
    - Register indirect jumps
    - Stencil buffering, 8-bit
    - Two-sided stencil
    - Unlimited dependent texture reads
    - Virtualized texture samplers
    - 4-level hierarchical Z and stencil operations
    - 4 times and 16 times Full Scene Anti-Aliasing (FSAA)
    - 4-bit per texel texture compression
  - Geometry Processor
    - Programmable vertex shader
    - Autonomous operation tile list generation
    - Flexible input and output formats
    - Indexed and non-indexed geometry input
    - Primitive constructions with points, lines, triangles and quads.

2 Bus Architecture

Figure 2.1 shows the GRP bus overall architecture.

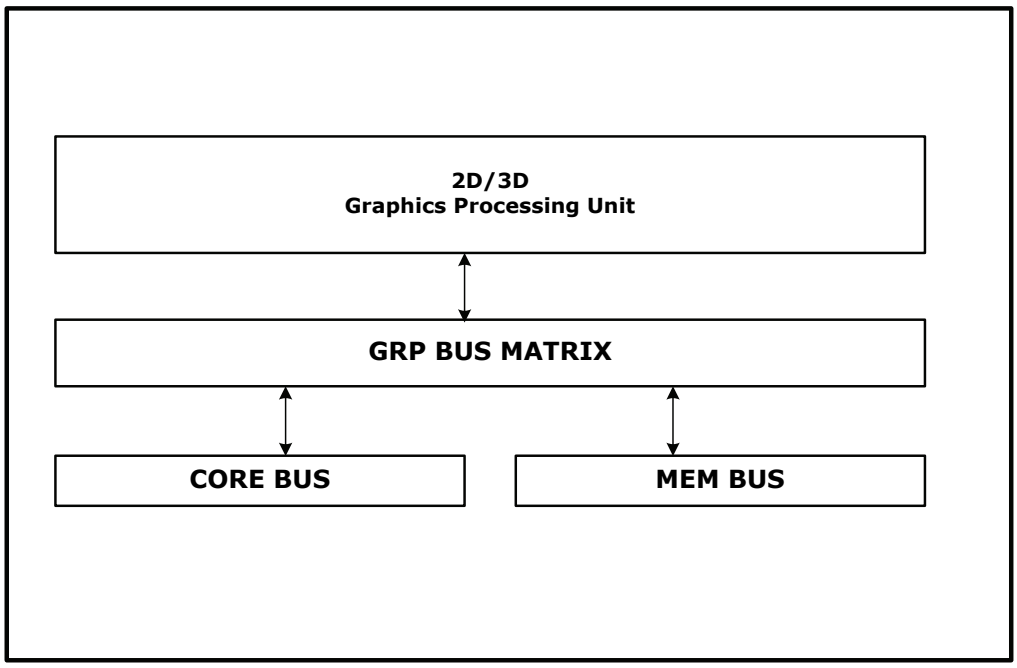


Figure 2.1 The GRP Hardware Bus Architecture

3 Address and Register Map

The NVS2310 has 2D/3D GPU peripherals. These peripherals can be configured appropriately by its own registers that can be accessed through specially allocated address. These address maps are represented in the following table.

Refer to corresponding sections for detail information of each peripheral.

Base Address		Peripherals
0xB0700000		2D/3D GPU
0xB0704000		GRPBUS Configuration

4 2D/3D GPU

4.1 Overview

The 2D/3D GPU is a hardware accelerator for 2D and 3D graphics systems.
The GPU consists of:

- A pixel processor
- A geometry processor
- A Memory Management Unit (MMU)
- Associated software

The GPU and its associated software are compatible with the following graphics standards:

- OpenGL ES 2.0
- OpenGL ES 1.1
- OpenVG 1.0

4.1.1 Pixel Processor Features

The pixel processor features are:

- Programmable fragment shader
- Access to framebuffer from fragment shaders
- Alpha blending
- Arbitrary memory reads and writes
- Complete non-power-of-2 texture support
- Cube mapping
- Dynamic recursion
- Fast dynamic branching
- Fast trigonometric functions, including arctangent
- Full floating-point arithmetic
- Framebuffer blend with destination Alpha
- High Dynamic Range (HDR) textures and framebuffers
- Indexable texture samplers
- Line, quad, triangle and point sprites
- Multiple render targets
- No limit on program length
- Perspective Anisotropic Filtering (AF)
- Perspective correct texturing
- Point sampling, bilinear, and trilinear filtering
- Programmable mipmap level-of-detail biasing and replacement
- Register indirect jumps
- Stencil buffering, 8-bit
- Two-sided stencil
- Unlimited dependent texture reads
- Virtualized texture samplers
- 4-level hierarchical Z and stencil operations
- 4 times and 16 times Full Scene Anti-Aliasing (FSAA)
- 4-bit per texel texture compression

4.1.2 Geometry Processor Features

The geometry processor features are:

- Programmable vertex shader
- Autonomous operation tile list generation
- Flexible input and output formats
- Indexed and non-indexed geometry input

- Primitive constructions with points, lines, triangles and quads.

4.2 GPU Structure

This section gives a brief description of the structure of the GPU.

- The pixel processor. It uses a list of primitives generated by the geometry processor to produce a final image that is displayed on the screen. The pixel processor uses a total of 23 SRAM macros. There are 33 instances that yield a total size of 32.76KB.
- A programmable geometry processor that generates lists of primitives for the pixel processor to draw. The geometry processor uses a total of nine SRAM macros. There are 15 instances that yield a total size of 16.81KB
- A full-featured Memory Management Unit (MMU). All memory accesses from the pixel and geometry processor use the MMU for access checking and translation.
- AXI interconnect system bus protocol targeted at high performance, high clock frequency system designs.

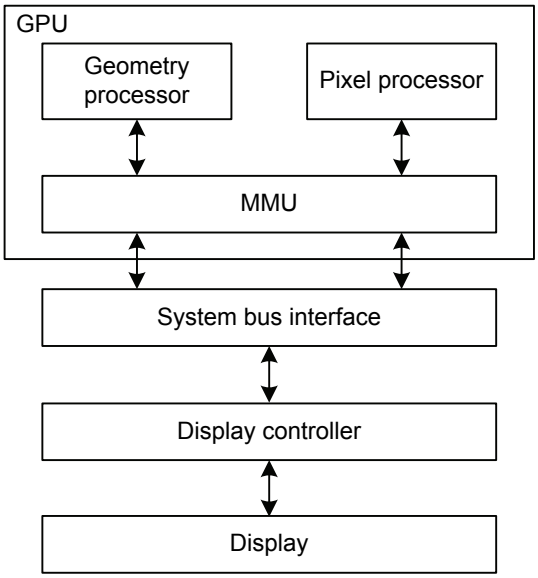


Figure 4.1 2D/3D Graphics System

4.3 Pixel Processor Register Description

The pixel renderer configuration registers consist of an 8KB address space divided by a set of register blocks. The memory extends from a bse address 0x0000 to a maximum address of 0x1F40. Figure 4.2 shows the pixel register map split into resions.

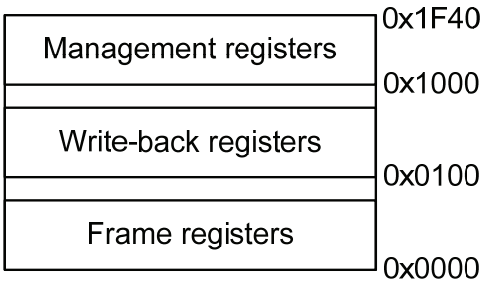


Figure 4.2 Pixel Processor Register Map

Frame registers

The Frame Registers contain frame data that is static during the rendering of a frame, and not required in the memory data structures. Because this data is typically written to in bursts from the driver, the registers are laid out as 18 contiguous registers.

Write-back registers

The processor is equipped with three write-back units WB0(0x0100), WB1(0x0200), and WB2(0x0300), that can be configured independently by three similar blocks. Each register block consists of a unique set of the following registers. In the description, WBx means either WB0, WB1, or WB2 depending on the base address. In the addresses, the underscore means the differentiating nibble between the WBs, for example, 0x0_1C is 0x011C for WB0 0x021C for WB1 and 0x031C for WB2.

Management registers

The Management Registers covering 0x1000-0x10F0 are all control and configuration registers that are not directly connected to rendering of a frame.

Table 4.1 Pixel Processor Register Map (Base Address = 0xB0700000)

Name	Offset	Type	Reset	Description
REND_LIST_ADDR	0x0000	R/W	0x00000000	Renderer List Address
REND_RSW_BASE	0x0004	R/W	0x00000000	Renderer State Word Base Address
REND_VERTEX_BASE	0x0008	R/W	0x00000000	Renderer Vertex Base Address

Renderer List Address Register (REND_LIST_ADDR) **0xB0700000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REND_LIST_ADDR[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REND_LIST_ADDR[15:5]											Reserved				

Field	Name	RW	Reset	Description
31-5	REND_LIST_ADDR	R/W	0x0000000	Renderer List Address

Holds the start address of the polygon list for the current frame
Address value is 32-bytes aligned.

Renderer State Word Base Address Register (REND_RSW_BASE) **0xB0700004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REND_RSW_BASE[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REND_RSW_BASE[15:6]										Reserved					

Field	Name	RW	Reset	Description
31-6	REND_RSW_BASE	R/W	0x0000000	Default renderer state word base address

Holds the default renderer state word base address
Address value is 64-bytes aligned.

Renderer Vertex Base Register (REND_VERTEX_BASE) **0xB0700008**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REND_VERTEX_BASE[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REND_VERTEX_BASE[15:6]										Reserved					

Field	Name	RW	Reset	Description
31-6	REND_VERTEX_BASE	R/W	0x0000000	Default vertex bundles base address

Holds the default base address for vertex bundles
Address value is 64-bytes aligned.

4.4 Geometry Processor Register Description

Figure 4.3 shows that the geometry configuration register has three memory location:

- The Polygon List Builder Configuration Registers are 64Bytes.
- The Control Registers are 168Bytes.
- The Vertex Shader Registers are 304Bytes.

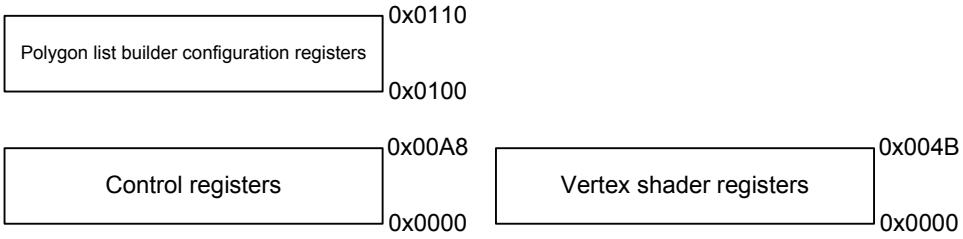


Figure 4.3 Geometry Processor Configuration Register Map

Control registers

The control registers are a set of registers that monitor and control the operation of the geometry processor. The APB bus accesses and controls these registers.

Polygon list builder configuration registers

The PLB configuration registers are a set of registers that control the operation of the polygon list builder, but can only be written from the PLB command list.

Vertex shader registers

The vertex shader configuration registers are s set of registers that control the operation of vertex shader, but can only be written from the vertex shader command list.

To use Polygon list builder configuration registers and Vertex shader registers:

1. Create a command list.
2. Put the command list into memory.
3. Enable end of command list interrupt.
4. Use the APB bus to set the start and end of the command list.
5. Start the command list processor by writing to the Command Register in the APB bus.
6. Wait for interrupt.

Table 4.2 Geometry Processor Control Register Map (Base Address = 0xB0700000)

Name	Offset	Type	Reset	Description
CONTR_REG_VSCL_START_ADDR	0x2000	R/W	0x00000000	Control Register VSCL Start Address
CONTR_REG_VSCL_END_ADDR	0x2004	R/W	0x00000000	Control Register VSCL End Address
CONTR_REG_PLBCL_START_ADDR	0x2008	R/W	0x00000000	Control Register PLBCL Start Address
CONTR_REG_PLBCL_END_ADDR	0x200C	R/W	0x00000000	Control Register PLBCL End Address
CONTR_REG_PLB_ALLOC_START_ADDR	0x2010	R/W	0x00000000	Control Register PLB Allocate Start Address
CONTR_REG_PLB_ALLOC_END_ADDR	0x2014	R/W	0x00000000	Control Register PLB Allocate End Address

Table 4.3 Geometry Processor PLB Configuration Register Map

Name	Offset	Type	Reset	Description
PLB_CONF_REG_VERTEX_ARRAY_ADDR	0x0100	W	0x00000000	PLB Configuration Register Vertex Array Address
PLB_CONF_REG_INDEX_ARRAY_ADDR	0x0101	W	0x00000000	PLB Configuration Register Index Array Address
PLB_CONF_REG_POINT_SIZE_ADDR	0x0102	W	0x00000000	PLB Configuration Register Point Size Address
PLB_CONF_REG_HEAP_START_ADDR	0x0103	W	0x00000000	PLB Configuration Register Heap Start Address
PLB_CONF_REG_HEAP_END_ADDR	0x0104	W	0x00000000	PLB Configuration Register Heap End Address

Table 4.4 Geometry Processor Vertex Shader Register Map

Name	Offset	Type	Reset	Description
VS_CONF_REG_INP_ADDR	0x0000-0x001E	W	0x00000000	VS Configuration Register Input Address
VS_CONF_REG_INP_SPEC	0x0001-0x001F	W	0x0000003F	VS Configuration Register Input Specifier
VS_CONF_REG_OUTP_ADDR	0x0020-0x003E	W	0x00000000	VS Configuration Register Output Address
VS_CONF_REG_OUTP_SPEC	0x0021-0x003F	W	0x0000003F	VS Configuration Register Output Specifier

Control Register VSCL Start Address Register (CONRT_REG_VSCL_START_ADDR) 0xB0702000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VSCL_START_ADDR[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VSCL_START_ADDR[15:3]													Reserved		

Field	Name	RW	Reset	Description
31-3	VSCL_START_ADDR	R/W	0x00000000	Start address of the VSCL

Reading from the CONTR_REG_VSCL_START_ADDR Register at 0x0000 returns the current command list item being processed. You can read the actual register value from register 0x0080.

Writing to the CONTR_REG_VSCL_START_ADDR Register sets the start address of the vertex shader command list. The value is not visible until the vertex shader is started.
Address value is 8-bytes aligned.

Control Register VSCL End Address Register (CONTR_REG_VSCL_END_ADDR) 0xB0702004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VSCL_END_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VSCL_END_ADDR[15:3]													Reserved		

Field	Name	RW	Reset	Description
31-3	VSCL_END_ADDR	R/W	0x00000000	End Address of the VSCL

The CONTR_REG_VSCL_END_ADDR Register provides the end address of the VSCL.
Address value is 8-bytes aligned.

Control Register PLBCL Start Address Register (CONTR_REG_PLBCL_START_ADDR) 0xB0702008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PLBCL_START_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLBCL_START_ADDR[15:3]													Reserved		

Field	Name	RW	Reset	Description
31-3	PLBCL_START_ADDR	R/W	0x00000000	Start address of the PLBCL

Reading from CONTR_REG_PLBCL_START_ADDR at 0x0008 returns the current command list item being processed. You can read the actual register value from register 0x0084.

Writing to the CONTR_REG_PLBCL_START_ADDR Register sets the start address of the PLB command list. The value is not visible until the vertex shader is started.
Address value is 8-bytes aligned.

Control Register PLBCL End Address Register (CONTR_REG_PLBCL_END_ADDR) 0xB070200C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PLBCL_END_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLBCL_END_ADDR[15:3]													Reserved		

Field	Name	RW	Reset	Description
31-3	PLBCL_END_ADDR	R/W	0x00000000	Start address of the VSCL

The CONTR_REG_PLBCL_END_ADDR Register provides the end address of the PLBCL. When the PLBCL execution reaches the end address, the command list processing terminates, and the IRQ_PLB_END_CMD_LIST interrupt is asserted. The command at this address is not executed.
Address value is 8-bytes aligned.

Control Register PLB Allocation Start Address Register (CONTR_REG_PLB_ALLOC_START_ADDR) 0xB0702010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PLB_ALLOC_START_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLB_ALLOC_START_ADDR[15:7]									Reserved						

Field	Name	RW	Reset	Description
31-7	PLB_ALLOC_START_ADDR	R/W	0x0000000	Start/current address for the polygon list allocation

Writing to the CONTR_REG_PLB_ALLOC_START_ADDR Register sets the start address for polygon list allocation. The register does not show the new value until the CMD_UPDATE_PLB_ALLOC command is given through the CONTR_REG_CMD Register.

Reading from this register returns the current address for polygon list allocation.
Address value is 128-bytes aligned.

Control Register PLB Allocation End Address Register (CONTR_REG_PLB_ALLOC_END_ADDR) 0xB0702014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PLB_ALLOC_END_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLB_ALLOC_END_ADDR[15:7]									Reserved						

Field	Name	RW	Reset	Description
31-7	PLB_ALLOC_END_ADDR	R/W	0x0000000	End address for the polygon list allocation

The CONTR_REG_PLB_ALLOC_END_ADDR Register provides the end address for polygon list allocation. If the current address for polygon list allocation reaches this address, the PLB is stalled and the IRQ_PLB_OUT_OF_MEM interrupt is asserted. The register does not show the new value until the CMD_UPDATE_PLB_ALLOC command is given through the CONTR_REG_CMD Register.
Address value is 128-bytes aligned.

PLB Configuration Register Vertex Array Address Register (PLB_CONF_REG_VERTEX_ARRAY_ADDR) 0x0100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VERTEX_ARRAY_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VERTEX_ARRAY_ADDR[15:0]															

Field	Name	RW	Reset	Description
31-0	VERTEX_ARRAY_ADDR	W	0x00000000	Vertex data array base address

The PLB_CONF_REG_VERTEX_ARRAY_ADDR Register is write-only. The vertex data array base address, used for glDrawElements mode, is written to this register.

PLB Configuration Register Index Array Address Register (PLB_CONF_REG_INDEX_ARRAY_ADDR) 0x0101

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INDEX_ARRAY_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INDEX_ARRAY_ADDR[15:0]															

Field	Name	RW	Reset	Description
31-0	INDEX_ARRAY_ADDR	W	0x00000000	Vertex index array base address

The vertex index array base address, used for glDrawElements mode, is written to the PLB_CONF_REG_INDEX_ARRAY_ADDR

PLB Configuration Register Point Size Address Register (PLB_CONF_REG_POINT_SIZE_ADDR) 0x0102

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
POINT_SIZE_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
POINT_SIZE_ADDR[15:0]															

Field	Name	RW	Reset	Description
31-0	POINT_SIZE_ADDR	W	0x00000000	Point size address

The point size address, used for glDrawElements mode, is written to the PLB_CONF_REG_POINT_SIZE_ADDR Register.

PLB Configuration Register Heap Start Address Register (PLB_CONF_REG_HEAP_START_ADDR) 0x0103

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HEAP_START_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HEAP_START_ADDR[15:7]									Reserved						

Field	Name	RW	Reset	Description
31-7	HEAP_START_ADDR	W	0x0000000	Polygon list heap start address

The PLB_CONF_REG_VERTEX_ARRAY_ADDR Register is write-only. The vertex data array base address, used for glDrawElements mode, is written to this register.
Address value is 128-bytes aligned.

PLB Configuration Register Heap End Address Register (PLB_CONF_REG_HEAP_END_ADDR) 0x0104

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HEAP_END_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HEAP_END_ADDR[15:7]									Reserved						

Field	Name	RW	Reset	Description
31-7	HEAP_END_ADDR	W	0x0000000	Polygon list heap end address

The polygon list heap end address is written to the PLB_CONF_HEAP_END_ADDR Register.
Address value is 128-bytes aligned.

VS Configuration Register Input Address(X) Register (VS_CONF_REG_INP_ADDR) 0x0000+(2*X)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INP_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INP_ADDR[15:0]															

Field	Name	RW	Reset	Description
31-0	INP_ADDR	W	0x00000000	Base address of vertex array

The VS_CONF_REG_INP_ADDR(X) Register contains the base data address for input data stream X.

VS Configuration Register Input Specifier(X) Register (VS_CONF_REG_INP_SPEC) 0x0001+(2*X)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BE	STRIDE[30:16]														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STRIDE[15:11]					COMMA[10:6]					VERTEX_DATA_FORMAT[5:0]					

Field	Name	RW	Reset	Description
31	BE	W	0X0	Big Endian 0= The value is stored in little-endian byte order, and is converted on loading. 1= The value is stored in big-endian byte order, and is converted on loading.
30-11	STRIDE	W	0X00000	Stride Value in 1-byte unit. The stride is defined as the number of bytes separating two consecutive vertices.
10-6	COMMA	W	0X00	Comma location, for fixed point formats only.
5:0	Vertex Data Format	W	0x3F	See Table 4.5 for vertex data format value. The reset value is 0x3F

The PLB_CONF_REG_Z_FAR Register holds the far Z-plane position in FP32 format.

Table 4.5 VS_CONF_REG_INP_SPEC Vertex Data Format

Value	Name	Format
[63]	VS_VSTREAM_NO_DATA	No data
[62:60]	Unused	Reserved
[59:56]	VS_VSTREAM_FORMAT_1_NORM_U32 VS_VSTREAM_FORMAT_4_NORM_U32	1-4 32-bit unsigned normalized
[55:52]	VS_VSTREAM_FORMAT_1_NORM_S32 VS_VSTREAM_FORMAT_4_NORM_S32	1-4 32-bit signed normalized
[51:48]	VS_VSTREAM_FORMAT_1_FP24 VS_VSTREAM_FORMAT_4_FP24	1-4 floats, FP24
[47:44]	VS_VSTREAM_FORMAT_1_NORM_U16 VS_VSTREAM_FORMAT_4_NORM_U16	1-4 16-bit unsigned normalized
[43:40]	VS_VSTREAM_FORMAT_1_NORM_S16 VS_VSTREAM_FORMAT_4_NORM_S16	1-4 16-bit signed normalized
[39:36]	VS_VSTREAM_FORMAT_1_NORM_U8 VS_VSTREAM_FORMAT_4_NORM_U8	1-4 8-bit unsigned normalized
[35:32]	VS_VSTREAM_FORMAT_1_NORM_S8 VS_VSTREAM_FORMAT_4_NORM_S8	1-4 8-bit signed normalized
[31:28]	VS_VSTREAM_FORMAT_1_FIX_U8 VS_VSTREAM_FORMAT_4_FIX_U8	1-4 8-bit fixed point, unsigned
[27:24]	VS_VSTREAM_FORMAT_1_FIX_S8 VS_VSTREAM_FORMAT_4_FIX_S8	1-4 8-bit fixed point, unsigned
[23:20]	VS_VSTREAM_FORMAT_1_FIX_U16 VS_VSTREAM_FORMAT_4_FIX_U16	1-4 16-bit fixed point, unsigned
[19:16]	VS_VSTREAM_FORMAT_1_FIX_S16 VS_VSTREAM_FORMAT_4_FIX_S16	1-4 16-bit fixed point, signed
[15:12]	VS_VSTREAM_FORMAT_1_FP16 VS_VSTREAM_FORMAT_4_FP16	1-4 floats, FP16
[11:8]	VS_VSTREAM_FORMAT_1_FIX_U32 VS_VSTREAM_FORMAT_4_FIX_U32	1-4 32-bit fixed point, unsigned
[7:4]	VS_VSTREAM_FORMAT_1_FIX_S32 VS_VSTREAM_FORMAT_4_FIX_S32	1-4 32-bit fixed point, signed
[3:0]	VS_VSTREAM_FORMAT_1_FP32 VS_VSTREAM_FORMAT_4_FP32	1-4 floats, FP32

VS Configuration Register Output Specifier(X) Register (VS_CONF_REG_OUTP_SPEC) 0x0020+(2*X)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OUTP_ADDR[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OUTP_ADDR[15:0]															

Field	Name	RW	Reset	Description
31-0	OUTP_ADDR	W	0x00000000	Base address of the output vertex stream specifier

The VS_CONF_REG_OUTP_ADDR(X) Register contains the base data address for output data stream X.

VS Configuration Register Output Specifier(X) Register (VS_CONF_REG_OUTP_SPEC) 0x0021+(2*X)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BE	STRIDE[30:16]														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STRIDE[15:11]					COMMA[10:6]					VERTEX_DATA_FORMAT[5:0]					

Field	Name	RW	Reset	Description
31	BE	W	0x0	Big Endian 0= The value is stored in little-endian byte order, and is converted on loading. 1= The value is stored in big-endian byte order, and is converted on loading.
30-11	STRIDE	W	0x00000	Stride Value in 1-byte unit. The stride is defined as the number of bytes separating two consecutive vertices.
10-6	COMMA	W	0x00	Comma location, for fixed point formats only.
5-0	Vertex Data Format	W	0x3F	See Table 4.6 for vertex data format value. The reset value is 0x3F

The PLB_CONF_REG_Z_FAR Register holds the far Z-plane position in FP32 format.

Table 4.6 VS_CONF_REG_OUTP_SPEC Vertex Data Format

Value	Name	Format
[63]	VS_VSTREAM_NO_DATA	No data
[62:53]	Unused	Reserved
[51:48]	VS_VSTREAM_FORMAT_FP24 VS_VSTREAM_FORMAT_FP24	1-4 floats, FP24
[47]	Unused	Reserved
[46]	VS_VSTREAM_FORMAT_RGB565	RGB 565, output only
[45:34]	Unused	Reserved
[33]	VS_VSTREAM_FORMAT_POINT_SIZE	Point size in FP32 format, output only
[32]	VS_VSTREAM_FORMAT_VDATA_BLOCK(XYZW)	Vertex co-ordinates in FP32 format, output only
[31:28]	VS_VSTREAM_FORMAT_1_FIX_U8 VS_VSTREAM_FORMAT_4_FIX_U8	1-4 8-bit fixed point, unsigned
[27:24]	VS_VSTREAM_FORMAT_1_FIX_S8 VS_VSTREAM_FORMAT_4_FIX_S8	1-4 8-bit fixed point, unsigned
[23:20]	VS_VSTREAM_FORMAT_1_FIX_U16 VS_VSTREAM_FORMAT_4_FIX_U16	1-4 16-bit fixed point, unsigned
[19:16]	VS_VSTREAM_FORMAT_1_FIX_S16 VS_VSTREAM_FORMAT_4_FIX_S16	1-4 16-bit fixed point, signed
[15:12]	VS_VSTREAM_FORMAT_1_FP16 VS_VSTREAM_FORMAT_4_FP16	1-4 floats, FP16
[11:8]	VS_VSTREAM_FORMAT_1_FIX_U32 VS_VSTREAM_FORMAT_4_FIX_U32	1-4 32-bit fixed point, unsigned
[7:4]	VS_VSTREAM_FORMAT_1_FIX_S32 VS_VSTREAM_FORMAT_4_FIX_S32	1-4 32-bit fixed point, signed
[3:0]	VS_VSTREAM_FORMAT_1_FP32 VS_VSTREAM_FORMAT_4_FP32	1-4 floats, FP32

4.5 MMU configuration Register Description

Table 4.7 MMU Configuration Register Map (Base Address = 0xB0700000)

Name	Offset	Type	Reset	Description
MMU_DTE_ADDR	0x3000	R/W	0x00000000	MMU Current Page Table Address

MMU Current Page Table Address Register (MMU_DTE_ADDR) **0xB0703000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DTE_ADDR[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DTE_ADDR[15:0]															

Field	Name	RW	Reset	Description
31-0	DTE_ADDR	R/W	0x00000000	Value

The MMU_DTE_ADDR Register contains the base address of the current page tables. The address must be 4KB aligned. This register can only be written when the MMU is idle or stalled.

5 GRPBUS Configuration

The GRPBUS Configuration block has several registers named as PWRDOWN, SWRESET, GPU_IDLE, BWRAPCTRL.

Table 5.1 GRPBUS Configuration Register Map (Base Address = 0xB0704000)

Name	Offset	Type	Reset	Description
GRPBUS_PWRDOWN	0x0000	R/W	0x00000000	Graphics bus power down
GRPBUS_SWRESET	0x0004	R/W	0x00000000	Graphics bus software reset
GRPBUS_GPU_IDLE	0x0008	R/W	0x00000003	GPU idle configuration

Graphic Bus Power Down Register (GRPBUS_PWRDOWN) 0xB0704000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															GRP

Field	Name	RW	Reset	Description
0	GRPBUS SLEEP	R/W	0x00000000	GRPBUS SLEEP 0= 3D GPU normal operation 1= 3D GPU power down

Graphic Bus Software Reset Register (GRPBUS_SWRESET) 0xB0704004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															GRP

Field	Name	RW	Reset	Description
0	GRPBUS SWRESET	R/W	0x00000000	GRPBUS SWRESET 0= 3D GPU keep working 1= 3D GPU software reset active state

GPU Idle Configuration Register (GRPBUS_GPU_IDLE) 0xB0704008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													BOTH	GP	PP

Field	Name	RW	Reset	Description
2	BOTH	R/W	0	GRPBUS SWRESET 0= 3D GPU keep working 1= 3D GPU software reset active state
1	GP	R/W	1	3D Geometry Processor Idle 0= The clock signal is enabled regardless of IDLE signal of Geometry Processor. If BOTH is high, this field will be ignored. 1= The clock signal is disabled regardless of IDLE signal of Geometry Processor. If BOTH is high, this field will be ignored.
0	PP	R/W	1	3D Pixel Processor Idle 0= The clock signal is enabled regardless of IDLE signal of Pixel Processor. If BOTH is high, this field will be ignored. 1= The clock signal is disabled regardless of IDLE signal of Pixel Processor. If BOTH is high, this field will be ignored.

This register is for IDLE configuration of 3D GPU. Geometry Processor and Pixel Processor has it's own IDLE bit, which indicates whether the status is in IDLE or WORKING. And each bit can enable or disable the internal clock signal.

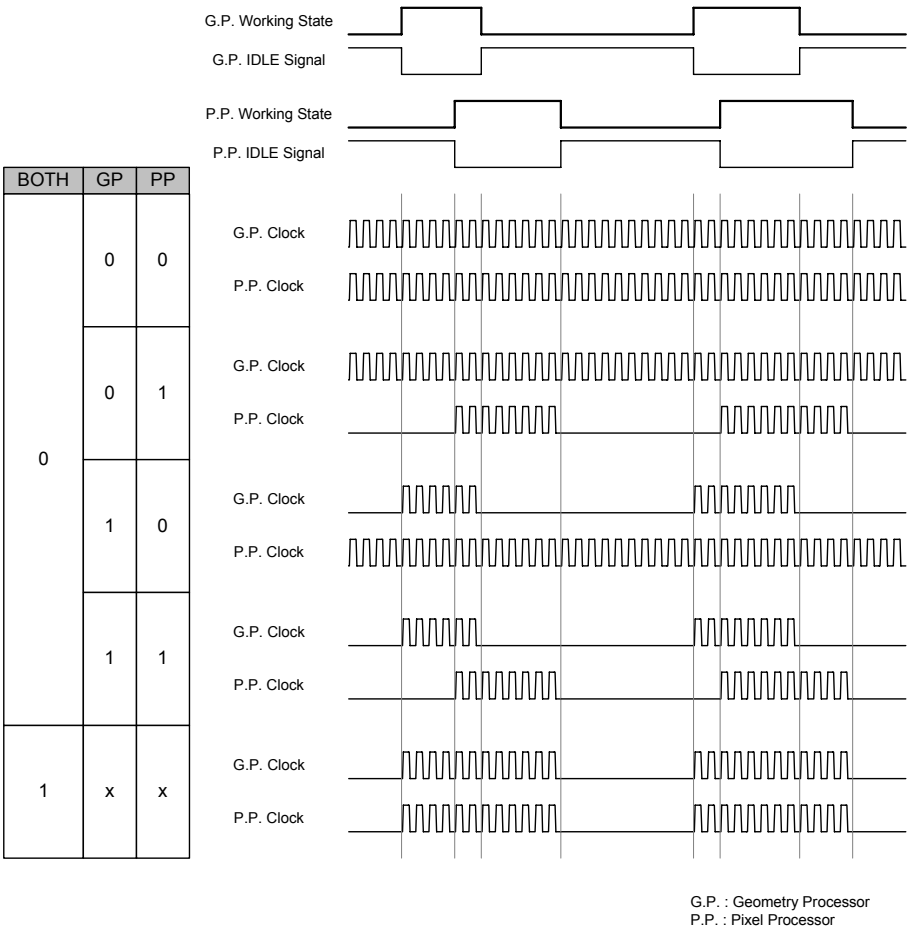


Figure 5.1 GPU Idle Configuration Register and Clock

PART4 – MEMORY BUS
NVS2310

Rev. 1.01

Apr 22, 2011

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial Release
2011-04-22	1.01	∗ Correct the register table and document format. ∗ Correct HCLKMASK register description

TABLE OF CONTENTS

Contents

Figures

Tables

1 Memory Bus Architecture

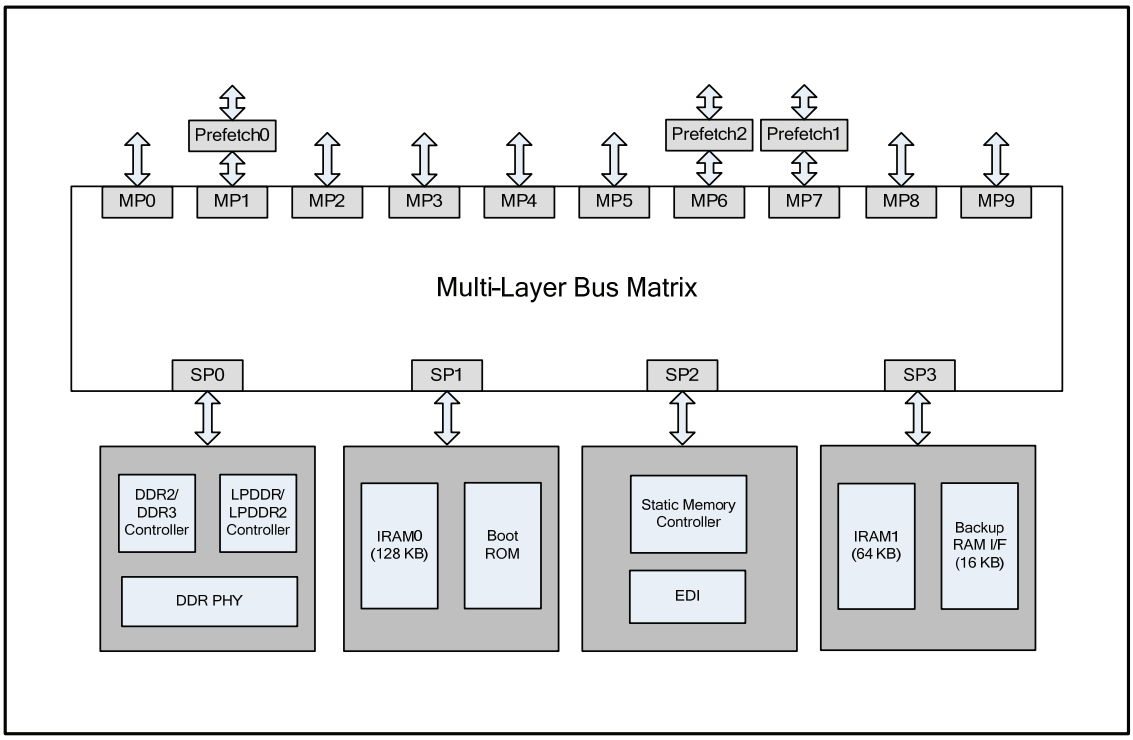


Figure 1.1 The NVS2310 Memory Bus Architecture

The memory bus of NVS2310 consist of the high performance multi-layer bus matrix, the DDR memory I/F, internal memory I/F and the static memory I/F as shown in the above figure. The internal memory I/F is separated between IRAM0 and IRAM1. Thus simultaneous access from the different bus masters to IRAM0 and IRAM1 is possible. The DDR memory I/F consist of LPDDR/LPDDR2 memory controllers and DDR2/DDR3 controllers and DDR PHY. The high performance multi-layer bus matrix consists of 10 bus master ports and 4 slave ports for connecting all the internal bus masters to the various memory interfaces. The below table shows the defined master/slave port connections of NVS2310.

Table 1.1 Master ports configuration

Master Ports	Description
Master Port 0 (MP0)	ARM Instruction
Master Port 1 (MP1)	ARM Data
Master Port 2 (MP2)	Video Bus (secondary)
Master Port 3 (MP3)	Video Bus (primary)
Master Port 4 (MP4)	Graphic Bus
Master Port 5 (MP5)	Overlay Mixer
Master Port 6 (MP6)	High Speed IO Bus (HSIO Bus)
Master Port 7 (MP7)	IO Bus
Master Port 8 (MP8)	Camera Bus
Master Port 9 (MP9)	Display Bus

Table 1.2 Slave ports configuration

Master Ports	Description
Slave Port 0 (SP0)	DDR Bus
Slave Port 1 (SP1)	IRAM0 Bus
Slave Port 2 (SP2)	Static Memory Bus
Slave Port 3 (SP3)	IRAM1 Bus

The prefetch buffers are instantiated in front of the MP1(for ARM data), MP6(fro high speed IO bus) and MP7(IO bus) master ports for reducing the overall bandwidth.

2 Address and Register Map

The NVS2310 provides versatile DDR SDRAM memory interfaces such as LPDDR(mDDR), LPDDR2(mDDR2), DDR2 and DDR3. The NVS2310 supports various external static memory access such as NOR, SRAM, IDE type memories and also provides three separated internal memories(128KB, 64 KB, 16 KB, IRAM0, IRAM1 and backup RAM respectively)
The overall address map of the functional blocks in memory bus are described in below table.

Table 2.1 Block Base Address

Base Address		Peripherals
0xB0300000		Multi-layer Bus Matrix
0xB0301000		LPDDR SDRAM Controller Registers
0xB0302000		DDR2 SDRAM Controller Registers
0xB0303000		DDR I/F Configuration Registers
0xB0304400		DDR3 PHY Configuration Registers
0xB0305000		LPDDR/LPDDR2/DDR2 SDRAM Controller Registers
0xB0306000		Static Memory Controller Registers
0xB0307000		Internal Memory I/F Configuration Registers
0xB0308000		EDI (External Device Interface) Registers
0xB0309000		Reserved
0xB030A000		Reserved
0xB030B000		Memory Bus Configuration Registers
0xB030C000		DDR2/DDR3 SDRAM Controller Registers
0xC0000000		Hardware MMU Table.

Refer to corresponding sections for detail information of each base address

3 Multi-layer Bus Matrix

3.1 Overview

The multi-layer bus of the memory bus provides the priority control for the master ports and QoS service.

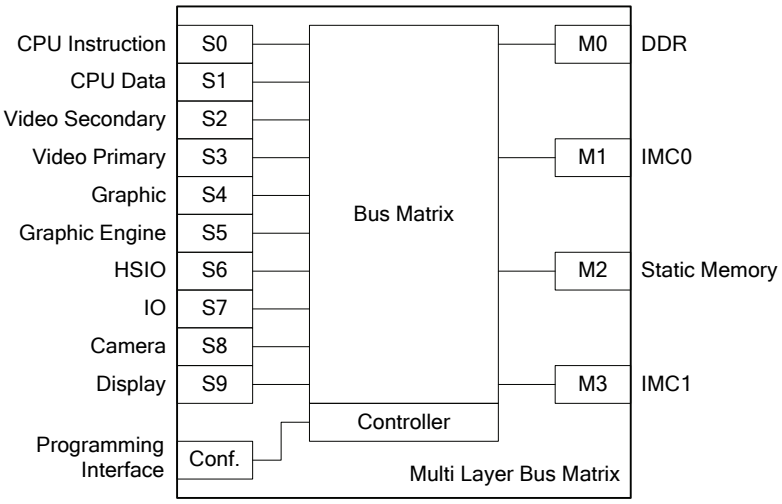


Figure 3.1 Multi-layer Bus Matrix Block Diagram

3.2 Priority Control

The Multi-layer Bus Matrix has a programmable Least Recently Granted (LRG) scheme.

The AW and AR channels have separate arbiters and can be programmed, if applicable, and interrogated separately through the APB programming interface, but both AW and AR channels are configured identically. Because the AW and AR channels are arbitrated separately, an MI can permit simultaneous read and write transactions from different SIs.

The arbitration mechanism registers the arbitration decision for use in the subsequent cycle. An arbitration decision taken in the current cycle does not affect the current cycle.

If no SIs are active, the arbiter adopts default arbitration, that is, the highest priority SI. If this occurs and then the highest priority interface becomes active in the same cycle as, or before any other SI, then this does not constitute a grant to an active SI and the arbitration scheme does not change its state as a result of that transfer.

If a QoS provision is enabled and active, only a subset of SIs are permitted to win arbitration, and it cannot be guaranteed that the default arbitration is among these. In these circumstances, no transaction is permitted to use the default arbitration, and arbitration must occur when there is an active SI.

3.3 QoS Service

The QoS scheme works by tracking the number of outstanding transactions, and when a specified number is reached, only permits transactions from particular, specified masters.

The QoS scheme has no effect until the AXI bus matrix calculates that, at a particular MI, there are a number of outstanding transactions equal to the value stored in the QoS tidemark Register. It then accepts transactions only from slave ports specified in the QoS access control Register. This restriction remains until the number of outstanding transactions is again less than the value stored in the QoS tidemark Register.

3.4 Register Descriptions

Table 3.1 Multi-layer Bus Matrix Register Map (Base Address = 0xB0300000)

Name	Offset	Type	Reset	Description
QT0	0x400	RW	0	QoS Tidemark for MI0
QA0	0x404	RW	0	QoS Access Control for MI0
AR0	0x408	RW	0	AR channel arbitration value for MI0
AW0	0x40C	RW	0	AW channel arbitration value for MI0
Reserved	0x410-0x41C	-	-	Reserved
QT1	0x420	RW	0	QoS Tidemark for MI1
QA1	0x424	RW	0	QoS Access Control for MI1
AR1	0x428	RW	0	AR channel arbitration value for MI1
AW1	0x42C	RW	0	AW channel arbitration value for MI1
Reserved	0x430-0x43C	-	-	Reserved
QT2	0x440	RW	0	QoS Tidemark for MI2
QA2	0x444	RW	0	QoS Access Control for MI2
AR2	0x448	RW	0	AR channel arbitration value for MI2
AW2	0x44C	RW	0	AW channel arbitration value for MI2
Reserved	0x450-0x45C	-	-	Reserved
QT3	0x460	RW	0	QoS Tidemark for MI3
QA3	0x464	RW	0	QoS Access Control for MI3
AR3	0x468	RW	0	AR channel arbitration value for MI3
AW3	0x46C	RW	0	AW channel arbitration value for MI3

QTx (QoS tidemark for MIx) **0xB00004n[1]0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TIDEMARK							

Field	Name	RW	Reset	Description
31-8	Reserved	R	0	Reserved
7-0	TIDEMARK	R/W	0	QoS Tidemark 0 : QoS disable Others : the number of outstanding transactions that are permitted before the QoS scheme becomes active

[1] MI number x 2, ex) MI3 (IMC1) : n = 3*2 = 6

QAx (QoS access control for Mix) 0xB00004n[1]4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						EN									

Field	Name	RW	Reset	Description
31-10	Reserved	R	0	Reserved
9-0	EN	R/W	0	QoS Enable 1 : QoS Enable 0 : QoS Disable EN[0] – CPU Instruction EN[1] – CPU Data EN[2] – Video Slave EN[3] – Video Primary EN[4] – Graphic EN[5] – Graphic Engine EN[6] – HSIO EN[7] – IO EN[8] – Camera EN[9] – Display

ARx (AR channel arbitration value for MIx) 0xB00004n[1]8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SI / Reserved								Reserved							
15	14	13	12	11	10	9	8	7	6	5	4	S3	2	1	0
PRI								Reserved / SI							

Field	Name	RW	Reset	Description
31-24	SI	W	-	Slave Interface Number
23-16	Reserved	W	-	Should be 0
15-8	PRI	W	-	These bits set the interface priority value for the arbitration scheme. The range of values permitted is from 0 to 255 where: 0 is the highest priority 255 is the lowest priority
7-0	Reserved	W	-	Should be 0

To reading this register, you must perform a write followed by a read operation to the same address. The write transfer sets the interface number that you want to access. The following read operation returns the priority value for that interface number.
The write data is encoded with bits [31:24] set to 0xFF and the interface number whose value is to be returned encoded in bits [7:0], with all other bits set to 0.

Field	Name	RW	Reset	Description
31-16	Reserved	R	0	reserved
15-8	PRI	R	-	These bits set the interface priority value for the arbitration scheme. The range of values permitted is from 0 to 255 where: 0 is the highest priority 255 is the lowest priority
7-0	SI	R	-	Should be 0

[1] MI number x 2, ex) MI3 (IMC1) : n = 3*2 = 6

AWx (AW channel arbitration value for Mix) 0xB00004n[1]C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SI / Reserved								Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**S3**	**2**	**1**	**0**
PRI								Reserved / SI							

Field	Name	RW	Reset	Description
31-24	SI	W	-	Slave Interface Number
23-16	Reserved	W	-	Should be 0
15-8	PRI	W	-	These bits set the interface priority value for the arbitration scheme. The range of values permitted is from 0 to 255 where: 0 is the highest priority 255 is the lowest priority
7-0	Reserved	W	-	Should be 0

To reading this register, you must perform a write followed by a read operation to the same address. The write transfer sets the interface number that you want to access. The following read operation returns the priority value for that interface number.
The write data is encoded with bits [31:24] set to 0xFF and the interface number whose value is to be returned encoded in bits [7:0], with all other bits set to 0.

Field	Name	RW	Reset	Description
31-16	Reserved	R	0	reserved
15-8	PRI	R	-	These bits set the interface priority value for the arbitration scheme. The range of values permitted is from 0 to 255 where: 0 is the highest priority 255 is the lowest priority
7-0	SI	R	-	Should be 0

4 DDR SDRAM Memory Controller

4.1 Overview

The NVS2310 has a DDR SDRAM controller for various kinds of SDRAM for digital media en-decoding system. It can manipulate LPDDR, LPDDR2, DDR2 and DDR3 SDRAM memories. The data bus width can be configured for each chip select separately
The memory controller provides the power saving function for DDR SDRAM (self refresh).

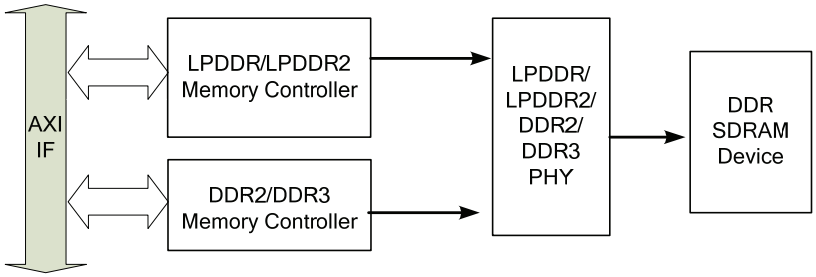


Figure 4.1 DDR SDRAM Controller Block Diagram

4.2 LPDDR SDRAM Controller

The LPDDR SDRAM controller is a high-performance, area-optimized Mobile DDR memory controller compatible with the AMBA AXI protocol.

The memory controller block offers the following features:

- scalable pipeline
- interface between AMBA AXI bus fabric and LPDDR (Mobile DDR)
- Quality of Service (QoS) and request arbitration features for low latency transfers and optimal use of memory bandwidth
- write data interleaving supported
- multiple outstanding addresses supported
- support for ARMv6 outstanding exclusive accesses
- support synchronous and asynchronous operation between AXI bus fabric and external memory bus
- programmable support for memory power saving modes including Deep Power Down (DPD), active power-down, precharge power-down and self-refresh
- programmable through AMBA APB interface
- optimized utilization of external memory bus
- configurable for single port for all chip selects on individual cke port per external chip select.

4.3 DDR2 SDRAM Controller

The DDR2 SDRAM controller is a high-performance, area-optimized DDR2 SDRAM memory controller compatible with the AMBA AXI protocol.

The memory controller block offers the following features:

- active and precharge power-down supported in the DDR2 SDRAM
- Quality of Service (QoS) features for low latency transfers
- optimized utilization of external memory bus
- programmable selection of external memory width, see Supported memory widths
- multiple outstanding transactions
- write data interleaving supported
- hardware resource that can be rendered to optimize area versus performance
- support for a configurable number of ARMv6 outstanding exclusive accesses
- support synchronous and asynchronous operation between AXI bus fabric and external memory bus

4.4 LPDDR/LPDDR2/DDR2 SDRAM Controller

To support high speed memory devices, the controller uses the SEC SDRAM PHY interface. The controller includes an advanced embedded scheduler to utilize memory device efficiently and an optimized pipeline stage to minimize latency

- Compatible with JEDEC DDR2, low power DDR and low power DDR2 SDRAM specification
- Supports up to two external chip selects and 1/2/4/8 banks per one chip
- Supports 128Mb, 256Mb, 512Mb, 1Gb, 2Gb, 4Gb and 8Gbit density
- Supports 16/32bit wide memory data width
- Optimized pipeline stage for low latency
- Supports QoS scheme to ensure low latency for real-time applications
- The advanced embedded scheduler enables out-of order operations to utilize memory device efficiently
- Supports excellent chip/bank interleaving and memory interrupting
- Adapts to various low power schemes to reduce the dynamic and static current of memory
- Supports outstanding exclusive accesses
- Supports bank selective precharge policy

4.5 DDR2/DDR3 SDRAM Controller

The DDR2/DDR3 Controller supports DDR2 and DDR3 SDRAM memory Devices. The controller provides the flexibility of interfacing up to two memory chips by making the chip select a configurable feature

The Controller has been designed keeping the objective that it should support variety of configurations of memory of variable standard interfaces. Following is the list of features supported by the Controller.

- Support DDR2, DDR3 memory devices
  - DDR2- x4, x8, x16
  - DDR3- x4, x8, x16
  - Number of devices required for 32 Bits wide Memory Databus depends on Memory Device Width
- All parameters, Special function registers, programmable through APB Interface
  - Programmable row and column address bit widths
  - Support for configurable chip selects, i.e. one or two chip selects (maximum support chip select lines is 2)
  - Configurable number of banks selection : 4 or 8 for DDR2, 8 for DDR3
  - Burst lengths : 4 for DDR2, 4(fixed BC4 i.e. fixed burst chop 4)or 8 for DDR3
  - Configurable memory device timing parameters
  - Programmable CAS latency.
  - Programmable Write latency.
  - Programmable 16 bit auto refresh timer
- Supports multiple data rates for DDR3 and DDR2
- Separate Bank Management block for optimizing memory accesses
- Support Open page policy (keeps row open after completion of write or read command) and Close page policy (closes row after completion of write or read command)
- Support for Quality of Service (QoS) for READ operation.
- Supports power saving features
  - Self refresh through APB interface request
  - Active and Pre-charge power down through APB interface request
  - Dynamic Self refresh during idle AXI cycles
  - Dynamic Power down during idle AXI cycles
  - Dynamic row precharge during idle AXI cycles
  - AXI 4K boundary crossing only supported for external memory devices that have column number of bits less than or equal to 10.
- Supports External Bus Interface of widths 16, 32

4.6 Functional Overview

4.6.1 LPDDR/LPDDR2/DDR2 SDRAM Controller

4.6.1.1 Address Mapping

The controller modifies the address of the bus transaction coming from the AXI slave port into a memory address – chip select, bank address, row address, column address and memory data width.
To map chip select0 of memory device to a specific area of the address map, the chip_base and chip_mask bit-fields of the MemConfig0 register needs to be set (refer to register desciption). If chip1 of the memory device exists, the MemConfig1 register must also be set.
Then, the AXI address requested by AXI Masters is divided into the AXI base address and AXI offset address. The AXI base address activates the appropriate memory chip select and the AXI offset address is mapped to a memory address according to the bank, row, column number, and data width set by the MemConfig0/1 and MemControl register.
There are two ways to map the AXI offset address as shown below: 1) Linear mapping 2) Interleaved mapping.

4.6.1.1.1 Linear Mapping

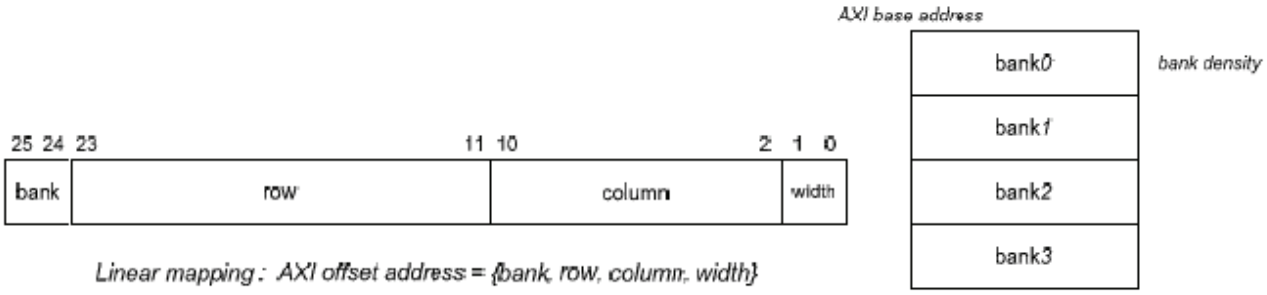


Figure 4.2 Linear Mapping

As shown in the above figure, the linear mapping method maps the AXI address in the order of bank, row, column and width. Since the bank address does not change for at least one bank size, applications that use linear address mapping have a high possibility to access the same bank.

4.6.1.1.2 Interleaved Mapping

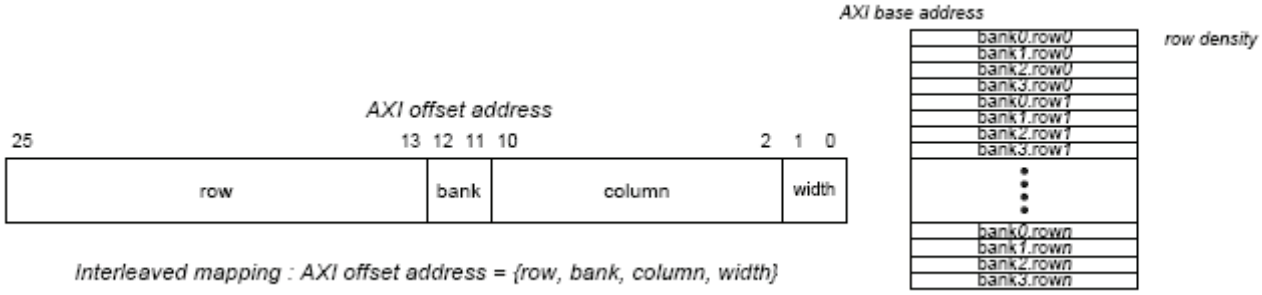


Figure 4.3 Interleaved Mapping

As shown in the above figure, the interleaved mapping method maps the AXI address in the order of row, bank, column and width. The difference between above two methods is that the bank and row order is different. For accesses beyond a row size, interleaved mapping accesses a different bank. Therefore, applications that use interleaved mapping access numerous banks. It causes better performance but more power consumption.

4.6.1.2 Low Power Operation

The controller executes a low power memory operation in four ways as described below. Each feature is independent of each other and executed at the same time.

4.6.1.2.1 Dynamic Power Down

The DDR SDRAM device has an active/ precharge power down mode. This mode is entered if CKE becomes LOW. To enter active power down mode minimum one row of a bank must be open. To enter precharge power down mode CKE must be low.
If no AXI transaction enters the controller and the command queue becomes empty for a specific number of cycles (PwrdnConfig.dpwrdn_cyc register), the controller changes the memory device's state to active/precharge power down automatically. Then, there are two ways to enter the active/precharge power down state.

1) Active/precharge power down mode: Enter power down w/o considering whether there is a row open or not.

2) Forced precharge power down mode: Enter power down after closing all banks.

If a new AXI transaction enters the controller, the controller automatically wakes up the memory device from power down state and executes in a normal operation state.

4.6.1.2.2 Dynamic Self Refresh

Similar to the dynamic power down feature, if the command queue is empty for a specific amount of cycles (PwrdnConfig.dsref_cyc register**)**, the memory device enters self-refresh mode. Since exiting power down mode requires many cycles, we recommend to choose a greater cycle size for dynamic self-refresh entry than dynamic power down.

4.6.1.2.3 Clock Stop

To reduce the I/O power of the memory device and the controller, it is possible to stop the clock if the LPDDR / LPDDR2 is in idle mode, or self refresh mode and DDR2 is in self refresh mode. If this feature is enabled, the controller automatically executes the clock stop feature.

4.6.1.2.4 Direct Command

Use the direct command feature to send a command to the memory device through the APB port. This way, you force the memory device to enter active/ precharge power down, self-refresh or deep power down mode.

4.6.1.3 Quality of Service

QoS is defined as a method to increase the arbitration priority of a master that requires low latency read/write data. The QoS is determined if the control queue receives the command through the AXI bus and the QoS count starts depreciating at this moment. If the count reaches zero, this command becomes the highest priority among the other commands that are in the control queue.

There are three types of QoS.

- qos_cnt
- qos_cnt_f
- default_qos

4.6.1.3.1 qos_cnt

There are 16 configurable QoSControls, which have independent qos_masks that masks the AXID from one bit up to the AXID width. All 16 QoSControls are either be enabled or disabled,

1) If the command is received via the AXI bus, the AXID is masked by the qos_masks **QoSConfig(n).qos_mask**) from the 16 QoSControls that are enabled.

2) The masked results are then compared to the qos_ids (**QoSConfig**(**n**).**qos_id**). If any one of the result are equal, the qos_cnt (**QoSControl**(**n**).**qos_cnt**) value is applied to the command and saved in the control queue.

4.6.1.3.2 qos_cnt_f

To service latency sensitive commands faster, an adaptive DDR SDRAM QoS scheme called QoS fast can be enabled. This policy cannot be done by the memory controller itself, but the IP has to observe its FIFO level.
For read transactions, for example, when the IP's FIFO is less than 1/4th full, there is no margin of time available between the FIFO and the memory controller. At this moment, if the IP flags the memory controller, the qos_cnt_f value that is specified for the IP is applied to the command to give a higher QoS priority over other IP commands.

For write transactions, for example, when the IP's FIFO is more than 3/4th full, there is almost no margin of time available before the FIFO becomes full. At this moment, if the IP flags the memory controller, the qos_cnt_f value that is specified for the IP is applied to the command to give a higher QoS priority over other IP commands.
Figure 5 shows the adaptive DRAM QoS scheme configuration in SoC.

4.6.1.4 Read Data Capture

A memory device that receives a read command sends the data to the controller after a read latency (i.e. CAS latency). After clearing the DQS, the PHY uses the PHY DLL to phase shift the DQS 90 degrees. Using the shifted DQS, the PHY samples the read data and saves the data into the read data input FIFO, which is located inside the PHY. Then, the controller fetches the data from the PHY while considering the read latency and the read fetch delay, and then sends it to the AXI read channel. The following figures show the read data capture process's timing diagram for each memory type.

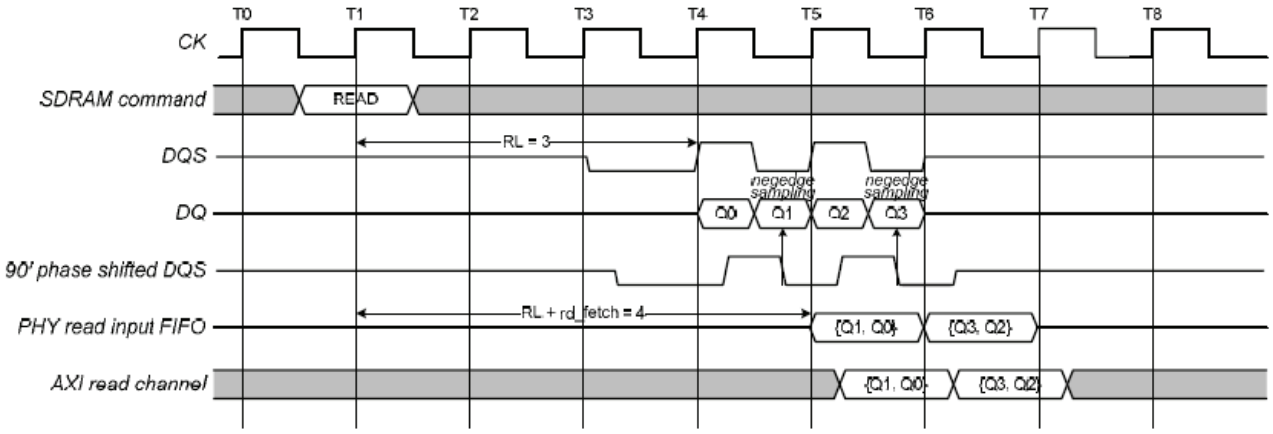


Figure 4.4 Timing diagram of read data capture (DDR/DDR2, zero delay, RL=3, rd_fetch=1)

The above figure is for DDR/DDR2 having an internal DLL. An internal DLL exists which allows it to send the data after an exact amount of read latency. If we assume there are minimal or no board/PHY input delay, if sampling the negedge (Q1, Q3 sampling), since the data gets saved into the PHY read data input FIFO, the controller sends the read data to the AXI read channel in 'read latency + 1(read fetch)' cycles. The read fetch cycle is set using the **ConControl.rd_fetch** bit-field.

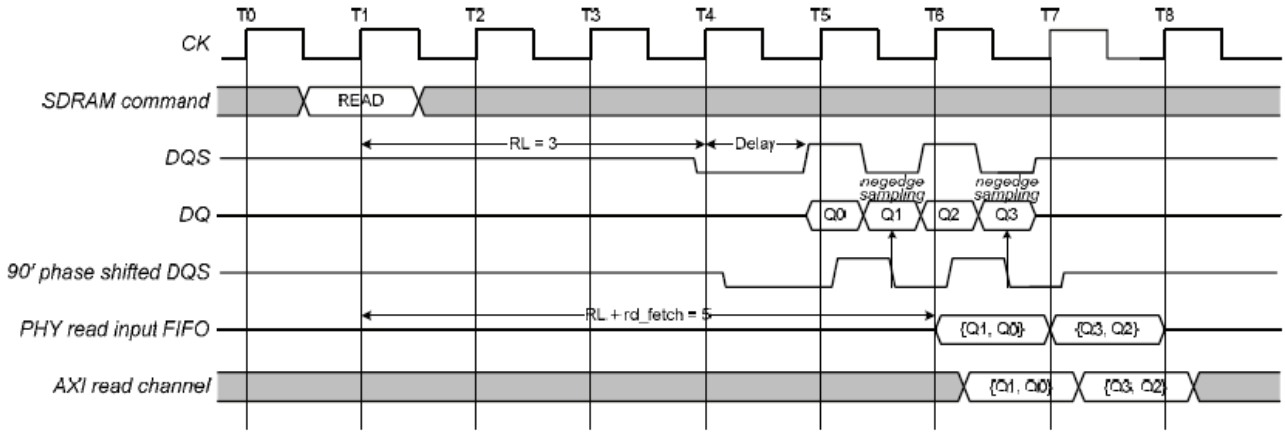


Figure 4.5 Timing diagram of read data capture (DDR/DDR2, non-zero delay, RL=3, rd_fetch=2)

Figure 4.5 is different from Figure 4.4 because a delay exists. Negedge sampling happens at T5 and T6, which is one cycle slower than T4/T5 shown in Figure 6. Therefore, the read fetch cycle should be set to two since the sampled read data is saved into the read input FIFO slower.

To calculate the DDR/DDR2 rd_fetch value:

rd_fetch (DDR/DDR2) = INT((Delay + 0.5T + 0.25T)/T) = INT(Delay/T + 0.75),

Delay: board delay + PHY input/output delay, T: clock period, INT(x): the rounded-up integer value of x

Therefore, rd_fetch must have minimum one value

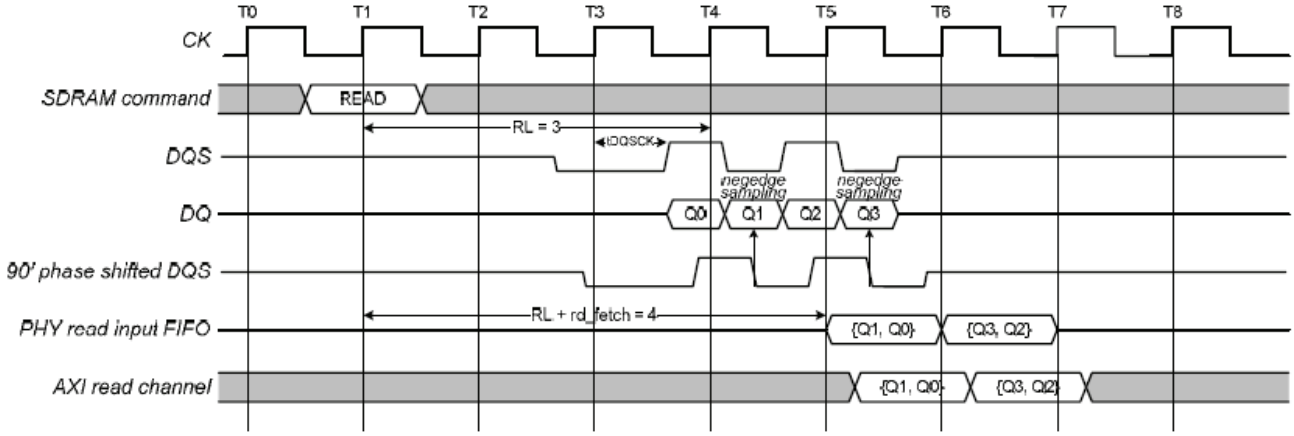


Figure 4.6 Timing diagram of read data capture (LPDDR/LPDDR2, zero delay, RL=3, rd_fetch=1)

An LPDDR/LPDDR2 does not have an internal DLL. Without an internal DLL as you may see in Figure 7, the data is sent out after tDQSCK before the read latency is over. Even if we assume zero delay, since tDQSCK becomes relatively large in high frequencies, the read fetch cycle should be set to one.

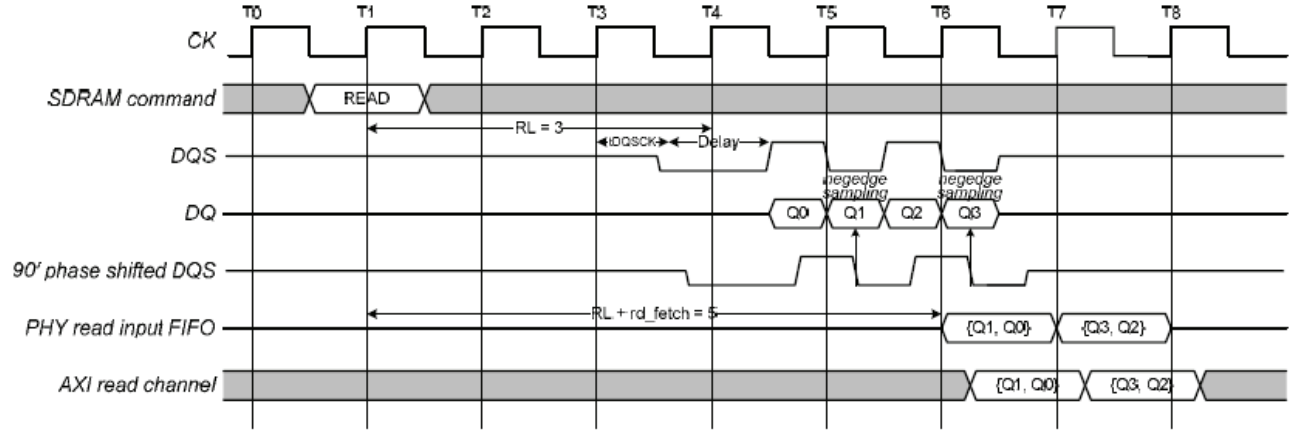


Figure 4.7 Timing diagram of read data capture (LPDDR/LPDDR2, zero delay, RL=3, rd_fetch=2)

If a delay exists such as Figure 4.8, a bigger value should be assigned to rd_fetch.

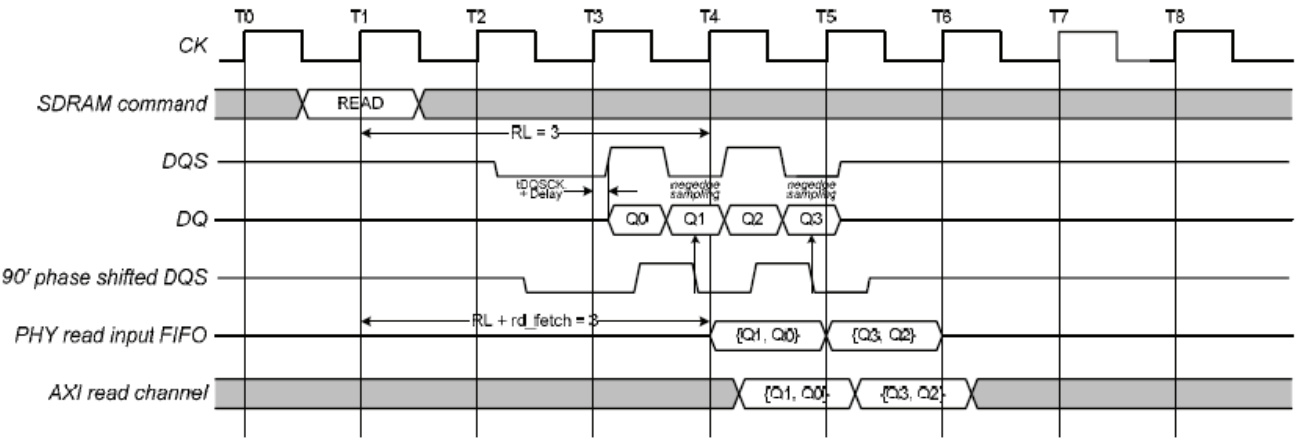


Figure 4.8 Timing diagram of read data capture (LPDDR/LPDDR2, zero delay, RL=3, rd_fetch=0)

tDQSCK + Delay is relatively small compared to the clock period during low frequencies as shown in Figure 10. In this situation, negedge sampling happens before read latency and therefore read fetch is set to zero.

To calculate the LPDDR/LPDDR2 rd_fetch value:

rd_fetch (LPDDR/LPDDR2) = INT((-1 + Delay + 0.5T + 0.25T)/T) = INT(Delay/T - 0.25),

Delay: board delay + PHY input delay, T: clock period, INT(x): the rounded-up integer value of x

Therefore, if the value of Delay/T is less than 0.25, rd_fetch is set to zero

4.6.1.5 Initialization

An initialization procedure consists of three procedures such as PHY DLL initialization, setting controller register and memory initialization. For memory initialization, please refer to JEDEC specifications and data sheets of memory devices. According to the memory types, initialization sequences are as follows.

4.6.1.5.1 LPDDR

1) To provide stable power for controller and memory device, the controller must assert and hold CKE to a logic high level. Then apply stable clock.

2) Set the **PhyControl0.ctrl_start_point** and **PhyControl0.ctrl_inc** bit-fields to correct value according to clock frequency. Set the **PhyControl0.ctrl_dll_on** bit-field to '1' to activate the PHY DLL.

3) DQS Cleaning : set the **PhyControl1.ctrl_shiftc** and **PhyControl1.ctrl_offsetc** bit-fields to the proper value according to clock frequency, board delay and memory tDQSCK parameter.

4) Set the **PhyControl0**.**ctrl_start** bit-field to '1'.

5) Set the **ConControl**. At this moment, an auto refresh counter should be off.

6) Set the **MemControl**. At this moment, all power down modes should be off.

7) Set the **MemConfig0** register. If there are two external memory chips, also set the **MemConfig1** register.

8) Set the **PrechConfig** and **PwrdnConfig** registers.

9) Set the **TimingAref, TimingRow, TimingData** and **TimingPower** registers according to memory AC parameters.

10) If QoS scheme is required, set the **QosControl0~15** and **QosConfig0~15** registers.

11) Wait for the **PhyStatus0.ctrl_locked** bit-fields to change to '1'. Check whether PHY DLL is locked.

12) PHY DLL compensates the changes of delay amount caused by PVT variation during memory operation. Therefore, it should not be off for reliable operation. It can be off except runs at low frequency. If off mode is used, set the **PhyControl0**.**ctrl_force** bit-field to the correct value according to the **PhyStatus0**.**ctrl_lock_value[9:2]** bit-field for fix delay amount. Clear the **PhyControl0.ctrl_dll_on** bit-field to turn off PHY DLL.

13) Confirm whether stable clock issues minimum 200us after power on

14) Issue a ***PALL*** command using the **DirectCmd** register.

15) Issue two ***Auto Refresh*** commands using the **DirectCmd** register.

16) Issue a ***MRS*** command using the **DirectCmd** register to program the operating parameters.

17) Issue an ***EMRS*** command using the **DirectCmd** register to program the operating parameters.

18) If there are two external memory chips, perform steps 14) ~ 17) steps for chip1 memory device.

19) Set the **ConControl** to turn on an auto refresh counter

20) If power down modes are required, set the **MemControl** register

4.6.1.5.2 LPDDR2

1) To provide stable power for controller and memory device, the controller must assert and hold CKE to a logic low level. Then apply stable clock.

2) Set the **PhyControl0.ctrl_start_point** and **PhyControl0.ctrl_inc** bit-fields to correct value according to clock

frequency. Set the **PhyControl0.ctrl_dll_on** bit-field to '1' to activate the PHY DLL.

3) DQS Cleaning : set the **PhyControl1.ctrl_shiftc** and **PhyControl1.ctrl_offsetc** bit-fields to the proper value according to clock frequency, board delay and memory tDQSCK parameter.

4) Set the **PhyControl0**.**ctrl_start** bit-field to '1'.

5) Set the **ConControl**. At this moment, an auto refresh counter should be off.

6) Set the **MemControl**. At this moment, all power down modes should be off.

7) Set the **MemConfig0** register. If there are two external memory chips, also set the **MemConfig1** register.

8) Set the **PrechConfig** and **PwrdnConfig** registers.

9) Set the **TimingAref, TimingRow, TimingData** and **TimingPower** registers according to memory AC parameters.

10) If QoS scheme is required, set the **QosControl0~15** and **QosConfig0~15** registers.

11) Wait for the **PhyStatus0.ctrl_locked** bit-fields to change to '1'. Check whether PHY DLL is locked.

12) PHY DLL compensates the changes of delay amount caused by PVT variation during memory operation. Therefore, it should not be off for reliable operation. It can be off except runs at low frequency. If off mode is used, set the **PhyControl0**.**ctrl_force** bit-field to the correct value according to the **PhyStatus0**.**ctrl_lock_value[9:2]** bit-field for fix delay amount. Clear the **PhyControl0.ctrl_dll_on** bit-field to turn off PHY DLL.

13) Set the **PhyControl1**.**fp_resync** bit-field to '1' to update DLL information.

14) Confirm that CKE has been as a logic low level at least 100ns after power on

15) Issue a ***NOP*** command using the **DirectCmd** register to assert and to hold CKE to a logic high level.

16) Wait for minimum 200us.

17) Issue a ***MRS*** command using the **DirectCmd** register to reset memory devices and program the operating parameters.

18) Wait for minimum 1us.

19) Issue a ***MRR*** command using the **DirectCmd** register to poll the DAI bit of the **MRStatus** register to know whether Device Auto-Initialization is completed or not.

20) If there are two external memory chips, perform steps 15) ~ 19) for chip1 memory device.

21) Set the **ConControl** to turn on an auto refresh counter.

22) If power down modes are required, set the **MemControl** register.

4.6.1.5.3 DDR2

1) To provide stable power for controller and memory device, the controller must assert and hold CKE to a logic low level. Then apply stable clock.

2) If on die termination is required, enable **PhyControl1.term_write_en**, **PhyControl1.term_read_en** and **PhyZQControl.ctrl_zq_mode_noterm.** Enable **PhyZQControl.ctrl_zq_start** so that the PHY automatically calibrates the I/Os to match the driving and termination impedance by referencing resistor value of an external resistor and updates the matched value during auto refresh cycles.

3) Set the **PhyControl0.ctrl_start_point** and **PhyControl0.ctrl_inc** bit-fields to correct value according to clock frequency. Set the **PhyControl0.ctrl_dll_on** bit-field to '1' to activate the PHY DLL.

4) DQS Cleaning : set the **PhyControl1.ctrl_shiftc** and **PhyControl1.ctrl_offsetc** bit-fields to the proper value according to clock frequency, board delay and memory tDQSCK parameter.

5) Set the **PhyControl0**.**ctrl_start** bit-field to '1'.

6) Set the **ConControl**. At this moment, an auto refresh counter should be off.

7) Set the **MemControl**. At this moment, all power down modes should be off.

8) Set the **MemConfig0** register. If there are two external memory chips, also set the **MemConfig1** register.

9) Set the **PrechConfig** and **PwrdnConfig** registers.

10) Set the **TimingAref, TimingRow, TimingData** and **TimingPower** registers according to memory AC parameters.

11) If QoS scheme is required, set the **QosControl0~15** and **QosConfig0~15** registers.

12) Wait for the **PhyStatus0.ctrl_locked** bit-fields to change to '1'. Check whether PHY DLL is locked.

13) PHY DLL compensates the changes of delay amount caused by PVT variation during memory operation. Therefore, it should not be off for reliable operation. It can be off except runs at low frequency. If off mode is used, set the **PhyControl0**.**ctrl_force** bit-field to the correct value according to the **PhyStatus0**.**ctrl_lock_value[9:2]** bit-field for fix delay amount. Clear the **PhyControl0.ctrl_dll_on** bit-field to turn off PHY DLL.

14) Confirm whether stable clock issues minimum 200us after power on.

15) Issue a ***NOP*** command using the **DirectCmd** register to assert and to hold CKE to a logic high level.

16) Wait for minimum 400ns.

17) Issue a ***PALL*** command using the **DirectCmd** register.

18) Issue an ***EMRS2*** command using the **DirectCmd** register to program the operating parameters.

19) Issue an ***EMRS3*** command using the **DirectCmd** register to program the operating parameters.

20) Issue an ***EMRS*** command using the **DirectCmd** register to enable the memory DLL.

21) Issue a ***MRS*** command using the **DirectCmd** register to reset the memory DLL.

22) Issue a ***PALL*** command using the **DirectCmd** register.

23) Issues two ***Auto Refresh*** commands using the **DirectCmd** register.

24) Issues a ***MRS*** command using the **DirectCmd** register to program the operating parameters without resetting the memory DLL.

25) Wait for minimum 200 clock cycles.

26) Issue an ***EMRS*** command using the **DirectCmd** register to program the operating parameters. If OCD calibration is not used, issue an ***EMRS*** command to set OCD Calibration Default. After that, issue an ***EMRS*** command to exit OCD Calibration Mode and program the operating parameters.

27) If there are two external memory chips, perform steps 15) ~ 26) procedures for chip1 memory device.

28) Set the **ConControl** to turn on an auto refresh counter.

29) If power down modes are required, set the **MemControl** register.

4.6.2 DDR2/DDR3 SDRAM Controller

DDR2/3 Controller supports DDR2 and DDR3 memory Devices. The controller provides the flexibility of interfacing up to two memory chips by making the chip select a configurable feature. However, all the memories devices should be of the same type. The Memory Devices should be either DDR2 or DDR3. The Controller does not support a mix of these.
The uniqueness of the design is in the programmability of not only the memory specific features such as timing parameters, interface bandwidth, burst length, but also AXI interface features like AXI Burst Type, Data Size etc.
Only AXI Databus Width of 64 Bits and Memory Databus Width of 32 Bits is supported. The Controller also provides status reporting through APB Interface, by performing APB Read Operations over the internal Status Registers. Refer to the Figure 2 for system level block diagram of usage model.
The Controller has AMBA-AXI Interface and AMBA-APB Interface. AMBA-AXI is used for data transfer to/from the memory. AMBA-APB is used for the configuration of the internal registers of the Controller. AMBA-APB can also be used to obtain status of the Controller.
The Controller has Command Buffer to store AXI Read/Write Commands issued from AXI Master. The depth of the

Command Buffer is 8. This buffer is common for AXI Read/Write Commands. Therefore, the Combined Acceptance Capability of the Controller is 8.
DDR2/3 Controller supports Power Management Logic by which the memory devices can be configured into 'Power Down' mode to save power consumption by the devices.
AXI 4K boundary crossing supported in case where number of column bits for external memory device is less than or equal to 10.

4.6.2.1 DDR2/3 Controller AXI Read/Write Commands Arbitration

Arbitration of AXI Read Write Commands is performed over the Timeout, Hazard, Read and Write Parameters. Timeout is the highest priority and Write is the least Priority.

4.6.2.1.1 Timeout

The AXI Read/Write command with timeout has the highest priority for Arbitration over other AXI Read/Write Commands. There are two types of timeout that can occur for the AXI Read Write Commands in the Queues.

1. QoS Timeout
   This occurs for AXI Read Commands. Timeout can be set corresponding to AXI ID (RIDs) through APB Configuration.
2. Default Timeout

This timeout occurs for both Write and Read AXI Command in the Queue. This is to ensure that every command is processed within specified time. This time can be set through APB Configuration.
Combo controller provides a Quality of Service (QoS) scheme that has provision to set up to 8 different timeout clock cycles for 8 different qos_id's. This QoS of service scheme can be enable or disabled and configured thorough APB Configuration of QoS Control Registers (Address offsets 0x300 to 0x33C).

The scheme allows following configurable parameters in QoS scheme

1. qos_en :
   This parameter enables or disables a particular QoS value (i.e the time out number of cycles for a given ARID). If this qos_en is disabled, then default timeout value will be considered for the ARID's for which QoS is not applied.
   For example:
   Let us assume that we would like to have following scheme, with default time out count as 0xFF:

Table 4.1 Example QoS Settins

QoS ID (qos_id)	QoS Time Out to be applied for matching QoS ID (qos_cnt)	QoS enable
0x0000	0x09F	1'b1
0x0010	0x08F	1'b1
0x0020	0x07F	1'b0
0x0030	0x06F	1'b0
0x0040	0x05F	1'b1
0x0050	0x04F	1'b1
0x0060	0x03F	1'b1
0x0070	0x02F	1'b1

   The above scheme shows an example where we have set specific time out count counts to QoS ID's 0000= 0x09F, 0010 = 0x08F, 0040= 0x05F, 0050= 0x04F, 0060=0x03F, 0070= 0x02F.
   In the above example, QoS for ID's 0x0020 and 0x0030, are not enabled (qos_en = 1'b0, for these ID's).
   Therefore, Setting of qos_en to 1'b1 is needed to ensure enabling of QoS for any particular ID.
   Programming of QoS counts only, is not sufficient to enable QoS over ID's.

2. qos_mask:
   This parameter is a masking value used to mask unwanted bits in the ARID. For example, as shown in the above example table 4, we are interested in ARID bits [7:4].
   The controller uses an "and" operation to extract required bits. So in order to use ARID bits [7:4] and mask the remaining bits, we need to program qos_mask bits as 0x00F0.

3. qos_id:
   The value programmed in this parameter decides to which ARID's this QoS time out (qos_cnt) value is valid. For example:

   Let us assume that
   qos_id = 0x0030;
   qos_mask = 0x0030;

   QoS time out (i.e qos_cnt discussed in the next paragraph) = 0x06F.
   All the ARID's masked values (i.e. masked with qos_mask) that match with qos_id 0x0030 will have queue time out value 0x06F.
   So, ARID values 0x0030, 0x0070, 0x00B0 and 0x00F0 after masking (i.e. ARID & qos_mask) with 0x0030 will lead to values 0x0030 that match with qos_id.

Thus AXI transactions with ARID values 0x0030, 0x0070, 0x00B0 and 0x00F0 will have QoS time out equal to 0x06F (qos_cnt)

4. qos_cnt :
   As discussed in above example, this parameter sets the number of clock cycles time out value, once a given ARID command with matching qos_id is more than this qos_cnt number of clock cycles the command gets highest priority when compared to other priorities (i.e hazard etc).

The QoS scheme flow can be explained with the following flow diagram.

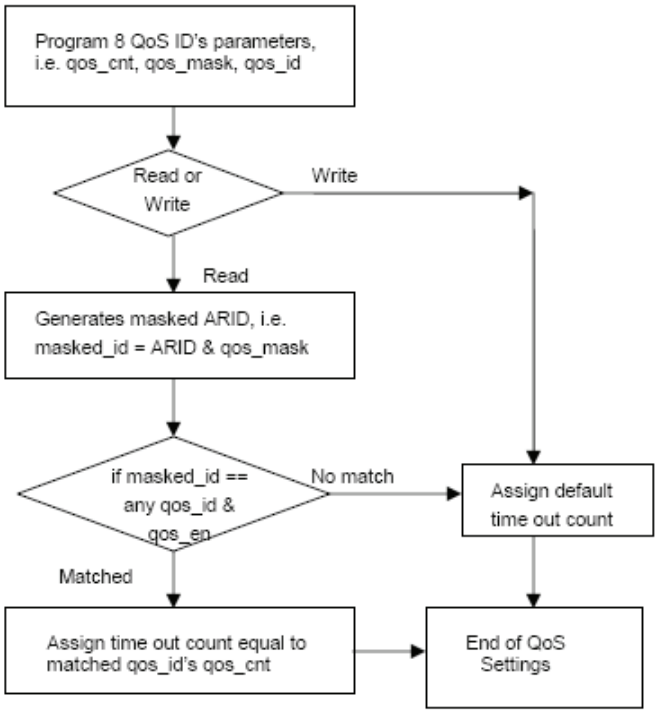


Figure 4.9 Flow Diagram of QoS Setting

4.6.2.2 Priority of Bank commands to Memory Device

Bank Commands to the Memory Device are Auto Refresh, Wakeup from Power Down, AXI Low Power Request (Self Refresh Entry and Self Refresh Exit), Dynamic Power Down, Direct Commands (through APB Interface), Memory Read/Write (AXI Read/Write Requests) and Force Precharge Command. Auto Refresh has the highest priority and Force Precharge Command has the least priority.

4.6.2.2.1 Auto Refresh

Auto Refresh Command is generated periodically, based on Auto Refresh Timer Timeout. The Auto Refresh Timer can be set through APB Configuration. This has the highest priority.

4.6.2.2.2 Wakeup From Power Down

Wakeup from Power Down Command is generated when AXI Master initiates any Read/Write Transactions, when the Memory Device is in Dynamic Power Down Mode.

4.6.2.2.3 Dynamic Power Down

Dynamic Power Down can be initiated based on the AXI Commands Queue Empty Period. There are two types of commands – Self Refresh Entry and Power Down Entry. These commands are generated based on two counters, which keep track of the time for which the AXI Command Queue are empty. These counters can be set through APB Configuration.

4.6.2.2.4 Direct Command

These are the Commands, for the Memory Device, which are initiated through APB Interface. These commands are processed only when Memory Device is in idle state.

4.6.2.2.5 Memory Read/Write

These commands correspond to AXI Read/Write Requests.

4.6.2.2.6 Force Precharge

Force Precharge command is generated when AXI bus is idle for some time period. This time period can be set through APB Configuration.

4.6.2.3 Initialization

4.6.2.3.1 DDR3

1) During providing the controller and memory device to stable power, the controller must assert and hold CKE to a logic low level. After that, apply stable clock.

2) Set the **PhyControl0**.ctrl_start_point and **PhyControl0**.ctrl_inc bit-fields to the proper value according to clock frequency. Also, turn on the PHY DLL by setting the PhyControl0.ctrl_dll_on bit-field to '1'.

3) For DQS cleaning, set the **PhyControl1**.ctrl_shiftc and **PhyControl1**.ctrl_offsetc bit-fields to the proper value according to clock frequency and memory tAC parameters.

4) Set the **PhyControl0**.ctrl_start bit-field to '1'.

5) Set the **MemConfig0** register. If there are two external memory chips, also set the **MemConfig1** register.

6) Set the **PrechConfig** and **PwrdnConfig** registers.

7) Set all the timing registers according to memory AC parameters.

8) If QoS scheme is needed, set the **QosControl0~7** and **QosConfig0~7** registers.

9) Wait until the **PhyStatus0**.ctrl_clock and **PhyStatus0**.ctrl_flock bit-fields go to '1'. It is necessary to check PHY DLL locking is finished.

10) PHY DLL can compensate the changes of delay amount caused by PVT variation during memory operation. Therefore, it should not be off for reliable operation. It can be off except runs at low frequency. If off mode is used, fix delay amount by setting the **PhyControl0**.ctrl_force bit-field to the proper value according to the **PhyStatus0**.ctrl_lock_value bit-field.

11) Set the **PhyControl5**.ctrl_zq_start to start the ZQ calibration.

12) Wait 150 PHY clk cycles.

13) Read **PhyStatus2**.ctrl_zq_end and **PhyStatus2**.ctrl_zq_error. If ctrl_zq_end is 1 then proceed further, else if ctrl_zq_error is 1, an error has occurred so stop and recheck the phy settings.

14) After the power is stable, RESET# must be low for at least 200us to begin the initialization process.

15) After RESET# transitions HIGH wait 500us.

16) Issue a NOP command using the **DirectCmd** register to assert and to hold CKE to a logic high level.

17) Issue MRS to MR3 with applicable settings.

18) Issue MRS to MR1 with applicable settings, including enabling the DLL and configuring ODT.

19) Issue MRS command to MR0 with applicable settings, including a DLL RESET command. 512 cycles of clock are required to lock the DLL.

20) Issue a ZQCL command to calibrate. Prior to normal operation, tZQinit must be satisfied.

21) DDR3 SDRAM is ready for normal operation.

22) If there are two external memory chips, perform above 16~20 procedures for chip1 memory device.

23) Set the **ConControl** and **MemControl** registers.

4.6.3 AXI ID and Corresponding Hardware

4.7 Register Descriptions

Table 4.2 LPDDR SDRAM Controller Register Map (Base Address = 0xB0301000)

Name	Offset	Type	Reset	Description
M0STAT	0x000	RO	-	Status Register
M0CMD	0x004	WO	-	Command Register
M0DCMD	0x008	WO	-	Direct COmmnad Register
M0CFG	0x00C	R/W	0x00010020	Configuration Register
M0REF	0x010	R/W	0x00000A60	Refresh Period Register
M0CAS	0x014	R/W	0x00000006	CAS Latency Register
M0DQSS	0x018	R/W	0x00000001	t_dqss Register
M0MRD	0x01C	R/W	0x00000002	t_mrd Register
M0RAS	0x020	R/W	0x00000007	t_ras Register
M0RC	0x024	R/W	0x0000000B	t_rc Register
M0RCD	0x028	R/W	0x0000001D	t_rcd Register
M0RFC	0x02C	R/W	0x000000212	t_rfc Register
M0RP	0x030	R/W	0x0000001D	t_rp Register
M0RRD	0x034	R/W	0x00000002	t_rrd Register
M0WR	0x038	R/W	0x00000003	t_wr Register
M0WTR	0x03C	R/W	0x00000002	t_wtr Register
M0XP	0x040	R/W	0x00000001	t_xp Register
M0XSR	0x044	R/W	0x0000000A	t_xsr Register
M0ESR	0x048	R/W	0x00000014	t_esr Register
M0CFG2	0x04C	R/W	-	Memory_cfg2 Register
M0CFG3	0x050	R/W	0x00000007	Memory_cfg3 Register
-	0x054-0x0FC			Reserved
M0ID0	0x100	R/W	0x00000000	AXI ID0 configuration Register
M0ID1	0x104	R/W	0x00000000	AXI ID1 configuration Register
M0ID2	0x108	R/W	0x00000000	AXI ID2 configuration Register
M0ID3	0x10C	R/W	0x00000000	AXI ID3 configuration Register
M0ID4	0x110	R/W	0x00000000	AXI ID4 configuration Register
M0ID5	0x114	R/W	0x00000000	AXI ID5 configuration Register
M0ID6	0x118	R/W	0x00000000	AXI ID6 configuration Register
M0ID7	0x11C	R/W	0x00000000	AXI ID7 configuration Register
M0ID8	0x120	R/W	0x00000000	AXI ID8 configuration Register
M0ID9	0x124	R/W	0x00000000	AXI ID9 configuration Register
M0ID10	0x128	R/W	0x00000000	AXI ID10 configuration Register
M0ID11	0x12C	R/W	0x00000000	AXI ID11 configuration Register
M0ID12	0x130	R/W	0x00000000	AXI ID12 configuration Register
M0ID13	0x134	R/W	0x00000000	AXI ID13 configuration Register
M0ID14	0x138	R/W	0x00000000	AXI ID14 configuration Register
M0ID15	0x13C	R/W	0x00000000	AXI ID15 configuration Register
-	0x140-0x1FC			Reserved
M0CH0	0x200	R/W	0x0000FF00	CHIP ID0 configuration Register
M0CH1	0x204	R/W	0x0000FF00	CHIP ID1 configuration Register
M0CH2	0x208	R/W	0x0000FF00	CHIP ID2 configuration Register
M0CH3	0x20C	R/W	0x0000FF00	CHIP ID3 configuration Register

Table 4.3 DDR2 SDRAM Controller Register Map (Base Address = 0xB0302000)

Name	Offset	Type	Reset	Description
M1STAT	0x000	RO	-	Status Register
M1CMD	0x004	WO	-	Command Register
M1DCMD	0x008	WO	-	Direct Command Register
M1CFG	0x00C	R/W	0x00010020	Configuration Register
M1REF	0x010	R/W	0x00000A2C	Refresh Period Register
M1CAS	0x014	R/W	0x0000000A	CAS Latency Register
M1WRL	0x018	RO	0x00000004	Write latency Register
M1MRD	0x01C	R/W	0x00000002	t_mrd Register
M1RAS	0x020	R/W	0x0000000E	t_ras Register
M1RC	0x024	R/W	0x00000012	t_rc Register
M1RCD	0x028	R/W	0x00000305	t_rcd Register
M1RFC	0x02C	R/W	0x000002123	t_rfc Register
M1RP	0x030	R/W	0x00000305	t_rp Register
M1RRD	0x034	R/W	0x00000004	t_rrd Register
M1WR	0x038	R/W	0x00000005	t_wr Register
M1WTR	0x03C	R/W	0x00000004	t_wtr Register
M1XP	0x040	R/W	0x00000002	t_xp Register
M1XSR	0x044	R/W	0x00000027	t_xsr Register
M1ESR	0x048	R/W	0x00000014	t_esr Register
M1CFG2	0x04C	R/W	-	Memory_cfg2 Register
M1CFG3	0x050	R/W	0x00000007	Memory_cfg3 Register
M1_FAW	0x054	R/W	0x00000011	t_faw Register
	0x058-0x0FC			Reserved
M1ID0	0x100	R/W	0x00000000	AXI ID0 configuration Register
M1ID1	0x104	R/W	0x00000000	AXI ID1 configuration Register
M1ID2	0x108	R/W	0x00000000	AXI ID2 configuration Register
M1ID3	0x10C	R/W	0x00000000	AXI ID3 configuration Register
M1ID4	0x110	R/W	0x00000000	AXI ID4 configuration Register
M1ID5	0x114	R/W	0x00000000	AXI ID5 configuration Register
M1ID6	0x118	R/W	0x00000000	AXI ID6 configuration Register
M1ID7	0x11C	R/W	0x00000000	AXI ID7 configuration Register
M1ID8	0x120	R/W	0x00000000	AXI ID8 configuration Register
M1ID9	0x124	R/W	0x00000000	AXI ID9 configuration Register
M1D10	0x128	R/W	0x00000000	AXI ID10 configuration Register
M1D11	0x12C	R/W	0x00000000	AXI ID11 configuration Register
M1ID12	0x130	R/W	0x00000000	AXI ID12 configuration Register
M1ID13	0x134	R/W	0x00000000	AXI ID13 configuration Register
M1ID14	0x138	R/W	0x00000000	AXI ID14 configuration Register
M1ID15	0x13C	R/W	0x00000000	AXI ID15 configuration Register
-	0x140-0x1FC			Reserved
M1CH0	0x200	R/W	0x0000FF00	CHIP ID0 configuration Register
M1CH1	0x204	R/W	0x0000FF00	CHIP ID1 configuration Register
M1CH2	0x208	R/W	0x0000FF00	CHIP ID2 configuration Register
M1CH3	0x20C	R/W	0x0000FF00	CHIP ID3 configuration Register

Table 4.4 LPDDR/LPDDR2/DDR2 SDRAM Controller Register Map (Base Address = 0xB0305000)

Name	Offset	Type	Reset	Description
CONCONTROL	0x000	R/W	0x3FFF_1310	Controller Control Register
MEMCONTROL	0x004	R/W	0x0020_2100	Memory Control Register
MEMCONFIG0	0x008	R/W	0x20F8_0312	Memory Chip0 Configuration Register
MEMCONFIG1	0x00C	R/W	0x28F8_0312	Memory Chip1 Configuration Register
DIRECTCMD	0x010	R/W	0x0000_0000	Memory Direct Command Register
PRECHCONFIG	0x014	R/W	0xFF00_0000	Precharge Policy Configuration Register
PHYCONTROL0	0x018	R/W	0x0010_1000	PHY Control0 Register
PHYCONTROL1	0x01C	R/W	0x0000_0000	PHY Control1 Register
PHYCONTROL2	0x020	R/W	0x0000_0000	PHY Control2 Register
PHYCONTROL3	0x024	R/W	0x0000_0000	PHY Control3 Register
PWRDNCONFIG	0x028	R/W	0xFFFF_00FF	Dynamic Power Down Configuration Register
TIMINGAREF	0x030	R/W	0x0000_040E	AC Timing Register for SDRAM Auto refresh
TIMINGROW	0x034	R/W	0x0F23_3286	AC Timing Register for SDRAM Row
TIMINGDATA	0x038	R/W	0x1213_0204	AC Timing Register for SDRAM Data
TIMINGPOWER	0x03C	R/W	0x0E1B_0422	AC Timing Register for Power Mode of SDRAM
PHYSTATUS	0x040	R	0x0000_0000	PHY Status Register
PHYZQCONTROL	0x044	R/W	0xE385_5731	PHY ZQ I/O Control Register
CHIP0STATUS	0x048	R	0x0000_0000	Memory Chip0 Status Register
CHIP1STATUS	0x04C	R	0x0000_0000	Memory Chip1 Status Register
AREFSTATUS	0x050	R	0x0000_FFFF	Counter Status Register for Auto Refresh
MRSTATUS	0x054	R	0x0000_0000	Memory Mode Registers Status Register
PHYTEST0	0x058	R/W	0x0000_0000	PHY Test Register 0
PHYTEST1	0x05C	R	0x0000_0000	PHY Test Register 1
QOSCONTROL0	0x060	R/W	0x0000_0000	Quality of Service Control Register 0
QOSCONFIG0	0x064	R/W	0x0000_0000	Quality of Service Configuration Register 0
QOSCONTROL1	0x068	R/W	0x0000_0000	Quality of Service Control Register 1
QOSCONFIG1	0x06C	R/W	0x0000_0000	Quality of Service Configuration Register 1
QOSCONTROL2	0x070	R/W	0x0000_0000	Quality of Service Control Register 2
QOSCONFIG2	0x074	R/W	0x0000_0000	Quality of Service Configuration Register 2
QOSCONTROL3	0x078	R/W	0x0000_0000	Quality of Service Control Register 3
QOSCONFIG3	0x07C	R/W	0x0000_0000	Quality of Service Configuration Register 3
QOSCONTROL4	0x080	R/W	0x0000_0000	Quality of Service Control Register 4
QOSCONFIG4	0x084	R/W	0x0000_0000	Quality of Service Configuration Register 4
QOSCONTROL5	0x088	R/W	0x0000_0000	Quality of Service Control Register 5
QOSCONFIG5	0x08C	R/W	0x0000_0000	Quality of Service Configuration Register 5
QOSCONTROL6	0x090	R/W	0x0000_0000	Quality of Service Control Register 6
QOSCONFIG6	0x094	R/W	0x0000_0000	Quality of Service Configuration Register 6
QOSCONTROL7	0x098	R/W	0x0000_0000	Quality of Service Control Register 7
QOSCONFIG7	0x09C	R/W	0x0000_0000	Quality of Service Configuration Register 7
QOSCONTROL8	0x0A0	R/W	0x0000_0000	Quality of Service Control Register 8
QOSCONFIG8	0x0A4	R/W	0x0000_0000	Quality of Service Configuration Register 8
QOSCONTROL9	0x0A8	R/W	0x0000_0000	Quality of Service Control Register 9
QOSCONFIG9	0x0AC	R/W	0x0000_0000	Quality of Service Configuration Register 9
QOSCONTROL10	0x0B0	R/W	0x0000_0000	Quality of Service Control Register 10
QOSCONFIG10	0x0B4	R/W	0x0000_0000	Quality of Service Configuration Register 10
QOSCONTROL11	0x0B8	R/W	0x0000_0000	Quality of Service Control Register 11
QOSCONFIG11	0x0BC	R/W	0x0000_0000	Quality of Service Configuration Register 11
QOSCONTROL12	0x0C0	R/W	0x0000_0000	Quality of Service Control Register 12
QOSCONFIG12	0x0C4	R/W	0x0000_0000	Quality of Service Configuration Register 12
QOSCONTROL13	0x0C8	R/W	0x0000_0000	Quality of Service Control Register 13
QOSCONFIG13	0x0CC	R/W	0x0000_0000	Quality of Service Configuration Register 13
QOSCONTROL14	0x0D0	R/W	0x0000_0000	Quality of Service Control Register 14
QOSCONFIG14	0x0D4	R/W	0x0000_0000	Quality of Service Configuration Register 14
QOSCONTROL15	0x0D8	R/W	0x0000_0000	Quality of Service Control Register 15
QOSCONFIG15	0x0DC	R/W	0x0000_0000	Quality of Service Configuration Register 15

Table 4.5 DDR2/DDR3 SDRAM Controller Register Map (Base Address = 0xB030C000)

Name	Offset	Type	Reset	Description
CONCONTROL	0x000	R/W	0x0FFF_1010	Controller control register
MEMCONTROL	0x004	R/W	0x0000_010A	Memory control register
MEMCONFIG0	0x008	R/W	0x20F8_0312	Memory chip0 configuration register
MEMCONFIG1	0x00C	R/W	0x28F8_0312	Memory chip1 configuration register
DIRECTCMD	0x010	R/W	0x0000_0000	Memory direct command register
PRECHCONFIG	0x014	R/W	0xFF00_0000	Precharge policy configuration register
MEMPWD	0x018	R/W	0x0000_0000	Memory power down register
PWRDNCONFIG	0x01C	R/W	0xFFFF_00FF	Dynamic power down configuration register
AREF	0x100	R/W	0x0000_040E	AC timing register for auto refresh of SDRAM
TRFC	0x104	R/W	0x0000_000F	Auto refresh to other command
TRRD	0x108	R/W	0x0000_0002	Bank to Bank active delay
TRP	0x10C	R/W	0x0000_0003	Row precharge delay
TRCD	0x110	R/W	0x0000_0003	Activated to read or write command
TRC	0x114	R/W	0x0000_000A	Same bank activate to activate delay
TRAS	0x118	R/W	0x0000_0006	Row open period
TWTR	0x11C	R/W	0x0000_0001	Write to read delay
TWR	0x120	R/W	0x0000_0002	Write recovery time
TRTP	0x124	R/W	0x0000_0001	Read to precharge delay
CL	0x128	R/W	0x0000_0003	CAS latency
WL	0x12C	R/W	0x0000_0002	Write latency
RL	0x130	R/W	0x0000_0004	Read latency
FAW	0x134	R/W	0x0000_001E	Quadrapule row activation command delay
TXSR	0x138		0x0000_001B	Self refresh exit delay
TXP	0x13C	R/W	0x0000_0004	Power down exit delay
TCKE	0x140	R/W	0x0000_0002	Minimum power down period
TMRD	0x144	R/W	0x0000_0002	Mode register set delay
TZQINIT	0x148	R/W	0x0000_0201	ZQ calibration delay after reset or power up
TZQOPER	0x14C	R/W	0x0000_0101	Full ZQ calibration delay
TZQS	0x150	R/W	0x0000_0041	Short ZQ calibration delay
AXISTATUS	0x200	R	0x0000_00031	Axi transaction status register
AREFSTATUS	0x204	R	0x0000_FFFF 1	Counter status register for auto refresh
CHIP0STATUS	0x208	R	0x0000_000	Memory chip0 status register
CHIP1STATUS	0x20C	R	0x0000_000	Memory chip1 status register
MRSTATUS	0x208	R	0x0000_000	Memory mode registers status register
QOSCONTROL0	0x300	R/W	0x00FF_0000	Quality of service control register 0
QOSCONFIG0	0x304	R/W	0x0000_0000	Quality of service configuration register 0
QOSCONTROL1	0x308	R/W	0x00FF_0000	Quality of service control register 1
QOSCONFIG1	0x30C	R/W	0x0000_0000	Quality of service configuration register 1
QOSCONTROL2	0x310	R/W	0x00FF_0000	Quality of service control register 2
QOSCONFIG2	0x314	R/W	0x0000_0000	Quality of service configuration register 2
QOSCONTROL3	0x318	R/W	0x00FF_0000	Quality of service control register 3
QOSCONFIG3	0x31C	R/W	0x0000_0000	Quality of service configuration register 3
QOSCONTROL4	0x320	R/W	0x00FF_0000	Quality of service control register 4
QOSCONFIG4	0x324	R/W	0x0000_0000	Quality of service configuration register 4
QOSCONTROL5	0x328	R/W	0x00FF_0000	Quality of service control register 5
QOSCONFIG5	0x32C	R/W	0x0000_0000	Quality of service configuration register 5
QOSCONTROL6	0x330	R/W	0x00FF_0000	Quality of service control register 6
QOSCONFIG6	0x334	R/W	0x0000_0000	Quality of service configuration register 6
QOSCONTROL7	0x338	R/W	0x00FF_0000	Quality of service control register 7
QOSCONFIG7	0x340	R/W	0x0000_0000	Quality of service configuration register 7
PHYCONTROL0	0x400	R/W	0x0000_0000	PHY control0 register
PHYCONTROL1	0x404	R/W	0x0000_0000	PHY control1 register
PHYCONTROL2	0x408	R/W	0x0000_0000	PHY control2 register
PHYCONTROL3	0x40C	R/W	0x0000_0000	PHY control3 register
PHYCONTROL4	0x410	R/W	0x0000_0000	PHY control4 register
PHYCONTROL5	0x414	R/W	0x0000_0000	PHY control5 register
PHYSTATUS0	0x418	R	0x0000_000	PHY status register 0
PHYSTATUS1	0x41C	R	0x0000_000	PHY status register 1

Name	Offset	Type	Reset	Description
PHYSTATUS2	0x420	R	0x0000_000	PHY status register 2
PHYTEST0	0x424	R/W	0x0000_0000	PHY feedback test signals
PHYTEST1	0x428	R	0x0000_000	PHY feedback status signals

Table 4.6 DDR I/F Configuration Register Map (Base Address = 0xB0303000)

Name	Offset	Type	Reset	Description
M0CFG0	0x00	R/W	0x80400000	LPDDR SDRAM Controller Configuration Register 0
M0CFG1	0x04	R/ RW	0x00000018	LPDDR SDRAM Controller Configuration Register 1
-	0x08-0x0C			Reserved
M1CFG0	0x10	R/W	0x80000000	DDR2 SDRAM Controller Configuration Register 0
M1CFG1	0x14	R/W	0x00000000	DDR2 SDRAM Controller Configuration Register 1
-	0x18-0x1C			Reserved
COMMON	0x20	R/W	0x00010103	Common Control Register
PHYCTRL	0x24	R/W	0x00000000	DDR PHY Control Register
-	0x28-0x2C			Reserved
M2CFG0	0x30		0x00000000	LPDDR/KPDDR2/DDR2 SDRAM Controller Configuration Register 0
M2CFG1	0x34		0x00000000	LPDDR/KPDDR2/DDR2 SDRAM Controller Configuration Register 1
-	0x38-0x40			Reserved
M3CFG0	0x44		0x00000000	DDR2/DDR3 SDRAM Controller Configuration Register

Table 4.7 DDR PHY Register Map (Base Address = 0xB0304400)

Name	Offset	Type	Reset	Description
REG0	0x400	R/W	0x00000000	PHY Mode Control Register
REG1	0x404	R/W	0x00000000	DLL Control & Status Register
REG2	0x408	R/W	0x00000000	DLL Phase Detector configuration Register
REG3	0x40C	R/W	0x00000000	Gate Control Register
REG4	0x410	R/W	0x00000000	Read Data Slice 0 Control Register
REG5	0x414	R/W	0x00000000	Read Data Slice 1 Control Register
REG6	0x418	RO	0x00000000	Read Data Slice 2 Control Register
REG7	0x41C	R/W	0x00000000	Read Data Slice 3 Control Register
REG8	0x420	R/W	0x00000000	CLK Delay Register
REG9	0x424	R/W	0x00000000	DLL Force Lock Value Register
REG10	0x428	R/W	0x00060000	ZQ Calibration Control Register
REG11	0x42C	RO	0x00000000	ZQ Calibration Status Register
REG12	0x430	R/W	0x00000000	Read Delay Register

4.7.1 LPDDR SDRAM Controller Registers

Status Register **0xB0301000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			BANK1	XMON		BANK0	CHIPS		TYPE			WIDTH		STAT	

Field	Name	RW	Reset	Description
12	BANK1	R	-	This returns part of the definition of the number of banks on each chip.
11-10	XMON	R	-	Returns the number of exclusive access monitor resources implemented in the memory controller: 2'b00 = 0 monitors 2'b01 = 1 monitor 2'b10 = 2 monitors 2'b11 = 4 monitors.
9	BANK0	R	-	This returns part of the definition of the number of banks on each chip.
8-7	CHIPS	R	-	Returns the number of different chip selects that the memory controller supports: 2'b00 = 1 chip 2'b01 = 2 chips 2'b10 = Reserved 2'b11 = Reserved.
6-4	TYPE	R	-	Returns the SDRAM that the memory controller supports: 3'b000 = Reserved 3'b001 = Reserved 3'b011 = LPDDR SDRAM 3'b010 = Reserved 3'b1xx = Reserved. If LPDDR SDRAM is supported, the cas_half_cycle bit at address offset 0x14 is ignored.
3-2	WIDTH	R	-	Returns the width of the external memory: 2'b00 = 16-bit 2'b01 = 32-bit 2'b10 = Reserved 2'b11 = Reserved.
1-0	STAT	R	-	Returns the state of the memory controller: 2'b00 = Config 2'b01 = Ready 2'b10 = Paused 2'b11 = Low_power.

Table 4.8 Memory Banks Chip Configuration

Memorybanks1 and Memorybanks0	Banks per memory chip
0	4
1	2
2	Reserved
3	Reserved

Command Register **0xB0301004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													CMD		

The write-only register enables the memory controller to be traversed. The command register controls the programmer's view FSM. By writing to this register, the FSM can be traversed.

Field	Name	RW	Reset	Description
2-0	CMD	W	-	Changes the state of the memory controller: 3'b000 = Go 3'b001 = Sleep 3'b010 = Wakeup 3'b011 = Pause 3'b100 = Configure 3'b111 = Active_Pause.

Note:

Active_Pause puts the memory controller into the Paused state without draining the arbiter queue. This enables you to enter low-power mode to change configuration settings such as memory frequency or timing register values without requiring coordination between masters in a multi-master system.

If the memory controller is put into low-power mode after using the Active_Pause command, you must not remove power from the memory controller because this results in data loss and violation of the AXI protocol. The memory controller does not issue refreshes while in the Config state. You must use low-power mode to make register updates because this ensures that the memory is put into self-refresh rather than entering the Config state when the memory contains valid data.

Direct Command Register 0xB0301008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									ECMD	CHNB		MCMD		BNKA	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
		ADR													

The write-only direct_cmd Register passes commands to the external memory. The configuration of the direct_cmd Register enables you to write to any type of Mode register supported by the external memory device, and also to generate NOP, Prechargeall, and Auto-refresh commands.

The direct_cmd Register therefore enables any initialization sequence that an external memory device might require. The only timing information associated with the direct_cmd Register are the command delays defined in the timing registers. Therefore, if an initialization sequence requires additional delays between commands, they must be timed by the master driving the initialization sequence.

Field	Name	RW	Reset	Description
22	ECMD	W	-	Extended memory command, see note after the table
21-20	CHNB	W	-	Bits mapped to external memory chip address bits
19-18	MCMD	W	-	Determines the command required, see note after the table
17-16	BNKA	W	-	Bits mapped to external memory bank address bits when command is Mode_reg access
15-14	-			Reserved
13-0	ADR	W	-	Bits mapped to external memory address bits [13:0] when command is Mode_reg access

Note:

Memory command encoding uses the ECMD bit concatenated to MCMD, therefore providing 3 bits as follows:

3'b000 = Prechargeall
3'b001 = Autorefresh
3'b010 = Modereg or Extended modereg access
3'b011 = NOP, for SDRAM only, Warning: If you have the NVM plug-in licensed, do not use the NOP command with NVM chip selects.
3'b100 = DPD

All other combinations are illegal and might cause undefined behavior.

A NOP command asserts all chip selects that are set as active_chips when the chip_nmbr is set to 0.

If chip_nmbr is set to 1 only cs_n[1] is asserted.
If chip_nmbr is set to 2 only cs_n[2] is asserted.
If chip_nmbr is set to 3 only cs_n[3] is asserted.

Configuration Register **0xB030100C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SR_EN	FP_TIME							FP_EN	ACTCH		QOS_MBITS			MBURST	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MBURST	STOP_MCLK	AUTO_PD	PD_PRD						AP_BIT	ROW_BITS			COL_BITS		

Field	Name	RW	Reset	Description
31	SR_EN[3]	R/W	0	Auto self-entry enable. Only if configured else read is undefined write as zero.
30-24	FP_TIME	R/W	0	Force precharge timeout count. Only valid if configured, else read is undefined write as zero.
23	FP_EN	R/W	0	Force precharge enable. Only valid if configured, else read undefined write as zero.
22-21	ACTCH	R/W	0	Enables the refresh command generation for the number of memory chips. It is only possible to generate commands up to and including the number of chips in the configuration that the status Register defines: 2'b00 = 1 chip 2'b01 = 2 chips 2'b10 = Reserved 2'b11 = Reserved.
20-18	QOS_MBITS	R/W	0	Encodes the four bits of the 8-bit AXI ARID that select one of the 16 QOS values: 3'b000 = ARID[3:0] 3'b001 = ARID[4:1] 3'b010 = ARID[5:2] 3'b011 = ARID[6:3] 3'b100 = ARID[7:4] 3'b101 = ARID[8:5] 3'b110 = ARID[9:6] 3'b111 = ARID[10:7]
17-15	MBURST	R/W	0x2	Encodes the number of data accesses that are performed to the SDRAM for each Read and Write command: 3'b000 = Burst 1 3'b001 = Burst 2 3'b010 = Burst 4 3'b011 = Burst 8 3'b100 = Burst 16.
14	STOP_MCLK[4]	R/W	0	When enabled, the memory clock is dynamically stopped when not performing an access to the LPDDR SDRAM.
13	AUTO_PD	R/W	0	When this is set, the memory interface automatically places the LPDDR SDRAM into power-down state by deasserting cke when the command FIFO has been empty for PD_PRD memory clock cycles.
12-7	PD_PRD	R/W	0	Number of memory clock cycles for AUTO_PD of the LPDDR SDRAM. The PD_PRD programmed must be greater than the programmed cas latency
6	AP_BIT	R/W	0	Encodes the position of the auto-precharge bit in the memory address: 1'b0 = address bit 10 1'b1 = address bit 8.
5-3	ROW_BITS	R/W	0x4	Encodes the number of bits of the AXI address that comprise the row address: 3'b000 = 11 bits 3'b001 = 12 bits

[3] Do not enable the SR_EN and STOP_MCLK bits at the same time because it does not support this.
[4] To comply to the JEDEC standard, the auto_power_down and stop_mem_clock bits must not be enabled at the same time.

				3'b010 = 13 bits 3'b011 = 14 bits 3'b100 = 15 bits 3'b101 = 16 bits. The combination of row size, column size, BRC/RBC, and memory width must ensure that neither the MSB of the row address nor the MSB of the bank address exceed address range [27:0].
2-0	COL_BITS	R/W	0	Encodes the number of bits of the AXI address that comprise the column address: 3'b000 = 8 bits 3'b001 = 9 bits 3'b010 = 10 bits 3'b011 = 11 bits 3'b100 = 12 bits.

Refresh Period Register 0xB0301010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	REF_PRD														

Field	Name	RW	Reset	Description
14-0	REF_PRD	R/W	0x0A60	Memory refresh period in memory clock cycles

CAS Latency Register 0xB0301014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												CAS_LAT			CAS_HALF

Field	Name	RW	Reset	Description
3-1	CAS_LAT	R/W	0x3	CAS latency in memory clock cycles.
0	CAS_HALF	R/W	0	Encodes whether the CAS latency is half a memory clock cycle more than the value given in bits [3:1]: 1'b0 = Zero cycles offset to value in [3:1]. b0 is forced to 0 in LPDDR SDRAM mode. 1'b1 = Half cycle offset to value in [3:1].

t_dqss Register 0xB0301018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														t_dqss	

The read/write t_dqss Register writes to DQS in memory clock cycles.

Field	Name	RW	Reset	Description
1-0	t_dqss	R/W	0x1	Write to DQS in memory clock cycles

t_mrd Register 0xB030101C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									t_mrd						

The read/write t_mrd Register sets the mode register command time in memory clock cycles.

Field	Name	RW	Reset	Description
6-0	t_mrd	R/W	0x2	Sets mode register command time in memory clock cycles

t_ras Register 0xB0301020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												t_ras			

The read/write t_ras Register sets the RAS to precharge delay in memory clock cycles.

Field	Name	RW	Reset	Description
3-0	t_ras	R/W	0x7	Sets RAS to precharge delay in memory clock cycles

t_rc Register 0xB0301024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												t_rc			

The read/write t_rc Register sets the Active bank x to Active bank x delay in memory clock cycles.

Field	Name	RW	Reset	Description
3-0	t_rc	R/W	0xB	Sets Active bank x to Active bank x delay in memory clock cycles

t_rcd Register 0xB0301028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										SCH_RCD			t_rcd		

The read/write t_rcd Register sets the RAS to CAS minimum delay in memory clock cycles.

Field	Name	RW	Reset	Description
5-3	SCH_RCD	R/W	0x3	Sets the RAS to CAS minimum delay in aclk cycles-3.
2-0	t_rcd	R/W	0x5	Sets the RAS to CAS minimum delay in memory clock cycles.

t_rfc Register 0xB030102C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						SCH_RFC					t_rfc				

The read/write t_rfc Register sets the auto-refresh command time in memory clock cycles.

Field	Name	RW	Reset	Description
9-5	SCH_RFC	R/W	0x10	Sets the autorefresh command time in aclk cycles-3.
4-0	t_rfc	R/W	0x12	Sets the auto-refresh command time in memory clock cycles.

t_rp Register 0xB0301030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										SCH_RP			t_rp		

The read/write t_rp Register sets the precharge to RAS delay in memory clock cycles.

Field	Name	RW	Reset	Description
5-3	SCH_RP	R/W	0x3	Sets the precharge to RAS delay in aclk cycles -3.
2-0	t_rp	R/W	0x5	Sets the precharge to RAS delay in memory clock cycles.

t_rrd Register 0xB0301034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												t_rrd			

The read/write t_rrd Register sets the Active bank x to Active bank y delay in memory clock cycles.

Field	Name	RW	Reset	Description
3-0	t_rrd	R/W	0x2	Sets Active bank x to Active bank y delay in memory clock cycles

t_wr Register 0xB0301038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														t_wr	

The read/write t_wr Register sets the write to precharge delay in memory clock cycles.

Field	Name	RW	Reset	Description
2-0	t_wr	R/W	0x3	Sets the write to precharge delay in memory clock cycles

t_wtr Register 0xB030103C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													t_wtr		

The read/write t_wtr Register sets the write to read delay in memory clock cycles.

Field	Name	RW	Reset	Description
2-0	t_wtr	R/W	0x2	Sets the write to precharge delay in memory clock cycles

t_xp Register 0xB0301040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								t_xp							

The read/write t_wtr Register sets the write to read delay in memory clock cycles.

Field	Name	RW	Reset	Description
7-0	t_xp	R/W	0x1	Sets the exit power-down command time in memory clock cycles

t_xsr Register 0xB0301044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								t_xsr							

The read/write t_xsr Register sets exit self-refresh command time in memory clock cycles.

Field	Name	RW	Reset	Description
7-0	t_xsr	R/W	0xA	Sets the exit self-refresh command time in memory clock cycles

t_esr Register 0xB0301048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								t_esr							

The read/write t_xsr Register sets self-refresh command time in memory clock cycles.

Field	Name	RW	Reset	Description
7-0	t_esr	R/W	0x14	Sets the self-refresh command time in memory clock cycles

Memory_cfg2 Register **0xB030104C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					RD_DLY		TYPE			WIDTH		CKE_INIT	DQM_INIT	A_GT_M_SYNC	SYNC

The read/write memory_cfg2 Register determines the operating state of the memory controller and the memory. At reset the register value is set by the tie-off pins with the same names as the register names. After power-up you can change the register value through the APB interface.

Field	Name	RW	Reset	Description
10-9	RD_DLY	R/W	-	Sets the latency in clock cycles of the pad interface.
8-6	TYPE	R/W	-	Sets the memory type: 0 = Reserved 1 = Reserved 2 = Reserved 3 = LPDDR (Mobile DDR). Note: It is only legal to program the memory type between SDR and (LP)DDR for a memory controller configuration that supports it.
5-4	WIDTH	R/W	-	Sets the width of the external memory: 2'b00 = 16-bit 2'b01 = 32-bit 2'b10 = Reserved 2'b11 = Reserved. Note: Only a memory width that is legal for the memory controller can be programmed.
3	CKE_INIT	R/W	-	Sets the level for the cke outputs after reset.
2	DQM_INIT	R/W	-	Sets the level for the dqm outputs after reset.
1	A_GT_M_SYNC	R/W	-	Requires to be set HIGH when running the aclk and mclk synchronously but with aclk running faster than mclk. **** Should be "0" in this chip.***
0	SYNC	R/W	-	Set high when aclk and mclk are synchronous. **** Should be "1" in this chip.***

Memory_cfg3 Register **0xB0301050**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			PRESCALE										MAX_OUT_REFS		

The read/write memory_cfg3 Register determines the operating state of the memory controller and the memory. At reset the register value is set by the tie-off pins with the same names as the register names. After power-up you can change the register value through the APB interface.

Field	Name	RW	Reset	Description
12-3	PRESCALE	R/W	0x0	Prescalar counter value.
2-0	MAX_OUT_REFS	R/W	0x7	Maximum number of outstanding refresh commands.

AXI ID n Configuration Register

0xB0301100+4*n

n=0~15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						QOS_MAX								QOS_MIN	QOS_EN

The read/write IDnCFG Registers are 16 registers that set the QoS and span address locations 0x100-0x200.

Field	Name	RW	Reset	Description
9-2	QOS_MAX	R/W	0	Sets a maximum QoS.
1	QOS_MIN	R/W	0	Sets a minimum QoS.
0	QOS_EN	R/W	0	Enables a QoS value to be applied to memory reads from address ID n.

Chip n Configuration Register

0xB0301200+4*n

n=0~3

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
															BRC_N_RBC
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_MATCH								ADDR_MASK							

The read/write CHIPnCFG Registers are registers that set up the external memory device configuration. The number of external chips supported, and therefore the number of these registers, depends on your configuration. They span address locations 0x200-0x300. There is one register per memory device. The registers configure the base address and address decoding method.

Field	Name	RW	Reset	Description
16	BRC_N_RBC	R/W	0	Selects the memory organization as decoded from the AXI address: 1'b0 = Row, bank, column organization 1'b1 = Bank, row, column organization.
15-8	ADDR_MATCH	R/W	0xFF	Comparison value for AXI address bits [31:24] to determine the chip that is selected.
7-0	ADDR_MASK	R/W	0	The mask for AXI address bits [31:24] to determine the chip that is selected: 1 = corresponding address bit is to be used for comparison.

4.7.2 DDR2 SDRAM Controller Registers

Status Register **0xB0302000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		BANK		XMON		BANK0	CHIPS		TYPE			WIDTH		STAT	

Field	Name	RW	Reset	Description
13-12	BANK	R	-	Returns the maximum number of banks per memory chip. This is a rendered option, and is static for a given configuration: b00 = 4 banks b01 = 2 banks, not supported b10 = 1 banks, not supported b11 = 8 banks.
11-10	XMON	R		Returns the number of exclusive access monitor resources implemented in the memory controller: 2'b00 = 0 monitors 2'b01 = 1 monitor 2'b10 = 2 monitors 2'b11 = 4 monitors.
9	-			Reserved
8-7	CHIPS	R		Returns the number of different chip selects that the memory controller supports: 2'b00 = 1 chip 2'b01 = 2 chips 2'b10 = Reserved 2'b11 = Reserved.
6-4	TYPE	R		Returns the type of SDRAM that the DDR2 DMC supports: b000-b100 = Reserved b101 = DDR2 SDRAM b110-b111 = Reserved.
3-2	WIDTH	R		Returns the width of the external memory: 2'b00 = 16-bit 2'b01 = 32-bit 2'b10 = Reserved 2'b11 = Reserved.
1-0	STAT	R		Returns the state of the memory controller: 2'b00 = Config 2'b01 = Ready 2'b10 = Paused 2'b11 = Low_power.

Command Register 0xB0302004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													CMD		

Writing to the register enables the programmer's view FSM to be traversed.

Field	Name	RW	Reset	Description
2-0	CMD	W	-	Changes the state of the memory controller: 3'b000 = Go 3'b001 = Sleep 3'b010 = Wakeup 3'b011 = Pause 3'b100 = Configure 3'b111 = Active_Pause.

Note:

Active_Pause command puts the DDR2 SDRAM Controller into the Paused state without draining the arbiter queue. This enables you to enter Low-power state to change configuration settings such as memory frequency or timing register values without requiring co-ordination between masters in a multi-master system.

If the DDR2 SDRAM Controller is put into Low-power state after using the Active_Pause command, you must not remove power from it because this results in data loss and violation of the AXI protocol.

The DDR2 DMC does not issue refreshes when in the Config state. It is recommended therefore that you use low-power mode to make register updates because this ensures that the memory is put into self-refresh rather than entering the Config state when the memory contains valid data.

If you entered the Paused state using the Active_Pause command, you must not attempt to move to the Config state by using the Configure command.

Direct Command Register 0xB0302008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
										CHNB		MCMD		BNKA	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		ADR													

The write-only direct_cmd Register passes commands to the external memory. The configuration of the direct_cmd Register enables you to write to any type of Mode register supported by the external memory device, and also to generate NOP, Prechargeall, and Auto-refresh commands.

The direct_cmd Register therefore enables any initialization sequence that an external memory device might require. The only timing information associated with the direct_cmd Register are the command delays defined in the timing registers. Therefore, if an initialization sequence requires additional delays between commands, they must be timed by the master driving the initialization sequence.

Field	Name	RW	Reset	Description
21-20	CHNB	W	-	Bits mapped to external memory chip address bits
19-18	MCMD	W	-	Determines the command required: b00 = Prechargeall b01 = Autorefresh b10 = Modereg or Extended modereg access b11 = NOP.
17-16	BNKA	W	-	Bits mapped to external memory bank address bits when command is Mode_reg access
15-14	-			Reserved
13-0	ADR	W	-	Bits mapped to external memory address bits [13:0] when command is Mode_reg access

Configuration Register 0xB030200C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									ACTCH		QOS_MBITS			MBURST	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MBURST	STOP_MCLK	AUTO_PD	PD_PRD							ROW_BITS			COL_BITS		

Field	Name	RW	Reset	Description
22-21	ACTCH	R/W	0	Enables the refresh command generation for the number of memory chips. It is only possible to generate commands up to and including the number of chips in the configuration that the status Register defines: 2'b00 = 1 chip 2'b01 = 2 chips 2'b10 = Reserved 2'b11 = Reserved
20-18	QOS_MBITS	R/W	0	Controls which ARID signals that the DDR2 SDRAM Controller uses when it selects the QoS value for the AXI read transfer: b000 = ARID[3:0] b001 = ARID[4:1] b010 = ARID[5:2] b011 = ARID[6:3] b100 = ARID[7:4] b101 = ARID[8:5] b110 = ARID[9:6] b111 = ARID[10:7].
17-15	MBURST	R/W	0x2	Encodes the number of data accesses that are performed to the DDR2 SDRAM for each Read and Write command: b010 = Burst 4. This value must also be programmed into the DDR2 SDRAM mode register using the Direct Command Register.
14	STOP_MCLK	R/W	0	When set to 1, the clk_out[MEMORIES-1:0] signals are dynamically stopped after the memories enter self-refresh mode.
13	AUTO_PD	R/W	0	When this is set, the memory interface automatically places the DDR2 SDRAM into power-down state by deasserting cke when the command FIFO has been empty for PD_PRD memory clock cycles.
12-7	PD_PRD	R/W	0	Number of memory clock cycles for auto power-down of the DDR2 SDRAM.
6	-			Reserved
5-3	ROW_BITS	R/W	0x4	Encodes the number of bits of the AXI address that comprise the row address: 3'b000 = 11 bits 3'b001 = 12 bits 3'b010 = 13 bits 3'b011 = 14 bits 3'b100 = 15 bits 3'b101 = 16 bits. b110-b111 = reserved.
2-0	COL_BITS	R/W	0	Encodes the number of bits of the AXI address that comprise the column address: b000 = Reserved. b001 = 9 bits. b010 = 10 bits. b011 = 11 bits. This means that A0-A9, and A11 are used for column address because A10 is a dedicated AP bit. b100-b111 = Reserved.

Refresh Period Register 0xB0302010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	REF_PRD														

Field	Name	RW	Reset	Description
14-0	REF_PRD	R/W	0x0A2C	Memory refresh period in memory clock cycles

CAS Latency Register 0xB0302014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												CAS			

Field	Name	RW	Reset	Description
3-1	CAS	R/W	0xA	CAS latency in memory clock cycles.

Write Latency Register 0xB0302018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														WRL	

The write_latency Register indicates the write latency in memory clock cycles.

Field	Name	RW	Reset	Description
2-0	t_dqss	RO	0x4	Write latency in memory clock cycles

t_mrd Register 0xB030201C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									t_mrd						

The read/write t_mrd Register sets the mode register command time in memory clock cycles.

Field	Name	RW	Reset	Description
6-0	t_mrd	R/W	0x2	Sets mode register command time in memory clock cycles

t_ras Register 0xB0302020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
											t_ras				

The read/write t_ras Register sets the RAS to precharge delay in memory clock cycles.

Field	Name	RW	Reset	Description
4-0	t_ras	R/W	0xE	Sets RAS to precharge delay in memory clock cycles

t_rc Register 0xB0302024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
											t_rc				

The read/write t_rc Register sets the Active bank x to Active bank x delay in memory clock cycles.

Field	Name	RW	Reset	Description
4-0	t_rc	R/W	0x12	Sets Active bank x to Active bank x delay in memory clock cycles

t_rcd Register 0xB0302028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					SCH_RCD								t_rcd		

The read/write t_rcd Register sets the RAS to CAS minimum delay in memory clock cycles.

Field	Name	RW	Reset	Description
10-8	SCH_RCD	R/W	0x3	Sets the RAS to CAS minimum delay in clock cycles minus 3. It is used as a scheduler delay and values in the range 0-4 are supported.
7-3	-	-	-	Reserved
2-0	t_rcd	R/W	0x5	Sets the RAS to CAS minimum delay in memory clock cycles.

t_rfc Register 0xB030202C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SCH_RFC								t_rfc						

The read/write t_rfc Register sets the auto-refresh command time in memory clock cycles.

Field	Name	RW	Reset	Description
14-8	SCH_RFC	R/W	0x21	Sets the Autorefresh command time in clock cycles minus 3. It is used as a scheduler delay.
7	-	-	-	Reserved
6-0	t_rfc	R/W	0x23	Sets the auto-refresh command time in memory clock cycles.

t_rp Register 0xB0302030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					SCH_RP								t_rp		

The read/write t_rp Register sets the precharge to RAS delay in memory clock cycles.

Field	Name	RW	Reset	Description
10-8	SCH_RP	R/W	0x3	Sets the precharge to RAS delay in clock cycles, minus 3. It is used as a scheduler delay and values in the range 0-4 are supported.
3-0	t_rp	R/W	0x5	Sets the precharge to RAS delay in memory clock cycles.

t_rrd Register 0xB0302034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												t_rrd			

The read/write t_rrd Register sets the Active bank x to Active bank y delay in memory clock cycles.

Field	Name	RW	Reset	Description
3-0	t_rrd	R/W	0x4	Sets Active bank x to Active bank y delay in memory clock cycles

t_wr Register 0xB0302038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													t_wr		

The read/write t_wr Register sets the write to precharge delay in memory clock cycles.

Field	Name	RW	Reset	Description
2-0	t_wr	R/W	0x5	Sets the write to precharge delay in memory clock cycles

t_wtr Register 0xB030203C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													t_wtr		

The read/write t_wtr Register sets the write to read delay in memory clock cycles.

Field	Name	RW	Reset	Description
2-0	t_wtr	R/W	0x4	Sets the write to precharge delay in memory clock cycles

t_xp Register 0xB0302040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								t_xp							

The read/write t_wtr Register sets the write to read delay in memory clock cycles.

Field	Name	RW	Reset	Description
7-0	t_xp	R/W	0x2	Sets the exit power-down command time in memory clock cycles

t_xsr Register 0xB0302044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								t_xsr							

The read/write t_xsr Register sets exit self-refresh command time in memory clock cycles.

Field	Name	RW	Reset	Description
7-0	t_xsr	R/W	0x27	Sets the exit self-refresh command time in memory clock cycles

t_esr Register 0xB0302048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								t_esr							

The read/write t_xsr Register sets self-refresh command time in memory clock cycles.

Field	Name	RW	Reset	Description
7-0	t_esr	R/W	0x14	Sets the self-refresh command time in memory clock cycles

Memory_cfg2 Register 0xB030204C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								WIDTH		BNK_BITS		CKE_INIT	DQM_INIT	CLK_CFG	

The read/write memory_cfg2 Register determines the operating state of the memory controller and the memory. At reset the register value is set by the tie-off pins with the same names as the register names. After power-up you can change the register value through the APB interface.

Field	Name	RW	Reset	Description
7-6	WIDTH	R/W	-	Encodes the physical memory width of the attached memory device. This is half the memory_width in the Memory Controller Status Register. The value is dependent on the rendered configuration of the implementation: b00 = 16-bit b01 = 32-bit b10 = Reserved b11 = Reserved.
5-4	BNK_BITS	R/W	-	Encodes the number of bits of the AXI address that comprise the bank address. The value is dependent on the rendered configuration of the implementation: b10 = 0 bits, not supported by the DDR2 SDRAM Controller b01 = 1 bits, not supported by the DDR2 SDRAM Controller b00 = 2 bits b11 = 3 bits.
3	CKE_INIT	R/W	-	State of cke when mresetn is de-asserted..
2	DQM_INIT	R/W	-	State of dqm[MEMBYTES-1:0] when mresetn is de-asserted.
1-0	CLK_CFG	R/W	-	Encodes the clocking scheme: b00 = aclk and mclk are asynchronous b01 = aclk and mclk are synchronous, and aclk is the same frequency or slower than mclk b10 = reserved b11 = aclk and mclk are synchronous, and aclk is greater than mclk.

Memory_cfg3 Register 0xB0302050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													REF_TIMEOUT		

The memory_cfg3 Register determines the number of acceptable outstanding refreshes on a chip before a refresh timeout occurs and refreshes are raised to the highest priority in the queue.

Field	Name	RW	Reset	Description
2-0	REF_TIMEOUT	R/W	0x7	Sets the number of acceptable outstanding refreshes on a chip before a refresh timeout occurs and refreshes are raised to the highest priority in the queue.

t_faw Register 0xB0302054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			SCH_FAW								t_faw				

The t_faw Register sets four bank activate time in memory clock cycles

Field	Name	RW	Reset	Description
12-8	SCH_FAW	R/W	0x1	t_faw in clock cycles, minus 3. Used as a scheduler delay.
7-5	-	-	-	Reserved
4-0	t_faw	R/W	0x1	Four-bank activate period in clock cycles.

4.7.3 LPDDR/LPDDR2/DDR2 SDRAM Controller Registers

Controller control Register(CONCONTROL) **0xB0305000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
phy_type				timeout_cnt											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
rd_fetch				qos_fast_en	dq_swap	chip1_empty	Chip0_empty	drv_en	Reserved	aref_en	out_of	div_pipe	clk_ratio		async

Field	Name	RW	Reset	Description
31-28	phy_type	R/W	0x3	Type of a SDRAM PHY 0x0 = reserved, 0x1 = LPDDR2 PHY, 0x2 = reserved, 0x3 = DDR3 PHY, 0x4 ~ 0xf = reserved
27-16	timeout_cnt	R/W	0xFFF	Default Timeout Cycles 0xn = n aclk cycles (aclk : AXI clock) This counter prevents transactions in the command queue from starvation. This counter starts if a new AXI transaction comes into the queue. If the counter becomes zero, the corresponding transaction becomes the highest priority command of all the transactions in the command queue. This is a default timeout counter and can be overridden by the QoS counter if the ARID or AWID matched with the QoS ID comes into the command queue.
15-12	rd_fetch	R/W	0x1	Read Data Fetch Cycles 0xn = n mclk cycles (mclk : Memory clock) This register is for the unpredictable latency of read data coming from memory devices by tDQSCK variation or the board flying time. The read fetch delay of PHY read FIFO must be controlled by this parameter. The controller will fetch read data from PHY after read_latency + n mclk cycles.
11	qos_fast_en	R/W	0x0	Adaptive QoS Enable 0x0 = disable, 0x1 = enable, If enabled, the controller loads QoS counter value from QoSControl.qos_cnt_f instead of QoSControl.qos_cnt if the corresponding input pin qos_fast is turned on.
10	dq_swap	R/W	0x0	DQ Swap 0x0 = Disable, 0x1 = Enable, If enabled, the controller reverses the bit order of memory data pins. (For example, DQ[31] <-> DQ[0], DQ[30] <-> DQ[1])
9	chip1_empty	R	0x1	Command Queue Status of Chip1 0x0 = Not Empty, 0x1 = Empty There is no AXI transaction corresponding to chip1 memory in the command queue entries
8	chip0_empty	R	0x1	Command Queue Status of Chip0 0x0 = Not Empty, 0x1 = Empty There is no AXI transaction corresponding to chip0 memory in the command queue entries
7	drv_en	R/W	0x0	PHY Driving 0x0 = Disable, 0x1 = Enable During the high-Z state of the memory bidirectional pins, PHY can drive these pins

				with the zeros or pull down these pins for preventing current leakage. Set PhyControl1.drv_type register to select driving type
6	Reserved	R	0x0	should be zero
5	aref_en	R/W	0x0	Auto Refresh Counter 0x0 = Disable, 0x1 = Enable Enable this to decrease the auto refresh counter by 1 at the rising edge of the mclk
4	out_of	R/W	0x1	Out of Order Scheduling 0x0 = Disable, 0x1 = Enable The embedded scheduler enables out-of order operation to improve SDRAM utilization
3	div_pipe	R/W	0x0	Pipeline Stage Dividing 0x0 = Disable, 0x1 = Enable For high speed memory devices (> 200MHz), the pipeline stage of the scheduler should be divided. This function should be enabled if use the high speed PHY
2-1	clk_ratio	R/W	0x0	Clock Ratio of Bus Clock to Memory Clock 0x0 = freq.(aclk) : freq.(mclk) = 1 : 1, 0x1 = Reserved 0x2 ~ 0x3 = Reserved
0	async	R/W	0x0	Multi Clock Operation 0x0 = Synchronous operation between AXI and SDRAM domain, 0x1 = Reserved ***Should be set to "0x0"***

Memory Control Register(MemControl) 0xB0305004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									bl			num_chip			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
mem_width				mem_type				add_lat_pall		dsref_en	tp_en	dpwrdn_type		dpwrdn_en	clk_stop_en

Field	Name	RW	Reset	Description
31-23	Reserved	-	-	
22-20	bl	R/W	0x2	**Memory Burst Length** 0x0 = Reserved, 0x1 = 2, 0x2 = 4, 0x3 = 8, 0x4 = 16, 0x5 ~ 0x7 = Reserved In case of DDR2/LPDDR2, the controller only supports burst length 4.
19-16	num_chip	R/W	0x0	**Number of Memory Chips** 0x0 = 1 chip, 0x1 = 2 chips, 0x2 ~ 0xf = Reserved
15-12	mem_width	R/W	0x2	**Width of Memory Data Bus** 0x0 = Reserved, 0x1 = 16-bit, 0x2 = 32-bit, 0x3 ~ 0xf = Reserved
11-8	mem_type	R/W	0x1	**Type of Memory** 0x0 = Reserved, 0x1 = LPDDR 0x2 = LPDDR2, 0x3 = Reserved, 0x4 = DDR2, 0x5 ~ 0xf = Reserved
7-6	add_lat_pall	R/W	0x0	**Additional Latency for PALL** 0x0 = 0 cycle, 0x1 = 1 cycle 0x2 = 2 cycle, 0x3 = 3 cycle If all banks precharge command is issued, the latency of precharging will be tRP + add_lat_pall
5	dsref_en	R/W	0x0	**Dynamic Self Refresh** 0x0 = Disable, 0x1 = Enable Refer to chapter 5.2. Dynamic power down for detailed information.
4	tp_en	R/W	0x0	**Timeout Precharge** 0x0 = Disable, 0x1 = Enable If tp_en is enabled, it automatically precharges an open bank after a specified amount of mclk cycles (if no access has been made in between the cycles) in an open page policy. If **PrechConfig.tp_cnt** bit-field is set, it specifies the amount of mclk cycles to wait until timeout precharge precharges the open bank.
3-2	dpwrdn_type	R/W	0x0	**Type of Dynamic Power Down** 0x0 = Active/precharge power down,

				0x1 = Forced precharge power down 0x2 ~ 0x3 = Reserved Refer to chapter 5.2. Dynamic power down for detailed information.
1	dpwrdn_en	R/W	0x0	**Dynamic Power Down** 0x0 = Disable, 0x1 = Enable
0	clk_stop_en	R/W	0x0	**Dynamic Clock Control** 0x0 = Always running, 0x1 = Stops during idle periods Refer to chapter 5.4. Clock stop for detailed information

Memory Chip0 Configuration Register(MemConfig0) 0xB0305008

9	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chip_base								chip_mask							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
chip_map				chip_col				chip_row				chip_bank			

Field	Name	RW	Reset	Description
31-24	chip_base	R/W	0x20	**AXI Base Address** AXI base address [31:24] = chip_base, For example, if chip_base = 0x20, then AXI base address of memory chip0 becomes 0x2000_0000.
23-16	chip_mask	R/W	0xF8	**AXI Base Address Mask** Upper address bit mask to determine AXI offset address of memory chip0. 0 = Corresponding address bit is not to be used for comparison 1 = Corresponding address bit is to be used for comparison For example, if chip_mask = 0xF8, then AXI offset address becomes 0x0000_0000 ~ 0x07FF_FFFF. If AXI base address of memory chip0 is 0x2000_0000, then memory chip0 has an address range of 0x2000_0000 ~ 0x27FF_FFFF.
15-12	chip_map	R/W	0x0	**Address Mapping Method (AXI to Memory)** 0x0 = Linear ({bank, row, column, width}), 0x1 = Interleaved ({row, bank, column, width}), 0x2 ~ 0xf = Reserved
11-8	chip_col	R/W	0x3	**Number of Column Address Bits** 0x0 = Reserved 0x1 = 8 bits, 0x2 = 9 bits, 0x3 = 10 bits, 0x4 = 11 bits, 0x5 ~ 0xf = Reserved
7-4	chip_row	R/W	0x1	**Number of Row Address Bits** 0x0 = 12 bits, 0x1 = 13 bits, 0x2 = 14 bits, 0x3 = 15 bits, 0x4 ~ 0xf = Reserved
3-0	chip_bank	R/W	0x2	**Number of Banks** 0x0 = 1 bank, 0x1 = 2 banks, 0x2 = 4 banks, 0x3 = 8 banks, 0x4 ~ 0xf = Reserved

Memory Chip1 Configuration Register(MemConfig1) 0xB030500C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chip_base								chip_mask							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
chip_map				chip_col				chip_row				chip_bank			

Field	Name	RW	Reset	Description
31-24	chip_base	R/W	0x28	**AXI Base Address** AXI base address [31:24] = chip_base, For example, if chip_base = 0x28, then AXI base address of memory chip1 becomes 0x2800_0000.
23-16	chip_mask	R/W	0xF8	**AXI Base Address Mask** Upper address bit mask to determine AXI offset address of memory chip0. 0 = Corresponding address bit is not to be used for comparison 1 = Corresponding address bit is to be used for comparison For example, if chip_mask = 0xF0, then AXI offset address becomes 0x0000_0000 ~ 0x0FFF_FFFF. If AXI base address of memory chip1 is 0x2800_0000, then memory chip1 has an address range of 0x2800_0000 ~ 0x37FF_FFFF.
15-12	chip_map	R/W	0x0	**Address Mapping Method (AXI to Memory)** 0x0 = Linear ({bank, row, column, width}), 0x1 = Interleaved ({row, bank, column, width}), 0x2 ~ 0xf = Reserved
11-8	chip_col	R/W	0x3	**Number of Column Address Bits** 0x0 = Reserved 0x1 = 8 bits, 0x2 = 9 bits, 0x3 = 10 bits, 0x4 = 11 bits, 0x5 ~ 0xf = Reserved
7-4	chip_row	R/W	0x1	**Number of Row Address Bits** 0x0 = 12 bits, 0x1 = 13 bits, 0x2 = 14 bits, 0x3 = 15 bits, 0x4 ~ 0xf = Reserved
3-0	chip_bank	R/W	0x2	**Number of Banks** 0x0 = 1 bank, 0x1 = 2 banks, 0x2 = 4 banks, 0x3 = 8 banks, 0x4 ~ 0xf = Reserved

Memory Direct Command Register(DirectCmd) 0xB0305010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reseved				cmd_type				Reserved			cmd_chip		cmd_bank		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	cmd_addr														

Field	Name	RW	Reset	Description
31-28	reseved	-	-	
27-24	cmd_type	R/W	0x0	**Type of Direct Command** 0x0 = MRS/EMRS (mode register setting), 0x1 = PALL (all banks precharge), 0x2 = PRE (per bank precharge), 0x3 = DPD (deep power down), 0x4 = REFS (self refresh), 0x5 = REFA (auto refresh), 0x6 = CKEL (active/precharge power down), 0x7 = NOP (exit from active/precharge power down or deep power down, 0x8 = REFSX (exit from self refresh) 0x9 = MRR (mode register reading), 0xa ~ 0xf = Reserved When a direct command is issued, AXI masters must not access memory. It is strongly recommended to check the command queue's state by **ConControl.chip0/1_empty** before issuing a direct command. And dynamic power down, dynamic self refresh and force precharge function (**MemControl** register) must be disabled. MRS/EMRS or MRR commands should be issued if all banks are in idle state. If MRS/EMRS or MRR is issued to LPDDR2, the CA pins must be mapped as follows. MA[7:0] = {cmd_addr[1:0], cmd_bank[2:0], cmd_addr[12:10]}, OP[7:0] = cmd_addr[9:2]
23-21	Reserved	-	-	
20	cmd_chip	R/W	0x0	**Chip Number to send the direct command to** 0 = Chip 0 1 = Chip 1
18-16	cmd_bank	R/W	0x0	**Related Bank Address when issuing a direct command** To send a direct command to a chip, additional information such as the bank address is required. This register is used in such situations.
14-0	cmd_addr	R/W	0x0	**Related Address value when issuing a direct command** To send a direct command to a chip, additional information such as the address is required. This register is used in such situations.

Precharge Policy Configuration Register(PrechConfig) 0xB0305014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
tp_cnt								Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
chip1_policy								chip0_policy							

Field	Name	RW	Reset	Description
31-24	tp_cnt	R/W	0xFF	**Timeout Precharge Cycles** 0xn = n mclk cycles, If the timeout precharge function (**MemControl.tp_en**) is enabled and the timeout precharge counter becomes zero, the controller forces the activated memory bank into the precharged state. Refer to chapter 6.2.Timeout precharge for detailed information.
23-16	Reserved	-	-	
15-8	chip1_policy	R/W	0x0	**Memory Chip1 Precharge Bank Selective Policy** 0x0 = Open page policy, 0x1 = Close page (auto precharge) policy chip1_policy[n], n is the bank number of chip1. Open Page Policy: After a READ or WRITE, the row that was accessed is left open. Close Page (Auto Precharge) Policy: Right after a READ or WRITE command, memory devices automatically precharges the bank. This is a bank selective precharge policy. For example, if chip1_policy[2] is 0x0, bank2 of chip1 has an open page policy and if chip1_policy[6] is 0x1, bank6 of chip1 has a close page policy.
7-0	chip0_policy	R/W	0x0	**Memory Chip0 Precharge Bank Selective Policy** 0x0 = Open page policy, 0x1 = Close page (auto precharge) policy Chip0_policy[n], n is the bank number of chip0. This is for memory chip0.

PHY Control0 Registger(PhyControl0) 0xB0305018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ctrl_force								ctrl_inc							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ctrl_start_point								dqs_delay				ctrl_dfdqs	ctrl_half	ctrl_dll_on	ctrl_start

Field	Name	RW	Reset	Description
31-24	ctrl_force	R/W	0x0	**DLL Force Delay** This field is used instead of **PhyStatus.ctrl_lock_value**[9:2] from the DLL only when **PhyControl0.ctrl_dll_on** is LOW. (i.e. If the DLL is off, this field is used to generate 270' clock and shift DQS by 90'.)
23-16	ctrl_inc	R/W	0x10	**DLL Delay Increment** Increase the amount of start point This value should be 0x10
15-8	ctrl_start_point	R/W	0x10	**DLL Lock Start Point** Initial DLL lock start point. This is the number of delay cells and is the start point where "DLL" start tracing to lock. Calculates initial delay time by multiplying the unit delay of delay cell and this value. This value should be 0x10
7-4	dqs_delay	R/W	0x0	**Delay Cycles for DQS Cleaning** This register is to enable PHY to clean incoming DQS signals delayed by external circumstances. If DQS is coming with read latency plus *n* mclk cycles, this registers must be set to *n* mclk cycles.
3	ctrl_dfdqs	R/W	0x0	**Differential DQS** If enabled, PHY generates differential DQS out signals for write command and receives differential DQS input signals for read command. This function can be used in case of DDR2/LPDDR2.
2	ctrl_half	R/W	0x0	**DLL Low Speed** HIGH active signal to turn on the low speed mode for DLL. If this bit is set, DLL can run at low speed (80MHz ~ 100MHz)
1	ctrl_dll_on	R/W	0x0	**DLL On** HIGH active start signal to turn on the DLL. This signal should be kept HIGH for normal operation. If this signal becomes LOW, DLL is turned off and ctrl_clock and ctrl_flock become HIGH. This bit should be kept set before ctrl_start is set to turn on the DLL
0	ctrl_start	R/W	0x0	**DLL Start** HIGH active start signal to make the DLL run and lock. This signal should be kept HIGH during normal operation. If this signal becomes LOW, DLL stops running. To re-run DLL, make this signal HIGH again. In the case of re-running, DLL loses previous lock information. Before ctrl_start is set, make sure that ctrl_dll_on is HIGH.

PHY Control1 Registger(PhyControl1) 0xB030501C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
term_write_en	term_read_en	ctrl_shgate	ctrl_pd					ctrl_cmosrcv	ctrl_offsetd						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
drv_type	ctrl_offsetc							ctrl_ref				fp_resync	ctrl_shiftc		

Field	Name	RW	Reset	Description
31	term_write_en	R/W	0x0	**Termination Enable for Write** At high-speed memory operation, it can be necessary to turn on termination resistance in memory devices. This register control s an ODT pin of a memory device.
30	term_read_en	R/W	0x0	**Termination Enable for Read** At high-speed memory operation, it can be necessary to turn on termination resistance in the PHY. This register should control an ODT pin of memory device
29	ctrl_shgate	R/W	0x0	**Duration of DQS Gating Signal** This field controls the gate control signal 1'b0 = (gate signal length = "burst length / 2" (<= 200MHz)) 1'b1 = (gate signal length = "burst length / 2" – 1 (> 200MHz))
28-24	ctrl_pd	R/W	0x0	**Input Gate for Power Down** If this field is set, input buffer is off for power down. This field should be 0 for normal operation. ctrl_pd[3:0] = for each data slice, ctrl_pd[4] = for control slice.
23	ctrl_cmosrcv	R/W	0x0	**I/O Type** This field controls the input mode of I/O 1'b0 = Differential receiver mode for high speed operation 1'b1 = CMOS receiver mode for low speed operation (< 200MHz)
22-16	ctrl_offsetd	R/W	0x0	This field is for debug purpose. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. offset amount for 270' clock generation ctrl_offsetd[6] = 1 : (tFS : fine step delay) 270' delay amount - ctrl_offsetd[5:0] **x tFS** ctrl_offsetd[6] = 0 : 270' delay amount + ctrl_offsetd[5:0] **x tFS**
15	drv_type	R/W	0x0	**Driving Type of Bidirectional Pins in Idle State** 0x0 = Drive all to zeros, 0x1 = Pull down all If CAS or read data latency is 2, don't set this register to 0x0.
14-8	ctrl_offsetc	R/W	0x0	**Delay Offset for DQS Cleaning** Gate offset amount for DDR. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. ctrl_offsetc[6] = 1 : (tFS : fine step delay) GATEout delay amount - ctrl_offsetc[5:0] **x tFS**
7-4	ctrl_ref	R/W	0x0	**Reference Count for DLL Lock Confirmation** This field determines the period of time when

				ctrl_locked is cleared. 4'b0000 : Don't use. 4'b0001 : ctrl_flock is de-asserted during 6 clock cycles, ctrl_locked is deasserted. 4'b0010 : ctrl_flock is de-asserted during 9 clock cycles, ctrl_locked is deasserted. ~ 4'b1111 : ctrl_flock is de-asserted during 48 clock cycles, ctrl_locked is deasserted.
3	fp_resync	R/W	0x0	**Force DLL Resyncronization** Active RISIG-EDGE signal. This signal should become LOW after set HIGH for normal operation. When this bit transits from LOW to HIGH, pointers of FIFO and DLL information is updated. In general, this bit should be set during initialization and refresh cycles. Duration of ctrl_resync should be 1 to 2 cycles. Before set and clear this signal, ctrl_clock or ctrl_flock should be checked during initialization. During memory access, ctrl_clock or ctrl_flock doesn't need to be checked. ctrl_resync should be set and cleared only when ctrl_clock is set after auto-refresh is started. Refer to the application note for ctrl_resync timing.
2-0	ctrl_shiftc	R/W	0x0	**Phase Delay for DQS Cleaning** GATEout signal delay amount for DDR. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. 000 = T/128 (2.8125' shift), 001 = T/64 (5.625' shift), 010 = T/32 (11.25' shift), 011 = T/16 (22.5' shift), 100 = T/8 (45' shift), 101 = T/4 (90' shift), 110 = T/2 (180' shift), 111 = T (360' shift) Recommended values according to memory type : 100 when LPDDR/LPDDR2, 110 when DDR/DDR2

PHY Control2 Registger(PhyControl2) 0xB0305020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ctrl_offsetr3								ctrl_offsetr2						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	ctrl_offsetr1								ctrl_offsetr0						

Field	Name	RW	Reset	Description
31	Reserved	-	-	
30-24	ctrl_offsetr3	R/W	0x0	
23	Reserved	-	-	
22-16	ctrl_offsetr2	R/W	0x0	This field can be used to give offset to read DQS. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. Read DQS offset amount : ctrl_offsetr2[6] = 1 : (tFS : fine step delay) Read DQS 90° delay amount − ctrl_offsetr2[5:0] x tFS ctrl_offsetr2
15	Reserved	-	-	
14-8	ctrl_offsetr1	R/W	0x0	This field can be used to give offset to read DQS. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. Read DQS offset amount : ctrl_offsetr1[6] = 1 : (tFS : fine step delay) Read DQS 90° delay amount − ctrl_offsetr1[5:0] x tFS ctrl_offsetr1[6] = 0 : Read DQS 90° delay amount + ctrl_offsetr1[5:0] x tFS.
7	Reserved	-	-	
6-0	ctrl_offsetr0	R/W	0x0	This field can be used to give offset to read DQS. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. Read DQS offset amount : ctrl_offsetr0[6] = 1 : (tFS : fine step delay) Read DQS 90° delay amount − ctrl_offsetr0[5:0] x tFS ctrl_offsetr0[6] = 0 : Read DQS 90° delay amount + ctrl_offsetr0[5:0] x tFS.

PHY Control3 Registger(PhyControl3) 0xB0305024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ctrl_offsetw3								ctrl_offsetw2						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ctrl_offsetw1								ctrl_offsetw0						

Field	Name	RW	Reset	Description
31	Reserved	-	-	
30-24	ctrl_offsetw3	R/W	0x0	This field can be used to give offset to write DQS. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. Write DQS offset amount : ctrl_offsetw3[6] = 1 : (tFS : fine step delay) Write DQS 90° delay amount − ctrl_offsetw3[5:0] x tFS ctrl_offsetw3[6] = 0 : Write DQS 90° delay amount + ctrl_offsetw3[5:0] x tFS.
23	Reserved	-	-	
22-16	ctrl_offsetw2	R/W	0x0	This field can be used to give offset to write DQS. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. Write DQS offset amount : ctrl_offsetw2[6] = 1 : (tFS : fine step delay) Write DQS 90° delay amount − ctrl_offsetw2[5:0] x tFS ctrl_offsetw2[6] = 0 : Write DQS 90° delay amount + ctrl_offsetw2[5:0] x tFS.
15	Reserved	-	-	
14-8	ctrl_offsetw1	R/W	0x0	This field can be used to give offset to write DQS. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. Write DQS offset amount : ctrl_offsetw1[6] = 1 : (tFS : fine step delay) Write DQS 90° delay amount − ctrl_offsetw1[5:0] x tFS ctrl_offsetw1[6] = 0 : Write DQS 90° delay amount + ctrl_offsetw1[5:0] x tFS.
7	Reserved	-	-	
6-0	ctrl_offsetw0	R/W	0x0	This field can be used to give offset to write DQS. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. Write DQS offset amount : ctrl_offsetw0[6] = 1 : (tFS : fine step delay) Write DQS 90° delay amount − ctrl_offsetw0[5:0] x tFS ctrl_offsetw0[6] = 0 : Write DQS 90° delay amount +

				ctrl_offsetw0[5:0] x tFS.

Dynamic Power Down Configuration Register(PwrdnConfig) 0xB0305028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dsref_cyc															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								dpwrdn_cyc							

Field	Name	RW	Reset	Description
31-16	dsref_cyc	R/W	0xFFFF	**Number of Cycles for dynamic self refresh entry** 0xn = n aclk cycles, If the command queue is empty for n+1 cycles, the controller forces memory devices into self refresh state
15-8	Reserved	-	-	
7-0	dpwrdn_cyc	R/W	0xFF	**Number of Cycles for dynamic power down entry** 0xn = n aclk cycles, If the command queue is empty for n+1 cycles, the controller forces the memory device into active/precharge power down state.

AC Timing Register for Auto Refresh of Memory(TimingAref) 0xB0305030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
t_refi															

Field	Name	RW	Reset	Description
31-16	Reserved	-	-	
15-0	t_refi	R/W	0x40E	**Average Periodic Refresh Interval** t_refi * T(mclk) should be greater than or equal to the minimum value of memory tREFI (all bank), for example, for the all bank refresh periord of 7.8us, and an mclk frequency of 133MHz, the following value should be programmed into it : 7.8 us * 133 MHz = 1038

AC Timing Register for the Row of Memory (TimingRow) 0xB0305034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
t_rfc								t_rrd				t_rp			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
t_rcd				t_rc						t_ras					

Field	Name	RW	Reset	Description
31-24	t_rfc	R/W	0xF	**Auto refresh to Active / Auto refresh command period, in cycles** t_rfc * T(mclk) should be greater than or equal to the minimum value of memory tRFC
23-20	t_rrd	R/W	0x2	**Active bank A to Active bank B delay, in cycles** t_rrd * T(mclk) should be greater than or equal to the minimum value of memory tRRD
19-16	t_rp	R/W	0x3	**Precharge command period, in cycles** t_rp * T(mclk) should be greater than or equal to the minimum value of memory tRP
15-12	t_rcd	R/W	0x3	**Active to Read or Write delay, in cycles** t_rcd * T(mclk) should be greater than or equal to the minimum value of memory tRCD
11-6	t_rc	R/W	0xA	**Active to Active period, in cycles** t_rc * T(mclk) should be greater than or equal to the minimum value of memory tRC
5-0	t_ras	R/W	0x6	**Active to Precharge command period, in cycles** t_ras * T(mclk) should be greater than or equal to the minimum value of memory tRAS

AC Timing Register for the Data of Memory (TimingData) 0xB0305038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
t_wtr				t_wr				t_rtp				cl			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				wl								rl			

Field	Name	RW	Reset	Description
31-28	t_wtr	R/W	0x1	**Internal write to Read command delay, in cycles** t_wtr * T(mclk) should be greater than or equal to the minimum value of memory tWTR t_wtr must be 0x1 in case of JEDEC LPDDR/DDR
27-24	t_wr	R/W	0x2	**A Write recovery time, in cycles** t_wr * T(mclk) should be greater than or equal to the minimum value of memory tWR
23-20	t_rtp	R/W	0x1	**Internal read to Precharge command delay, in cycles** t_rtp * T(mclk) should be greater than or equal to the minimum value of memory tRTP t_rtp must be 0x1 in case of JEDEC LPDDR/DDR
19-16	cl	R/W	0x3	**CAS Latency (for LPDDR/DDR/DDR2), in cycles** cl should be greater than or equal to the minimum value of memory CL
15-12	reserved	-	-	
11-8	wl	R/W	0x2	**Active to Active period, in cycles** t_rc * T(mclk) should be greater than or equal to the minimum value of memory tRC
7-4	reserved	-	-	
3-0	rl	R/W	0x4	**Read data latency (for only LPDDR2), in cycles** rl should be greater than or equal to the minimum value of memory RL

AC Timing Register for the Power Modes of Memory(TimingPower) 0xB030503C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		t_faw						t_xsr							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
t_xp								t_cke				t_mrd			

Field	Name	RW	Reset	Description
31-30	Reserved	-	-	
29-24	t_faw	R/W	0xE	**Four Active Window(for DDR2/LPDDR2)** t_faw * T(mclk) should be greater than or equal to the minimum value of memory tFAW
23-16	t_xsr	R/W	0x1B	**Self refresh exit power down to next valid command delay, in cycles** t_xsr * T(mclk) should be greater than or equal to the minimum value of memory tXSR
15-8	t_xp	R/W	0x4	**Exit power down to next valid command delay, in cycles** t_xp * T(mclk) should be greater than or equal to the minimum value of memory tXP
7-4	t_cke	R/W	0x2	**CKE minimum pulse width (minimum power down mode duration), in cycles** t_cke should be greater than or equal to the minimum value of memory tCKE
3-0	t_mrd	R/W	0x2	**Mode Register Set command period, in cycles** t_mrd should be greater than or equal to the minimum value of memory tMRD

PHY Status Register(PhyStatus) 0xB0305040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					ctrl_zq_pmon				ctrl_zq_nmon					ctrl_zq_error	ctrl_zq_end
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
		ctrl_lock_value											ctrl_locked	ctrl_flock	ctrl_clock

Field	Name	RW	Reset	Description
31-27	Reserved	-	-	
26-24	ctrl_zq_pmon	R	0x0	**Control Code Found by Auto Calibration for Pull Up**
23	Reserved	-	-	
22-20	ctrl_zq_nmon	R	0x0	**Control Code Found by Auto Calibration for Pull Down**
19-18	Reserved	-	-	
17	ctrl_zq_error	R	0x0	**Calibration Error** If this register is set to 0x1, the auto ZQ calibration is finished with error.
16	ctrl_zq_end	R	0x0	**Calibration Completion** If this register is set to 0x1, the auto ZQ calibration is finished without error
15-14	Reserved	-	-	
13-4	ctrl_lock_value	R	0x0	**Locked Delay** Locked delay line encoding value ctrl_lock_value[9:2] = Number of delay cells for coarse lock ctrl_lock_value[1:0] = Control value for fine lock
3	Reserved	-	-	
2	ctrl_locked	R	0x0	**DLL Lock** 0 = DLL is unlocked, 1 = DLL is locked
1	ctrl_flock	R	0x0	**Fine Lock Information**
0	ctrl_clock	R	0x0	**Coarse Lock Information**

PHY ZQ I/O Control Register(PhyZQControl) 0xB0305044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ctrl_dcc												ctrl_zq_force_impp			ctrl_zq_force_impn
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ctrl_zq_force_impn		ctrl_zq_mode_term			ctrl_zq_mode_dds				ctrl_zq_div				ctrl_zq_force	ctrl_zq_start	ctrl_zq_mode_noterm

Field	Name	RW	Reset	Description
31-20	ctrl_dcc	R/W	0xE38	**Clock Duty Control** PHY DCC (Duty Correction Circuit) can shift clock negative edge from the original position to the leaded or lagged position in compensation for the process variations. The following value shoes the delay value at the best condition. The maximum value which can be shifted by two pairs of DCC will be about +/-60ps. ctrl_dcc[2:0] = 0x0 : Default value, ctrl_dcc[2:0] = 0x1 : Negative edge is lagged by 20ps, ctrl_dcc[2:0] = 0x2 : Negative edge is lagged by 26ps, ctrl_dcc[2:0] = 0x3 : Negative edge is lagged by 29ps, ctrl_dcc[2:0] = 0x4 : Negative edge is lagged by 31ps, ctrl_dcc[2:0] = 0x5 : Negative edge is lagged by 33ps, ctrl_dcc[2:0] = 0x6 : Negative edge is lagged by 34ps, ctrl_dcc[2:0] = 0x7 : Negative edge is lagged by 35ps, ctrl_dcc[5:3] = 0x7 : Default value, ctrl_dcc[5:3] = 0x1 : Negative edge is leaded by 20ps, ctrl_dcc[5:3] = 0x2 : Negative edge is leaded by 26ps, ctrl_dcc[5:3] = 0x3 : Negative edge is leaded by 29ps, ctrl_dcc[5:3] = 0x4 : Negative edge is leaded by 31ps, ctrl_dcc[5:3] = 0x5 : Negative edge is leaded by 33ps, ctrl_dcc[5:3] = 0x6 : Negative edge is leaded by 34ps, ctrl_dcc[5:3] = 0x7 : Negative edge is leaded by 35ps, ctrl_dcc[8:6] : Refer to ctrl_dcc[2:0], ctrl_dcc[11:9] : Refer to ctrl_dcc[5:3],
19-17	ctrl_zq_force_impp	R/W	0x2	**Immediate Control Code for Pull-up**
16-14	ctrl_zq_force_impn	R/W	0x5	**Immediate Control Code for Pull-down**
13-11	ctrl_zq_mode_term	R/W	0x2	**On-die-termination Resistor Value Selection**
10-8	ctrl_zq_mode_dds	R/W	0x7	**Driver Strength Selection**
7	Reserved	-	-	
6-4	ctrl_zq_div	R	0x3	**Calibration I/O Clock Selection Signal** 0x0 = mclk / 2, 0x1 = mclk / 4, 0x2 = mclk / 8, 0x3 = mclk / 16, 0x4 = mclk / 32,

				0x5 ~ 0x7 = Reserved
3	Reserved	-	-	
2	ctrl_zq_force	R/W	0x0	**Force Calibration** If this register is set, ctrl_force_impp[2:0]/impn[2:0] are used instead of calibration control code found after auto calibration
1	ctrl_zq_start	R/W	0x0	**Auto Calibration Start Signal** ZQ I/O calibration starts by setting this register
0	ctrl_zq_mode_noterm	R/W	0x1	**Termination Disable Selection** 0x0 = Termination enable 011 = Termination disable

Memory Chip0 Status Register(Chip0Status) 0xB0305048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bank7_state				bank6_state				bank5_state				bank4_state			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
bank3_state				bank2_state				bank1_state				bank0_state			

Field	Name	RW	Reset	Description
31-28	bank7_state	R	0x0	**Current State of Bank 7 of Memory Chip0**
27-24	bank6_state	R	0x0	**Current State of Bank 6 of Memory Chip0**
23-20	bank5_state	R	0x0	**Current State of Bank 5 of Memory Chip0**
19-16	bank4_state	R	0x0	**Current State of Bank 4 of Memory Chip0**
15-12	bank3_state	R	0x0	**Current State of Bank 3 of Memory Chip0**
11-8	bank2_state	R	0x0	**Current State of Bank 2 of Memory Chip0**
7-4	bank1_state	R	0x0	**Current State of Bank 1 of Memory Chip0**
3-0	bank0_state	R	0x0	**Current State of Bank 0 of Memory Chip0** 0x0 = Idle (precharged), 0x1 = MRS/EMRS, 0x2 = Deep power down, 0x3 = Self refresh, 0x4 = Auto refresh, 0x5 = Precharge power down, 0x6 = Row active, 0x7 = Active power down, 0x8 = Write, 0x9 = Write with auto precharge, 0xA = Read, 0xB = Read with auto precharge, 0xC = Burst stop, 0xD = Precharging, 0xE = MRR, 0xF = Reserved

Memory Chip1 Status Register(Chip1Status) 0xB030504C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bank7_state				bank6_state				bank5_state				bank4_state			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
bank3_state				bank2_state				bank1_state				bank0_state			

Field	Name	RW	Reset	Description
31-28	bank7_state	R	0x0	**Current State of Bank 7 of Memory Chip0**
27-24	bank6_state	R	0x0	**Current State of Bank 6 of Memory Chip1**
23-20	bank5_state	R	0x0	**Current State of Bank 5 of Memory Chip1**
19-16	Bank4_state	R	0x0	**Current State of Bank 4 of Memory Chip1**
15-12	Bank3_state	R	0x0	**Current State of Bank 3 of Memory Chip1**
11-8	bank2_state	R	0x0	**Current State of Bank 2 of Memory Chip1**
7-4	bank1_state	R	0x0	**Current State of Bank 1 of Memory Chip1**
3-0	bank0_state	R	0x0	**Current State of Bank 0 of Memory Chip1** 0x0 = Idle (precharged), 0x1 = MRS/EMRS, 0x2 = Deep power down, 0x3 = Self refresh, 0x4 = Auto refresh, 0x5 = Precharge power down, 0x6 = Row active, 0x7 = Active power down, 0x8 = Write, 0x9 = Write with auto precharge, 0xA = Read, 0xB = Read with auto precharge, 0xC = Burst stop, 0xD = Precharging, 0xE = MRR, 0xF = Reserved

Auto Refresh Clear Status RegisterChip0Status) 0xB0305050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
aref_cnt															

Field	Name	RW	Reset	Description
31-16	Reserved	-	-	
15-0	aref_cnt	R	0x0	**Current Value of Auto Refresh Counter** Shows the current value of all bank auto refresh counter. This is updated if a new t_refi is programmed into the **TimingAref** register and decreases by 1 at the rising edge of mclk. An all bank auto refresh command is issued to memory device and this counter is reloaded with **TimingAref.t_ref** if this becomes zero.

Memory Mode Register(MrStatus) 0xB0305054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								mr_status							

Field	Name	RW	Reset	Description
31-8	Reserved	-	-	
7-0	mr_status	R	0x0	**Mode Registers Status**

PHY Test Register 0 (PhyTest0) 0xB0305058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ctrl_fb_cnt4											ctrl_fb_oky				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ctrl_dis			ctrl_fb_err					ctrl_fnc_fb			ctrl_fb_start				

Field	Name	RW	Reset	Description
31-24	ctrl_fb_cnt4	R	0x0	**Count Value for Control Channel**
23-21	Reserved	R	0x0	ctrl_fb_okay[4] = Okay for control, ctrl_fb_okay[3:0] = Okay for data
20-16	ctrl_fb_oky	R/W	0x0	ctrl_fb_okay[4] = Okay for control, ctrl_fb_okay[3:0] = Okay for data
15	ctrl_dis	R/W	0x0	**Clock Output Disable** This field controls the CK/CKB 1'b0 = Clock output is enabled 1'b1 = Clock output is disabled
14-13	Reserved	-	-	
12-8	ctrl_fb_err	R	0x0	ctrl_fb_err[4] = Error for control, ctrl_fb_err[3:0] = Error for data
7-5	ctrl_fnc_fb	R/W	0x0	**Function Feedback Test** Valid only when {mode_phy, mode_nand, mode_scan, mode_bypass, mode_mux} is 0. 0x0 = Normal operation mode, 0x1 = Reserved, 0x2 = External FNC feedback test mode, 0x3 = Internal FNC feedback test mode, 0x4 = Board PHY external read feedback, 0x5 = Board PHY internal read feedback, 0x6 = Board PHY internal write feedback, 0x7 = Reserved
4-0	ctrl_fb_start	R/W	0x0	ctrl_fb_start[4] = Start for control, ctrl_fb_start[3:0] = Start for data

PHY Test Register 1 (PhyTest1) 0xB030505C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ctrl_fb_cnt3								ctrl_fb_cnt2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ctrl_fb_cnt1								ctrl_fb_cnt0							

Field	Name	RW	Reset	Description
31-24	ctrl_fb_cnt3	R	0x0	**Count Value for Data3 Channel**
23-16	ctrl_fb_cnt2	R	0x0	**Count Value for Data2 Channel**
15-8	ctrl_fb_cnt1	R	0x0	**Count Value for Data1 Channel**
7-0	ctrl_fb_cnt0	R	0x0	**Count Value for Data0 Channel**

Quality of Service Control Register (QoSControl) 0xB0305060 + 8n(n=0,…,15)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				qos_cnt											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
qos_cnt_f															qos_en

Field	Name	RW	Reset	Description
31-28	Reserved	-	-	
27-16	qos_cnt	R/W	0x0	**QoS Cycles** 0xn = n aclk cycles The matched ARID uses this value for its timeout counters instead of **ConControl.timeout_cnt**.
15-4	qos_cnt_f	R/W	0x0	**QoS Cycles for Fast Request** 0xn = n aclk cycles When **Concontrol.qos_fast_en** is enabled and input pin qos_fast[n] bit is 1, this qos_cnt_f value is loaded to the timeout counter
3-1	Reserved	-	-	
0	qos_en	R/W	0x0	**QoS Enable** 0x0 = Disable, 0x1 = Enable If this function is enabled, its timeout counter works and the ARID is masked

Quality of Service Configuration Register n (QoSConfig) 0xB0305064 + 8n(n=0,…,15)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qos_mask															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
qos_id															

Field	Name	RW	Reset	Description
31-16	qos_mask	R/W	0x0	**QoS Mask Bits** This is used to mask the incoming ARID to compare with the **qos_id**. For example, to have 0b00110XX000 IDs the same QoS, the 4th and 5th bits need to be masked. Therefore, **qos_mask** would be 0b1111100111.
15-0	qos_id	R/W	0x0	**QoS ID** This is used to compare with the masked ARID to check whether its timeout counter should be used for QoS. After applying the qos_mask to the ARID, it is compared with **qos_id**. The **qos_id** would be 0b001100_0000 using the example above. Comparing the masked ID, if the result is equal to the **qos_id**, then the **QoSControl0.qos_cnt** is applied to this ARID transaction for timeout. Don't care bits must be assigned zeros.

4.7.4 DDR2/DDR3 Controller Registers

Controller control Register(CONCONTROL) **0xB030C000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ddr3_rst	odt_en	zqinit_dis	timeout_cnt											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
rd_fetch								dq_swap	ctc_rtr_gap	aref_en	out_of	clk_ratio			async

Field	Name	RW	Reset	Description
31	Reserved	-	-	
30	ddr3_rst	R/W	0x0	1'b0: Assert Reset pin of DDR3 Memory (for DDR3 reset is a active low reset). The reset signal should be asserted for atleast 200us. 1'b1: Deassert reset pin of DDR3 memory.
29	odt_en	R/W	0x0	Enable ODT operation. 1'b1: Enables ODT logic to send ODT signal to memory device. 1'b0 : Disable ODT logic
28	zqinit_dis	R/W	0x0	After reset when ZQCL is issued if this bit 0 : tZQinit will be taken care 1: tZQoper will be taken care
27-16	timeout_cnt	R/W	0xFFF	Default timeout cycles 0xn : n aclk cycles To prevent transactions in a command queue from starvation, this down counter is used. This counter starts when a new AXI transaction comes into a queue. If the counter becomes zero, the corresponding transaction becomes the highest priority command of all the transactions in the command queue. This is a default timeout counter and can be overridden by the QoS counter when the ARID matched with the QoS ID comes into the command queue.
15-12	rd_fetch	R	0x1	Read data delay cycles 0xn : n mclk cycles This register is for the unpredictable latency of read data coming from memory devices by tDQSCK variation or the board flying time. The read fetch delay of PHY read FIFO must be controlled by this parameter. The controller will fetch read data from PHY after read_latency + n mclk cycles.
11-8	Reserved	-	-	
7	dq_swap	R/W	0x0	ENDIAN
6	ctc_rtr_gap_	R/W	0x0	Read cycle gap for two different chips 0x0 : disable, 0x1 : enable To prevent collision between reads from two different memory devices, a one-cycle gap may be necessary. To insert the gap automatically for continuous reads from two different memory

				devices, enable this register
5	aref_en	R/W	0x0	Auto refresh counter 0x0 : disable, 0x1 : enable If enabled, the auto refresh counter decreases by 1 at the rising edge of the mclk
4	out_of	R/W	0x0	Out of order scheduling 0x0 : disable, 0x1 : enable The embedded scheduler enables out-of order operation to improve SDRAM utilization
3-1	clk_ratio	R/W	0x0	Clock ratio of bus clock to memory clock 0x0 : freq.(aclk) : freq.(mclk) = 1 : 1, 0x1 : Reserved 0x2 to 0x7 : Reserved.
0	async	R/W	0x0	Multi clock operation 0x0 : synchronous operation between AXI and SDRAM domain, 0x1 : asynchronous operation (TBD, Not supported in Combo DDR2 DDR3 Version 1.0, thus do not set this value in this version).

Memory Control Register(MemControl) **0xB030C004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				add_lat_pall		bl			num_chip		mem_width			mem_type	

Field	Name	RW	Reset	Description
31-12	Reserved	-	-	
11-10	add_lat_pal	R/W	0x0	Additional latency for PALL 0x0 : 0 cycle, 0x1 : 1 cycle 0x2 : 2 cycle, 0x3 : 3 cycle When all banks precharge command is issued, the latency of precharging will be tRP + add_lat_pall
9-7	bl	R/W	0x2	Memory burst length 0x0 : Reserved 0x1 : Reserved 0x2 : 4, Only for DDR2. or Fixed BC4 for DDR3 0x3 : 8, Only for DDR3. 0x4 : Reserved 0x5 : Reserved 0x6 : Reserved 0x7 : reserved
6-5	num_chip	R/W	0x2	Number of Memory chips 0x0 : 1 chip, 0x1 : 2 chips, 0x2 ~ 3xf : reserved
4-3	mem_width	R/W	0x1	Width of Memory data bus 0x0 : Reserved 0x1 : 32 bits 0x2 : Reserved 0x3 : Reserved
2	Reserved	-	-	
1-0	mem_type	R/W	0x2	Type of Memory 0x0 : DDR2 0x1 : Reserved 0x2 : DDR3 0x3 : Reserved

Memory Chip0 Configuration Register(MemConfig0) 0xB030C008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chip0_base								chip0_mask							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
chip0_map				chip0_col				chip0_row				chip0_bank			

Field	Name	RW	Reset	Description
31-24	chip0_base	R/W	0x20	AXI base address AXI base address [31:24] = chip_base, For example, if chip_base = 0x20, then AXI base address of memory chip0 becomes 0x2000_0000.
23-16	chip0_mask	R/W	0xF8	AXI base address mask Upper address bit mask to determine AXI offset address of memory chip0. 0 = corresponding address bit is not to be used for comparison 1 = corresponding address bit is to be used for comparison For example, if chip_mask = 0xF8, then AXI offset address becomes 0x0000_0000 ~ 0x07FF_FFFF. If AXI base address of memory chip0 is 0x2000_0000, then memory chip0 has an address range of 0x2000_0000 ~ 0x27FF_FFFF.
15-12	chip0_map	R/W	0x0	Address mapping method (AXI to memory) 0x0 : linear ({bank, row, column, width}), 0x1 : Interleaved ({row, bank, column, width}), 0x2 ~ 0xf : reserved
11-8	chip0_col	R/W	0x3	Number of column address bits 0x0 : 7 bits, 0x1 : 8 bits, 0x2 : 9 bits, 0x3 : 10 bits, 0x4 : 11 bits, 0x5 : 12 bits, 0x6 ~ 0xf : reserved
7-4	chip0_row	R/W	0x1	Number of row address bits 0x0 : 12 bits, 0x1 : 13 bits, 0x2 : 14 bits, 0x3 : 15 bits, 0x4: 16 bits 0x5~ 0xf : reserved
3-0	chip0_bank	R/W	0x2	Number of banks 0x0 : reserved, 0x1 : reserved, 0x2 : 4 banks, 0x3 : 8 banks, 0x4 ~ 0xf : reserved

Memory Chip1 Configuration Register(MemConfig1) 0xB030C00C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
chip1_base								chip1_mask							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
chip1_map				chip1_col				chip1_row				chip1_bank			

Field	Name	RW	Reset	Description
31-24	Chip1_base	R/W	0x28	AXI base address AXI base address [31:24] = chip_base, For example, if chip_base = 0x28, then AXI base address of chip1 becomes 0x2800_0000.
23-16	Chip1_mask	R/W	0xF8	AXI base address mask Upper address bit mask to determine AXI offset address of memory chip0. 0 = corresponding address bit is not to be used for comparison 1 = corresponding address bit is to be used for comparison For example, if chip_mask = 0xF0, then AXI offset address becomes 0x0000_0000 ~ 0x0FFF_FFFF. If AXI base address of memory chip1 is 0x2800_0000, then memory chip1 has an address range of 0x2800_0000 ~ 0x37FF_FFFF.
15-12	chip1_map	R/W	0x0	Address mapping method (AXI to memory) 0x0 : linear ({bank, row, column, width}), 0x1 : Interleaved ({row, bank, column, width}), 0x2 ~ 0xf : reserved
11-8	chip1_col	R/W	0x3	Number of column address bits 0x0 : 7 bits, 0x1 : 8 bits, 0x2 : 9 bits, 0x3 : 10 bits, 0x4 : 11 bits, 0x5 : 12 bits, 0x6 ~ 0xf : reserved
7-4	chip1_row	R/W	0x1	Number of row address bits 0x0 : 12 bits, 0x1 : 13 bits, 0x2 : 14 bits, 0x3 : 15 bits, 0x4: 16 bits 0x5~ 0xf : reserved
3-0	Chip1_bank	R/W	0x2	Number of banks 0x0 : reserved, 0x1 : reserved, 0x2 : 4 banks, 0x3 : 8 banks, 0x4 ~ 0xf : reserved

Memory Direct Command Register(DirectCmd) **0xB030C010**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reseved				dmd_type				Reserved			dmd_chip		dmd_bank		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	dmd_addr														

Field	Name	RW	Reset	Description
31-28	reseved	-	-	
27-24	dmd_type	R/W	0x0	Type of direct command 0x0 : MRS/EMRS (mode register setting), 0x1 : PALL (all banks precharge), 0x2 : PRE (per bank precharge), 0x3 : DPD (deep power down), 0x4 : REFS (self refresh), 0x5 : REFA (auto refresh), 0x6 : CKEL (active/precharge power down), 0x7 : NOP (exit from active/precharge power down or deep power down, 0x8 : REFSX (exit from self refresh) 0x9 : MRR (mode register reading), 0xA : ZQCL, 0xB : ZQCS. 0xC ~ 0xf :reserved When a direct command is issued, AXI masters must not access memory. And dynamic power down, dynamic self refresh and force precharge function (MemControl register) must be disabled. MRS/EMRS and MRR commands should be issued when all banks are idle state.
23-21	Reserved	-	-	
20	dmd_chip	R/W	0x0	The chip number to send the direct command to 0 : chip 0 1 : chip 1
18-16	dmd_bank	R/W	0x0	Related bank address when issuing a direct command When a direct command is sent to a chip, additional information such as the bank address could be needed. This register is used in such situations..
15-0	dmd_addr	R/W	0x0	Related address value when issuing a direct command When a direct command is sent to a chip, additional information such as the address could be needed. This register is used in such situations.

Precharge Policy Configuration Register(PrechConfig) **0xB030C014**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
fp_cnt								Reserved							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
chip1_policy								chip0_policy							

Field	Name	RW	Reset	Description
31-24	fp_cnt	R/W	0xFF	Force precharge cycles 0xn : n mclk cycles, If the force precharge function (MemControl.fp_en) is enabled and the force precharge counter becomes zero, the controller forces the activated memory bank into the precharged state
23-16	Reserved	-	-	
15-8	chip1_policy	R/W	0x0	memory chip1 precharge bank selective policy 0x0: open page policy, 0x1: close page (auto precharge) policy chip1_policy[n], n is the bank number of chip1. Open Page Policy: After a READ or WRITE, the row that was accessed is left open. Close Page (Auto Precharge) Policy: Right after a READ or WRITE command, memory devices automatically performs precharging of the bank. This is a bank selective precharge policy. For example, if chip1_policy[2] is 0x0, bank2 of chip1 has an open page policy and if chip1_policy[6] is 0x1, bank6 of chip1 has a close page policy..
7-0	chip0_policy	R/W	0x0	memory chip0 precharge bank selective policy 0x0: open page policy, 0x1: close page (auto precharge) policy Chip0_policy[n], n is the bank number of chip0. This is for memory chip0.

Memory Power Control Register(MEMPWD) 0xB030C018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
										dsref_en	fp_en	dpwrdn_type		dpwrdn_en	clk_stop_en

Field	Name	RW	Reset	Description
31-6	Reserved	-	-	
5	dsref_en	R/W	0x0	Dynamic self refresh 0x0 : disable, 0x1 : enable
4	fp_en	R/W	0x0	Force precharge 0x0 : disable, 0x1 : enable Enabling fp_en will automatically precharge an open bank after a specified amount of mclk cycles (if no access has been made in between the cycles) in an open page policy. Setting PrechConfig.fp_cnt bit-field will specify the amount of mclk cycles to wait until force precharge precharges the open bank.
3-2	dpwrdn_type	R/W	0x0	Type of dynamic power down 0x0 : active/precharge power down, 0x1 : force precharge power down 0x2 ~ 0x3 : reserved
1	dpwrdn_en	R/W	0x0	Dynamic power down 0x0 : disable, 0x1 : enable
0	clk_stop_en	R/W	0x0	Dynamic clock control 0x0 : always running, 0x1 : stopping during idle periods

Dynamic Power Down Configuration Register(PwrdnConfig) 0xB030C01C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
dsref_cyc															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								dpwrdn_cyc							

Field	Name	RW	Reset	Description
31-16	dsref_cyc	R/W	0xFFFF	Number of cycles for dynamic self refresh entry 0xn : n aclk cycles, If the command queue has been empty for n+1 cycles, the controller will force the memory device into self refresh state
15-8	Reserved	-	-	
7-0	dpwrdn_cyc	R/W	0xFF	Number of cycles for dynamic power down entry 0xn : n aclk cycles, If the command queue has been empty for n+1 cycles, the controller will force the memory device into active/precharge power down state

Auto Refresh Period(TimingAref) 0xB0305100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
t_refi															

Field	Name	RW	Reset	Description
31-16	Reserved	-	-	
15-0	t_refi	R/W	0x40E	Average periodic refresh interval Should be min. memory tREFI (all bank) < t_refi * T(mclk), for example, for the all bank refresh period of 7.8us, and an mclk frequency of 133MHz, the following value should be programmed into it : 7.8 us * 133 MHz = 1038

Auto Refresh to other valid command delay 0xB030C104

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								t_rfc							

Field	Name	RW	Reset	Description
31-8	Reserved	-	-	
7-0	t_rfc	R/W	0xF	Auto refresh to Active / Auto refresh command period, in cycles Should be memory min. tRFC < t_rfc * T(mclk)

Bank to Bank Activate delay 0xB030C108

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												t_rrd			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	t_rrd	R/W	0x2	Active bank A to Active bank B delay, in cycles Should be memory min. tRRD < t_rrd * T(mclk)

Row Precharge Delay 0xB030C10C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												t_rp			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	t_rp	R/W	0x3	Precharge command period, in cycles Should be memory min. tRP < t_rp * T(mclk),

Activate to Read or Write Delay 0xB030C110

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
												t_rcd			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	t_rcd	R/W	0x3	Active to Read or Write delay, in cycles Should be memory min. tRCD < t_rcd * T(mclk)

Same Bank Activate to Activate Delay 0xB030C114

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
										t_rc					

Field	Name	RW	Reset	Description
31-6	Reserved	-	-	
5-0	t_rc	R/W	0xA	Active to Active period, in cycles Should be memory min. tRC < t_rc * T(mclk)

Row Open Period 0xB030C118

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
										t_ras					

Field	Name	RW	Reset	Description
31-6	Reserved	-	-	
5-0	t_ras	R/W	0x6	Active to Precharge command period, in cycles Should be memory min. t_ras * T(mclk) < tRAS

Write to Read Delay 0xB030C11C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
												t_wtr			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	t_wtr	R/W	0x1	Internal write to Read command delay, in cycles Should be memory min. tWTR < t_wtr * T(mclk)

Write Recovery Time 0xB030C120

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
												t_wr			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	t_wr	R/W	0x2	Write recovery time, in cycles Should be memory min. tWR < t_wr * T(mclk)

Read to Precharge Delay 0xB030C124

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
												t_rtp			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	t_rtp	R/W	0x1	Internal read to Precharge command delay, in cycles Should be memory min. tRTP < t_rtp * T(mclk)

CAS Latency 0xB030C128

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
												cl			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	cl	R/W	0x3	CAS Latency in memory clock cycles Should be memory CL = cl

Write Latency 0xB030C12C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
												wl			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	wl	R/W	0x2	Write data latency in cycles Should be memory Write latency = wl

Read Latency 0xB030C130

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
												rl			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	rl	R/W	0x4	Read data latency in cycles Should be memory Read latency = rl

Quadruple Row Activate Command Delay 0xB030C134

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
										t_faw					

Field	Name	RW	Reset	Description
31-6	Reserved	-	-	
5-0	t_faw	R/W	0x1E	Four Active Window Should be min. tFAW < t_faw * T(mclk)

Self Refresh Exit Delay 0xB030C138

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						t_xsr									

Field	Name	RW	Reset	Description
31-10	Reserved	-	-	
9-0	t_xsr	R/W	0x1B	Self refresh exit power down to next valid command delay, in cycles Should be min. tXSR < t_xsr * T(mclk)

Power Down Exit Delay 0xB030C13C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						t_xp									

Field	Name	RW	Reset	Description
31-10	Reserved	-	-	
9-0	t_xp	R/W	0x4	Exit power down to next valid command delay, in cycles Should be min. tXP < t_xp * T(mclk)

Minimum Power Down period 0xB030C140

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												t_cke			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	t_cke	R/W	0x2	CKE minimum pulse width (minimum power down mode duration), in cycles Should be min. tCKE < t_cke * T(mclk)

Mode Register Set Delay. 0xB030C144

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												t_mrd			

Field	Name	RW	Reset	Description
31-4	Reserved	-	-	
3-0	t_mrd	R/W	0x2	Mode Register Set command period, in cycles Should be min. tMRD < t_mrd * T(mclk)

ZQ calibration time after Reset or Power up. 0xB030C148

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						t_zqinit									

Field	Name	RW	Reset	Description
31-10	Reserved	-	-	
9-0	t_zqinit	R/W	0x201	Time delay required for ZQ calibration after power up or Reset Should be min. 512 cycles

Full ZQ calibration delay. 0xB030C14C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						t_zqoper									

Field	Name	RW	Reset	Description
31-10	Reserved	-	-	
9-0	t_zqoper	R/W	0x101	Time delay required for full ZQ calibration during normal operation. Should be min. 256 cycles

Short ZQ calibration time. 0xB030C150

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						t_zqcs									

Field	Name	RW	Reset	Description
31-10	Reserved	-	-	
9-0	t_zqcs	R/W	0x041	Time delay required for short ZQ calibration during normal operation. Should be min. 64 cycles

AXI Transaction Status Register 0xB030C200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
														chip1_empty	chip0_empty

Field	Name	RW	Reset	Description
31-2	Reserved	-	-	
1	chip1_empty	R	0x1	Command queue status of chip1 0x0 : not empty, 0x1 : empty There are no AXI transaction corresponding to chip1 memory in the command queue entries
0	chip0_empty	R	0x1	Command queue status of chip0 0x0 : not empty, 0x1 : empty There are no AXI transaction corresponding to chip0 memory in the command queue entries

Auto Refresh Counter Status 0xB030C204

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
aref_cnt															

Field	Name	RW	Reset	Description
31-16	Reserved	-	-	
15-0	aref_cnt	R	0x0	The current value of auto refresh counter Shows the current value of the all bank auto refresh counter. This will be updated when a new t_refi is programmed into the TimingAref register and decreases by 1 at the rising edge of mclk. An all bank auto refresh command will be issued to memory device and this counter will be

Memory Chip0 Status Register(Chip0Status) 0xB030C208

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bank7_state				bank6_state				bank5_state				bank4_state			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
bank3_state				bank2_state				bank1_state				bank0_state			

Field	Name	RW	Reset	Description
31-28	state	R	0x0	**Current State of Bank 7 of Memory Chip0**
27-24	bank6_state	R	0x0	**Current State of Bank 6 of Memory Chip0**
23-20	bank5_state	R	0x0	**Current State of Bank 5 of Memory Chip0**
19-16	Bank4_state	R	0x0	**Current State of Bank 4 of Memory Chip0**
15-12	Bank3_state	R	0x0	**Current State of Bank 3 of Memory Chip0**
11-8	bank2_state	R	0x0	**Current State of Bank 2 of Memory Chip0**
7-4	bank1_state	R	0x0	**Current State of Bank 1 of Memory Chip0**
3-0	bank0_state	R	0x0	**Current State of Bank 0 of Memory Chip0**

Note: Encoding for 4 Bits of the States
0x0 : idle (precharged),
0x1 : MRS/EMRS,
0x2 : deep power down,
0x3 : self refresh,
0x4 : auto refresh,
0x5 : precharge power down,
0x6 : row active,
0x7 : active power down,
0x8 : write,
0x9 : write with auto precharge,
0xA : read,
0xB : read with auto precharge,
0xC : burst stop,
0xD : precharging,
0xE : MRR,
0xF: ZQCL/ZQCS

Memory Chip1 Status Register(Chip1Status) 0xB030520C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
bank7_state				bank6_state				bank5_state				bank4_state			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
bank3_state				bank2_state				bank1_state				bank0_state			

Field	Name	RW	Reset	Description
31-28	state	R	0x0	**Current State of Bank 7 of Memory Chip0**
27-24	bank6_state	R	0x0	**Current State of Bank 6 of Memory Chip1**
23-20	bank5_state	R	0x0	**Current State of Bank 5 of Memory Chip1**
19-16	Bank4_state	R	0x0	**Current State of Bank 4 of Memory Chip1**
15-12	Bank3_state	R	0x0	**Current State of Bank 3 of Memory Chip1**
11-8	bank2_state	R	0x0	**Current State of Bank 2 of Memory Chip1**
7-4	bank1_state	R	0x0	**Current State of Bank 1 of Memory Chip1**
3-0	bank0_state	R	0x0	**Current State of Bank 0 of Memory Chip1**

Note: Encoding for 4 Bits of the Bank States
0x0 : idle (precharged),
0x1 : MRS/EMRS,
0x2 : deep power down,
0x3 : self refresh,
0x4 : auto refresh,
0x5 : precharge power down,
0x6 : row active,
0x7 : active power down,
0x8 : write,
0x9 : write with auto precharge,
0xA : read,
0xB : read with auto precharge,
0xC : burst stop,
0xD : precharging,
0xE : MRR,
0xF: ZQCL/ZQCS

Memory Mode Registers Status Register(MrStatus) 0xB030C210

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								mr_status							

Field	Name	RW	Reset	Description
31-8	Reserved	-	-	
7-0	mr_status	R	0x0	Mode registers status

Quality of Service Control Register 0(QoSControl0) 0xB0305300

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				qos_cnt0											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
															qos_en0

Field	Name	RW	Reset	Description
31-28	Reserved	-	-	
27-16	qos_cnt0	R/W	0xFF	QoS cycles 0xn : n aclk cycles The matched ARID will use this value for its timeout counter instead of ConControl.timeout_cnt.
0	qos_en0	R/W	0x0	QoS enable 0x0 : disable, 0x1 : enable If this function is enabled, its timeout counter works and the ARID is masked with QoSConfig.qos_mask and compared with QoSConfig.qos_id

Quality of Service Configuration Register 0 (QoSConfig0) 0xB0305304

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qos_mask0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
qos_id0															

Field	Name	RW	Reset	Description
31-16	qos_mask0	R/W	0x0	QoS mask bits A mask to be used to mask the incoming ARID to compare with the qos_id. For example, to have 0b00110XX000 IDs the same QoS, the 4th and 5th bits need to be masked. Therefore, qos_mask would be 0b1111100111
15-0	qos_id0	R/W	0x0	QoS ID A QoS ID to be compared with the masked ARID to check whether its timeout counter should be used for QoS. After applying the qos_mask to the ARID, it is compared to qos_id. The qos_id would be 0b001100_0000 using the example above. Comparing the masked ID, if the result is equal to the qos_id, then the QoSControl0.qos_cnt is applied to this ARID transaction for timeout.

Quality of Service Control Register 1(QoSControl1) 0xB0305308

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				qos_cnt1											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															qos_en1

Field	Name	RW	Reset	Description
31-28	Reserved	-	-	
27-16	qos_cnt1	R/W	0xFF	QoS cycles 0xn : n aclk cycles
0	qos_en1	R/W	0x0	QoS enable 0x0 : disable, 0x1 : enable

Quality of Service Configuration Register 1 (QoSConfig1) 0xB030530C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qos_mask1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
qos_id1															

Field	Name	RW	Reset	Description
31-16	qos_mask1	R/W	0x0	QoS mask bits
15-0	qos_id1	R/W	0x0	QoS ID

Quality of Service Control Register 2(QoSControl2) 0xB0305310

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				qos_cnt2											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															qos_en2

Field	Name	RW	Reset	Description
31-28	Reserved	-	-	
27-16	qos_cnt2	R/W	0xFF	QoS cycles 0xn : n aclk cycles
0	qos_en2	R/W	0x0	QoS enable 0x0 : disable, 0x1 : enable

Quality of Service Configuration Register 2 (QoSConfig2) 0xB0305314

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qos_mask2															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
qos_id2															

Field	Name	RW	Reset	Description
31-16	qos_mask2	R/W	0x0	QoS mask bits
15-0	qos_id2	R/W	0x0	QoS ID

Quality of Service Control Register 3(QoSControl3) 0xB0305318

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				qos_cnt3											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															qos_en3

Field	Name	RW	Reset	Description
31-28	Reserved	-	-	
27-16	qos_cnt3	R/W	0xFF	QoS cycles 0xn : n aclk cycles
0	qos_en3	R/W	0x0	QoS enable 0x0 : disable, 0x1 : enable

Quality of Service Configuration Register 3 (QoSConfig3) 0xB030531C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qos_mask3															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
qos_id3															

Field	Name	RW	Reset	Description
31-16	qos_mask3	R/W	0x0	QoS mask bits
15-0	qos_id3	R/W	0x0	QoS ID

Quality of Service Control Register 4(QoSControl4) 0xB0305320

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				qos_cnt4											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															qos_en4

Field	Name	RW	Reset	Description
31-28	Reserved	-	-	
27-16	qos_cnt4	R/W	0xFF	QoS cycles 0xn : n aclk cycles
0	qos_en4	R/W	0x0	QoS enable 0x0 : disable, 0x1 : enable

Quality of Service Configuration Register 4 (QoSConfig4) 0xB0305324

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qos_mask4															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
qos_id4															

Field	Name	RW	Reset	Description
31-16	qos_mask4	R/W	0x0	QoS mask bits
15-0	qos_id4	R/W	0x0	QoS ID

Quality of Service Control Register 5(QoSControl5) 0xB0305328

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				qos_cnt5											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															qos_en5

Field	Name	RW	Reset	Description
31-28	Reserved	-	-	
27-16	qos_cnt5	R/W	0xFF	QoS cycles 0xn : n aclk cycles
0	qos_en5	R/W	0x0	QoS enable 0x0 : disable, 0x1 : enable

Quality of Service Configuration Register 5 (QoSConfig5) 0xB030532C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qos_mask5															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
qos_id5															

Field	Name	RW	Reset	Description
31-16	qos_mask5	R/W	0x0	QoS mask bits
15-0	qos_id5	R/W	0x0	QoS ID

Quality of Service Control Register 6(QoSControl6) 0xB0305330

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				qos_cnt6											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															qos_en6

Field	Name	RW	Reset	Description
31-28	Reserved	-	-	
27-16	qos_cnt6	R/W	0xFF	QoS cycles 0xn : n aclk cycles
0	qos_en6	R/W	0x0	QoS enable 0x0 : disable, 0x1 : enable

Quality of Service Configuration Register 6 (QoSConfig6) 0xB0305334

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qos_mask6															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
qos_id6															

Field	Name	RW	Reset	Description
31-16	qos_mask6	R/W	0x0	QoS mask bits
15-0	qos_id6	R/W	0x0	QoS ID

Quality of Service Control Register 7(QoSControl7) 0xB0305338

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qos_cnt7															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															qos_en7

Field	Name	RW	Reset	Description
31-28	Reserved	-	-	
27-16	qos_cnt7	R/W	0xFF	QoS cycles 0xn : n aclk cycles
0	qos_en7	R/W	0x0	QoS enable 0x0 : disable, 0x1 : enable

Quality of Service Configuration Register 7 (QoSConfig7) 0xB030533C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
qos_mask7															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
qos_id7															

Field	Name	RW	Reset	Description
31-16	qos_mask7	R/W	0x0	QoS mask bits
15-0	qos_id7	R/W	0x0	QoS ID

PHY Control0 Registger(PhyControl0) 0xB030C400

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ctrl_force								ctrl_inc							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ctrl_start_point								ctrl_fnc_fb				ctrl_dfdqs	ctrl_half	ctrl_dll_on	ctrl_start

Field	Name	RW	Reset	Description
31-24	ctrl_force	R/W	0x0	DLL force delay This field is used instead of ctrl_lock_value[9:2] to be found by the DLL only when ctrl_dll_on is LOW. (i.e. If the DLL is off, this field is used to generate 270' clock and shift DQS by 90'.)
23-16	ctrl_inc	R/W	0x10	DLL delay increment Set this to 8'h10
15-8	ctrl_start_point	R/W	0x0	DLL lock start point Initial DLL lock start point. This is the number of delay cells and is the start point where "DLL" start tracing to be locked. Initial delay time is calculated by multiplying the unit delay of delay cell and this value.
7-5	ctrl_fnc_fb	R/W	0x0	Valid only when {mode_phy, mode_nand, mode_scan, mode_mux} is 4'b00000. For normal operation 3'b000 : Normal operation mode. For ATE test purpose 3'b010 : External FNC read feedback test mode. 3'b011 : Internal FNC read feedback test mode. For Board test purpose 3'b100 : External PHY read feedback test mode. When memory is not attached on the board

				3'b101 : Internal PHY read feedback test mode. mode_highz should be set. 3'b110 : Internal PHY write feedback test mode. mode_highz should be set.
4	reserved	-	-	
3	ctrl_dfdqs	R/W	0x0	Differential DQS 1'b0 : single-ended DQS 1'b1 : differential DQS
2	ctrl_half	R/W	0x0	DLL low speed HIGH active signal to turn on the low speed mode for DLL. If this bit is set, DLL can run at low speed(< 333MHz).
1	ctrl_dll_on	R/W	0x0	DLL on HIGH active start signal to turn on the DLL. This signal should be kept HIGH for normal operation. If this signal becomes LOW , DLL is turned off and ctrl_clock and ctrl_flock become HIGH. This bit should be kept set before ctrl_start is set to turn on the DLL.
0	ctrl_start	R/W	0x0	HIGH active start signal to make the DLL run and lock. This signal should be kept HIGH during normal operation. If this signal becomes LOW , DLL stops running. To re-run DLL, make this signal HIGH again. In the case of rerunning, DLL loses previous lock information. Before ctrl_start is set, be sure that ctrl_dll_on is HIGH.

PHY Control1 Registger(PhyControl1) 0xB030C404

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									ctrl_offsetd						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ctrl_offsetc								ctrl_ref				fp_resync	ctrl_shiftc		

Field	Name	RW	Reset	Description
31-24	Reserved	-	-	
22-16	ctrl_offsetd	R/W	0x0	This field is for debug purpose. (For LPDDR2) If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. offset amount for 270°clock generation: ctrl_offsetd[6] = 1 : (tFS : fine step delay) 270° delay amount − ctrl_offsetd[5:0] x tFS ctrl_offsetd[6] = 0 : 270° delay amount + ctrl_offsetd[5:0] x tFS
15-8	ctrl_offsetc	R/W	0x0	Gate offset amount for DDR. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. ctrl_offsetc [6] = 1 : (tFS : fine step delay) GATEout delay amount − ctrl_offsetc [5:0] x tFS ctrl_offsetc [6] = 0 : GATEout delay amount + ctrl_offsetc [5:0] x tFS

7-4	ctrl_ref	R/W	0x0	This field determines the period of time when ctrl_locked is cleared. 4'b0000 : Don't use. 4'b0001 : ctrl_flock is de-asserted during 6 clock cycles, ctrl_locked is deasserted. 4'b0010 : ctrl_flock is de-asserted during 9 clock cycles, ctrl_locked is deasserted. ~ 4'b1111 : ctrl_flock is de-asserted during 48 clock cycles, ctrl_locked is deasserted.
3	fp_resync	R/W	0x0	Active RISIG-EDGE signal. This signal should become LOW after set HIGH for normal operation. When this bit transits from LOW to HIGH, pointers of FIFO and DLL information is updated. In general, this bit should be set during initialization and refresh cycles. Duration of ctrl_resync should be 1 to 2 cycles. Before set and clear this signal, ctrl_clock or ctrl_flock should be checked during initialization. During memory access, ctrl_clock or ctrl_flock doesn't need to be checked. ctrl_resync should be set and cleared only when ctrl_clock is set after auto-refresh is started. Refer to the application note for ctrl_resync timing.
2-0	ctrl_shiftc	R/W	0x0	GATEout signal delay amount for DDR. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. 000 : T/128(2.8125° shift), 001 : T/64(5.625° shift) 010 : T/32(11.25° shift) , 011 : T/16(22.5° shift) 100 : T/8(45° shift) , 101 : T/4(90° shift) 110 : T/2(180° shift) , 111 : T(360° shift) Should be 3'b110 for DDR2/DDR3.

PHY Control2 Registger(PhyControl2) 0xB030C408

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ctrl_offsetr3								ctrl_offsetr2						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ctrl_offsetr1								ctrl_offsetr0						

Field	Name	RW	Reset	Description
31	Reserved	-	-	-
30-24	ctrl_offsetr3	R/W	0x0	-
23	Reserved	-	-	-
22-16	ctrl_offsetr2	R/W	0x0	This field is for debug purpose. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. rd_slice_2 offset amount : ctrl_offset2[6] = 1 : 90' delay amount - ctrl_offset0[5:0] ctrl_offset2[6] = 0 : 90' delay amount + ctrl_offset0[5:0]
15	Reserved	-	-	-
14-8	ctrl_offsetr1	R/W	0x0	This field is for debug purpose. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. rd_slice_1 offset amount : ctrl_offset1[6] = 1 90' delay amount - ctrl_offset0[5:0] ctrl_offset1[6] = 0 : 90' delay amount + ctrl_offset0[5:0]
7	Reserved	-	-	-
6-0	ctrl_offsetr0	R/W	0x0	This field is for debug purpose. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. rd_slice_0 offset amount : ctrl_offset0[6] = 1 : 90' delay amount - ctrl_offset0[5:0] ctrl_offset0[6] = 0 : 90' delay amount + ctrl_offset0[5:0] 0x100 : T/8 (45' shift), 0x101 : T/4 (90' shift), 0x110 : T/2 (180' shift), 0x111 : T (360' shift)

PHY Control3 Registger(PhyControl3) 0xB030C40C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	ctrl_offsetw3								ctrl_offsetw2						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ctrl_offsetw1								ctrl_offsetw0						

Field	Name	RW	Reset	Description
31	Reserved	-	-	-
30-24	ctrl_offsetw3	R/W	0x0	This field is for debug purpose. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. rd_slice_3 offset amount : ctrl_offset3[6] = 1 : 90' delay amount - ctrl_offset0[5:0] ctrl_offset3[6] = 0 : 90' delay amount + ctrl_offset0[5:0]
23	Reserved	-	-	-
22-16	ctrl_offsetw2	R/W	0x0	This field is for debug purpose. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. rd_slice_2 offset amount : ctrl_offset2[6] = 1 : 90' delay amount - ctrl_offset0[5:0] ctrl_offset2[6] = 0 : 90' delay amount + ctrl_offset0[5:0]
15	Reserved	-	-	-
14-8	ctrl_offsetw1	R/W	0x0	This field is for debug purpose. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. rd_slice_1 offset amount : ctrl_offset1[6] = 1 : 90' delay amount - ctrl_offset0[5:0] ctrl_offset1[6] = 0 : 90' delay amount + ctrl_offset0[5:0]
7	Reserved	-	-	-
6-0	ctrl_offsetw0	R/W	0x0	This field is for debug purpose. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. rd_slice_0 offset amount : ctrl_offset0[6] = 1 : 90' delay amount - ctrl_offset0[5:0] ctrl_offset0[6] = 0 : 90' delay amount + ctrl_offset0[5:0] 0x100 : T/8 (45' shift), 0x101 : T/4 (90' shift), 0x110 : T/2 (180' shift), 0x111 : T (360' shift)

PHY Control4 Registger(PhyControl4) 0xB030C410

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		ctrl_ddr3									ctrl_dcc				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ctrl_dcc							ctrl_pd[4:0](32b), ctrl_pd[2:0](16b)					ctrl_ckdis	ctrl_shgate	ctrl_read_width	ctrl_hcke

Field	Name	RW	Reset	Description
31-30	Reserved	-	-	-
29	ctrl_ddr3	R/W	0x0	1'b0 : DDR2 and LPDDR2 1'b1 : DDR3
28-21	Reserved	-	-	-
20-9	ctrl_dcc	R/W	0x0	This field controls the Clock Duty in the PHY. DCC (Duty Correction Circuit) can shift clock negative edge from the original position to the leaded or lagged position in compensation for the process variations. The default value should be 12'b111_000_111_000. The following value shows the delay value at the best condition. The maximum value which can be shifted by two pairs of DCC will be about ±60ps. ctrl_dcc[2:0] = 3'b000 : The default value ctrl_dcc[2:0] = 3'b001 : The negative edge is lagged by 20ps. ctrl_dcc[2:0] = 3'b010 : The negative edge is lagged by 26ps. ctrl_dcc[2:0] = 3'b011 : The negative edge is lagged by 29ps. ctrl_dcc[2:0] = 3'b100 : The negative edge is lagged by 31ps. ctrl_dcc[2:0] = 3'b101 : The negative edge is lagged by 33ps. ctrl_dcc[2:0] = 3'b110 : The negative edge is lagged by 34ps. ctrl_dcc[2:0] = 3'b111 : The negative edge is lagged by 35ps. ctrl_dcc[5:3] = 3'b111 : The default value ctrl_dcc[5:3] = 3'b110 : The negative edge is leaded by 20ps. ctrl_dcc[5:3] = 3'b110 : The negative edge is leaded by 26ps. ctrl_dcc[5:3] = 3'b100 : The negative edge is leaded by 29ps. ctrl_dcc[5:3] = 3'b011 : The negative edge is leaded by 31ps. ctrl_dcc[5:3] = 3'b010 : The negative edge is leaded by 33ps. ctrl_dcc[5:3] = 3'b001 : The negative edge is leaded by 34ps. ctrl_dcc[5:3] = 3'b000 : The negative edge is leaded by 35ps. ctrl_dcc[8:6] : please refer to ctrl_dcc[2:0] ctrl_dcc[11:9] : please refer to ctrl_dcc[5:3]
8-4	ctrl_pd[4:0](32b) ctrl_pd[2:0](16b)	R/W	0x0	This field controls the input buffer of I/O If this field is set, input buffer is turned-off for power down This field should be 0 for normal operation. ctrl_pd[3:0] : for each data slice/ctrl_pd[4] : for control slice
3	ctrl_ckdis	R/W	0x0	This field controls the CK/CKB 1'b0 : Clock output is enabled

				1’b1 : Clock output is disabled
2	ctrl_shgate	R/W	0x0	This field controls the gate control signal 1’b0 : gate signal length = “burst length / 2” (for LPDDR2) 1’b1 : gate signal length = “burst length / 2” – 1 (for DDR2/DDR3)
1	ctrl_read_width	R/W	0x0	1’b0 : Termination on period is (BL/2+1.5) cycle, DDR2 and DDR3 1’b1 : Termination on period is (BL/2+1) cycle, DDR2 under 333MHz
0	ctrl_hcke	R/W	0x0	This signal decides the reset value of CKE and some signals. If this bit is set, reset value of io_adct_out[19] and io_cke_out is 1. Otherwise, reset value of them is 0.

PHY Control5 Registger(PhyControl5) **0xB030C414**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
														ctrl_zq_mode_dds	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ctrl_zq_mode_dds	ctrl_zq_mode_noterm		ctrl_zq_mode_term		ctrl_zq_force_impn			ctrl_zq_force_impp			ctrl_zq_forc	ctrl_zq_start	ctrl_zq_div		

Field	Name	RW	Reset	Description
31-18	Reserved	-	-	-
17-15	ctrl_zq_mode_dds	R/W	0x0	Driver strength selection.
14	ctrl_zq_mod e_noterm	R/W	0x0	Terminations disable selection. 1 : termination disable. 0 : termination enable. DDR : 1'b1 DDR2 : 1'b0(recommended) or 1'b1(when termination is not used) DDR3/gDDR3 : 1'b0
13-11	ctrl_zq_mode_term	R/W	0x0	On-die-termination resistor value selection.
10-8	ctrl_zq_force_impn	R/W	0x0	Immediate control code for pull-down
7-5	ctrl_zq_force_impp	R/W	0x0	Immediate control code for pull-up.
4	ctrl_zq_forc	R/W	0x0	If this field is set, ctrl_force_impp[2:0]/impn[2:0] are used instead of calibration control code found after auto calibration
3	ctrl_zq_start	R/W	0x0	Auto calibration start signal.
2-0	ctrl_zq_div	R/W	0x0	Calibration I/O clock selection signasl. 3'b000 : system clock / 2 3'b001 : system clock / 4 3'b010 : system clock / 8 3'b011 : system clock / 16 (recommend value for 400MHz) 3'b100 : system clock / 32 Don't use the other setting.

PHY Status Register 0(PhyStatus0) 0xB030C418

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
			ctrl_lock_value										ctrl_locked	ctrl_flock	ctrl_clock

Field	Name	RW	Reset	Description
31-13	Reserved	-	-	
12-4	ctrl_lock_value	R	0x0	Locked Delay Locked delay line encoding value ctrl_lock_value[9:2]: number of delay cells for coarse lock ctrl_lock_value[1:0]: control value for fine lock
3	Reserved	-	-	
2	ctrl_locked	R	0x0	DLL lock 0 : DLL is unlocked, 1 : DLL is locked
1	ctrl_flock	R	0x0	Fine lock information
0	ctrl_clock	R	0x0	Coarse lock information

PHY Status Register 1 (PhyStatus1) 0xB030C41C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved															

PHY Status Register 2(PhyStatus2) 0xB030C420

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								ctrl_zq_nmon			ctrl_zq_pmon			ctrl_zq_error	ctrl_zq_end

Field	Name	RW	Reset	Description
31-8	Reserved	-	-	
7-5	ctrl_zq_nmon	R	0x0	Control code found by auto calibration for pull-down.
4-2	ctrl_zq_pmon	R	0x0	Control code found by auto calibration for pull-up.
1	ctrl_zq_error	R	0x0	DLL lock 0 : DLL is unlocked, 1 : DLL is locked
0	ctrl_zq_end	R	0x0	Status indicating calibration completion without error.

PHY Test Register 0 (PhyTest0) 0xB030C424

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ctrl_fb_cnt4											ctrl_fb_oky				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			ctrl_fb_error								ctrl_fb_start				

Field	Name	RW	Reset	Description
31-24	ctrl_fb_cnt4	R	0x0	Test count of control channel.
23-21	Reserved	-	-	
20-16	ctrl_fb_oky	R/W	0x0	Feedback test completion without error for each slice. ctrl_fb_start[0] : okay of data channel0. ctrl_fb_start[1] : okay of data channel1. ctrl_fb_start[2] : okay of data channel2. okay of control channel for 16-bit PHY. ctrl_fb_start[3] : okay of data channel3 for 32-bit PHY. ctrl_fb_start[4] : okay of control channel for 32-bit PHY.
15-13	Reserved	-	-	
12-8	ctrl_fb_error	R	0x0	Feedback test stop with error for each slice. ctrl_fb_err[0] : error of data channel0. ctrl_fb_err[1] : error of data channel1. ctrl_fb_err[2] : error of data channel2. error of control for 16-bit PHY. ctrl_fb_err[3] : error of data channel3 for 32-bit PHY. ctrl_fb_err[4] : error of control channel for 32-bit PHY.
7-5	Reserved	-	-	
4-0	ctrl_fb_start	R/W	0x0	Feedback test start signal for each slice. ctrl_fb_start[0] : start of data channel0 ctrl_fb_start[1] : start of data channel1 ctrl_fb_start[2] : start of data channel2 for 32-bit PHY start of control channel for 16-bit PHY ctrl_fb_start[3] : start of data channel3 for 32-bit PHY. ctrl_fb_start[4] : start of control channel for 32-bit PHY. For this test, mode_highz should be set when memory is on the board

PHY Test Register 1 (PhyTest1) 0xB030C428

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ctrl_fb_cnt3								ctrl_fb_cnt2							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ctrl_fb_cnt1								ctrl_fb_cnt0							

Field	Name	RW	Reset	Description
31-24	ctrl_fb_cnt3	R	0x0	Test count of data channel3 for 32-bit PHY
23-16	ctrl_fb_cnt2	R	0x0	Test count of data channel2 for 32-bit PHY. Test count of control channel for 16-bit PHY.
15-8	ctrl_fb_cnt1	R	0x0	Test count of data channel1..
7-0	ctrl_fb_cnt0	R	0x0	Test count of data channel0.

4.7.5 Miscellaneous Configuration Registers

LPDDR SDRAM Controller Configuration Register 0 (M0CFG0) **0xB0303000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SYNC	RDW		TYPE			WIDTH		SYNC_OPT	SLOW						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
QOS_OVR															

Field	Name	RW	Reset	Description
31	SYNC	R/W	1	When HIGH, indicates aclk is synchronous to mclk. Otherwise, they are asynchronous. * Should be “1”
30-29	RDW	R/W	0	This value varies the delay permitted before the capture of read data into the memory clock domain from the memory device.
28-26	TYPE	R/W	0	Selects the memory type of the current configuration. Not all values are valid for a single configuration. You can override this value using the APB interface.
25-24	WIDTH	R/W	0	These configure the external memory width. Note: You can use each memory configuration at half of its memory width, by setting the memory_width tie-off, provided that: • the new memory width is not less than 16 bits • the effective memory width is not less than half the AXI interface width.
23	SYNC_OPT	R/W	0	Synchronizing option 0: two step synchronizer 1: one step synchronizer
22	SLOW	R/W	1	Additional Register Slice Insertion 0 : Slice inserted makes additional wait. 1 : Slice inserted will be bypassed
21-16	-	-	-	Reserved
15-0	QOS_OVR	R/W	0	When one or more bits are HIGH, coincident with arvalid and arready, and when the arid match bits are equivalent to the qos_override bit(s), then the QoS for the read access is forced to minimum latency.

LPDDR SDRAM Controller Configuration Register 1 (M0CFG1) 0xB0303004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									RST_BYPASS	EBIOFF	EBIGNT	EBIUSE	DQM_INIT	CKE_INIT	A_GT_M_SYNC

Field	Name	RW	Reset	Description
6	RST_BYPASS	R/W	0	This signal is used for ATPG testing only.
5	EBIOFF	R/W	0	External memory bus access backoff. The EBI backoff signal goes active HIGH when the EBI wants to remove the memory controller from the memory bus so that another memory controller can be granted the memory bus.
4	EBIGNT	R/W	0	External memory bus grant. The EBI grant signal goes HIGH when the EBI grants the external memory bus.
3	EBIUSE	R/W	0	Use EBI
2	DQM_INIT	R/W	0	The dqm output ports to the external memory reset to this value.
1	CKE_INIT	R/W	0	The cke output port to the external memory resets to this value.
0	A_GT_M_SYNC	R/W	0	When HIGH, indicates aclk is greater than mclk but is still synchronous.

DDR2 SDRAM Controller Configuration Register 0 (M1CFG0) 0xB0303010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SYNC	BANK					WIDTH		SYNC_OPT							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QOS_OVR															

Field	Name	RW	Reset	Description
31	SYNC	R/W	1	When HIGH, indicates aclk is synchronous to mclk. Otherwise, they are asynchronous. * Should be “1”
30-29	BANK	R/W	0	Encodes number of bits of AXI address that comprise the bank address.
28-26	-	-	-	Reserved
25-24	WIDTH	R/W	0	These configure the external memory width. Note: You can use each memory configuration at half of its memory width, by setting the memory_width tie-off, provided that: • the new memory width is not less than 16 bits • the effective memory width is not less than half the AXI interface width.
23	SYNC_OPT	R/W	0	Synchronizing option 0: two step synchronizer 1: one step synchronizer
22-16	-	-	-	Reserved
15-0	QOS_OVR	R/W	0	When one or more bits are HIGH, coincident with arvalid and arready, and when the arid match bits are equivalent to the qos_override bit(s), then the QoS for the read access is forced to minimum latency.

DDR2 SDRAM Controller Configuration Register 1 (M1CFG1) 0xB0303014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									RST_BYPASS	-			DQM_INIT	CKE_INIT	-

Field	Name	RW	Reset	Description
6	RST_BYPASS	R/W	0	This signal is used for ATPG testing only.
5-3	-	-	-	Reserved
2	DQM_INIT	R/W	0	The dqm output ports to the external memory reset to this value.
1	CKE_INIT	R/W	0	The cke output port to the external memory resets to this value.
0	-	-	-	Reserved

Common Configuration Register (COMMON) 0xB0303020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PAUSE_REQ		PAUSE_ACK	SLOW	FB_SEL			OPT_AMAP						CSYSREQ3	CSYSREQ2	CSYSREQ1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
							CSYSREQ0					AXI_SEL1	MODE_1CS	IO_SEL	AXI_SEL0

Field	Name	RW	Reset	Description
31	PAUSE_REQ	R/W	0	If this bit set to high and Queue of DRAM controller is empty, PAUSE_ACK is high and AXI BUS is busy state. If this bit reset, AXI BUS is ready state and can be possible to access. When LPDDR or Combo DDR2/DDR3 Controller is is used to control DRAM memory, PAUSE_REQ should be set before self-refresh start and have to wait until PAUSE_ACK set to high. This bit is not automatically cleared.
30	-	-	-	Reserved
29	PAUSE_ACK	R	0	Empty flag of DRAM controller Queue when PAUSE_REQ set to high
28	BUSSLOW	R/W	1	DDR Controller Slice Enable 0 : Enable. 1 : Disable ***This flag should be set to "0", If $F_{mbus} > F_{mbus\_max}/2$***
27	FB_SEL	R/W	0	If this bit set to high, operation mode selection(ctrl_fnc_fb) configuration bit from a DDR sdram controller is used. otherwise PHYCCFG[ctrl_fnc_fb] of Miscellneous Configuration Register is used as mode selection bits
24	OPT_AMAP	R/W	0	Optional Address mapping If set to high, araddr[30] and awaddr[30] is forced to 0
18	CSYSREQ3	-	-	DDR2/DDR3 SDRAM Controller system low-power request. This signal is a request to enter a Low-power state
17	CSYSREQ2	-	-	LPDDR/LPDDR2/DDR2 SDRAM Controller system low-power request. This signal is a request to enter a Low-power state
16	CSYSREQ1	R/W	1	DDR2 SDRAM Controller system low-power request. This signal is a request to enter a Low-power state
15-9	-	-	-	Reserved
8	CSYSREQ0	R/W	1	LPDDR SDRAM Controller system low-power request. This signal is a request to enter a Low-power state
7-4	-	-	-	Reserved
3	AXI_SEL1	-	-	AXI Interface mux selection signal 1 between memory bus and sdram controllers. Refer to note1)
2	MODE_1CS	R/W	0	In the DDR2 case, the 1 CS mode enabled. In this case, the CSN1 and ODT1 can be used to extra addresses.
1	IO_SEL	R/W	1	IO Interface mux selection signal between SDRAM PHY and SDRAM Controller 0: MDDR/LPDDR SDRAM Controller 1: DDR2 SDRAM Controller
0	AXI_SEL0	R/W	1	AXI Interface mux selection signal0 between memory bus and SDRAM Controllers Refer to notes 1) r

*note 1)
AXI Interface of dram memory bus is available for each sdram controller as combination of AXI_SEL1 and AXI_SEL0
{AXI_SEL1, AXI_SEL0} = 00b → AXI Interface of dram memory bus is assigned to LPDDR SDRAM Controller
= 01b → AXI Interface of dram memory bus is assigned to DDR2 SDRAM Controller
= 10b → AXI Interface of dram memory bus is assigned to LPDDR/LPDDR2/DDR2 SDRAM Controller
= 11b → AXI Interface of dram memory bus is assigned to DDR2/DDR3 Controller

PHY Configuration Register (PHYCFG) 0xB0303024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						PHYCTRLSEL								ctrl_fnc_fb	

Field	Name	RW	Reset	Description
9-8	PHYCTRLSEL	R/W	0	PHY Configuration register selection 00b,01b : PHY is configured by the PHY control registers of SDRAM PHY Registers(MDDR/LPDDR or DDR2 SDRAM) 10b : PHY is configured by the PHY control registers of LPCON SDRAM controller. 11b : PHY is configured by the PHY control registers of DDR3 SDRAM controller
2-0	ctrl_fnc_fb	R/W	0	Valid only when {mode_phy, mode_nand, mode_scan, mode_mux} is 4'b00000. For normal operation 3'b000: Normal operation mode. For ATE test purpose 3'b010: External FNC read feedback test mode. 3'b011: Internal FNC read feedback test mode. For Board test purpose 3'b100: External PHY read feedback test mode. When memory is not attached on the board 3'b101: Internal PHY read feedback test mode. mode_highz should be set. 3'b110: Internal PHY write feedback test mode. mode_highz should be set The function is only available to the PHY When COMMON[FB_SEL] is low.

LPDDR/LPDDR2/DDR2 SDRAM Controller Configuration Register 0(M2CFG0) 0xB0303030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QOS_FAST															

Field	Name	RW	Reset	Description
15-0	QOS_FAST	R/W	0	When one or more bits are HIGH, coincident with arvalid and arready, and when the arid match bits are equivalent to the qos_fast bit(s), then the QoS fast count is used as time-out count

LPDDR/LPDDR2/DDR2 SDRAM SDRAM Controller Configuration Register 1(M2CFG1) 0xB0303034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														CKE_INIT	

Field	Name	RW	Reset	Description
1	CKE_INIT	R/W	0	The cke output port to the external memory resets to this value.

DDR2/DDR3 SDRAM Controller Configuration Register 0(M3CFG0) 0xB0303044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														CKE_INIT	

Field	Name	RW	Reset	Description
1	CKE_INIT	R/W	0	The cke output port to the external memory resets to this value.

4.7.6 DDR PHY Registers

PHY Mode Control Register 0xB0304400

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ctrl_pulld_dqs				ctrl_pulld_dq											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	HIGHZ	MUX	ctrl_cmosrcv	shgate	ctrl_read_width	ckdis	ctrl_pd					ctrl_dfdqs	Ctrl_hcke	ddr3	lpddr2

Field	Name	RW	Reset	Description
31-29	ctrl_pulld_dqs	R/W	0	Active **HIGH** signal to pull-up or down PDQS/NDQS signals. For normal operation this field should be zero. When bus is idle, it can be set to pull-down or up the bus not to make bus high-z state (PDQS/NDQS is pulled-down/up). **For LPDDR2 mode, this field should be always set for normal operation to make P/NDQS signals pull-down/up.**
27-24	ctrl_pulld_dq	R/W	0	Active **HIGH** signal to down DQ signals. For normal operation this field should be zero. When bus is idle, it can be set to pull-down or up the bus not to make bus high-z state.
14	HIGHZ	R/W	0	High-z test mode for I/O If this bit is set, output is disabled
13	MUX	R/W	0	For full mux mode, mode_mux should be set. {mode_phy, mode_nand,mode_scan, mode_mux} should be 4'b0001. 1'b0: PHY is used. 1'b1: PHY is not used.
12	ctrl_cmosrcv	R/W	0	This field controls the input mode of I/O 1'b0: Differential receiver mode for high speed operation 1'b1: CMOS receiver mode for low speed operation (< 200MHz)
11	ctrl_shgate	R/W	0	This field controls the gate control signal 1'b0: gate signal length = "burst length / 2" (for LPDDR2)

				1'b0: gate signal length = "burst length / 2" + 1 (for DQS Pull-Down mode, PUD_dqs[3:0] == 4'b1111) 1'b1: gate signal length = "burst length / 2" − 1 (for DDR2/DDR3)
10	ctrl_read_width	R/W	0	1'b0: Termination on period is (BL/2+1.5) cycle (Default) 1'b1: Termination on period is (BL/2+1) cycle, It can be used under 333MHz
9	Ctrl_ckdis	R/W	0	This field controls the CK/CKB 1'b0: Clock output is enabled 1'b1: Clock output is disabled
8-4	ctrl_pd	R/W	0	This field controls the input buffer of I/O If this field is set, input buffer is turned-off for power down. This field should be 0 for normal operation. ctrl_pd[3:0]: for each data slice/ctrl_pd[4]: for control slice
3	ctrl_dfdqs	R/W	0	1'b0: single-ended DQS 1'b1: differential DQS
2	ctrl_hcke	R/W	0	This signal decides the reset value of CKE and some signals. If this bit is set, reset value of io_adct_out[19] and io_cke_out is 1. Otherwise, reset value of them is 0.
1	ctrl_ddr3	R/W	0	1'b0: DDR2 and LPDDR2 1'b1: DDR3
0	ctrl_lpddr2	R/W	0	1'b0: DDR2 and DDR3 1'b1: LPDDR2

DLL Control Register **0xB0304404**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
											LOCKED	CTRL_REF			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LOCK										FLOCK	CLOCK	HALF	START	DLLON

Field	Name	RW	Reset	Description
20	LOCKED	R	-	DLL stable lock information. This field is set after flock is set. This field is cleared when flock is de-asserted for some period of time specified by ctrl_ref[3:0] value. This field is required for stable lock status check.
19-16	CTRL_REF	R/W	0	This field determines the period of time when ctrl_locked is cleared. 4'b0000: Don't use. 4'b0001: ctrl_flock is de-asserted during 6 clock cycles, ctrl_locked is deasserted. 4'b0010: ctrl_flock is de-asserted during 9 clock cycles, ctrl_locked is deasserted ~ 4'b1111: ctrl_flock is de-asserted during 48 clock cycles, ctrl_locked is deasserted.
14-5	LOCK	R	-	Locked delay line encoding value. lock [9:2] : number of delay cells for coarse lock. lock [1:0] : control value for fine lock.
4	FLOCK	R	-	Fine lock information. Specifies that DLL has fine locked the clock.
3	CLOCK	R	-	Coarse lock information. Specifies that DLL has been locked
2	HALF	R/W	0	HIGH active start signal to turn on the low speed mode for DLL. If this bit is set, DLL can run at low speed(80MHz ~ 100MHz).
1	START	R/W	0	HIGH active start signal to make the DLL run and lock. This signal should be kept HIGH during normal operation. If this signal becomes LOW, DLL stops running. To re-run DLL, make this signal HIGH again. In the case of re-running, DLL loses previous lock information. Before ctrl_start is set, be sure that ctrl_dll_on is HIGH.
0	DLL_ON	R/W	0	HIGH active start signal to turn on the DLL. This signal should be kept HIGH for for normal operation. If this signal becomes LOW, DLL is turned off and ctrl_clock and ctrl_flock become HIGH. This bit should be kept set before ctrl_start is set to turn on the DLL.

Notes:
{CLOCK, FLOCK = 2'b00} : DLL is not locked.
{CLOCK, FLOCK = 2'b01} : Impossible value.
{CLOCK, FLOCK = 2'b10} : Locked and "phase offset error" is less than 160ps.
{CLOCK, FLOCK = 2'b11} : Locked and "phase offset error" is less than 80ps.

The reset value of READ ONLY Bits is provided as "-" as the status of these registers immediately after reset is depends on PHY and does not have any significance. The significance of these is only after DLL lock operation is started.

DLL Phase Detector Configuration Register 0xB0304408

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INC								START_POINT							

Field	Name	RW	Reset	Description
15-8	INC	R/W	0	Increase amount of start point
7-0	START_POINT	R/W	-	Initial DLL lock start point. This is the number of delay cells and is the start point where "DLL" start tracing to be locked. Initial delay time is calculated by multiplying the unit delay of delay cell and this value.

Gate Control Register 0xB030440C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					ADB_EN	OFFSETC							SHIFTC		

Field	Name	RW	Reset	Description
10	ADB_EN	R/W	0	ADB ENABLE If this bit is set, PHY starts 32-bit CRC generation according to control and address signals.(ADCT[19:0], RAS, CAS, WE and CKE). If this bit is cleared, it stops CRC generation.
9-3	OFFSETC	R/W	0	Gate offset amount for DDR. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. OFFSETC[6] = 1 : **(tFS: fine step delay)** GATEout delay amount − OFFSETC [5:0] x tFS OFFSETC[6] = 0 : GATEout delay amount + OFFSETC [5:0] x tFS
2-0	SHIFTC	R/W	0	GATEout signal delay amount for DDR. If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes **HIGH** and **LOW**. This value is limited by the half of the maximum delay in Master Delay Line. 000: T/128(2.8125° shift), 001: T/64(5.625° shift) 010: T/32(11.25° shift) , 011: T/16(22.5° shift) 100: T/8(45° shift) , 101: T/4(90° shift) 110: T/2(180° shift) , 111: T(360° shift)

Read Data Slice 0 Control Register **0xB0304410**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									WROFFSET1						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
									RDOFFSET0						

Field	Name	RW	Reset	Description
6:0	RDOFFSET	R/W	0	**This field can be used to give offset to read DQS.** If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes **HIGH** and **LOW**. The right-shifted value is limited by the quarter of the maximum delay in Master Delay Line. Read DQS offset amount: **RDOFFSET0[6] = 1: (tFS: fine step delay)** **Read DQS 90° delay amount −** RDOFFSET0**[5:0] x tFS** **RDOFFSET0[6] = 0:** Read DQS 90° delay amount + ctrl_offsetr0[5:0] x tFS
22:16	WROFFSET	R/W	0	**This field can be used to give offset to write DQ.** If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes **HIGH** and **LOW**. The right-shifted value is limited by the quarter of the maximum delay in Master Delay Line. Write DQ offset amount : **WROFFSET0[6] = 1 : (tFS: fine step delay)** **Write DQ 270° delay amount −** WROFFSET0**[5:0] x tFS** **WROFFSET0 [6] = 0 :** Write DQ 270° delay amount + ctrl_offsetw0[5:0] x tFS

Read Data Slice 1 Control Register 0xB0304414

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									WROFFSET1						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									RDOFFSET1						

Field	Name	RW	Reset	Description
6-0	RDOFFSET1	R/W	0	**This field can be used to give offset to read DQS.** If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. The right-shifted value is limited by the quarter of the maximum delay in Master Delay Line. Read DQS offset amount : **RDOFFSET1[6] = 1 : (tFS: fine step delay) Read DQS 90° delay amount −** RDOFFSET1**[5:0] x tFS** **RDOFFSET1[6] = 0 :** Read DQS 90° delay amount + ctrl_offsetr1[5:0] x tFS
22-16	WROFFSET1	R/W	0	**This field can be used to give offset to write DQ.** If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes **HIGH** and **LOW**. The right-shifted value is limited by the quarter of the maximum delay in Master Delay Line. Write DQ offset amount : **WROFFSET1[6] = 1 : (tFS: fine step delay) Write DQ 270° delay amount −** WROFFSET1**[5:0] x tFS** **WROFFSET1[6] = 0 :** Write DQ 270° delay amount + ctrl_offsetw1[5:0] x tFS

Read Data Slice 2 Control Register 0xB0304418

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									WROFFSET2						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
									RDOFFSET2						

Field	Name	RW	Reset	Description
22-16	WROFFSET2	R/W	0	**This field can be used to give offset to write DQ.** If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes **HIGH** and **LOW**. The right-shifted value is limited by the quarter of the maximum delay in Master Delay Line. Write DQ offset amount : WROFFSET2**[6] = 1 : (tFS: fine step delay)** **Write DQ 270° delay amount −** WROFFSETW2**[5:0] x tFS** **WROFFSETW2[6] = 0 :** Write DQ 270° delay amount + ctrl_offsetw2[5:0] x tFS
6-0	RDOFFSET2	R/W	0	**This field can be used to give offset to read DQS.** If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. The right-shifted value is limited by the quarter of the maximum delay in Master Delay Line. Read DQS offset amount : **RDOFFSETR2[6] = 1 : (tFS: fine step delay)** **Read DQS 90° delay amount −** RDOFFSETR2**[5:0] x tFS** **RDOFFSETR2[6] = 0:** Read DQS 90° delay amount + ctrl_offsetr2[5:0] x tFS

Read Data Slice 3 Control Register 0xB030441C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									WROFFSET3						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									RDOFFSET3						

Field	Name	RW	Reset	Description
22-16	WROFFSET3	R/W	0	**This field can be used to give offset to write DQ.** If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes **HIGH** and **LOW**. The right-shifted value is limited by the quarter of the maximum delay in Master Delay Line. Write DQ offset amount : **WROFFSET3[6] = 1 : (tFS: fine step delay)** **Write DQ 270° delay amount −** WROFSET3**[5:0] x tFS** **WROFFSET3[6] = 0 :** Write DQ 270° delay amount + ctrl_offsetw3[5:0] x tFS
6-0	RDOFFSET3	R/W	0	**This field can be used to give offset to read DQS.** If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. The right-shifted value is limited by the quarter of the maximum delay in Master Delay Line. Read DQS offset amount : **RDOFFSETDD3[6] = 1 : (tFS: fine step delay)** **Read DQS 90° delay amount − RDOFFSETD3 [5:0] x tFS** **RDOFFSETD3[6] = 0 :** Read DQS 90° delay amount + ctrl_offsetr3[5:0] x tFS

CLK Delay Register 0xB0304420

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									OFFSETD						

Field	Name	RW	Reset	Description
6-0	OFFSETD	R/W	0	This field is for debug purpose. (For LPDDR2) If this field is fixed, this should not be changed during operation. This value is valid only after ctrl_resync becomes HIGH and LOW. The right-shifted value is limited by the quarter of the maximum delay in Master Delay Line. offset amount for 270° clock generation: OFFSETD[6] = 1 : **(tFS: fine step delay)** 270° delay amount − OFFSETD[5:0] x tFS OFFSED[6] = 0 : 270° delay amount + OFFSETD[5:0] x tFS

DLL Force Lock Value Register 0xB0304424

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								FORCE							

Field	Name	RW	Reset	Description
7-0	FORCE	R/W	0	This field is used instead of ctrl_lock_value[9:2] found by the DLL only when ctrl_dll_on is **LOW** ,i.e. If the DLL is off, this field is used to generate 270° clock and shift DQS by 90°.

ZQ Calibration Control Register 0xB0304428

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								ZQ_DIV			PRD_CAL				PRD_CEN
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRV_STR			TERM_DIS	TERM_VAL			PULL_DOWN			PULL_UP			ZQ	UPDATE	CAL_START

Field	Name	RW	Reset	Description
23-21	ZQ_DIV	R/W	0	Calibration I/O clock selection signals. 3'b000 : system clock / 2 3'b001 : system clock / 4 3'b010 : system clock / 8 3'b011 : system clock / 16 (recommend value for 400MHz) 3'b100 : system clock / 32 Don't use the other setting.
20-17	PRD_CAL	R/W	0	Periodic calibration Update counter load value
16	PRD_CEN	R/W	0	Periodic calibration Enable
15-13	DRV_STR	R/W	0	Driver strength selection.
12	TERM_DIS	R/W	0	Termination disable selection. 1 : termination disable. 0 : termination enable.
11-9	TERM_VAL	R/W	0	On-die-termination resistor value selection.
8-6	PULL_DOWN	R/W	0	Immediate control code for pull-down
5-3	PULL_UP	R/W	0	Immediate control code for pull-up
2	ZQ	R/W	0	Override IMPP[2:0]/IMPN[2:0] instead of calibration control code found after auto calibration.
1	UPDATE	R/W	0	Update calibration control code found by auto calibration
0	CAL_START	R/W	0	Auto calibration start signal.

Note: Periodic ZQ calibration bit should be enabled only in case when periodic ZQ calibration method is intended for ZQ calibration. Periodic ZQ calibration bit should be enabled simultaneously along with the CAL_START signal once the intial software calibration is finished.

ZQ Calibration Status Register 0xB030442C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								NMON			PMON			ERROR	END

Field	Name	RW	Reset	Description
7-5	NMON	RO	-	Control code found by auto calibration for pull-down.
4-2	PMON	RO	-	Control code found by auto calibration for pull-up.
1	ERROR	RO	-	Status indicating calibration completion with error.
0	END	RO	-	Status indicating calibration completion without error.

Read Delay Register 0xB0304430

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													READ_DLY		

Field	Name	RW	Reset	Description
2-0	READ_DLY	R/W	0	Programmable read delay between controller and memory device 000 – Zero Delay 001 – 1 clock cycle delay 010 – 2 clock cycles delay 011 – 3 clock cycles delay

5 SMC(Static Memory Controller)

5.1 Overview

The NVS2310 has a memory controller for various kind of static memory for digital media en-decoding system. It can manipulate NOR type Flash, ROM, SRAM type memories. These memories are selected by nCS3 ~ nCS0 pins. The data bus width can be configured for each chip select separately. The cycle parameter for accessing external memory can be configured by internal registers.

- 4 nCS for a external static memory and 1 nCS for IDE secondary nCS(nCS_IDE)
- 8 / 16 / 32 bit configurable data bus width in each chip select
- Normal / Wrapping / Indirect address mode
- The external static memory of SRAM, ROM and IDE Type is supportable

Figure 5.1 shows a block diagram of SMC. There is 16-depth wirte buffer that helps SMC core with reading from or writing to external static memory.

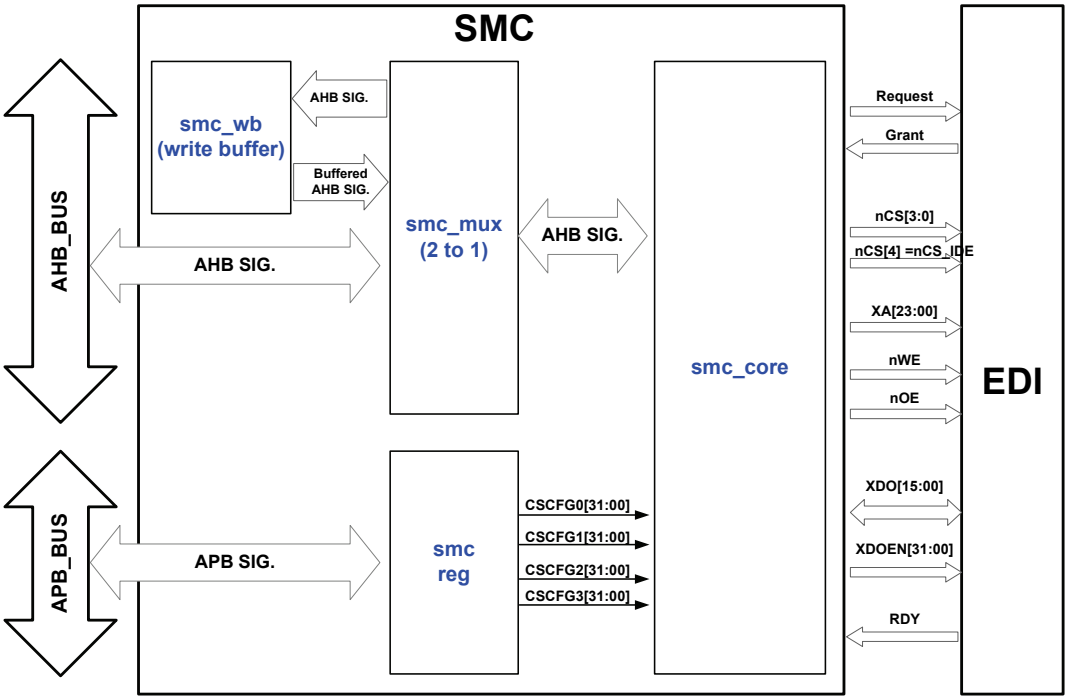


Figure 5.1 SMC Block Diagram

As shown in
Figure 5.2, SMC interface signals are connected to external static memory through EDI(External Device Interface) block. For that reason, EDI configuration & GPIO(A, B and F) function selection should be done, prior to access to external devices.

Figure 5.2 shows the default connection from SMC to external memory. The connection can be changed by the configuration of EDI_CSNCFG0, EDI_CSNCFG1, EDI_WENCFG, EDI_OENCFG or EDI_RDYCFG register, Refer to EDI for the detail description

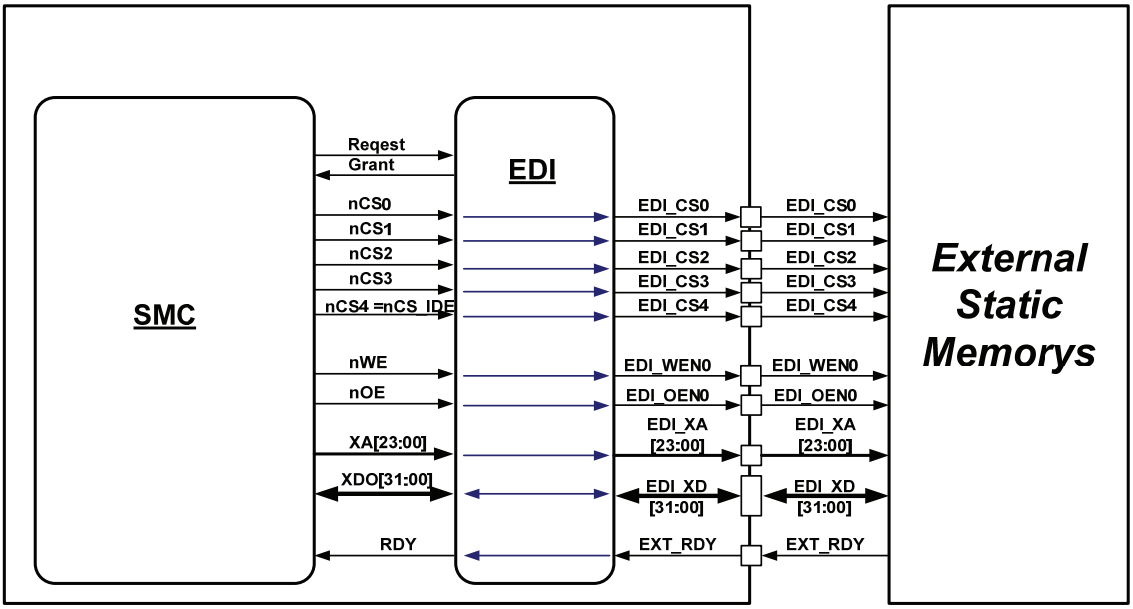


Figure 5.2 The Interface from SMC to External Static Memory in Default Configuration

There are four global chip select signals and one dedicated secondary IDE chip select signal. The base address mapping to access to external static memory is shown in Table 5.1.

Table 5.1 The Base Address for Each Chip Select

MODE	BASE ADDRESS	ETC
nCS0	0x20000000	If any nCS is configured to IDE Type, nCS_IDE is used for the second CS in this type
nCS1	0x30000000	
nCS2	0x80000000	
nCS3	0x90000000	

5.2 Register Description

Table 5.2 Memory Controller Register Map (Base Address = 0xB0306000)

Name	Address	Type	Reset	Description
STATUS	0x00	R	0x00010001	Status Register
CSNCFG0	0x20	R/W	0x4b40_3183	External Chip Select0 Config Register
CSNCFG1	0x24	R/W	0x4b40_1104	External Chip Select1 Config Register
CSNCFG2	0x28	R/W	0x4b40_4082	External Chip Select2 Config Register
CSNCFG3	0x2C	R/W	0x4b40_20C5	External Chip Select3 Config. Register

Status Register (STATUS) **0xB0306000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
										CST					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
															RDY

Field	Name	RW	Reset	Description
21-16	CST	RW	0	SMC Current Status 6'b00_0001 : ST_IDLE 6'b00_0010 : ST_WAIT … 6'b01_0000 : ST_WR0 6'b01_0001 : ST_WR1 6'b01_0010 : ST_WR2 6'b01_0100 : ST_WR3 6'b10_0000 : ST_RD0 6'b10_0001 : ST_RD1 6'b10_0010 : ST_RD2 6'b10_0100 : ST_RD3
0	RDY	RW	0	External Ready Input Status

In indirect address mode, indirect address register value should be changed when the CST is in ST_IDLE state, otherwise unexpected address could be generated.

External Chip Select Configuration Register (CSCFGn, n=0,1,2,3) **0xB0306020 + (n*4)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTnOE	PTnWE	Reserved		MSIZE		MTYPE		URDY	RDY	Reserved					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STP				PW								HLD			

Field	Name	RW	Reset	Description
31	PTnOE	RW	0x0	nOE Pulse Type 0 : STP and HLD timming parameter values would be applied as set below 1 : When STP and HLD paramter values are all zeros, HLD is forced to be 1, corresponding to OE signal.
30	PTnWE	RW	0x1	nWE Pulse Type 0 : STP and HLD timming parameter values would be applied as set below 1 : When STP and HLD paramter values are all zeros, HLD is forced to be 1, corresponding to WE signal.
27-26	MSIZE	RW	0x2	Memory Bus Size 0,1 : Bus width = 32 bit 2 : Bus width = 16 bit 3 : Bus width = 8 bit
25-24	MTYPE	RW	0x0	Type of External Memory 0 : Reserved 1 : IDE Type 2 : SMEM_0 type (Ex : ROM, NOR flash) Byte-write control signal (DQM) is not needed 3 : SMEM_1 type (Ex : SRAM) Byte-write control signal (DQM) is needed
23	MTYPE	RW	0x0	Use Ready 0 : None 1 : Ready / Busy signal monitoring is enabled The memory controller extends access cycle until the state of READY pin indicates that the access request has accomplished
22	RDY	RW	0x1	Ready / Busy Select Note : Refer to Figure 5.3 for ready/busy cycle extension. 0 : The READY pin indicates the READY signal. The memory controller extends access cycle until this pin goes to high state. 1 : The READY pin indicates the BUSY signal. The memory controller extends access cycle until this pin goes to low state.
15-12	STP	RW	0x0	N cycle is issued between the falling edge of nCS[n] and nOE / nWE
11-4	PW	RW	0x0	(N+1) cycle is issued between the falling and rising edge of nOE / new (N= 0~255)
3:0	HLD	RW	0x0	N : N cycle is issued between the rising edge of nOE / nWE and nCS[n].

The following figure displays the element cycle diagram for external memories.

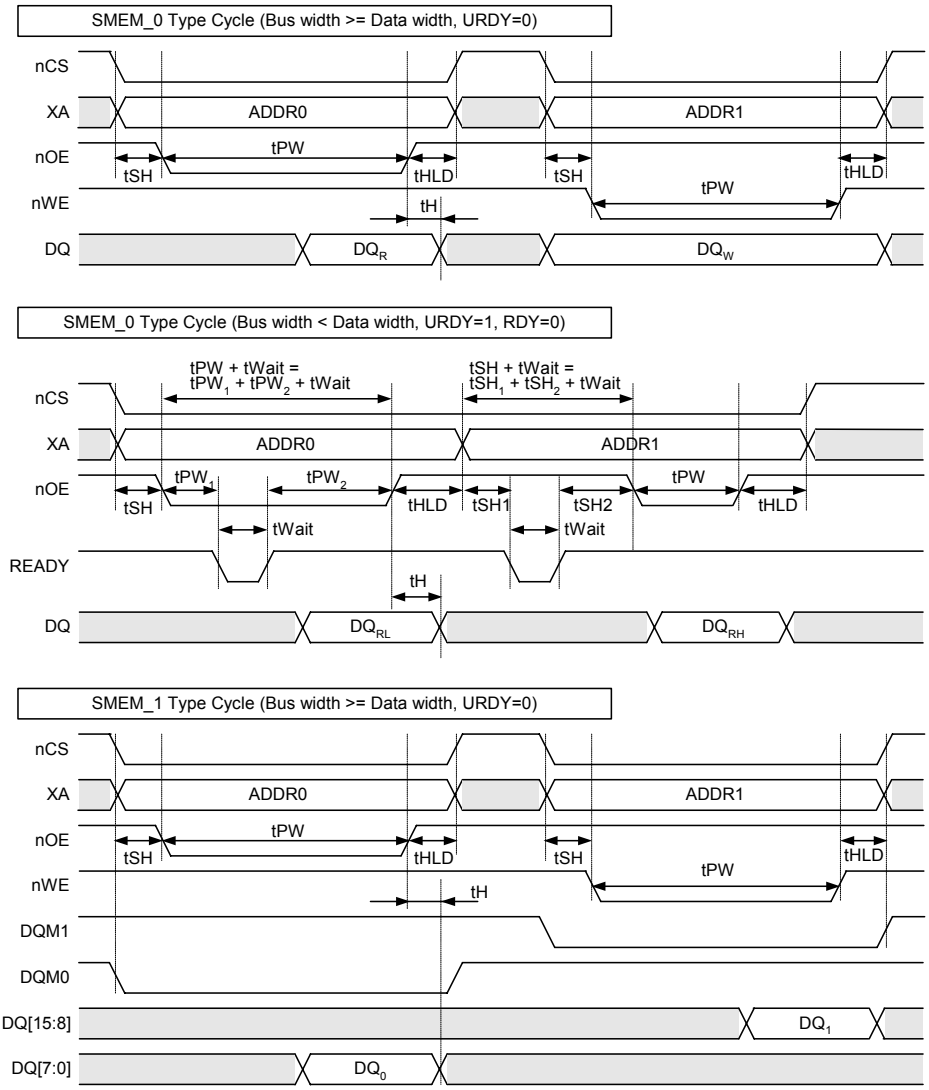


Figure 5.3 Basic Timing Diagram For External Memories

In case of IDE type memory, there are two chip-enable signals for it. In the NVS2310, each enable signal can be controlled by offset address space. ‘nCS0’ reflects that the offset address range of 0 ~ 0x1F is accessed, ‘nCS1’ reflects that 0x20 ~ 0x3F is accessed. For larger address than 0x3F, bit5 of address value means which enable signal is activated. (0 to ‘nCS0’, 1 to ‘nCS1’)

6 EDI (External Device Interface)

6.1 Overview

The NVS2310 EDI(External Device Interface) makes it possible to share common output ports, used for control, address and data signal of NFC and SMC. EDI arbitrates requests for using the ports from SMC and NFC and grant the ports one of them. The arbitration policy is, by default, Round-Robin and optionally can be change to Fixed-Priority when NFC and SMC stimutaneously access the output ports.

The supported main features are as follows

- Channel Arbitration for SMC and NFC
- 9 input CSN(chip select) from NFC and SMC is configurably mapped to 7 output EDI_CSN which is mapped to GPIO
- 6 input EDI_RDY from GPIO is configurably mapped to 5 output RDY, one is assigned to SMC, the others is assigned to NFC.
- Arbitration policy for common shared port is one of the Round-Robin and Fixed Priotiry arbitration configured by register.

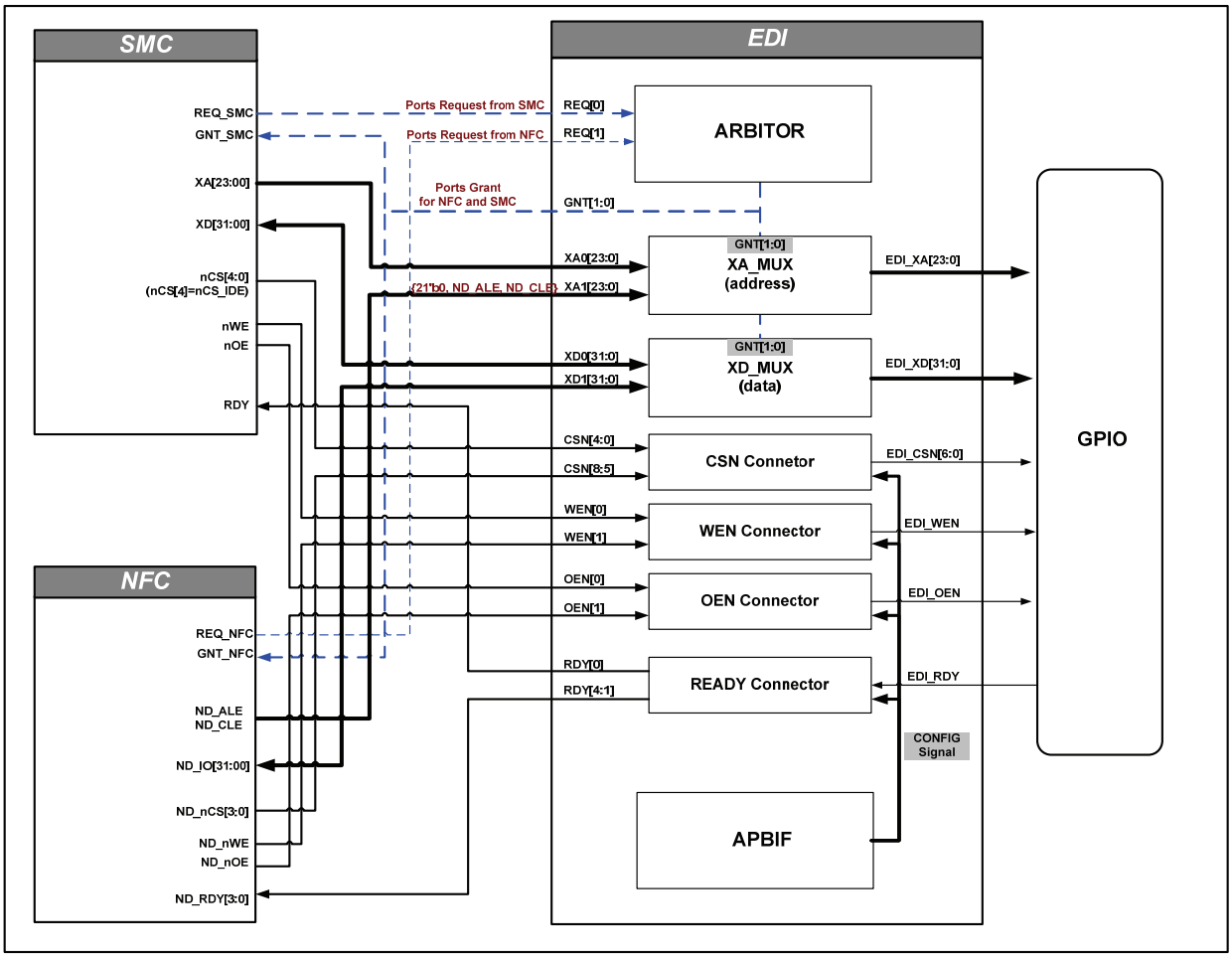


Figure 6.1 Block Diagram and Interface of EDI

6.2 Register Description

Table 6.1 EDI Register Map (BASE ADDRESS : 0xB0308000)

Name	Offset	Type	Reset	Description
EDI_CTRL	0x00	R/W	0x00000000	EDI Control Register.
EDI_CSNCFG0	0x04	R/W	0x00543210	EDI CSN Configuration Register 0.
EDI_CSNCFG1	0x08	R/W	0x00BA9876	EDI CSN Configuration Register 1.
EDI_OENCFG	0x0C	R/W	0x00000004	EDI OEN Configuration Register
EDI_WENCFG	0x10	R/W	0x00000004	EDI WEN Configuration Register
EDI_RDYCFG	0x14	R/W	0x76543210	EDI Ready Configuration Register
Reserved	0x18	-	-	-
Reserved	0x1C	-	-	
EDI_REQOFF	0x20	R/W	0x00000000	EDI Request OFF Flag Register

EDI Control Register (EDI_CTRL) 0xB0308000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0				R*			AM	0			PRIORITY[4:0]				

Field	Name	RW	Reset	Description
8	AM	R/W	0x0	EDI arbitration mode selection 0 : Round-Robine arbitration mode 1 : Fixed-Priority arbitration mode
4-0	PRIORITY	R/W	0x0	EDI Fixed-Priority Configuration 5'b00000 : CH0 → CH1 5'b00110 : CH1 → CH0

EDI CSN Configuration Register 0 (EDI_CSNCFG0) 0xB0308004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				0				CFGCS5				CFGCS4			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CFGCS3				CFGCS2				CFGCS1				CFGCS0			

Refer to Table 6.2 and Figure 6.2

Table 6.2 EDI_CSNCFG0[CFGCSn] Configuration

Field	Name	RW	Reset	Description
23-20	CFGCS5	R/W	0x5	0 : CSN[0] of EDI is connected to EDI_CSN[5] 1 : CSN[1] of EDI is connected to EDI_CSN[5] 2 : CSN[2] of EDI is connected to EDI_CSN[5] 3 : CSN[3] of EDI is connected to EDI_CSN[5] 4 : CSN[4] of EDI is connected to EDI_CSN[5] 5 : CSN[5] of EDI is connected to EDI_CSN[5] 6 : CSN[6] of EDI is connected to EDI_CSN[5] 7 : CSN[7] of EDI is connected to EDI_CSN[5] 8 : CSN[8] of EDI is connected to EDI_CSN[5] o/w : reserved
19-16	CFGCS4	R/W	0x4	0 : CSN[0] of EDI is connected to EDI_CSN[4] 1 : CSN[1] of EDI is connected to EDI_CSN[4] 2 : CSN[2] of EDI is connected to EDI_CSN[4] 3 : CSN[3] of EDI is connected to EDI_CSN[4] 4 : CSN[4] of EDI is connected to EDI_CSN[4] 5 : CSN[5] of EDI is connected to EDI_CSN[4] 6 : CSN[6] of EDI is connected to EDI_CSN[4] 7 : CSN[7] of EDI is connected to EDI_CSN[4] 8 : CSN[8] of EDI is connected to EDI_CSN[4] o/w : reserved
15-12	CFGCS3	R/W	0x3	0 : CSN[0] of EDI is connected to EDI_CSN[3] 1 : CSN[1] of EDI is connected to EDI_CSN[3] 2 : CSN[2] of EDI is connected to EDI_CSN[3] 3 : CSN[3] of EDI is connected to EDI_CSN[3]

				4 : CSN[4] of EDI is connected to EDI_CSN[3] 5 : CSN[5] of EDI is connected to EDI_CSN[3] 6 : CSN[6] of EDI is connected to EDI_CSN[3] 7 : CSN[7] of EDI is connected to EDI_CSN[3] 8 : CSN[8] of EDI is connected to EDI_CSN[3] o/w : reserved
11-8	CFGCS2	R/W	0x2	0 : CSN[0] of EDI is connected to EDI_CSN[2] 1 : CSN[1] of EDI is connected to EDI_CSN[2] 2 : CSN[2] of EDI is connected to EDI_CSN[2] 3 : CSN[3] of EDI is connected to EDI_CSN[2] 4 : CSN[4] of EDI is connected to EDI_CSN[2] 5 : CSN[5] of EDI is connected to EDI_CSN[2] 6 : CSN[6] of EDI is connected to EDI_CSN[2] 7 : CSN[7] of EDI is connected to EDI_CSN[2] 8 : CSN[8] of EDI is connected to EDI_CSN[2] o/w : reserved
7-4	CFGCS1	R/W	0x1	0 : CSN[0] of EDI is connected to EDI_CSN[1] 1 : CSN[1] of EDI is connected to EDI_CSN[1] 2 : CSN[2] of EDI is connected to EDI_CSN[1] 3 : CSN[3] of EDI is connected to EDI_CSN[1] 4 : CSN[4] of EDI is connected to EDI_CSN[1] 5 : CSN[5] of EDI is connected to EDI_CSN[1] 6 : CSN[6] of EDI is connected to EDI_CSN[1] 7 : CSN[7] of EDI is connected to EDI_CSN[1] 8 : CSN[8] of EDI is connected to EDI_CSN[1] o/w : reserved
3-0	CFGCS0	R/W	0x0	0 : CSN[0] of EDI is connected to EDI_CSN[0] 1 : CSN[1] of EDI is connected to EDI_CSN[0] 2 : CSN[2] of EDI is connected to EDI_CSN[0] 3 : CSN[3] of EDI is connected to EDI_CSN[0] 4 : CSN[4] of EDI is connected to EDI_CSN[0] 5 : CSN[5] of EDI is connected to EDI_CSN[0] 6 : CSN[6] of EDI is connected to EDI_CSN[0] 7 : CSN[7] of EDI is connected to EDI_CSN[0] 8 : CSN[8] of EDI is connected to EDI_CSN[0] o/w : reserved

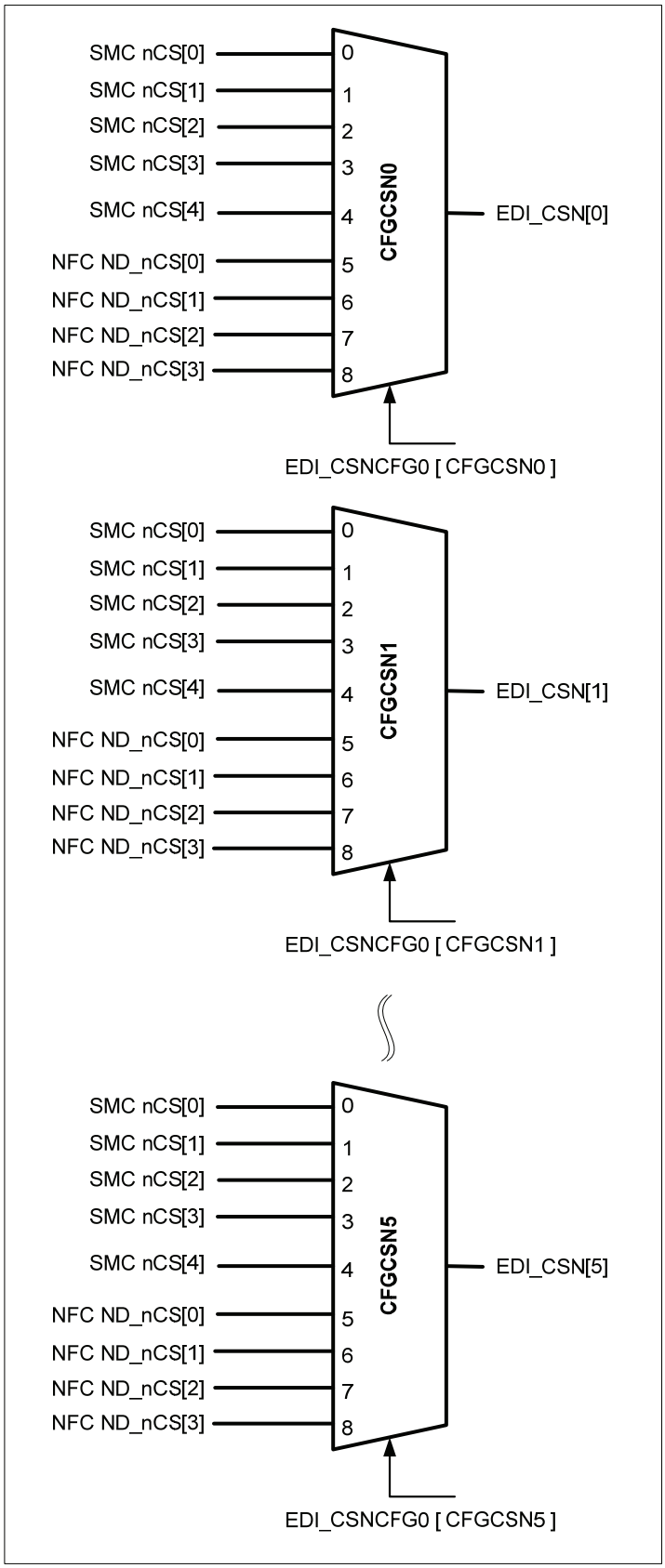


Figure 6.2 Block Diagram for EDI_CSN[5:0]

EDI CSN Configuration Register 1 (EDI_CSNCFG1) **0xB0308008**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				0				Reserved				Reserved			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				Reserved				Reserved				CFGCS6			

Field	Name	RW	Reset	Description
3-0	CFGCS6	R/W	0x6	0 : CSN[0] of EDI is connected to EDI_CSN[6] 1 : CSN[1] of EDI is connected to EDI_CSN[6] 2 : CSN[2] of EDI is connected to EDI_CSN[6] 3 : CSN[3] of EDI is connected to EDI_CSN[6] 4 : CSN[4] of EDI is connected to EDI_CSN[6] 5 : CSN[5] of EDI is connected to EDI_CSN[6] 6 : CSN[6] of EDI is connected to EDI_CSN[6] 7 : CSN[7] of EDI is connected to EDI_CSN[6] 8 : CSN[8] of EDI is connected to EDI_CSN[6] o/w : reserved

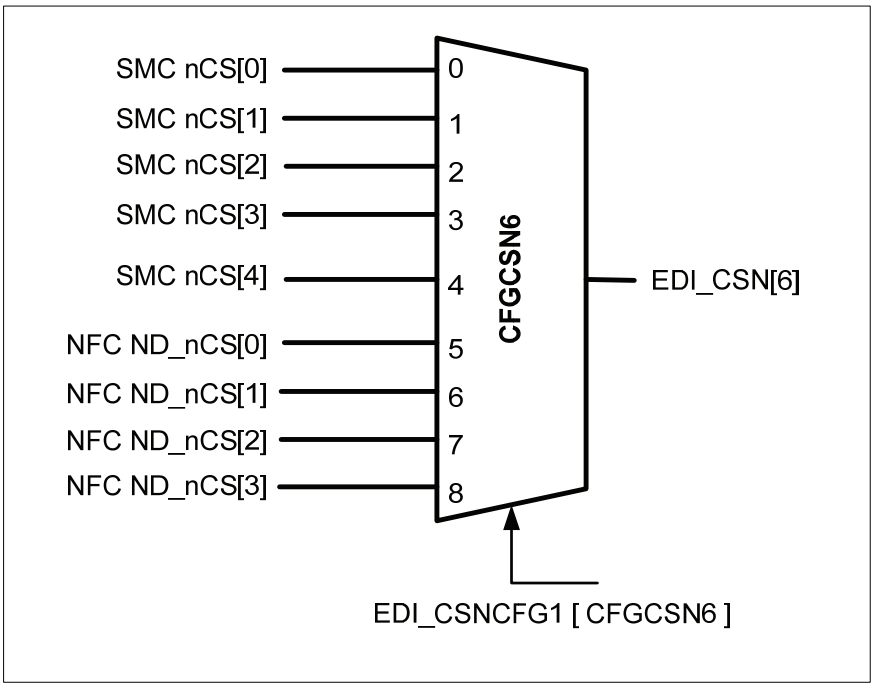


Figure 6.3 Block Diagram for EDI_CSN[6]

EDI OEN Configuration Register (EDI_OENCFG) 0xB030800C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												CFGOEN1		CFGOEN0	

Field	Name	RW	Reset	Description
3-2	CFGOEN1	R/W	0x1	OEN1 Configuration 0 : OEN[0] of EDI is connected to EDI_OEN[1] 1 : OEN[1] of EDI is connected to EDI_OEN[1] o/w : reserved
1-0	CFGOEN0	R/W	0x0	OEN0 Configuration 0 : OEN[0] of EDI is connected to EDI_OEN[0] 1 : OEN[1] of EDI is connected to EDI_OEN[0] o/w : reserved

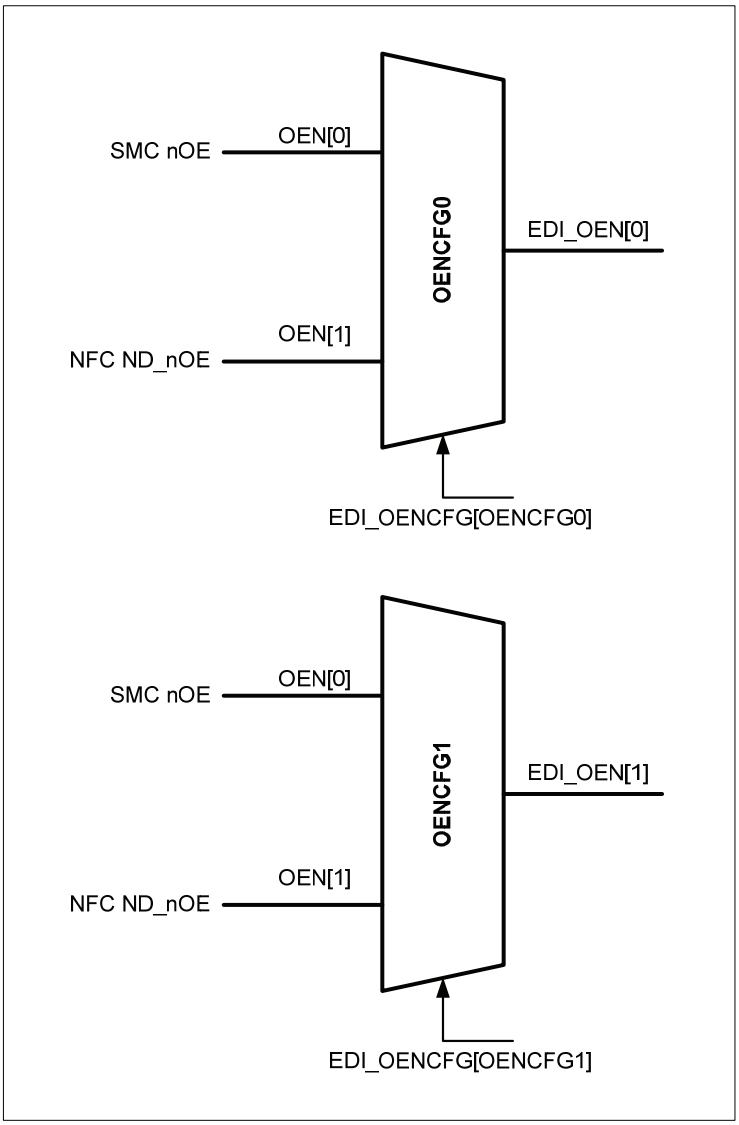


Figure 6.4 Block Diagram for EDI OENCFG

EDI WEN Configuration Register (EDI_WENCFG) **0xB0308010**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												CFGWEN1		CFGWEN0	

Field	Name	RW	Reset	Description
3-2	CFGWEN1	R/W	0x1	WEN1 Configruation 0 : WEN[0] of EDI is connected to EDI_WEN[1] 1 : WEN[1] of EDI is connected to EDI_WEN[1] o/w : reserved
1-0	CFGWEN0	R/W	0x0	WEN0 Configruation 0 : WEN[0] of EDI is connected to EDI_WEN[0] 1 : WEN[1] of EDI is connected to EDI_WEN[0] o/w : reserved

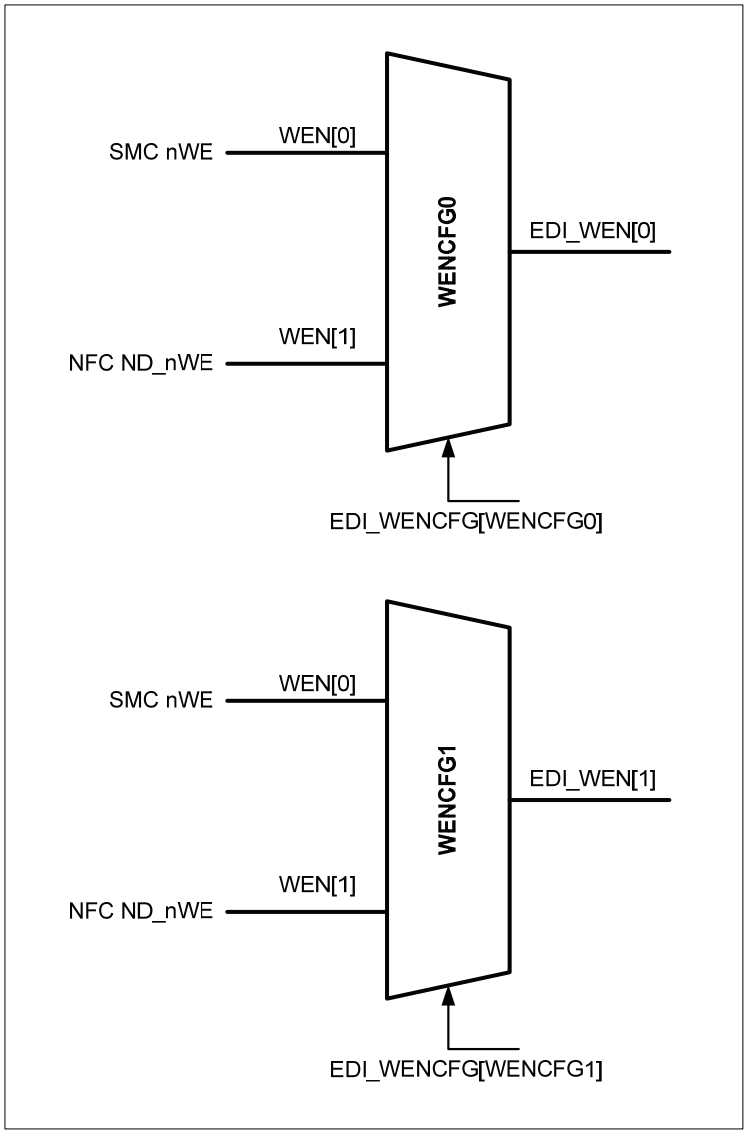


Figure 6.5 Block Diagram for EDI WENCFG

EDI READY Configuration Register 0 (EDI_RDYCFG) 0xB0308014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	Reserved			0	Reserved			0	Reserved			0	CFGRDY4		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	CFGRDY3			0	CFGRDY2			0	CFGRDY1			0	CFGRDY0		

Field	Name	RW	Reset	Description
18-16	CFGRDY4	R/W	0x4	RDY4 Configruation 0 : EDI_RDY[0] of EDI is connected to RDY[4] 1 : EDI_RDY[1] of EDI is connected to RDY[4] 2 : EDI_RDY[2] of EDI is connected to RDY[4] 3 : EDI_RDY[3] of EDI is connected to RDY[4] 4 : EDI_RDY[4] of EDI is connected to RDY[4] 5 : EDI_RDY[5] of EDI is connected to RDY[4] o/w : reserved
14-12	CFGRDY3	R/W	0x3	RDY3 Configruation 0 : EDI_RDY[0] of EDI is connected to RDY[3] 1 : EDI_RDY[1] of EDI is connected to RDY[3] 2 : EDI_RDY[2] of EDI is connected to RDY[3] 3 : EDI_RDY[3] of EDI is connected to RDY[3] 4 : EDI_RDY[4] of EDI is connected to RDY[3] 5 : EDI_RDY[5] of EDI is connected to RDY[3] o/w : reserved
10-8	CFGRDY2	R/W	0x2	RDY3 Configruation 0 : EDI_RDY[0] of EDI is connected to RDY[2] 1 : EDI_RDY[1] of EDI is connected to RDY[2] 2 : EDI_RDY[2] of EDI is connected to RDY[2] 3 : EDI_RDY[3] of EDI is connected to RDY[2] 4 : EDI_RDY[4] of EDI is connected to RDY[2] 5 : EDI_RDY[5] of EDI is connected to RDY[2] o/w : reserved
6-4	CFGRDY1	R/W	0x1	RDY3 Configruation 0 : EDI_RDY[0] of EDI is connected to RDY[1] 1 : EDI_RDY[1] of EDI is connected to RDY[1] 2 : EDI_RDY[2] of EDI is connected to RDY[1] 3 : EDI_RDY[3] of EDI is connected to RDY[1] 4 : EDI_RDY[4] of EDI is connected to RDY[1] 5 : EDI_RDY[5] of EDI is connected to RDY[1] o/w : reserved
2-0	CFGRDY1	R/W	0x0	RDY3 Configruation 0 : EDI_RDY[0] of EDI is connected to RDY[0] 1 : EDI_RDY[1] of EDI is connected to RDY[0] 2 : EDI_RDY[2] of EDI is connected to RDY[0] 3 : EDI_RDY[3] of EDI is connected to RDY[0] 4 : EDI_RDY[4] of EDI is connected to RDY[0] 5 : EDI_RDY[5] of EDI is connected to RDY[0] o/w : reserved

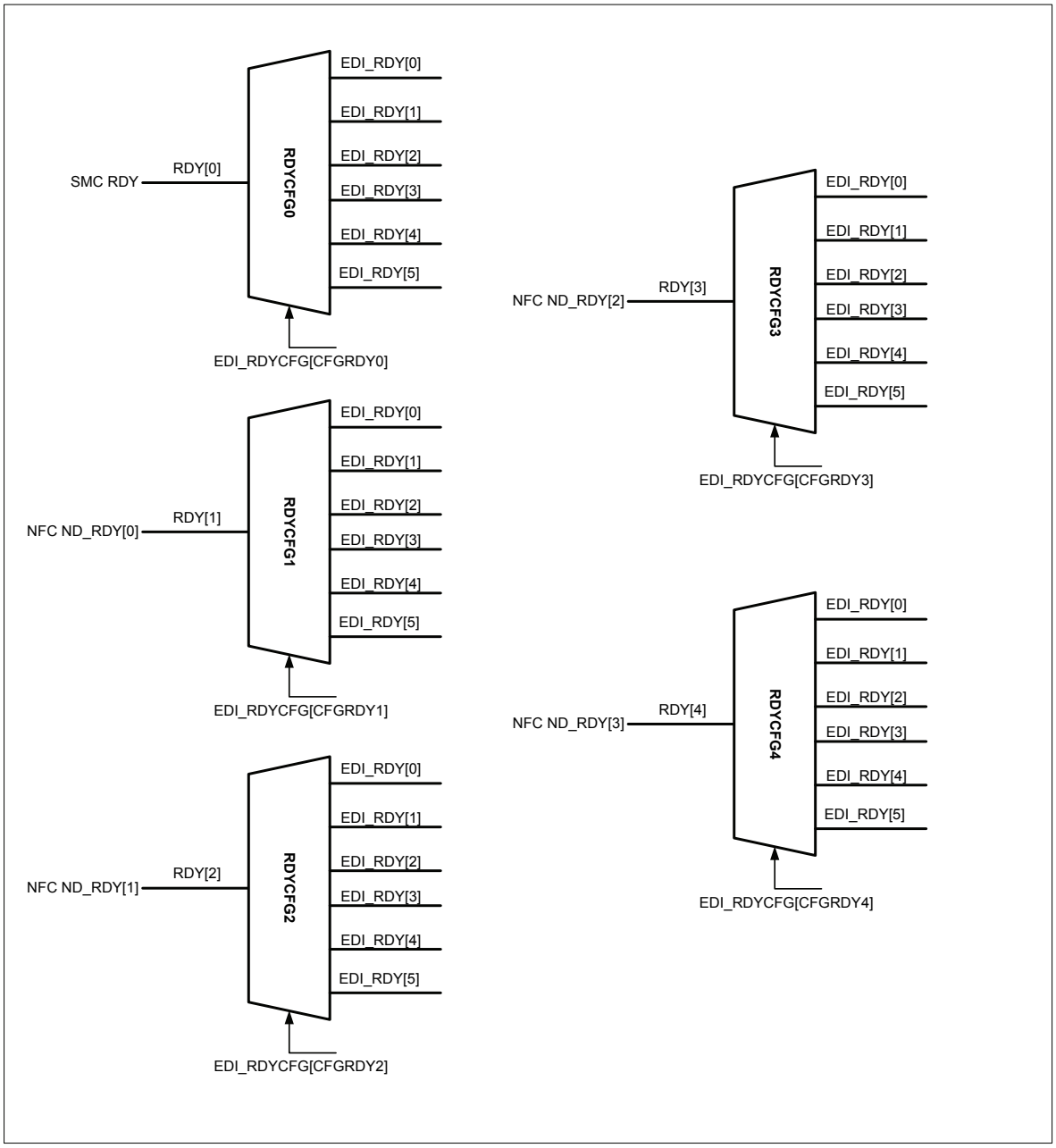


Figure 6.6 Block Diagram for EDI RDY[4:0]

EDI Request Off Flag Register (EDI_REQOFF) 0xB0308020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												Reserved		REQOFF	

This register shows current request pending state of EDI. Namely, when one channel(ch0:SMC, ch1:NFC) has acquired grant for access to EDI and started read/write operation, the request state of the other channel is reflected in this register.

Field	Name	RW	Reset	Description
1-0	REQOFF	R	0x0	0 : o/w 1 : SMC is granted by EDI and NFC is requesting for grant to EDI. 2 : NFC is granted by EDI and SMC is Requesting for grant to EDI. 3 : -

7 Prefetch Buffer

The prefetch buffer is instantiated in front of the three master ports, ARM data, high speed IO bus and IO bus for reducing bandwidth requirement. When enabled, the prefetch buffer conducts 64 words burst loading from the accessed start address and maintains the loaded data until prefetch buffer miss is occurred Therefore, If the accessed address space from the three master ports is linearly increased, the first access loads the multiple data from the addressed storage memory devices and the second and following access load the data not from the addressed storage memory devices but from the prefetch buffer. The prefetch buffer can be enabled and the four address space can be defined for the prefetch address space through the defined register in memory bus configuration block

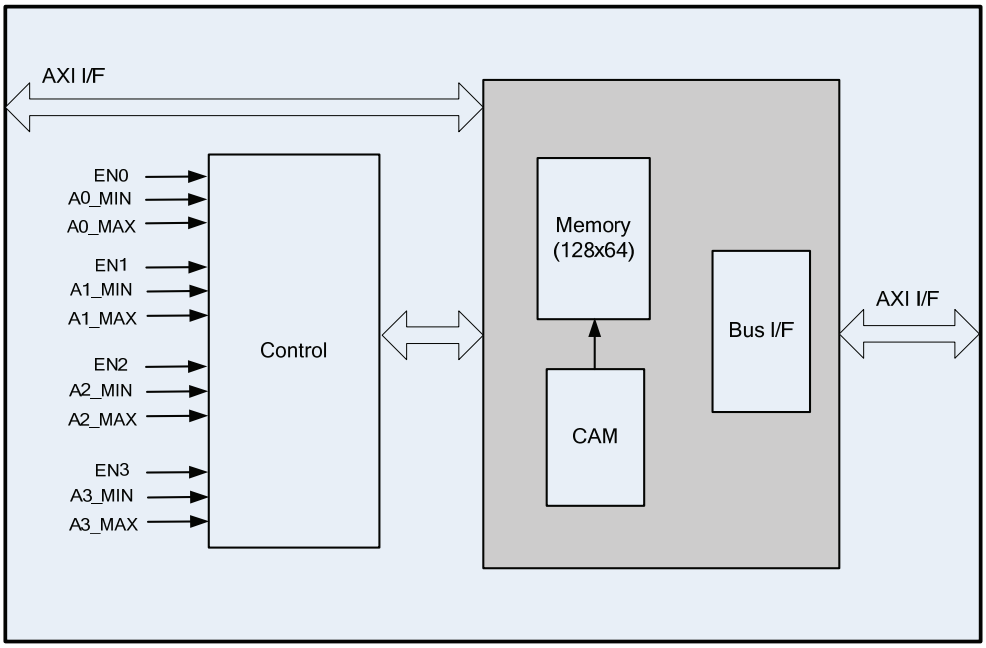


Figure 7.1 Block Diagram of Prefetch Buffer

8 Memory BUS Configuration

8.1 Register Description

Table 8.1 Memory Bus CFG Register Map (Base Address : 0xB030B000)

Name	Offset	Type	Reset	Description
HCLKMASK	0x00	R/W	0x00000000	Module Clock Mask Register
SWRESET	0x04	R/W	0x00543210	Memory Bus Soft Reset Register
BM	0x08	R/W	0x00BA9876	Multi-layer Bus Matrix Configuration Register
SLOWBUS	0x0C	R/W	0x00000004	Memrory Bus Slow Bus Configuration Register
PB0EN	0x20	R/W	0x00000000	Prefetch Buffer 0 (ARM Data) Enable Register
PB0A0MIN	0x24	R/W	0x00000000	Prefetch Buffer 0 (ARM Data) Address 0 MIN Register
PB0A0MAX	0x28	R/W	0x00000000	Prefetch Buffer 0 (ARM Data) Address 0 MAX Register
PB0A1MIN	0x2C	R/W	0x00000000	Prefetch Buffer 0 (ARM Data) Address 1 MIN Register
PB0A1MAX	0x30	R/W	0x00000000	Prefetch Buffer 0 (ARM Data) Address 1 MAX Register
PB0A2MIN	0x34	R/W	0x00000000	Prefetch Buffer 0 (ARM Data) Address 2 MIN Register
PB0A2MAX	0x38	R/W	0x00000000	Prefetch Buffer 0 (ARM Data) Address 2 MAX Register
PB0A3MIN	0x3C	R/W	0x00000000	Prefetch Buffer 0 (ARM Data) Address 3 MIN Register
PB0A3MAX	0x40	R/W	0x00000000	Prefetch Buffer 0 (ARM Data) Address 3 MAX Register
PB1EN	0x44	R/W	0x00000000	Prefetch Buffer 1 (IO Bus) Enable Register
PB1A0MAX	0x48	R/W	0x00000000	Prefetch Buffer 1 (IO Bus) Address 0 MIN Register
PB1A1MIN	0x4C	R/W	0x00000000	Prefetch Buffer 1 (IO Bus) Address 0 MAX Register
PB1A1MAX	0x50	R/W	0x00000000	Prefetch Buffer 1 (IO Bus) Address 1 MIN Register
PB1A2MIN	0x54	R/W	0x00000000	Prefetch Buffer 1 (IO Bus) Address 1 MAX Register
PB1A2MAX	0x58	R/W	0x00000000	Prefetch Buffer 1 (IO Bus ta) Address 2 MIN Register
PB1A3MIN	0x5C	R/W	0x00000000	Prefetch Buffer 1 (IO Bus) Address 2 MAX Register
PB1A3MAX	0x60	R/W	0x00000000	Prefetch Buffer 1 (IO Bus) Address 3 MIN Register
PB1A0MAX	0x64	R/W	0x00000000	Prefetch Buffer 1 (IO Bus) Address 3 MAX Register
PB2EN	0x68	R/W	0x00000000	Prefetch Buffer 2 (HSIO Bus) Enable Register
PB2A0MAX	0x6C	R/W	0x00000000	Prefetch Buffer 2 (HSIO Bus) Address 0 MIN Register
PB2A1MIN	0x70	R/W	0x00000000	Prefetch Buffer 2 (HSIO Bus) Address 0 MAX Register
PB2A1MAX	0x74	R/W	0x00000000	Prefetch Buffer 2 (HSIO Bus) Address 1 MIN Register
PB2A2MIN	0x78	R/W	0x00000000	Prefetch Buffer 2 (HSIO Bus) Address 1 MAX Register
PB2A2MAX	0x7C	R/W	0x00000000	Prefetch Buffer 2 (HSIO Bus ta) Address 2 MIN Register
PB2A3MIN	0x80	R/W	0x00000000	Prefetch Buffer 2 (HSIO Bus) Address 2 MAX Register
PB2A3MIN	0x84	R/W	0x00000000	Prefetch Buffer 2 (HSIO Bus) Address 3 MIN Register
PB2A0MAX	0x88	R/W	0x00000000	Prefetch Buffer 2 (HSIO Bus) Address 3 MAX Register

Memory Bus HCLK Mask Register(HCLKMASK) **0xB030B000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						HCLKMASK									

Field	Name	RW	Reset	Description
12-0	HCLKMASK	R/W	0x0	Enable signals of HCLK clock gating logic AHB HCLKs, supplied to each block of the memory bus, are controlled by clock enable signal. 0 : Enable clock 1 : Disable clock The bit position indicates each sub-block which is controlled by. BIT 0: LPDDR SDRAM Controller BIT 1: DDR2 SDRAM Controller BIT 2: LPDDR/LPDDR2/DDR2 SDRAM Controller BIT 3: Reserved BIT 4: Reserved BIT 5: Static Memory Bus BIT 6: EDI Controller BIT 7: Internal Memory Busr BIT 8: Internal Memory Bus Configuration Block BIT 9: DDR2/DDR3 SDRAM Controller

Memory Bus Soft Reset Register(SWRESET0) 0xB030B004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			SWRESET												

Field	Name	RW	Reset	Description
9-0	SWRESET	R/W	0x0	Enable signals of Software Reset AHB SWRESETs, supplied to each peripheral block, are controlled by SWRESET signal. 0 : Disable SWRESET 1 : Enable SWRESET The bit position indicates each sub-block which is controlled by. BIT 0: DDR Memory Bus BIT 1: Reserved BIT 2: Reserved BIT 3: Reserved BIT 4: Reserved BIT 5: Static Memory Bus BIT 6: EDI controller BIT 7: Internal Memory Bus BIT 8: Internal Memroy Bus Configuration Registers r BIT 9: Reserved BIT 10: Prefetch Buffer 1 (IO Bus) BIT 11: Prefetch Buffer 2 (High Speed IO Bus) BIT 12: Prefetch Buffer 0 (ARM Data)

Multi-Layer Bus Matrix Configuration Register(BM0) 0xB030B008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										PROTBM					

Field	Name	RW	Reset	Description
5-2	PROTBM	R/W	0xF	Slave Ports Access Enable 0 : Enable 1 : Disable BIT0 : DDR Bus BIT1 : IMC0 Bus BIT2 : Static Memory Bus BIT3 : IMC1 Bus

Memory Slow BUS Configuration Register(REGSLICE) 0xB030B00C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										HS2M	P2M			CDX	CIX

Field	Name	RW	Reset	Description
5	HS2M	R/W	0x0	High Speed IO Bus Slice Enable 0 : Enable 1 : Disable ***This flag should be set to "0", If $F_{mbus} > F_{mbus\_max}/2$***
4	P2M	R/W	0x0	IO Bus Slice Enable 0 : Enable 1 : Disable ***This flag should be set to "0", If $F_{mbus} > F_{mbus\_max}/2$***
3	-	-	-	Reserved
2	-	-	-	Reserved
1	CDX	R/W	0x0	ARM Data Slice Enable 0 : Enable 1 : Disable ***This flag should be set to "0", If $F_{mbus} > F_{mbus\_max}/2$***
0	CIX	R/W	0x0	ARM Instruction Slice Enable 0 : Enable 1 : Disable ***This flag should be set to "0", If $F_{mbus} > F_{mbus\_max}/2$***

Prefetch Buffer0 (ARM Data) Enable Register(PB0EN) 0xB030B020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												EN3	EN2	EN1	EN0

Field	Name	RW	Reset	Description
3	EN3	R/W	0x0	Prefetch Address Space3 Enable 0 : Disable 1 : Enable
2	EN2	R/W	0x0	Prefetch Address Space2 Enable 0 : Disable 1 : Enable
1	EN1	R/W	0x0	Prefetch Address Space1 Enable 0 : Disable 1 : Enable
0	EN0	R/W	0x0	Prefetch Address Space0 Enable 0 : Disable 1 : Enable

Prefetch Buffer0 (ARM Data) Address0 MIN Register(PB0A0MIN) 0xB030B024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB0_ADDR0_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB0_ADDR0_MIN															

Field	Name	RW	Reset	Description
31-0	PB00_ADDR0_MIN	R/W	0x0	Lower Address for Prefetch Address Space 0 for PB0

Prefetch Buffer0 (ARM Data) Address0 MAX Register(PB0A0MAX) 0xB030B028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB0_ADDR0_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB0_ADDR0_MAX															

Field	Name	RW	Reset	Description
31-0	PB0_ADDR0_MAX	R/W	0x0	Upper Address for Prefetch Address Space 0 for PB0

Prefetch Buffer0 (ARM Data) Address1 MIN Register(PB0A1MIN) 0xB030B02C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB0_ADDR1_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB0_ADDR1_MIN															

Field	Name	RW	Reset	Description
31-0	PB0_ADDR1_MIN	R/W	0x0	Lower Address for Prefetch Address Space 1 for PB0

Prefetch Buffer0 (ARM Data) Address1 MAX Register(PB0A1MAX) 0xB030B030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB0_ADDR1_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB0_ADDR1_MAX															

Field	Name	RW	Reset	Description
31-0	PB0_ADDR1_MAX	R/W	0x0	Upper Address for Prefetch Address Space 1 for PB0

Prefetch Buffer0 (ARM Data) Address2 MIN Register(PB0A2MIN) 0xB030B034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB0_ADDR2_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB0_ADDR2_MIN															

Field	Name	RW	Reset	Description
31-0	PB0_ADDR2_MIN	R/W	0x0	Lower Address for Prefetch Address Space 2 for PB0

Prefetch Buffer0 (ARM Data) Address2 MAX Register(PB0A2MAX) 0xB030B038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB0_ADDR2_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB0_ADDR2_MAX															

Field	Name	RW	Reset	Description
31-0	PB0_ADDR2_MAX	R/W	0x0	Upper Address for Prefetch Address Space 2 for PB0

Prefetch Buffer0 (ARM Data) Address3 MIN Register(PB0A3MIN) 0xB030B03C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB0_ADDR3_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB0_ADDR3_MIN															

Field	Name	RW	Reset	
31-0	PB0_ADDR3_MIN	R/W	0x0	Lower Address for Prefetch Address Space 3 for PB0

Prefetch Buffer0 (ARM Data) Address3 MAX Register(PB0A3MAX) 0xB030B040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB0_ADDR3_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB0_ADDR3_MAX															

Field	Name	RW	Reset	Description
31-0	PB0_ADDR3_MAX	R/W	0x0	Upper Address for Prefetch Address Space 3 for PB0

Prefetch Buffer1 (IO Bus) Enable Register(PB1EN) 0xB030B044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												EN3	EN2	EN1	EN0

Field	Name	RW	Reset	Description
3	EN3	R/W	0x0	Prefetch Address Space3 Enable 0 : Disable 1 : Enable
2	EN2	R/W	0x0	Prefetch Address Space2 Enable 0 : Disable 1 : Enable
1	EN1	R/W	0x0	Prefetch Address Space1 Enable 0 : Disable 1 : Enable
0	EN0	R/W	0x0	Prefetch Address Space0 Enable 0 : Disable 1 : Enable

Prefetch Buffer 1 (IO Bus) Address0 MIN Register(PB1A0MIN) 0xB030B048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB1_ADDR0_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB1_ADDR0_MIN															

Field	Name	RW	Reset	Description
31-0	PB1_ADDR0_MIN	R/W	0x0	Lower Address for Prefetch Address Space 0 for PB1

Prefetch Buffer 1 (IO Bus) Address0 MAX Register(PB1A0MAX) 0xB030B04C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB1_ADDR0_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB1_ADDR0_MAX															

Field	Name	RW	Reset	Description
31-0	PB0_ADDR0_MAX	R/W	0x0	Upper Address for Prefetch Address Space 0 for PB1

Prefetch Buffer 1 (IO Bus) Address1 MIN Register(PB1A1MIN) 0xB030B050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB1_ADDR1_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB1_ADDR1_MIN															

Field	Name	RW	Reset	Description
31-0	PB1_ADDR0_MIN	R/W	0x0	Lower Address for Prefetch Address Space 1 for PB1

Prefetch Buffer 1 (IO Bus) Address1 MAX Register(PB1A1MAX) 0xB030B054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB1_ADDR1_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB1_ADDR1_MAX															

Field	Name	RW	Reset	Description
31-0	PB1_ADDR1_MAX	R/W	0x0	Upper Address for Prefetch Address Space 1 for PB1

Prefetch Buffer 1 (IO Bus) Address2 MIN Register(PB1A2MIN) 0xB030B058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB1_ADDR2_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB1_ADDR2_MIN															

Field	Name	RW	Reset	Description
31-0	PB1_ADDR2_MIN	R/W	0x0	Lower Address for Prefetch Address Space 2 for PB1

Prefetch Buffer 1 (IO Bus) Address2 MAX Register(PB1A2MAX) 0xB030B05C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB1_ADDR2_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB1_ADDR2_MAX															

Field	Name	RW	Reset	Description
31-0	PB1_ADDR2_MAX	R/W	0x0	Upper Address for Prefetch Address Space 2 for PB1

Prefetch Buffer 1 (IO Bus) Address3 MIN Register(PB1A3MIN) 0xB030B060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB1_ADDR3_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB1_ADDR3_MIN															

Field	Name	RW	Reset	Description
31-0	PB1_ADDR3_MIN	R/W	0x0	Lower Address for Prefetch Address Space 3 for PB1

Prefetch Buffer 1 (IO Bus) Address3 MAX Register(PB1A3MAX) 0xB030B064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB1_ADDR3_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB1_ADDR3_MAX															

Field	Name	RW	Reset	Description
31-0	PB1_ADDR3_MAX	R/W	0x0	Upper Address for Prefetch Address Space 3 for PB1

Prefetch Buffer1 (HSIO Bus) Enable Register(PB2EN) 0xB030B044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												EN3	EN2	EN1	EN0

Field	Name	RW	Reset	Description
3	EN3	R/W	0x0	Prefetch Address Space3 Enable 0 : Disable 1 : Enable
2	EN2	R/W	0x0	Prefetch Address Space2 Enable 0 : Disable 1 : Enable
1	EN1	R/W	0x0	Prefetch Address Space1 Enable 0 : Disable 1 : Enable
0	EN0	R/W	0x0	Prefetch Address Space0 Enable 0 : Disable 1 : Enable

Prefetch Buffer 2 (HSIO Bus) Address0 MIN Register(PB2A0MIN) 0xB030B048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB2_ADDR0_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB2_ADDR0_MIN															

Field	Name	RW	Reset	Description
31-0	PB2_ADDR0_MIN	R/W	0x0	Lower Address for Prefetch Address Space 0 for PB2

Prefetch Buffer 2 (HSIO Bus) Address0 MAX Register(PB2A0MAX) 0xB030B04C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB2_ADDR0_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB2_ADDR0_MAX															

Field	Name	RW	Reset	Description
31-0	PB2_ADDR0_MAX	R/W	0x0	Upper Address for Prefetch Address Space 0 for PB2

Prefetch Buffer 2 (HSIO Bus) Address1 MIN Register(PB2A1MIN) 0xB030B050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB2_ADDR1_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB2_ADDR1_MIN															

Field	Name	RW	Reset	Description
31-0	PB2_ADDR1_MIN	R/W	0x0	Lower Address for Prefetch Address Space 1 for PB2

Prefetch Buffer 2 (HSIO Bus) Address1 MAX Register(PB2A1MAX) 0xB030B054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB2_ADDR1_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB2_ADDR1_MAX															

Field	Name	RW	Reset	Description
31-0	PB2_ADDR1_MAX	R/W	0x0	Upper Address for Prefetch Address Space 1 for PB2

Prefetch Buffer 2 (HSIO Bus) Address2 MIN Register(PB2A2MIN) 0xB030B058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB2_ADDR2_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB2_ADDR2_MIN															

Field	Name	RW	Reset	Description
31-0	PB2_ADDR2_MIN	R/W	0x0	Lower Address for Prefetch Address Space 2 for PB2

Prefetch Buffer 2 (HSIO Bus) Address2 MAX Register(PB2A2MAX) 0xB030B05C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB2_ADDR2_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB2_ADDR2_MAX															

Field	Name	RW	Reset	Description
31-0	PB2_ADDR2_MAX	R/W	0x0	Upper Address for Prefetch Address Space 2 for PB2

Prefetch Buffer 2 (HSIO Bus) Address3 MIN Register(PB2A3MIN) 0xB030B060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB2_ADDR3_MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB2_ADDR3_MIN															

Field	Name	RW	Reset	Description
31-0	PB2_ADDR3_MIN	R/W	0x0	Lower Address for Prefetch Address Space 3 for PB2

Prefetch Buffer 2 (HSIO Bus) Address3 MAX Register(PB2A3MAX) 0xB030B064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PB2_ADDR3_MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PB2_ADDR3_MAX															

Field	Name	RW	Reset	Description
31-0	PB2_ADDR3_MAX	R/W	0x0	Upper Address for Prefetch Address Space 3 for PB2

9 Internal Memory BUS Configuration

9.1 Register Description

Table 9.1 Memory Controller Register Map (Base Address = 0xB0307000)

Name	Address	Type	Reset	Description
REGION0	0x00	R/W	0x0	Region Configuration Register 0
REGION1	0x04	R/W	0x0	Region Configuration Register 1
REGION2	0x08	R/W	0x0	Region Configuration Register 2
REGION3	0x0C	R/W	0x0	Region Configuration Register 3
REGION4	0x10	R/W	0x0	Region Configuration Register 4
REGION5	0x14	R/W	0x0	Region Configuration Register 5
REGION6	0x18	R/W	0x0	Region Configuration Register 6
REGION7	0x1C	R/W	0x0	Region Configuration Register 7
RDWCFG	0x28	R/W	0x11	Memory RDW Configuration Register
IRAMPWRCFG	0x2C	R/W	0x0	Internal ROM Power Down Register

Region Configuration Register (REGIONx) **0xB0307000~0xB030701C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SA												-	0	-	SIZE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SIZE				AP		EN	DO				-	C	B	10b	

Field	Name	RW	Reset	Description
31-20	SA	R/W	0x0	Section Base Address * Defined same as ARM1176JZF-S
16-12	SIZE	R/W	0x0	Size Register 0x13 : 1MB 0x14 : 2MB … 0x1E : 2GB 0x1F : 4GB
11-10	AP	R/W	0x0	Access Permission Register * Defined same as ARM1176JZF-S
9	EN	R/W	0x0	Region Enable Register
8-5	DO	R/W	0x0	Domain Register * Defined same as ARM1176JZF-S
3	C	R/W	0x0	Cacheable Register * Defined same as ARM1176JZF-S
2	B	R/W	0x0	Bufferable Register * Defined same as ARM1176JZF-S

All the bit field except for SZ and EN are same as defined in ARM1176JZF-S technical reference manual.

The REGION7 has the highest priority and the REGION0 has the lowest priority.

After setting according register, the address of TABBASE should be written into TLB base address for ARM1176JZF-S.

Internal Memory Wait Control Register(RDWCFG) 0xB0307028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											nBRAM	nTLB	nRAM1	nRAM0	nROM

Field	Name	RW	Reset	Description
4	nBRAM	R/W	0x1	Backup RAM Read Wait Control Enable 0 : Disable 1 : Enable ***This flag should be set to "1", If $F_{mbus} > F_{mbus\_max}/2$***
3	nTLB	R/W	0x0	HW MMU Table Read Wait Control Enable 0 : Disable 1 : Enable
2	nRAM1	R/W	0x0	IRAM1 Wait Read Control Enable 0 : Disable 1 : Enable ***This flag should be set to "1", If $F_{mbus} > F_{mbus\_max}/2$***
1	nRAM0	R/W	0x0	IRAM0 Wait Read Control Enable 0 : Disable 1 : Enable ***This flag should be set to "1", If $F_{mbus} > F_{mbus\_max}/2$***
0	nROM	R/W	0x1	Internal ROM Read Wait Control Enable 0 : Disable 1 : Enable ***This flag should be set to "1", If $F_{mbus} > F_{mbus\_max}/2$***

Internal ROM Power Down Register(IROMPWRCFG) 0xB030702C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														PDN	SLN

Field	Name	RW	Reset	Description
1	PDN	R/W	0x0	Internal ROM Power Down mode 0 : Normal 1 : Power Down Enable
0	SLN	R/W	0x0	Internal ROM Sleep mode 0 : Normal 1 : Sleep Enable

PART5 – IO BUS
NVS2310

Rev. 1.02

Jun 01, 2011

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial Release
2011-04-22	1.01	∗ Correct the register table and document format. ∗ Correct register table - GPIO Pull up/down register and ECID. ∗ Correct IOBAPB and IO_MCFG address. ∗ Correct Audio Registers.
2011-06-01	1.02	∗ Correct the error of bit description in Line Control Register.

TABLE OF CONTENTS

Contents

Figures

Tables

1 Introduction

NVS2310 IOBUS provides connections between the internal bus master and external device controller. Also on chip peripherals are located in it. The sort of interfaces supported is as follows : storage interface, audio interface, high speed interface, miscellaneous interface.

[Features]

- STORAGE INTERFACES
  - Memory Card Interfaces for SD/MMC
    - 1/4 bit SDIO, SDMA, ADMA mode
    - 8bit MMC, CE-ATA mode
  - NAND flash controller
  - SRAM interface
- AUDIO INTERFACES
  - AUDIO I/O with DMA master
    - AHB master operation
    - I2S Master & Slave Interface
    - S/PDIF RX/TX Interface
  - AUDIO I/O with internal DMA support
    - AHB slave operation
    - Support interfaces to combine with internal DMA controller
    - I2S Master & Slave Interface
    - S/PDIF TX Interface
- HIGH SPEED INTERFACES
  - High speed USB2.0 FS/HS OTG controller
  - 2-Channel host port interface via 80/68 compatible – 8 / 16 bits
- MISCELLANEOUS INTERFACE
  - 6-Channel GPSB with TS interface for supporting various serial interfaces
  - 2-Channel dedicated TS parallel interface
  - Remote control interface
  - Configurable 9-Channel I2C
    - 6-Channel Masters
    - 3 Channel I2C Slave
  - 6-Channel UART for serial host interface
    - 3-channel without hardware flow control
    - 3-channel with hardware flow control
  - Pulse Width Modulation
  - Vectored Priority Interrupt Controller
    - 64 interrupt sources
  - External host interface
- ON-CHIP PERIPHERALS
  - Three 3 Channel DMA for transferring bulk data
    - Including Hardware DMA Request for Each Channel
  - Touch Screen ADC controller
    - 10/12 bit ADC
  - General purpose I/O

2 Bus Architecture

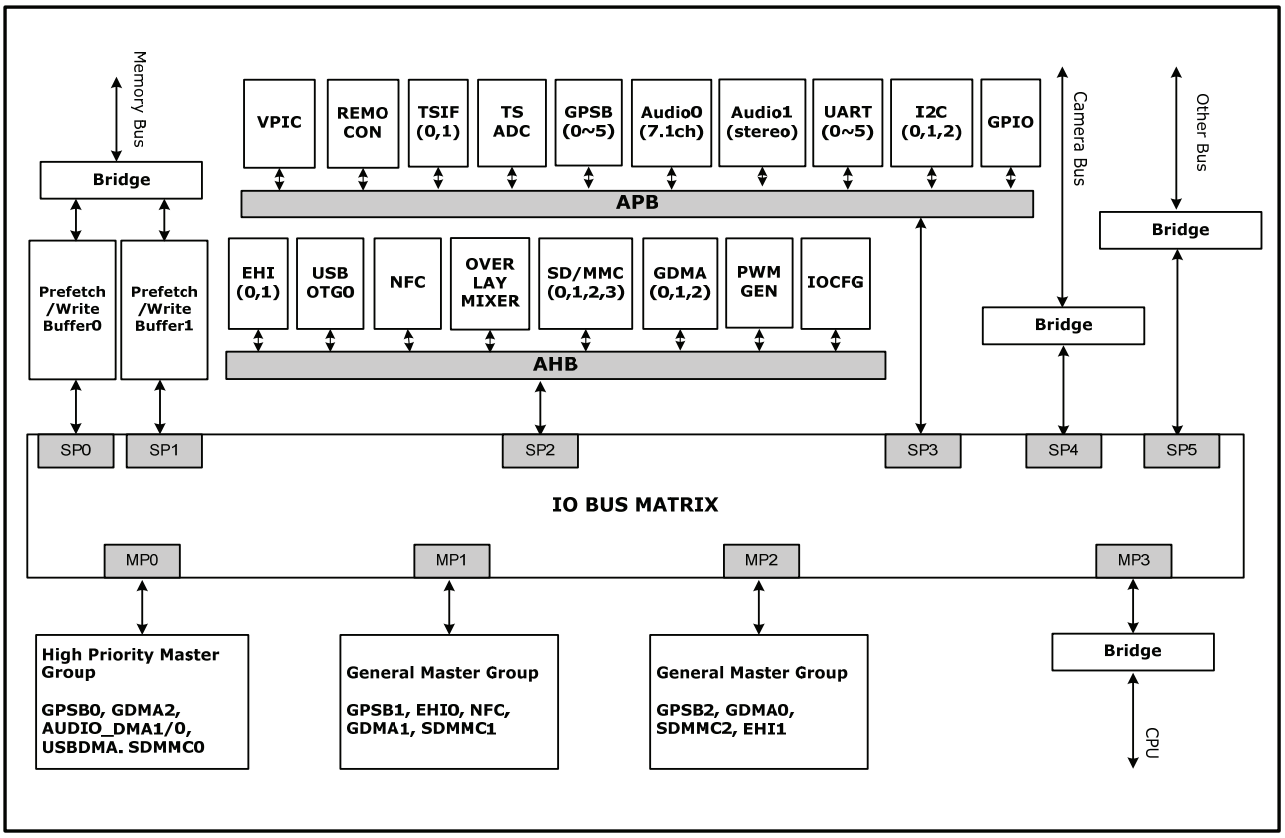


Figure 2.1 The I/O Hardware Bus Architecture

3 Address and Register Map

The NVS2310 has various peripherals for specific interface controllers or on-chip hardware components. These peripherals can be configured appropriately by its own registers that can be accessed through specially allocated address. These address maps are represented in the following table.

Refer to corresponding sections for detail information of each peripheral.

Base Address		Peripherals
0xB0000000		External Host Interface 0
0xB0010000		External Host Interface 1
0xB0020000	0xB0020000	SD/MMC Controller 0
	0xB0022000	SD/MMC Controller 1
	0xB0024000	SD/MMC Controller 2
	0xB0026000	SD/MMC Controller 3
0xB0030000	0xB0030000	Central DMA Controller0
	0xB0030100	Central DMA Controller1
	0xB0030200	Central DMA Controller2
	0xB0030400	Central DMA IRQ
0xB0040000		USB2.0 OTG Controller
0xB0050000		NAND Flash Controller
0xB0060000		PWM Controller
0xB0070000		Overlay Mixer
0xB0080000		IOBUS Configuration Registers
0xB0100000		TSADC Controller
0xB0101000		Remote Control Interface
0xB0102000	0xB0102000	UART Channel 0 with DMA support
	0xB0102100	UART Channel 1 with DMA support
	0xB0102200	UART Channel 2 with DMA support
	0xB0102300	UART Channel 3 with DMA support
	0xB0102400	UART Channel 4 with DMA support
	0xB0102500	UART Channel 5 with DMA support
	0xB0102600	UART Port Configuration Registers
0xB0103000		Audio DMA Controller0
0xB0104000		Audio DAI Controller0
0xB0105000		Audio DMA Controller1
0xB0106000		Audio DAI Controller1
0xB0107000	0xB0107000	GPSB channel 0 without DMA support
	0xB0107100	GPSB channel 1 without DMA support
	0xB0107200	GPSB channel 2 without DMA support
	0xB0107300	GPSB channel 3 with DMA support
	0xB0107400	GPSB channel 4 with DMA support
	0xB0107500	GPSB channel 5 with DMA support
	0xB0107600	GPSB Port Configuration Registers
	0xB0107700	GPSB PID Table
0xB0108000	0xB0108000	TS Interface 0
	0xB0108100	TS Interface 1
	0xB0108200	TS Interface Port Configuration Registers
0xB0109000	0xB0109000	I2C Controller0
	0xB0109100	I2C Controller1
	0xB0109200	I2C Controller2
0xB010A000		GPIO Controller
0xB010B000		SPDIF0 Tx/Rx Controller
Reserved		
0xB010D000		SPDIF1 Tx Controller
0xB0600000		VPIC

4 SD/SDIO/MMC Host Controller

4.1 Overview

The NVS2310 complies with following versions of specification concerned.

- SD Host Controller Specification Version 3.0 Draft 0.61
- SDIO Card Specification Version 2.0
- SD Memory Card Specification Draft Version 3.0
- SD Memory Card Security Specification Version 1.01
- MMC Specification Version 3.31, 4.2 and 4.4

There are four host controllers in NVS2310 and each of them has one slot in it. For more effective data transaction, a 2K dual port FIFO for one host controller is used. You may access those using inherent SDMA and ADMA. Non-DMA access is also available.
NVS2310 can support following SD cards.
- For Capacity of Memory
  1) SDSC : Up to and including 2GB
  2) SDHC : More than 2GB and up to and including 32GB
  3) SDXC : More than 32GB and up to and including 2TB
- For Bus Speed Mode
  1) Default Speed : 3.3V signaling, Frequency up to 25MHz
  2) High Speed : 3.3V signaling, Frequency up to 50MHz
  3) SDR12 : 1.8V signaling, Frequency up to 25MHz
  4) SDR25 : 1.8V signaling, Frequency up to 50MHz
  5) SDR50 : 1.8V signaling, Frequency up to 100MHz
  6) SDR104 : 1.8V signaling, Frequency up to 208MHz
  7) DDR50 : 1.8V signaling, Frequency up to 50MHz

and following MMC cards.
- 1, 4, 8 bit SDR mode and 1, 4, 8 bit DDR mode
- MMC Plus and MMC Mobile
- Normal boot and Alternative boot mode

When the host controller runs under SDR12, SDR50, SDR104 and DDR50 mode, signal voltage switching 3.3V to 1.8V is needed. Switching sequence is described at **1.5.2.**
Refer to the SD Host Controller Standard Specification V3.0 of SD Association to know more specific explanation.

4.2 Block Diagram of Each Host Controller

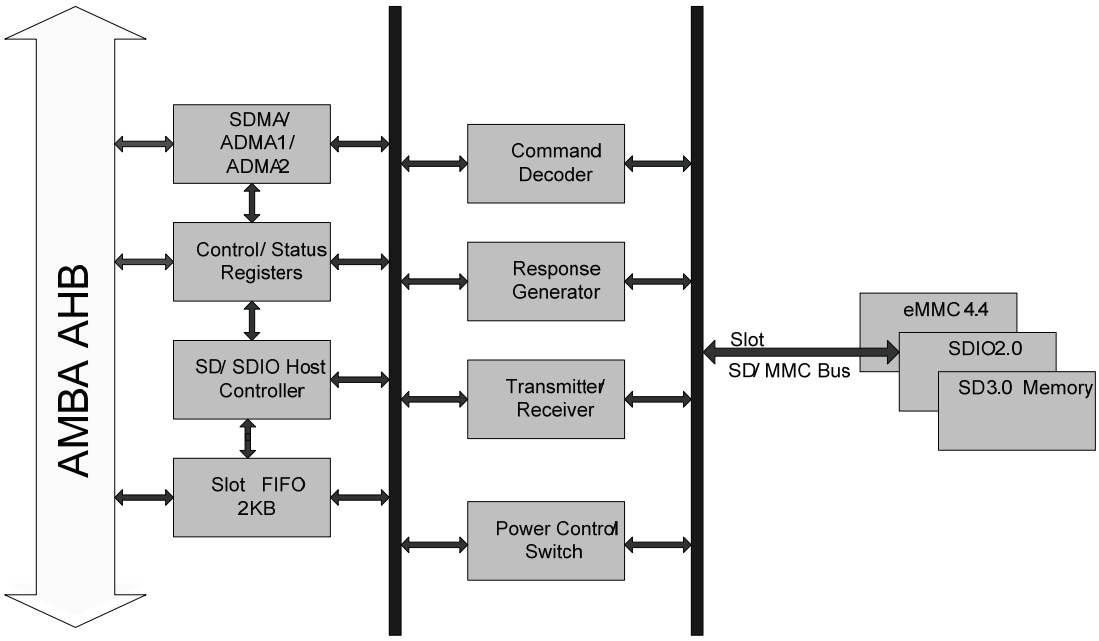


Figure 4.1 SD/SDIO/MMC Block Diagram

4.3 Register Description

Table 4.1 Register Map (Base Address = 0xB0020000)

Name	Address	Type	Reset	Description
SDMA	0x000	R/W	0x0000	SDMA System Address
BSIZE	0x004	R/W	0x0000	Block Size
BCNT	0x006	R/W	0x0000	Block Count
ARG	0x008	R/W	0x0000	Argument
TMODE	0x00C	R/W	0x0000	Transfer Mode
CMD	0x00E	R/W	0x0000	Command
RESP0	0x010	R	0x0000	Response0
RESP1	0x012	R	0x0000	Response1
RESP2	0x014	R	0x0000	Response2
RESP3	0x016	R	0x0000	Response3
RESP4	0x018	R	0x0000	Response4
RESP5	0x01A	R	0x0000	Response5
RESP6	0x01C	R	0x0000	Response6
RESP7	0x01E	R	0x0000	Response7
DATAL	0x020	R/W	-	Buffer Data Port(Low)
DATAH	0x022	R/W	-	Buffer Data Port(High)
STATEL	0x024	R	0x0000	Present State(Low)
STATEH	0x026	R	0x0000	Present State(High)
CONTL	0x028	R/W	0x0000	Power Control / Host Control
CONTH	0x02A	R/W	0x0000	Wakeup Control / Block Gap Control
CLK	0x02C	R/W	0x0000	Clock Control
TIME	0x02E	R/W	0x0000	Software Reset / Timeout Control
STSL	0x030	R	0x0000	Normal Interrupt Status
STSH	0x032	R	0x0000	Error Interrupt Status
STSENL	0x034	R/W	0x0000	Normal Interrupt Status Enable
STSENH	0x036	R/W	0x0000	Error Interrupt Status Enable
INTENL	0x038	R/W	0x0000	Normal Interrupt Signal Enable
INTENH	0x03A	R/W	0x0000	Error Interrupt Signal Enable
CMD12ERR	0x03C	R	0x0000	Auto CMD12 Error Status
CONT2	0x03E	R/W	0x0000	Host Control2
CAPL	0x040	R	0x30B0	Capabilities(Low)
CAPH	0x042	R	0x69EF	Capabilities(High)
CAPL2	0x044	R	0x0000	Capabilities2(Low)
CURL	0x048	R	0x0001	Maximum Current Capabilities(Low)
CURH	0x04A	R	0x0000	Maximum Current Capabilities(High)
FORCEL	0x050	W	0x0000	Force event for AutoCmd Error Status
FORCEH	0x052	W	0x0000	Force event for Error Interrupt Status
ADMAERR	0x054	R/W	0x0000	ADMA Error Status
ADDR0	0x058	R/W	0x0000	ADMA Address[15:0]
ADDR1	0x05A	R/W	0x0000	ADMA Address[31:16]
ADDR2	0x05C	R/W	0x0000	ADMA Address[47:32]
ADDR3	0x05E	R/W	0x0000	ADMA Address[63:48]
BTCONTL	0x070	R/W	0x0000	Boot data timeout control(Low)
BTCONTH	0x072	R/W	0x0000	Boot data timeout control(High)
DBGSEL	0x074	R/W	0x0000	Debug Selection
SPIINT	0x0F0	R	0x0000	SPI Interrupt Support
SLOT	0x0FC	R	0x0000	Slot Interrupt Status
VERSION	0x0FE	R	0x0002	Host Controller Version

The address map of registers is arranged by the half word(16bits) to specify those functions. But The NVS2310 accesses data by the word(32bits). NVS2310 has four Host Controllers. They have the same register map except their own base address. Table 4.2 shows base address of each Host Controller.

Table 4.2 Base Address of Each Slot

Host Controller Number	Base Address
0	0xB002_0000
1	0xB002_0200
2	0xB002_0400
3	0xB002_0600

Table 4.3 Channel Control Register Map (Base Address = 0xB0020800)

Name	Address	Type	Reset	Description
SDCTRL	0x00	R/W	0x0000	Host Controller Control Register
SD0CMDDAT	0x04	R/W	0x0000	SD/MMC0 output delay control register
SD1CMDDAT	0x08	R/W	0x0000	SD/MMC1 output delay control register
SD2CMDDAT	0x0C	R/W	0x0000	SD/MMC2 output delay control register
SD3CMDDAT	0x10	R/W	0x0000	SD/MMC3 output delay control register
SD0PRESET1	0x14	R/W	0x0000	SD/MMC0 Preset Register1
SD0PRESET2	0x18	R/W	0x0000	SD/MMC0 Preset Register2
SD0PRESET3	0x1C	R/W	0x0000	SD/MMC0 Preset Register3
SD0PRESET4	0x20	R/W	0x0000	SD/MMC0 Preset Register4
SD0PRESET5	0x24	R/W	0x0000	SD/MMC0 Preset Register5
SD0PRESET6	0x28	R/W	0x0000	SD/MMC0 Preset Register6
SD1PRESET1	0x2C	R/W	0x0000	SD/MMC1 Preset Register1
SD1PRESET2	0x30	R/W	0x0000	SD/MMC1 Preset Register2
SD1PRESET3	0x34	R/W	0x0000	SD/MMC1 Preset Register3
SD1PRESET4	0x38	R/W	0x0000	SD/MMC1 Preset Register4
SD1PRESET5	0x3C	R/W	0x0000	SD/MMC1 Preset Register5
SD1PRESET6	0x40	R/W	0x0000	SD/MMC1 Preset Register6
SD2PRESET1	0x44	R/W	0x0000	SD/MMC2 Preset Register1
SD2PRESET2	0x48	R/W	0x0000	SD/MMC2 Preset Register2
SD2PRESET3	0x4C	R/W	0x0000	SD/MMC2 Preset Register3
SD2PRESET4	0x50	R/W	0x0000	SD/MMC2 Preset Register4
SD2PRESET5	0x54	R/W	0x0000	SD/MMC2 Preset Register5
SD2PRESET6	0x58	R/W	0x0000	SD/MMC2 Preset Register6
SD3PRESET1	0x5C	R/W	0x0000	SD/MMC3 Preset Register1
SD3PRESET2	0x60	R/W	0x0000	SD/MMC3 Preset Register2
SD3PRESET3	0x64	R/W	0x0000	SD/MMC3 Preset Register3
SD3PRESET4	0x68	R/W	0x0000	SD/MMC3 Preset Register4
SD3PRESET5	0x6C	R/W	0x0000	SD/MMC3 Preset Register5
SD3PRESET6	0x70	R/W	0x0000	SD/MMC3 Preset Register6

SDCTRL **0xB0020800**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CD3	CD2	CD1	CD0	WP3	WP2	WP1	WP0	R				TRQ3	TRQ2	TRQ1	TRQ0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DDRTAP3				DDRTAP2				DDRTAP1				DDRTAP0			

Field	Name	RW	Reset	Description
31	CD3	R/W	0x0	Card Detection for SLOT 3 This bit is connected to card detection signal of SLOT3
30	CD2	R/W	0x0	Card Detection for SLOT 2 This bit is connected to card detection signal of SLOT2
29	CD1	R/W	0x0	Card Detection for SLOT 1 This bit is connected to card detection signal of SLOT1
28	CD0	R/W	0x0	Card Detection for SLOT 0 This bit is connected to card detection signal of SLOT0
27	WP3	R/W	0x0	Write Protect for SLOT 3 This bit is connected to write protect signal of SLOT3
26	WP2	R/W	0x0	Write Protect for SLOT 2 This bit is connected to write protect signal of SLOT2
25	WP1	R/W	0x0	Write Protect for SLOT 1 This bit is connected to write protect signal of SLOT1
24	WP0	R/W	0x0	Write Protect for SLOT 0 This bit is connected to write protect signal of SLOT0
23-15	-	-	-	Reserved
14-12	SLOT3	R/W	0x0	These bits specify what ports are used for SLOT3. 0-7: Port Number
11	-	-	-	Reserved
10-8	SLOT2	R/W	0	These bits specify what ports are used for SLOT2. 0-7: Port Number
7	-	-	-	Reserved
6-4	SLOT1	R/W	0	These bits specify what ports are used for SLOT1. 0-7: Port Number
3	-	-	-	Reserved
2-0	SLOT0	R/W	0	These bits specify what ports are used for SLOT0. 0-7: Port Number

SD0CMDDAT **0xB0020804**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R		DLCMD0EN						R		DLCMD0					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R		DLDAT0EN						R		DLDAT0					

Field	Name	RW	Reset	Description
31-30	R	R	0	Reserved
29-24	DLCMD0EN	R/W	0	1 ~ 63 : Adjust delay timing of CMD0EN output 0 : Disable
23-22	R	R	0	Reserved
21-16	DLCMD0	R/W	0	1 ~ 63 : Adjust delay timing of CMD0 output 0 : Disable
15-14	R	R	0	Reserved
13-8	DLDAT0EN	R/W	0	1 ~ 63 : Adjust delay timing of DAT0EN output 0 : Disable
7-6	R	R	0	Reserved
5-0	DLDAT0	R/W	0	1 ~ 63 : Adjust delay timing of DAT0 output 0 : Disable

SD1CMDDAT 0xB0020808

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R		DLCMD1EN						R		DLCMD1					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R		DLDAT1EN						R		DLDAT1					

Field	Name	RW	Reset	Description
31-30	R	R	0	Reserved
29-24	DLCMD1EN	R/W	0	1 ~ 63 : Adjust delay timing of CMD1EN output 0 : Disable
23-22	R	R	0	Reserved
21-16	DLCMD1	R/W	0	1 ~ 63 : Adjust delay timing of CMD1 output 0 : Disable
15-14	R	R	0	Reserved
13-8	DLDAT1EN	R/W	0	1 ~ 63 : Adjust delay timing of DAT1EN output 0 : Disable
7-6	R	R	0	Reserved
5-0	DLDAT1	R/W	0	1 ~ 63 : Adjust delay timing of DAT1 output 0 : Disable

SD2CMDDAT 0xB002080C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R		DLCMD2EN						R		DLCMD2					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R		DLDAT2EN						R		DLDAT2					

Field	Name	RW	Reset	Description
31-30	R	R	0	Reserved
29-24	DLCMD2EN	R/W	0	1 ~ 63 : Adjust delay timing of CMD2EN output 0 : Disable
23-22	R	R	0	Reserved
21-16	DLCMD2	R/W	0	1 ~ 63 : Adjust delay timing of CMD2 output 0 : Disable
15-14	R	R	0	Reserved
13-8	DLDAT2EN	R/W	0	1 ~ 63 : Adjust delay timing of DAT2EN output 0 : Disable
7-6	R	R	0	Reserved
5-0	DLDAT2	R/W	0	1 ~ 63 : Adjust delay timing of DAT2 output 0 : Disable

SD3CMDDAT 0xB0020810

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R		DLCMD3EN						R		DLCMD3					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R		DLDAT3EN						R		DLDAT3					

Field	Name	RW	Reset	Description
31-30	R	R	0	Reserved
29-24	DLCMD3EN	R/W	0	1 ~ 63 : Adjust delay timing of CMD3EN output 0 : Disable
23-22	R	R	0	Reserved
21-16	DLCMD3	R/W	0	1 ~ 63 : Adjust delay timing of CMD3 output 0 : Disable
15-14	R	R	0	Reserved
13-8	DLDAT3EN	R/W	0	1 ~ 63 : Adjust delay timing of DAT3EN output 0 : Disable
7-6	R	R	0	Reserved
5-0	DLDAT3	R/W	0	1 ~ 63 : Adjust delay timing of DAT3 output 0 : Disable

SD0PRESET1 **0xB0020814**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CAPAREG0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPAREG0[15:0]															

Field	Name	RW	Reset	Description
31-0	CAPAREG0	R/W	0	Capabilities1& 2 register can be used only for reading. For flexibility, this register can change contents of Capabilities1 register. Following bits influence those of Capabilities1 register. 31 => DDR50 Support 30 => SDR104 Support 29 => SDR50 Support 28:27 => Slot Type 26 => Asynchronous Interrupt Support 25 => 64bit System Bus Support 24 => Voltage Support 1.8V 23 => Voltage Support 3.0V 22 => Voltage Support 3.3V 21 => Suspend / Resume Support 20 => SDMA Support 19 => High Speed Support 18 => ADMA2 Support 17 => Extended Media Bus Support 16:15 => Max Block Length 14:7 => Base Clock Frequency For SD Clock 6 => Timeout Clock Unit 5:0 => Timeout Clock Frequency

SD0PRESET2 **0xB0020818**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												CAPAREG0[51:48]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPAREG0[47:32]															

Field	Name	RW	Reset	Description
31-20	R	R	0	Reserved
19-0	CAPAREG0	R/W	0	Capabilities2 register can be used only for reading. For flexibility, this register can change contents of Capabilities1 register. Following bits influence those of Capabilities1 register. 19 => SPI Block Mode 18 => SPI Mode 17:10 => Clock Multiplier 9:8 => Re-Tuning Modes 7 => Use Tuning for SDR50 6:3 => Timer Count for Re-Tuning 2 => Driver Type D Support 1 => Driver Type C Support 0 => Driver Type A Support

SD0PRESET3 0xB002081C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			INITAL0[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			DSPEED0[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	INITIAL0	R/W	0	Initialization
12-0	DSPEED0	R/W	0	Default Speed

SD0PRESET4 0xB0020820

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			HSPEED0[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			SDR12P0[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	HSPEED0	R/W	0	High Speed
12-0	SDR12P0	R/W	0	SDR12 Preset Value

SD0PRESET5 0xB0020824

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			SDR25P0[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			SDR50P0[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	SDR25P0	R/W	0	SDR25 Preset Value
12-0	SDR50P0	R/W	0	SDR50 Preset Value

SD0PRESET6 0xB0020828

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			SDR104P0[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			DDR50P0[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	SDR104P0	R/W	0	SDR104 Preset Value
12-0	DDR50P0	R/W	0	DDR50 Preset Value

SD1PRESET1 0xB002082C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CAPAREG1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPAREG1[15:0]															

Field	Name	RW	Reset	Description
31-0	CAPAREG1	R/W	0	Provides the Host Driver with information specific to the Host Controller implementation. Bit31~0

SD1PRESET2 0xB0020830

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												CAPAREG1[51:48]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPAREG1[47:32]															

Field	Name	RW	Reset	Description
31-20	R	R	0	Reserved
19-0	CAPAREG1	R/W	0	Provides the Host Driver with information specific to the Host Controller implementation. Bit51~32

SD1PRESET3 0xB0020834

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			INITAL1[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			DSPEED1[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	INITIAL1	R/W	0	Initialization
12-0	DSPEED1	R/W	0	Default Speed

SD1PRESET4 0xB0020838

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			HSPEED1[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			SDR12P1[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	HSPEED1	R/W	0	High Speed
12-0	SDR12P1	R/W	0	SDR12 Preset Value

SD1PRESET5 0xB002083C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			SDR25P1[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			SDR50P1[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	SDR25P1	R/W	0	SDR25 Preset Value
12-0	SDR50P1	R/W	0	SDR50 Preset Value

SD1PRESET6 0xB0020840

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			SDR104P1[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			DDR50P1[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	SDR104P1	R/W	0	SDR104 Preset Value
12-0	DDR50P1	R/W	0	DDR50 Preset Value

SD2PRESET1 0xB0020844

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CAPAREG2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPAREG2[15:0]															

Field	Name	RW	Reset	Description
31-0	CAPAREG2	R/W	0	Provides the Host Driver with information specific to the Host Controller implementation. Bit31~0

SD2PRESET2 0xB0020848

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												CAPAREG0[51:48]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPAREG2[47:32]															

Field	Name	RW	Reset	Description
31-20	R	R	0	Reserved
19-0	CAPAREG2	R/W	0	Provides the Host Driver with information specific to the Host Controller implementation. Bit51~32

SD2PRESET3 0xB002084C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			INITAL2[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			DSPEED2[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	INITIAL2	R/W	0	Initialization
12-0	DSPEED2	R/W	0	Default Speed

SD2PRESET4 0xB0020850

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			HSPEED2[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			SDR12P2[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	HSPEED2	R/W	0	High Speed
12-0	SDR12P2	R/W	0	SDR12 Preset Value

SD2PRESET5 0xB0020854

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			SDR25P2[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			SDR50P2[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	SDR25P2	R/W	0	SDR25 Preset Value
12-0	SDR50P2	R/W	0	SDR50 Preset Value

SD2PRESET6 **0xB0020858**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			SDR104P2[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			DDR50P2[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	SDR104P2	R/W	0	SDR104 Preset Value
12-0	DDR50P2	R/W	0	DDR50 Preset Value

SD3PRESET1 **0xB002085C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CAPAREG3[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPAREG3[15:0]															

Field	Name	RW	Reset	Description
31-0	CAPAREG3	R/W	0	Provides the Host Driver with information specific to the Host Controller implementation. Bit31~0

SD3PRESET2 **0xB0020860**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												CAPAREG3[51:48]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAPAREG3[47:32]															

Field	Name	RW	Reset	Description
31-20	R	R	0	Reserved
19-0	CAPAREG3	R/W	0	Provides the Host Driver with information specific to the Host Controller implementation. Bit51~32

SD3PRESET3 **0xB0020864**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			INITAL3[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			DSPEED3[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	INITIAL3	R/W	0	Initialization
12-0	DSPEED3	R/W	0	Default Speed

SD3PRESET4 **0xB0020868**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			HSPEED3[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			SDR12P3[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	HSPEED3	R/W	0	High Speed
12-0	SDR12P3	R/W	0	SDR12 Preset Value

SD3PRESET5 0xB002086C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			SDR25P3[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			SDR50P3[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	SDR25P3	R/W	0	SDR25 Preset Value
12-0	SDR50P3	R/W	0	SDR50 Preset Value

SD3PRESET6 0xB0020870

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			SDR104P3[12:0]												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			DDR50P3[12:0]												

Field	Name	RW	Reset	Description
31-29 15-13	R	R	0	Reserved
28-16	SDR104P3	R/W	0	SDR104 Preset Value
12-0	DDR50P3	R/W	0	DDR50 Preset Value

SDMA System Address 0xB0020*n*[1]00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SDMA[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SDMA[15:0]															

Field	Name	RW	Reset	Description
31-0	SDMA	R/W	0	System memory address for a DMA transfer This contains system memory address for a SDMA transfer. This should be initialized before starting a SDMA transaction. After SDMA has stopped, the next system address of the next contiguous data position can be read from this register. The SDMA transfer waits at the every boundary specified by the Host SDMA Buffer Boundary in the Block Size register.

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively

Block Count and Size **0xB0020*n*[1]04**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BCNT[15:0]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BSIZE[12]	SDMABUF[2:0]			BSIZE[11:0]											

Field	Name	RW	Reset	Description
31-16	BCNT	R/W	0	This is enabled when Block Count Enable in the Transfer Mode register is set to 1 and is valid only for multiple block transfers. 0000h - Stop Count 0001h - 1 block 0002h - 2 blocks --- --- FFFFh - 65535 blocks
15, 11-0	BSIZE	R/W	0	The block size for block data transfers for CMD17, CMD18, CMD24, CMD25, and CMD53. Values ranging from 1 up to the maximum buffer size can be set 0000h - No Data Transfer 0001h - 1 Byte 0002h - 2 Bytes 0003h - 3 Bytes 0004h - 4 Bytes --- --- 01FFh - 511 Bytes 0200h - 512 Bytes --- --- 0800h - 2048 Bytes 1000h - 4096 Bytes
14-12	SDMABUF	R/W	0	These bits specify the size of contiguous buffer in the system memory. The SDMA transfer shall wait at the every boundary specified by these fields and the Host Controller generates the DMA Interrupt to request the Host Driver to update the SDMA System Address register. At the end of transfer, the Host Controller may issue or may not issue DMA Interrupt. In particular, DMA Interrupt shall not be issued after Transfer Complete Interrupt is issued. 000b - 4KB(Detects A11 Carry out) 001b - 8KB(Detects A12 Carry out) 010b - 16KB(Detects A13 Carry out) 011b - 32KB(Detects A14 Carry out) 100b - 64KB(Detects A15 Carry out) 101b -128KB(Detects A16 Carry out) 110b - 256KB(Detects A17 Carry out) 111b - 512KB(Detects A18 Carry out)

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively

Argument 0xB0020*n*[1]08

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ARG[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARG[15:0]															

Field	Name	RW	Reset	Description
31-0	ARG	R/W	0	This is specified as bit39-8 of Command-Format. See the Table 5.4

Table 4.4 Command Format

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'1'	x	x	x	'1'
Description	start bit	transmission bit	response index	Card status	CRC7	end bit

Command and Transfer Mode 0xB0020*n*[1]0C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R		CMDINDEX						CTYPE		DATSEL	CICHK	CRCHK	R	RTYPE	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								SPI	ATACMD	MS	DIR		ACMD12	BCNTEN	DMAEN

Field	Name	RW	Reset	Description
31-30 18 15-8 3	R	R	0	Reserved
29-24	CMDINDEX	R/W	0	Command Index This bit shall be set to the command number (CMD0-63, ACMD0-63).
23-22	CTYPE	R/W	0	Command Type There are three types of special commands. - Abort(11b) CMD12, CMD52 for writing 'I/O Abort' in CCCR - Resume(10b) CMD52 for writing "Function Select" in CCCR - Suspend(00b) CMD52 for writing "Bus Suspend" in CCCR - Normal(00b) all other commands
21	DATSEL	R/W	0	Data Present Select 1: This bit is set to 1 to indicate that data is present and shall be transferred using the DAT line. 0: No Data Present
20	CICHK	R/W	0	Command Index Check Enable 1: If this bit is set to 1, the HC shall check the index field in the response to see if it has the same value as the command index. 0: Disable
19	CRCHK	R/W	0	Command CRC Check Enable 1: If this bit is set to 1, the HC shall check the CRC field in the response. 0: Disable
17-16	RTYPE	R/W	0	Response Type

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

				Response Type Select 00 - No Response 01 - Response length 136 (R2) 10 - Response length 48 (R3, R4, R1, R5, R7) 11 - Response length 48 check Busy after response (R1b, R5b)
7	SPI	R/W	0	SPI Mode 1: SPI Mode 0: SD Mode
6	ATACMD	R/W	0	CMD Completion Enable for CE-ATA Device will send command completion signal for CE-ATA Device will not send command completion signal for CE-ATA
5	MS	R/W	0	Multi/Single Block Select This bit is set when issuing multiple-block transfer commands using DAT line. For any other commands, this bit shall be set to 0. 1: Multiple Block 0: Single Block
4	DIR	R/W	0	Data Transfer Direction Select 1: Read (Card to Host) 0: Write (Host to Card)
2	m	R/W	0	Auto CMD12 Enable 1: When this bit is set to 1, the HC shall issue CMD12 automatically when last block transfer is completed. 0: Disable
1	BCNTEN	R/W	0	Block Count Enable 1: enable the Block count register, which is only relevant for multiple block transfers. 0: Disable
0	DMAEN	R/W	0	DMA Enable DMA can be enabled only if DMA Support bit in the Capabilities register is set. 1: A DMA operation shall begin when the HD writes to the upper byte of Command register (0x00F). 0: No data transfer or Non-DMA data transfer

Response1/0 **0xB0020*n*[1]10**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESPONSE1[15:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESPONSE0[15:0]															

Response3/2 **0xB0020*n*[2]14**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESPONSE3[15:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESPONSE2[15:0]															

Response5/4 **0xB0020*n*[3]18**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESPONSE5[15:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESPONSE4[15:0]															

Response7/6 **0xB0020*n*[4]1C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESPONSE7[15:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESPONSE6[15:0]															

Field	Name	RW	Reset	Description
127-0	RESPONSE	R	0	command responses from the SD Bus. It consists of four 32-bit registers.

Table 4.5 Response Register

Kind of Response	Meaning of Response	Response Field	Response Register
R1, R1b(normal response)	Card Status	R[39:8]	RESPONSE[31:0]
R1b(Auto CMD12 response)	Card Status for Auto CMD12	R[39:8]	RESPONSE[127:96]
R2(CID, CSD register)	CID or CSD reg. incl.	R[127:8]	RESPONSE[119:0]
R3(OCR register)	OCR register for memory	R[39:8]	RESPONSE[31:0]
R4(OCR register)	OCR register for I/O etc	R[39:8]	RESPONSE[31:0]
R5, R5b	SDIO response	R[39:8]	RESPONSE[31:0]
R6(Published RCA response)	New published RCA[31:16] etc	R[39:8]	RESPONSE[31:0]

Buffer Data Port **0xB0020*n*[5]20**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[15:0]															

Field	Name	RW	Reset	Description
31-0	DATA	R/W	X	Buffer Data The Host Controller Buffer can be accessed through this 32-bit Data Port Register.

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.
[2] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.
[3] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.
[4] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively
[5] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

Present State **0xB0020*n*[1]24**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			DAT[7:4]				CMD	DAT[3:0]				SDWP	SDCD	CDST	CDIN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R				RDEN	WREN	RDACT	WRACT	R				RTREQ	DATACT	NODAT	NOCMD

Field	Name	RW	Reset	Description
31-29 15-12 7-4	R	R	0	Reserved
28-25 23-20	DAT	R	0xff	Data Line Signal Level This status is used to check DAT[7:0] line level to recover from errors, and for debugging.
24	CMD	R	1	CMD Line Signal Level This status is used to check CMD line level to recover from errors, and for debugging.
19	SDWP	R	0	Write Protect Switch Pin Level 1 : This bit reflects the SDWP# pin. Write enabled 0 : Write protected
18	SDCD	R	0	Card Detect Pin Level 1 : This bit reflects the inverse value of the SDCD# pin. Card present. 0 : No Card present.
17	CDST	R	0	Card State Stable 1 : If this bit is set to 1, it means the Card Detect Pin Level is stable. No Card or Inserted. 0 : Reset of Debouncing.
16	CDIN	R	0	Card Inserted 1 : This bit indicates whether a card has been inserted. 0 : Reset or Debouncing or No Card.
11	RDEN	R	0	Buffer Read Enable 1 : This status is used for non-DMA read transfers. If this bit is 1, readable data exists in the buffer. 0 : Read Disable
10	WREN	R	0	Buffer Write Enable 1 : This status is used for non-DMA read transfers. If this bit is 1, data can be written to the buffer. 0 : Write Disable
9	RDACT	R	0	Read Transfer Active 1 : This status is used for detecting completion of a read transfer. 0 : No valid data.
8	WRACT	R	0	Write Transfer Active 1 : This status indicates a write transfer is active. 0 : No valid data
3	RTREQ	R	0	Re-Tuning Request 1 : Sampling clock needs re-tuning 0 : Fixed or well tuned sampling clock
2	DATACT	R	0	DAT Line Active 1 : This bit indicates whether one of the DAT line on SD bus is in use. 0 : DAT line inactive.
1	NODAT	R	0	Command Inhibit(DAT) This status bit is generated if either the DAT Line Active or the Read transfer Active is set to 1. It cannot issue command which uses the DAT line 1 : Cannot issue command which uses the DAT line 0 : Can issue command which uses the DAT line
0	NOCMD	R	0	Command Inhibit(CMD) 1 : indicates the CMD line is in use. 0 : indicates the CMD line is not in use and the HC can issue a SD command using the CMD line.

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

Wakeup/Block Gap/Power/Host Control 0xB0020*n*[1]28

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R					WKOUT	WKIN	WKINT	R	ABTEN	BTEN	SPI	BGINT	RDWAIT	CONREQ	BGSTOP
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R			HRST	R			POW	DETSEL	DETCD	SD8	SELDMA		HS	SD4	LED

Field	Name	RW	Reset	Description
31-27 23 15-13	R	R	0	Reserved
26	WKOUT	R/W	0	Wakeup Enable On Card Removal This bit enables wakeup event via Card Removal assertion in the Normal Interrupt Status register. FN_WUS (Wake Up Support) in CIS does not affect this bit.
25	WKIN	R/W	0	Wakeup Enable On Card Insertion This bit enables wakeup event via Card Insertion assertion in the Normal Interrupt Status register. FN_WUS (Wake Up Support) in CIS does not affect this bit.
24	WKINT	R/W	0	Wakeup Enable On Card Interrupt This bit enables wakeup event via Card Interrupt assertion in the Normal Interrupt Status register. This bit can be set to 1 if FN_WUS (Wake Up Support) in CIS is set to 1.
22	ABTEN	R/W	0	Alternative Boot Enable To start boot code access in alternative mode. 1 : To start alternative boot mode access 0 : To stop alternative boot mode access
21	BTEN	R/W	0	Boot Enable To start boot code access 1 : To start boot code access 0 : To stop boot code access
20	SPI	R/W	0	SPI Mode Enable 1 : SPI mode 0 : SD mode
19	BGINT	R/W	0	Interrupt At Block Gap This bit is valid only in 4-bit mode of the SDIO card and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer.
18	RDWAIT	R/W	0	Read Wait Control If the card supports read wait, set this bit to enable use of the read wait protocol to stop read data using DAT[2] line.
17	CONREQ	R/W	0	Continue Request This bit is used to restart a transaction which was stopped using the Stop At Block Gap Request.
16	BGSTOP	R/W	0	Stop At Block Gap Request This bit is used to stop executing a transaction at the next block gap for non- DMA,SDMA and ADMA transfers. The Host Driver shall leave this bit set to 1 until the Transfer Complete is set to 1.
12	HRST	R/W	0	Hardware Reset If this bit is set, hardware reset signal will be generated for eMMC4.4 card.
11-9	R	R/W	0	Reserved
8	POW	R/W	0	SD Card Power 1 : Supply card power 0 : Stop supplying card power
7	DETSEL	R/W	0	Card Detect Signal Selection 1 : The card detect test level is selected for test

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

				purpose. 0 : SDCD# is selected for normal use
6	DETCD	R/W	0	Card Detect Test Level 1 : This bit is enabled while the Card Detect Signal Selection is set to 1 and it indicates card inserted 0 : No Card
5	SD8	R/W	0	1 : 8 bit mode is selected 0 : 8 bit mode is not selected
4-3	SELDMA	R/W	0	DMA Select One of supported DMA modes can be selected. Use of selected DMA is determined by DMA Enable of the Transfer Mode register. 00 - SDMA is selected 01 - 32-bit Address ADMA1 is selected 10 - 32-bit Address ADMA2 is selected 11 - 64-bit Address ADMA2 is selected
2	HS	R/W	0	High Speed Enable 1 : If this bit is set to 1, the HC outputs CMD line and DAT lines at the rising edge of the SD clock (up to 50 MHz) 0 : The Host Controller outputs CMD line and DAT lines at the falling edge of the SD Clock (up to 25MHz).
1	SD4	R/W	0	Data Transfer Width 1 : 4bit mode 0 : 1bit mode
0	LED	R/W	0	LED Control This bit is used to caution the user not to remove the card while the SD card is being accessed. *This will be supported in the future*.

Software Reset/Timeout/Clock Control **0xB0020*n*[1]2C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R					RSTDAT	RSTCMD	RSTALL	R				TIMEOUT[3:0]			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SDCLKSEL[7:0]								R		CGENSEL	R		SCKEN	CLKRDY	CLKEN

Field	Name	RW	Reset	Description
31-27 23-20 7-6 4-3	R	R	0	Reserved
26	RSTDAT	R/W	0	Software Reset for DAT Line Only part of data circuit is reset. The following registers and bits are cleared by this bit: Buffer Data Port Register - Buffer is cleared and Initialized. Present State register - Buffer read Enable - Buffer write Enable - Read Transfer Active - Write Transfer Active - DAT Line Active - Command Inhibit (DAT) Block Gap Control register - Continue Request - Stop At Block Gap Request

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

				Normal Interrupt Status register - Buffer Read Ready - Buffer Write Ready - Block Gap Event - Transfer Complete
25	RSTCMD	R/W	0	Software Reset for CMD Line Only part of command circuit is reset. The following registers and bits are cleared by this bit: Present State register - Command Inhibit (CMD) Normal Interrupt Status register - Command Complete
24	RSTALL	R/W	0	Software Reset for All This reset affects the entire HC except for the card detection circuit. If this bit is set to 1, the host driver should issue reset command and reinitialize the SD card.
19-16	TIMEOUT	R/W	0	Data Timeout Counter Value This value determines the interval by which DAT line time-outs are detected. 1111 - Reserved 1110 - TMCLK * 2^27 ---------- ---------- 0001 - TMCLK * 2^14 0000 - TMCLK * 2^13
15-8	SDCLKSEL	R/W	0	SDCLK Frequency Select This register is used to select the frequency of the SDCLK pin. 80h - base clock divided by 256 40h - base clock divided by 128 20h - base clock divided by 64 10h - base clock divided by 32 08h - base clock divided by 16 04h - base clock divided by 8 02h - base clock divided by 4 01h - base clock divided by 2 00h - base clock(10MHz-63MHz) According to the Physical Layer Specification, the maximum SD Clock frequency is 25 MHz in normal speed mode and 50MHz in high speed mode, and shall never exceed this limit.
5	CGENSEL	R/W	0	Clock Generator This bit is used to select the clock generator mode in SDCLK Frequency Select. This bit depends on the setting of Preset Value Enable in the Host Control2 register. If the Preset Value Enable = 0, this bit is set by Host Driver. If the Preset Value Enable = 1, this bit is automatically set to a value specified in one of Preset Value registers. 1 : Programmable Clock Mode 0 : Divided Clock Mode
2	SCKEN	R/W	0	SD Clock Enable The Host Controller shall stop SDCLK when writing this bit to 0. SDCLK Frequency Select can be changed when this bit is 0. 1 : Enable 0 : Disable
1	CLKRDY	R	0	Internal Clock Stable This bit is set to 1 when SD clock is stable after

				writing to Internal Clock Enable in this register to 1. The SD Host Driver shall wait to set SD Clock Enable until this bit is set to 1. 1 : Ready 0 : Not Ready
0	CLKEN	R/W	0	Internal Clock Enable Clock starts to oscillate when this bit is set to 1. When clock oscillation is stable, the Host Controller shall set Internal Clock Stable in this register to 1. This bit shall not affect card detection. 1 : Oscillate 0 : Stop

Normal Interrupt Status **0xB0020*n*[1]30**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VEND[1:0]		CRR	TRERR	R		ADMA	ACMD12	R	DATEND	DATCRC	DATTIME	CINDEX	CMDEND	CMDCRC	CMDTIME
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ERR	BEINT	BAR	RTE	R			CDINT	CDOUT	CDIN	RDRDY	WRRDY	DMA	BLKGAP	TDONE	CDONE

* These status bits may be cleared by writing '1's except read only registers.

Field	Name	RW	Reset	Description
27-26 11-9	R	R	0	Reserved
31-30	VEND	R/W	0	Vendor Specific Error Status Additional status bits can be defined in this register by the vendor.
29	CRR	R/W	0	CEATA Error Status Occurs when ATA command termination has occurred due to an error condition the device has encountered. 1 : Error 0 : No Error
28	TRERR	R/W	0	Target Response Error Occurs when detecting Error during DMA transaction. 1 : Error 0 : No Error
25	ADMA	R/W	0	ADMA Error This bit is set when the Host Controller detects errors during ADMA based data transfer. In addition, the Host Controller generates this Interrupt when it detects invalid descriptor data (Valid=0) at the ST_FDS state. The Host Driver may find that Valid bit is not set at the error descriptor.
24	ACMD12	R/W	0	Auto CMD12 Error This bit is set when detecting that one of the bits in Auto CMD12 Error Status register has changed from 0 to 1. This bit is set to 1, not only when the errors in Auto CMD12 occur but also when Auto CMD12 is not executed due to the previous command error.
23	R	R/W	0	Reserved
22	DATEND	R/W	0	Data End Bit Error Occurs when detecting 0 at the end bit position of read data which uses the DAT line or the end bit position of the CRC status.
21	DATCRC	R/W	0	Data CRC Error Occurs when detecting CRC error when transferring read data which uses the DAT line or when detecting the Write CRC Status having a value of other than “010”.
20	DATTIME	R/W	0	Data Timeout Error Occurs when detecting one of following timeout conditions. 1) Busy Timeout for R1b, R5b type. 2) Busy Timeout after Write CRC status 3) Write CRC status Timeout 4) Read Data Timeout
19	CINDEX	R/W	0	Command Index Error Occurs if a Command Index error occurs in the Command Response.
18	CMDEND	R/W	0	Command End Bit Error Occurs when detecting that the end bit of a command response is 0.
17	CMDCRC	R/W	0	Command CRC Error Command CRC Error is generated in two cases. 1) If a response is returned and the Command Time-out Error is set to 0, this bit is set to 1 when detecting a CRC error in the command response 2) The HC detects a CMD line conflict by monitoring the

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

				CMD line when a command is issued.
16	CMDTIME	R/W	0	Command Timeout Error This bit is set only if the no response is returned within 64 SDCLK cycles from the end bit of the command. If the Host Controller detects a CMD line conflict, this bit shall be set without waiting for 64 SD clock cycles because the command will be aborted by the Host Controller.
15	ERR	R	0	Error Interrupt If any of the bits in the Error Interrupt Status Register are set, then this bit is set. Therefore the HD can test for an error by checking this bit first.
14	BEINT	R/W	0	Boot Terminate Interrupt This status is set if the boot operation get terminated 1 : Boot operation is terminated. 0 : Boot operation is not terminated.
13	BAR	R/W	0	Boot ACK Received This status is set if the boot acknowledge is received from device. 1 : Boot ack is receivced. 0 : Boot ack is not received.
12	RTE	R	0	Re-Tuning Event This Status is set if Re-Tuning Request in the Present State register changes from 0 to 1. 1 : Re-Tuning should be performed. 0 : Re-Tuning is not required.
8	CDINT	R/W	0	Card Interrupt Card interrupt is generated. In 1-bit mode, the Host Controller shall detect the Card Interrupt without SD Clock to support wakeup. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, so there are some sample delays between the interrupt signal from the SD card and the interrupt to the Host System. It is necessary to define how to handle this delay.
7	CDOUT	R/W	0	Card Removal This status is set if the Card Inserted in the Present State register changes from 1 to 0. When the Host Driver writes this bit to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed.
6	CDIN	R/W	0	Card Insertion This status is set if the Card Inserted in the Present State register changes from 0 to 1. When the Host Driver writes this bit to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed.
5	RDRDY	R/W	0	Buffer Read Ready This status is set if the Buffer Read Enable changes from 0 to 1 and it is ready to read buffer.
4	WRRDY	R/W	0	Buffer Write Ready This status is set if the Buffer Write Enable changes from 0 to 1 and it is ready to read buffer.
3	DMA	R/W	0	DMA Interrupt This status is set if the Host Controller detects the Host DMA Buffer boundary during transfer. In case of ADMA, by setting Int field in the descriptor table, Host Controller generates this interrupt. This interrupt shall not be generated after the Transfer Complete.
2	BLKGAP	R/W	0	Block Gap Event If the Stop At Block Gap Request in the Block Gap Control register is set, this bit is set when both a read / write transaction is stopped at a block gap. If Stop At Block Gap Request is not set to 1, this bit is not set to 1. - Read Transaction: This bit is set at the falling edge of the DAT Line Active Status (When the transaction is stopped at SD Bus timing. - Write Transaction: This bit is set at the falling edge of Write Transfer Active

				Status (After getting CRC status at SD Bus timing).
1	TDONE	R/W	0	Transfer Complete This bit is set when a read / write transaction is completed. - Read Transaction: This bit is set at the falling edge of Read Transfer Active Status. This interrupt is generated in two cases. The first is when a data transfer is completed as specified by data length (After the last data has been read to the Host System). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request in the Block Gap Control register(After valid data has been read to the Host System). - Write Transaction: This bit is set at the falling edge of the DAT Line Active Status. This interrupt is generated in two cases. The first is when the last data is written to the SD card as specified by data length and the busy signal released. The second is when data transfers are stopped at the block gap by setting Stop At Block Gap Request in the Block Gap Control register and data transfers completed. (After valid data is written to the SD card and the busy signal released). - Write Transaction: This bit is set when busy is de-asserted.
0	CDONE	R/W	0	Command Complete This bit is set when get the end bit of the command response (Except Auto CMD12).

Table 4.6 Relation between Transfer Complete and Data Timeout Error

Transfer Complete	Data Timeout Error	Meaning of the status
0	0	Interrupted by another factor
0	1	Timeout occur during transfer
1	Don't Care	Command Execution complete

Table 4.7 Relation between Command Complete and Command Timeout Error

Command Complete	Command Timeout Error	Meaning of the status
0	0	Interrupted by another factor
Don't Care	1	Response not received within 64 SDCLK cycles
1	0	Response received

Normal Interrupt Status Enable **0xB0020*n*[1]34**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VEND		CERR	TRERR	R		ADMA	ACMD12	R	DATEND	DATCRC	DATTIME	CINDEX	CMDEND	CMDCRC	CMDTIME
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BEINT	BAR	RTE	R			CDINT	CDOUT	CDIN	RDRDY	WRRDY	DMA	BLKGAP	TDONE	CDONE

Field	Name	RW	Reset	Description
27-26 15 11-9	R	R	0	Reserved
31-30	VEND	R/W	0	Vendor Specific Error Status Enable 1 : Enable 0 : Disable
29	CRR	R/W	0	CEATA Error Status Enable 1 : Enable 0 : Disable
28	TRERR	R/W	0	Target Response Error Status Enable 1 : Enable 0 : Disable
25	ADMA	R/W	0	ADMA Error Status Enable 1 : Enable 0 : Disable
24	ACMD12	R/W	0	Auto CMD12 Error Status Enable 1 : Enable 0 : Disable
23	R	R/W	0	Reserved
22	DATEND	R/W	0	Data End Bit Error Status Enable 1 : Enable 0 : Disable
21	DATCRC	R/W	0	Data CRC Error Status Enable 1 : Enable 0 : Disable
20	DATTIME	R/W	0	Data Timeout Error Status Enable 1 : Enable 0 : Disable
19	CINDEX	R/W	0	Command Index Error Status Enable 1 : Enable 0 : Disable
18	CMDEND	R/W	0	Command End Bit Error Status Enable 1 : Enable 0 : Disable
17	CMDCRC	R/W	0	Command CRC Error Status Enable 1 : Enable 0 : Disable
16	CMDTIME	R/W	0	Command Timeout Error Status Enable 1 : Enable 0 : Disable
14	BEINT	R/W	0	Boot Terminate Interrupt Enable 1 : Enable 0 : Disable
13	BAR	R/W	0	Boot ACK Received Enable 1 : Enable 0 : Disable
12	RTE	R	0	Re-Tuning Event Status Enable 1 : Enable 0 : Disable
8	CDINT	R/W	0	Card Interrupt Status Enable 1 : Enable 0 : Disable
7	CDOUT	R/W	0	Card Removal Status Enable 1 : Enable

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

				0 : Disable
6	CDIN	R/W	0	Card Insertion Status Enable 1 : Enable 0 : Disable
5	RDRDY	R/W	0	Buffer Read Ready Status Enable 1 : Enable 0 : Disable
4	WRRDY	R/W	0	Buffer Write Ready Status Enable 1 : Enable 0 : Disable
3	DMA	R/W	0	DMA Interrupt Status Enable 1 : Enable 0 : Disable
2	BLKGAP	R/W	0	Block Gap Event Status Enable 1 : Enable 0 : Disable
1	TDONE	R/W	0	Transfer Complete Status Enable 1 : Enable 0 : Disable
0	CDONE	R/W	0	Command Complete Status Enable 1 : Enable 0 : Disable

Normal Interrupt Signal Enable

0xB0020*n*[1]38

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VEND		CERR	TRERR	R		ADMA	ACMD12	CLIMIT	DATEND	DATCRC	DATTIME	CINDEX	CMDEND	CMDCRC	CMDTIME
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R	BEINT	BAR	RTE	R			CDINT	CDOUT	CDIN	RDRDY	WRRDY	DMA	BLKGAP	TDONE	CDONE

Field	Name	RW	Reset	Description
27-26 15 11-9	R	R	0	Reserved
31-30	VEND	R/W	0	Vendor Specific Error Signal Enable 1 : Enable 0 : Disable
29	CRR	R/W	0	CEATA Error Signal Enable 1 : Enable 0 : Disable
28	TRERR	R/W	0	Target Response Error Signal Enable 1 : Enable 0 : Disable
25	ADMA	R/W	0	ADMA Error Signal Enable 1 : Enable 0 : Disable
24	ACMD12	R/W	0	Auto CMD12 Error Signal Enable 1 : Enable 0 : Disable
23	R	R/W	0	Reserved
22	DATEND	R/W	0	Data End Bit Error Signal Enable 1 : Enable 0 : Disable
21	DATCRC	R/W	0	Data CRC Error Signal Enable 1 : Enable 0 : Disable
20	DATTIME	R/W	0	Data Timeout Error Signal Enable 1 : Enable 0 : Disable
19	CINDEX	R/W	0	Command Index Error Signal Enable 1 : Enable 0 : Disable

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

18	CMDEND	R/W	0	Command End Bit Error Signal Enable 1 : Enable 0 : Disable
17	CMDCRC	R/W	0	Command CRC Error Signal Enable 1 : Enable 0 : Disable
16	CMDTIME	R/W	0	Command Timeout Error Signal Enable 1 : Enable 0 : Disable
14	BEINT	R/W	0	Boot Terminate Interrupt Signal Enable 1 : Enable 0 : Disable
13	BAR	R/W	0	Boot ACK Received Signal Enable 1 : Enable 0 : Disable
13	BAR	R/W	0	Boot ACK Received Signal Enable 1 : Enable 0 : Disable
12	RTE	R	0	Re-Tuning Event Signal Enable 1 : Enable 0 : Disable
8	CDINT	R/W	0	Card Interrupt Signal Enable 1 : Enable 0 : Disable
7	CDOUT	R/W	0	Card Removal Signal Enable 1 : Enable 0 : Disable
6	CDIN	R/W	0	Card Insertion Signal Enable 1 : Enable 0 : Disable
5	RDRDY	R/W	0	Buffer Read Ready Signal Enable 1 : Enable 0 : Disable
4	WRRDY	R/W	0	Buffer Write Ready Signal Enable 1 : Enable 0 : Disable
3	DMA	R/W	0	DMA Interrupt Signal Enable 1 : Enable 0 : Disable
2	BLKGAP	R/W	0	Block Gap Event Signal Enable 1 : Enable 0 : Disable
1	TDONE	R/W	0	Transfer Complete Signal Enable 1 : Enable 0 : Disable
0	CDONE	R/W	0	Command Complete Signal Enable 1 : Enable 0 : Disable

Auto CMD12 Error Status

0xB0020*n*[1]3C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PREN	AIEN	R						SCSEL	TUN	R		R	UHSSEL		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R								NOCMD			INDEX	ENDBIT	CRC	TIMEOUT	NORUN

Field	Name	RW	Reset	Description
29-24 15-8	R	R	0	Reserved
31	PREN	R/W	0	Preset Value Enable This bit enables the functions defined in the Preset Value registers. 1 : Automatic Selection by Preset Value are Enabled 0 : SDCLK and Driver Strength are controlled by Host Driver
30	AIEN	R/W	0	Asynchronous Interrupt Enable This bit can be set to 1 if a card support asynchronous interrupt and Asynchronous Interrupt Support is set to 1 in the Capabilities register. 1 : Enabled 0 : Disabled
23	SCSEL	R/W	0	Sampling Clock Select This bit is set by tuning procedure when Execute Tuning is cleared. Host Controller uses this bit to select sampling clock to receive CMD and DAT. 1 : Tuned clock is used to sample data 0 : Fixed clock is used to sample data
22	TUN	R/W	0	Execute Tuning This bit is set to 1 to start tuning procedure and automatically cleared when tuning procedure is completed. The result of tuning is indicated to Sampling Clock Select. 1 : Execute Tuning 0 : Not Tuned or Tuning Completed
21-20	R	R/W	0	Reserved
19	R	R/W	0	Reserved
18-16	UHSSEL	R/W	0	UHS Mode Select This field is used to select one of UHS-I modes and effective when 1.8V Signaling Enable is set to 1. 000b - SDR12 001b - SDR25 010b - SDR50 011b - SDR104 100b - DDR50 101b - 111 Reserved
7	NOCMD	R/W	0	Command Not Issued Setting this bit to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 error (D04 - D01)
4	INDEX	R/W	0	Auto CMD12 Index Error This bit is set if the Command Index error occurs in response to a command.
3	ENDBIT	R/W	0	Auto CMD12 End Bit Error This bit is set when detecting that the end bit of command response is 0.
2	CRC	R/W	0	Auto CMD12 CRC Error This bit is set when detecting a CRC error in the command response.
1	TIMEOUT	R/W	0	Auto CMD12 Timeout Error This bit is set if the no response is returned within 64 SDCLK cycles from the end bit of the command. If this bit is set to1, the other error status bits (D04-D02) are meaningless.
0	NORUN	R/W	0	Auto CMD12 Not Executed Setting this bit to 1 means the HC cannot issue Auto CMD12 to stop memory multiple block transfer due to some error. If this bit is set to 1, other error status bits (D04-D01) are meaningless.

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

Table 4.8 Relation between Auto CMD12 CRC Error and Auto CMD12 Timeout Error

Auto CMD12 CRC Error	Auto CMD12 Timeout Error	Kinds of Error
0	0	No Error
0	1	Response Timeout Error
1	0	Response CRC Error
1	1	CMD line conflict

Capabilities1 **0xB0020*n*[1]40**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLTP		AIS	BUS64	R	V18	V30	V33	RESUME	SDMA	HS	R	ADMA2	EXTBUS	MAXBLK	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BASECLK								TUNIT	R	TIMEOUTCLK					

Field	Name	RW	Reset	Description
27 20 6	R	R	0	Reserved
31-30	SLTP	R	0	Slot Type This field indicates usage of a slot by a specific Host System. (A host controller register set is defined per slot.) 00 : Removable Card Slot 01 : Embedded Slot for One Device 10 : Shared Bus Slot 11 : Reserved
29	AIS	R	0	Asynchronous Interrupt Support 1 : Asynchronous Interrupt Supported 0 : Asynchronous Interrupt Not Supported
28	BUS64	R	0	64bit System Bus Support Setting 1 to this bit indicates that the Host Controller supports 64-bit address descriptor mode and is connected to 64-bit address system bus.
26	V18	R	1	Voltage Support 1.8V 1 : Supported 0 : Not Supported
25	V30	R	1	Voltage Support 3.0V 1 : Supported 0 : Not Supported
24	V33	R	1	Voltage Support 3.3V 1 : Supported 0 : Not Supported
23	RESUME	R	1	Suspend / Resume Support 1 : Supported 0 : Not Supported
22	SDMA	R	1	SDMA Support 1 : Supported 0 : Not Supported
21	HS	R	1	High Speed Support 1 : Supported 0 : Not Supported
19	ADMA2	R	1	ADMA2 Support 1 : Supported 0 : Not Supported
18	EXTBUS	R	1	Extended Media Bus Support 1 : Supported 0 : Not Supported
17-16	MAXBLK	R	0x3	Max Block Length This value indicates the maximum block size that the Host Driver can read and write to the buffer in the Host Controller. The buffer shall transfer this block size without wait cycles. It is noted that transfer block length shall be always 512 bytes for SD Memory Cards regardless of this field.

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

				00 - 512 byte 01 - 1024 byte 10 - 2048 byte 11 - 4096 byte
15-8	BASECLK	R	0xd0	Base Clock Frequency For SD Clock (1) 6-bit Base Clock Frequency This mode is supported by the Host Controller Version 1.00 and 2.00. Upper 2-bit is not effective and always 0. Unit values are 1MHz. The supported clock range is 10MHz to 63MHz. 11xx xxxxb : Not supported 0011 1111b : 63MHz 0000 0010b : 2MHz 0000 0001b : 1MHz 0000 0000b : Get information via another method (2) 8-bit Base Clock Frequency This mode is supported by the Host Controller Version 3.00.Unit values are 1MHz. The supported clock range is 10MHz to 255MHz. FFh : 255MHz 02h : 2MHz 01h : 1MHz 00h Get information via another method
7	TUNIT	R	0	Timeout Clock Unit This bit shows the unit of base clock frequency used to detect Data Timeout Error. 1 : The unit is MHz. 0 : The unit is KHz.
5-0	TIMEOUTCLK	R	0	Timeout Clock Frequency This bit shows the base clock frequency used to detect Data Timeout Error. The Timeout Clock Unit defines the unit of this field's value. Timeout Clock Unit =0 [KHz] unit: 1KHz to 63KHz Timeout Clock Unit =1 [MHz] unit: 1MHz to 63MHz Not 0 - 1KHz to 63Khz or 1MHz to 63MHz 000000b - Get Information via another method

Capabilities2 **0xB0020*n*[1]44**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R						SPIBLK	SPI	CLKMULT							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTMODE		SDR50	R	RTTCNT				R	DRTD	DRTC	DRTA	R	DDR50	SDR104	SDR50

Field	Name	RW	Reset	Description
31-26 12 7 3	R	R	0	Reserved
25	SPIBLK	R	1	SPI Block Mode 0 - Not Supported 1 - Supported
24	SPI	R	1	SPI Mode 0 - Not Supported 1 - Supported
23-16	CLKMULT	R	0	Clock Multiplier This field indicates clock multiplier value of programmable clock generator. Refer to Clock Control register. Setting 00h means that Host Controller does not support programmable clock generator. FFh Clock Multiplier M = 256 .... 02h Clock Multiplier M = 3 01h Clock Multiplier M = 2 00h Clock Multiplier is Not Supported
15-147	RTMODE	R	0	Re-Tuning Modes This field defines the re-tuning capability of a Host Controller and how to manage the data transfer length and a Re-Tuning Timer by the Host Driver 00 - Mode1 01 - Mode2 10 - Mode3 11 - Reserved
13	SDR50	R	1	Use Tuning for SDR50 If this bit is set to 1, this Host Controller requires tuning to operate SDR50. (Tuning is always required to operate SDR104.) 1 : SDR50 requires tuning 0 : SDR50 does not require tuning
11-8	RTTCNT	R	0x1	Timer Count for Re-Tuning This field indicates an initial value of the Re-Tuning Timer for Re-Tuning Mode 1 to 3. 0h - Get information via other source 1h = 1 seconds 2h = 2 seconds 3h = 4 seconds 4h = 8 seconds -- n = $2^{(n-1)}$ seconds -- Bh = 1024 seconds Fh - Ch = Reserved
6	DRTD	R	1	Driver Type D Support This bit indicates support of Driver Type D for 1.8 Signaling. 1 : Driver Type D is Supported 0 : Driver Type D is Not Supported
5	DRTC	R	1	Driver Type C Support This bit indicates support of Driver Type C for 1.8 Signaling. 1 : Driver Type C is Supported 0 : Driver Type C is Not Supported

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

4	DRTA	R	1	Driver Type A Support This bit indicates support of Driver Type A for 1.8 Signaling. 1 : Driver Type A is Supported 0 : Driver Type A is Not Supported
2	DDR50	R	1	DDR50 Support 1 : DDR50 is Supported 0 : DDR50 is Not Supported
1	SDR104	R	1	SDR104 Support 1 : SDR104 is Supported 0 : SDR104 is Not Supported
0	SDR50	R	1	SDR50 Support 1 : SDR50 is Supported 0 : SDR50 is Not Supported

Maximum Current Capabilities

0xB0020*n*[1]48

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R								MAXCURV18							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MAXCURV30								MAXCURV33							

Field	Name	RW	Reset	Description
31-24	R	R	0	Reserved
23-16	MAXCURV18	R	0	Maximum Current for 1.8V *This will be supported in the future.*
15-8	MAXCURV30	R	0	Maximum Current for 3.0V *This will be supported in the future.*
7-0	MAXCURV33	R	0	Maximum Current for 3.3V *This will be supported in the future.*

Force Event for Error Interrupt / Auto CMD12 Error Status

0xB0020*n*[2]50

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VEND		CERR	TRERR	R		ADMA	ACMD12	R	DATEND	DATCRC	DATTIME	CINDEX	CMDEND	CMDCRC	CMDTIME
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R								NOCMD	R		INDEX	ENDBIT	CRC	TIMEOUT	NORUN

Field	Name	RW	Reset	Description
27-26 15-8 6-5	R	R	0	Reserved
31-30	VEND	R/W	0	Force Vendor Specific Error 1 : Enable 0 : Disable
29	CRR	R/W	0	Force CEATA Error 1 : Enable 0 : Disable
28	TRERR	R/W	0	Force Target Response Error 1 : Enable 0 : Disable
25	ADMA	R/W	0	Force ADMA Error 1 : Enable 0 : Disable
24	ACMD12	R/W	0	Force Auto CMD12 Error 1 : Enable 0 : Disable
23	R	R/W	0	Reserved
22	DATEND	R/W	0	Force Data End Bit Error

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.
[2] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

				1 : Enable 0 : Disable
21	DATCRC	R/W	0	Force Data CRC Error 1 : Enable 0 : Disable
20	DATTIME	R/W	0	Force Data Timeout Error 1 : Enable 0 : Disable
19	CINDEX	R/W	0	Force Command Index Error 1 : Enable 0 : Disable
18	CMDEND	R/W	0	Force Command End Bit Error 1 : Enable 0 : Disable
17	CMDCRC	R/W	0	Force Command CRC Error 1 : Enable 0 : Disable
16	CMDTIME	R/W	0	Force Command Timeout Error 1 : Enable 0 : Disable
7	NOCMD	R/W	0	Force Event For Command Not Issued 1 : Enable 0 : Disable
4	INDEX	R/W	0	Force Event For Auto CMD12 Index Error 1 : Enable 0 : Disable
3	ENDBIT	R/W	0	Force Event For Auto CMD12 End Bit Error 1 : Enable 0 : Disable
2	CRC	R/W	0	Force Event For Auto CMD12 CRC Error 1 : Enable 0 : Disable
1	TIMEOUT	R/W	0	Force Event For Auto CMD12 Timeout Error 1 : Enable 0 : Disable
0	NORUN	R/W	0	Force Event For Auto CMD12 Not Executed 1 : Enable 0 : Disable

ADMA Error Status

0xB0020*n*[1]54

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R													LEN	ERRSTATE	

Field	Name	RW	Reset	Description
31-3	R	R	0	Reserved
2	LEN	R/W	0	ADMA Length Mismatch Error This error occurs in the following 2 cases. (1) While Block Count Enable being set, the total data length specified by the Descriptor table is different from that specified by the Block Count and Block Length. (2) Total data length can not be divided by the block length.
1:0	ERRSTATE	R/W	0	ADMA Error State This field indicates the state of ADMA when error is occurred during ADMA data transfer. This field never indicates "10" because ADMA never stops in this state. 00 - ST_STOP (Stop DMA) Points next of the error descriptor 01 - ST_FDS (Fetch Descriptor) Points the error descriptor 10 - Never set this state (Not used) 11 - ST_TFR (Transfer Data) Points the next of the error descriptor

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

ADMA System Address0 **0xB0020*n*[1]58**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR[15:0]															

ADMA System Address1 **0xB0020*n*[2]5C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADDR[63:48]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR[47:32]															

Field	Name	RW	Reset	Description
63-0	ADDR	R/W	0	ADMA System Address This register holds byte address of executing command of the Descriptor table. 32-bit Address Descriptor uses lower 32-bit of this register. At the start of ADMA, the Host Driver shall set start address of the Descriptor table. The ADMA increments this register address, which points to next line, when every fetching a Descriptor line. When the ADMA Error Interrupt is generated, this register shall hold valid Descriptor address depending on the ADMA state. The Host Driver shall program Descriptor Table on 32-bit boundary and set 32-bit boundary address to this register. ADMA2 ignores lower 2-bit of this register and assumes it to be 00b.

Refer to the SD Host Controller Standard Specification V3.00 of SD Association to know how to make the descriptor table.

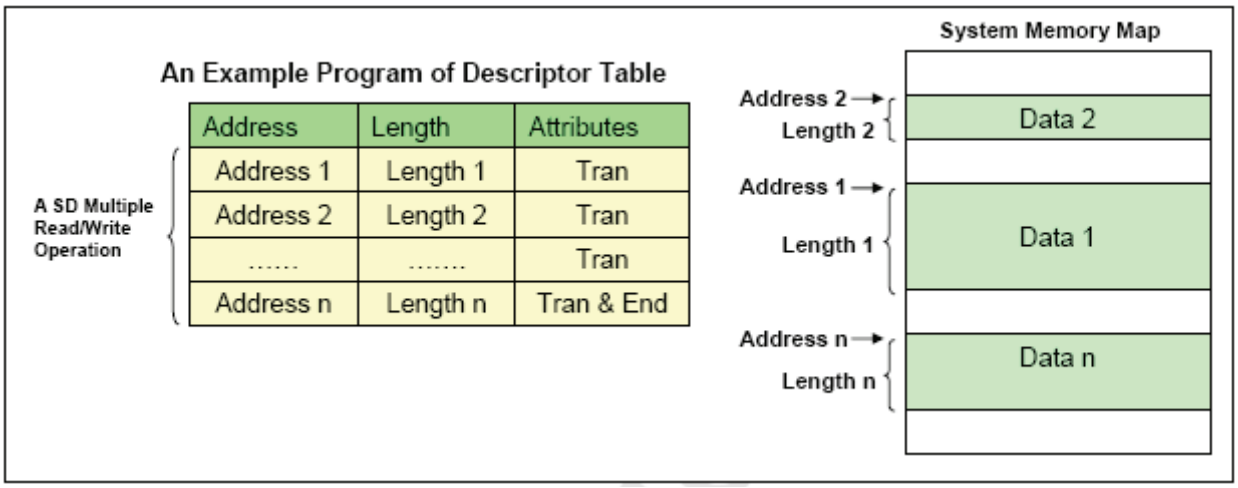
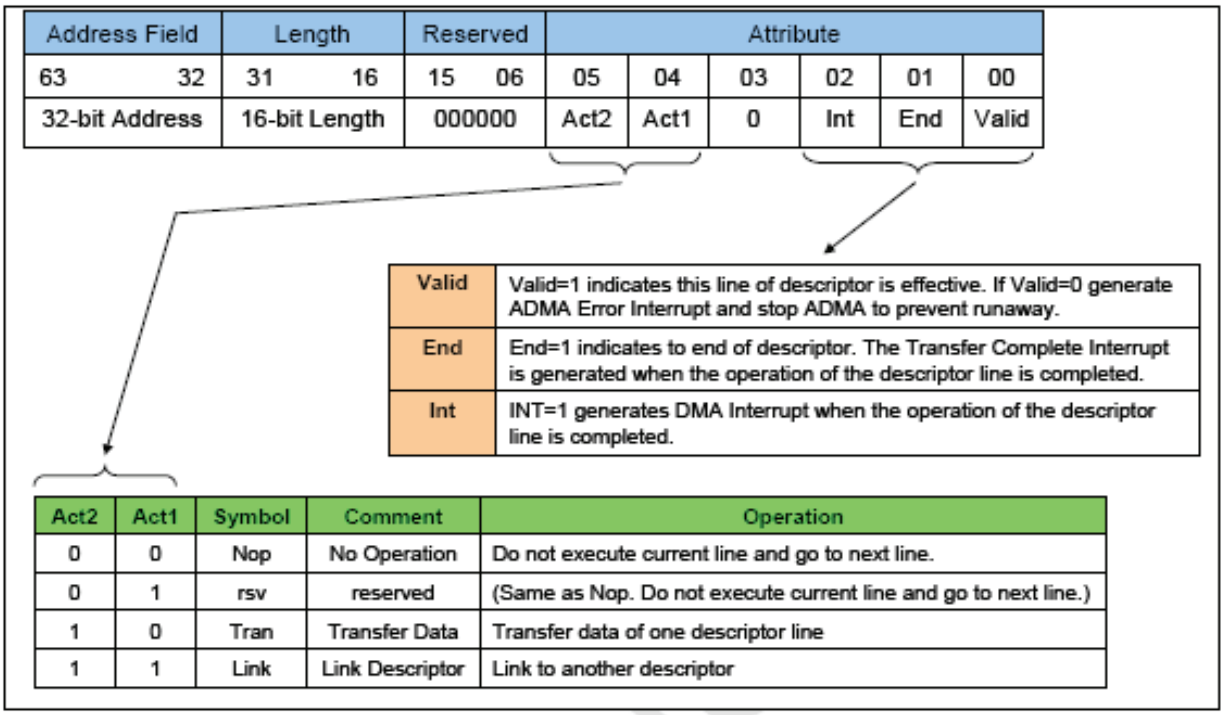


Figure 4.2 An Example of ADMA2 Data Transfer

Figure 4.2 shows a typical ADMA2 descriptor program. The data area is sliced in various lengths and each slice is placed somewhere in system memory. The Host Driver describes the Descriptor Table with set of address, length and attributes. Each sliced data is transferred in turns as programmed in descriptor.

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.
[2] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.



Address Field	Length	Reserved	Attribute					
63 32	31 16	15 06	05	04	03	02	01	00
32-bit Address	16-bit Length	000000	Act2	Act1	0	Int	End	Valid

Valid	Valid=1 indicates this line of descriptor is effective. If Valid=0 generate ADMA Error Interrupt and stop ADMA to prevent runaway.
End	End=1 indicates to end of descriptor. The Transfer Complete Interrupt is generated when the operation of the descriptor line is completed.
Int	INT=1 generates DMA Interrupt when the operation of the descriptor line is completed.

Act2	Act1	Symbol	Comment	Operation
0	0	Nop	No Operation	Do not execute current line and go to next line.
0	1	rsv	reserved	(Same as Nop. Do not execute current line and go to next line.)
1	0	Tran	Transfer Data	Transfer data of one descriptor line
1	1	Link	Link Descriptor	Link to another descriptor

Figure 4.3 32-bit Address Descriptor Table

Preset Values **0xB0020*n*[1]60**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRSTR		R			CG	SDCLKFRQ									
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DRSTR		R			CG	SDCLKFRQ									

Offset	Preset Value Registers	Signal Value
0x60	Preset Value for Initialization	3.3V or 1.8V
0x62	Preset Value for Default Speed	3.3V
0x64	Preset Value for High Speed	3.3V
0x66	Preset Value for SDR12	1.8V
0x68	Preset Value for SDR25	1.8V
0x6a	Preset Value for SDR50	1.8V
0x6c	Preset Value for SDR104	1.8V
0x6e	Preset Value for DDR50	1.8V

Field	Name	RW	Reset	Description
29-27 13-11	R	R	0	Reserved
15-14	DRSTR	R	0	Driver Strength Select Value Driver Strength is supported by 1.8V signaling bus speed modes. This field is meaningless for 3.3V signaling. 11b Driver Type D is Selected 10b Driver Type C is Selected 01b Driver Type A is Selected 00b Driver Type B is Selected
10	CG	R	0	Clock Generator Select Value This bit is effective when Host Controller supports programmable clock generator. 1 : Programmable Clock Generator 0 : Host Controller Ver2.00 Compatible Clock Generator
9-0	SDCLKFRQ	R	0	SDCLK Frequency Select Value 10-bit preset value to set SDCLK Frequency Select in the Clock Control Register is described by a host system.

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

Boot Timeout Control 0xB0020*n*[1]70

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BTOUTCNT [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BTOUTCNT[15:0]															

Field	Name	RW	Reset	Description
31-0	BTOUTCNT	R/W	0	Boot Data Time-out Counter Value This value determines the interval by which DAT line time-outs are detected during boot operation for eMMC4.4 card. The value is in number of sd clock

SPI Interrupt Support 0xB0020*n*[2]F0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								SPIINT[7:0]							

Field	Name	RW	Reset	Description
31-8	R	R	0	Reserved
7-0	SPIINT	R/W	0	SPI Interrupt Support This bit is set to indicate the assertion of interrupts in the SPI mode at any time, irrespective of the status of the card select (CS) line. If this bit is zero, then SDIO card can only assert the interrupt line in the SPI mode when the CS line is asserted.

Host Controller Version / Slot Interrupt Status 0xB0020*n*[3]FC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								SLOTINT[7:0]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VENDOR[7:0]								SPEC[7:0]							

Field	Name	RW	Reset	Description
31-24	R	R	0	Reserved
23-16	SLOTINT	R/W	0	Interrupt Signal For Each Slot These status bit indicate the logical OR of Interrupt signal and Wakeup signal for each slot. Bit 00 - Slot 1 Bit 01 - Slot 2 Bit 02 - Slot 3 ----- ----- Bit 07 - Slot 8
15-8	VENDOR	R/W	0	Vendor Version Number This status is reserved for the vendor version number. The HD should not use this status.
7-0	SPEC	R/W	0	Specification Version Number This Status indicates the Host Controller Spec Version. 00 - SD Host Specification version 1.0 01 - SD Host Specification version 2.00 including only the feature of theTest Register 02 - SD Host Specification version 2.00 including the feature of the Test Register and ADMA others – Reserved

[1] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.
[2] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.
[3] n = 0 , 2, 4, 6 for Host Controller0, 1, 2, 3 respectively.

4.4 Command Format

All commands have a fixed code length of 48 bits, needing a transmission time of 1.92 μs @ 25 MHz and 0.96 μs @ 50 MHz.

Table 4.9 Command Format

Bit Position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'1'	x	x	x	'1'
Description	Start bit	Transmission bit	Command index	Argument	CRC7	End bit

A command always starts with a start bit (always 0), followed by the bit indicating the direction of transmission (host = 1). The next 6 bits indicate the index of the command, this value being interpreted as a binary coded number (between 0 and 63). Some commands need an argument (e.g. an address), which is coded by 32 bits. A value denoted by 'x' in the table above indicates this variable is dependent on the command. All commands are protected by a CRC. Every command codeword is terminated by the end bit (always 1).

4.5 Timing Diagram

4.5.1 Timing of the Transaction in SD Mode

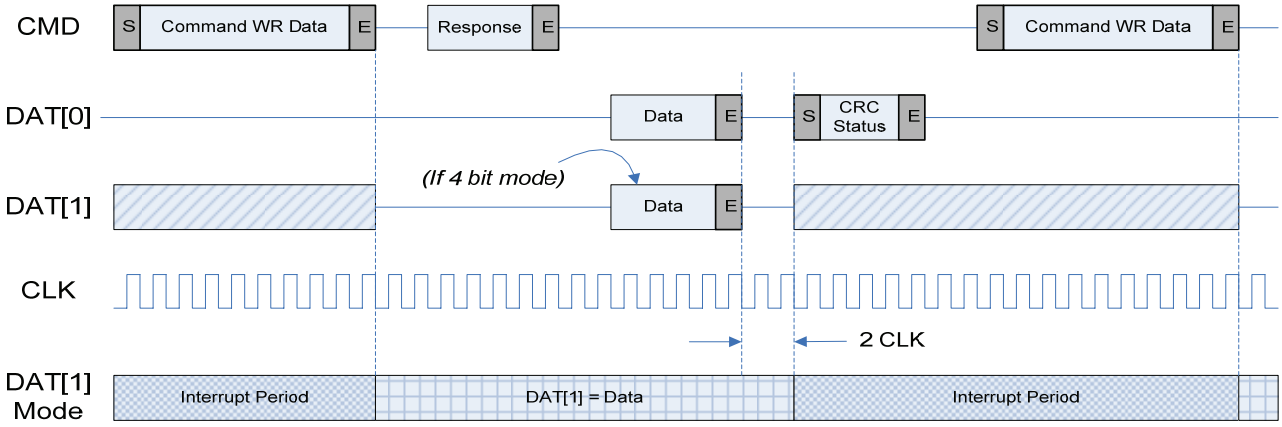


Figure 4.4 SDIO/SD – Write Interrupt Cycle Timing

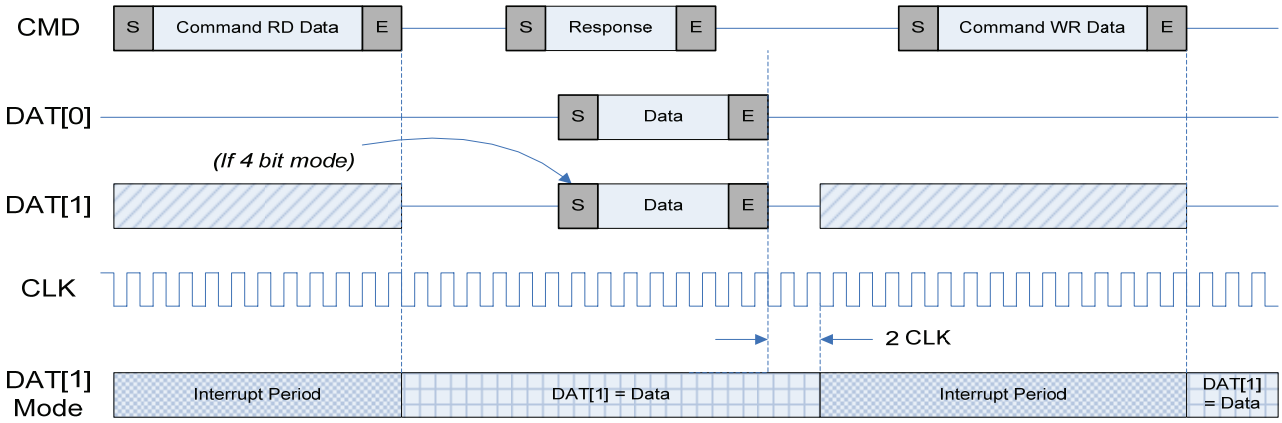


Figure 4.5 SDIO/SD – Read Interrupt Cycle Timing

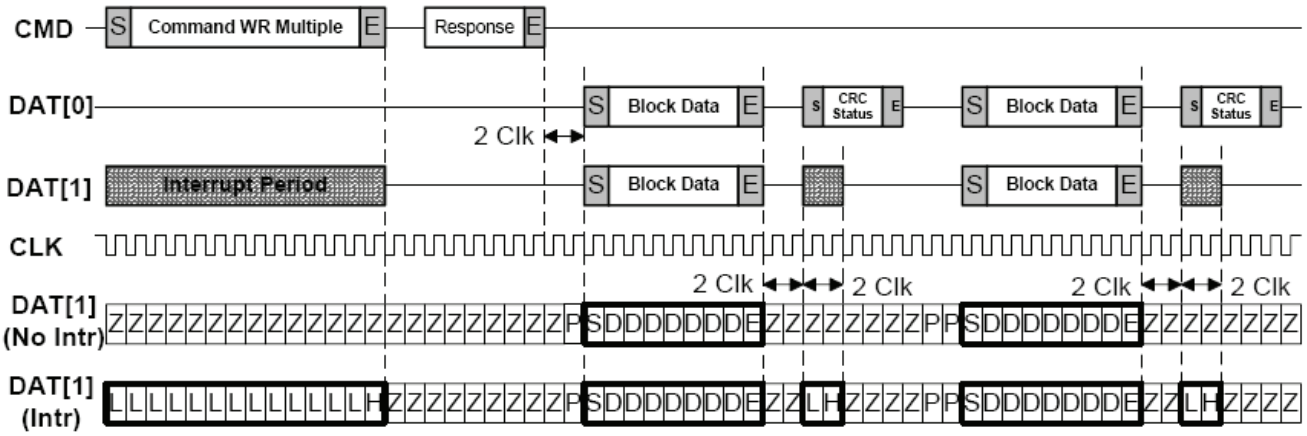


Figure 4.6 SDIO/SD – Multiple Block 4-Bit Write Interrupt Cycle Timing

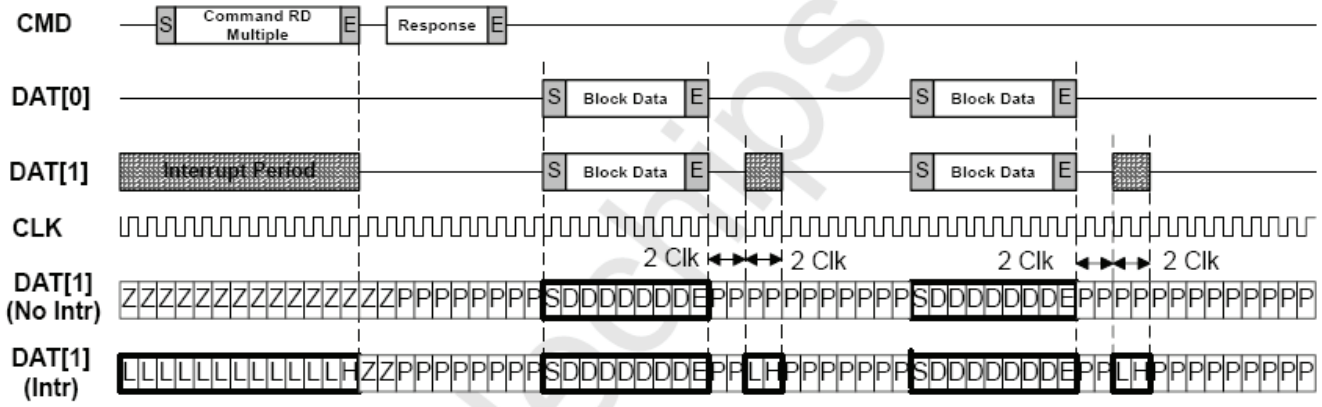


Figure 4.7 SDIO/SD – Multiple Block 4-Bit Read Interrupt Cycle Timing

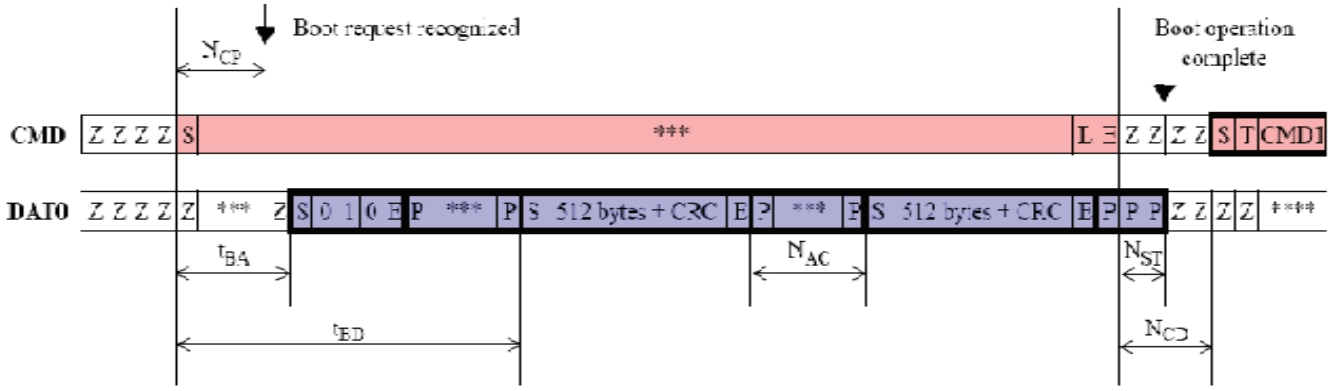


Figure 4.8. Boot Operation Timing with Termination Between Consecutive Data Blocks

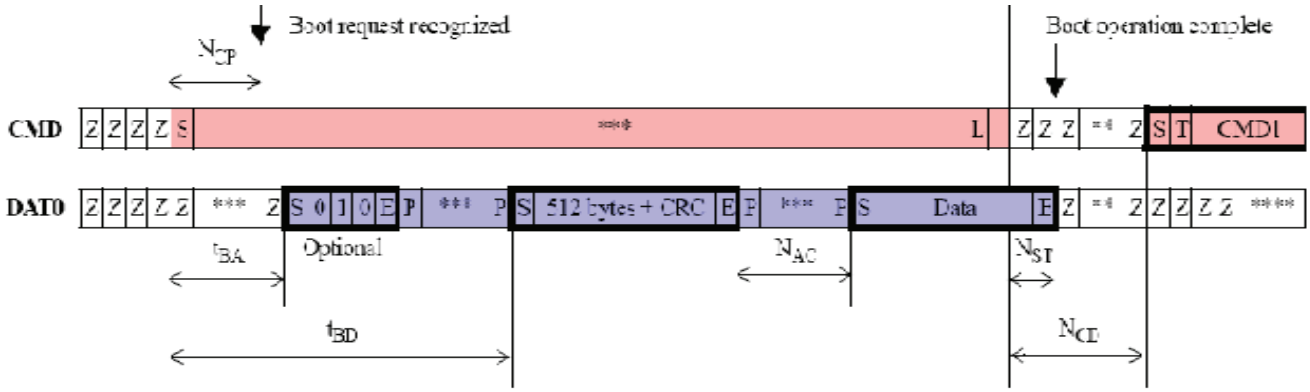
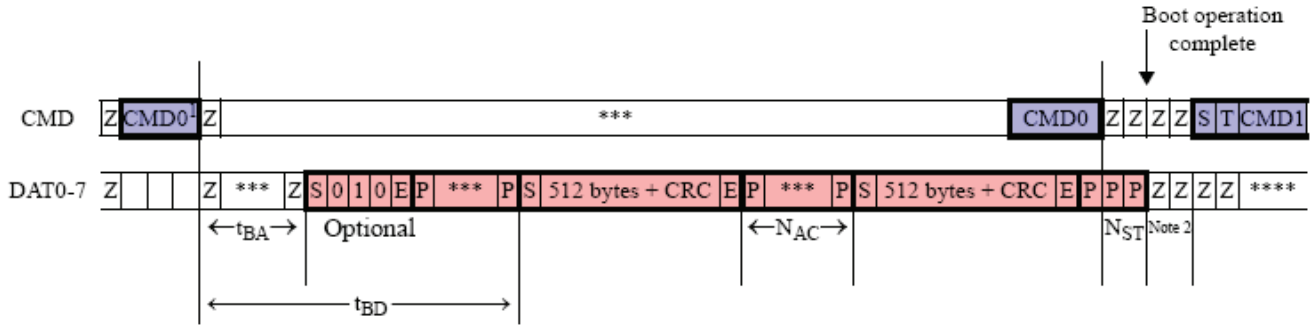
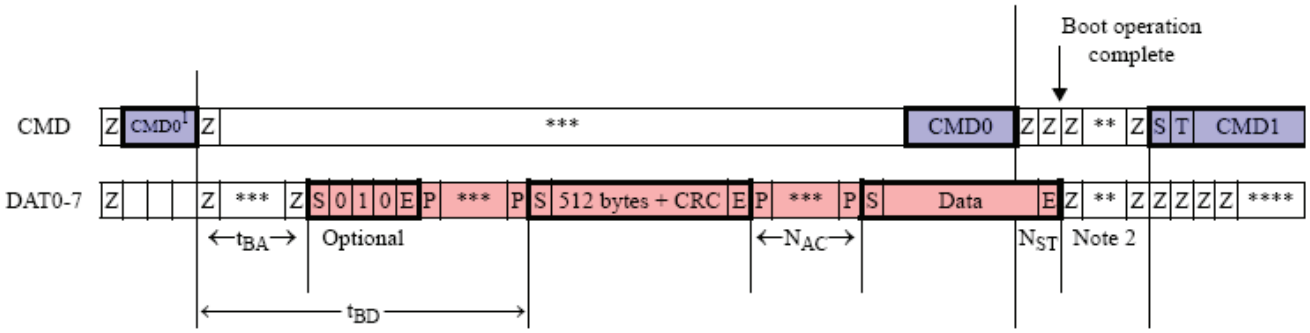


Figure 4.9. Boot Operation Timing with Termination During Transfer



NOTE 1. CMD0 with argument 0xFFFFFFFA.

Figure 4.10. Alternative boot operation, termination between consecutive data blocks



NOTE 1. CMD0 with argument 0xFFFFFFFA.

Figure 4.11. Alternative boot operation, termination during transfer

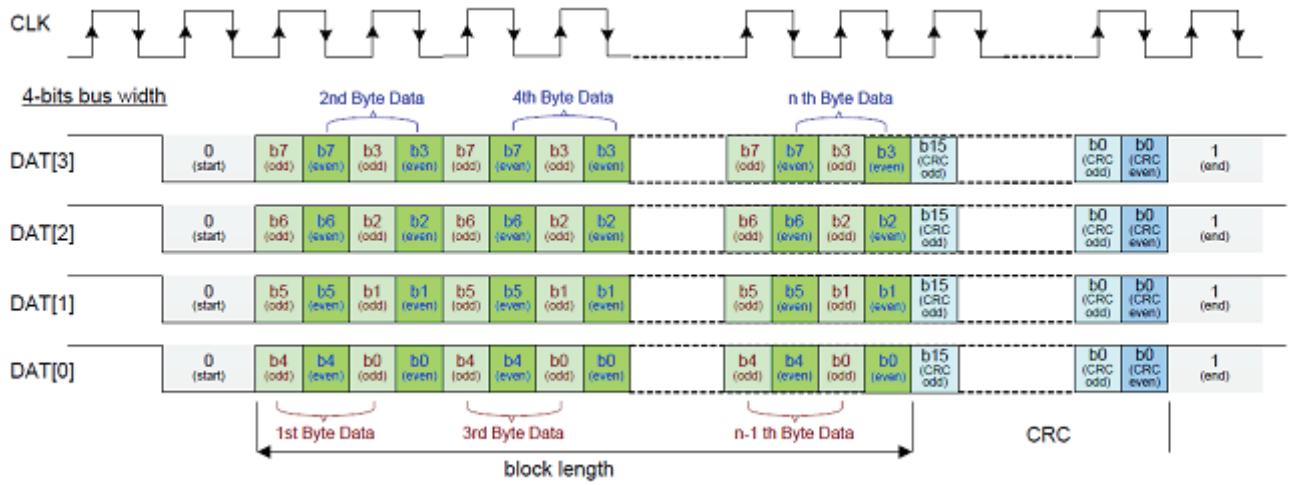


Figure 4.12. Data Packet Format in DDR50 mode – Usual Data

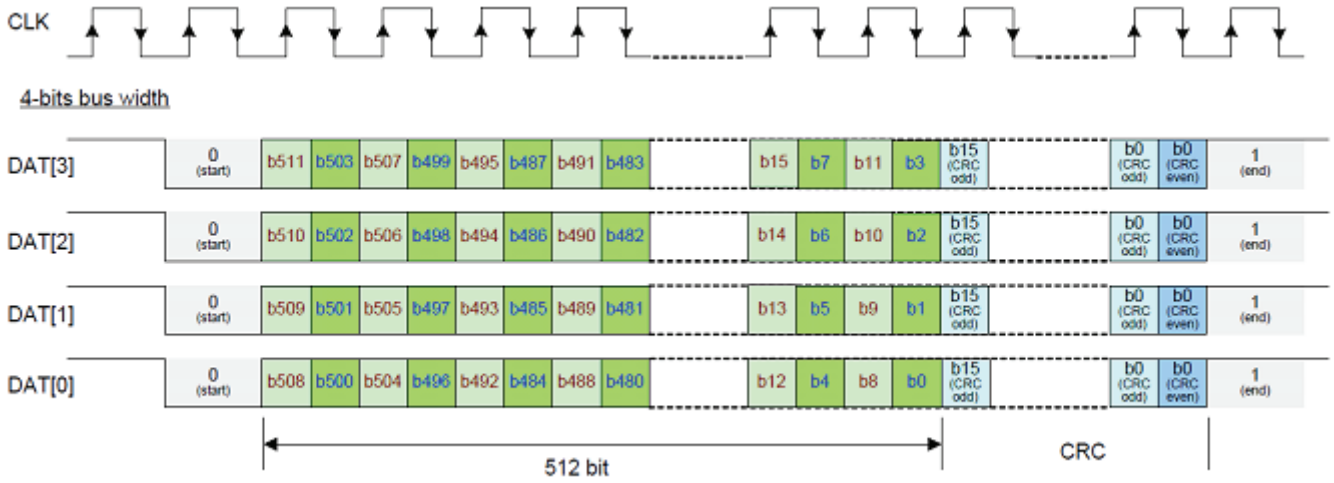


Figure 4.13. Data Packet Format in DDR50 mode – Wide Width Data

4.5.2 Timing to Switch Signal Voltage

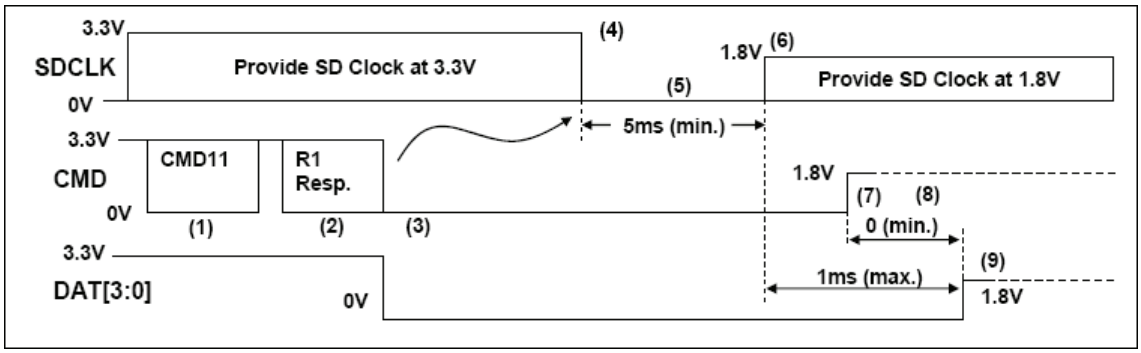


Figure 4.14 Signal Voltage Switching Sequence

(1) Host issues CMD11 to start voltage switch sequence.
(2) The card returns R1 response.
(3) The card drives CMD and DAT[3:0] to low immediately after the response.
(4) The host stops supplying SDCLK. The card shall start switching voltage after host stops SDCLK. The time to stop SDCLK is not specified. The host can detect whether the sequence starts by checking signal level of either one of CMD, DAT[3:0]. Which signal should be checked depends on ability of the host. If low level is not detected, the host should abort the sequence and execute power cycle.
(5) 1.8V output of voltage regulator in card shall be stable within 5ms. Host keeps SDCLK low at least **5ms**. This means that 5ms is the maximum for the card and the minimum for the host.
(6) After 5ms from (4) and host voltage regulator is stable, the host starts providing SDCLK at 1.8V. The card can check whether SDCLK voltage is 1.8V.
(7) By detecting SDCLK, the card drives CMD to high at 1.8V at least one clock and then stop driving (tri-state). CMD is triggered by rising edge of SDCLK (SDR timing).
(8) The card can check whether host drives CMD to 1.8V through the host pull-up resister.
(9) If switching to 1.8V signaling is completed successfully, the card drives DAT[3:0] to high at 1.8V at least one clock and then stop driving (tri-state). DAT[3:0] is triggered by rising edge of SDCLK (SDR timing). DAT[3:0] shall be high within **1ms** from start of providing SDCLK. Host check whether DAT[3:0] is high after 1ms from supplying SDCLK. This means that 1ms is the maximum for the card and the minimum for the host.

● Power Down and Power Cycle

- When the host shuts down the power, the card V_{DD} shall be lowered to less than 0.5Volt for a minimum period of 1ms. During power down, DAT, CMD, and CLK should be disconnected or driven to logical 0 by the host to avoid a situation that the operating current is drawn through the signal lines.
- If the host needs to change the operating voltage, a power cycle is required. Power cycle means the power is turned off and supplied again. Power cycle is also needed for accessing cards that are already in *Inactive State*. To create a power cycle the host shall follow the power down description before power up the card (i.e. the card V_{DD} shall be once lowered to less than 0.5Volt for a minimum period of 1ms).

5 EHI

5.1 Overview

This LSI has the external host interface (EHI) that allows the external host device to be connected to the on-chip system bus. The external host device can be directly connected to 68/80-series interfaces and access the memory area of this LSI. For software based data transfer, EHI can generate the internal interrupt of this LSI, and this LSI can also send interrupt request to the external host controller.

The features of EHI are as follows.

- 68/80 series interface with 8/16bits data can be supported.
- Burst transfer is supported and address can be incremented automatically.
- External host device can generate an internal interrupt of this LSI.
- Interrupt request can be sent to the external host by programming specific bits in EHI control register.
- Semaphore is supported for improving data transfer efficiency.

The interface signals are shown in Table 5.1. HPCSN_L and HPCSN should not be asserted at the same time.

Table 5.1 EHI External Interface Pin

PIN Name	I/O	Function
HPCSN_L	I	Chip select signal 1
HPCSN	I	Chip select signal 0
HPXA[0]	I	Address signal. It is used for switching between normal access and EHIND/EHST register access. Normal access (HPXA [0]= 0): EHI register indicated by EHIND register is accessed. EHIND/EHST access (HPXA[0] = 1) : Writing to EHIND register and reading from EHST register.
HPWRN	I	68-interface: Enable signal 80-interface: Write strobe signal
HPRDN	I	68-interface: Data reading or writing select signal 80-interface: Read strobe signal
HPXD[17:0]	B	Data bus
HPINTO	O	External host interrupt request signal / Ready signal for HPCSN
HPINTO1	O	External host interrupt request signal / Ready signal for HPCSN_L

5.2 Register Description

The EHI registers are shown in Table 5.2. Chip Select 0 (HPCSN) base address is 0xB0000000 and Chip Select 1 (HPCSN_L) base address is 0xB0010000.

Table 5.2 EHI register map(EHI_Base = 0xB0000000, 0xB0010000)

Name	Offset	Int.	Ext.	Initial	Description
EHST	0x00	R/W	R/W	0x00000080	Status register
EHIINT *	0x04	R/W	R/W	0x00000000	Internal interrupt control register
EHEINT*	0x08	R/W	R/W	0x00000000	External interrupt control register
EHA	0x0C	R	R/W	0x00000000	Address register
EHAM	0x10	R/W	R	0x00000000	Address masking register
EHD	0x14	R/W	R/W	0x00000000	Data register
EHSEM*	0x18	R/W	R/W	0x00000000	Semaphore register
EHCFG*	0x1C	R/W	R/W	0x00000000	Configuration registers
EHIND	0x20	R	W	0x00000000	Index register
EHRWCS*	0x24	R	R/W	0x00000000	Read/Write Control/Status register

Note : *) When the external host device writes to these registers, EHIND [1:0] bits are ignored.

EHI has two clock inputs; one is HCLK, which is for the on-chip system bus interface and the other is ECLK (EHI CLK) which is for interface with the external host.

Before the on-chip CPU accesses these registers, HCLK and ECLK should be enabled.

ECLK should be enabled for the external host to access these registers except for EHST register. Refer to 5.3.1 Access to EHI registers for more information.

EHST (Status Register) EHI_BASE+0x00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			RFNE	WFNE	WFF	FUR	FOR	RDY	ST[6:0]						

Field	Name	RW(Int.)	RW(Ext.)	Reset	Description
31-13	-	R	R	0	These bits are read as 0.
12	RFNE	R	R	0	It represents that Read-FIFO is not empty. When it is equal to 0, Read-FIFO is empty.
11	WFNE	R	R	0	It represents that the Write-FIFO is not empty. When it is equal to 0, Write-FIFO is empty.
10	WFF	R	R	0	It represents that the Write-FIFO is full.
9	FUR	R/C	R	0	It represents that Read-FIFO underrun is occurred. It is cleared when the on-chip CPU writes 1 to this bit.
8	FOR	R/C	R	0	It represents that Write-FIFO overrun is occurred. It is cleared when the on-chip CPU writes 1 to this bit.
7	RDY	R	R	1	It represents that an external host device can access the on-chip system bus.
6-0	ST	R/W	R/W	0	It is read and written by an external host connected to EHI and by the on-chip CPU.

The external host device has two methods for reading this register. One is that the external host device issues the read operation while HPXA is high. The other is that the external host sets EHIND to 0x00 or 0x01 (EHST offset) and issues the read operation while HPXA is low.

When using the 8-bit interface, EHST[7:0] can be read by the external host device while HPXA is high regardless of ECLK. EHST[15:8] is read by the external host device while HPXA is low and EHIND is set to 0x01. Therefore, when ECLK is disabled, the external host device can not read EHST[15:8].

When using the 16-bit interface, EHST[15:0] can be read by the external host device while HPXA is high. Therefore the external host device can read EHST[15:0] regardless of ECLK.

EHIINT (Internal Interrupt Control Register) EHI_BASE+0x04

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								IIRQ_ST							IIRQ

Field	Name	RW(Int.)	RW(Ext.)	Reset	Description
31-8	-	R	R	0	These bits are read as 0.
7-1	IIRQ_ST	R/W	R/W	0	It is only updated by software. It can specify the interrupt request number.
0	IIRQ	R/W	R/W	0	When it is equal to 1, EHI generates the interrupt request and it is sent to the on-chip interrupt controller. Note that it should be cleared manually. The external host device may need to check that the previous issued interrupt is processed by the on-chip CPU before issuing the new interrupt request. It is only updated by software.

EHIINT register is used to generate interrupts from the external host device. When the interrupt is generated, the on-chip CPU processes the interrupt service routine and should be clear EHIINT.IIRQ bit. Therefore, the external host can detect whether the requested interrupt is processed as it reads EHIINT register.

EHEINT (External Interrupt Control Register) EHI_BASE+0x08

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								EIRQ_ST							EIRQ

Field	Name	RW(Int.)	RW(Ext.)	Reset	Descriptions
31-8	-	R	R	0	These bits are read as 0
7-1	EIRQ_ST	R/W	R/W	0	This specifies EIRQ interrupt source. The external host device can detect interrupt source via EIRQ_ST.
0	EIRQ	R/W	R/W	0	If HPINT is connected to the external interrupt input of the external host, this LSI can send interrupt request to the external host. When EHCFG.RDYE is equal to 0, EHEINT.EIRQ value is output through HPINT pin. Otherwise, this bit is not applicable.

EHEINT is used to issue interrupts to the external host via HPINT pin by the on-chip CPU. HPINT pin is used for interrupt request pin when EHCFG.RDYE is set to 0.

EHA (Address Register) EHI_BASE+0x0C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHA[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EHA[15:2]														0	

Field	Name	RW(Int.)	RW(Ext.)	Reset	Description
31-2	-	R	R/W	0	The address for the on-chip system bus access of this LSI.
1-0	-	R	R	0	These bits are always 0.

EHA maps to the memory space of this LSI when the external host accesses the system bus of this LSI.

When the external host issues the incremental writing operation (EHRWCS.AI = 1 and EHRWCS.RW = 1), EHA increments automatically during data transfer and when it is completed (EHRWCS.RW = 0), EHA has the incremented address as the number of written data.

But, when the incremental reading operation is completed, EHA do not has the incremented address as the number of read data, but the prefetched address. Therefore, when the new incremental reading operation is issued, EHA should be updated.

EHA can be masked by EHAM register. Refer to the next page, EHAM register description, for more information.

EHAM (Address Masking Register) EHI_BASE+0x10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHAM[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EHAM[15:2]														0	

Field	Name	RW(Int.)	RW(Ext.)	Reset	Description
31-2	-	R/W	R	0	It masks EHA register. Therefore, the effective address on the on-chip system bus is EHA[31:2] & ~EHAM[31:2].
1-0	-	R	R	0	These bits are read as 0.

EHA can be masked by EHAM register and then the effective address is EHA[31:2] & ~EHAM[31:2]. When EHRWCS.AI is set to 1, the generated address of every transfers is (EHA[31:2] + 1) & ~EHAM[31:2].

For example, when EHA = 0x1000000, EHRWCS.AI=1, and EHAM = 0x0F00, EHA has the address between 0x10000000 and 0x100000FC.

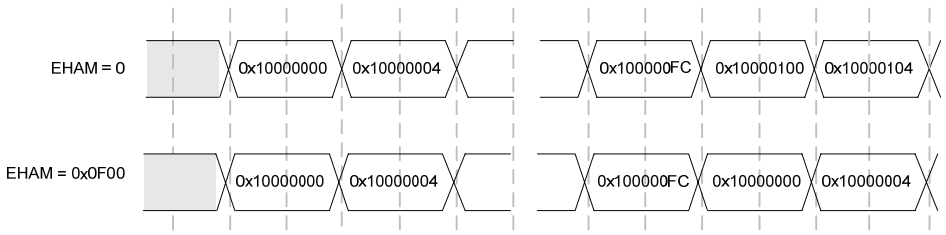


Figure 5.1 Example of EHAM usage

EHD (Data Register) EHI_BASE+0x14

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHD[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EHD[15:0]															

Field	Name	RW(Int.)	RW(Ext.)	Reset	Description
31-0	-	R/W	R/W	0	While EHRWCS.RW = 1 or 2, the external host device accesses the memory area of this LSI using EHA register. In this case, it is used for writing or reading data.

EHD is used to transfer data for system bus access mode.

EHSEM (Semaphore Register) **EHI_BASE+0x18**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ST						FLG	

Field	Name	RW(Int.)	RW(Ext.)	Reset	Description
7-2	ST	-	-	0	FLG = 00b Reading only: On-chip CPU, External host device FLG = 01b Reading/Writing: External host device Reading only: On-chip CPU FLG = 10b Reading/Writing: On-chip CPU Reading only: External host device
1-0	FLG	-	-	0	If EHSEM is read and then FLG is 00b, this means EHSEM is not occupied by any master. If the external host device reads EHSEM which is not occupied by this LSI, then EHSEM.FLG becomes 01b. If the on-chip CPU reads EHSEM which is not occupied by the external host device, then EHSEM.FLG becomes 10b. If the on-chip CPU and the external host read EHSEM simultaneously, return value for external host is 00b and it for the on-chip CPU is 01b. 00b (NOT OCCUPIED) Reading only: On-chip CPU, External host device 01b (EXTERNAL HOST) Reading/Writing: External host device Reading only: On-chip CPU 10b (ON-CHIP CPU) Reading/Writing: On-chip CPU Reading only: External host device 11b N/A

EHCFG (Configuration Register) **EHI_BASE+0x1C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								WIRQ	CSIRQ		RDYP	RDYE	BW		MD

Field	Name	RW(Int.)	RW(Ext.)	Reset	Description
31-8	-	R	R	0	These bits are read as 0.
7	WIRQ	R/W	R/W	0	Writing operation becomes the internal interrupt source. When HPCSN and HPWRN are asserted in 80-mode or HPCSN, HPWRN(E), and HPRDN(R/W) in 68-mode, the internal interrupt request is generated. For preventing additional interrupts by write operation, this bit should be cleared manually.
6	CSIRQ	R/W	R/W	0	HPCSN becomes the internal interrupt source. When HPCSN is asserted regardless of HPWRN and HPRDN, the internal interrupt request is generated. For preventing additional interrupts by HPCSN, this bit should be cleared manually.
5	-	R	R	0	This bit reads as 0.
4	RDYP	R/W	R/W	0	When RDYE is equal to 1, EHST.RDY ⊕ EHCFG.RDYP is outputted through HPINT. Therefore, it is only valid when RDYE is equal to 1. Refer to RDYE description.
3	RDYE	R/W	R/W	0	0: HPINT is used for the external interrupt. EHEINT.EIRQ is outputted through HPINT. 1: HPINT is used for READY signal. In this case, RDYP is valid.
2-1	BW	R/W	R/W	0	00b: 8-bit interface mode 01b: Reserved 10b: 16-bit interface mode 11b: 18-bit interface mode 18-bit data is written to and is read from lower 18bits of 32-bit word memory. Therefore, upper 14bits in 32-bit word memory cannot be used.
0	MD	R/W	R/W	0	0 : 80 interface mode 1 : 68 interface mode

EHCFG configures the overall EHI operation. The on-chip CPU should select the interface mode, 80-interface mode or 68-interface mode.

When the external device can not write to the EHI registers, because of clock configuration or interface timing problem, it can issues interrupts for the on-chip CPU using EHCFG.WIRQ or EHCFG.CSIRQ (Figure 5.2).

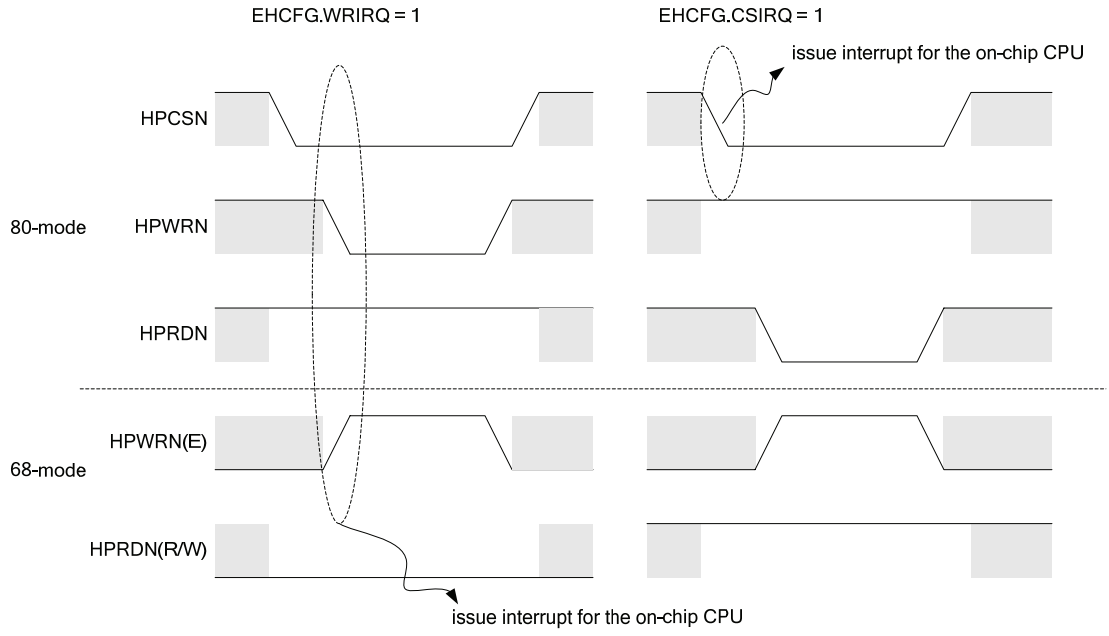


Figure 5.2 Interrupt request using EHCFG.WRIRQ and CSIRQ

EHIND (Index Register) EHI_BASE+0x20

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								EHIND[7:0]							

Field	Name	RW(Int.)	RW(Ext.)	Reset	Description
31-8	-	R	R	0	These bits read as 0.
7-0	EHIND	R	W	0	It selects the internal register for the external host. It can be only written by the external host device while HPXA is high. Refer to Table 5.2 for value corresponding to each register. 0x00: EHST 0x04: EHIINT 0x08: EHEINT 0x0C: EHA 0x10: EHAM 0x14: EHD 0x18: EHSEM 0x1C: EHCFG 0x20: EHIND 0x24: EHRWCS When writing to EHIINT , EHEINT, EHSEM, EHCFG, and EHRWCS, EHIND[1:0] is ignored. For example, when writing to EHIINT[7:0] register, EHIND can be 4, 5, 6, or 7

EHIND is used to index other EHI registers and only written by the external host while HPXA is high.

EHRWCS (Read/Write Control/Status Register) EHI_BASE+0x24

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								AI	LK	RW		BSIZE			

Field	Name	RW(Int.)	RW(Ext.)	Reset	Description
31-8	-	R	R	0	These bits are read as 0.
7	AI	R	R/W	0	EHA is auto-incremented while EHRWCS.RW is 1 or 2.
6	LK	R	R/W	0	Once the external host device is a master of the system bus of this LSI, bus-handover cannot be occurred until burst transfer as EHRWCS.BSIZE is completed.
5-4	RW	R	R/W	0	0 : Access to EHI registers 1 : Writing to the system bus of this LSI. When BSIZE = 0(single transfer), it is cleared automatically. 2 : Reading from the system bus of this LSI. When BSIZE = 0, it is cleared automatically. 3 : Undefined
3-0	BSIZE	R	R/W	0	It specifies how many words(1word = 32bits) will be transferred and EHRWCS.BSIZE + 1 words will be transferred. It should be programmed before EHRWCS.RW is set to 2 (Reading operation) or 1(Writing operation). For continuous burst transfer, EHST.RDY may need to be checked every burst size transfer.

EHRWCS register is used to control the on-chip system bus access.

Before EHRWCS.RW is issued for reading or writing operation, other fields of this register and EHA should be configured. Refer to 5.3.2 Access to the On-chip System Bus about reading and writing operation for the on-chip system bus.

5.3 Operation

5.3.1 Access to EHI registers

Table 5.3 Access to the EHI

HCLK	ECLK	Reading From EHI register		Writing to EHI register		access to the on-chip system bus
		on-chip	Ext.Host	on-chip	Ext.Host	Ext.Host
X	X	X	△	X	X	X
X	O	X	O	X	O	X
O	X	X	△	X	X	X
O	O	O	O	O	O	O

△ = When 8bits interface is used, EHST[7:0] can be only read. When 16bits interface is used, EHST[15:0] can be only read.

ECLK and HCLK should be enabled for the on-chip CPU to access the EHI registers. And the external host can only access the EHI registers when ECLK is enabled except for EHST register. When ECLK is disabled, the external host can only read EHST[15:0] register in the 16-bit interface mode and EHST[7:0] register in the 8-bit interface mode. The relationship between clocks and accessibility is shown in Table 5.3.

Most of EHI registers are accessed by indirect addressing using EHIND register. To write to or read from an EHI register except for EHIND and EHST registers, EHIND register should be set to its offset value in advance and then HPXA should be low during reading or writing operation. If the external host intends to write data to EHIND , it should drive HPXA pin to high. When the external host device issues the reading operation while HPXA is high, it reads EHST register. Note that EHIND register is write-only. When the external host device intents to write to EHST register, it should also use EHIND register. The following shows how to write data (=0x12345678) to EHA register.

(1) Write EHA[15:0] offset value(=0x0C) while HPXA = 1. EHIND register indicates EHA[15:0] register.
(2) Write 0x5678 for EHA[15:0] while HPXA=0.
(3) Write EHA[31:16] offset value(=0x0E) while HPXA = 1. EHIND register indicates EHA[31:16] register.
(4) Write 0x1234 for EHA[31:16] while HPXA = 0.

Figure 5.3 and Figure 5.4 show that the external host device writes to and reads from EHA register.

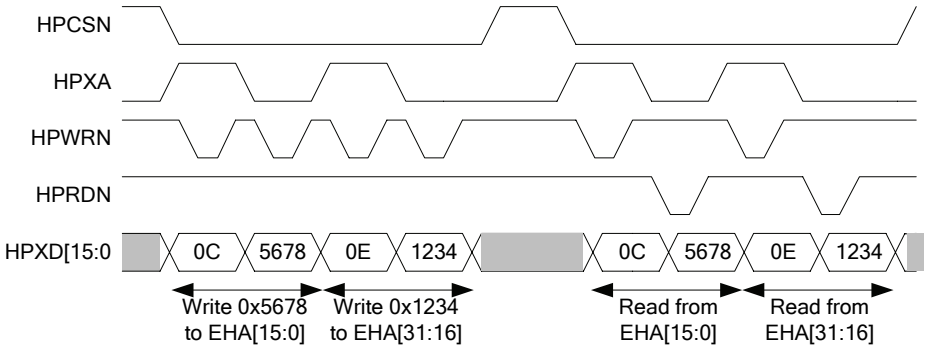


Figure 5.3 Example of writing / reading operation (80 interface, 16bits)

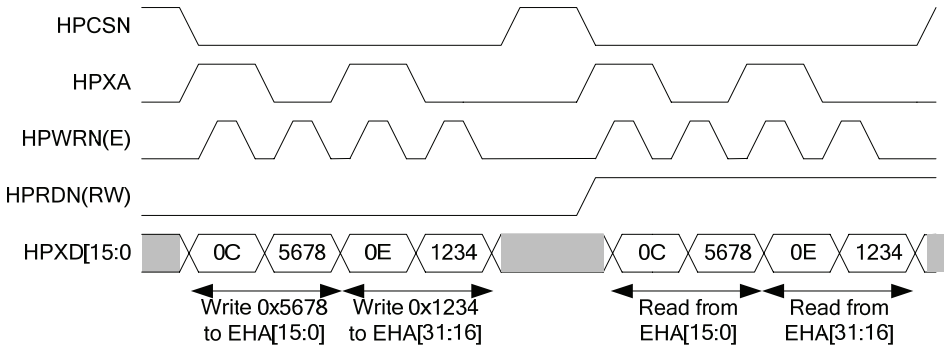


Figure 5.4 Example of writing / reading operation (68 interface, 16bits)

5.3.2 Access to the On-chip System Bus

The external host device that is connected to this LSI via EHI can access the on-chip system bus. This connection supports word-aligned (32-bits) burst transfer. When the external host device access the on-chip system bus, HCLK and ECLK should be enabled in advance.

The length of burst transfer is determined by EHRWCS.BSIZE. Once transfer is occurred, the length of burst cannot be modified until the requested transfer is completed. Therefore, EHRWCS.BSIZE, EHRWCS.AI, and EHRWCS.LK can be only modified while EHRWCS.RW = 0. If EHST.RDY is checked every burst transfer, the FIFO underrun or overrun cannot be occurred. But, if the burst transfers are intended to be issued without EHST.RDY check to improve data transfer efficiency, HCLK of this LSI should be fast enough not to underrun and overrun. FIFO underrun and overrun can be detected via EHST.FUR and EHST.FOR respectively.

Figure 5.5 shows how to program the external host device for access to the on-chip system bus.

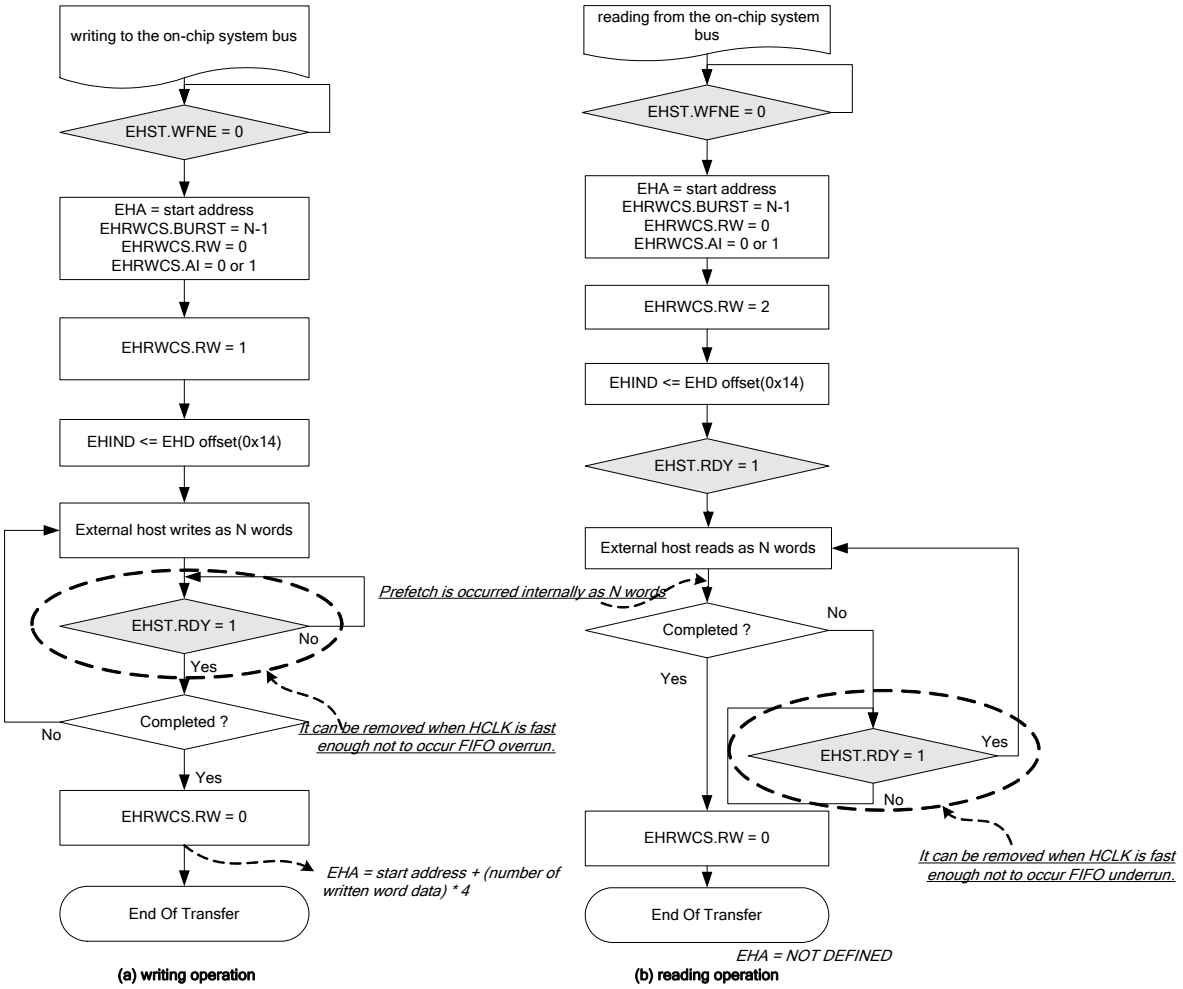


Figure 5.5 Access to the On-chip System Bus

Write-operation

The external host should configure writing operation before writing data; start address, address auto increment, lock or normal, and burst length.

The start address should be written to EHA register. When address auto increment is enable, address of the on-chip system bus is increment automatically every word transfer. This generated address is masked by EHAM. Therefore, address on the on-chip system bus is next EHA = (EHA[31:2] + 1) & ~EHAM[31:2]. When the writing operation is completed (EHRWCS.RW = 0), EHA has start address + number of written data.

EHRWCS.BSIZE specifies the maximum of burst length on the on-chip system. When the lock transfer mode is enabled, the bus handover is not occurred during burst transfer. But, when lock transfer mode is disabled and a higher priority master requests bus access, the bus handover is occurred and EHI burst transfer is terminated. And when the higher priority master looses access to bus, EHI burst transfer is continued. But, when the lock transfer mode is enabled, the overall system performance of the on-chip can be degraded.

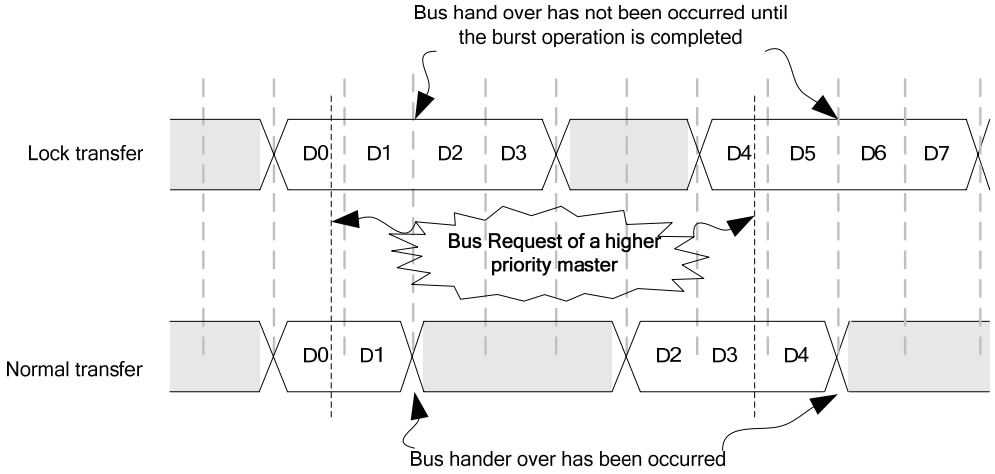


Figure 5.6 Lock tranfser vs Normal transfer (EHRWCS.BSIZE=3)

After configuring writing operation, the external host enables writing operation as EHRWCS.RW is set to 1 and EHIND is set to EHD offset (0x14) sequentially. From now on, the external host can write data to the on-chip system bus while HPXA is low. For finishing the writing operation, the external host should set EHRWCS.RW to 0. After that, to confirm that all of written data arrived at destination area, the external host can check whether EHST.RDY is equal to 1.

The auto-increment writing operation which has that start address is 0x12345678, and the burst length is 8 is shown in Figure 5.7.

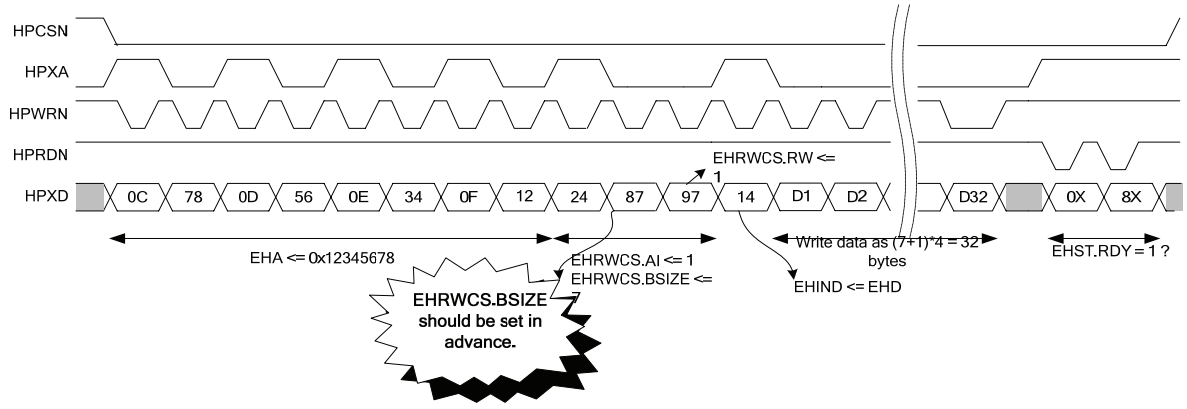


Figure 5.7 Example of Writing to the On-chip System Bus (80 interface, 8bits)

Read-operation

The configuration of reading operation is similar to that of the writing operation; start address, address auto increment, lock or normal, and burst length.

EHST.RDY may be checked every EHRWCS.BSIZE+1 words. If HCLK is fast enough not to underrun, EHST.RDY do not need to be check.

The auto-increment reading operation which has that start address is 0x12345678, and burst length is 8 is shown in Figure 5.8.

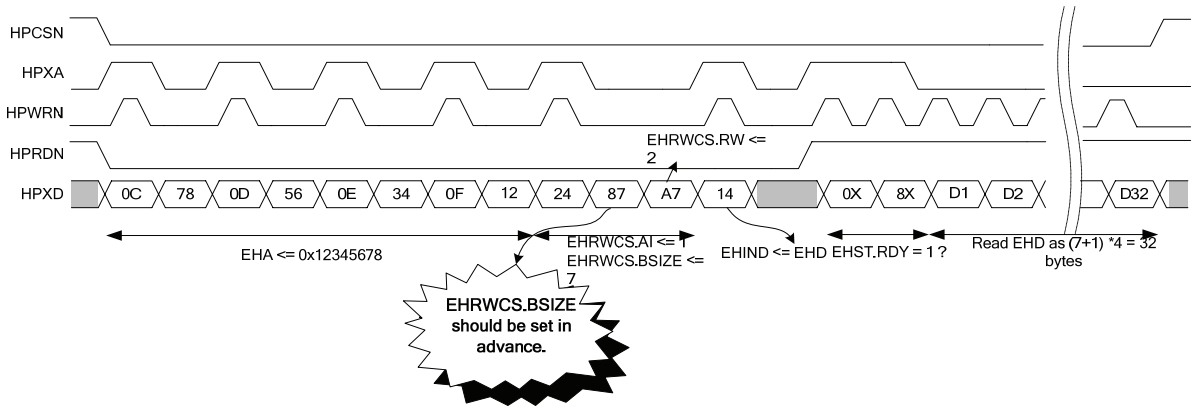


Figure 5.8 Example of Reading from the on-chip System Bus (68 interface, 8bits)

Notice that EHRWCS.RW is cleared to 0 automatically after the single reading and writing operation.

5.3.3 Entrance to the Critical Section Using the Semaphore Register

EHI supports the semaphore for improving data transfer efficiency and EHSEM register is used.

```
int get_sem (void)
{
        if (EHSEM.FLG & 1)
        {
                return FALSE;
        }
        else
        {
                return TRUE;
        }
}

void release_sem (void)
{
        EHSEM.FLG = 0;
}
```

On-Chip CPU

```
int get_sem (void)
{
        sem = read EHSEM reg.;
        if (sem & 2)
        {
                return FALSE;
        }
        else
        {
                return TRUE;
        }
}

void release_sem (void)
{
        write 0 to EHSEM.FLG;
}
```

External Host

Figure 5.9 Pseudo code for getting and releasing a semaphore

5.3.4 Interrupt Request

The EHI has three kinds of interrupt requests; external software interrupt request, internal software interrupt request, and internal CS/WR interrupt request

The first, the external software interrupt request is generated via HPINT pin. Therefore, the HPINT pin should be configured as interrupt request pin. It is default value. After this configuration, the HPINT pin is controlled by EHEINT.IRQ bit directly.
The second, the external host can request an interrupt to the on-chip CPU. This internal software interrupt request is issued when EHIINT .IIRQ is set to 1.

Finally, internal CS/WR interrupt request uses HPCSN, HPWRN, HPRDN signals as interrupt source. For using HPCSN signal as the interrupt source, EHCFG.CSIRQ bit should be set to 1. When EHCFG.CSIRQ is equal to 1 and HPCSN is asserted, the interrupt request is issued. The interrupt service routine of the on-chip CPU should clear EHCFG.CSIRQ to 0 not to generate the additional interrupt request by HPCSN signal. In the case of using writing operation as interrupt source, EHCFG.WRIRQ bit should be set to 1. After that, when a writing operation is issued (80-mode: HPCSN=0 HPWRN=0, 68-mode: HPCSN=0, HPWRN=1, HPRDN=0), the interrupt is generated. The external host can issue this interrupt request regardless of ECLK.

6 DMA CONTROLLER

6.1 Overview

The NVS2310 has four 3-channel general DMA (GDMA) controllers for data transfer. IOBUS has three general controller. GDMA that the other has HSIOBUS. The block diagram of GDMA controller is in the following figure.

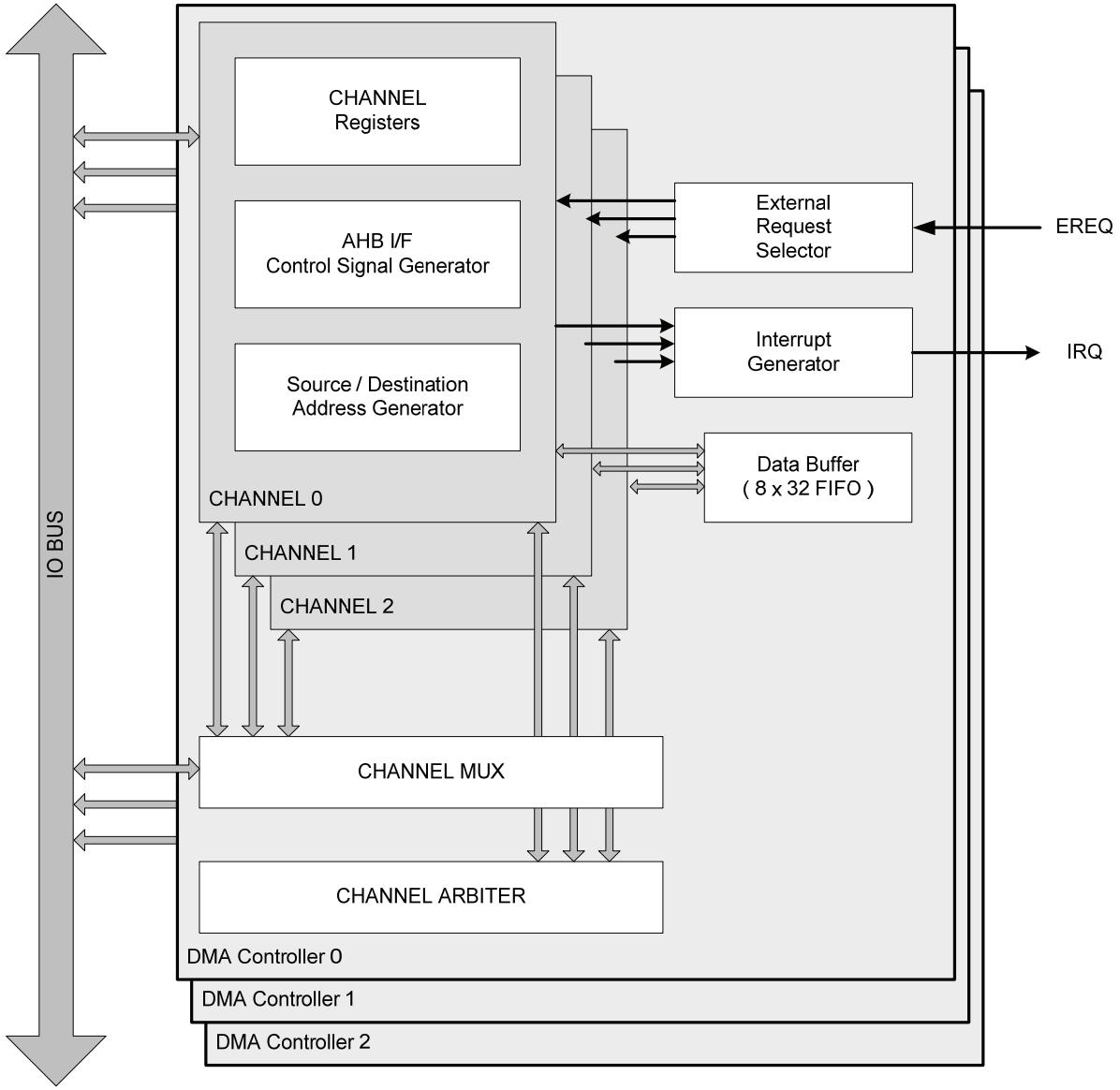


Figure 6.1 GDMA Controller Block Diagram In IOBUS

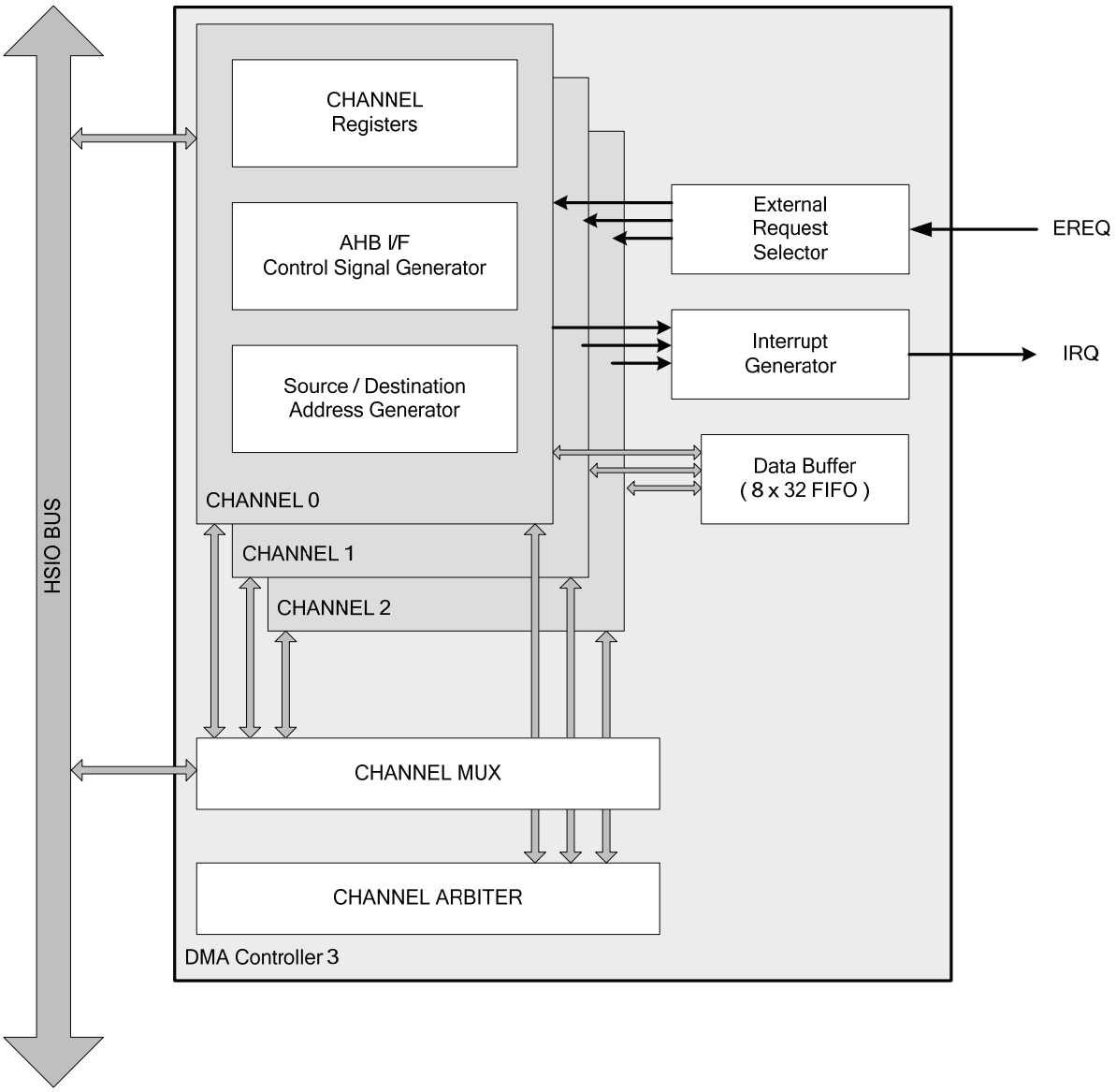


Figure 6.2 GDMA Controller Block Diagram In HSIO BUS

6.2 Register Description

Table 6.1 General DMA Controller Register Map in IOBUS (DMA Base Address = 0xB0030n00, n=0,1,2)

Name		offset	Type	Reset	Description
CHANNEL 0	ST_SADR0	0x00	R/W	0x00000000	Start Address of Source Block
	SPARAM0	0x04	R/W	0x00000000	Parameter of Source Block
	C_SADR0	0x0C	R	0x00000000	Current Address of Source Block
	ST_DADR0	0x10	R/W	0x00000000	Start Address of Destination Block
	DPARAM0	0x14	R/W	0x00000000	Parameter of Destination Block
	C_DADR0	0x1C	R	0x00000000	Current Address of Destination Block
	HCOUNT0	0x20	R/W	0x00000000	Initial and Current Hop count
	CHCTRL0	0x24	R/W	0x00000000	Channel Control Register
	RPTCTRL0	0x28	R/W	0x00000000	Repeat Control Register
	EXTREQ0	0x2C	R/W	0x00000000	External DMA Request Register
CHANNEL 1	ST_SADR1	0x30	R/W	0x00000000	Start Address of Source Block
	SPARAM1	0x34	R/W	0x00000000	Parameter of Source Block
	C_SADR1	0x3C	R	0x00000000	Current Address of Source Block
	ST_DADR1	0x40	R/W	0x00000000	Start Address of Destination Block
	DPARAM1	0x44	R/W	0x00000000	Parameter of Destination Block
	C_DADR1	0x4C	R	0x00000000	Current Address of Destination Block
	HCOUNT1	0x50	R/W	0x00000000	Initial and Current Hop count
	CHCTRL1	0x54	R/W	0x00000000	Channel Control Register
	RPTCTRL1	0x58	R/W	0x00000000	Repeat Control Register
	EXTREQ1	0x5C	R/W	0x00000000	External DMA Request Register
CHANNEL 2	ST_SADR2	0x60	R/W	0x00000000	Start Address of Source Block
	SPARAM2	0x64/0x68	R/W	0x00000000	Parameter of Source Block
	C_SADR2	0x6C	R	0x00000000	Current Address of Source Block
	ST_DADR2	0x70	R/W	0x00000000	Start Address of Destination Block
	DPARAM2	0x74/0x78	R/W	0x00000000	Parameter of Destination Block
	C_DADR2	0x7C	R	0x00000000	Current Address of Destination Block
	HCOUNT2	0x80	R/W	0x00000000	Initial and Current Hop count
	CHCTRL2	0x84	R/W	0x00000000	Channel Control Register
	RPTCTRL2	0x88	R/W	0x00000000	Repeat Control Register
	EXTREQ2	0x8C	R/W	0x00000000	External DMA Request Register
CHCONFIG		0x90	R/W	0x00000000	Channel Configuration Register

Table 6.2 General DMA Controller Register Map in HSIO BUS (DMA Base Address = 0xB0830000)

Name		offset	Type	Reset	Description
CHANNEL 0	ST_SADR0	0x00	R/W	0x00000000	Start Address of Source Block
	SPARAM0	0x04	R/W	0x00000000	Parameter of Source Block
	C_SADR0	0x0C	R	0x00000000	Current Address of Source Block
	ST_DADR0	0x10	R/W	0x00000000	Start Address of Destination Block
	DPARAM0	0x14	R/W	0x00000000	Parameter of Destination Block
	C_DADR0	0x1C	R	0x00000000	Current Address of Destination Block
	HCOUNT0	0x20	R/W	0x00000000	Initial and Current Hop count
	CHCTRL0	0x24	R/W	0x00000000	Channel Control Register
	RPTCTRL0	0x28	R/W	0x00000000	Repeat Control Register
	EXTREQ0	0x2C	R/W	0x00000000	External DMA Request Register
CHANNEL 1	ST_SADR1	0x30	R/W	0x00000000	Start Address of Source Block
	SPARAM1	0x34	R/W	0x00000000	Parameter of Source Block
	C_SADR1	0x3C	R	0x00000000	Current Address of Source Block
	ST_DADR1	0x40	R/W	0x00000000	Start Address of Destination Block
	DPARAM1	0x44	R/W	0x00000000	Parameter of Destination Block
	C_DADR1	0x4C	R	0x00000000	Current Address of Destination Block
	HCOUNT1	0x50	R/W	0x00000000	Initial and Current Hop count
	CHCTRL1	0x54	R/W	0x00000000	Channel Control Register
	RPTCTRL1	0x58	R/W	0x00000000	Repeat Control Register
	EXTREQ1	0x5C	R/W	0x00000000	External DMA Request Register
CHANNEL 2	ST_SADR2	0x60	R/W	0x00000000	Start Address of Source Block
	SPARAM2	0x64/0x68	R/W	0x00000000	Parameter of Source Block
	C_SADR2	0x6C	R	0x00000000	Current Address of Source Block
	ST_DADR2	0x70	R/W	0x00000000	Start Address of Destination Block
	DPARAM2	0x74/0x78	R/W	0x00000000	Parameter of Destination Block
	C_DADR2	0x7C	R	0x00000000	Current Address of Destination Block
	HCOUNT2	0x80	R/W	0x00000000	Initial and Current Hop count
	CHCTRL2	0x84	R/W	0x00000000	Channel Control Register
	RPTCTRL2	0x88	R/W	0x00000000	Repeat Control Register
	EXTREQ2	0x8C	R/W	0x00000000	External DMA Request Register
CHCONFIG		0x90	R/W	0x00000000	Channel Configuration Register

The GDMA registers are listed in Table 6.1. and Table 6.12 The NVS2310 has 4 GDMA, which are GDMA0, GDMA1, GDMA2, and GDMA3. And their base addresses are 0xB0030000, 0xB0030100, 0xB0030200 and 0xB0830000 respectively. One GDMA has 3 channels, which are Channel 0, Channel 1 and Channel 2. Channel offset is 0x00, 0x30, and, 0x60 as shown in Table 6.1 and Table 6.3

Table 6.3 BASE Address(DMA_BASE) of All GDMA Channles

GDMA NAME	GDMA BASE	CHANNEL #	BASE Address
GDMA0	0xB0030000	0	0xB0030000
		1	0xB0030030
		2	0xB0030060
GDMA1	0xB0030100	0	0xB0030100
		1	0xB0030130
		2	0xB0030160
GDMA2	0xB0030200	0	0xB0030200
		1	0xB0030230
		2	0xB0030260
GDMA3	0xB0830000	0	0xB0830000
		1	0xB0830000
		2	0xB0830000

Start Source Address Register (ST_SADR) DMA_BASE+0x00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ST_SADR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST_SADR[15:0]															

Field	Name	RW	Reset	Description
31-0	ST_SADR	R/W	0x00000000	DMA Start Source Address

This register contains the start address of source memory block for DMA transfer. The transfer begins reading data from this address.

Source Block Parameter Register (SPARAM) DMA_BASE + 0x04

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SMASK[23:8]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMASK[7:0]								SINC[7:0]							

Field	Name	RW	Reset	Description
31-8	SMASK	R/W	0x0000000	0 : non-masked 1 : Masked so that source address bit doesn't be changed during DMA transfer
7-0	SINC	R/W	0x00	Sinc : Source address is added by amount of sinc at every write cycles. sinc is represented as 2's complement, so if SINC[7] is 1, the source address is decremented.

Each bit field controls the dedicated bit of source address field. That is, if SMASK[23] is set to 1, the 28th bit of source address is masked, and if SMASK[22] is set to 1, the 27th bit of source address is masked, and so on. If a bit is masked, a corresponding bit of address bus is not changed during DMA transfer. This function can be used to generate circular buffer address.

The addresses of DMA transfer are 32bit wide, but the upper 4bit of them are not affected during DMA transfer. If the source or destination address reaches its maximum address space like 0x7FFFFFFF or 0x2FFFFFFF, the next transfer is starting from 0x70000000 or 0x20000000 not from 0x80000000 or 0x30000000.

Current Source Address Register (C_SADR) DMA_BASE + 0x0C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C_SADR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
C_SADR[15:0]															

Field	Name	RW	Reset	Description
31-0	C_SADR	R	0x00000000	DMA Current Source Address

This register contains the current source address of DMA transfer. It represents that the current transfer read data from this address. This is read only register.

Start Destination Address Register (ST_DADR) DMA_BASE + 0x10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ST_DADR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST_DADR[15:0]															

Field	Name	RW	Reset	Description
31-0	ST_DADR	R/W	0x00000000	DMA Start Destination Address

This register contains the start address of destination memory block for DMA transfer.

Destination Block Parameter Register (DPARAM) DMA_BASE + 0x14

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMASK[23:8]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMASK[7:0]								DINC[7:0]							

Field	Name	RW	Reset	Description
31-8	DMASK	R/W	0x000000	0 : non-masked 1 : Masked so that destination address bit doesn't be changed during DMA transfer
7-0	DINC	R/W	0x00	Dinc : Destination address is added by amount of dinc at every write cycles. dinc is represented as 2's complement, so if DINC[7] is 1, the destination address is decremented.

Each bit field controls the corresponding bit of source address field. That is, if DMASK[23] is set to 1, the 28th bit of source address is masked, and if DMASK[22] is set to 1, the 27th bit of source address is masked, and so on. If a bit is masked, a corresponding bit of address bus is not changed during DMA transfer. This function can be used to generate circular buffer address.

The addresses of DMA transfer have 32bit wide, but the upper 4bit of them are not affected during DMA transfer. If the source or destination address reaches its maximum address space like 0x7FFFFFFF or 0x2FFFFFFF, the next transfer is starting from 0x70000000 or 0x20000000 not from 0x80000000 or 0x30000000.

Current Destination Address Register (C_DADR) DMA_BASE + 0x1C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C_DADR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
C_DADR[15:0]															

Field	Name	RW	Reset	Description
31-0	C_DADR	R	0x00000000	DMA Current Destination Address

This register contains current destination address of DMA transfer. It represents that the current transfer write data to this address. This is read only register.

HOP Count Register (HCOUNT) DMA_BASE + 0x20

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C_HCOUNT[15:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST_HCOUNT[15:0]															

Field	Name	RW	Reset	Description
31-16	C_HCOUNT	R	0x0000	Cn : Represent cn number of Hop transfer remains
15-0	ST_HCOUNT	R/W	0x0000	Sn : DMA transfers data by amount of sn Hop transfers

At the beginning of transfer, the C_HCNT is updated by ST_HCNT register. At the end of every hop transfer, this is decremented by 1 until it reaches to zero. When this reaches to zero, the DMA finishes its transfer and may or may not generate its interrupt according to IEN flag of CHCTRL register.

Channel Control Register (CHCTRL) DMA_BASE + 0x24

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CONT	DTM	SYN	HRD	LOCK	BST	TYPE		BSIZE		WSIZE		FLG	IEN	REP	EN

Field	Name	RW	Reset	Description
15	CONT	R/W	0x0	0 : DMA transfer begins from ST_SADR / ST_DADR address 1 : DMA transfer begins from C_SADR / C_DADR address It must be used after the former transfer has been executed, so that C_SADR and C_DADR contain a meaningful value.
14	DTM	R/W	0x0	0 : Differential Transfer Mode Disable 1 : Differential Transfer Mode Enable for WSIZE = 10 and BSIZE = 11: 32 bit-to- 16bit transfer 4 Read(Word Unit) / 8 Write(Half-Word Unit) for WSIZE = 11 and BSIZE = 11 : 16 bit-to- 32bit transfer. 8 Read(Half-Word Unit) / 4 Write(Word Unit)
13	SYN	R/W	0x0	0 : Do not Synchronize Hardware Request. 1 : Synchronize Hardware Request.
12	HRD	R/W	0x0	0 : ACK/EOT signals are issued when DMA-Read Operation. 1 : ACK/EOT signals are issued When DMA-Write Operation.
11	LOCK	R/W	0x0	Lock field controls the LOCK signal (refer to AHB specification). When the LOCK is set to 1, the DMA transfer is not bothered by other AHB masters like LCD controller, ARM etc. This field is only meaningful in case of non-burst type transfers. 1 : DMA transfer executed with lock transfer
10	BST	R/W	0x0	Arbitration means that at the end of every HOP transfer, the AHB bus is released from DMA channel so other master can occupy the bus when that master has requested the bus. Burst means that once the DMA request occurs, all of transfers are executed without further DMA requests. 0 : DMA transfer executed with arbitration. 1 : DMA transfer executed with no arbitration. (burst operation)
9-8	TYPE	R/W	0x0	In SINGLE Type, after one Hop data transferring DMA checks External DMA Request (DREQ) and then if its bit is active, DMA transfers next hop data. DREQ is detected level-sensitive or edge-triggered by SINGLE transfer TYPE. The 1 Hop of transfer means 1 burst of read followed by 1 burst of write. 1 burst means 1, 2 or 4 consecutive read or write-cycles defined by BSIZE field of CHCTRL register. The Figure 6.3 illustrates the relation among the above transfers. 00 : SINGLE transfer with edge-triggered detection 11 : SINGLE transfer with level-sensitive detection 01 : HW transfer 10 : SW transfer
7-6	BSIZE	R/W	0x0	0 : 1 Burst transfer consists of 1 read or write cycle. 1 : 2 Burst transfer consists of 1 read or write cycles. 2 : 4 Burst transfer consists of 1 read or write cycles. 3 : 8 Burst transfer consists of 1 read or write cycles.
5-4	WSIZE	R/W	0x0	0 : Each cycle read or write 8bit data 1 : Each cycle read or write 16bit data

				2, 3 : Each cycle read or write 32bit data
3	FLG	R/W	0x0	It is not automatically cleared by starting another transfer, so before starting any other DMA transfer, user must clear this flag to 0 for checking DMA status correctly. 0 : Not yet completed DMA transfer 1 : Represents that all hop of transfers are fulfilled. When writing 1 to this bit, it is cleared to 0
2	IEN	R/W	0x0	To generate IRQ or FIQ interrupt, the DMA flag of IEN register in the interrupt controller must be set to 1 ahead. 0 : DMA interrupt disabled 1 : At the same time the FLAG goes to 1, DMA interrupt request is generated.
1	REP	R/W	0x0	0 : After all of hop transfer has executed, the DMA channel is disabled 1 : The DMA channel remains enabled. When another DMA request has occurred, the DMA channel start transfer data again with the same manner (type, address, increment, mask) as the latest transfer of that channel.
0	EN	R/W	0x0	0 : DMA channel is terminated and disabled. It does not affect the HCOUNT register, so if the current hop counter is not zero when channel is disabled, it is possible that the transfer illegally starts right after channel is re-enabled. Make sure that HCOUNT is zero not to continue transfer after channel is re-enabled. 1 : DMA channel is enabled. If software type transfer is selected, this bit generates DMA request directly, or if hardware type transfer is used, the selected interrupt request flag generate DMA request.

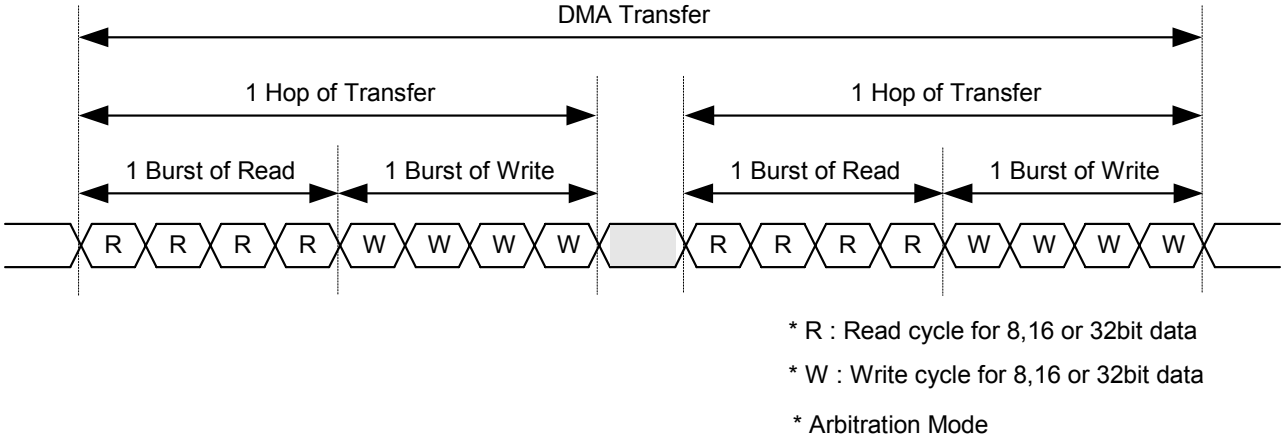


Figure 6.3 Relation between Hop and Burst Transfers (If burst size is 4.)

Hardware type transfer means that the DMA transfer is triggered by external or internal hardware blocks selected by DMASEL field in CHCTRL register.

Software type transfer means that the DMA transfer is triggered by EN bit of CHCTRL register. When this is set to 1, transfer request signal is generated internally and then the transfer begins immediately.

Hardware demand type transfer (HW_DEMAND) means that once the DMA request occurs,

DMA checks request signal each hope transfer, and if request signal is set, DMA transfer one hope's data. After transferring all hope's data, DMA operation will be finished.

Figure 6.4 shows the example of various types of transfer.

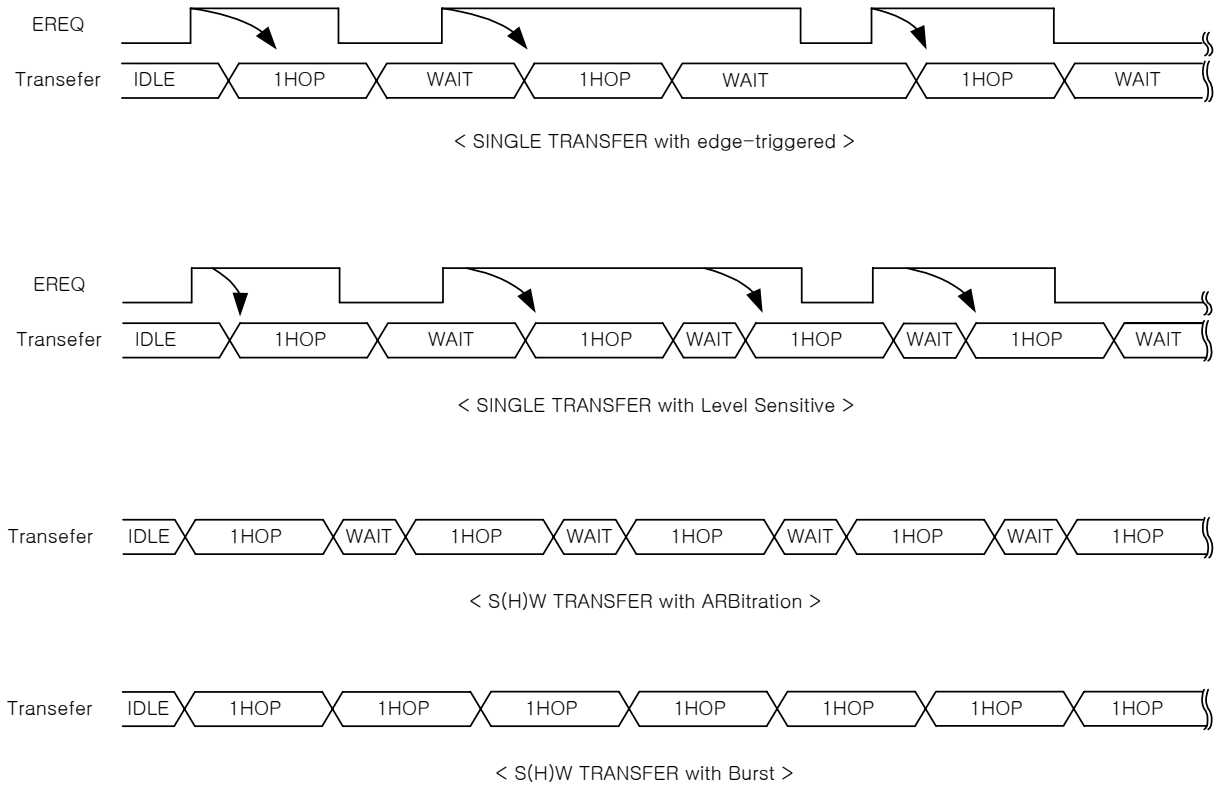


Figure 6.4 The Example Of Various Types of Transfer.

Repeat Control Register (RPTCTRL) **DMA_BASE + 0x28**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRI	EOT	0						RPTCNT[23:16]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RPTCNT[15:0]															

Field	Name	RW	Reset	Description
31	DRI	R/W	0x0	This bit is meaningful when Repeat Mode is enabled. 0 : DMA Interrupt is occurred at the end of each repeated DMA operation. 1 : DMA Interrupt is occurred at the last DMA repeated DMA operation.
30	EOT	R/W	0x0	This bit is meaningful when Repeat Mode is enabled. 0 : EOT signal is occurred at the end of each repeated DMA operation in HW(including Single) transfer mode. 1 : EOT Signal is occurred at the last repeated DMA operation in HW(including Single) transfer mode.
23-0	RPTCNT	R/W	0x000000	This bit is meaningful when Repeat Mode is enabled. When this bit is cleared in repeat mode, DMA will run endlessly. To exit endless repeat mode, clear EN bit of DMACTRL or disable Repeat Mode. It's possible to circular transfer using repeat count. 0 : DMA transfer data endlessly. None zero : DMA transfer the number of (N + 1) * HCOUNT data

External DMA Request Register (EXTREQn)

DMA_BASE + 0x2C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMASEL[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMASEL[15:0]															

Field	Name	RW	Reset	Description
31-0	DMASEL	R/W	0x00000000	

DMASEL	Connected Hardware
[0]	GPSB Channel 3 TX
[1]	GPSB Channel 4 TX
[2]	GPSB Channel 5 TX
[3]	TS I/F 0
[4]	GPSB Channel 3 RX
[5]	GPSB Channel 4 RX
[6]	GPSB Channel 5 RX
[7]	TS I/F 1
[8]	UART Channel 2 Transmitter
[9]	UART Channel 2 Receiver
[10]	UART Channel 3 Transmitter
[11]	UART Channel 3 Receiver
[12]	TS I/F PID0
[13]	-
[14]	-
[15]	Test Block
[16]	Memory Stick 0
[17]	Memory Stick 1
[18]	NAND Flash Controller
[19]	I2C Channel 0
[20]	SPDIF Transmitter - Packet(Audio) Data
[21]	SPDIF Transmitter - User Data
[22]	CD I/F
[23]	DAI Transmitter
[24]	DAI Receiver
[25]	I2C Channel 1
[26]	UART Channel 0 Transmitter
[27]	UART Channel 0 Receiver
[28]	SPDIF Receiver
[29]	UART Channel 1 Transmitter
[30]	UART Channel 1 Receiver
[31]	TS I/F PID1

Channel Configuration Register (CHCONFIG) DMA_BASE +0x90

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									IS2	IS1	IS0	0	MIS2	MIS1	MIS0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0					SWP2	SWP1	SWP0	0	PRI[2:0]			0			FIX

Field	Name	RW	Reset	Description
22	IS2	R/W	0x0	Without regard to Interrupt enable bit(IEN) of the channel2, this bit indicates the channel2 interrupt status. This bit is automatically cleared when FLAG bit of the corresponding channel is cleared. This bit is read only. 0 : No interrupt in the corresponding channel 1 : The corresponding channel interrupt is occurred
21	IS1	R/W	0x0	Except for channel difference, this bit is the same as IS2 bit. 0 : No interrupt in channel 1 1 : Channel1 Interrupt is occurred
20	IS0	R/W	0x0	Except for channel difference, this bit is the same as IS2 bit. 0 : No interrupt in channel 0 1 : Channel1 Interrupt is occurred
18	MIS2	R/W	0x0	This bit is set when the channel2 interrupt occurs and interrupt enable bit (IEN) of channel2 is set. This bit is automatically cleared when FLAG bit of the channel2 is cleared. This bit is read only. 0 : Masked interrupt is not occurred in channel 2 1 : Channel2 Masked Interrupt is occurred
17	MIS1	R/W	0x0	Except for channel difference, this bit is the same as MIS2 bit. 0 : Masked interrupt is not occurred in channel 1 1 : Channel1 Masked Interrupt is occurred
16	MIS0	R/W	0x0	Except for channel difference, this bit is the same as MIS1 bit. 0 : Masked interrupt is not occurred in channel 0 1 : Channel0 Masked Interrupt is occurred
10	SWP2	R/W	0x0	When this bit is set, data to be written to destination address will be swapped. For example, the 32bit source data which consists of 4bytes {D3, D2, D1, D0} will be stored {D0, D1, D2, D3} in destination address. The 16bit source data which consists of 2bytes {D1, D0} will be stored {D0,D1} in destination address. 0 : Do not swap Data. 1 : Swap Channel Data.
9	SWP1	R/W	0x0	Except for channel difference, the function controlled by this bit is the same as its SWP2 bit. 0 : Do not Swap Channel1 Data. 1 : Swap Channel1 Data.
8	SWP0	R/W	0x0	Except for channel difference, the function controlled by this bit is the same as its SWP2 bit. 0 : Do not Swap Channel0 Data. 1 : Swap Channel Data.
6-4	PRI	R/W	0x0	PRI bits is meaningful when fix[0] bit is enabled. 000 : CH0 > CH1 > CH2 001 : CH0 > CH2 > CH1 010 : CH1 > CH0 > CH2 011 : CH1 > CH2 > CH0 100 : CH2 > CH1 > CH0 101 : CH2 > CH0 > CH1
0	FIX	R/W	0x0	In round-robin mode, each channel is enabled one by one every one hop transferring. 0 : Round-Robin (Cyclic) Mode. 1 : Fixed Priority Mode.

In Fixed mode, according to PRI bit, the highest channel is serviced first and lower priority channel is serviced after higher priority channel operation is finished. See
Figure 6.5 for more information.

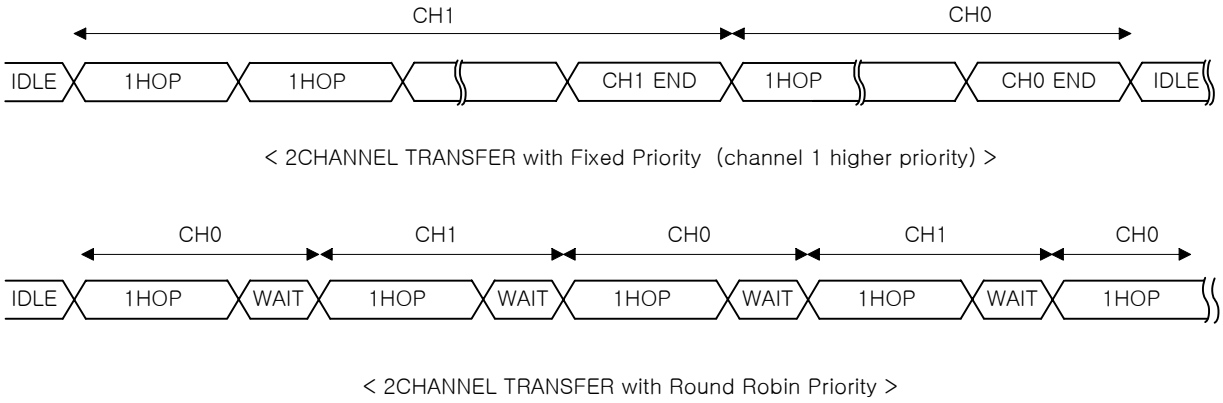


Figure 6.5 Data transfer when Channel0 and Channel1 are enabled

7 GPIO & Port Multiplexing

7.1 Overview

The NVS2310 has a lot of general purpose I/Os which is able to be programmed by setting internal registers. All I/Os are set to input mode at reset. The block diagram of GPIO is in the following figure.

The GPIO in the NVS2310 has the following key features.

The I/Os have the multiple functions shared by various on-chip hardware controllers.

The GPSB, memory stick host controller, UART, and SD/MMC controller have the capability of interfacing to external device via the multi-channel ports by multiplexing the in/out signals of the corresponding hardware.

Most of the I/Os can be pulled-up or pulled-down by reconfigurable register.

The external interrupts passed to the interrupt controller can be selected from the various sources.

The GPIO controller has the special function. And the various I/Os are pulled-up or pulled-down at system reset to reduce the boot-up power consumption during the system reset, which prevents the reset state of the various I/Os from being floating state.

But, for this reason, if you want to design the specific I/Os with pull-up state which are pulled-down by reset configuration, you should be careful in choosing the value of resistor and the value of the pull-up/down control registers should be changed as soon as possible – for example, in the reset handler of initial boot code.

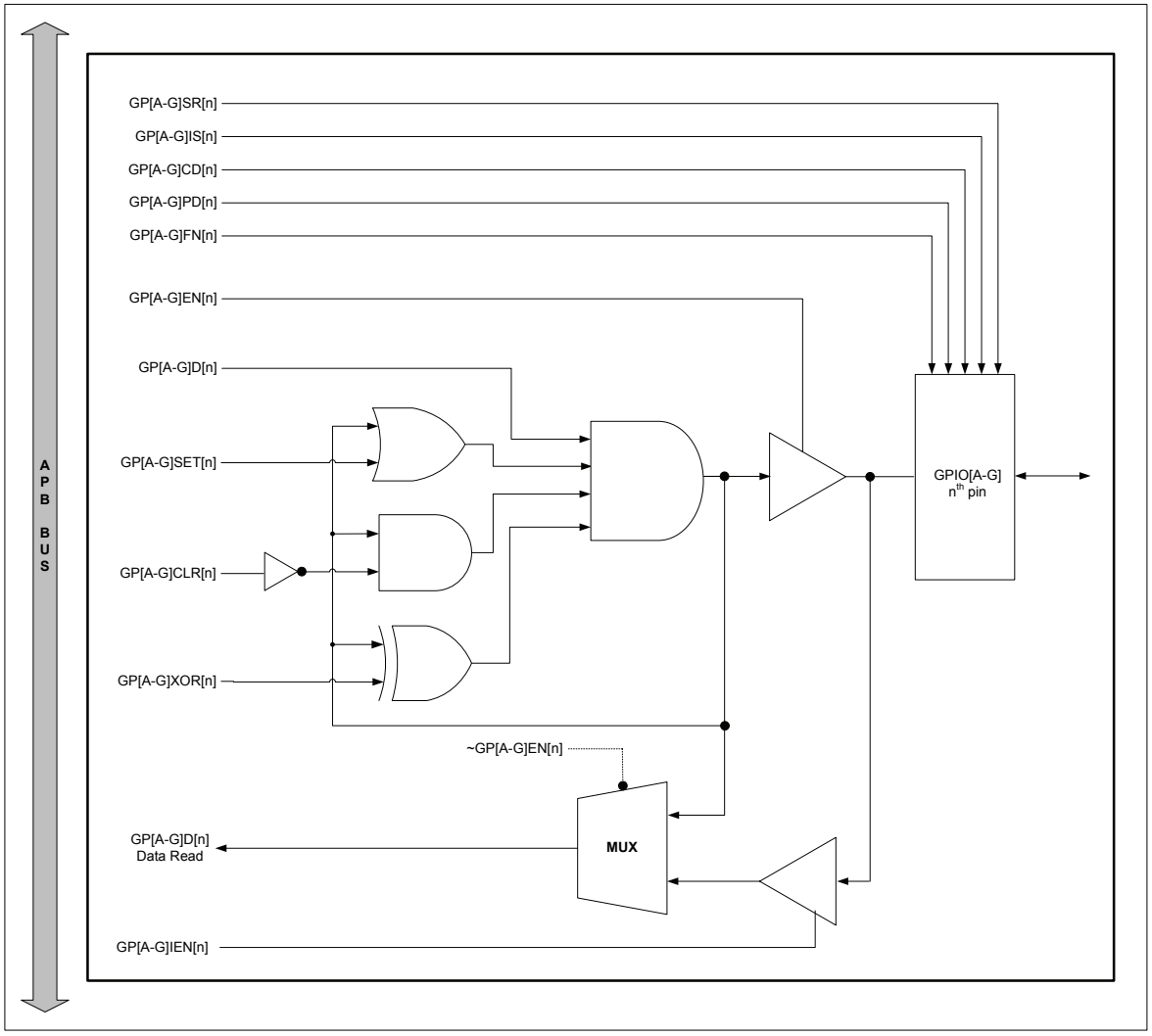


Figure 7.1 Block Diagram of GPIO

7.2 Register Description

Table 7.1 GPIO Register Map (Base Address = 0xB010A000)

Name	Offset	Type	Reset	Description
GPADAT	0x000	R/W	0x00000000	GPA Data Register
GPAOEN	0x004	R/W	0x00000000	GPA Output Enable Register
GPASET	0x008	W	-	OR function on GPA Output Data
GPACLR	0x00C	W	-	BIC function on GPA Output Data
GPAXOR	0x010	W	-	XOR function on GPA Output Data
GPACD0	0x014	R/W	0x55555555	Driver strength Control 0 on GPA Output Data
GPACD1	0x018	R/W	0x00000000	Driver strength Control 1 on GPA Output Data
GPAPD0	0x01C	R/W	0x55555555	Pull UP/DOWN Control Register 0 on GPA Output Data
GPAPD1	0x020	R/W	0x00000000	Pull UP/DOWN Control Register 1 on GPA Output Data
GPAFN0	0x024	R/W	0x00000000	Port Configuration on GPA Output Data
GPAFN1	0x028	R/W	0x00000000	Port Configuration on GPA Output Data
GPAFN2	0x02C	R/W	0x00000000	Port Configuration on GPA Output Data
GPAFN3	0x030	R/W	0x00000000	Port Configuration on GPA Output Data
GPAIEN	0x034	R/W	0xFFFFFFFF	GPA Input Enable Register
GPAIS	0x038	R/W	0x00000000	GPA input Type Selection Register
GPASR	0x03C	R/W	0x00000000	GPA Slew rate Control Register
-	0x034-0x03C			Reserved
GPBDAT	0x040	R/W	0x00000000	GPB Data Register
GPBEN	0x044	R/W	0x00000000	GPB Output Enable Register
GPBSET	0x048	W	-	OR function on GPB Output Data
GPBCLR	0x04C	W	-	BIC function on GPB Output Data
GPBXOR	0x050	W	-	XOR function on GPB Output Data
GPBCD0	0x054	R/W	0x55555555	Driver strength Control 0 on GPB Output Data
GPBCD1	0x058	R/W	0x00000000	Driver strength Control 1 on GPB Output Data
GPBPD0	0x05C	R/W	0x55555555	Pull UP/DOWN Control Register 0 on GPB Output Data
GPBPD1	0x060	R/W	0x00000000	Pull UP/DOWN Control Register 1 on GPB Output Data
GPBFN0	0x064	R/W	0x00000000	Port Configuration on GPB Output Data
GPBFN1	0x068	R/W	0x00000000	Port Configuration on GPB Output Data
GPBFN2	0x06C	R/W	0x00000000	Port Configuration on GPB Output Data
GPBFN3	0x070	R/W	0x00000000	Port Configuration on GPB Output Data
GPBIEN	0x064	R/W	0xFFFFFFFF	GPB Input Enable Register
GPBIS	0x068	R/W	0x00000000	GPB input Type Selection Register
GPBSR	0x06C	R/W	0x00000000	GPB Slew rate Control Register
-	0x074-0x07C			Reserved
GPCDAT	0x080	R/W	0x00000000	GPC Data Register
GPCEN	0x084	R/W	0x00000000	GPC Output Enable Register
GPCSET	0x088	W	-	OR function on GPC Output Data
GPCCLR	0x08C	W	-	BIC function on GPC Output Data
GPCXOR	0x090	W	-	XOR function on GPC Output Data
GPCCD0	0x094	R/W	0x55555555	Driver strength Control 0 on GPC Output Data
GPCCD1	0x098	R/W	0x00000000	Driver strength Control 1 on GPC Output Data
GPCPD0	0x09C	R/W	0x55555555	Pull UP/DOWN Control Register 0 on GPC Output Data
GPCPD1	0x0A0	R/W	0x00000000	Pull UP/DOWN Control Register 1 on GPC Output Data
GPCFN0	0x0A4	R/W	0x00000000	Port Configuration on GPC Output Data
GPCFN1	0x0A8	R/W	0x00000000	Port Configuration on GPC Output Data
GPCFN2	0x0AC	R/W	0x00000000	Port Configuration on GPC Output Data
GPCFN3	0x0B0	R/W	0x00000000	Port Configuration on GPC Output Data
GPCIEN	0x0B4	R/W	0xFFFFFFFF	GPC Input Enable Register
GPCIS	0x0B8	R/W	0x00000000	GPC input Type Selection Register
GPCSR	0x0BC	R/W	0x00000000	GPC Slew rate Control Register
-	0x0B4-0x0BC			Reserved

GPDDAT	0x0C0	R/W	0x00000000	GPD Data Register
GPDEN	0x0C4	R/W	0x00000000	GPD Output Enable Register
GPDSET	0x0C8	W	-	OR function on GPD Output Data
GPDCLR	0x0CC	W	-	BIC function on GPD Output Data
GPDXOR	0x0D0	W	-	XOR function on GPD Output Data
GPDCD0	0x0D4	R/W	0x55555555	Driver strength Control 0 on GPD Output Data
GPDCD1	0x0D8	R/W	0x00000000	Driver strength Control 1 on GPD Output Data
GPDPD0	0x0DC	R/W	0x55555555	Pull UP/DOWN Control Register 0 on GPD Output Data
GPDPD1	0x0E0	R/W	0x00000000	Pull UP/DOWN Control Register 1 on GPD Output Data
GPDFN0	0x0E4	R/W	0x00000000	Port Configuration on GPD Output Data
GPDFN1	0x0E8	R/W	0x00000000	Port Configuration on GPD Output Data
GPDFN2	0x0EC	R/W	0x00000000	Port Configuration on GPD Output Data
GPDFN3	0x0F0	R/W	0x00000000	Port Configuration on GPD Output Data
GPDIEN	0x0F4	R/W	0xFFFFFFFF	GPD Input Enable Register
GPDIS	0x0F8	R/W	0x00000000	GPD input Type Selection Register
GPDSR	0x0FC	R/W	0x00000000	GPD Slew rate Control Register
-	0x0F4-0x0FC			Reserved
GPEDAT	0x100	R/W	0x00000000	GPE Data Register
GPEEN	0x104	R/W	0x00000000	GPE Output Enable Register
GPESET	0x108	W	-	OR function on GPE Output Data
GPECLR	0x10C	W	-	BIC function on GPE Output Data
GPEXOR	0x110	W	-	XOR function on GPE Output Data
GPECD0	0x114	R/W	0x55555555	Driver strength Control 0 on GPE Output Data
GPECD1	0x118	R/W	0x00000000	Driver strength Control 1 on GPE Output Data
GPEPD0	0x11C	R/W	0x55555555	Pull UP/DOWN Control Register 0 on GPE Output Data
GPEPD1	0x120	R/W	0x00000000	Pull UP/DOWN Control Register 1 on GPE Output Data
GPEFN0	0x124	R/W	0x00000000	Port Configuration on GPE Output Data
GPEFN1	0x128	R/W	0x00000000	Port Configuration on GPE Output Data
GPEFN2	0x12C	R/W	0x00000000	Port Configuration on GPE Output Data
GPEFN3	0x130	R/W	0x00000000	Port Configuration on GPE Output Data
GPEIEN	0x134	R/W	0x00000000	GPE Input Enable Register
GPEIS	0x138	R/W	0x00000000	GPE input Type Selection Register
GPESR	0x13C	R/W	0x00000000	GPE Slew rate Control Register
-	0x134-0x13C			Reserved
GPFDAT	0x140	R/W	0x00000000	GPF Data Register
GPFEN	0x144	R/W	0x00000000	GPF Output Enable Register
GPFSET	0x148	W	-	OR function on GPF Output Data
GPFCLR	0x14C	W	-	BIC function on GPF Output Data
GPFXOR	0x150	W	-	XOR function on GPF Output Data
GPFCD0	0x154	R/W	0x55555555	Driver strength Control 0 on GPF Output Data
GPFCD1	0x158	R/W	0x00000000	Driver strength Control 1 on GPF Output Data
GPFPD0	0x15C	R/W	0x55555555	Pull UP/DOWN Control Register 0 on GPF Output Data
GPFPD1	0x160	R/W	0x00000000	Pull UP/DOWN Control Register 1 on GPF Output Data
GPFFN0	0x164	R/W	0x00000000	Port Configuration on GPF Output Data
GPFFN1	0x168	R/W	0x00000000	Port Configuration on GP Output Data
GPFFN2	0x16C	R/W	0x00000000	Port Configuration on GPF Output Data
GPFFN3	0x170	R/W	0x00000000	Port Configuration on GPF Output Data
GPFIEN	0x174	R/W	0x00000000	GPF Input Enable Register
GPFIS	0x178	R/W	0x00000000	GPF input Type Selection Register
GPFSR	0x17C	R/W	0x00000000	GPF Slew rate Control Register
-	0x174-0x17C			Reserved
GPGDAT	0x180	R/W	0x00000000	GPG Data Register
GPGEN	0x184	R/W	0x00000000	GPG Output Enable Register
GPGSET	0x188	W	-	OR function on GPG Output Data
GPGCLR	0x18C	W	-	BIC function on GPG Output Data
GPGXOR	0x190	W	-	XOR function on GPG Output Data

GPGCD0	0x194	R/W	0x55555555	Driver strength Control 0 on GPG Output Data
GPGCD1	0x198	R/W	0x00000000	Driver strength Control 1 on GPG Output Data
GPGPD0	0x19C	R/W	0x55555555	Pull UP/DOWN Control Register 0 on GPG Output Data
GPGPD1	0x1A0	R/W	0x00000000	Pull UP/DOWN Control Register 1 on GPG Output Data
GPGFN0	0x1A4	R/W	0x00000000	Port Configuration on GPG Output Data
GPGFN1	0x1A8	R/W	0x00000000	Port Configuration on GPG Output Data
GPGFN2	0x1AC	R/W	0x00000000	Port Configuration on GPG Output Data
GPGFN3	0x1B0	R/W	0x00000000	Port Configuration on GPG Output Data
GPGIEN	0x1B4	R/W	0x00000000	GPG Input Enable Register
GPGIS	0x1B8	R/W	0x00000000	GPG input Type Selection Register
GPGSR	0x1BC	R/W	0x00000000	GPG Slew rate Control Register
EINTSEL0	0x204	R/W	0x00000000	External Interrupt Select Register 0
EINTSEL1	0x208	R/W	0x00000000	External Interrupt Select Register 1
EINTSEL2	0x210	R/W	0x00000000	External Interrupt Select Register 2
MON	0x21C	R/W	0x00000000	System Monitor Enable Register
ECID0	0x214	R/W	0x00000000	CID CFG Register
ECID1	0x218	R	-	CID serial data Register
ECID2	0x21C	R	-	CID parallel data 0 Register [31:00]
ECID3	0x220	R	-	CID parallel data 1 Register [47:32]

GPIO Data Register (GPADAT, GPBDAT, GPCDAT, GPDDAT, GPEDAT, GPFDAT, GPGDATA)

0xB010A000, 0xB010A040, 0xB010A080, 0xB010A0C0, 0xB010A100, 0xB010A140, 0xB010A0180

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPDAT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPDAT															

Field	Name	RW	Reset	Description
31-0	GPDAT	R/W	-	GPIOD Data

If writing to this register, the written data would be stored in the output buffer. And the direction of corresponding GPIO is the output, the stored data would be out to pin. In case of reading this register, if the direction of the corresponding GPIO is out, the data read is output buffer, otherwise (input mode) the data read is the status of the corresponding pin

GPIO Direction Control Register (GPAEN, GPBEN, GPCEN, GPDEN, GPEEN, GPFEN, GPGEN)

0xB010A004, 0xB010A044, 0xB010A084, 0xB010A0C4, 0xB010A104, 0xB010A144, 0xB010A184

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPEN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPEN															

Field	Name	RW	Reset	Description
31-0	GPEN	R/W	0x0	GPIOD Output Enable Register

If the GPEN[i] is '1'(i = 0 ~ 31), the direction of the corresponding bit of GPIO port is changed to output mode, otherwise to input mode.

GPIO Set Register (GPASET, GPBSET, GPCSET, GPDSET, GPESET, GPFSET, GPGSET)

0xB010A008, 0xB010A048, 0xB010A088, 0xB010A0C8, 0xB010A108, 0xB010A148, 0xB010A188

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPSET															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPSET															

Field	Name	RW	Reset	Description
31-0	GPSET	W	-	Set Value

The equivalent function is GPADAT = GPADAT | GPSET, when GPADAT is the value of output buffer.

GPIO Clear Register (GPACLR, GPBCLR, GPCCLR, GPDCLR, GPECLR, GPFCLR, GPGCLR)
0xB010A00C, 0xB010A04C, 0xB010A08C, 0xB010A0CC, 0xB010A10C, 0xB010A14C, 0xB010A18C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPCLR															

Field	Name	RW	Reset	Description
31-0	GPCLR	W	-	Reset Value

The equivalent function is GPADAT = GPADAT & ~GPCLR, when GPADAT is the value of output buffer.

GPIO XOR Register (GPAXOR, GPBXOR, GPCXOR, GPDXOR GPEXOR, GPFXOR, GPGXOR)
0xB010A010, 0xB010A050, 0xB010A090, 0xB010A0D0, 0xB010A110, 0xB010A150, 0xB010A190

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPXOR															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPXOR															

Field	Name	RW	Reset	Description
31-0	GPXOR	W	-	XOR Value

The equivalent function is GPADAT = GPADAT ^ GPXOR, when GPADAT is the value of output buffer.

Driver Strength Control Register 0(GPACD0, GPBCD0, GPCCD0, GPDCD0, GPECD0, GPFCD0, GPGCD0)
0xB010A014, 0xB010A054, 0xB010A094, 0xB010A0D4, 0xB010A114, 0xB010A154,0xB010A194

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPCD15		GPCD14		GPCD13		GPCD12		GPCD11		GPCD10		GPCD9		GPCD8	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPCD7		GPCD6		GPCD5		GPCD4		GPCD3		GPCD2		GPCD1		GPCD0	

Field	Name	RW	Reset	Description
31-0	GPCDn	R/W	-	Drive strength of GPIO n'th port in corresponding GPIO group

The NVS2310 has the function of programmable port driver strength. The control value for driver strength of the corresponding port can become 0~3 and 3 is the strongest value.

Driver Strength Control Register 1(GPACD1, GPBCD1, GPCCD1, GPDCD1, GPECD1, GPFCD1, GPGCD1)
0xB010A018, 0xB010A058, 0xB010A098, 0xB010A0D8, 0xB010A118, 0xB010A158,0xB010A198

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPCD31		GPCD130		GPCD29		GPCD28		GPCD27		GPCD26		GPCD25		GPCD24	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPCD23		GPCD22		GPCD21		GPCD20		GPCD19		GPCD18		GPCD17		GPCD16	

Field	Name	RW	Reset	Description
31-0	GPCDn	R/W	-	Drive strength of GPIO n'th port in corresponding GPIO group

The NVS2310 has the function of programmable port driver strength. The control value for driver strength of the corresponding port can become 0~3 and 3 is the strongest value.

Each field of pull up/down control registers is the same with corresponding port name and consists of two bits. When PUx is "1", pull-up will be enabled,and when PDx is "1", pull-down will be enabled. It value should be not written with both pull-up and pull-down are enabled.

Pull UP/DOWN Control Register 0(GPAPD0, GPBPD0, GPCPD0, GPDPD0, GPEPD0, GPFPD0, GPGPD0)
0xB010A01C, 0xB010A05C, 0xB010A09C, 0xB010A0DC, 0xB010A11C, 0xB010A15C, 0xB010A19C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PD15	PE15	PD14	PE14	PD13	PE13	PD12	PE12	PD11	PE11	PD10	PE10	PD9	PE9	PD8	PE8
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD7	PE7	PD6	PE6	PD5	PE5	PD4	PE4	PD3	PE3	PD2	PE2	PD1	PE1	PD0	PE0

Field	Name	RW	Reset	Description
31-0	PUn, PDn	R/W	-	Pull-up/down of GPIO n'th port in corresponding GPIO group

Each field of pull up/down enable registers is the same with corresponding port name and consists of two bits. When PEn(n=0,…,15) is "1", pull-up/down of the corresponding port is controlled by PDn(n=0,…,15). If PDn is "1', pull-up of the port is enabled. When PEn(n=0,…,15) is "0", pull-up/down is not available.

Field	Name	RW	Reset	Description
n*2 (n=0,…,15)	PEn (n=0,…,15)	R/W	0x0	Pull UP/DOWN Function Enable 0 : pull-up/pull-down is disabled 1 : pull-up or pull-down is enabled, In this case pull up or pull down is controlled by PDn(n=0,…,15)

Field	Name	RW	Reset	Description
n*2 + 1 (n=0,…,15)	PDn (n=0,…,15)	R/W	0x0	Pull UP/DOWN Selection 0 : Pull down is enabled, if corresponding PEn is "1" 1 : Pull up is enabled if corresponding PEn is "1"

Pull UP/DOWN Control Register 1(GPAPD1, GPBPD1, GPCPD1, GPDPD1, GPEPD1, GPFPD1, GPGPD1)
0xB010A020, 0xB010A060, 0xB010A0A0, 0xB010A0E0, 0xB010A120, 0xB010A160, 0xB010A1A0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PD31	PE31	PD30	PE30	PD29	PE29	PD28	PE28	PD27	PE27	PD26	PE26	PD25	PE25	PD24	PE24
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD23	PE23	PD22	PE22	PD21	PE21	PD20	PE20	PD19	PE19	PD18	PE18	PD17	PE17	PD16	PE16

Field	Name	RW	Reset	Description
31-0	PUn, PDn	R/W	-	Pull-up/down of GPIO n'th port in corresponding GPIO group

Each field of pull up/down enable registers is the same with corresponding port name and consists of two bits. When PEn(n=16,…,31) is "1", pull-up/down of the corresponding port is controlled by PDn(n=16,…,31). If PDn is "1', pull-up of the port is enabled. When PEn(n=16,…,31) is "0", pull-up/down is not available.

Field	Name	RW	Reset	Description
n*2 (n=16,…,31)	PEn (n=16,…,31)	R/W	0x0	Pull UP/DOWN Function Enable 0 : pull-up/pull-down is disabled 1 : pull-up or pull-down is enabled, In this case pull up or pull down is controlled by PDn(n=16,…,31)

Field	Name	RW	Reset	Description
n*2 + 1 (n=16,…,31)	PDn (n=16,…,31)	R/W	0x0	Pull UP/DOWN Selection 0 : Pull down is enabled, if corresponding PEn is "1" 1 : Pull up is enabled if corresponding PEn is "1"

Port Configuration Register 0 (GPAFN0, GPBFN0, GPCFN0, GPDFN0, GPEFN0, GPFFN0, GPGFN0)
0xB010A024, 0xB010A064, 0xB010A0A4, 0xB010A0E4, 0xB010A124, 0xB010A164, 0xB010A1A4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPFN7				GPFN6				GPFN5				GPFN4			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPFN3				GPFN2				GPFN1				GPFN0			

Field	Name	RW	Reset	Description
31-0	GPFNn	R/W	0x0	Function of n'th port in corresponding GPIO group

Port Configuration Register 1 (GPAFN1, GPBFN1, GPCFN1, GPDFN1, GPEFN1, GPFFN1, GPGFN1)
0xB010A028, 0xB010A068, 0xB010A0A8, 0xB010A0E8, 0xB010A128, 0xB010A168, 0xB010A1A8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPFN15				GPFN14				GPFN13				GPFN12			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPFN11				GPFN10				GPFN9				GPFN8			

Field	Name	RW	Reset	Description
31-0	GPFNn	R/W	0x0	Function of n'th port in corresponding GPIO group

Port Configuration Register 2 (GPAFN2, GPBFN2, GPCFN2, GPDFN2, GPEFN2, GPFFN2, GPGFN2)
0xB010A02C, 0xB010A06C, 0xB010A0AC, 0xB010A0EC, 0xB010A12C, 0xB010A16C, 0xB010A1AC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPFN23				GPFN22				GPFN21				GPFN20			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPFN19				GPFN18				GPFN17				GPFN16			

Field	Name	RW	Reset	Description
31-0	GPFNn	R/W	0x0	Function of n'th port in corresponding GPIO group

Port Configuration Register 3 (GPAFN3, GPBFN3, GPCFN3, GPDFN3, GPEFN3, GPFFN3, GPGFN3)
0xB010A030, 0xB010A070, 0xB010A0B0, 0xB010A0F0, 0xB010A130, 0xB010A170, 0xB010A1B0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPFN31				GPFN30				GPFN29				GPFN28			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GPFN27				GPFN26				GPFN25				GPFN24			

Field	Name	RW	Reset	Description
31-0	GPFNn	R/W	0x0	Function of n'th port in corresponding GPIO group

Table 7.2 Port Configuration of GPIOA

Name	Function						
	0	1	2	3	4	5	6
GPIOA[0]	GPIO_A[0]	I2C0(SCL0)					
GPIOA[1]	GPIO_A[1]	I2C0(SDA0)					
GPIOA[2]	GPIO_A[2]	CLK_OUT0	I2C1(SCL0)				
GPIOA[3]	GPIO_A[3]	CLK_OUT1	I2C1(SDA0)				
GPIOA[4]	GPIO_A[4]	WDTRSTO	TCO0	I2C1(SCL1)			
GPIOA[5]	GPIO_A[5]	IRDI	TCO1	I2C1(SDA1)			
GPIOA[6]	GPIO_A[6]	HDMI_CEC	TCO2	I2C1(SCL0)			EDIXA[19]
GPIOA[7]	GPIO_A[7]	UTM1_DRVVBUS	TCO3	I2C1(SDA0)			EDIXA[20]
GPIOA[8]	GPIO_A[8]	I2C0(SCL1)	EDIXA[19]				
GPIOA[9]	GPIO_A[9]	I2C0(SDA1)	EDIXA[20]				
GPIOA[10]	GPIO_A[10]	I2C1(SCL1)	CDATA(1)				
GPIOA[11]	GPIO_A[11]	I2C1(SDA1)	CBCLK(1)				
GPIOA[12]	GPIO_A[12]	TCO0	CLRCK(1)				
GPIOA[13]	GPIO_A[13]	EXTCLKI	SPDIF_TX(0)				
GPIOA[14]	GPIO_A[14]	HDMI_HPD	TCO4				
GPIOA[15]	GPIO_A[15]	UTM0_DRVVBUS	TCO5				
GPIOA[16]	GPIO_A[16]	PWM_OUT[0]	SPDIF_TX(1)	I2C2(SCL0)			
GPIOA[17]	GPIO_A[17]	PWM_OUT[1]	CDATA(0)	I2C2(SDA0)			
GPIOA[18]	GPIO_A[18]	PWM_OUT[2]	CBCLK(0)	I2C2(SCL1)			
GPIOA[19]	GPIO_A[19]	PWM_OUT[3]	CLRCK(0)	I2C2(SDA1)			

FUNCTION	DESCRIPTION
GPIOA[19:0]	GPIO A Group Ports
I2C0(SCL0, SDA0)	I2C0 Channel 0
12C1(SCL0, SDA0)	I2C1 Channel 0
12C1(SCL1, SDA1)	I2C1 Channel 1
I2C2(SCL0, SDA0)	I2C2 Channel 0
I2C2(SCL1, SDA1)	I2C2 Channel 1
CLK_OUT0, CLK_OUT1	External Clock Outputs Generated by the CKC.
WDTRSTO	Watchdog Reset Output
IRDI	Remocon Receiver Port
HDMI_CECO, HDMI_CECI, HDMI_HPD	HDMI Option Ports
CBCLK(0), CLRCK(0), CDATA(0)	CD Interface Controller 0
CBCLK(1), CLRCK(1), CDATA(1)	CD Interface Controller 1
SPDIF_TX(0)	SPDIF Interface 0
SPDIF_TX(1)	SPDIF Interface 1
UTM_DRVVBUS	USB OTG Interface Port
TCO0 ~ TCO5	Timer Counter Output from Timer
EDIXA[20:19]	EDI Interface Ports
PWM_OUT[3:0]	PWM Interface Ports

Table 7.3 Port Configuration of GPIOB

Name	Function						
	0	1	2	3	4	5	6
GPIOB[0]	GPIO_B[0]	EDIXD0		MS_D0(5)			
GPIOB[1]	GPIO_B[1]	EDIXD1		MS_D1(5)			
GPIOB[2]	GPIO_B[2]	EDIXD2		MS_D2(5)			
GPIOB[3]	GPIO_B[3]	EDIXD3		MS_D3(5)			
GPIOB[4]	GPIO_B[4]	EDIXD4		MS_D4(5)			
GPIOB[5]	GPIO_B[5]	EDIXD5		MS_D5(5)			
GPIOB[6]	GPIO_B[6]	EDIXD6		MS_D6(5)			
GPIOB[7]	GPIO_B[7]	EDIXD7		MS_D7(5)			
GPIOB[8]	GPIO_B[8]	EDIXD8			SFRM(1)		
GPIOB[9]	GPIO_B[9]	EDIXD9			SCLK(1)		
GPIOB[10]	GPIO_B[10]	EDIXD10			SDI(1)		
GPIOB[11]	GPIO_B[11]	EDIXD11			SDO(1)		
GPIOB[12]	GPIO_B[12]	EDIXD12		MS_BUS(5)			
GPIOB[13]	GPIO_B[13]	EDIXD13		MS_CLK(5)			
GPIOB[14]	GPIO_B[14]	EDIXD14					
GPIOB[15]	GPIO_B[15]	EDIXD15					
GPIOB[16]	GPIO_B[16]	EDIWEN0					EDIRDY4
GPIOB[17]	GPIO_B[17]	EDIWEN1			SFRM(0)		
GPIOB[18]	GPIO_B[18]	EDIOEN0					EDIRDY5
GPIOB[19]	GPIO_B[19]	EDIOEN1			SCLK(0)		
GPIOB[20]	GPIO_B[20]	EDIXA[0]					
GPIOB[21]	GPIO_B[21]	EDIXA[1]		MS_D4(6)			
GPIOB[22]	GPIO_B[22]	EDIXA[2]		MS_D5(6)			
GPIOB[23]	GPIO_B[23]	EDICSN0		MS_D6(6)			EDIRDY2
GPIOB[24]	GPIO_B[24]	EDICSN1		MS_D7(6)			EDIRDY3
GPIOB[25]	GPIO_B[25]	EDICSN2		MS_D0(6)			
GPIOB[26]	GPIO_B[26]	EDICSN3	I2C2(SCL0)	MS_D1(6)			
GPIOB[27]	GPIO_B[27]	EDICSN4	I2C2(SDA0)	MS_D2(6)			EDIXA[23]
GPIOB[28]	GPIO_B[28]	EDIRDY0		MS_D3(6)			
GPIOB[29]	GPIO_B[29]	EDIRDY1		MS_BUS(6)			
GPIOB[30]	GPIO_B[30]	EDICSN5	I2C2(SCL1)	MS_CLK(6)	SDI(0)		EDIXA[22]
GPIOB[31]	GPIO_B[31]	EDICSN6	I2C2(SDA1)		SDO(0)		EDIXA[21]

FUNCTION	DESCRIPTION
GPIOB[31:0]	GPIO B Group Ports
MS_BUS(5), MS_CLK(5), MS_D0(5) ~ MS_D7(5)	Memory Stick Interface Port 5
I2C2(SCL0, SDA0)	I2C2 Channel 0
I2C2(SCL1, SDA1)	I2C2 Channel 1
MS_BUS(6), MS_CLK(6), MS_D0(6) ~ MS_D7(6)	Memory Stick Interface Port 6
SFRM(0), SCLK(0), SDI(0), SDO(0)	GPSB Interface Port 0
SFRM(1), SCLK(1), SDI(1), SDO(1)	GPSB Interface Port 1
EDIXA[2:0], EDIXA[23:21]	EDI Interface Ports
EDIXD[15:0]	EDI Interface Ports
EDICSN0~6, EDIRDY0~5	EDI Interface Ports

Table 7.4 Port Configuration of GPIOC

Name	Function						
	0	1	2	3	4	5	6
GPIOC[0]	GPIO_C[0]	LXD[0]	L0_LPD[0]	TSD5(3)		L1_LPD[0]	GPIOF[0]
GPIOC[1]	GPIO_C[1]	LXD[1]	L0_LPD[1]	TSD6(3)		L1_LPD[1]	GPIOF[1]
GPIOC[2]	GPIO_C[2]	LXD[2]	L0_LPD[2]	TSD7(3)		L1_LPD[2]	GPIOF[2]
GPIOC[3]	GPIO_C[3]	LXD[3]	L0_LPD[3]			L1_LPD[3]	GPIOF[3]
GPIOC[4]	GPIO_C[4]	LXD[4]	L0_LPD[4]			L1_LPD[4]	GPIOF[4]
GPIOC[5]	GPIO_C[5]	LXD[5]	L0_LPD[5]			L1_LPD[5]	GPIOF[5]
GPIOC[6]	GPIO_C[6]	LXD[6]	L0_LPD[6]			L1_LPD[6]	GPIOF[6]
GPIOC[7]	GPIO_C[7]	LXD[7]	L0_LPD[7]			L1_LPD[7]	GPIOF[7]
GPIOC[8]	GPIO_C[8]	LXD[8]	L0_LPD[8]			L1_LPD[8]	GPIOF[8]
GPIOC[9]	GPIO_C[9]	LXD[9]	L0_LPD[9]			L1_LPD[9]	GPIOF[9]
GPIOC[10]	GPIO_C[10]	LXD[10]	L0_LPD[10]	SDO(2)		L1_LPD[10]	GPIOF[10]
GPIOC[11]	GPIO_C[11]	LXD[11]	L0_LPD[11]	SDI(2)		L1_LPD[11]	GPIOF[11]
GPIOC[12]	GPIO_C[12]	LXD[12]	L0_LPD[12]	SCLK(2)		L1_LPD[12]	GPIOF[12]
GPIOC[13]	GPIO_C[13]	LXD[13]	L0_LPD[13]	SFRM(2)		L1_LPD[13]	GPIOF[13]
GPIOC[14]	GPIO_C[14]	LXD[14]	L0_LPD[14]		MS_D7(0)	L1_LPD[14]	GPIOF[14]
GPIOC[15]	GPIO_C[15]	LXD[15]	L0_LPD[15]		MS_D6(0)	L1_LPD[15]	GPIOF[15]
GPIOC[16]	GPIO_C[16]	LXD[16]	L0_LPD[16]		MS_D5(0)	L1_LPD[16]	GPIOF[16]
GPIOC[17]	GPIO_C[17]	LXD[17]	L0_LPD[17]		MS_D4(0)	L1_LPD[17]	GPIOF[17]
GPIOC[18]	GPIO_C[18]	LXD[18]	L0_LPD[18]		MS_D3(0)	L1_LPD[18]	
GPIOC[19]	GPIO_C[19]	LXD[19]	L0_LPD[19]		MS_D2(0)	L1_LPD[19]	
GPIOC[20]	GPIO_C[20]	LXD[20]	L0_LPD[20]		MS_D1(0)	L1_LPD[20]	
GPIOC[21]	GPIO_C[21]	LXD[21]	L0_LPD[21]		MS_D0(0)	L1_LPD[21]	
GPIOC[22]	GPIO_C[22]	LXD[22]	L0_LPD[22]		MS_CLK(0)	L1_LPD[22]	
GPIOC[23]	GPIO_C[23]	LXD[23]	L0_LPD[23]		MS_BUS(0)	L1_LPD[23]	
GPIOC[24]	GPIO_C[24]	LWEN	L0_LDE	TSD4(3)		L1_LDE	GPIOF[19]
GPIOC[25]	GPIO_C[25]	LOEN	L0_LCK	TSD3(3)		L1_LCK	GPIOF[18]
GPIOC[26]	GPIO_C[26]	LXA[0]	L0_LHS	TSD2(3)		L1_LHS	GPIOF[22]
GPIOC[27]	GPIO_C[27]	LCSN0	L0_LVS	TSD1(3)		L1_LVS	GPIOF[20]
GPIOC[28]	GPIO_C[28]	LCSN1	SDO(10)	TSVALID(3)			GPIOF[21]
GPIOC[29]	GPIO_C[29]		SDI(10)	TSCLK(3)			
GPIOC[30]	GPIO_C[30]	EXTLVS1(0)	SCLK(10)	TSD0(3)			
GPIOC[31]	GPIO_C[31]	EXTLVS0(0)	SFRM(10)	TSSYNC(3)			

FUNCTION	DESCRIPTION
GPIOC[31:0]	GPIO C Group Ports
GPIOF[22:0]	GPIO F Signals – In this case, the GPIO C Group is Output Mode and Functionally the same as the bypass by BPEN.
LCD0_LPD[23:0], LCD0_LDE, LCD0_LCK, LCD0_LHS, LCD0_LVS	LCD RGB Interface Ports from LCD Controller 0
LCD1_LPD[23:0], LCD1_LDE, LCD1_LCK, LCD1_LHS, LCD1_LVS	LCD RGB Interface Ports from LCD Controller 1
LXD[17:0], LWEN, LOEN, LXA[0], LCSN0, LCSN1	LCD CPU Interface Ports Which Can Be Shared By LCD Controller 0, 1 and CPU.
EXTLVS0(0)	External VSYNC Input Port 0 for LCD Controller 0
EXTLVS1(0)	External VSYNC Input Port 1 for LCD Controller 1
SFRM(10), SCLK(10), SDI(10), SDO(10)	GPSB Interface Port 10
SFRM(3), SCLK(3), SDI(3), SDO(3)	GPSB Interface Port 3
SFRM(2), SCLK(2), SDI(2), SDO(2)	GPSB Interface Port 2
MS_BUS(0), MS_CLK(0), MS_D0(0) ~ MS_D7(0)	Memory Stick Interface Port 0
TSSYNC(3), TSCLK(3), TSVALID(3), TSD0(3) ~ TSD7(3)	TS Parallel Interface Port 3

Table 7.5 Port Configuration of GPIOD

Name	Function						
	0	1	2	3	4	5	6
GPIOD[0]	GPIO_D[0]	DAI0_BCLK					
GPIOD[1]	GPIO_D[1]	DAI0_LRCLK					
GPIOD[2]	GPIO_D[2]	DAI0_MCLK					
GPIOD[3]	GPIO_D[3]	DAI0_SDO[0]					
GPIOD[4]	GPIO_D[4]	DAI0_SDI[0]					
GPIOD[5]	GPIO_D[5]	DAI0_SDO[1]	SFRM(11)				
GPIOD[6]	GPIO_D[6]	DAI0_SDI[1]	SCLK(11)				
GPIOD[7]	GPIO_D[7]	DAI0_SDO[2]	SDI(11)				
GPIOD[8]	GPIO_D[8]	DAI0_SDI[2]	SDO(11)				
GPIOD[9]	GPIO_D[9]	DAI0_SDO[3]	SFRM(6)	TSD7(2)			
GPIOD[10]	GPIO_D[10]	DAI0_SDI[3]	SCLK(6)	TSD6(2)			
GPIOD[11]	GPIO_D[11]	SPDIF_TX(0)	SDI(6)				
GPIOD[12]	GPIO_D[12]	SPDIF_RX	SDO(6)	TSSYNC(2)			
GPIOD[13]	GPIO_D[13]	UTXD(4)	I2C1(SCL1)	TSD5(2)			
GPIOD[14]	GPIO_D[14]	URXD(4)	I2C1(SDA1)	TSD4(2)			
GPIOD[15]	GPIO_D[15]	UCTS(4)	SFRM(12)	TSVALID(2)			
GPIOD[16]	GPIO_D[16]	URTS(4)	SCLK(12)	TSCLK(2)			
GPIOD[17]	GPIO_D[17]	UTXD(5)	I2C2(SCL0)	TSD3(2)			
GPIOD[18]	GPIO_D[18]	URXD(5)	I2C2(SDA0)	TSD2(2)	I2C0(SCL1)		
GPIOD[19]	GPIO_D[19]	UCTS(5)	SDI(12)	TSD1(2)	I2C0(SDA1)		
GPIOD[20]	GPIO_D[20]	URTS(5)	SDO(12)	TSD0(2)	I2C0(SCL0)		
GPIOD[21]	GPIO_D[21]	DAI1_BCLK			I2C0(SDA0)		
GPIOD[22]	GPIO_D[22]	DAI1_LRCLK			I2C1(SCL0)		
GPIOD[23]	GPIO_D[23]	DAI1_MCLK			I2C1(SDA0)		
GPIOD[24]		DAI1_SDO[0]					
GPIOD[25]		DAI1_SDI[0]					

FUNCTION	DESCRIPTION
GPIOD[25:0]	GPIO D Group Ports
BCLK(1), LRCK(1), MCLK(1), DAO0(1) ~ DAO3(1), DAI0(1) ~ DAI3(1)	I2S Controller 1 – Multiple channel with DMA
BCLK(0), LRCK(0), MCLK(0), DAO0(0), DAI0(0)	I2S Controller 0 – Single channel without DMA
SFRM(11), SCLK(11), SDI(11), SDO(11)	GPSB Interface Port 11
SFRM(12), SCLK(12), SDI(12), SDO(12)	GPSB Interface Port 12
SFRM(6), SCLK(6), SDI(6), SDO(6)	GPSB Interface Port 6
SPD_TX(1), SPD_RX(1)	SPDIF Transmit/Receiver Controller 1
SPD_TX(0)	SPDIF Transmitter Controller 0
UTXD(4), URXD(4), UCTS(4), URTS(4)	UART Interface Port 4
UTXD(5), URXD(5), UCTS(5), URTS(5)	UART Interface Port 5
TSSYNC(2), TSVALID(2), TSCLK(2), TSD0(2) ~ TSD7(2)	TS Parallel Interface Port 2
I2C1(SCL1,SDA1)	I2C1 Channel1
I2C2(SCL0, SDA0)	I2C2 Channel0
I2C0(SCL1,SDA1)	I2C0 Channel1
I2C0(SCL0,SDA0)	I2C0 Channel0
I2C1(SCL0,SDA0)	I2C1 Channel0

Table 7.6 Port Configuration of GPIOE

Name	Function						
	0	1	2	3	4	5	6
GPIOE[0]	GPIO_E[0]	UTXD(0)				SD3_XD[0]	
GPIOE[1]	GPIO_E[1]	URXD(0)				SD3_XD[1]	
GPIOE[2]	GPIO_E[2]	UCTS(0)	SFRM(5)			SD3_XD[2]	
GPIOE[3]	GPIO_E[3]	URTS(0)	SCLK(5)			SD3_XD[3]	
GPIOE[4]	GPIO_E[4]	UTXD(1)				SD3_XD[4]	
GPIOE[5]	GPIO_E[5]	URXD(1)				SD3_XD[5]	
GPIOE[6]	GPIO_E[6]	UCTS(1)	SDI(5)		MS_CLK(4)	SD3_CLK	
GPIOE[7]	GPIO_E[7]	URTS(1)	SDO(5)		MS_BUS(4)	SD3_CMD	
GPIOE[8]	GPIO_E[8]	UTXD(2)	SFRM(4)	CPD[8]	MS_D0(4)	SD3_XD[6]	
GPIOE[9]	GPIO_E[9]	URXD(2)	SCLK(4)	CPD[9]	MS_D1(4)	SD3_XD[7]	
GPIOE[10]	GPIO_E[10]	UTXD(3)	SDI(4)	CPD[10]	MS_D2(4)		
GPIOE[11]	GPIO_E[11]	URXD(3)	SDO(4)	CPD[11]	MS_D3(4)		
GPIOE[12]	GPIO_E[12]	CPD[0]		TSD0(1)	MS_D0(2)		
GPIOE[13]	GPIO_E[13]	CPD[1]		TSD1(1)	MS_D1(2)		
GPIOE[14]	GPIO_E[14]	CPD[2]		TSD2(1)	MS_D2(2)		
GPIOE[15]	GPIO_E[15]	CPD[3]		TSD3(1)	MS_D3(2)		
GPIOE[16]	GPIO_E[16]	CPD[4]		TSD4(1)	MS_D4(2)		
GPIOE[17]	GPIO_E[17]	CPD[5]		TSD5(1)	MS_D5(2)		
GPIOE[18]	GPIO_E[18]	CPD[6]		TSD6(1)	MS_D6(2)		
GPIOE[19]	GPIO_E[19]	CPD[7]		TSD7(1)	MS_D7(2)		
GPIOE[20]	GPIO_E[20]	CCKI		TSCLK(1)	MS_CLK(2)		
GPIOE[21]	GPIO_E[21]	CVS		TSSYNC(1)	MS_BUS(2)		
GPIOE[22]	GPIO_E[22]	CHS		TSVALID(1)			
GPIOE[23]	GPIO_E[23]	CCKO	CFIELD				
GPIOE[24]	GPIO_E[24]	SENSOR_PWDN					
GPIOE[25]	GPIO_E[25]	FL_TRIG					
GPIOE[26]	GPIO_E[26]	FLASH_TRIG		MS_BUS(7)			
GPIOE[27]	GPIO_E[27]	PRELIGHT_TRIG		MS_CLK(7)			
GPIOE[28]	GPIO_E[28]	SHUTTER_TRIG	CFIELD	MS_D0(7)	TS_AIN[0]		
GPIOE[29]	GPIO_E[29]	SHUTTER_OPEN		MS_D1(7)	TS_AIN[1]		
GPIOE[30]	GPIO_E[30]	I2C0(SCL1)		MS_D2(7)	TS_AIN[2]		
GPIOE[31]	GPIO_E[31]	I2C0(SDA1)		MS_D3(7)	TS_AIN[3]		

FUNCTION	DESCRIPTION
GPIOE[31:0]	GPIO E Group Ports
TS_AIN[3:0]	ADC Input Ports
UTXD(0), URXD(0), UCTS(0), URTS(0)	UART Interface Port 0
UTXD(1), URXD(1), UCTS(1), URTS(1)	UART Interface Port 1
UTXD(2), URXD(2), UCTS(2), URTS(2)	UART Interface Port 2
UTXD(3), URXD(3), UCTS(3), URTS(3)	UART Interface Port 3
CPD[7:0], CCKI, CVS, CHS, CCKO, CFIELD	Camera or Video Input Interface Ports
SFRM(5), SCLK(5), SDI(5), SDO(5)	GPSB Interface Port 5
SFRM(4), SCLK(4), SDI(4), SDO(4)	GPSB Interface Port 4
SD_CMD(3), SD_CLK(3), SD_D0(3) ~ SD_D7(3)	SD/MMC Interface Port 3
MS_BUS(4), MS_CLK(4), MS_D0(4) ~ MS_D3(4)	Memory Stick Interface Port 4
MS_BUS(2), MS_CLK(2), MS_D0(2) ~ MS_D7(2)	Memory Stick Interface Port 2
MS_BUS(7), MS_CLK(7), MS_D0(7) ~ MS_D3(7)	Memory Stick Interface Port 7
TSSYNC(1), TSVALID(1), TSCLK(1), TSD0(1) ~ TSD7(1)	TS Parallel Interface Port 1
SENSOR_PWDN, FL_TRIG FLASH_TRIG,PREIGHT_TRIG SUTTER_TRIG,SUTTER_OPEN	Camera Interface Ports

Table 7.7 Port Configuration of GPIOF

Name	Function							
	0	1	2	3	4	5	6	7
GPIOF[0]	GPIO_F[0]	HPXD[0]		GMAC_RXD[4]	TSD0(0)	L1_LPD[0]	EDIXA[3]	
GPIOF[1]	GPIO_F[1]	HPXD[1]		GMAC_RXD[5]	TSD1(0)	L1_LPD[1]	EDIXA[4]	
GPIOF[2]	GPIO_F[2]	HPXD[2]		GMAC_RXD[6]	TSD2(0)	L1_LPD[2]	EDIXA[5]	
GPIOF[3]	GPIO_F[3]	HPXD[3]		GMAC_RXD[7]	TSD3(0)	L1_LPD[3]	EDIXA[6]	
GPIOF[4]	GPIO_F[4]	HPXD[4]		GMAC_TXD[4]	TSD4(0)	L1_LPD[4]	EDIXA[7]	
GPIOF[5]	GPIO_F[5]	HPXD[5]		GMAC_TXD[5]	TSD5(0)	L1_LPD[5]	EDIXA[8]	
GPIOF[6]	GPIO_F[6]	HPXD[6]		GMAC_TXD[6]	TSD6(0)	L1_LPD[6]	EDIXA[9]	
GPIOF[7]	GPIO_F[7]	HPXD[7]		GMAC_TXD[7]	TSD7(0)	L1_LPD[7]	EDIXA[10]	
GPIOF[8]	GPIO_F[8]	HPXD[8]		GMAC_CLK_RX	TSVALID(0)	L1_LPD[8]	EDIXA[11]	
GPIOF[9]	GPIO_F[9]	HPXD[9]		GMAC_RXDV	TSCLK(0)	L1_LPD[9]	EDIXA[12]	
GPIOF[10]	GPIO_F[10]	HPXD[10]	SDO(7)	GMAC_RXER	TSSYNC(0)	L1_LPD[10]	EDIXA[13]	
GPIOF[11]	GPIO_F[11]	HPXD[11]	SDI(7)	GMAC_RXD[0]		L1_LPD[11]	EDIXA[14]	
GPIOF[12]	GPIO_F[12]	HPXD[12]	SCLK(7)	GMAC_RXD[1]		L1_LPD[12]	EDIXA[15]	
GPIOF[13]	GPIO_F[13]	HPXD[13]	SFRM(7)	GMAC_RXD[2]		L1_LPD[13]	EDIXA[16]	
GPIOF[14]	GPIO_F[14]	HPXD[14]	SDO(8)	GMAC_RXD[3]		L1_LPD[14]	EDIXA[17]	EDIXD16
GPIOF[15]	GPIO_F[15]	HPXD[15]	SDI(8)	GMAC_CLK_TX		L1_LPD[15]	EDIXA[18]	EDIXD17
GPIOF[16]	GPIO_F[16]	HPXD[16]	SCLK(8)	GMAC_TXEN		L1_LPD[16]	EDIXA[19]	EDIXD18
GPIOF[17]	GPIO_F[17]	HPXD[17]	SFRM(8)	GMAC_TXER		L1_LPD[17]	EDIXA[20]	EDIXD19
GPIOF[18]	GPIO_F[18]	HPRDN		GMAC_TXD[0]	MS_D3(1)	L1_LPD[18]	EDIXA[21]	EDIXD20
GPIOF[19]	GPIO_F[19]	HPWRN		GMAC_TXD[1]	MS_D2(1)	L1_LPD[19]	EDIXA[22]	EDIXD21
GPIOF[20]	GPIO_F[20]	HPCSN0		GMAC_TXD[2]	MS_D1(1)	L1_LPD[20]		EDIXD22
GPIOF[21]	GPIO_F[21]	HPCSN1		GMAC_TXD[3]	MS_D0(1)	L1_LPD[21]		EDIXD23
GPIOF[22]	GPIO_F[22]	HPXA0		GMAC_COL	MS_BUS(1)	L1_LPD[22]		EDIXD24
GPIOF[23]	GPIO_F[23]	HPINT0		GMAC_CRS	MS_CLK(1)	L1_LPD[23]		EDIXD25
GPIOF[24]	GPIO_F[24]	HPINT1	SDO(9)	GMAC_MDC		L1_LDE		EDIXD26
GPIOF[25]	GPIO_F[25]	HPXA1	SDI(9)	GMAC_MDIO		L1_LCK		EDIXD27
GPIOF[26]	GPIO_F[26]	EXTLVS1(1)	SCLK(9)	GMAC_CLK_GTX		L1_LHS		EDIXD28
GPIOF[27]	GPIO_F[27]	EXTLVS0(1)	SFRM(9)			L1_LVS		EDIXD29
GPIOF[28]	GPIO_F[28]	HPCSN2						EDIXD30
GPIOF[29]	GPIO_F[29]							EDIXD31

FUNCTION	DESCRIPTION
GPIOF[29:0]	GPIO F Group Ports
HPXD[17:0], HPRDN, HPWRN, HPCSN0, HPCSN1, HPINT0, HPINT1	External Host Interface Ports
EXTLVS1(1)	External VSYNC Port 1 to LCD Controller 1
EXTLVS0(1)	External VSYNC Port 1 to LCD Controller 0
SFRM(7), SCLK(7), SDI(7), SDO(7)	GPSB Interface Port 7
SFRM(8), SCLK(8), SDI(8), SDO(8)	GPSB Interface Port 8
SFRM(9), SCLK(9), SDI(9), SDO(9)	GPSB Interface Port 9
HDDXD0 ~ HDDXD15, HDDXA0 ~ HDDXA2, HDDAK, HDDRQ, HDDRDY, HDDCSN0, HDDCSN1, HDDIOW, HDDIOR	UDMA Interface Ports
CAN_TX, CAN_RX	CAN Interface Ports
MS_BUS(3), MS_CLK(3), MS_D0(3) ~ MS_D7(3)	Memory Stick Interface Port 3
MS_BUS(1), MS_CLK(1), MS_D0(1) ~ MS_D3(1)	Memory Stick Interface Port 1
TSSYNC(0), TSVALID(0), TSCLK(0), TSD0(0) ~ TSD7(0)	TS Parallel Interface Port 0
EDIXA[22:3], EDIXD[31:16]	EDI Interface Ports

Table 7.8 Port Configuration of GPIOG

Name	Function						
	0	1	2	3	4	5	6
GPIOG[0]	GPIO_G[0]	SD0_XD[0]		MS_D0(8)			
GPIOG[1]	GPIO_G[1]	SD0_XD[1]		MS_D1(8)			
GPIOG[2]	GPIO_G[2]	SD0_XD[2]		MS_D2(8)			
GPIOG[3]	GPIO_G[3]	SD0_XD[3]		MS_D3(8)			
GPIOG[4]	GPIO_G[4]	SD0_XD[4]	UCTS(2)	MS_D4(8)	I2C2(SCL0)		
GPIOG[5]	GPIO_G[5]	SD0_XD[5]	URTS(2)	MS_D5(8)	I2C2(SDA0)		
GPIOG[6]	GPIO_G[6]	SD0_XD[6]	URXD(2)	MS_D6(8)	I2C2(SCL1)		
GPIOG[7]	GPIO_G[7]	SD0_XD[7]	UTXD(2)	MS_D7(8)	I2C2(SDA1)		
GPIOG[8]	GPIO_G[8]	SD0_CLK		MS_CLK(8)			
GPIOG[9]	GPIO_G[9]	SD0_CMD		MS_BUS(8)			
GPIOG[10]	GPIO_G[10]	SD1_XD[0]					
GPIOG[11]	GPIO_G[11]	SD1_XD[1]	URXD(3)		I2C1(SCL0)		
GPIOG[12]	GPIO_G[12]	SD1_XD[2]	UTXD(3)		I2C1(SDA0)		
GPIOG[13]	GPIO_G[13]	SD1_XD[3]					
GPIOG[14]	GPIO_G[14]	SD1_CLK					
GPIOG[15]	GPIO_G[15]	SD1_CMD					
GPIOG[16]	GPIO_G[16]	SD2_XD[0]		MS_D0(9)			
GPIOG[17]	GPIO_G[17]	SD2_XD[1]	URXD(4)	MS_D1(9)	I2C1(SCL1)		
GPIOG[18]	GPIO_G[18]	SD2_XD[2]	UTXD(4)	MS_D2(9)	I2C1(SDA1)		
GPIOG[19]	GPIO_G[19]	SD2_XD[3]		MS_D3(9)			
GPIOG[20]	GPIO_G[20]	SD2_CLK		MS_CLK(9)			
GPIOG[21]	GPIO_G[21]	SD2_CMD		MS_BUS(9)			
GPIOG[22]	GPIO_G[22]	TS_AIN[4]	UCTS(0)	SDO(3)			
GPIOG[23]	GPIO_G[23]	TS_AIN[5]	URTS(0)	SDI(3)			
GPIOG[24]	GPIO_G[24]	TS_AIN[10]	URXD(0)	SCLK(3)			
GPIOG[25]	GPIO_G[25]	TS_AIN[11]	UTXD(0)	SFRM(3)			
GPIOG[26]	GPIO_G[26]	TS_AIN[12]	UCTS(1)	SD0_BUS_POW	I2C2(SCL0)		
GPIOG[27]	GPIO_G[27]	TS_AIN[13]	URTS(1)	SD1_BUS_POW	I2C2(SDA0)		
GPIOG[28]	GPIO_G[28]	TS_AIN[14]	URXD(1)	SD2_BUS_POW	I2C2(SCL1)		
GPIOG[29]	GPIO_G[29]	TS_AIN[15]	UTXD(1)	SD3_BUS_POW	I2C2(SDA1)		

FUNCTION	DESCRIPTION
GPIOG[29:0]	GPIO G Group Ports
SD0_XD[7:0], SD0_CLK, SD0_CMD	SD/MMC 0 Ports
SD1_XD[3:0], SD1_CLK, SD1_CMD	SD/MMC 1 Ports
SD2_XD[3:0], SD2_CLK, SD2_CMD	SD/MMC 2 Ports
UCTS(2),URTS(2),URXD(2),UTXD(2)	UART2 Ports
URXD(3),UTXD(3)	UART3 Ports
URXD(4),UTXD(4)	UART4 Ports
UCTS(0),URTS(0),URXD(0),UTXD(0)	URST0 Ports
UCTS(1),URTS(1),URXD(1),UTXD(1)	URST1 Ports
MS_D7(8)~MS_D0(8),MS_CLK(8), MS_BUS(8)	Memory Stick Interface Port 8
SD0_BUS_POW, SD1_BUS_POW SD2_BUS_POW, SD3_BUS_POW	SD/MMC Ports
I2C2(SCL0, SDA0)	I2C2 Channel 0
I2C1(SCL0,SDA0)	I2C1 Channel 0
I2C1(SCL1,SDA1)	I2C1 Channel 1
I2C2(SCL0,SDA0)	I2C2 Channel 0
I2C2(SCL1,SDA1)	I2C2 Channel 1

Input Enable Register (GPAIEN, GPBIEN, GPCIEN, GPDIEN, GPEIEN, GPFIEN, GPGIEN)

0xB010A034, 0xB010A074, 0xB010A0B4, 0xB010A0E4, 0xB010A134, 0xB010A174, 0xB010A1B4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IEN31	IEN30	IEN29	IEN28	IEN27	IEN26	IEN25	IEN24	IEN23	IEN22	IEN21	IEN20	IEN19	IEN18	IEN17	IEN16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IEN15	IEN14	IEN13	IEN12	IEN11	IEN10	IEN9	IEN8	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IEN0

Field	Name	RW	Reset	Description
31-0	GPIEN	R/W	0x00000000	IEN Value

Each field of input enble control registers is the same with corresponding port name and consists of 1-bit. When IENn(n=0,…,31) is ‘1’, GPIO Input signal is available and which can be read by GPIO Data Register..

Input Type Selection Register (GPAIS, GPBIS, GPCIS, GPDIS, GPEIS, GPFIS, GPGIS)

0xB010A038, 0xB010A078, 0xB010A0B8, 0xB010A0E8, 0xB010A138, 0xB010A178, 0xB010A1B8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IS31	IS30	IS29	IS28	IS27	IS26	IS25	IS24	IS23	IS22	IS21	IS20	IS19	IS18	IS17	IS16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IS15	IS14	IS13	IS12	IS11	IS10	IS9	IS8	IS7	IS6	IS5	IS4	IS3	IS2	IS1	IS0

Field	Name	RW	Reset	Description
31-0	GPIS	R/W	0x00000000	Input Type Selection Value

Each field of input type control registers is the same with corresponding port name and consists of 1-bit. When ISn(n=0,…,31) is ‘1’, the corresponding port regard as Shmitt input typeand otherwise as CMOS input type.

Slew Rate Control Register (GPASR, GPBISR, GPCSR, GPDSR, GPESR, GPFSR, GPGSR)

0xB010A03C, 0xB010A07C, 0xB010A0BC, 0xB010A0EC, 0xB010A13C, 0xB010A17C, 0xB010A1BC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SR31	SR30	SR29	SR28	SR27	SR26	SR25	SR24	SR23	SR22	SR21	SR20	SR19	SR18	SR17	SR16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SR15	SR14	SR13	SR12	SR11	SR10	SR9	SR8	SR7	SR6	SR5	SR4	SR3	SR2	SR1	SR0

Field	Name	RW	Reset	Description
31-0	GPSR	R/W	0x00000000	Slew Rate Value

Each field of input type control registers is the same with corresponding port name and consists of 1-bit. When SRn(n=0,…,31) is ‘1’, the srew rate of the corresponding port is slow and otherwise fast.

8 GPSB (General Purpose Serial Bus)

8.1 Functional Description

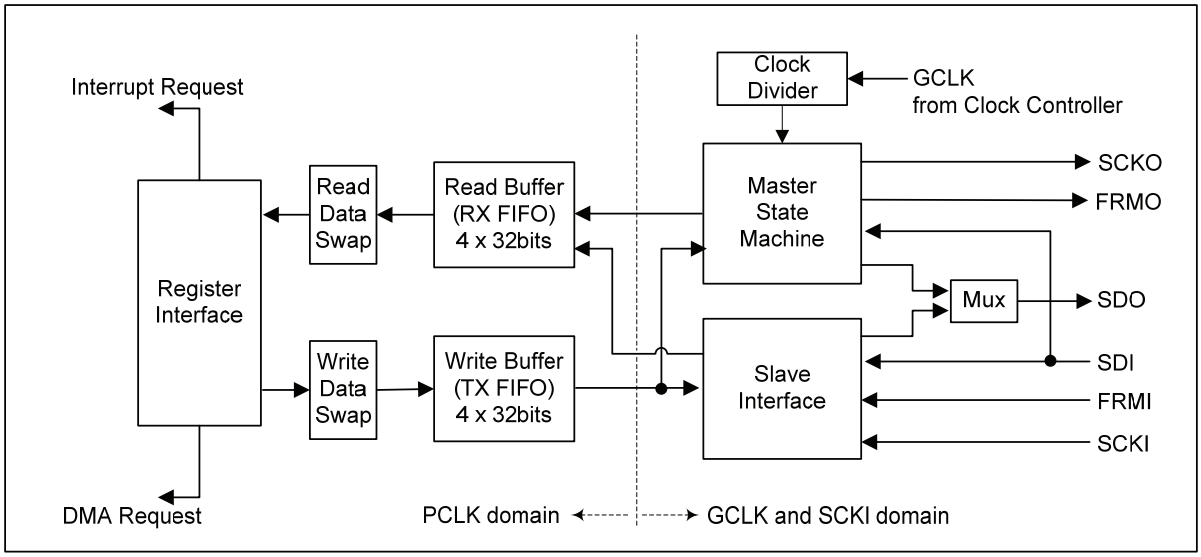


Figure 8.1 GPSB Interface Block Diagram

The Figure 8.1 shows the block diagram of GPSB interface circuit. The following figure shows the overall hardware block diagram for GPSB.

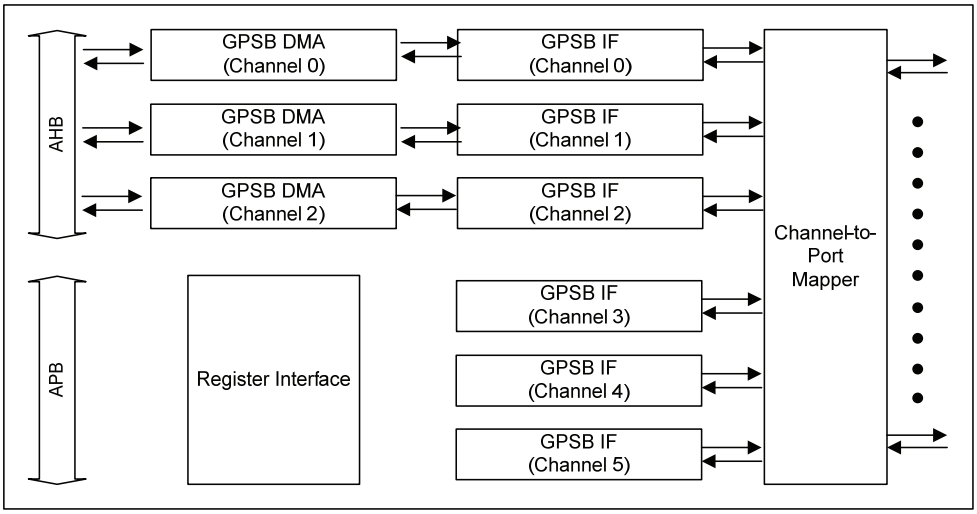


Figure 8.2 Overall GPSB Block Diagram

The NVS2310 provides 6 channel General Purpose Serial Bus Controller, which can be configured as a SPI compatible master/slave and MPEG-2 TS serial interface. The channel 0,1 and 2 have the DMA function which can transfer the bulk data. The channel 3,4 and 5 can interface the general purpose DMA for fast and bulk data transfer. The GPSB DMA can be used for MPEG-2 TS serial interface with PID matching function.

The “Channel-to-Port Mapper” is for mapping the hardware channel and external ports. Each channel can be mapped to one of the ports.

8.2 Feature

The feature of GPSB in NVS2310 supports Motorola SPI, TI SSP compatible timing and changes the programmable master / slave configuration, timing parameters, data bit width, shift direction, polarity, data swap and so on.
The frequency of the SCKO pin is determined by the Clock Divider block which uses GCLK as clock source. The GCLK is generated by the Clock Controller Module.

8.3 Register Description

Table 8.1 GPSB Register Map (Base Address = 0xB0107000)

Ch	Name	Addr. Offset	Type	Reset	Description
Channel 0	PORT	0x000	R/W	0x0000	Data port
	STAT	0x004	R/W	0x0000	Status register
	INTEN	0x008	R/W	0x0000	Interrupt enable
	MODE	0x00C	R/W	0x0004	Mode register
	CTRL	0x010	R/W	0x0000	Control register
	EVTCTRL	0x014	R/W	0x0000	Counter & Ext. Event Control
	CCV	0x018	R	0x0000	Counter Current Value
	TXBASE	0x020	R/W	0x0000	TX base address register
	RXBASE	0x024	R/W	0x0000	RX base address register
	PACKET	0x028	R/W	0x0000	Packet register
	DMA_CTRL	0x02C	R/W	0x0000	DMA control register
	DMA_STAT	0x030	R/W	0x0000	DMA status register
	DMA_ICR	0x034	R/W	0x0000	DMA interrupt control register
Channel 1	PORT	0x100	R/W	0x0000	Data port
	STAT	0x104	R/W	0x0000	Status register
	INTEN	0x108	R/W	0x0000	Interrupt enable
	MODE	0x10C	R/W	0x0004	Mode register
	CTRL	0x110	R/W	0x0000	Control register
	EVTCTRL	0x114	R/W	0x0000	Counter & Ext. Event Control
	CCV	0x118	R	0x0000	Counter Current Value
	TXBASE	0x120	R/W	0x0000	TX base address register
	RXBASE	0x124	R/W	0x0000	RX base address register
	PACKET	0x128	R/W	0x0000	Packet register
	DMA_CTRL	0x12C	R/W	0x0000	DMA control register
	DMA_STAT	0x130	R/W	0x0000	DMA status register
	DMA_ICR	0x134	R/W	0x0000	DMA interrupt control register
Channel 2	PORT	0x200	R/W	0x0000	Data port
	STAT	0x204	R/W	0x0000	Status register
	INTEN	0x208	R/W	0x0000	Interrupt enable
	MODE	0x20C	R/W	0x0004	Mode register
	CTRL	0x210	R/W	0x0000	Control register
	EVTCTRL	0x214	R/W	0x0000	Counter & Ext. Event Control
	CCV	0x218	R	0x0000	Counter Current Value
	TXBASE	0x220	R/W	0x0000	TX base address register
	RXBASE	0x224	R/W	0x0000	RX base address register
	PACKET	0x228	R/W	0x0000	Packet register
	DMA_CTRL	0x22C	R/W	0x0000	DMA control register
	DMA_STAT	0x230	R/W	0x0000	DMA status register
	DMA_ICR	0x234	R/W	0x0000	DMA interrupt control register
Channel 3	PORT	0x300	R/W	0x0000	Data port
	STAT	0x304	R/W	0x0000	Status register
	INTEN	0x308	R/W	0x0000	Interrupt enable
	MODE	0x30C	R/W	0x0004	Mode register
	CTRL	0x310	R/W	0x0000	Control register
	EVTCTRL	0x314	R/W	0x0000	Counter & Ext. Event Control
	CCV	0x318	R	0x0000	Counter Current Value
Channel 4	PORT	0x400	R/W	0x0000	Data port
	STAT	0x404	R/W	0x0000	Status register
	INTEN	0x408	R/W	0x0000	Interrupt enable
	MODE	0x40C	R/W	0x0004	Mode register
	CTRL	0x410	R/W	0x0000	Control register
	EVTCTRL	0x414	R/W	0x0000	Counter & Ext. Event Control
	CCV	0x418	R	0x0000	Counter Current Value
Channel 5	PORT	0x500	R/W	0x0000	Data port
	STAT	0x504	R/W	0x0000	Status register
	INTEN	0x508	R/W	0x0000	Interrupt enable
	MODE	0x50C	R/W	0x0004	Mode register
	CTRL	0x510	R/W	0x0000	Control register
	EVTCTRL	0x514	R/W	0x0000	Counter & Ext. Event Control

	CCV	0x518	R	0x0000	Counter Current Value
Port Config	PCFG0	0x800	R/W	0x03020100	Port Configuration Register 0
	PCFG1	0x804	R/W	0x00000504	Port Configuration Register 1
	CIRQST	0x808	R	0x0000	Channel IRQ Status Register
PID Table	PIDT	0xF00	R/W	-	

PORT Register (PORT) 0xB0107*n*00 (n = Channel Number 0~5)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA[15:0]															

Field	Name	RW	Reset	Description
31-0	PORT	R/W	0	Data Read / Write Port. Any data written to this register is sent to the write(TX) FIFO. Any data read from this register is from the read(RX) FIFO. Read data is valid only if RBVCNT is not zero. Although this port can be written even when the core disabled, no serial bus cycle is generated until the core is enabled with valid parameters.

Status Register (STAT) 0xB0107*n*04(n = Channel Number 0~5)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-				WBVCNT				-				RBVCNT			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-							WOR	RUR	WUR	ROR	RF	WE	RNE	WTH	RTH

Field	Name	Reset	RW	Description
31-28	-	-	-	Undefined
27-24	WBVCNT	-	R/W	Write(Transmit) FIFO valid entry count Maximum value dependent on the data bit width
23-20	-	-	-	Undefined
19-16	RBVCNT	-	R/W	Read(Receive) FIFO valid entry count Maximum value dependent on the data bit width
15-9	-	-	-	Undefined
8	WOR	-	R/C	Write FIFO over-run error flag When writing 1, it is cleared. If INTEN.RC = 1, it is also cleared when reading it.
7	RUR	-	R/C	Read FIFO under-run error flag When writing 1, it is cleared. If INTEN.RC = 1, it is also cleared when reading it.
6	WUR	-	R/C	Write FIFO under-run error flag When writing 1, it is cleared. If INTEN.RC = 1, it is also cleared when reading it.
5	ROR	-	R/C	Read FIFO over-run error flag When writing 1, it is cleared. If INTEN.RC = 1, it is also cleared when reading it.
4	RF	-	R	Read FIFO full flag
3	WE	-	R	Write FIFO empty flag
2	RNE	-	R	Read FIFO not empty flag
1	WTH	-	R	Write FIFO valid entry count is under threshold.
0	RTH	-	R	Read FIFO valid entry increased over threshold.

Interrupt Enable Register (INTEN) **0xB0107*n*08 (n = Channel Number 0~5)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DW	DR	--		SHT	SBT	SHR	SBR	-	CFGWTH			-	CFGRTH		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RC	--						IRQEN								

Field	Name	RW	Reset	Description
31	DW	R/W	0	DMA request enable for TX FIFO '1' for enable, '0' for disable
30	DR	R/W	0	DMA request enable for RX FIFO '1' for enable, '0' for disable
27	SHT	R/W	0	TX half-word swap in word
26	SBT	R/W	0	TX byte swap in half-word
25	SHR	R/W	0	RX half-word swap in word
24	SBR	R/W	0	RX byte swap in half-word
22-20	CFGWTH	R/W	0	Transmit FIFO threshold for interrupt/DMA request
18-16	CFGRTH	R/W	0	Receive FIFO threshold for interrupt/DMA request
15	RC	R/W	0	Clear status[8:0] at the end of read cycle. When this bit is set as "1", status[8:0] is all cleared whenever GPSBSTAT register is read.
14-9	-	-	-	Undefined
8-0	IRQEN	R/W	0	Interrupt enable signals [8] : TX FIFO over-run error [7] : RX FIFO under-run error [6] : TX FIFO under-run error [5] : RX FIFO over-run error [4] : RX FIFO full [3] : TX FIFO empty [2] : RX FIFO not empty [1] : Valid entry count of TX FIFO is under threshold [0] : Valid entry count of RX FIFO is more than threshold

Mode Register (MODE) **0xB0107*n*0C(n = Channel Number 0~5)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIVLDV								TRE	THL	TSU	PCS	PCD	PWD	PRD	PCK
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CRF	CWF	-	BWS					SD	LB	SDO	CTF	EN	SLV	MD	

Field	Name	RW	Reset	Description
31-24	DIVLDV	R/W	0	Clock divider load value FSCKO[27] = FGCLK[28] / ((DIVLDV+1) * 2)
23	TRE	R/W	0	Master FRM recovery time tRECV = (TRE + 1) * (SCKO period)
22	THL	R/W	0	Master FRM hold time tHOLD = (THL + 1) * (SCKO period)
21	TSU	R/W	0	Master FRM setup time tSETUP = (TSU + 1) (SCKO period)
20	PCS	R/W	0	Polarity control for CS(FRM) – Master Only '0' for active low (default) / must be '0' for SSP '1' for active high
19	PCD	R/W	0	Polarity control for CMD(FRM) – Master only '0' for active low / must be '1' for SSP '1' for active high
18	PWD	R/W	0	Polarity control for transmitting data – Master Only '0' for falling edge of SCK '1' for rising edge of SCK / must be '1' for SSP
17	PRD	R/W	0	Polarity control for receiving data – Master only '0' for rising edge of SCK '1' for falling edge of SCK / must be '1' for SSP
16	PCK	R/W	0	Polarity control for serial clock '0' for master SCKO start from "low" for SPI timing 0, and "high" for SSP timing '0' for slave SCKI not inverted For SPI timing 0 and 3 '1' for master SCKO starts from "1" For SPI timing 2 and 3. '1' for slave SCKI inverted. For SPI timing 1, 2 and SSP timing
15	CRF	R/W	0	Clear receive FIFO counter
14	CWF	R/W	0	Clear transmit FIFO counter
12-8	BWS	R/W	0	Bit width selection Data bit width == BWS + 1 Valid range for BWS is 7 ~ 31 The FIFOs are configured according to BWS[4] as follows, "BWS[4]==1", 4x32 bit. "BWS[4]==0", 8x16bits.
7	SD	R/W	0	Data shift direction control

[27] FSCKO = Frequency of SCKO
[28] FGCLK = Frequency of GCLK (The GCLK is generated by the clock controller module.)

				'0' for shift left (MSB first)
6	LB	R/W	0	Data loop-back enable * SDO is feedback to SDI internally and the incoming data from external I/O is ignored. SDO output is not affected by this bit.
5	SDO	R//W	0	SDO output disable (slave mode only) '0' for enable '1' for disable
4	CTF	R/W	0	Continuous transfer mode enable '0' for single mode, '1' for continuous mode *If set to continuous mode, the CS signal keeps the active state until that 'CTF' is cleared and transmit FIFO is empty.*
3	EN	R/W	0	Operation enable bit '0' for disable '1' for enable
2	SLV	R/W	0	Slave mode configuration '0' for master mode '1' for slave mode
1-0	MD	R/W	0	Operation mode "00" for SPI compatible "01" for SSP compatible "1x" reserved for future use

For tRECV, tHOLD and tSETUP refer to tFRMW, tFRMH and tFRMS in Master Mode of Table 8.3 GPSB Interface Timing Parameters for SPI Timing 0

Control Register (CTRL) **0xB0107*n*10(n = Channel Number 0~5)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LCW	LCR		CMDEND					-			CMDSTART				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-			RDSTART					PLW	-		PSW				

Field	Name	RW	Reset	Description
31	LCW	R/W	0	Last clock disable for write cycle '1' for enable
30	LCR	R/W	0	Last clock disable for read cycle '1' for enable
29	-	-	-	Undefined
28-24	CMDEND	R/W	0	Command end position
20-16	CMDSTART	R/W	0	Command start position
12-8	RDSTART	R/W	0	Read data start position
7	PLW	R/W	0	Polarity control for write command
6-5	-	-	-	Undefined
4-0	PSW	R/W	0	Write command position

Counter & Ext. Event Control Register (EVTCTRL) 0xB0107*n*14

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXCRX	TXCREP	EXTEN	EXTDCHK	EXTDPOL											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXCV															

Field	Name	RW	Reset	Description
31	TXCRX	R/W	0	Tx Counter Rx Mode
30	TXCREP	R/W	0	Tx Counter Repeate Enable
29	EXTEN	R/W	0	External Event Enable
28	EXTDCHK	R/W	0	Data Check Enable
27	EXTDPOL	R/W	0	GSDI Polarity
26-16	-	-	-	Undefined
15-0	TXCV	R/W	0	Tx Counter Load Value

Counter Current Value Register (CCV) 0xB0107*n*18

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FSDI															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXC															

Field	Name	RW	Reset	Description
31	FSDI	R	0	Current Serial data Input
30-16	-	-	-	Undefined
15-0	TXC	R	0	Current Tx Counter Value

TX Base Register (TX base) 0xB0107*m*20(m = Channel Number 0~2)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TX_BASE [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX_BASE [15:0]															

Field	Name	RW	Reset	Description
31-0	TX_BASE	R/W	0	TX Base Register

RX Base Register (RX base) 0xB0107*m* 24

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RX_BASE [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RX_BASE [15:0]															

Field	Name	RW	Reset	Description
31-0	RX_BASE	R/W	0	RX Base Register

Packet Register (PACKET) 0xB0107*m* 28

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-			COUNT												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-			SIZE												

Field	Name	RW	Reset	Description
28-16	COUNT	R/W	0	Packet number information (COUNT + 1)
12-0	SIZE	R/W	0	Packet size information

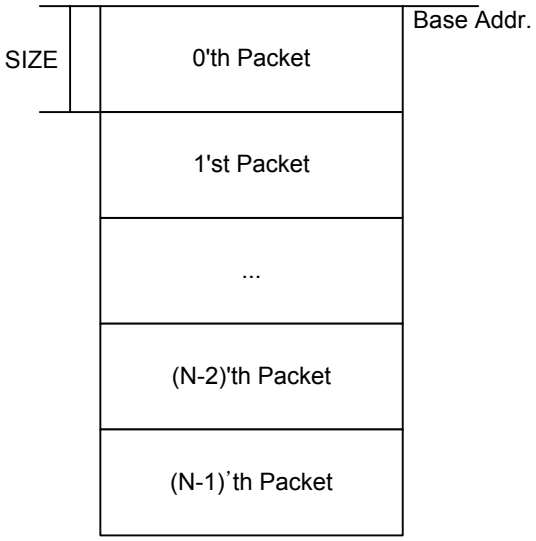


Figure 8.3 Packet Structure

DMA Control Register (DMA_CTRL) **0xB0107*m*2C(m = Channel Number 0~2)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DTE	DRE	CT	END	-								MP	MS	TXAM	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXAM		-								MD		-	PCLR	-	EN

Field	Name	RW	Reset	Description
31	DTE	R/W	0	Transmit DMA request enable '1' for enable
30	DRE	R/W	0	Receive DMA request enable '1' for enable
29	CT	R/W	0	Continuous mode enable (Master) '1' for enable
28	END	R/W	0	Byte endian mode register '0' for little-endian '1' for big-endian *If the byte or half-word transfer mode of GPSB mode were used, this field should be '1'.* *The value of '0' is allowed for word transfer only.*
19	MP	R/W	0	PID match mode register '1' for enable (MPEG2-TS mode)
18	MS	R/W	0	Sync byte match control register '1' for enable (MPEG2-TS mode)
17-16	TXAM	R/W	0	TX addressing mode '0' : Multiple Packet '1' : Fixed address (base) '2', '3' : Single Packet
15-14	RXAM	R/W	0	RX addressing mode '0' : Multiple Packet '1' : Fixed address (base) '2', '3' : Single Packet
5-4	MD	R/W	0	DMA mode register "00" : normal mode "01" : MPEG2-TS mode "1x" : Reserved
2	PCLR	W	0	Clear TX/RX packet counter
0	EN	R/W	0	DMA enable register '1' for enable

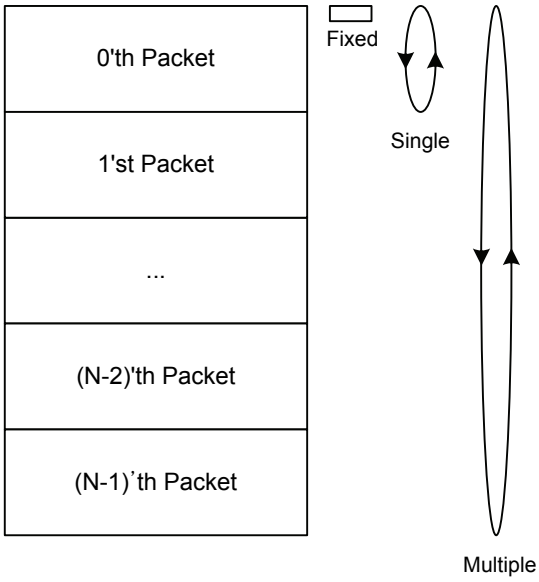


Figure 8.4 TX/RX Addressing Modes

DMA STATUS Register (DMA_STAT) **0xB0107*m*30(m = Channel Number 0~2)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		RXPCNT													0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0		TXPCNT													

Field	Name	RW	Reset	Description
29-17	RXPCNT	R	0	Receive packet count register
12-0	TXPCNT	R	0	Transmit packet count register

DMA IRQ Register (DMAICR) **0xB0107*m*34**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-		ISD	ISP	-							IRQS	-		IED	IEP
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-		IRQPCNT													

Field	Name	RW	Reset	Description
29	ISD	R/C	0	IRQ status for “Done Interrupt” When the number of packets which DMA transfers/receives is equal to the number of packets specified in PACKET register, it is issued. Therefore, it is not valid in continuous mode. When writing 1, it is cleared.
28	ISP	R/C	0	IRQ status for “Packet Interrupt”. It is issued every IRQPCNT packets which DMA transfers/receives. When writing 1, it is cleared.
20	IRQS	R/W	0	IRQ select register ‘0’ for receiving ‘1’ for transmitting
17	IED	R/W	0	IRQ enable for “Done Interrupt”
16	IEP	R/W	0	IRQ enable for “Packet Interrupt”
12-0	IRQPCNT	R/W	0	IRQ packet count register

PID Table (PIDT) 0xB0107F00 ~ 0xB0107F80

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH2	CH1	CH0													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-			PID												

Field	Name	RW	Reset	Description
31	CH2	R/W	X	Channel 2 enable
30	CH1	R/W	X	Channel 1 enable
29	CH0	R/W	X	Channel 0 enable
12-0	PID	R/W	X	PID value

- Before starting the PID matching, the CH2, CH1 and CH0 fields of all the table entries should be initialized.

Port Configuration Register 0(PCFG0) 0xB0107800

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3								CH2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CH1								CH0							

Field	Name	RW	Reset	Description
31-24	CH3	R/W	3	Channel 3 port mapping register
23-16	CH2	R/W	2	Channel 2 port mapping register
15-8	CH1	R/W	1	Channel 1 port mapping register
7-0	CH0	R/W	0	Channel 0 port mapping register

- The port map value for each channel should be different.

Port Configuration Register 1(PCFG1) 0xB0107804

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CH5								CH4							

Field	Name	RW	Reset	Description
15-8	CH5	R/W	5	Channel 5 port mapping register
7-0	CH4	R/W	4	Channel 4 port mapping register

- The port map value for each channel should be different.

Channel IRQ Status Register (CIRQST) **0xB0107808**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				0	ISTC5	0	ISTC4	0	ISTC3	ISTD2	ISTC2	ISTD1	ISTC1	ISTD0	ISTC0

Field	Name	RW	Reset	Description
10	ISTC5	R	0	GPSB core IRQ status register for channel 5
8	ISTC4	R	0	GPSB core IRQ status register for channel 4
6	ISTC3	R	0	GPSB core IRQ status register for channel 3
5	ISTD2	R	0	GPSB DMA IRQ status register for channel 2
4	ISTC2	R	0	GPSB core IRQ status register for channel 2
3	ISTD1	R	0	GPSB DMA IRQ status register for channel 1
2	ISTC1	R	0	GPSB core IRQ status register for channel 1
1	ISTD0	R	0	GPSB DMA IRQ status register for channel 0
0	ISTC0	R	0	GPSB core IRQ status register for channel 0

** If IRQ status of each channel were cleared, the corresponding field would be read '0'.*
** The IRQ mode of GPSB channel is preferred to "level-trigger" mode.*

8.4 GPSB Timing Diagram

The table and figures on the following page are the examples of GPSB MODE values to be programmed for SPI and SSP interface.

Table 8.2 GPSB Mode Values for SPI, SSP

Timing	CPOL/CPHA	Master	Slave
		PWD,PRD,PCK	PCK
SPI timing 0	00	000	0
SPI timing 1	01	110	1
SPI timing 2	10	001	1
SPI timing 3	11	111	0
SSP timing	NA	110	1

The figures below show the timing of each case listed in the table DIVLDV = 0 (GCLK / 2), Single Transfer Mode assumed

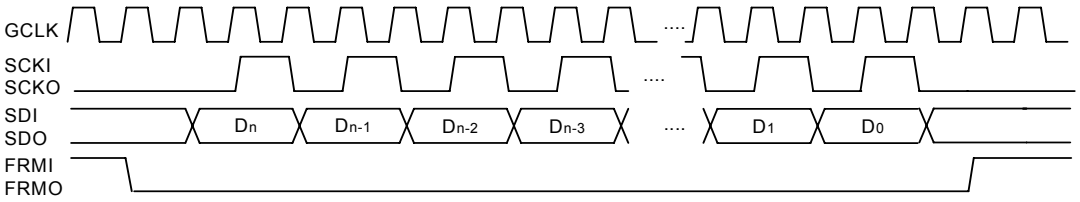


Figure 8.5 SPI Timing 0

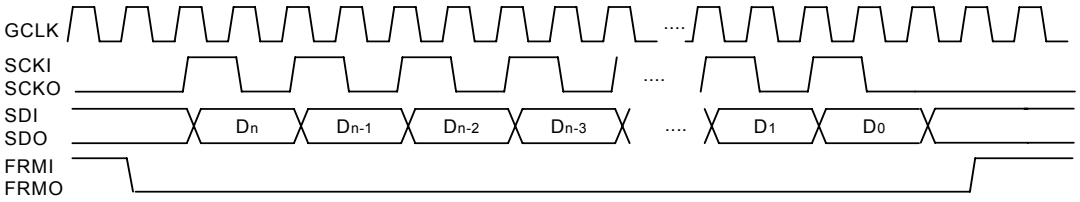


Figure 8.6 SPI Timing 1

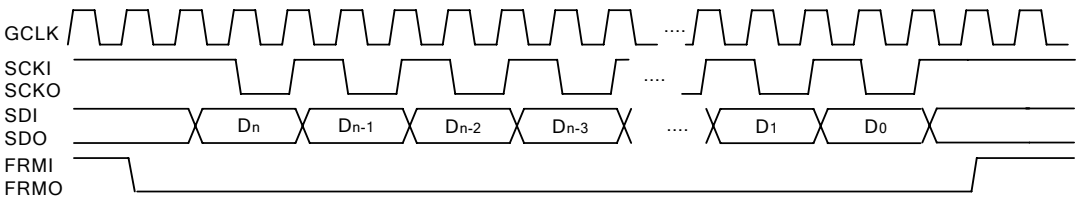


Figure 8.7 SPI Timing 2

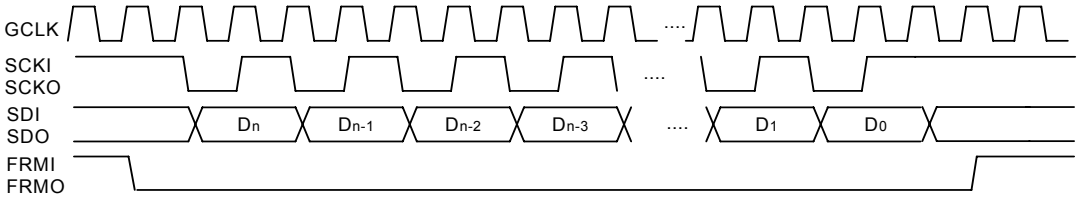


Figure 8.8 SPI Timing 3

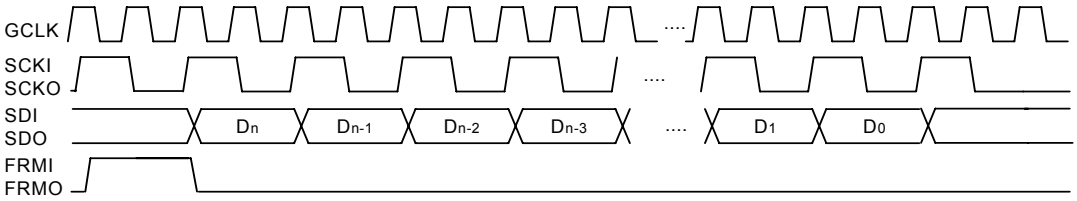


Figure 8.9 SSP Timing

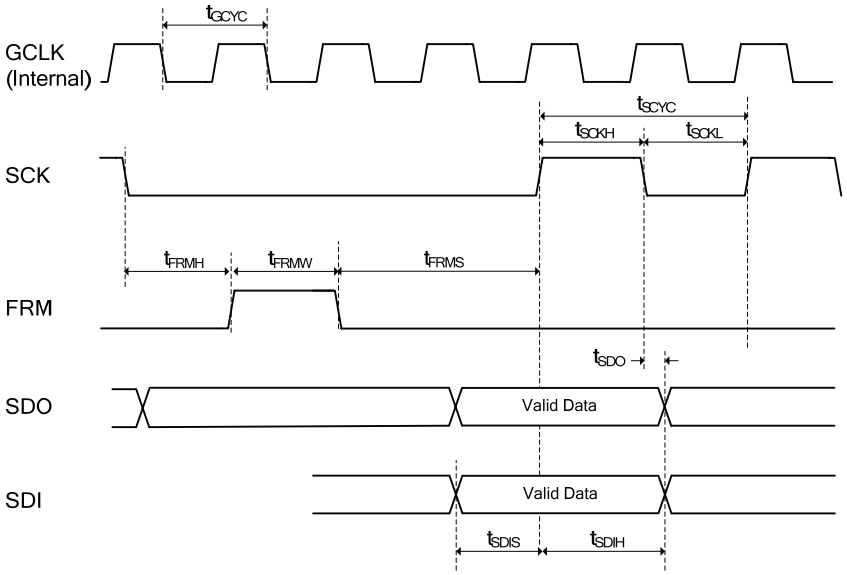


Figure 8.10 GPSB Interface Timing

Table 8.3 GPSB Interface Timing Parameters for SPI Timing 0

Parameter	Symbol	Min	Max	Unit
GCLK cycle time	t_{GCYC}	20	-	ns
SCK cycle time	t_{SCYC}	2 * t_{GCYC}	-	ns
SCK pulse width high	t_{SCKH}	t_{GCYC} - t_{SCKR}	-	ns
SCK pulse width low	t_{SCKL}	t_{GCYC} - t_{SCKF}	-	ns
SCK rise time	t_{SCKR}	-	10	ns
SCK fall time	t_{SCKF}	-	10	ns
Master Mode				
FRM output low to SCK rising edge	t_{FRMS}	t_{SCYC}	-	ns
FRM output high from SCK falling edge	t_{FRMH}	t_{SCYC}	-	ns
FRM output high pulse width	t_{FRMW}	t_{SCYC}	-	ns
SDO output delay from SCK falling edge	t_{SDO}	-	10	ns
SDI setup time to SCK rising edge	t_{SDIS}	10	-	ns
SDI hold time from SCK rising edge	t_{SDIH}	10	-	ns
Slave Mode				
FRM input low setup time to SCK rising edge	t_{FRMS}	25	-	ns
FRM input low hold time from SCK falling edge	t_{FRMH}	10	-	ns
FRM input high pulse width	t_{FRMW}	20	-	ns
SDO output delay from SCK falling edge	t_{SDO}	2	15	ns
SDI setup time to SCK rising edge	t_{SDIS}	10	-	ns
SDI hold time from SCK rising edge	t_{SDIH}	10	-	ns

Note:

1. CL = 30pF
2. GCLK is an internal signal
3. Parameters are for SPI Timing 0

8.5 MPEG2-TS Interface

The figures on the following page are the examples of MPEG2-TS bit stream. The packet of MPEG2-TS (Transport Stream) is composed of two groups that are the 188-bytes Payload data and 16bit Parity data. The Payload data including TS header within Sync byte is shown below Figure 8.11.

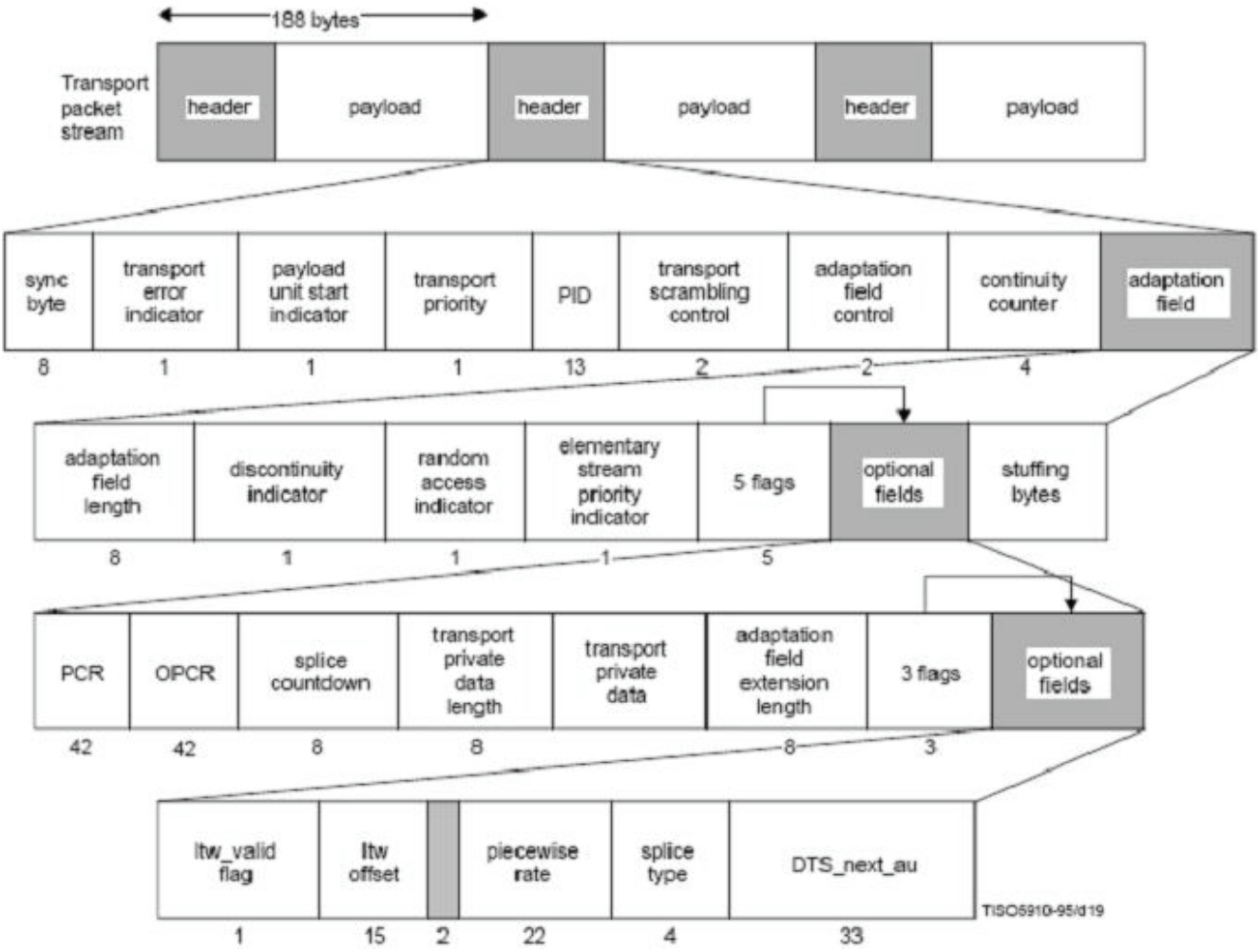


Figure 8.11 Bitstream of MPEG2-TS

In the Figure 8.11, the first byte of TS packet header is '0x47'. This sync byte arranges the serial stream data that is processed, stored, and read.

The input of MPEG2-TS interface is TS_CLK and TS packet data. TS packet data is composed in 204-byte unit and is separated by SYNC signal. Figure 8.12 shows the TS timing information.

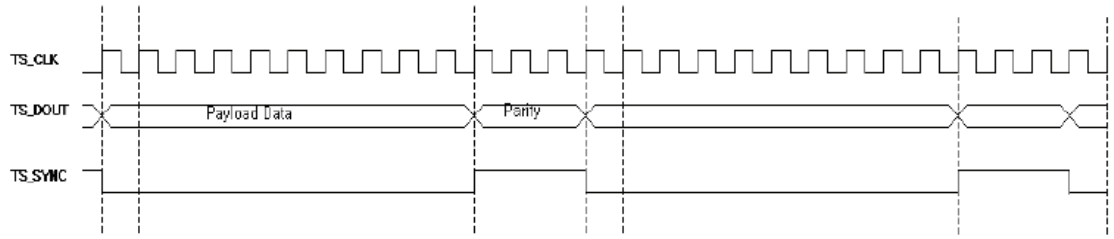


Figure 8.12 MPEG2-TS timing information

9 Overlay Mixer

9.1 Overview

Figure 9.1 shows the whole block diagram of Overlay Mixer and its corresponding registers

As can be seen below, Overlay Mixer supports Rotating / Mirroring image, bitwise ROP, Alpha-Blending, Arithmetic or Format Converting function. These functions can simultaneously work. Overlay Mixer has two different source channels and one destination channel. In other words, one output image can be created with two Input images by using an appropriate function of Overlay Mixer. Moreover, specific local region operation is available since Overlay Mixer supports Source Image Offset, Destination Image Offset and Window Offset.

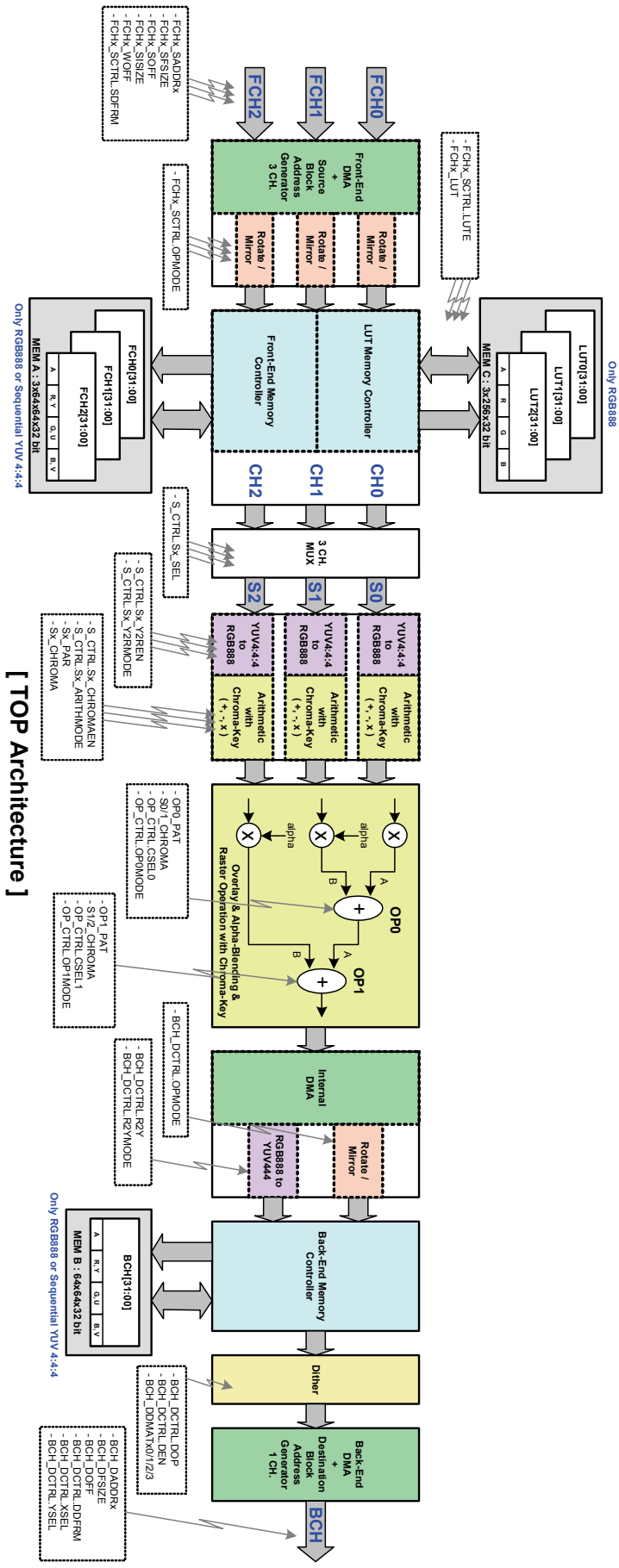


Figure 9.1 Overlay Mixer Block Diagram

9.2 Features

The features of Overlay Mixer are as follows

- Input / Output Image Format
  - Indexed Color Format : 8bpp
  - RGB Format : RGB332, RGB444, RGB454, RGB555, RGB565, RGB666, RGB888
  - Alpha-RGB(ARGB) Format : ARGB4444, ARGB3454, ARGB1555, ARGB4666, ARGB6666, ARGB4888, ARGB8888
  - Sequential YUV : YUV4:4:4, YUV4:2:2
  - Separated YUV : YUV4:4:4, YUV4:4:0, YUV4:2:2, YUV4:2:0, YUV4:1:1, YUV4:1:0
  - Interleaved YUV : YUV4:2:2, YUV4:2:0
- Rotate/Mirror Operation
  - 2 Channel Source 90/180/270 Rotate with Window Offset and Source Offset
  - 1 Channel Destination 90/180/270 Rotate with Destination Offset
  - 2 Channel Source Vertical/Horizontal/Vertical & Horizontal Mirror with Window Offset and Source Offset
  - 1 Channel Destination Vertical/Horizontal/Vertical & Horizontal Mirror with Destination Offset
- Arithmetic Operation
  - 2 Channel Arithmetic Operation with Chroma-Key Function
    - ◆ Arithmetic Function : Bypass, Fill, Inversion, Addition, Substract, Multiplication
- Raster Operation & Alpha-Blending & Overlay
  - 2 Channel to 1Channel ROP Operation with Chroma-Key & Window Offset
  - 2 Channel to 1Channel Alpha Blending with Chroma-Key & Window Offset
  - ROP Function : 16 Type
  - Alpha-Blending Operation by Fixed Alpha-Value
  - Alpha-Blending Operation by ARGB Input Image Format
- YUV to RGB Format Converting
  - 2 Independent Source YUV to RGB Format Converting
  - 4 YUV to RGB Format Converting Type
- RGB to YUV Format Converting
  - 1 Destination RGB to YUV Format Converting
  - 4 RGB to YUV Format Converting Type
- Color LUT (Lookup Table)
  - 3 Channel Source Indexed Color with LUT : 8bpp
  - 3 Channel Source Alpha with LUT : ARGB4444, ARGB3454, ARGB1555, ARGB4666, ARGB6666, ARGB4888, ARGB8888
- RGB Format Dithering
  - 1 Channel Destination Dithering : RGB332, RGB444, ARGB4444, RGB454, ARGB3454, RGB555, ARGB1555, RGB565, RGB666, ARGB4666, ARGB6666, ARGB4888

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Indexed Color Format	8bpp	Data3[7:0]								Data2[7:0]								Data1[7:0]								Data0[7:0]							
RGB Format	RGB332(8bit)	R3[2:0]			G3[2:0]			B3[1:0]		R2[2:0]			G2[2:0]			B2[1:0]		R1[2:0]			G1[2:0]			B1[1:0]		R0[2:0]			G0[2:0]			B0[1:0]
	RGB444(12bit)	PAD				R1[3:0]				G1[3:0]				B1[3:0]				PAD				R0[3:0]				G0[3:0]				B0[3:0]		
	ARGB4444(16bit)	A1[3:0]				R1[3:0]				G1[3:0]				B1[3:0]				A0[3:0]				R0[3:0]				G0[3:0]				B0[3:0]		
	RGB454(13bit)	PAD			R1[3:0]					G1[4:0]					B1[3:0]			PAD			R0[3:0]					G0[4:0]					B0[3:0]	
	ARGB3454(16bit)	A1[2:0]			R1[3:0]					G1[4:0]					B1[3:0]			A0[2:0]			R0[3:0]					G0[4:0]					B0[3:0]	
	RGB555(15bit)	P	R1[4:0]					G1[4:0]						B1[4:0]				P	R0[4:0]					G0[4:0]					B0[4:0]			
	ARGB1555(16bit)	A1	R1[4:0]					G1[4:0]						B1[4:0]				A0	R0[4:0]					G0[4:0]					B0[4:0]			
	RGB565(16bit)	R1[4:0]					G1[5:0]							B1[4:0]				R0[4:0]					G0[5:0]					B0[4:0]			
	RGB666(18bit)	PAD														R[5:0]						G[5:0]						B[5:0]			
	ARGB4666(22bit)	PAD											A[3:0]			R[5:0]						G[5:0]						B[5:0]			
	ARGB6666(24bit)	PAD								A[5:0]						R[5:0]						G[5:0]						B[5:0]			
	RGB888(24bit)	PAD								R[7:0]								G[7:0]								B[7:0]							
	ARGB4888(28bit)	PAD			A[3:0]					R[7:0]								G[7:0]								B[7:0]							
	ARGB8888(32bit)	A[7:0]								R[7:0]								G[7:0]								B[7:0]							
Sequential YUV Format	Sequential YUV 4:4:4	PAD								Y[7:0]								U[7:0]								V[7:0]							
	Sequential YUV 4:2:2	V0[7:0]								Y1[7:0]								U0[7:0]								Y0[7:0]							
Separated YUV Format	Separated YUV 4:4:4																																
	Separated YUV 4:4:0	Y[7:0]								Y[7:0]								Y[7:0]								Y[7:0]							
	Separated YUV 4:2:2	U[7:0]								U[7:0]								U[7:0]								U[7:0]							
	Separated YUV 4:2:0	V[7:0]								V[7:0]								V[7:0]								V[7:0]							
	Separated YUV 4:1:1																																
	Separated YUV 4:1:0																																
Interleaved YUV Format	Interleaved YUV 4:2:2	Y[7:0]								Y[7:0]								Y[7:0]								Y[7:0]							
	Interleaved YUV 4:2:0	V[7:0]								U[7:0]								V[7:0]								U[7:0]							
	Interleaved YUV 4:2:2 (Swap U and V)	Y[7:0]								Y[7:0]								Y[7:0]								Y[7:0]							
	Interleaved YUV 4:2:0 (Swap U and V)	U[7:0]								V[7:0]								U[7:0]								V[7:0]							

*** Refer to Figure 9.4 for detail information of YUV Format**

Figure 9.2 Supported Data Format of Overlay Mixer

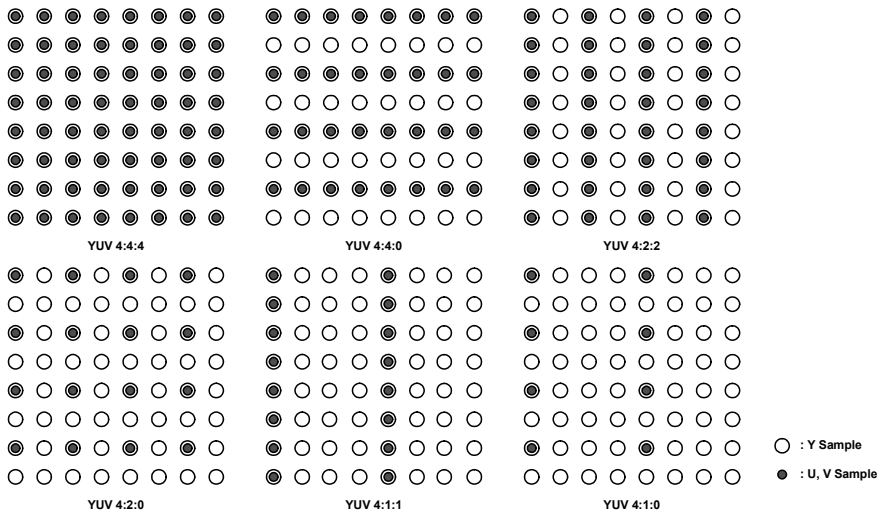


Figure 9.3 Sampling of YUV Format

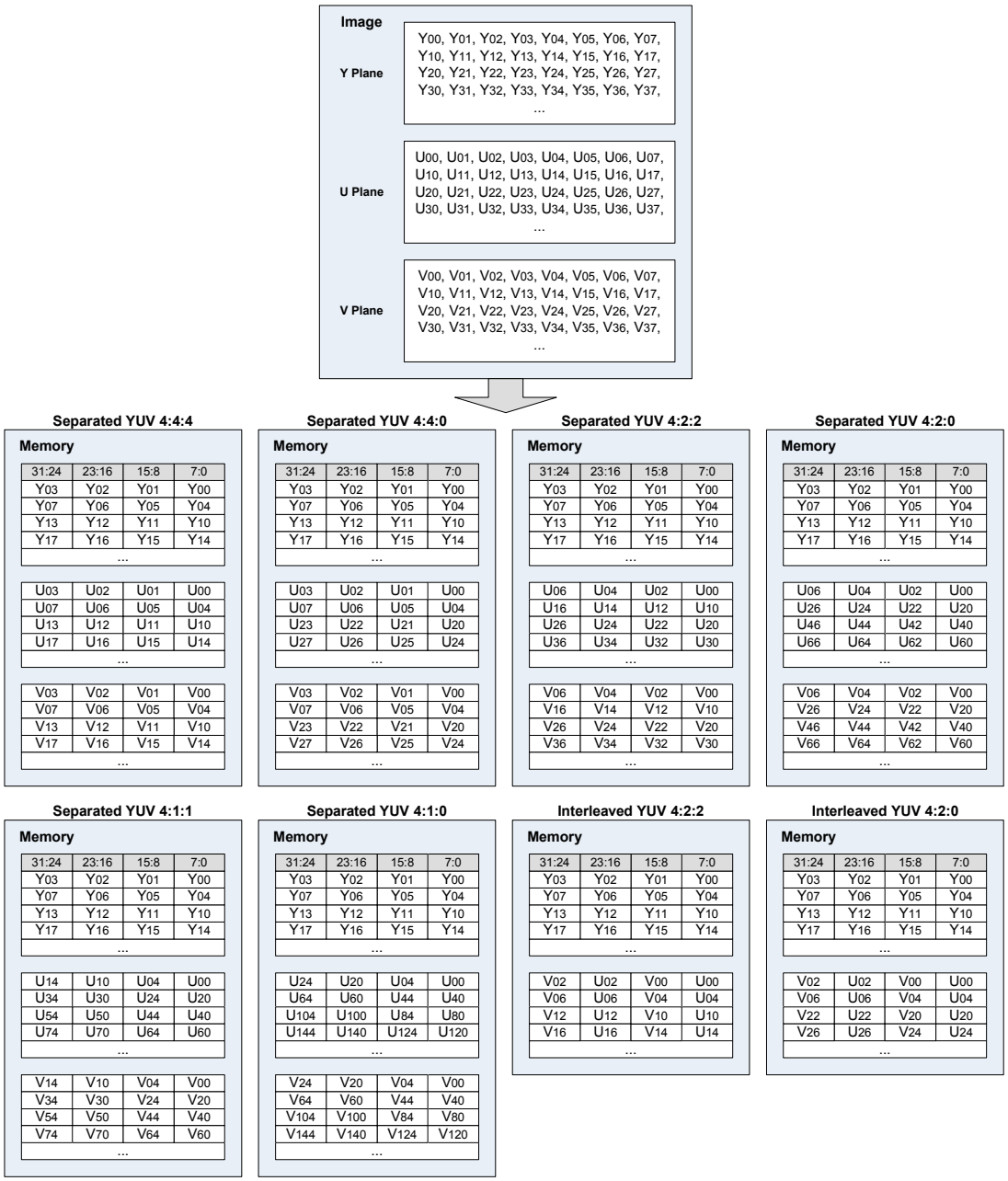


Figure 9.4 Detail Information of YUV Format

9.3 Register Description

Table 9.1 Overlay Mixer Register Map (Base Address = 0xB0070000)

Name	Address	Type	Reset	Description
FCH0_SADDR0	0x00	R/W	0x00000000	Front-End Channel 0 Source Address 0
FCH0_SADDR1	0x04	R/W	0x00000000	Front-End Channel 0 Source Address 1
FCH0_SADDR2	0x08	R/W	0x00000000	Front-End Channel 0 Source Address 2
FCH0_SFSIZE	0x0C	R/W	0x00000000	Front-End Channel 0 Source Frame Pixel Size
FCH0_SOFF	0x10	R/W	0x00000000	Front-End Channel 0 Source Pixel Offset
FCH0_SISIZE	0x14	R/W	0x00000000	Front-End Channel 0 Source Image Pixel Size
FCH0_WOFF	0x18	R/W	0x00000000	Front-End Channel 0 Window Pixel Offset
FCH0_SCTRL	0x1C	R/W	0x00000000	Front-End Channel 0 Control
FCH1_SADDR0	0x20	R/W	0x00000000	Front-End Channel 1 Source Address 0
FCH1_SADDR1	0x24	R/W	0x00000000	Front-End Channel 1 Source Address 1
FCH1_SADDR2	0x28	R/W	0x00000000	Front-End Channel 1 Source Address 2
FCH1_SFSIZE	0x2C	R/W	0x00000000	Front-End Channel 1 Source Frame Pixel Size
FCH1_SOFF	0x30	R/W	0x00000000	Front-End Channel 1 Source Pixel Offset
FCH1_SISIZE	0x34	R/W	0x00000000	Front-End Channel 1 Source Image Pixel Size
FCH1_WOFF	0x38	R/W	0x00000000	Front-End Channel 1 Window Pixel Offset
FCH1_SCTRL	0x3C	R/W	0x00000000	Front-End Channel 1 Control
FCH2_SADDR0	0x40	R/W	0x00000000	Front-End Channel 2 Source Address 0
FCH2_SADDR1	0x44	R/W	0x00000000	Front-End Channel 2 Source Address 1
FCH2_SADDR2	0x48	R/W	0x00000000	Front-End Channel 2 Source Address 2
FCH2_SFSIZE	0x4C	R/W	0x00000000	Front-End Channel 2 Source Frame Pixel Size
FCH2_SOFF	0x50	R/W	0x00000000	Front-End Channel 2 Source Pixel Offset
FCH2_SISIZE	0x54	R/W	0x00000000	Front-End Channel 2 Source Image Pixel Size
FCH2_WOFF	0x58	R/W	0x00000000	Front-End Channel 2 Window Pixel Offset
FCH2_SCTRL	0x5C	R/W	0x00000000	Front-End Channel 2 Control
S0_CHROMA	0x60	R/W	0x00000000	Source 0 Chroma-Key Parameter
S0_PAR	0x64	R/W	0x00000000	Source 0 Arithmetic Parameter
S1_CHROMA	0x68	R/W	0x00000000	Source 1 Chroma-Key Parameter
S1_PAR	0x6C	R/W	0x00000000	Source 1 Arithmetic Parameter
S2_CHROMA	0x70	R/W	0x00000000	Source 2 Chroma-Key Parameter
S2_PAR	0x74	R/W	0x00000000	Source 2 Arithmetic Parameter
S_CTRL	0x78	R/W	0x00000000	Source Control Register
-	0x7C	-	-	Reserved
OP0_PAT	0x80	R/W	0x00000000	Source Operator 0 Pattern
OP1_PAT	0x84	R/W	0x00000000	Source Operator 1 Pattern
OP_CTRL	0x88	R/W	0x00000000	Source Operation Control Register
-	0x8C	-	-	Reserved
BCH_DADDR0	0x90	R/W	0x00000000	Back-End Channel Destination Address 0
BCH_DADDR1	0x94	R/W	0x00000000	Back -End Channel Destination Address 1
BCH_DADDR2	0x98	R/W	0x00000000	Back -End Channel Destination Address 2
BCH_DFSIZE	0x9C	R/W	0x00000000	Back -End Channel Destination Frame Pixel Size
BCH_DOFF	0xA0	R/W	0x00000000	Back -End Channel Destination Pixel Offset
BCH_DCTRL	0xA4	R/W	0x00000000	Back -End Channel Control
-	0xA8 – 0xAF	-	-	Reserved
BCH_DDMAT0	0xB0	R/W	0x00000000	Back-End Channel Destination Dither Matrix 0
BCH_DDMAT1	0xB4	R/W	0x00000000	Back-End Channel Destination Dither Matrix 1
BCH_DDMAT2	0xB8	R/W	0x00000000	Back-End Channel Destination Dither Matrix 2
BCH_DDMAT3	0xBC	R/W	0x00000000	Back-End Channel Destination Dither Matrix 3
OM_CTRL	0xC0	R/W	0x00000000	Overlay Mixer Control
OM_IREQ	0xC4	R/W	0x00000000	Overlay Mixer Interrupt Request
-	0xC8 – 0x3FF	-	-	Reserved
FCH0_LUT	0x400 – 0x7FF	R/W	-	Front-End Channel 0 Lookup Table
FCH1_LUT	0x800 – 0xBFF	R/W	-	Front-End Channel 1 Lookup Table
FCH2_LUT	0xC00 – 0xFFF	R/W	-	Front-End Channel 2 Lookup Table

Front-End Channel 0 Source Address 0(FCH0_SADDR0) 0xB0070000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR0[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	SADDR0	R/W	0x0	Source Address 0 • N (0~ 4095) This register determines the Y image base address of FCH0 when the data format of FCH0 is separated YUV or interleaved YUV format. But, in case the data format of FCH0 is a sequential YUV or an RGB, it determines the base address of the whole image. The data format of FCH0 is determined by the FCH0_SCTRL.SDFRM bits.
1-0	-	R/W	0x0	Reserved

Front-End Channel 0 Source Address 1 (FCH0_SADDR1) 0xB0070004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR1[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	SADDR1	R/W	0x0	Source Address 1 • N (0~ 4095) This register determines the U image base address of FCH0 when the data format of FCH0 is separated YUV or interleaved YUV format. But, in case the data format of FCH0 image is a sequential YUV or an RGB, it is not used.
1-0	-	R/W	0x0	Reserved

Front-End Channel 0 Source Address 2 (FCH0_SADDR2) 0xB0070008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR2[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	SADDR2	R/W	0x0	Source Address 2 • N (0~ 4095) This register determines the V image base address of FCH0 image when the data format of FCH0 is separated YUV format. But, in case the data format of FCH0 image is a sequential YUV, interleaved YUV or an RGB, it is not used.
1-0	-	R/W	0x0	Reserved

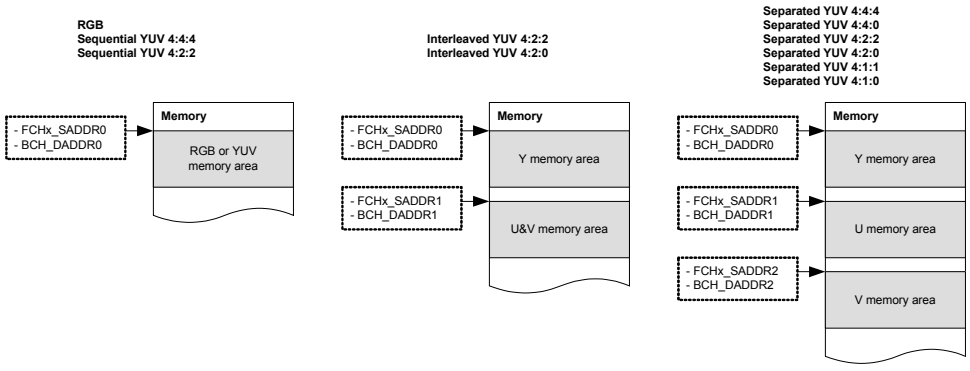


Figure 9.5 Source/Destination Base Address of Data Format

Front-End Channel 0 Source Frame Pixel Size (FCH0_SFSIZE) 0xB007000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SFSIZE_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				SFSIZE_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	SFSIZE_Y	R/W	0x0	Source Frame Pixel Size Y • N (0~ 4095) Height of FCH0 image by pixel units
15-12	-	R/W	0x0	Reserved
11-0	SFSIZE_X	R/W	0x0	Source Frame Pixel Size X • N (0~ 4095) Width of FCH0 image by pixel units

Front-End Channel 0 Source Pixel Offset (FCH0_SOFF) 0xB0070010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SOFF_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				SOFF_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	SOFF_Y	R/W	0x0	Source Pixel Offset Y • N (0~ 4095) Y Axis Source Pixel Offset
15-12	-	R/W	0x0	Reserved
11-0	SOFF_X	R/W	0x0	Source Pixel Offset X • N (0~ 4095) X Axis Source Pixel Offset

Front-End Channel 0 Source Image Pixel Size (FCH0_SISIZE) 0xB0070014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SISIZE_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				SISIZE_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	SISIZE_Y	R/W	0x0	Source Image Pixel Size Y • N (0~ 4095) Y Axis Source Image Pixel Size
15-12	-	R/W	0x0	Reserved
11-0	SISIZE_X	R/W	0x0	Source Image Pixel Size X • N (0~ 4095) X Axis Source Image Pixel Size

Front-End Channel 0 Window Pixel Offset (FCH0_WOFF) 0xB0070018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				WOFF_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				WOFF_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	WOFF_Y	R/W	0x0	Window Pixel Offset Y • N (0~ 4095) Y Axis Window Pixel Offset
15-12	-	R/W	0x0	Reserved
11-0	WOFF_X	R/W	0x0	X Axis Window Pixel Offset • N (0~ 4095) Window Pixel Offset X

When FCH0 Image is selected as Source 0 by SCTRL.S0SEL register, Window Offset should definitely be '0'.

Front-End Channel 0 Control (FCH0_SCTRL) 0xB007001C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					SSB[2:0]			Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MABC	Reserved		LUTE	SSUV	OPMODE[2:0]			0	ZF[1:0]		SDFRM[4:0]				

Field	Name	RW	Reset	Description
31-27	-	R/W	0x0	Reserved
26-24	SSB	R/W	0x0	Source Swap Byte • 000b: ARGB • 001b: ARBG • 010b: AGRB • 011b: AGBR • 100b: ABRG • 101b: ABGR • 110b: RABG • 111b: BGRA This field applies only when Data Format is RGB Format. In case of 110b or 111b, the operations using Alpha Value are not supported.
23-16	-	R/W	0x0	Reserved
15	MABC	R/W	0x0	AXI Bus Master Address Boundary Control • 0: AXI bus master address boundary is set 4KByte. • 1: AXI bus master address boundary is set 1KByte.
14-13	-	R/W	0x0	Reserved
12	LUTE	R/W	0x0	LUT Enable • 0: LUT disable. • 1: LUT enable.
11	SSUV	R/W	0x0	Source Swap U for V • 0: Keep on U and V. (LSB = U, MSB = V) • 1: Swap U for V. (LSB = V, MSB = U) This filed applies only when Data Format is Interleaved YUV Format.
10-8	OPMODE	R/W	0x0	Operation Mode • 00xb: Data Copy • 010b: Horizontal Mirror • 011b: Vertical Mirror • 100b: Vertical & Horizontal Mirror • 101b: 90 Degree Rotate (Clockwise) • 110b: 180 Degree Rotate (Clockwise) • 111b: 270 Degree Rotate (Clockwise)
7	-	R/W	0x0	Reserved
6-5	ZF	R/W	0x0	Zero Fill • 00b: MSB Fill Mode Enable. • 01b: Zero Fill Mode Enable.

				• 1xb: HOB(High-Order Bits) Fill Mode Enable. /This field applies only when Data Format is RGB Format, not RGB888/ ARGB8888. Refer to the following figure “Example: Input Source Image Data to RGB888/ARGB8888 Conversion” for details.
4-0	SDFRM	R/W	0x0	Source Data Format • 00000b: Separated YUV 4:4:4 • 00001b: Separated YUV 4:4:0 • 00010b: Separated YUV 4:2:2 • 00011b: Separated YUV 4:2:0 • 00100b: Separated YUV 4:1:1 • 00101b: Separated YUV 4:1:0 • 00110b: Interleaved YUV 4:2:2 • 00111b: Interleaved YUV 4:2:0 • 01000b: Sequential YUV 4:4:4 • 01001b: Sequential YUV 4:2:2 • 0101xb: RGB332 (8bpp) • 01100b: RGB444 • 01101b: ARGB4444 • 01110b: RGB454 • 01111b: ARGB3454 • 10000b: RGB555 • 10001b: ARGB1555 • 1001xb: RGB565 • 10100b: RGB666 • 10101b: ARGB4666 • 10110b: ARGB6666 • 10111b: RGB888 • 11000b: ARGB4888 • 11001b: ARGB8888 • else: Not used.

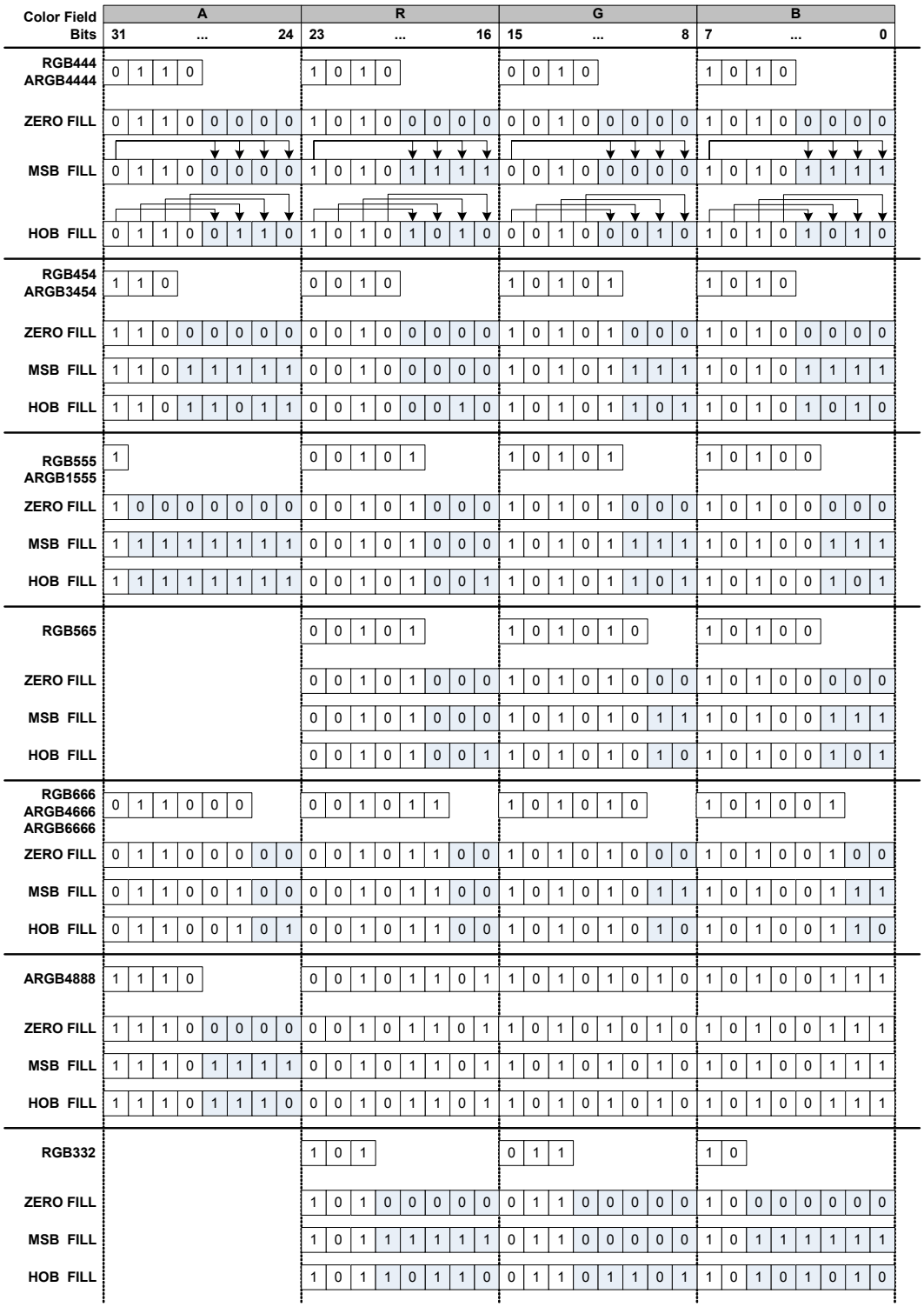


Figure 9.6 Example: Input Source Image Data to RGB888/ARGB8888 Conversion

Front-End Channel 1 Source Address 0 (FCH1_SADDR0) 0xB0070020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR0[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	SADDR0	R/W	0x0	Source Address 0 • N (0~ 4095) This register determines the Y image base address of FCH1 when the data format of FCH1 is separated YUV or interleaved YUV format. But, in case the data format of FCH1 is a sequential YUV or an RGB, it determines the base address of the whole image. The data format of FCH1 is determined by the FCH0_SCTRL.SDFRM bits.
1-0	-	R/W	0x0	Reserved

Front-End Channel 1 Source Address 1 (FCH1_SADDR1) 0xB0070024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR1[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	SADDR1	R/W	0x0	Source Address 1 • N (0~ 4095) This register determines the U image base address of FCH1 when the data format of FCH1 is separated YUV or interleaved YUV format. But, in case the data format of FCH1 image is a sequential YUV or an RGB, it is not used.
1-0	-	R/W	0x0	Reserved

Front-End Channel 1 Source Address 2 (FCH1_SADDR2) 0xB0070028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR2[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	SADDR2	R/W	0x0	Source Address 2 • N (0~ 4095) This register determines the V image base address of FCH1 image when the data format of FCH1 is separated YUV format. But, in case the data format of FCH1 image is a sequential YUV, interleaved YUV or an RGB, it is not used.
1-0	-	R/W	0x0	Reserved

Front-End Channel 1 Source Frame Pixel Size (FCH1_SFSIZE) 0xB007002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SFSIZE_Y[11:0]											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				SFSIZE_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	RW	0x0	Reserved
27-16	SFSIZE_Y	RW	0x0	Source Frame Pixel Size Y • N (0~ 4095) Height of FCH1 image by pixel units
15-12	-	RW	0x0	Reserved
11-0	SFSIZE_X	RW	0x0	Source Frame Pixel Size X • N (0~ 4095) Width of FCH1 image by pixel units

Front-End Channel 1 Source Pixel Offset (FCH1_SOFF) 0xB0070030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SOFF_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				SOFF_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	SOFF_Y	R/W	0x0	Source Pixel Offset Y • N (0~ 4095) Y Axis Source Pixel Offset
15-12	-	R/W	0x0	Reserved
11-0	SOFF_X	R/W	0x0	Source Pixel Offset X • N (0~ 4095) X Axis Source Pixel Offset

Front-End Channel 1 Source Image Pixel Size (FCH1_SISIZE) 0xB0070034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SISIZE_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				SISIZE_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	SISIZE_Y	R/W	0x0	Source Image Pixel Size Y • N (0~ 4095) Y Axis Source Image Pixel Size
15-12	-	R/W	0x0	Reserved
11-0	SISIZE_X	R/W	0x0	Source Image Pixel Size X • N (0~ 4095) X Axis Source Image Pixel Size

Front-End Channel 1 Window Pixel Offset (FCH1_WOFF) 0xB0070038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				WOFF_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				WOFF_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	WOFF_Y	R/W	0x0	Window Pixel Offset Y • N (0~ 4095) Y Axis Window Pixel Offset
15-12	-	R/W	0x0	Reserved
11-0	WOFF_X	R/W	0x0	X Axis Window Pixel Offset • N (0~ 4095) Window Pixel Offset X

When FCH1 Image is selected as Source 0 by SCTRL.S0SEL register, Window Offset should definitely be ‘0’.

Front-End Channel 1 Control (FCH1_SCTRL) 0xB007003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					SSB[2:0]			Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MABC	Reserved		LUTE	SSUV	OPMODE[2:0]			0	ZF[1:0]		SDFRM[4:0]				

Field	Name	RW	Reset	Description
31-27	-	R/W	0x0	Reserved
26-24	SSB	R/W	0x0	Source Swap Byte • 000b: ARGB • 001b: ARBG • 010b: AGRB • 011b: AGBR • 100b: ABRG • 101b: ABGR • 110b: RABG • 111b: BGRA This field applies only when Data Format is RGB Format. In case of 110b or 111b, the operations using Alpha Value are not supported.
23-16	-	R/W	0x0	Reserved
15	MABC	R/W	0x0	AXI Bus Master Address Boundary Control • 0: AXI bus master address boundary is set 4KByte. • 1: AXI bus master address boundary is set 1KByte.
14-13	-	R/W	0x0	Reserved
12	LUTE	R/W	0x0	LUT Enable • 0: LUT disable. • 1: LUT enable.
11	SSUV	R/W	0x0	Source Swap U for V • 0: Keep on U and V. (LSB = U, MSB = V) • 1: Swap U for V. (LSB = V, MSB = U) This filed applies only when Data Format is Interleaved YUV Format.
10-8	OPMODE	R/W	0x0	Operation Mode • 00xb: Data Copy • 010b: Horizontal Mirror • 011b: Vertical Mirror • 100b: Vertical & Horizontal Mirror • 101b: 90 Degree Rotate (Clockwise) • 110b: 180 Degree Rotate (Clockwise) • 111b: 270 Degree Rotate (Clockwise)
7	-	R/W	0x0	Reserved
6-5	ZF	R/W	0x0	Zero Fill • 00b: MSB Fill Mode Enable. • 01b: Zero Fill Mode Enable. • 1xb: HOB(High-Order Bits) Fill Mode Enable. This field applies only when Data Format is RGB Format, not RGB888/ ARGB8888. Refer to the above figure “Example: Input Source Image Data to RGB888/ARGB8888 Conversion” for details.
4-0	SDFRM	R/W	0x0	Source Data Format • 00000b: Separated YUV 4:4:4 • 00001b: Separated YUV 4:4:0 • 00010b: Separated YUV 4:2:2 • 00011b: Separated YUV 4:2:0 • 00100b: Separated YUV 4:1:1 • 00101b: Separated YUV 4:1:0 • 00110b: Interleaved YUV 4:2:2 • 00111b: Interleaved YUV 4:2:0 • 01000b: Sequential YUV 4:4:4 • 01001b: Sequential YUV 4:2:2 • 0101xb: RGB332 (8bpp) • 01100b: RGB444

				• 01101b: ARGB4444 • 01110b: RGB454 • 01111b: ARGB3454 • 10000b: RGB555 • 10001b: ARGB1555 • 1001xb: RGB565 • 10100b: RGB666 • 10101b: ARGB4666 • 10110b: ARGB6666 • 10111b: RGB888 • 11000b: ARGB4888 • 11001b: ARGB8888 • else: Not used.

Front-End Channel 2 Source Address 0 (FCH2_SADDR0) 0xB0070040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR0[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	SADDR0	R/W	0x0	Source Address 0 • N (0~ 4095) This register determines the Y image base address of FCH2 when the data format of FCH2 is separated YUV or interleaved YUV format. But, in case the data format of FCH2 is a sequential YUV or an RGB, it determines the base address of the whole image. The data format of FCH2 is determined by the FCH0_SCTRL.SDFRM bits.
1-0	-	R/W	0x0	Reserved

Front-End Channel 2 Source Address 1 (FCH2_SADDR1) 0xB0070044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR1[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	SADDR1	R/W	0x0	Source Address 1 • N (0~ 4095) This register determines the U image base address of FCH2 when the data format of FCH2 is separated YUV or interleaved YUV format. But, in case the data format of FCH2 image is a sequential YUV or an RGB, it is not used.
1-0	-	R/W	0x0	Reserved

Front-End Channel 2 Source Address 2 (FCH2_SADDR2) 0xB0070048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR2[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	SADDR2	R/W	0x0	Source Address 2 • N (0~ 4095) This register determines the V image base address of FCH2 image when the data format of FCH2 is separated YUV format. But, in case the data format of FCH2 image is a sequential YUV, interleaved YUV or an RGB, it is not used.
1-0	-	R/W	0x0	Reserved

Front-End Channel 2 Source Frame Pixel Size (FCH2_SFSIZE) 0xB007004C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SFSIZE_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				SFSIZE_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	SFSIZE_Y	R/W	0x0	Source Frame Pixel Size Y • N (0~ 4095) Height of FCH2 image by pixel units
15-12	-	R/W	0x0	Reserved
11-0	SFSIZE_X	R/W	0x0	Source Frame Pixel Size X • N (0~ 4095) Width of FCH2 image by pixel units

Front-End Channel 2 Source Pixel Offset (FCH2_SOFF) 0xB0070050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SOFF_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				SOFF_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	SOFF_Y	R/W	0x0	Source Pixel Offset Y • N (0~ 4095) Y Axis Source Pixel Offset
15-12	-	R/W	0x0	Reserved
11-0	SOFF_X	R/W	0x0	Source Pixel Offset X • N (0~ 4095) X Axis Source Pixel Offset

Front-End Channel 2 Source Image Pixel Size (FCH2_SISIZE) 0xB0070054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SISIZE_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				SISIZE_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	SISIZE_Y	R/W	0x0	Source Image Pixel Size Y • N (0~ 4095) Y Axis Source Image Pixel Size
15-12	-	R/W	0x0	Reserved
11-0	SISIZE_X	R/W	0x0	Source Image Pixel Size X • N (0~ 4095) X Axis Source Image Pixel Size

Front-End Channel 2 Window Pixel Offset (FCH2_WOFF) 0xB0070058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				WOFF_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				WOFF_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	WOFF_Y	R/W	0x0	Window Pixel Offset Y • N (0~ 4095) Y Axis Window Pixel Offset
15-12	-	R/W	0x0	Reserved
11-0	WOFF_X	R/W	0x0	X Axis Window Pixel Offset • N (0~ 4095) Window Pixel Offset X

When FCH1 Image is selected as Source 0 by SCTRL.S0SEL register, Window Offset should definitely be '0'.

Front-End Channel 2 Control (FCH2_SCTRL) 0xB007005C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					SSB[2:0]			Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MABC	Reserved		LUTE	SSUV	OPMODE[2:0]			0	ZF[1:0]		SDFRM[4:0]				

Field	Name	RW	Reset	Description
31-27	-	R/W	0x0	Reserved
26-24	SSB	R/W	0x0	Source Swap Byte • 000b: ARGB • 001b: ARBG • 010b: AGRB • 011b: AGBR • 100b: ABRG • 101b: ABGR • 110b: RABG • 111b: BGRA This field applies only when Data Format is RGB Format. In case of 110b or 111b, the operations using Alpha Value are not supported.
23-16	-	R/W	0x0	Reserved
15	MABC	R/W	0x0	AXI Bus Master Address Boundary Control • 0: AXI bus master address boundary is set 4KByte. • 1: AXI bus master address boundary is set 1KByte.
14-13	-	R/W	0x0	Reserved
12	LUTE	R/W	0x0	LUT Enable • 0: LUT disable. • 1: LUT enable.
11	SSUV	R/W	0x0	Source Swap U for V • 0: Keep on U and V. (LSB = U, MSB = V) • 1: Swap U for V. (LSB = V, MSB = U) This filed applies only when Data Format is Interleaved YUV Format.
10-8	OPMODE	R/W	0x0	Operation Mode • 00xb: Data Copy • 010b: Horizontal Mirror • 011b: Vertical Mirror • 100b: Vertical & Horizontal Mirror • 101b: 90 Degree Rotate (Clockwise) • 110b: 180 Degree Rotate (Clockwise) • 111b: 270 Degree Rotate (Clockwise)
7	-	R/W	0x0	Reserved
6-5	ZF	R/W	0x0	Zero Fill • 00b: MSB Fill Mode Enable. • 01b: Zero Fill Mode Enable.

				• 1xb: HOB(High-Order Bits) Fill Mode Enable. This field applies only when Data Format is RGB Format, not RGB888/ ARGB8888. Refer to the above figure “Example: Input Source Image Data to RGB888/ARGB8888 Conversion” for details.
4-0	SDFRM	R/W	0x0	Source Data Format • 00000b: Separated YUV 4:4:4 • 00001b: Separated YUV 4:4:0 • 00010b: Separated YUV 4:2:2 • 00011b: Separated YUV 4:2:0 • 00100b: Separated YUV 4:1:1 • 00101b: Separated YUV 4:1:0 • 00110b: Interleaved YUV 4:2:2 • 00111b: Interleaved YUV 4:2:0 • 01000b: Sequential YUV 4:4:4 • 01001b: Sequential YUV 4:2:2 • 0101xb: RGB332 (8bpp) • 01100b: RGB444 • 01101b: ARGB4444 • 01110b: RGB454 • 01111b: ARGB3454 • 10000b: RGB555 • 10001b: ARGB1555 • 1001xb: RGB565 • 10100b: RGB666 • 10101b: ARGB4666 • 10110b: ARGB6666 • 10111b: RGB888 • 11000b: ARGB4888 • 11001b: ARGB8888 • else: Not used.

Source 0 Chroma-Key Parameter (S0_CHROMA) **0xB0070060**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CHROMA_RY[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHROMA_GU[7:0]								CHROMA_BV[7:0]							

Field	Name	RW	Reset	Description
31-24	-	R/W	0x0	Reserved
23-16	CHROMA_RY	R/W	0x0	Chroma-Key Value R or Y Chroma-Key Value
15-8	CHROMA_GU	R/W	0x0	Chroma-Key Value G or U Chroma-Key Value
7-0	CHROMA_BV	R/W	0x0	Chroma-Key Value B or V Chroma-Key Value

Chroma-Key Value should be set to Sequential YUV4:4:4 or RGB888 irrespective of Input Source Format.

Writing should be done in RGB888 Format when Input Format is RGBxxx. When Input Format is YUVxxx and YUVtoRGB Converter is disabled, YUV444 Format is used for Writing. If input format is YUVxxx and YUVtoRGB Converter is enabled, writing is done in RGB888 Format.

S0_CHROMA Register is effective only when S_CTRL.S0_CE is enabled and OP_CTRL.CSEL0/1 Register is 2bit “01”.

Source 0 Arithmetic Parameter (S0_PAR) 0xB0070064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PAR_RY[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAR_GU[7:0]								PAR_BV[7:0]							

Field	Name	RW	Reset	Description
31-24	-	R/W	0x0	Reserved
23-16	PAR_RY	R/W	0x0	Parameter Value for Arithmetic Operation R or Y Parameter Value for Arithmetic Operation.
15-8	PAR_GU	R/W	0x0	Parameter Value for Arithmetic Operation G or U Parameter Value for Arithmetic Operation.
7-0	PAR_BV	R/W	0x0	Parameter Value for Arithmetic Operation B or V Parameter Value for Arithmetic Operation.

Parameter Value should be set to Sequential YUV4:4:4 or RGB888 irrespective of Input Source Format.

Writing should be done in RGB888 Format when Input Format is RGBxxx. When Input Format is YUVxxx and YUVtoRGB Converter is disabled, YUV444 Format is used for Writing. If input format is YUVxxx and YUVtoRGB Converter is enabled, writing is done in RGB888 Format.

S0_PAR Register is effective only when SCTRL.S0ARITHMODE is Fill, Add, Sub and Multiply.

Source 1 Chroma-Key Parameter (S1_CHROMA) 0xB0070068

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CHROMA_RY[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHROMA_GU[7:0]								CHROMA_BV[7:0]							

Field	Name	RW	Reset	Description
31-24	-	R/W	0x0	Reserved
23-16	CHROMA_RY	R/W	0x0	Chroma-Key Value R or Y Chroma-Key Value
15-8	CHROMA_GU	R/W	0x0	Chroma-Key Value G or U Chroma-Key Value
7-0	CHROMA_BV	R/W	0x0	Chroma-Key Value B or V Chroma-Key Value

Chroma-Key Value should be set to Sequential YUV4:4:4 or RGB888 irrespective of Input Source Format.

Writing should be done in RGB888 Format when Input Format is RGBxxx. When Input Format is YUVxxx and YUVtoRGB Converter is disabled, YUV444 Format is used for Writing. If input format is YUVxxx and YUVtoRGB Converter is enabled, writing is done in RGB888 Format.

S1_CHROMA Register is effective only when S_CTRL.S1_CE is enabled and OP_CTRL.CSEL0/1 Register is 2bit “01”.

Source 1 Arithmetic Parameter (S1_PAR) 0xB007006C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PAR_RY[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAR_GU[7:0]								PAR_BV[7:0]							

Field	Name	RW	Reset	Description
31-24	-	R/W	0x0	Reserved
23-16	PAR_RY	R/W	0x0	Parameter Value for Arithmetic Operation R or Y Parameter Value for Arithmetic Operation.
15-8	PAR_GU	R/W	0x0	Parameter Value for Arithmetic Operation G or U Parameter Value for Arithmetic Operation.
7-0	PAR_BV	R/W	0x0	Parameter Value for Arithmetic Operation B or V Parameter Value for Arithmetic Operation.

Parameter Value should be set to Sequential YUV4:4:4 or RGB888 irrespective of Input Source Format.

Writing should be done in RGB888 Format when Input Format is RGBxxx. When Input Format is YUVxxx and YUVtoRGB Converter is disabled, YUV444 Format is used for Writing. If input format is YUVxxx and YUVtoRGB Converter is enabled, writing is done in RGB888 Format.

S1_PAR Register is effective only when SCTRL.S1ARITHMODE is Fill, Add, Sub and Multiply.

Source 2 Chroma-Key Parameter (S2_CHROMA) 0xB0070070

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CHROMA_RY[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHROMA_GU[7:0]								CHROMA_BV[7:0]							

Field	Name	RW	Reset	Description
31-24	-	R/W	0x0	Reserved
23-16	CHROMA_RY	R/W	0x0	Chroma-Key Value R or Y Chroma-Key Value
15-8	CHROMA_GU	R/W	0x0	Chroma-Key Value G or U Chroma-Key Value
7-0	CHROMA_BV	R/W	0x0	Chroma-Key Value B or V Chroma-Key Value

Chroma-Key Value should be set to Sequential YUV4:4:4 or RGB888 irrespective of Input Source Format.

Writing should be done in RGB888 Format when Input Format is RGBxxx. When Input Format is YUVxxx and YUVtoRGB Converter is disabled, YUV444 Format is used for Writing. If input format is YUVxxx and YUVtoRGB Converter is enabled, writing is done in RGB888 Format.

S2_CHROMA Register is effective only when S_CTRL.S2_CE is enabled and OP_CTRL.CSEL1 Register is 2bit “01”.

Source 2 Arithmetic Parameter (S2_PAR) 0xB0070074

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PAR_RY[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAR_GU[7:0]								PAR_BV[7:0]							

Field	Name	RW	Reset	Description
31-24	-	R/W	0x0	Reserved
23-16	PAR_RY	R/W	0x0	Parameter Value for Arithmetic Operation R or Y Parameter Value for Arithmetic Operation.
15-8	PAR_GU	R/W	0x0	Parameter Value for Arithmetic Operation G or U Parameter Value for Arithmetic Operation.
7-0	PAR_BV	R/W	0x0	Parameter Value for Arithmetic Operation B or V Parameter Value for Arithmetic Operation.

Parameter Value should be set to Sequential YUV4:4:4 or RGB888 irrespective of Input Source Format.

Writing should be done in RGB888 Format when Input Format is RGBxxx. When Input Format is YUVxxx and YUVtoRGB Converter is disabled, YUV444 Format is used for Writing. If input format is YUVxxx and YUVtoRGB Converter is enabled, writing is done in RGB888 Format.

S2_PAR Register is effective only when SCTRL.S2ARITHMODE is Fill, Add, Sub and Multiply.

Source Control (S_CTRL) 0xB0070078

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				S2ARITHMODE[2:0]			S1ARITHMODE[2:0]			S0ARITHMODE[2:0]			S2YE	S1YE	S0YE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Res.	S2YM[1:0]		S1YM[1:0]		S0YM[1:0]		S2CE	S1CE	S0CE	S2SEL[1:0]		S1SEL[1:0]		S0SEL[1:0]	

Field	Name	RW	Reset	Description
31-28	-	RW	0x0	Reserved
27-25	S2ARITHMODE	R/W	0x0	Source 2 Arithmetic Mode • 00xb: Bypass Mode • 010b: Fill Mode • 011b: Inverter Mode • 100b: Add Mode • 101b: Substrate Type A Mode • 110b: Substrate Type B Mode • 111b: Multiplier Mode Since Arithmetic Operation is done after YUVtoRGB Converting Operation, operation is working in RGB888 or YUV4:4:4 Format irrespective of Input Source Format. Therefore, writing should be done in RGB888 or YUV4:4:4 format. If Chroma-Key Value and Selected Source Data are matched when Chroma-Key is enabled, BYPASS Operation is carried out.
24-22	S1ARITHMODE	R/W	0x0	Source 1 Arithmetic Mode • 00xb: Bypass Mode • 010b: Fill Mode • 011b: Inverter Mode • 100b: Add Mode • 101b: Substrate Type A Mode

Arithmetic Fnc.	Equation	A	B
BYPASS	Y = A	Source Image	Do not Used
FILL	Y = B	Source Image	S0/1_PAR
INV	Y = 255 - A	Source Image	Do not Used
ADD	Y = A + B	Source Image	S0/1_PAR
SUBA	Y = A – B	Source Image	S0/1_PAR
SUBB	Y = B – A	Source Image	S0/1_PAR
MUL	Y = A x B Mantissa = B[7:6] Fraction = B[5:0]	Source Image	S0/1_PAR

				• 110b: Substrate Type B Mode • 111b: Multiplier Mode Refer to the above “S2ARITHMODE” descriptions.
21-19	S0ARITHMODE	R/W	0x0	Source 0 Arithmetic Mode • 00xb: Bypass Mode • 010b: Fill Mode • 011b: Inverter Mode • 100b: Add Mode • 101b: Substrate Type A Mode • 110b: Substrate Type B Mode • 111b: Multiplier Mode Refer to the above “S2ARITHMODE” descriptions.
18	S2YE	R/W	0x0	Source 2 YUV to RGB Converter Enable • 0: YUV to RGB Converter Disable • 1: YUV to RGB Converter Enable To enable YUV to RGB Converter, Data Format of Source 2 decided by S_CTRL.S2SEL[1:0] should be YUVxxx.
17	S1YE	R/W	0x0	Source 1 YUV to RGB Converter Enable • 0: YUV to RGB Converter Disable • 1: YUV to RGB Converter Enable To enable YUV to RGB Converter, Data Format of Source 1 decided by S_CTRL.S1SEL[1:0] should be YUVxxx.
16	S0YE	R/W	0x0	Source 0 YUV to RGB Converter Enable • 0: YUV to RGB Converter Disable • 1: YUV to RGB Converter Enable To enable YUV to RGB Converter, Data Format of Source 0 decided by S_CTRL.S0SEL[1:0] should be YUVxxx.
15	-	R/W	0x0	Reserved
14-13	S2YM	R/W	0x0	Source 2 YUVtoRGB Format Convert Type • 00b: YUVtoRGB Converter Type0 • 01b: YUVtoRGB Converter Type1 • 10b: YUVtoRGB Converter Type2 • 11b: YUVtoRGB Converter Type3 YUV4:4:4 to RGB888 Converter Type YUVtoRGB Format Converter Type 0 R = Y + 1.371 (V – 128) G = Y – 0.336 (U – 128) – 0.698 (V – 128) B = Y + 1.732 (U – 128) YUVtoRGB Format Converter Type 1 R = 1.164 (Y – 16) + 1.159 (V – 128) G = 1.164 (Y – 16) – 0.391 (U – 128) – 0.813 (V – 128) B = 1.164 (Y – 16) + 2.018 (U – 128) YUVtoRGB Format Converter Type 2 R = Y + 1.540 (V – 128) G = Y – 0.183 (U – 128) – 0.459 (V – 128) B = Y + 1.816 (U – 128) YUVtoRGB Format Converter Type 3 R = 1.164 (Y – 16) + 1.793 (V – 128) G = 1.164 (Y – 16) – 0.213 (U – 128) – 0.534 (V – 128) B = 1.164 (Y – 16) + 2.115 (U – 128)
12-11	S1YM	R/W	0x0	Source 1 YUVtoRGB Format Convert Type • 00b: YUVtoRGB Converter Type0 • 01b: YUVtoRGB Converter Type1 • 10b: YUVtoRGB Converter Type2 • 11b: YUVtoRGB Converter Type3 Refer to the above “YUV4:4:4 to RGB888 Converter Type” descriptions for details.
10-9	S0YM	R/W	0x0	Source 0 YUVtoRGB Format Convert Type • 00b: YUVtoRGB Converter Type0

				• 01b: YUVtoRGB Converter Type1 • 10b: YUVtoRGB Converter Type2 • 11b: YUVtoRGB Converter Type3 Refer to the above “YUV4:4:4 to RGB888 Converter Type” descriptions for details.
8	S2CE	R/W	0x0	Source 2 Chroma-Key Enable for Arithmetic • 0: Chroma-Key Disable for Arithmetic • 1: Chroma-Key Enable for Arithmetic
7	S1CE	R/W	0x0	Source 1 Chroma-Key Enable for Arithmetic • 0: Chroma-Key Disable for Arithmetic • 1: Chroma-Key Enable for Arithmetic
6	S0CE	R/W	0x0	Source 0 Chroma-Key Enable for Arithmetic • 0: Chroma-Key Disable for Arithmetic • 1: Chroma-Key Enable for Arithmetic
5-4	S2SEL	R/W	0x0	Source 2 Selection • 00b: Source 2 Disable • 01b: Source 2 = Front-End Channel 0 • 10b: Source 2 = Front-End Channel 1 • 11b: Source 2 = Front-End Channel 2 Source 0 should select whatever image. In other words, if there is only one Input Source Image in either FCH0 or FCH1 or FCH2, the Input Source Image should be selected to Source 0. If there are more than 2 Input Source Images in two of FCH0, FCH1 and FCH2, the larger Input Source Image should be selected to S0 and the other should be selected to either S1 or S2.
3-2	S1SEL	R/W	0x0	Source 1 Selection • 00b: Source 1 Disable • 01b: Source 1 = Front-End Channel 0 • 10b: Source 1 = Front-End Channel 1 • 11b: Source 1 = Front-End Channel 2 Refer to the above “S2SEL” descriptions.
1-0	S0SEL	R/W	0x0	Source 0 Selection • 00b: Source 0 Disable • 01b: Source 0 = Front-End Channel 0 • 10b: Source 0 = Front-End Channel 1 • 11b: Source 0 = Front-End Channel 2 Refer to the above “S2SEL” descriptions.

Source Operator 0 Pattern (OP0_PAT) 0xB0070080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ALPHA[7:0]								PAT_RY[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAT_GU[7:0]								PAT_BV[7:0]							

Field	Name	RW	Reset	Description
31-24	ALPHA	R/W	0x0	Source Operator 0 Alpha Value Alpha Value for Alpha-Blending Operation of Source Operator 0 This field is effective only when OPCTRL.OP0MODE is Alpha-Blending Type 0.
23-16	PAT_RY	R/W	0x0	Source Operator 0 Pattern Value R or Y Pattern Value for Raster Operation of Source Operator 0
15-8	PAT_GU	R/W	0x0	Source Operator 0 Pattern Value G or U Pattern Value for Raster Operation of Source Operator 0
7-0	PAT_BV	R/W	0x0	Source Operator 0 Pattern Value B or V Pattern Value for Raster Operation of Source Operator 0

Source Operator 1 Pattern (OP1_PAT) 0xB0070084

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ALPHA[7:0]								PAT_RY[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PAT_GU[7:0]								PAT_BV[7:0]							

Field	Name	RW	Reset	Description
31-24	ALPHA	R/W	0x0	Source Operator 1 Alpha Value Alpha Value for Alpha-Blending Operation of Source Operator 1 This field is effective only when OPCTRL.OP1MODE is Alpha-Blending Type 0.
23-16	PAT_RY	R/W	0x0	Source Operator 1 Pattern Value R or Y Pattern Value for Raster Operation of Source Operator 1
15-8	PAT_GU	R/W	0x0	Source Operator 1 Pattern Value G or U Pattern Value for Raster Operation of Source Operator 1
7-0	PAT_BV	R/W	0x0	Source Operator 1 Pattern Value B or V Pattern Value for Raster Operation of Source Operator 1

Source Operation Control (OP_CTRL) 0xB0070088

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							ASEL1[1:0]		CSEL1[1:0]		OP1_MODE[4:0]				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							ASEL0[1:0]		CSEL0[1:0]		OP0_MODE[4:0]				

Field	Name	RW	Reset	Description
31-25	-	R/W	0x0	Reserved
24-23	ASEL1	R/W	0x0	Alpha Value Selection 1 for Alpha-Blending • 0xb: Alpha Value is 0 ~ 255 (0% ~ 99.6%) • 10b: Alpha Value is 1 ~ 256 (0.39% ~ 100%) • 11b: Alpha Value is 0 ~ 127 and 129 ~ 256 (0% ~ 49.6% and 50.39% ~ 100%)
22-21	CSEL1	R/W	0x0	Chroma-key Source Selection • 00b: Chroma-Key Operation Disable of Operator 1 • 01b: Chroma-Key Source = Source 0, Chroma-Key Value = S0_CHROMA • 10b: Chroma-Key Source = Source 1, Chroma-Key Value = S1_CHROMA • 11b: Chroma-Key Source = Source 2, Chroma-Key Value = S2_CHROMA
20-16	OP1_MODE	R/W	0x0	Operation 1 Mode **Operation 1 Mode** 00000b Blackness Result = 0 — Raster Operation 00001b Merged Copy Result = P & A — Raster Operation 00010b Merged Paint Result = ~ A | B — Raster Operation 00011b Pattern Copy Result = P — Raster Operation 00100b Pattern Inverter Result = P ^ B — Raster Operation 00101b Pattern Paint Result = (P | ~ A) | B — Raster Operation 00110b Source Copy Result = A — Raster Operation 00111b Source Inverter Result = A ^ B — Raster Operation 01000b Source Paint Result = A | B — Raster Operation 01001b Source AND Result = A & B — Raster Operation 01010b Source Erase Result = A & ~ B — Raster Operation 01011b Not Source Copy Result = ~ A — Raster Operation 01100b Not Source Erase Result = ~ (A | B) — Raster Operation 01101b Destination Copy Result = B — Raster Operation 01110b Destination Inverter Result = ~ B — Raster Operation 01111b Whiteness Result = 1 — Raster Operation 10xxxb Alpha-Blending Type 0 Alpha-Blending with Alpha-Value — Alpha-Blending 11xxxb Alpha-Blending Type 1 Alpha-Blending with Alpha-Value of ARGB — Alpha-Blending
15-9	-	RW	0x0	Reserved
8-7	ASEL0	R/W	0x0	Alpha Value Selection 0 for Alpha-Blending • 0xb: Alpha Value is 0 ~ 255 (0% ~ 99.6%) • 10b: Alpha Value is 1 ~ 256 (0.39% ~ 100%)

<table>
<tr><td></td><td></td><td></td><td></td><td>• 11b: Alpha Value is 0 ~ 127 and 129 ~ 256 (0% ~ 49.6% and 50.39% ~ 100%)</td></tr>
<tr><td>6-5</td><td>CSEL0</td><td>R/W</td><td>0x0</td><td>Chroma-key Source Selection
• 00b: Chroma-Key Operation Disable of Operator 0
• 01b: Chroma-Key Source = Source 0, Chroma-Key Value = S0_CHROMA
• 10b: Chroma-Key Source = Source 1, Chroma-Key Value = S1_CHROMA
• 11b: Chroma-Key Source = Source 2, Chroma-Key Value = S2_CHROMA</td></tr>
<tr><td>4-0</td><td>OP0_MODE</td><td>R/W</td><td>0x0</td><td>Operation 1 Mode
<table>
<tr><th colspan="4">Operation 0 Mode</th></tr>
<tr><td>00000b</td><td>Blackness</td><td>Result = 0</td><td rowspan="14">Raster Operation</td></tr>
<tr><td>00001b</td><td>Merged Copy</td><td>Result = P & A</td></tr>
<tr><td>00010b</td><td>Merged Paint</td><td>Result = ~ A | B</td></tr>
<tr><td>00011b</td><td>Pattern Copy</td><td>Result = P</td></tr>
<tr><td>00100b</td><td>Pattern Inverter</td><td>Result = P ^ B</td></tr>
<tr><td>00101b</td><td>Pattern Paint</td><td>Result = (P | ~ A) | B</td></tr>
<tr><td>00110b</td><td>Source Copy</td><td>Result = A</td></tr>
<tr><td>00111b</td><td>Source Inverter</td><td>Result = A ^ B</td></tr>
<tr><td>01000b</td><td>Source Paint</td><td>Result = A | B</td></tr>
<tr><td>01001b</td><td>Source AND</td><td>Result = A & B</td></tr>
<tr><td>01010b</td><td>Source Erase</td><td>Result = A & ~ B</td></tr>
<tr><td>01011b</td><td>Not Source Copy</td><td>Result = ~ A</td></tr>
<tr><td>01100b</td><td>Not Source Erase</td><td>Result = ~ (A | B)</td></tr>
<tr><td>01101b</td><td>Destination Copy</td><td>Result = B</td></tr>
<tr><td>01110b</td><td>Destination Inverter</td><td>Result = ~ B</td></tr>
<tr><td>01111b</td><td>Whiteness</td><td>Result = 1</td></tr>
<tr><td>10xxxb</td><td>Alpha-Blending Type 0</td><td>Alpha-Blending with Alpha-Value</td><td rowspan="2">Alpha-Blending</td></tr>
<tr><td>11xxxb</td><td>Alpha-Blending Type 1</td><td>Alpha-Blending with Alpha-Value of ARGB</td></tr>
</table></td></tr>
</table>

Raster Operation & Alpha-Blending of Operator 0

Mode	Equation	A	B	P	ALPHA	CHROMA
Blackness	Y = 0	Source 0	Not Used.	Not Used.	Not Used.	S1_CHROMA
Merged Copy	Y = P & A	Source 0	Not Used.	PAT0[23:00]	Not Used.	S1_CHROMA
Merged Paint	Y = ~ A | B	Source 0	Source 1	Not Used.	Not Used.	S1_CHROMA
Pattern Copy	Y = P	Source 0	Not Used.	PAT0[23:00]	Not Used.	S1_CHROMA
Pattern Inverter	Y = P ^ B	Source 0	Source 1	PAT0[23:00]	Not Used.	S1_CHROMA
Pattern Paint	Y = (P | ~ A) | B	Source 0	Source 1	PAT0[23:00]	Not Used.	S1_CHROMA
Source Copy	Y = A	Source 0	Not Used.	Not Used.	Not Used.	S1_CHROMA
Source Inverter	Y = A ^ B	Source 0	Source 1	Not Used.	Not Used.	S1_CHROMA
Source Paint	Y = A | B	Source 0	Source 1	Not Used.	Not Used.	S1_CHROMA
Source AND	Y = A & B	Source 0	Source 1	Not Used.	Not Used.	S1_CHROMA
Source Erase	Y = A & ~ B	Source 0	Source 1	Not Used.	Not Used.	S1_CHROMA
Not Source Copy	Y = ~ A	Source 0	Not Used.	Not Used.	Not Used.	S1_CHROMA
Not Source Erase	Y = ~ (A | B)	Source 0	Source 1	Not Used.	Not Used.	S1_CHROMA
Destination Copy	Y = B	Source 0	Source 1	Not Used.	Not Used.	S1_CHROMA
Destination Inverter	Y = ~ B	Source 0	Source 1	Not Used.	Not Used.	S1_CHROMA
Whiteness	Y = 1	Source 0	Not Used.	Not Used.	Not Used.	S1_CHROMA
Alpha-Blending Type 0	Y = ALPHA * B + (1 – ALPHA) *A	Source 0	Source 1	Not Used.	OP0_ALPHA	S1_CHROMA
Alpha-Blending Type 1	Y = ALPHA * B + (1 – ALPHA) *A	Source 0	Source 1	Not Used.	Alpha of Source 1	S1_CHROMA

In Alpha-Blending Type 0, OP0_ALPHA of OP_PAT0 Register is used as Alpha Value.

In Alpha-Blending Type 1, Alpha Value of Source 1 Data is used as Alpha Value. Therefore, in this case, Source Data Format should definitely be ARGB.

Raster Operation & Alpha-Blending of Operator 1

Mode	Equation	A	B	P	ALPHA	CHROMA
Blackness	Y = 0	Y of OP 0	Not Used.	Not Used.	Not Used.	S2_CHROMA
Merged Copy	Y = P & A	Y of OP 0	Not Used.	PAT1[23:00]	Not Used.	S2_CHROMA
Merged Paint	Y = ~ A | B	Y of OP 0	Source 2	Not Used.	Not Used.	S2_CHROMA
Pattern Copy	Y = P	Y of OP 0	Not Used.	PAT1[23:00]	Not Used.	S2_CHROMA
Pattern Inverter	Y = P ^ B	Y of OP 0	Source 2	PAT1[23:00]	Not Used.	S2_CHROMA
Pattern Paint	Y = (P | ~ A) | B	Y of OP 0	Source 2	PAT1[23:00]	Not Used.	S2_CHROMA
Source Copy	Y = A	Y of OP 0	Not Used.	Not Used.	Not Used.	S2_CHROMA
Source Inverter	Y = A ^ B	Y of OP 0	Source 2	Not Used.	Not Used.	S2_CHROMA
Source Paint	Y = A | B	Y of OP 0	Source 2	Not Used.	Not Used.	S2_CHROMA
Source AND	Y = A & B	Y of OP 0	Source 2	Not Used.	Not Used.	S2_CHROMA
Source Erase	Y = A & ~ B	Y of OP 0	Source 2	Not Used.	Not Used.	S2_CHROMA
Not Source Copy	Y = ~ A	Y of OP 0	Not Used.	Not Used.	Not Used.	S2_CHROMA
Not Source Erase	Y = ~ (A | B)	Y of OP 0	Source 2	Not Used.	Not Used.	S2_CHROMA
Destination Copy	Y = B	Y of OP 0	Source 2	Not Used.	Not Used.	S2_CHROMA
Destination Inverter	Y = ~ B	Y of OP 0	Source 2	Not Used.	Not Used.	S2_CHROMA
Whiteness	Y = 1	Y of OP 0	Not Used.	Not Used.	Not Used.	S2_CHROMA
Alpha-Blending Type 0	Y = ALPHA * B + (1 – ALPHA) *A	Y of OP 0	Source 2	Not Used.	OP1_ALPHA	S2_CHROMA
Alpha-Blending Type 1	Y = ALPHA * B + (1 – ALPHA) *A	Y of OP 0	Source 2	Not Used.	Alpha of Source 2	S2_CHROMA

In Alpha-Blending Type 0, OP0_ALPHA of OP_PAT0 Register is used as Alpha Value.

In Alpha-Blending Type 1, Alpha Value of Source 1 Data is used as Alpha Value. Therefore, in this case, Source Data Format should definitely be ARGB.

Back-End Destination Address 0 (BCH_DADDR0) 0xB0070090

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DADDR0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDR0[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	DADDR0	R/W	0x0	Destination Address 0 • N (0~ 4095) This register determines the Y image base address of BCH when the data format of BCH is separated YUV or interleaved YUV format. But, in case the data format of BCH is a sequential YUV or an RGB, it determines the base address of the whole image. The data format of BCH is determined by the FCH0_SCTRL.DDFRM bits.
1-0	-	R/W	0x0	Reserved

Back-End Destination Address 1 (BCH_DADDR1) 0xB0070094

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DADDR1[31:16]															
9	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DADDR1[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	DADDR1	R/W	0x0	Destination Address 1 • N (0~ 4095) This register determines the U image base address of BCH when the data format of BCH is separated YUV or interleaved YUV format. But, in case the data format of BCH image is a sequential YUV or an RGB, it is not used.
1-0	-	R/W	0x0	Reserved

Back-End Destination Address 2 (BCH_DADDR2) 0xB0070098

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DADDR2[31:16]															
9	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DADDR2[15:2]														Reserved	

Field	Name	RW	Reset	Description
31-2	DADDR2	R/W	0x0	Destination Address 2 • N (0~ 4095) This register determines the V image base address of BCH image when the data format of BCH is separated YUV format. But, in case the data format of BCH image is a sequential YUV, interleaved YUV or an RGB, it is not used.
1-0	-	R/W	0x0	Reserved

Back-End Channel Destination Frame Pixel Size (BCH_DFSIZE) 0xB007009C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				DFSIZE_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				DFSIZE_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	DFSIZE_Y	R/W	0x0	Destination Frame Pixel Size Y • N (0~ 4095) Height of BCH image by pixel units
15-12	-	R/W	0x0	Reserved
11-0	DFSIZE_X	R/W	0x0	Destination Frame Pixel Size X • N (0~ 4095) Width of BCH image by pixel units

Back-End Channel Destination Pixel Offset (BCH_DOFF) 0xB00700A0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				DOFF_Y[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				DOFF_X[11:0]											

Field	Name	RW	Reset	Description
31-28	-	R/W	0x0	Reserved
27-16	DOFF_Y	R/W	0x0	Destination Pixel Offset Y • N (0~ 4095) Y Axis Destination Pixel Offset
15-12	-	R/W	0x0	Reserved
11-0	DOFF_X	R/W	0x0	Destination Pixel Offset X • N (0~ 4095) X Axis Destination Pixel Offset

Destination Image Size & Frame Size & Offset Limitation

BCH_DCTRL.DDFRM	Destination Frame Size / Image Size / Offset Limitation
RGB332(8bpp)	No Limitation
RGB454	No Limitation
RGB565	No Limitation
RGB666	No Limitation
RGB888	No Limitation
Separated YUV 4:4:4	No Limitation
Separated YUV 4:4:0	X axis = No Limitation, Y axis = multiplies of 2
Separated YUV 4:2:2	X axis = multiplies of 2, Y axis = No Limitation
Separated YUV 4:2:0	X axis = multiplies of 2, Y axis = multiplies of 2
Separated YUV 4:1:1	X axis = multiplies of 4, Y axis = No Limitation
Separated YUV 4:1:0	X axis = multiplies of 4, Y axis = multiplies of 2
Interleaved YUV 4:2:2	X axis = multiplies of 2, Y axis = No Limitation
Interleaved YUV 4:2:0	X axis = multiplies of 2, Y axis = multiplies of 2
Sequential YUV 4:4:4	No Limitation
Sequential YUV 4:2:2	X axis = multiplies of 2, Y axis = No Limitation

Destination Image Size is decided by both Source 0 size controlled by S0_CTRL and Back-End Destination Operation Mode controlled by BCH_CTRL.

Back-End Channel Control (BCH_DCTRL) 0xB00700A4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					DSB[2:0]			Reserved		MABC	Reserved		YSEL	XSEL[1:0]	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R2Y	R2YM[1:0]		Res.	DSUV	OPMODE[2:0]			0	DOP	DEN	DDFRM[4:0]				

Field	Name	RW	Reset	Description
31-27	-	RW	0x0	Reserved
26-24	DSB	R/W	0x0	Destination Swap Byte • 000b: ARGB • 001b: ARBG • 010b: AGRB • 011b: AGBR • 100b: ABRG • 101b: ABGR • 110b: RABG • 111b: BGRA This field applies only when Data Format is RGB Format. In case of 110b or 111b, the operations using Alpha Value are not supported.
23-22	-	R/W	0x0	Reserved
21	MABC	R/W	0x0	AXI Bus Master Address Boundary Control • 0: AXI bus master address boundary is set 4KByte. • 1: AXI bus master address boundary is set 1KByte.
20-19	-	R/W	0x0	Reserved
18	YSEL	R/W	0x0	YUV4:4:4 to YUVx:x:x Y Control

				• N (0 ~ 1) Since inside Overlay Mixer, only RGB888 and Sequential YUV4:4:4 exist as Data Format, if Destination Data Format is RGBxxx, RGB888 is converted to RGBxxx. If Destination Data Format is Sequential / Separated YUVxxx, Sequential YUV4:4:4 is converted to Sequential / Separated YUVxxx. In this case, when Sequential YUV4:4:4 is converted to Sequential / Separated YUVxxx, BCH_DCTRL.X/YSEL Register is reflected.
17-16	XSEL	R/W	0x0	YUV4:4:4 to YUVx:x:x X Control • N (0 ~ 3) Refer to the above “YSEL” descriptions.
15	R2Y	R/W	0x0	Destination Format Converter Control • 0: RGB to YUV Converter Disable • 1: RGB to YUV Converter Enable
14-13	R2YM	R/W	0x0	RGBtoYUV Converter Type • 00b: RGBtoYUV Converter Type0 • 01b: RGBtoYUV Converter Type1 • 10b: RGBtoYUV Converter Type2 • 11b: RGBtoYUV Converter Type3 RGB to YUV Converter Type RGBtoYUV Format Converter Type 0 Y = 0.299 R + 0.587 G + 0.114 B U = -0.172 R – 0.339 G + 0.511 B + 128 V = 0.511 R – 0.428 G – 0.083 B + 128 RGBtoYUV Format Converter Type 1 Y = 0.257 R + 0.504 G + 0.098 B + 16 U = -0.148 R – 0.291 G + 0.439 B + 128 V = 0.439 R – 0.368 G – 0.071 B + 128 RGBtoYUV Format Converter Type 2 Y = 0.213 R + 0.715 G + 0.072 B U = -0.117 R – 0.394 G + 0.511 B + 128 V = 0.511 R – 0.464 G – 0.047 B + 128 RGBtoYUV Format Converter Type 3 Y = 0.183 R + 0.614 G + 0.062 B + 16 U = -0.101 R – 0.338 G + 0.439 B + 128 V = 0.439 R – 0.399 G – 0.040 B + 128
12	-	R/W	0x0	Reserved
11	DSUV	R/W	0x0	Destination Swap U for V • 0: Keep on U and V. (LSB = U, MSB = V) • 1: Swap U for V. (LSB = V, MSB = U) This filed applies only when Data Format is Interleaved YUV Format.
10-8	OPMODE	R/W	0x0	Operation Mode of Back-End DMA • 00xb: Data Copy • 010b: Horizontal Mirror • 011b: Vertical Mirror • 100b: Vertical & Horizontal Mirror • 101b: 90 Degree Rotate (Clockwise) • 110b: 180 Degree Rotate (Clockwise) • 111b: 270 Degree Rotate (Clockwise)
7	-	R/W	0x0	Reserved
6	DOP	R/W	0x0	Dithering Operation Type • 0: 1-Bit Toggle Operation • 1: Add 1 Operation This field applies only when Data Format is RGB Format, not RGB888 / ARGB8888 Format.
5	DEN	R/W	0x0	Dithering Enable • 0: Dithering Enable • 1: Dithering Disable
4-0	DDFRM	R/W	0x0	Source Data Format

				• 00000b: Separated YUV 4:4:4 • 00001b: Separated YUV 4:4:0 • 00010b: Separated YUV 4:2:2 • 00011b: Separated YUV 4:2:0 • 00100b: Separated YUV 4:1:1 • 00101b: Separated YUV 4:1:0 • 00110b: Interleaved YUV 4:2:2 • 00111b: Interleaved YUV 4:2:0 • 01000b: Sequential YUV 4:4:4 • 01001b: Sequential YUV 4:2:2 • 0101xb: RGB332 (8bpp) • 01100b: RGB444 • 01101b: ARGB4444 • 01110b: RGB454 • 01111b: ARGB3454 • 10000b: RGB555 • 10001b: ARGB1555 • 1001xb: RGB565 • 10100b: RGB666 • 10101b: ARGB4666 • 10110b: ARGB6666 • 10111b: RGB888 • 11000b: ARGB4888 • 11001b: ARGB8888 • else: Not used.

Back-End Channel Destination Dither Matrix 0 (BCH_DDMAT0) 0xB00700B0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			DDMAT03[4:0]					Reserved			DDMAT02[4:0]				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			DDMAT01[4:0]					Reserved			DDMAT00[4:0]				

Field	Name	RW	Reset	Description
31-29	-	R/W	0x0	Reserved
28-24	DDMAT03	R/W	0x0	Destination Dither Matrix Element 3
23-21	-	R/W	0x0	Reserved
20-16	DDMAT02	R/W	0x0	Destination Dither Matrix Element 2
15-13	-	R/W	0x0	Reserved
12-8	DDMAT01	R/W	0x0	Destination Dither Matrix Element 1
7-5	-	R/W	0x0	Reserved
4-0	DDMAT00	R/W	0x0	Destination Dither Matrix Element 0

Back-End Channel Destination Dither Matrix 1 (BCH_DDMAT1) 0xB00700B4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			DDMAT07[4:0]					Reserved			DDMAT06[4:0]				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			DDMAT05[4:0]					Reserved			DDMAT04[4:0]				

Field	Name	RW	Reset	Description
31-29	-	R/W	0x0	Reserved
28-24	DDMAT07	R/W	0x0	Destination Dither Matrix Element 7
23-21	-	R/W	0x0	Reserved
20-16	DDMAT06	R/W	0x0	Destination Dither Matrix Element 6
15-13	-	R/W	0x0	Reserved
12-8	DDMAT05	R/W	0x0	Destination Dither Matrix Element 5
7-5	-	R/W	0x0	Reserved
4-0	DDMAT04	R/W	0x0	Destination Dither Matrix Element 4

Back-End Channel Destination Dither Matrix 2 (BCH_DDMAT2) 0xB00700B8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			DDMAT11[4:0]					Reserved			DDMAT10[4:0]				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DDMAT09[4:0]					Reserved			DDMAT08[4:0]				

Field	Name	RW	Reset	Description
31-29	-	R/W	0x0	Reserved
28-24	DDMAT11	R/W	0x0	Destination Dither Matrix Element 11
23-21	-	R/W	0x0	Reserved
20-16	DDMAT10	R/W	0x0	Destination Dither Matrix Element 10
15-13	-	R/W	0x0	Reserved
12-8	DDMAT09	R/W	0x0	Destination Dither Matrix Element 9
7-5	-	R/W	0x0	Reserved
4-0	DDMAT08	R/W	0x0	Destination Dither Matrix Element 8

Back-End Channel Destination Dither Matrix 3 (BCH_DDMAT3) 0xB00700BC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			DDMAT15[4:0]					Reserved			DDMAT14[4:0]				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			DDMAT13[4:0]					Reserved			DDMAT12[4:0]				

Field	Name	RW	Reset	Description
31-29	-	R/W	0x0	Reserved
28-24	DDMAT15	R/W	0x0	Destination Dither Matrix Element 15
23-21	-	R/W	0x0	Reserved
20-16	DDMAT14	R/W	0x0	Destination Dither Matrix Element 14
15-13	-	R/W	0x0	Reserved
12-8	DDMAT13	R/W	0x0	Destination Dither Matrix Element 13
7-5	-	R/W	0x0	Reserved
4-0	DDMAT12	R/W	0x0	Destination Dither Matrix Element 12

These BCH_DDMAT registers are for setting dither matrix values for dithering. The patterned dithering algorithm using 4x4 dither matrix is used for dithering in Overlay Mixer. Each BCH_DDMAT 0/1/2/3 register sets 4 values in one row of the matrix. The below figure shows the relation between 4x4 dither matrix and BCH_DDMAT register and it provides simple setting value examples.

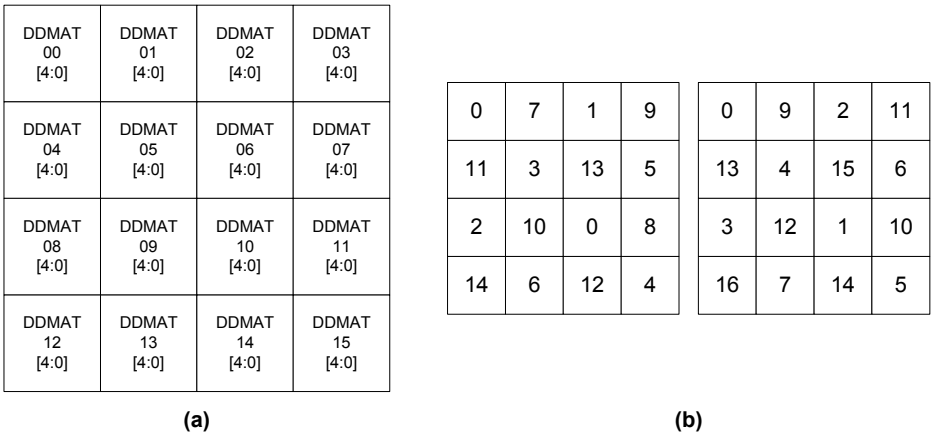


Figure 9.7 (a) 16 Level 4x4 Dither Matrix (b) Example of Dither Matrix Setting

Overlay Mixer Control (OM_CTRL) 0xB00700C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															IEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							D32M	Reserved					EN[2:0]		

Field	Name	RW	Reset	Description
31-17	-	R/W	0x0	Reserved
16	IEN	R/W	0x0	Overlay Mixer Interrupt Enable • 0: Interrupt Enable • 1: Interrupt Disable
15-9	-	R/W	0x0	Reserved
8	D32M	R/W	0x0	Overlay Mixer DMA 32-bit Mode • 0: Front-End / Back-End DMA Enable 64-bit Mode • 1: Front-End / Back-End DMA Enable 32-bit Mode
7-3	-	R/W	0x0	Reserved
2-0	EN	R/W	0x0	Overlay Mixer Enable • 000b: Overlay Mixer Disable • 001b: Overlay Mixer Enable With Front-End Channel 0 • 010b: Overlay Mixer Enable With Front-End Channel 1 • 011b: Overlay Mixer Enable With Front-End Channel 2 • 100b: Overlay Mixer Enable With Front-End Channel 0,1 • 101b: Overlay Mixer Enable With Front-End Channel 0,2 • 110b: Overlay Mixer Enable With Front-End Channel 1,2 • 111b: Overlay Mixer Enable With Front-End Channel 0,1,2 After all transfers complete, this register filed is automatically cleared to 3bit "000". This field should be set last after all Control Registers are set.

Overlay Mixer Interrupt Request Control (OM_IREQ) 0xB00700C4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															FLG
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved															IRQ

Field	Name	RW	Reset	Description
31-17	-	R/W	0x0	Reserved
16	FLG	R/W	0x0	Overlay Mixer Flag Bit • 0(Read): Transfer of Overlay Mixer is Not Completed. • 0(Write): Flag Bit Clear • 1: Transfer of Overlay Mixer is Completed.
15-1	-	R/W	0x0	Reserved
0	IRQ	R/W	0x0	Overlay Mixer Interrupt Request • 0(Read): Interrupt Request is not occurred. • 0(Write): Interrupt Request is cleared. • 1: Interrupt Request is occurred.

Front-End Channel 0 Lookup Table (FCH0_LUT) 0xB0070400 ~ 0xB00707FF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FCH0_LUTA[7:0]								FCH0_LUTR[7:0]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FCH0_LUTG[7:0]								FCH0_LUTB[7:0]							

Field	Name	RW	Reset	Description
31-24	FCH0_LUTA	R/W	0x0	Alpha of FCH0 Lookup Table This field should be use when FCH0 image format is ARGB format of RGB332(8bpp).
23-16	FCH0_LUTR	R/W	0x0	R of FCH0 Lookup Table This field should be use when FCH0 image format is RGB332(8bpp).
15-8	FCH0_LUTG	R/W	0x0	G of FCH0 Lookup Table This field should be use when FCH0 image format is RGB332(8bpp).
7-0	FCH0_LUTB	R/W	0x0	B of FCH0 Lookup Table This field should be use when FCH0 image format is RGB332(8bpp).

When Overlay Mixer is operating, CPU can't write and read this register.

Front-End Channel 1 Lookup Table (FCH1_LUT) 0xB0070800 ~ 0xB0070BFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FCH1_LUTA[7:0]								FCH1_LUTR[7:0]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FCH1_LUTG[7:0]								FCH1_LUTB[7:0]							

Field	Name	RW	Reset	Description
31-24	FCH1_LUTA	R/W	0x0	Alpha of FCH1 Lookup Table This field should be use when FCH0 image format is ARGB format of RGB332(8bpp).
23-16	FCH1_LUTR	R/W	0x0	R of FCH1 Lookup Table This field should be use when FCH1 image format is RGB332(8bpp).
15-8	FCH1_LUTG	R/W	0x0	G of FCH1 Lookup Table This field should be use when FCH1 image format is RGB332(8bpp).
7-0	FCH1_LUTB	R/W	0x0	B of FCH1 Lookup Table This field should be use when FCH1 image format is RGB332(8bpp).

When Overlay Mixer is operating, CPU can't write and read this register.

Front-End Channel 2 Lookup Table (FCH2_LUT) 0xB0070C00 ~ 0xB0070FFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FCH2_LUTA[7:0]								FCH2_LUTR[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FCH2_LUTG[7:0]								FCH2_LUTB[7:0]							

Field	Name	RW	Reset	Description
31-24	FCH2_LUTA	R/W	0x0	Alpha of FCH2 Lookup Table This field should be use when FCH0 image format is ARGB format of RGB332(8bpp).
23-16	FCH2_LUTR	R/W	0x0	R of FCH2 Lookup Table This field should be use when FCH1 image format is RGB332(8bpp).
15-8	FCH2_LUTG	R/W	0x0	G of FCH2 Lookup Table This field should be use when FCH1 image format is RGB332(8bpp).
7-0	FCH2_LUTB	R/W	0x0	B of FCH2 Lookup Table This field should be use when FCH1 image format is RGB332(8bpp).

When Overlay Mixer is operating, CPU can't write and read this register.

Use of LUT

FCHx_SCTRL.SDFRM	Supported LUT Field (ARGB8888)			
	A	R	G	B
RGB332(8bpp)	Yes	Yes	Yes	Yes
ARGB4444	Yes	No	No	No
ARGB3454	Yes			
ARGB1555	Yes			
ARGB4666	Yes			
ARGB6666	Yes			
ARGB4888	Yes			
ARGB8888	Yes			
RGB444	No	No	No	No
RGB454				
RGB555				
RGB565				
RGB666				
RGB888				
Separated YUV x:x:x				
Interleaved YUV x:x:x				

9.4 Overlay Mixer Operation

9.4.1 Source Rotate / Mirror and Destination Rotate / Mirror Operation

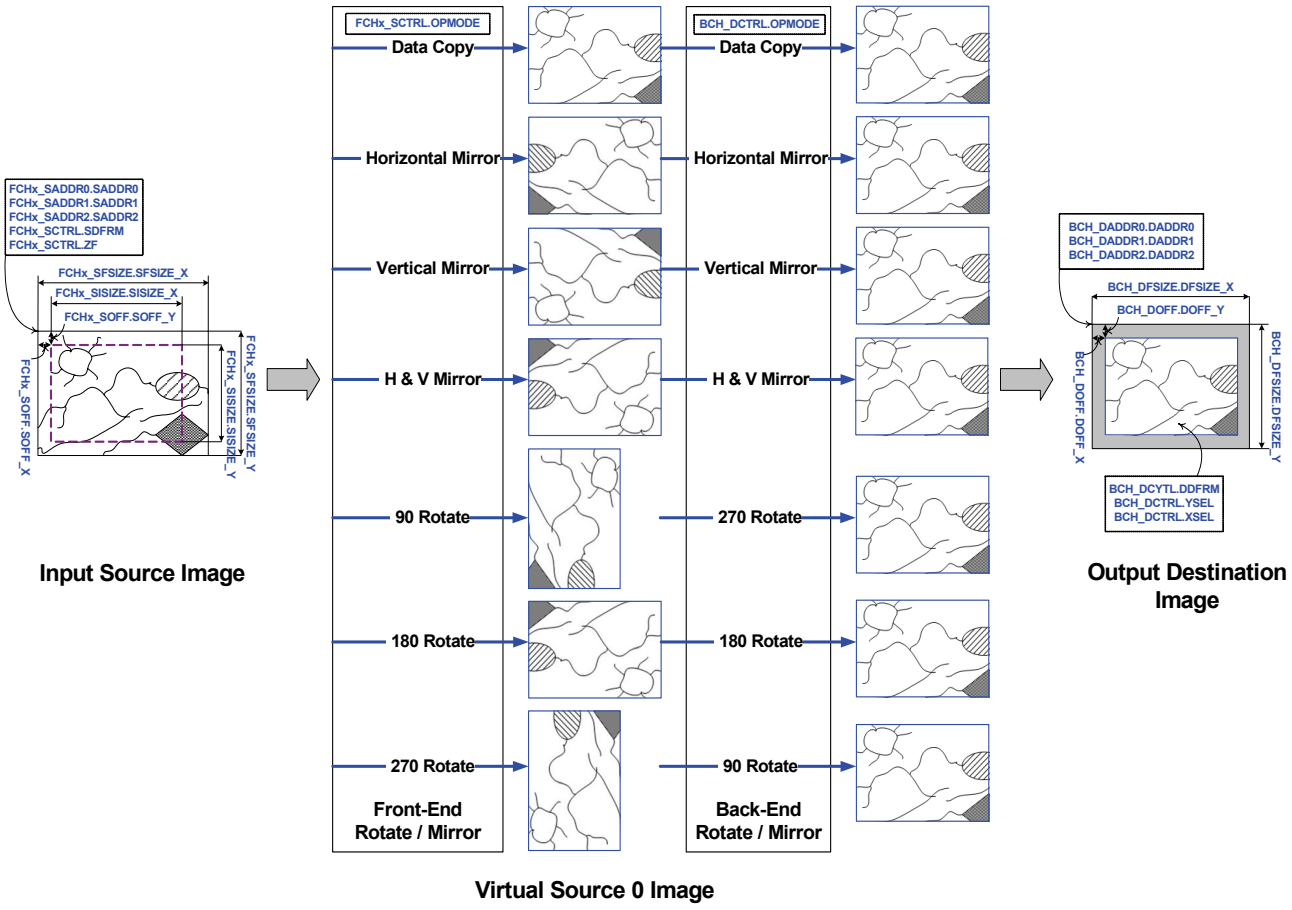


Figure 9.8 Source & Destination Mirror / Rotate Operation

Figure 9.8 illustrates one of functions supported by Overlay Mixer, Rotate / Mirror function. The figure shows Internal Register Setting when there is one Input Source Image and Input Image Source is FCH1.

As explained before, if the number of an image is One, the image should be selected to Source 0. Window Offset of the image which is selected to Source 0 should definitely be '0'. In addition to this, OM_CTRL.EN Register should be set last.

(1) FCH1_SADDR0 / 1 / 2 : The Input Source Image Start Address.
(2) FCH1_SCTRL.SDFRM, FCH1_SCTRL.ZF : The Input Source Image Data Format
(3) FCH1_SFSIZE : The Input Source Image Frame Size
(4) FCH1_SISIZE : The Input Source Image Size
(5) FCH1_SOFF : The Input Source Image Offset
(6) FCH1_SCTRL.OPMODE : FCH1 Operation Mod
(7) FCH1_WOFF : Should be Zero.
(8) S_CTRL.S0SEL : Should be Front-End Channel 1
(9) S_CTRL.S0_ARITHMOD : Should be Bypass Mode.
(10) S_CTRL.S0_YE, S0_CE : Should be Disable.
(11) OP_CTRL.CSEL0, CSEL1 : Should be Disable.
(12) OP_CTRL.OP0MODE, OP1MODE: Should be Source Copy
(13) BCH_DCTRL.OPMODE : BCH Operation Mode.
(14) BCH_DCTRL.Y2R : Should be Disable.
(15) BCH_DADDR0 / 1 / 2 : The Output Destination Image Start Address.

(16) BCH_DCTRL.DDFRM, DCH_DCTRL.X / YSEL : The Output Destination Image Data Format

(17) BCH_DFSIZE : The Output Destination Image Frame Size

(18) BCH_DOFF : The Output Destination Image Offset

(19) OM_CTRL : OM_CTRL.EN = b'010

9.4.2 YUV to RGB Format Conversion

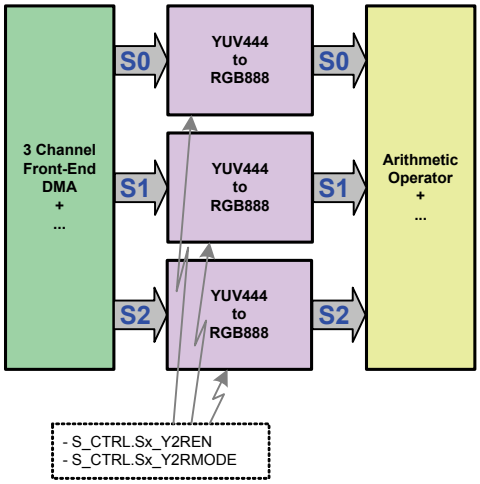


Figure 9.9 Source YUV to RGB Conversion

Figure 9.9 explains one of functions supported by Overlay Mixer, 3 Channel YUV to RGB Format Conversion function. As can be seen, 3 Channel YUV Source Image of Overlay Mixer can be converted to RGB888. S0 and S1 in Figure 9.9 are the images selected by S_CTRL.SxSEL Register.

Below shows 4 equations in YUV to RGB Conversion per each S_CTRL.Sx_YM Register.

- YUVtoRGB Format Converter Type 0
  - R = Y + 1.371 (V – 128)
  - G = Y – 0.336 (U – 128) – 0.698 (V – 128)
  - B = Y + 1.732 (U – 128)
- YUVtoRGB Format Converter Type 1
  - R = 1.164 (Y – 16) + 1.159 (V – 128)
  - G = 1.164 (Y – 16) – 0.391 (U – 128) – 0.813 (V – 128)
  - B = 1.164 (Y – 16) + 2.018 (U – 128)
- YUVtoRGB Format Converter Type 2
  - R = Y + 1.540 (V – 128)
  - G = Y – 0.183 (U – 128) – 0.459 (V – 128)
  - B = Y + 1.816 (U – 128)
- YUVtoRGB Format Converter Type 3
  - R = 1.164 (Y – 16) + 1.793 (V – 128)
  - G = 1.164 (Y – 16) – 0.213 (U – 128) – 0.534 (V – 128)
  - B = 1.164 (Y – 16) + 2.115 (U – 128)

9.4.3 Arithmetic Operation

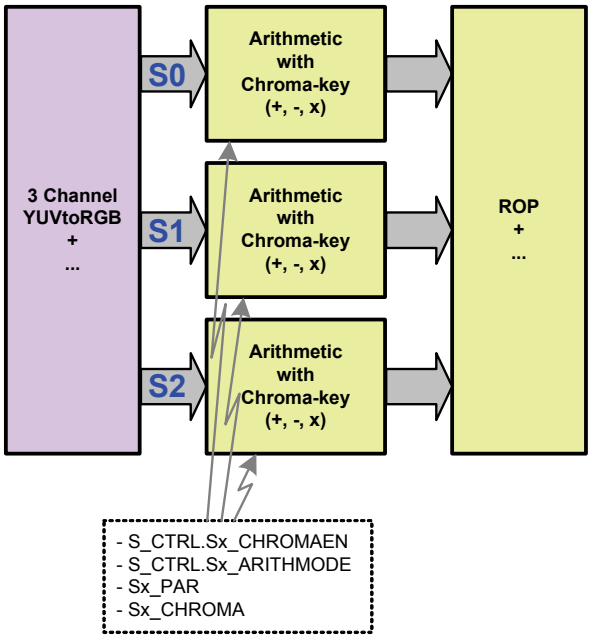


Figure 9.10 Arithmetic Operation

Figure 9.10 explains 3 Channel Arithmetic Operation, one of functions which Overlay Mixer supports. As can be seen above, arithmetic operation for 3 Channel Image Source is available in Overlay Mixer. In the figure, S0, S1 and S2 are the results of performing YUV to RGB Conversion in the images selected by S_CTRL.SxSEL Register.

The following is about Arithmetic Operation per each S_CTRL.Sx_ARITHMODE Register.

	A	B	Result	
			A = Sx_CHROMA	A != Sx_CHROMA
BYPASS	Source 0/1/2 Image	Do not Used	A	Y = A
FILL	Do not Used	Sx_PAR	A	Y = B
INV	Source 0/1/2 Image	Do not Used	A	Y = 255 - A
ADD	Source 0/1/2 Image	Sx_PAR	A	Y = A + B
SUBA	Source 0/1/2 Image	Sx_PAR	A	Y = A – B
SUBB	Source 0/1/2 Image	Sx_PAR	A	Y = B – A
MUL	Source 0/1/2 Image	Sx_PAR	A	Y = A x B Mantissa = B[7:6] Fraction = B[5:0]

As can be seen above, Arithmetic Operation is performed after Source YUV to RGB Conversion.

Therefore, for Arithmetic Operation with Chroma-Key, the value after Source YUV to RGB Conversion should be set as Chroma-Key Vaule (Sx_CHROMA).

9.4.4 Alpha-Blending and ROP with Window Offset and Chroma-Key

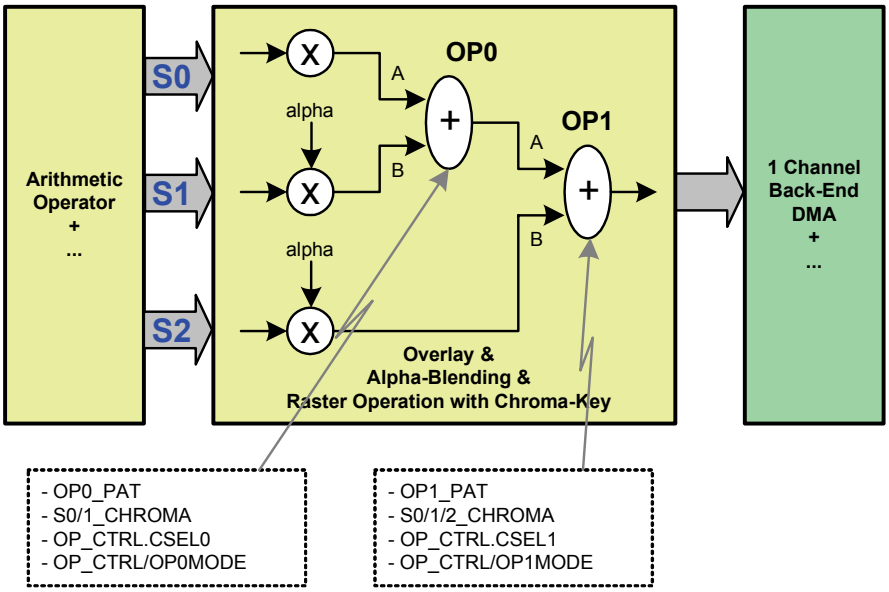


Figure 9.11 ROP / Alpha-Blending Block Diagram

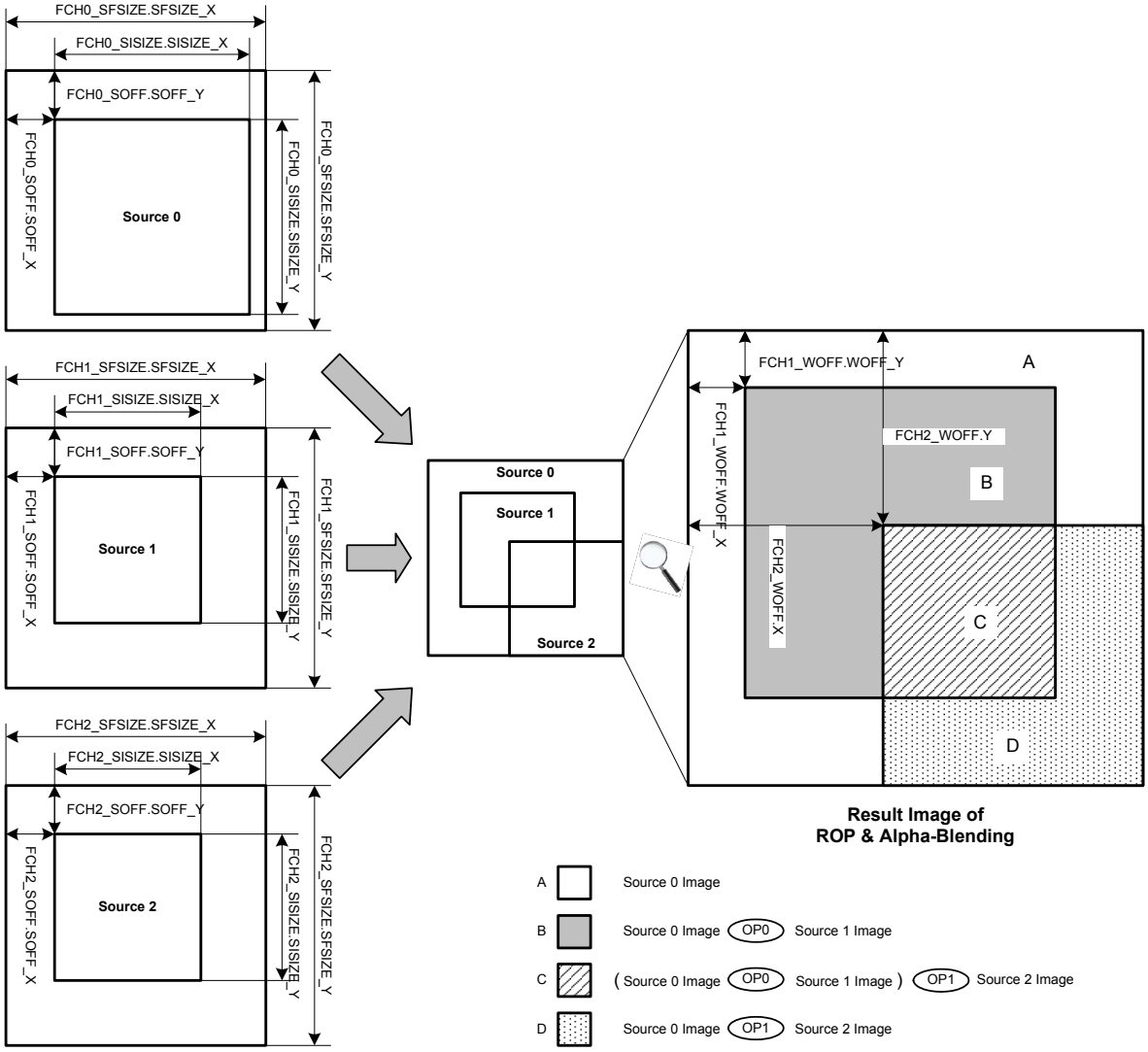


Figure 9.12 ROP and Alpha-Blending Operation Example

This figure illustrates ROP, Alpha-Blending Operation. It assumes that FCH0 is Source 0 and FCH1, FCH2 are Source 1 and Source 2 respectively. As the figure explains, Source 0 Image should definitely include other Source Images.

In addition, in case of Source 0 Image, Window Offset should definitely be '0'. Other than these, there are no requirements for ROP and Alpha-Blending Operation.

Below shows the function of Operator 0 / 1.

Mode	Result of Operator 0		
	A = Source 0 selected by S_CTRL.S0SEL		
	B = Source 1 selected by S_CTRL.S1SEL		
	P = OP0_PAT[23:0]		
	ALPHA0 = OP0_PAT[31:24] / 256		
	ALPHA1 = Alpha-Value of ARGB Format Source 1 Data		
	OPCTRL_CSEL0	01	10
		A = S0_CHROMA	B = S1_CHROMA
Blackness	Y = 0	B	A
Merged Copy	Y = P & A	B	A
Merged Paint	Y = ~ A | B	B	A
Pattern Copy	Y = P	B	A
Pattern Invert	Y = P ^ B	B	A
Pattern Paint	Y = (P | ~ A) | B	B	A
Src Copy	Y = A	B	A
Src Inverter	Y = A ^ B	B	A
Srd Paint	Y = A | B	B	A
Src AND	Y = A & B	B	A
Src Erase	Y = A & ~ B	B	A
Not Src Copy	Y = ~ A	B	A
Not Src Erase	Y = ~ (A | B)	B	A
Dst Copy	Y = B	B	A
Dst Inverter	Y = ~ B	B	A
Whiteness	Y = 1	B	A
Alpha-Blending 0	Y = ALPHA0 * B + (1 – ALPHA0) *A	B	A
Alpha-Blending 1	Y = ALPHA1 * B + (1 – ALPHA1) *A	B	A

Mode	Result of Operator 1		
	A = Result Image of Operator 0		
	B = Source 2 selected by S_CTRL.S2SEL		
	P = OP1_PAT[23:0]		
	ALPHA0 = OP1_PAT[31:24] / 256		
	ALPHA1 = Alpha-Value of ARGB Format Source 1 Data		
	OPCTRL_CSEL1	01	10
		A = S0_CHROMA	B = S1_CHROMA
Blackness	Y = 0	B	A
Merged Copy	Y = P & A	B	A
Merged Paint	Y = ~ A | B	B	A
Pattern Copy	Y = P	B	A
Pattern Invert	Y = P ^ B	B	A
Pattern Paint	Y = (P | ~ A) | B	B	A
Src Copy	Y = A	B	A
Src Inverter	Y = A ^ B	B	A
Srd Paint	Y = A | B	B	A
Src AND	Y = A & B	B	A
Src Erase	Y = A & ~ B	B	A
Not Src Copy	Y = ~ A	B	A
Not Src Erase	Y = ~ (A | B)	B	A
Dst Copy	Y = B	B	A
Dst Inverter	Y = ~ B	B	A
Whiteness	Y = 1	B	A
Alpha-Blending 0	Y = ALPHA0 * B + (1 – ALPHA0) *A	B	A
Alpha-Blending 1	Y = ALPHA1 * B + (1 – ALPHA1) *A	B	A

9.4.5 RGB to YUV Format Conversion

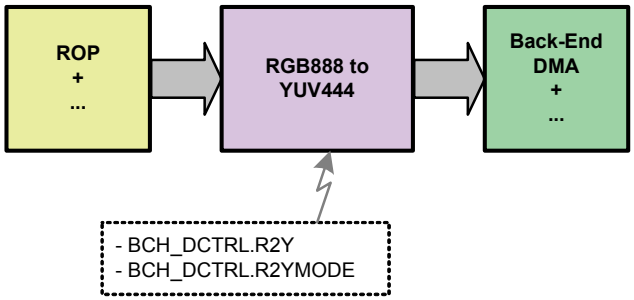


Figure 9.13 RGB to YUV Format Converter

The figure is about 1 Channel RGB to YUV Format Conversion function provided by Overlay Mixer.

As can be seen above, RGB to YUV Format Conversion is performed after ROP and Alpha-Blending Operation.

The following explains the equation of RGB to YUV Conversion per BCH_DCTRL.R2YMODE Register.

- RGBtoYUV Format Converter Type 0
  - Y = 0.299 R + 0.587 G + 0.114 B
  - U = -0.172 R – 0.339 G + 0.511 B + 128
  - V = 0.511 R – 0.428 G – 0.083 B + 128
- RGBtoYUV Format Converter Type 1
  - Y = 0.257 R + 0.504 G + 0.098 B + 16
  - U = -0.148 R – 0.291 G + 0.439 B + 128
  - V = 0.439 R – 0.368 G – 0.071 B + 128
- RGBtoYUV Format Converter Type 2
  - Y = 0.213 R + 0.715 G + 0.072 B
  - U = -0.117 R – 0.394 G + 0.511 B + 128
  - V = 0.511 R – 0.464 G – 0.047 B + 128
- RGBtoYUV Format Converter Type 3
  - Y = 0.183 R + 0.614 G + 0.062 B + 16
  - U = -0.101 R – 0.338 G + 0.439 B + 128
  - V = 0.439 R – 0.399 G – 0.040 B + 128

9.4.6 Destination RGB Dithering

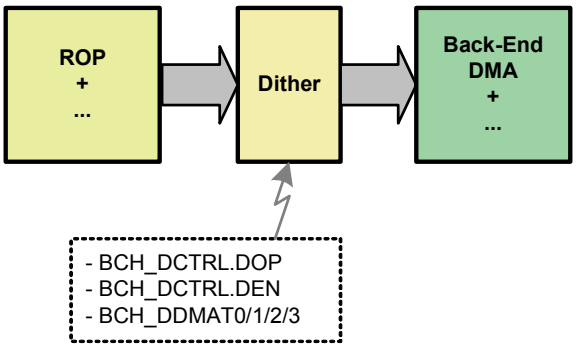


Figure 9.14 Destination RGB Dithering

The figure is about 1 Channel RGB dithering function provided by Overlay Mixer. Patterned Dithering Algorithm is used in dithering algorithm of Overlay Mixer. Below figure illustrates detail operation of Patterned Dithering.

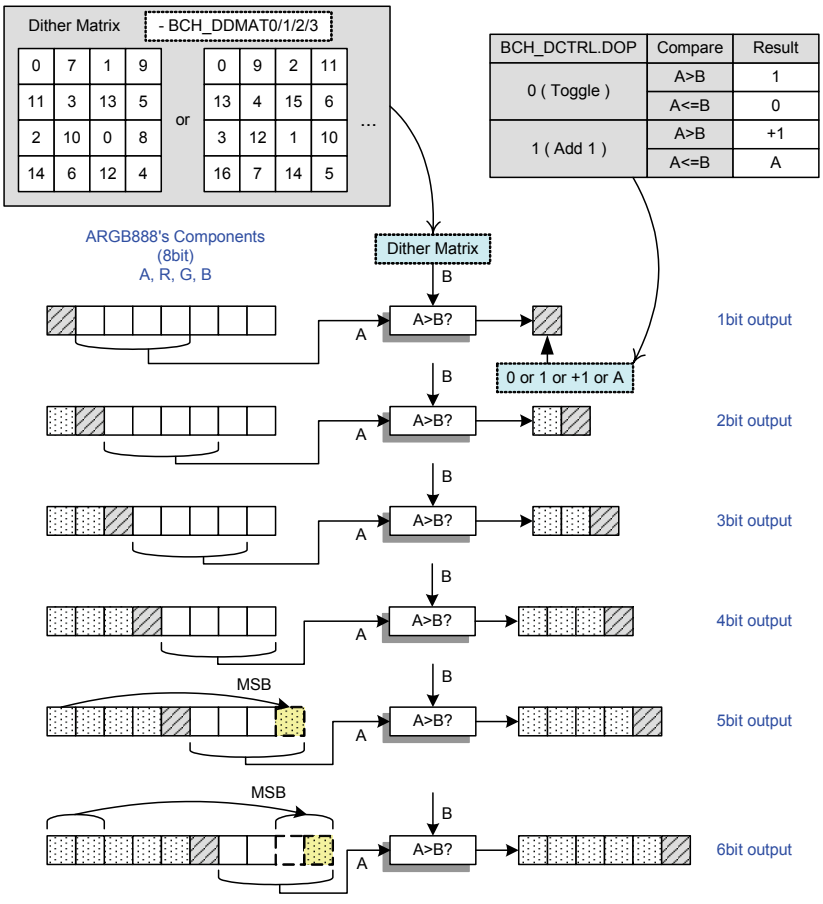


Figure 9.15 Destination RGB Dithering Operation

10 Audio0 (7.1ch)

10.1 Overview

10.1.1 AUDIO DMA

AUDIO DMA is a specialized DMA for Audio Interfaces such as DAI, CDIF and SPDIF. For earlier NEXTCHIP series, only general DMA is available. When a general DMA is used for transferring audio data, it has to read from memory and write to Audio Interface registers. It seems to have redundant bus cycle time from the viewpoint of time-critical applications. So we applied a dedicated DMA to Audio Interface to reduce redundant bus cycle time. AUDIO DMA enable you to utilize audio data for more flexible purpose. The structure of AUDIO DMA is similar to that of general DMA. You also use general DMA instead of AUDIO DMA. You are not recommended to use AUDIO DMA and general DMA at the same time.

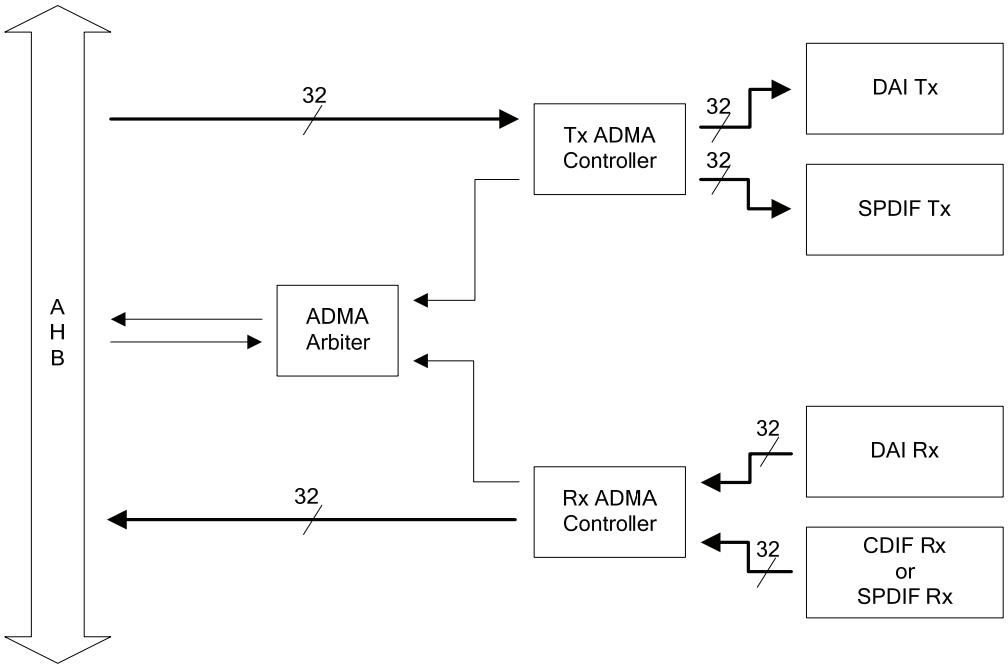


Figure 10.1 AUDIO DMA Top Block Diagram

10.1.2 DAI & CDIF

The block diagram of DAI is shown in below figure.

The NVS2310 provides digital audio interface that complies with IIS (Inter-IC Sound). The DAI has five input/output pins for IIS interface; MCLK, BCLK, LRCK, DAI, DAO. All DAI input/output pins are multiplexed with GPIO pins; GPIO_E[4:0].

The MCLK is the system clock pin that is used for CODEC system clock. In master mode, the MCLK can be generated from clock generator in which that is known as a DCLK, or fed from the outside of chip in slave mode. The DAI can process 256fs, 384fs and 512fs as a system clock. 256fs means that the system clock has 256 times of sampling frequency (fs).

The BCLK is the serial bit clock for IIS data exchange. The DAI can generate 64fs, 48fs and 32fs by dividing a system clock. The polarity of BCLK can be programmed. That is, the serial bit can be stable either rising edge of BCLK or falling edge of BCLK.

The LRCK is the frame clock for the stereo audio channel Left and Right. The frequency of LRCK is known as the “fs” – sampling frequency. Generally, for audio application – such as MP3 Player, CD player, the fs can be set to 8kHz, 16kHz, 11.05kHz, 24kHz, 32kHz, 44.1kHz and 48kHz. For supporting the wide range of sampling frequency in audio application, the DCO function is very useful to generate a system clock. Refer the chapter of clock generator for detail information.

All three clocks (MCLK, BCLK, LRCK) are selectable as master or slave.

The DAI, DAO are the serial data input output pins respectively.

The DAI has two 8-word input/output buffers. It has a banked buffer structure so that one side of buffer is receiving/transmitting data while the other side of that can be read/written through the DADI_XX/DADO_XX registers. The maximum data word size is 24 bit. Data is justified to MSB of 32bits and zeros are padded to LSB.

There are 2 types of interrupt from IIS; transmit done interrupt, receive done interrupt. The transmit-done interrupt is generated when the 8 words are transferred successfully in the output buffer. At this interrupt, user should fill another 8 more words into the other part of the output buffer in the interrupt service routine (ISR). In this ISR routine, 8 consecutive stores of word data to the DADO registers are needed. The receive-done interrupt is generated when the 8 words are received successfully in the input buffer. At this interrupt, user should read 8 received words from the input buffer using 8 consecutive load instructions from the DADI registers.

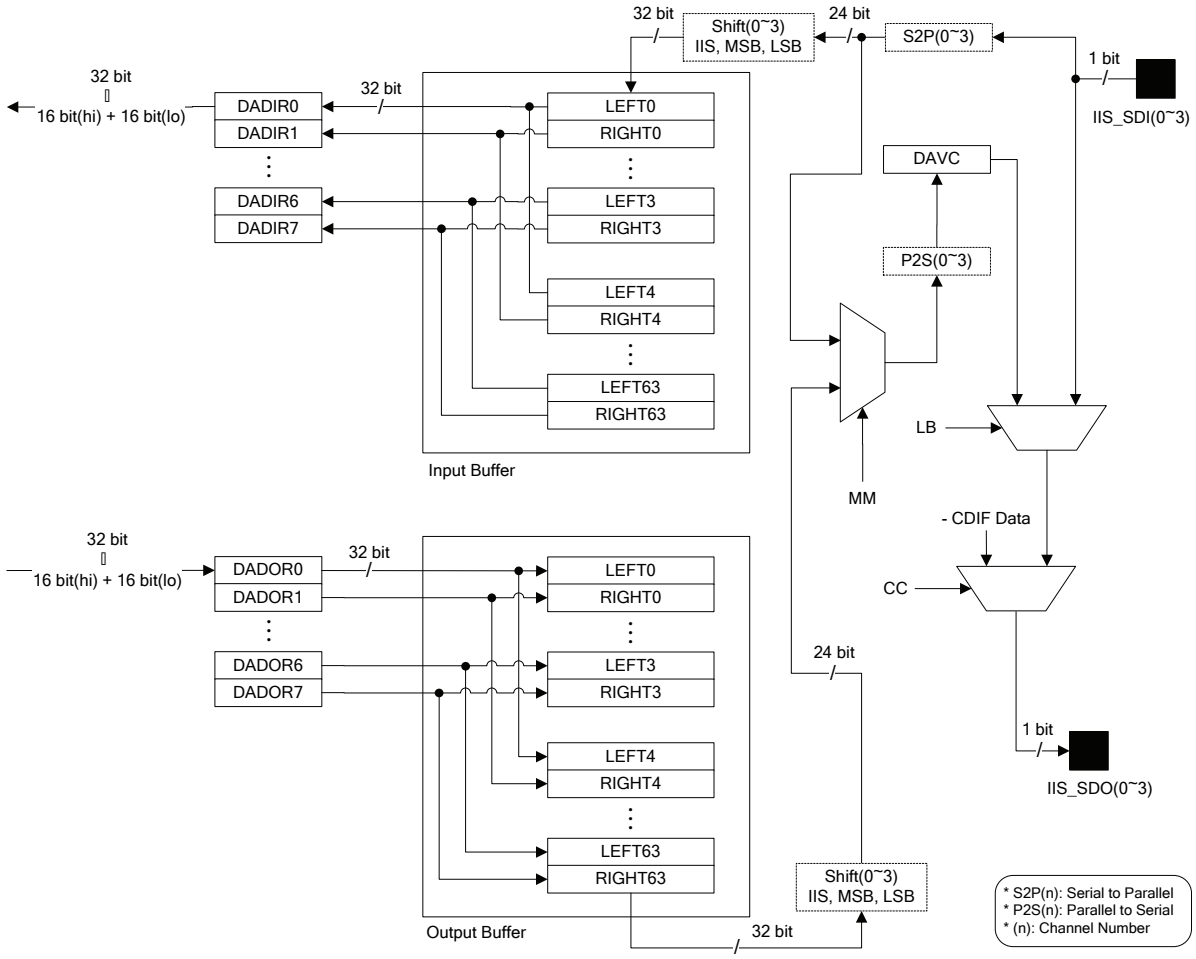


Figure 10.2 DAI Block Diagram

The block diagram of CDIF is illustrated in below figure.

The NVS2310 provides CD-ROM interface for feasible implementation of CD-ROM application such as CD-MP3 player. The CDIF supports the industry standard IIS format and the LSB justified format that is used as the most popular format for CD-ROM interface by Sony and Samsung.

The CDIF has three pins for interface; CBCLK, CLRCK, CDAI. These are multiplexed with GPIO_B[13], GPIO_B[14] and GPIO_B[12]. The CBCLK is the bit clock input pins of which frequency can be programmed by CICR for selection of 48fs and 32fs. The CLRCK is the frame clock input pin that indicates the channel of CD stereo digital audio data. The CDAI is the input data pin.

The CDIF has nine registers; CDDI_0 to CDDI_3 and CICR. The CDDI_0 to the CDDI_3 are the banked read only registers for access of data input buffer. The data input buffer is composed of sixteen 32 bit wide registers of which upper 16 bit is left channel data and lower is right channel data.

The CDIF receive the serial data from CDAI pin and store the data into the buffer through the serial to parallel register. Whenever the half of buffer is filled, the receive interrupt is generated. Only the half of input buffer can be accessible through the CDDI_0 to the CDDI_3.

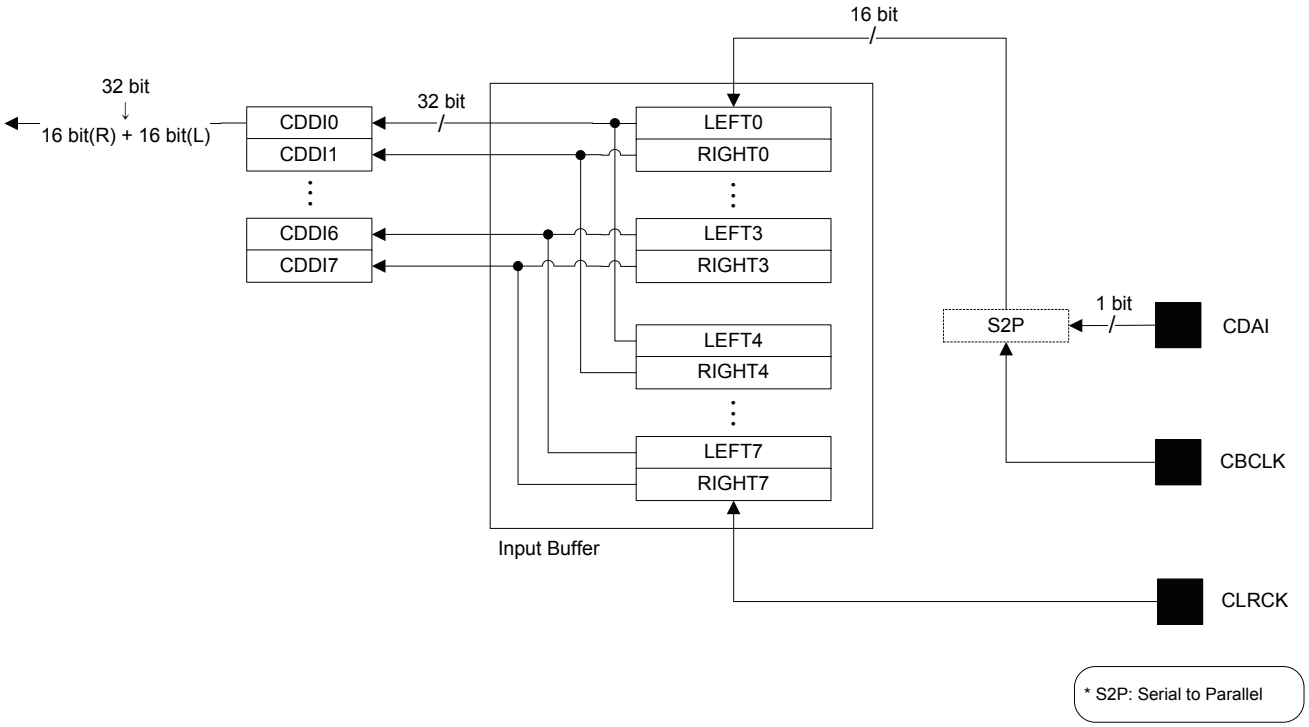


Figure 10.3 CDIF Block Diagram

10.1.3 SPDIF

The SPDIF (or AES/EBU, IEC950 standards) is a point-to-point protocol for serial transmission of digital audio through a single transmission line. The transmission medium can be either electrical or optical (e.g. TosLink). It provides two channels for audio data and some error detection capabilities. The audio data means a method for communicating control information. The control information is transmitted as one bit per sample and accumulates in a block structure. The data is by-phase encoded, which enables the receiver to extract a clock from the data. Coding violations, defined as preambles, are used to identify sample and block boundaries.

The SPDIF interfaces are found on most CD/DVD players, audio equipment and computer sound cards.

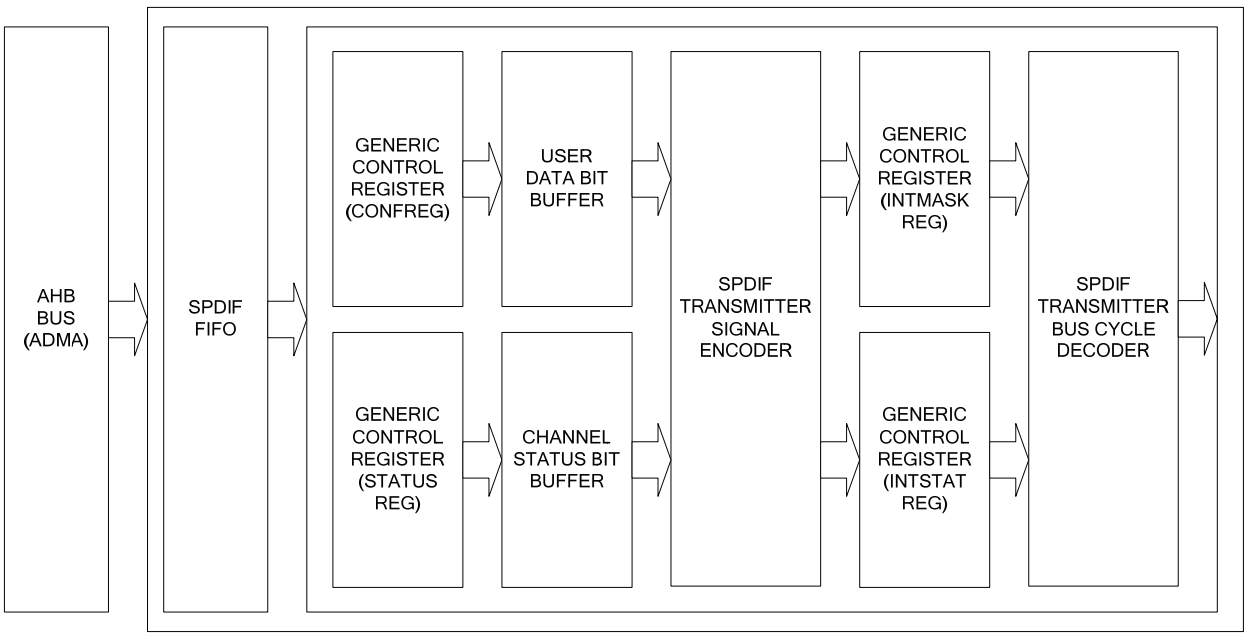


Figure 10.4 SPDIF Tx Block Diagram

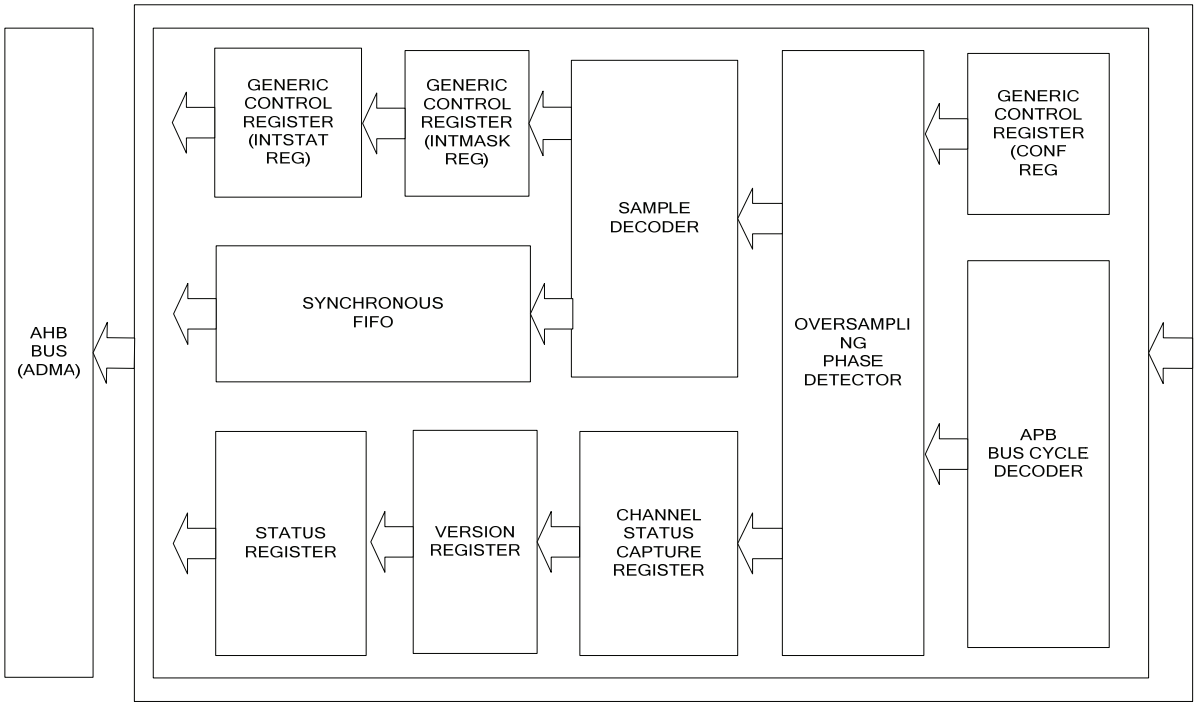


Figure 10.5 SPDIF Rx Block Diagram

10.2 Register Description

Table 10.1 AUDIO DMA Register Map (Base Address = 0xB0103000)

Name	Address	Type	Reset	Description
RxDaDar	0x00	R/W	0x00000000	DAI Rx (Right) Data Destination Address
RxDaParam	0x04	R/W	0x00000000	DAI Rx Parameters
RxDaTCnt	0x08	R/W	0x00000000	DAI Rx Transmission Counter Register
RxDaCdar	0x0C	R	0x00000000	DAI Rx (Right) Data Current Destination Address
RxCdDar	0x10	R/W	0x00000000	CDIF(SPDIF) Rx (Right) Data Destination Address
RxCdParam	0x14	R/W	0x00000000	CDIF(SPDIF) Rx Parameters
RxCdTCnt	0x18	R/W	0x00000000	CDIF(SPDIF) Rx Transmission Counter Register
RxCdCdar	0x1C	R	0x00000000	CDIF(SPDIF) Rx (Right) Data Current Destination Address
RxDaDarL	0x28	R/W	0x00000000	DAI Rx Left Data Destination Address
RxDaCdarL	0x2C	R	0x00000000	DAI Rx Left Data Current Destination Address
RxCdDarL	0x30	R/W	0x00000000	CDIF(SPDIF) Rx Left Data Destination Address
RxCdCdarL	0x34	R	0x00000000	CDIF(SPDIF) Rx Left Data Current Destination Address
TransCtrl	0x38	R/W	0x0000AA00	DMA Transfer Control Register
RptCtrl	0x3C	R/W	0x00000000	DMA Repeat Control Register
TxDaSar	0x40	R/W	0x00000000	DAI Tx (Right) Data Source Address
TxDaParam	0x44	R/W	0x00000000	DAI Tx Parameters
TxDaTCnt	0x48	R/W	0x00000000	DAI Tx Transmission Counter Register
TxDaCsar	0x4C	R	0x00000000	DAI Tx (Right) Data Current Source Address
TxSpSar	0x50	R/W	0x00000000	SPDIF Tx (Right) Data Source Address
TxSpParam	0x54	R/W	0x00000000	SPDIF Tx Parameters
TxSpTCnt	0x58	R/W	0x00000000	SPDIF Tx Transmission Counter Register
TxSpCsar	0x5C	R	0x00000000	SPDIF Tx (Right) Data Current Source Address
TxDaSarL	0x68	R/W	0x00000000	DAI Tx Left Data Source Address
TxDaCsarL	0x6C	R	0x00000000	DAI Tx Left Data Current Source Address
TxSpSarL	0x70	R/W	0x00000000	SPDIF Tx Left Data Source Address
TxSpCsarL	0x74	R	0x00000000	SPDIF Tx Left Data Current Source address
ChCtrl	0x78	R/W	0x00008000	DMA Channel Control Register
IntStatus	0x7C	R/W	0x00000000	DMA Interrupt Status Register
GIntReq	0x80	R/W	0x00000000	General Interrupt Request
GIntStatus	0x84	R	0x00000000	General Interrupt Status
RxDaDar1	0x100	R/W	0x00000000	DAI1 Rx (Right) Data Destination Address
RxDaDar2	0x104	R/W	0x00000000	DAI2 Rx (Right) Data Destination Address
RxDaDar3	0x108	R/W	0x00000000	DAI3 Rx (Right) Data Destination Address
RxDaCar1	0x10C	R	0x00000000	DAI1 Rx (Right) Data Current Destination Address
RxDaCar2	0x110	R	0x00000000	DAI2 Rx (Right) Data Current Destination Address
RxDaCar3	0x114	R	0x00000000	DAI3 Rx (Right) Data Current Destination Address
RxDaDarL1	0x118	R/W	0x00000000	DAI1 Rx Left Data Destination Address
RxDaDarL2	0x11C	R/W	0x00000000	DAI2 Rx Left Data Destination Address
RxDaDarL3	0x120	R/W	0x00000000	DAI3 Rx Left Data Destination Address
RxDaCarL1	0x124	R	0x00000000	DAI1 Rx Left Data Current Destination Address
RxDaCarL2	0x128	R	0x00000000	DAI2 Rx Left Data Current Destination Address
RxDaCarL3	0x12C	R	0x00000000	DAI3 Rx Left Data Current Destination Address
TxDaSar1	0x130	R/W	0x00000000	DAI1 Tx (Right) Data Source Address
TxDaSar2	0x134	R/W	0x00000000	DAI2 Tx (Right) Data Source Address
TxDaSar3	0x138	R/W	0x00000000	DAI3 Tx (Right) Data Source Address
TxDaCsar1	0x13C	R	0x00000000	DAI1 Tx (Right) Data Current Source Address
TxDaCsar2	0x140	R	0x00000000	DAI2 Tx (Right) Data Current Source Address
TxDaCsar3	0x144	R	0x00000000	DAI3 Tx (Right) Data Current Source Address
TxDaDarL1	0x148	R/W	0x00000000	DAI1 Tx Left Data Source Address
TxDaDarL2	0x14C	R/W	0x00000000	DAI2 Tx Left Data Source Address
TxDaDarL3	0x150	R/W	0x00000000	DAI3 Tx Left Data Source Address
TxDaCarL1	0x154	R	0x00000000	DAI1 Tx Left Data Current Source Address

TxDaCarL2	0x158	R	0x00000000	DAI2 Tx Left Data Current Source Address
TxDaCarL3	0x15C	R	0x00000000	DAI3 Tx Left Data Current Source Address

Table 10.2 DAI Register Map (Base Address = 0xB0104000)

Name	Address	Type	Reset	Description
DADIR0	0x00	R	0xXXXXXXXX	Digital Audio Input Register 0
DADIR1	0x04	R	0xXXXXXXXX	Digital Audio Input Register 1
DADIR2	0x08	R	0xXXXXXXXX	Digital Audio Input Register 2
DADIR3	0x0C	R	0xXXXXXXXX	Digital Audio Input Register 3
DADIR4	0x10	R	0xXXXXXXXX	Digital Audio Input Register 4
DADIR5	0x14	R	0xXXXXXXXX	Digital Audio Input Register 5
DADIR6	0x18	R	0xXXXXXXXX	Digital Audio Input Register 6
DADIR7	0x1C	R	0xXXXXXXXX	Digital Audio Input Register 7
DADOR0	0x20	R/W	0xXXXXXXXX	Digital Audio Output Register 0
DADOR1	0x24	R/W	0xXXXXXXXX	Digital Audio Output Register 1
DADOR2	0x28	R/W	0xXXXXXXXX	Digital Audio Output Register 2
DADOR3	0x2C	R/W	0xXXXXXXXX	Digital Audio Output Register 3
DADOR4	0x30	R/W	0xXXXXXXXX	Digital Audio Output Register 4
DADOR5	0x34	R/W	0xXXXXXXXX	Digital Audio Output Register 5
DADOR6	0x38	R/W	0xXXXXXXXX	Digital Audio Output Register 6
DADOR7	0x3C	R/W	0xXXXXXXXX	Digital Audio Output Register 7
DAMR	0x40	R/W	0x00000000	Digital Audio Mode Register
DAVC	0x44	R/W	0x0000	Digital Audio Volume Control Register
MCCR0	0x48	R/W	0x00000000	Multi Channel Control Register 0
MCCR1	0x4C	R/W	0x00000000	Multi Channel Control Register 1

The DAMR register must be set after set the MCCR register.

Table 10.3 CDIF Register Map (Base Address = 0xB0104080)

Name	Address	Type	Reset	Description
CDDI_0	0x80	R	0xXXXXXXXX	CD Digital Audio Input Register 0
CDDI_1	0x84	R	0xXXXXXXXX	CD Digital Audio Input Register 1
CDDI_2	0x88	R	0xXXXXXXXX	CD Digital Audio Input Register 2
CDDI_3	0x8C	R	0xXXXXXXXX	CD Digital Audio Input Register 3
CDDI_4	0x90	R	0xXXXXXXXX	CD Digital Audio Input Register 4
CDDI_5	0x94	R	0xXXXXXXXX	CD Digital Audio Input Register 5
CDDI_6	0x98	R	0xXXXXXXXX	CD Digital Audio Input Register 6
CDDI_7	0x9C	R	0xXXXXXXXX	CD Digital Audio Input Register 7
CICR	0xA0	R/W	0x0000	CD Interface Control Register

Table 10.4 SPDIF Tx Register Map (Base Address = 0xB0105000)

Name	Address	Type	Reset	Description
TxVersion	0x00	R	0x00003111	Version Register
TxConfig	0x04	R/W	0x00000000	Configuration Register
TxChStat	0x08	R/W	0x00000000	Channel Status Control Register
TxIntMask	0x0C	R/W	0x00000000	Interrupt Mask Register
TxIntStat	0x10	R/W	0x00000000	Interrupt Status Register
UserData	0x80~0xDC	W	-	User Data Buffer
ChStatus	0x100~0x15C	W	-	Channel Status Buffer
TxBuffer	0x200~0x23C	W	-	Transmit Data Buffer
DMACFG	0x400	R/W	0x00000007	Additional Configuration for DMA
CSBUDB	0x680~0x6DC	W	-	Merged Window for CSB/UDB

Table 10.5 SPDIF Rx Register Map (Base Address = 0xB0105800)

Name	Address	Type	Reset	Description
RxVersion	0x00	R	0x00080111	Version Register
RxConfig	0x04	R/W	0x00000000	Configuration Register
RxStatus	0x08	R	0x00000000	Signal Status Buffer
RxIntMask	0x0C	R/W	0x00000000	Interrupt Mask Register
RxIntStat	0x10	R/W	0x00000000	Interrupt Status Register
RxCapCtl [n]	0x40~0x7C(even)	RW	0x00000000	Channel Status Capture Control Register
RxCap [n]	0x40~0x7C(odd)	R	0x00000000	Captured Channel Status / user bit
RxBuffer	0xA00~0xA1C	R	-	Receive Data Buffer

10.2.1 Audio DMA Register

DAI0(CDIF)Rx (Right) Data Destination / DAI0(SPDIF)Tx (Right) Data Source Address Register
0xB0103000(0x10) / 0xB0103040(0x50)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RxDaDar(RxCdDar) / TxDaSar(TxSpSar)															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RxDaDar(RxCdDar) / TxDaSar(TxSpSar)															

Field	Name	RW	Reset	Description
31-0	RxDaDar(RxCdDar) / TxDaSar(TxSpSar)	R/W	32'h0	Rx(Right) Data Destination Address Register / Tx(Right) Data Source Address Register

This register contains the start address of target memory block for DAI0 Rx DMA transfer. The transfer begins reading data from this address. When LRMode is asserted, this is used for Right Data of DAI0 Rx.

** Multi Channel:*
- *DAI1 Rx / DAI1 Tx → 0x100 / 0x130*
- *DAI2 Rx / DAI2 Tx → 0x104 / 0x134*
- *DAI3 Rx / DAI3 Tx → 0x108 / 0x138*

DAI(CDIF)Rx / DAI(SPDIF)Tx Parameters Register **0xB0103004(0x14) / 0xB0103044(0x54)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SMASK															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SMASK								SINC							

Field	Name	RW	Reset	Description
31-8	SMASK	R/W	24'h0	Source Address Mask Register 24'h0: non-masked 24'hn: Masked so that source address bit doesn't be changed during DMA transfer
7-0	SINC	R/W	8'h0	Source Address Increment Register 8'hn: Source address is added by amount of sinc at every write cycles. sinc is represented as 2's complement, so if SINC[7] is 1, the source address is decremented.

Each bit field controls the dedicated bit of source address field. That is, if SMASK[23] is set to 1, the 28th bit of source address is masked, and if SMASK[22] is set to 1, the 27th bit of source address is masked, and so on. If a bit is masked, a corresponding bit of address bus is not changed during DMA transfer. This function can be used to generate circular buffer address.

The addresses of DMA transfer have 32bit wide, but the upper 4bit of them are not affected during DMA transfer. If the source or destination address reaches its maximum address space like 0x7FFFFFFF or 0x2FFFFFFF, the next transfer is starting from 0x70000000 or 0x20000000 not from 0x80000000 or 0x30000000.

DAI(CDIF) Rx / DAI(SPDIF) Tx Transmission Counter Register

0xB0103008(0x18) / 0xB0103048(0x58)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C_TCOUNT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST_TCOUNT															

Field	Name	RW	Reset	Description
31-16	C_TCOUNT	R/W	16'h0	Current Transmisson Count 16'hn: Represent cn number of Hop transfer remains
15-0	ST_TCOUNT	R/W	16'h0	Start Transmission Count 16'hn: DMA transfers data by amount of sn Hop transfers

At the beginning of transfer, the C_TCNT is updated by ST_TCNT register. At the end of every hop transfer, this is decremented by 1 until it reaches to zero. When this reaches to zero, the DMA finishes its transfer and may or may not generate its interrupt according to IEN flag of CHCTRL register.

DAI0(CDIF)Rx (Right) Data Current Destination / DAI0(SPDIF)Tx (Right) Data Current Source Address Register

0xB010300C(0x1C) / 0xB010304C(0x5C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RxDaCdar(RxCdCdar) / TxDaCsar(TxSpCsar)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RxDaCdar(RxCdCdar) / TxDaCsar(TxSpCsar)															

Field	Name	RW	Reset	Description
31-0	RxDaCdar(RxCdCdar) / TxDaCsar(TxSpCsar)	R/W	32'h0	Rx(Right) Data Current Destination Address Register / Tx(Right) Data Current Source Address Register

This register contains the current destination address of DAI0 Rx DMA transfer. It represents that the current transfer read data from this address. This is read only register. When LRMode is asserted, this is used for Right Data of DAI0 Rx.

** Multi Channel:*

- *DAI1 Rx / DAI1 Tx → 0x10C / 0x13C*
- *DAI2 Rx / DAI2 Tx → 0x110 / 0x140*
- *DAI3 Rx / DAI3 Tx → 0x114 / 0x144*

DAI0(CDIF)Rx Left Data Destination / DAI0(SPDIF)Tx Left Data Source Address Register

0xB0103028(0x30) / 0xB0103068(0x70)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RxDaDarL(RxCdDarL) / TxDaSarL(TxSpSarL)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RxDaDarL(RxCdDarL) / TxDaSarL(TxSpSarL)															

Field	Name	RW	Reset	Description
31-0	RxDaDarL(RxCdDarL) / TxDaSarL(TxSpSarL)	R/W	32'h0	Rx Left Data Destination Address Register / Tx Left Data Source Address Register

This register contains the start address of target memory block for DAI0 Rx DMA transfer. The transfer begins reading data from this address. When LR Mode is asserted, this is used for Left Data of DAI0 Rx, otherwise this is not available.

** Multi Channel:*

- *DAI1 Rx / DAI1 Tx → 0x118 / 0x148*
- *DAI2 Rx / DAI2 Tx → 0x11C / 0x14C*
- *DAI3 Rx / DAI3 Tx → 0x120 / 0x150*

DAI0(CDIF)Rx Left Data Current Destination / DAI0(SPDIF)Tx Left Data Current Source Address Register

0xB010302C(0x34) / 0xB010306C(0x74)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RxDaCdarL(RxCdCdarL) / TxDaCsarL(TxSpCsarL)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RxDaCdarL(RxCdCdarL) / TxDaCsarL(TxSpCsarL)															

Field	Name	RW	Reset	Description
31-0	RxDaCdarL(RxCdCdarL) / TxDaCsarL(TxSpCsarL)	R/W	32'h0	Rx Left Data Current Destination Address Register / Tx Left Data Current Source Address Register

This register contains the current destination address of DAI0 Rx DMA transfer. It represents that the current transfer read data from this address. This is read only register. When LRMode is asserted, this is used for Left Data of DAI0 Rx, otherwise this is not available

** Multi Channel:*

- *DAI1 Rx / DAI1 Tx → 0x124 / 0x154*
- *DAI2 Rx / DAI2 Tx → 0x128 / 0x158*
- *DAI3 Rx / DAI3 Tx → 0x12C / 0x15C*

Transfer Control Register

0xB0103038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*		RCN	TCN	CRLCK	DRLCK	STLCK	DTLCK	CRTRG	DRTRG	STTRG	DTTRG	CRRPT	DRRPT	STRPT	DTRPT
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRBSIZE		DRBSIZE		STBSIZE		DTBSIZE		CRWSIZE		DRWSIZE		STWSIZE		DTWSIZE	

Field	Name	RW	Reset	Description
31-30	R*	R	0	Reserved
29	RCN	R/W	0	Issue Continuous Transfer of Rx DMA 0 : Rx DMA transfer begins from source/destination address 1 : Rx DMA transfer begins from current source/destination address It must be used after the former transfer has been executed, so that current source/destination address registers contain a meaningful value.
28	TCN	R/W	0	Issue Continuous Transfer of Tx DMA 0 : Tx DMA transfer begins from source/destination address 1 : Tx DMA transfer begins from current source/destination address It must be used after the former transfer has been executed, so that current source/destination address registers contain a meaningful value.
27	CRLCK	R/W	0	Issue Locked Transfer of CDIF Rx DMA 0 : CDIF Rx DMA transfer is done without lock transfer 1 : CDIF Rx DMA transfer is done with lock transfer Lock field controls the LOCK signal (refer to AHB specification). When the LOCK is set to 1, the DMA transfer doesn't be bothered by other AHB masters like LCD controller, ARM etc.
26	DRLCK	R/W	0	Issue Locked Transfer of DAI Rx DMA 0 : DAI Rx DMA transfer is done without lock transfer 1 : DAI Rx DMA transfer is done with lock transfer
25	STLCK	R/W	0	Issue Locked Transfer of SPDIF Tx DMA 0 : SPDIF Tx DMA transfer is done without lock transfer 1 : SPDIF Tx DMA transfer is done with lock transfer
24	DTLCK	R/W	0	Issue Locked Transfer of DAI Tx DMA 0 : DAI Tx DMA transfer is done without lock transfer 1 : DAI Tx DMA transfer is done with lock transfer
23	CRTRG	R/W	0	Trigger Type of Rx DMA 0 : SINGLE edge-triggered detection of CDIF Rx DMA 1 : SINGLE level-sensitive detection of CDIF Rx DMA In SINGLE Type, After one Hop data transferring DMA checks External DMA Request (DREQ) and then if its bit is active , DMA transfers next hop data . DREQ is detected level-sensitive or edge-

				triggered by SINGLE transfer TYPE.
22	DRTRG	R/W	0	Trigger Type of Rx DMA 0 : SINGLE edge-triggered detection of DAI Rx DMA 1 : SINGLE level-sensitive detection of DAI Rx DMA
21	STTRG	R/W	0	Trigger Type of Tx DMA 0 : SINGLE edge-triggered detection of SPDIF Tx DMA 1 : SINGLE level-sensitive detection of SPDIF Tx DMA
20	DTTRG	R/W	0	Trigger Type of Tx DMA 0 : SINGLE edge-triggered detection of DAI Tx DMA 1 : SINGLE level-sensitive detection of DAI Tx DMA
19	CRRPT	R/W	0	Repeat Mode Control of CDIF Rx DMA 0 : After all of hop transfer has executed, CDIF Rx DMA channel is disabled. 1 : CDIF Rx DMA channel remains enabled. When another DMA request has occurred, CDIF Rx DMA channel start transfer data again with the same manner (type, address, increment, mask) as the latest transfer of that channel.
18	DRRPT	R/W	0	Repeat Mode Control of DAI Rx DMA 0 : After all of hop transfer has executed, DAI Rx DMA channel is disabled. 1 : DAI Rx DMA channel remains enabled. When another DMA request has occurred, DAI Rx DMA channel start transfer data again with the same manner (type, address, increment, mask) as the latest transfer of that channel.
17	STRPT	R/W	0	Repeat Mode Control of SPDIF Tx DMA 0 : After all of hop transfer has executed, SPDIF Tx DMA channel is disabled. 1 : SPDIF Tx DMA channel remains enabled. When another DMA request has occurred, SPDIF Tx DMA channel start transfer data again with the same manner (type, address, increment, mask) as the latest transfer of that channel.
16	DTRPT	R/W	0	Repeat Mode Control of DAI Tx DMA 0 : After all of hop transfer has executed, DAI Tx DMA channel is disabled. 1 : DAI Tx DMA channel remains enabled. When another DMA request has occurred, DAI Tx DMA channel start transfer data again with the same manner (type, address, increment, mask) as the latest transfer of that channel. The 1 Hop of transfer means 1 burst of read followed by 1 burst of write. 1 burst means 1, 2 or 4 consecutive read or write cycles defined by BSIZE field of CHCTRL register. The Figure 10.6 illustrates 1 Hop of transfer
15-14	CRBSIZE	R/W	2	Burst Size of CDIF Rx DMA 0 : 1 Burst transfer of CDIF Rx DMA consists of 1 read or write cycle. 1 : 1 Burst transfer of CDIF Rx DMA consists of 2 read or write cycles 2 : 1 Burst transfer of CDIF Rx DMA consists of 4 read or write cycles 3 : 1 Burst transfer of CDIF Rx DMA consists of 8 read or write cycles When LRMode is enabled, 8 Read/Write is not available.
13-12	DRBSIZE	R/W	2	Burst Size of DAI Rx DMA 0 : 1 Burst transfer of DAI Rx DMA consists of 1 read or write cycle. 1 : 1 Burst transfer of DAI Rx DMA consists of 2 read or write cycles 2 : 1 Burst transfer of DAI Rx DMA consists of 4 read or write cycles 3 : 1 Burst transfer of DAI Rx DMA consists of 8 read or write cycles When LRMode is enabled, 8 Read/Write is not available.
11-10	STBSIZE	R/W	2	Burst Size of SPDIF Tx DMA 0 : 1 Burst transfer of SPDIF Tx DMA consists of 1 read or write cycle. 1 : 1 Burst transfer of SPDIF Tx DMA consists of 2 read or write cycles 2 : 1 Burst transfer of SPDIF Tx DMA consists of 4 read or write cycles 3 : 1 Burst transfer of SPDIF Tx DMA consists of 8 read or write cycles When LRMode is enabled, 8 Read/Write is not available.
9-8	DTBSIZE	R/W	2	Burst Size of DAI Tx DMA 0 : 1 Burst transfer of DAI Tx DMA consists of 1 read or write cycle. 1 : 1 Burst transfer of DAI Tx DMA consists of 2 read or write cycles

				2 : 1 Burst transfer of DAI Tx DMA consists of 4 read or write cycles 3 : 1 Burst transfer of DAI Tx DMA consists of 8 read or write cycles When LRMode is enabled, 8 Read/Write is not available.
7-6	CRWSIZE	R/W	0	Word Size of CDIF Rx DMA 0 : Each cycle read or write 8bit data of CDIF Rx DMA 1 : Each cycle read or write 16bit data of CDIF Rx DMA 2, 3 : Each cycle read or write 32bit data of CDIF Rx DMA
5-4	DRWSIZE	R/W	0	Word Size of DAI Rx DMA 0 : Each cycle read or write 8bit data of DAI Rx DMA 1 : Each cycle read or write 16bit data of DAI Rx DMA 2, 3 : Each cycle read or write 32bit data of DAI Rx DMA
3-2	STWSIZE	R/W	0	Word Size of SPDIF Tx DMA 0 : Each cycle read or write 8bit data of SPDIF Tx DMA 1 : Each cycle read or write 16bit data of SPDIF Tx DMA 2, 3 : Each cycle read or write 32bit data of SPDIF Tx DMA
1-0	DTWSIZE	R/W	0	Word Size of DAI Tx DMA 0 : Each cycle read or write 8bit data of DAI Tx DMA 1 : Each cycle read or write 16bit data of DAI Tx DMA 2, 3 : Each cycle read or write 32bit data of DAI Tx DMA

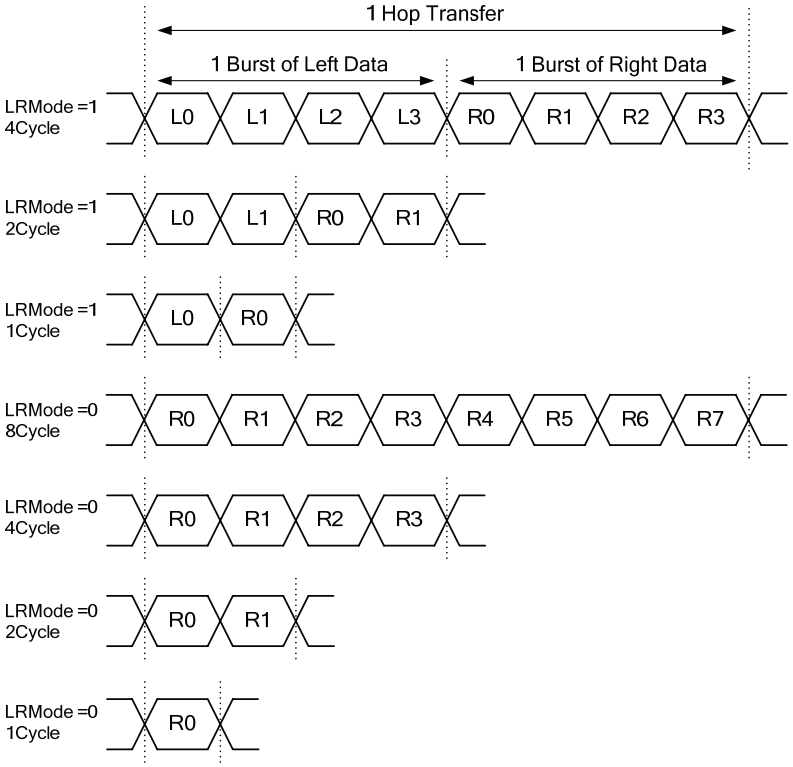


Figure 10.6 Relation between Hop and Burst Transfers

Repeat Control Register 0xB010303C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRI	R*			DBTH				RPTCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RPTCNT															

Field	Name	RW	Reset	Description
31	DRI	R/W	0	Disable Repeat Interrupt 0 : DMA Interrupt is occurred when the end of each Repeated DMA operation 1 : DMA Interrupt occur is occurred when the last DMA Repeated DMA operation This bit is meaningful when Repeat Mode is enabled.
30-28	R*	R	0	Reserved
27-24	DBTH	RW	0	DAI Buffer Threshold n : DMA transfer buffer threshold determines DMA transfer trigger condition This field sets the threshold condition of internal 64-depth buffer for DAI DMA. If audio data increases more than (DBTH+1)*2^DTBSIZE, the internal DMA starts the transfer. When MPE bit is enabled, DBTH should be fixed to 7 and also DTBSIZE should be 2.
23- 0	RPTCNT	R/W	0	Repeat Count 0 : DMA transfer will be repeated endlessly. n : DMA transfer will be repeated (n + 1) * TCOUNT times. This bit is meaningful when Repeat Mode is enabled. When this bit is cleared in repeat mode, DMA transfer will be repeated endlessly. To exit endless repeat mode, clear EN bit of ChCtrl or disable Repeat Mode. It's possible to circular transfer using repeat count.

Channel Control Register 0xB0103078

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CREN	DREN	STEN	DTEN	DSSC	CSS	DRMCM	DTMCM	DMRSEL		DMTSEL		CRLR	DRLR	STLR	DTLR
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CRDW	DREW	STDW	DTDW	CRSEN	DRSEN	STSEN	DTSEN	CRWB	DRWB	STWB	DTWB	CREN	DREN	STEN	DTEN

Field	Name	RW	Reset	Description
31	CREN	R/W	0	DMA Channel Enable of CDIF Rx 0 : CDIF Rx DMA channel is terminated and disabled. It does not affect the TCOUNT register, so if the current hop counter is not zero when channel is disabled, it is possible that the transfer illegally starts right after channel is re-enabled. Make sure that TCOUNT is zero not to continue transfer after channel is re-enabled. 1 : CDIF Rx DMA channel is enabled.
30	DREN	R/W	0	DMA Channel Enable of DAI Rx 0 : DAI Rx DMA channel is terminated and disabled. 1 : DAI Rx DMA channel is enabled.
29	STEN	R/W	0	DMA Channel Enable of SPDIF Tx 0 : SPDIF Tx DMA channel is terminated and disabled. 1 : SPDIF Tx DMA channel is enabled.
28	DTEN	R/W	0	DMA Channel Enable of DAI Tx 0 : DAI Tx DMA channel is terminated and disabled. 1 : DAI Tx DMA channel is enabled.
27	DSSC	R/W	0	DAI and SPDIF are have the same clock 0 : Disable 1 : Enable (DAI Block CLK → SPDIF Block CLK)
26	CSS	R/W	0	CDIF or SPDIF Select 0: CDIF Rx Mode 1: SPDIF Rx Mode
25	DRMCM	R/W	0	DMA Multi Channel Mode of DAI Rx 0 : Single Channel Mode. 1 : Multi Channel Mode.

24	DTMCM	R/W	0	DMA Multi Channel Mode of DAI Tx 0 : Single Channel Mode. 1 : Multi Channel Mode.
23-22	DMRSEL	R/W	0	DMA Multi Channel Select of DAI Rx 00 : DAI0 & DAI1 Channel Select (3.1Ch) 01 : DAI0 & DAI1 & DAI2 Channel Select (5.1Ch) 10 : DAI0 & DAI1 & DAI3 Channel Select (5.1Ch) 11 : DAI0 & DAI1 & DAI2 & DAI3 channel select (7.1Ch)
21-20	DMTSEL	R/W	0	DMA Multi Channel Select of DAI Tx 00 : DAI0 & DAI1 Channel Select (3.1Ch) 01 : DAI0 & DAI1 & DAI2 Channel Select (5.1Ch) 10: DAI0 & DAI1 & DAI3 Channel Select (5.1Ch) 11 : DAI0 & DAI1 & DAI2 & DAI3 channel select (7.1Ch)
19	CRLR	R/W	0	Left/Right Data Mode of CDIF Rx 0 : Disable LRMode of CDIF Rx DMA. 1 : Enable LRMode of CDIF Rx DMA. When LRMode is enabled, audio data will be written to memory separately according to whether it is left or right one.
18	DRLR	R/W	0	Left/Right Data Mode of DAI Rx 0 : Disable LRMode of DAI Rx DMA. 1 : Enable LRMode of DAI Rx DMA.
17	STLR	R/W	0	Left/Right Data Mode of SPDIF Tx 0 : Disable LRMode of SPDIF Tx DMA. 1 : Enable LRMode of SPDIF Tx DMA.
16	DTLR	R/W	0	Left/Right Data Mode of DAI Tx 0 : Disable LRMode of DAI Tx DMA. 1 : Enable LRMode of DAI Tx DMA.
15	CRDW	R/W	0	Width of Audio Data of CDIF Rx 0 : Assume width of CDIF Rx Data is 24bits. 1 : Assume width of CDIF Rx Data is 16bits. AUDIO DMA manipulates audio data as 16bit-length one. If it gets 24bit-length audio data from audio devices, it will cut out lsb-16bits and put those into DMA FIFO. For more details, refer to attached pictures.
14	DRDW	R/W	0	Width of Audio Data of DAI Rx 0 : Assume width of DAI Rx Data is 24bits. 1 : Assume width of DAI Rx Data is 16bits.
13	STDW	R/W	0	Width of Audio Data of SPDIF TX 0 : Assume width of SPDIF Tx Data is 24bits. 1 : Assume width of SPDIF Tx Data is 16bits.
12	DTDW	R/W	0	Width of Audio Data of DAI Tx 0 : Assume width of DAI Tx Data is 24bits. 1 : Assume width of DAI Tx Data is 16bits.
11	CRSEN	R/W	0	Swapping Half-word/Byte align of CDIF Rx 0 : Disable swapping of CDIF Rx DMA 1 : Swaps half-word or byte align according to WB bits of CDIF Rx DMA
10	DRSEN	R/W	0	Swapping Half-word/Byte align of DAI Rx 0 : Disable swapping of DAI Rx DMA 1 : Swaps half-word or byte align according to WB bits of DAI Rx DMA
9	STSEN	R/W	0	Swapping Half-word/Byte align of SPDIF Tx 0 : Disable swapping of SPDIF Tx DMA 1 : Swaps half-word or byte align according to WB bits of SPDIF Tx DMA
8	DTSEN	R/W	0	Swapping Half-word/Byte align of DAI Tx 0 : Disable swapping of DAI Tx DMA 1 : Swaps half-word or byte align according to WB bits of DAI Tx DMA
7	CRWB	R/W	0	Choose Half-word/Byte of CDIF Rx 0 : Byte swapping of CDIF Rx DMA. This is available only when SWP is enabled. 1 : Half-word swapping of CDIF Rx DMA. This is available only when SWP is enabled.
6	DRWB	R/W	0	Choose Half-word/Byte of DAI Rx 0 : Byte swapping of DAI Rx DMA. This is available only when SWP is enabled. 1 : Half-word swapping of DAI Rx DMA. This is available only when

				SWP is enabled.
5	STWB	R/W	0	Choose Half-word/Byte of SPDF Tx 0 : Byte swapping of SPDIF Tx DMA. This is available only when SWP is enabled. 1 : Half-word swapping of SPDIF Tx DMA. This is available only when SWP is enabled.
4	DTWB	R/W	0	Choose Half-word/Byte of DAI Tx 0 : Byte swapping of DAI Tx DMA. This is available only when SWP is enabled. 1 : Half-word swapping of DAI Tx DMA. This is available only when SWP is enabled.
3	CRIEN	R/W	0	Interrupt Enable of CDIF Rx 1 : At the same time the CRI goes to 1, CDIF Rx DMA interrupt request is generated. To generate IRQ or FIQ interrupt, the DMA flag of IEN register in the interrupt controller must be set to 1 ahead.
2	DRIEN	R/W	0	Interrupt Enable of DAI Rx 1 : At the same time the CRI goes to 1, DAI Rx DMA interrupt request is generated.
1	STIEN	R/W	0	Interrupt Enable of SPDIF Tx 1 : At the same time the CRI goes to 1, SPDIF Tx DMA interrupt request is generated.
0	DTIEN	R/W	0	Interrupt Enable of DAI Tx 1 : At the same time the CRI goes to 1, DAI Tx DMA interrupt request is generated.

DMA Interrupt Status Register **0xB010307C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*								CRI	DRI	STI	DTI	CRMI	DRMI	STMI	DTMI

Field	Name	RW	Reset	Description
31-8	R*	R	0	Reserved
7	CRI	R/W	0	DMA Interrupt Status of CDIF(SPDIF) Rx 0 : No interrupt occurred while CDIF Rx DMA transfer. 1 : Interrupt occurred while CDIF Rx DMA transfer. Without regard to Interrupt enable bit(CRIEN) of ChCtrl, this bit indicates the CDIF Rx DMA interrupt status. This bit is automatically cleared when FLAG bit of ChCtrl is cleared. This bit is read only.
6	DRI	R/W	0	DMA Interrupt Status of DAI Rx 0 : No interrupt occurred while DAI Rx DMA transfer. 1 : Interrupt occurred while DAI Rx DMA transfer. Without regard to Interrupt enable bit(DRIEN) of ChCtrl, this bit indicates the DAI Rx DMA interrupt status. This bit is automatically cleared when FLAG bit of ChCtrl is cleared. This bit is read only.
5	STI	R/W	0	DMA Interrupt Status of SPDIF Tx 0 : No interrupt occurred while SPDIF Tx DMA transfer. 1 : Interrupt occurred while SPDIF Tx DMA transfer. Without regard to Interrupt enable bit(STIEN) of ChCtrl, this bit indicates the SPDIF Tx DMA interrupt status. This bit is automatically cleared when FLAG bit of ChCtrl is cleared. This bit is read only.
4	DTI	R/W	0	DMA Interrupt Status of DAI Tx 0 : No interrupt occurred while DAI Tx DMA transfer. 1 : Interrupt occurred while DAI Tx DMA transfer. Without regard to Interrupt enable bit(DTIEN) of ChCtrl, this bit indicates the DAI Tx DMA interrupt status. This bit is automatically cleared when FLAG bit of ChCtrl is cleared. This bit is read only.
3	CRMI	R/W	0	DMA Masked Interrupt Status of CDIF(SPDIF) Rx 1 : Interrupt with enable asserted occurred while CDIF Rx DMA transfer. This bit indicates the CDIF Rx DMA interrupt status when CRIEN is asserted.
2	DRMI	R/W	0	DMA Masked Interrupt Status of DAI Rx 1 : Interrupt with enable asserted occurred while DAI Rx DMA transfer. This bit indicates the DAI Rx DMA interrupt status when DRIEN is asserted.
1	STMI	R/W	0	DMA Masked Interrupt Status of SPDIF Tx 1 : Interrupt with enable asserted occurred while SPDIF Tx DMA transfer. This bit indicates the SPDIF Tx DMA interrupt status when STIEN is asserted.
0	DTMI	R/W	0	DMA Masked Interrupt Status of DAI Tx 1 : Interrupt with enable asserted occurred while DAI Tx DMA transfer. This bit indicates the DAI Tx DMA interrupt status when DTIEN is asserted.

General Interrupt Register (Status Register) **0xB0103080(0x84)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*								GSTDI	GSTUI	GSTI	GSRDI	GSRI	GDTI	GDRI	GCRI
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R*											GDSTDI	GDSTUI	GDDTI	GDDRI	GDCRI

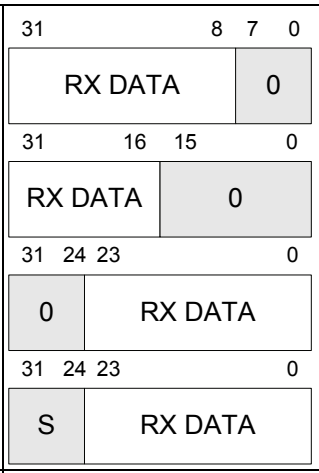
Field	Name	RW	Reset	Description
31-24	R*	-	0	Reserved
23	GSTDI	-	-	General Interrupt Enable of SPDIF Data Tx 0 : Disable 1 : Enable
22	GSTUI	-	-	General Interrupt Enable of SPDIF User Tx 0 : Disable 1 : Enable
21	GSTI	-	-	General Interrupt Enable of SPDIF Tx 0 : Disable 1 : Enable
20	GSRDI	-	-	General Interrupt Enable of SPDIF Data Rx 0 : Disable 1 : Enable
19	GSRI	-	-	General Interrupt Enable of SPDIF Rx 0 : Disable 1 : Enable
18	GDTI	-	-	General Interrupt Enable of DAI Tx 0 : Disable 1 : Enable
17	GDRI	-	-	General Interrupt Enable of DAI Rx 0 : Disable 1 : Enable
16	GCRI	-	-	General Interrupt Enable of CDIF Rx 0 : Disable 1 : Enable
15-5	R*	-	0	Reserved
4	GDSTDI	R/W	0	General DMA Interrupt Enable of SPDIF Data Tx 0 : Disable 1 : Enable
3	GDSTUI	R/W	0	General DMA Interrupt Enable of SPDIF User Tx 0 : Disable 1 : Enable
2	GDDTI	R/W	0	General DMA Interrupt Enable of DAI Tx 0 : Disable 1 : Enable
1	GDDRI	R/W	0	General DMA Interrupt Enable of DAI Rx 0 : Disable 1 : Enable
0	GDCRI	R/W	0	General DMA Interrupt Enable of CDIF Rx 0 : Disable 1 : Enable

10.2.2 DAI Register

Digital Audio Mode Register (DAMR) **0xB0104040**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BPS	LPS	DBTEN	MPE	AFE	DDL	DSP	NEWR	NEWT	RXE	RXS		TXS		LBT	SP
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EN	TE	RE	MD	SM	BM	FM	CC	BD		FD		BP	CM	MM	LB

Field	Name	RW	Reset	Description
31	BPS	R/W	0	BCLK Pad Select 0 : BCLK source select BCLK Pad 1 : BCLK direct at master mode
30	LPS	R/W	0	LRCK Pad Select 0 : LRCK source select LRCK Pad 1 : LRCK direct at master mode
29	DBTEN	R/W	0	DAI Buffer Threshodl Enable 0 : Disable 1 : Enable This bit enables DAI buffer threshold (DBTH) control register. If it is set to low, DAI DMA transfer starts right after new data is stored to DAI buffer. When MPE bit is set to high, this field should be set to high not to get wrong transfer in multi-port condition.
28	MPE	R/W	0	Multi-Port Enable 0 : Disable 1 : Enable In multi-port condition, DAI DMA transfer mode is fixed to one mode. That is, DBTEN, DBTH, DTBSIZE should be fixed to 1, 7 and 2 each.
27	AFE	R/W	0	Audio Filter Enable 1 : Audio filters will be enabled. When CD BCLK, CD LRCK, DAI MCLK, DAI BCLK and DAI LRCK are exposed to noises, you may suppress them by enable this bit. Audio Filter clock has to be enabled before enabling this bit and be at least four times faster than CD BCLK or DAI MCLK. 0 : Audio filters will be disabled.
26	DDL	R/W	0	DSP Mode Word Length 0 : Data Length 24bit 1 : Data Length 16bit This selects the number of bit width in Wolfson DSP & TDM mode. In Wolfson TDM mode, MCCR1 must be set additionally.
25	DSP	R/W	0	DSP Mode 0 : IIS 1 : DSP or TDM Mode MD =‘0’ & DSP=‘1’→ DSP Mode MD =‘0’ & DSP=‘1’& FD= “11” → TDM Mode
24	NMDR	R/W	0	Rx Justfied Mode 0 : Rx Left-justified Mode 1 : Rx Right-justified Mode MD =‘1’ & DSP=‘0’ & NMDR=‘0’ → Left-justified Rx Mode MD =‘1’ & DSP=‘0’ & NMDR=‘1’ → Right-justified Rx Mode
23	NMDT	R//W	0	Tx Justified Mode 0 : Tx Left-justified Mode 1 : Tx Right-justified Mode MD =‘1’ & DSP=‘0’ & NMDT=‘0’ → Left-justified Tx Mode MD =‘1’ & DSP=‘0’ & NMDR=‘1’ → Right-justified Tx Mode
22	RXE	R/W	0	DAI RX Data Sign Extension 0 : Disable (zero extension) 1 : Enable (sign bit extension)
21-20	RXS	R/W	0	DAI Rx Shift 0 : Bit-pack MSB and 24bit mode. 1 : Bit-pack MSB and 16bit mode. 2 : Bit-pack LSB and 24bit mode. 3 : Bit-pack LSB and 16bit mode.

				
19-18	TXS	R/W	0	DAI Tx Shift 0 : Bit-pack MSB mode. 1 : Bit-pack MSB mode. 2 : Bit-pack LSB and 24bit mode. 3 : Bit-pack LSB and 16bit mode. If DMA transfer bit width is 16, DAI TX Shift register should be set to "Bit-pack LSB and 16bit mode"(TXS=3). When DAI TX Shift register is "Bit-pack LSB and 24bit mode", Most Significant Byte of the second 16bit is filled with 0.
17	LBT	R/W	0	Loop Back Test(Tx to Rx) 0 : Disable 1 : Enable
16	SP	R/W	0	DAI System Clock Polatity 0 : Disable 1 : Enable
15	EN	R/W	0	DAI Master Enable 0 : Disable 1 : Enable
14	TE	R/W	0	DAI Transmitter Enable 0 : Disable 1 : Enable
13	RE	R/W	0	DAI Receiver Enable 0 : Disable 1 : Enable
12	MD	R/W	0	DAI Sync Mode 0 : Set DAI sync as IIS, DSP or TDM sync mode 1 : Set DAI sync as Left/Right-justified mode
11	SM	R/W	0	DAI System Clock Master Select 0 : Set that DAI system clock is come from external pin 1 : Set that DAI system clock is generated by the clock generator block The DAI system clock in clock generator is known as DCLK. Its frequency can be determined by PCK_DAI from the CKC.
10	BM	R/W	0	DAI Bit Clock Master Select 0 : Set that DAI bit clock is from external pin 1 : Set that DAI bit clock is generated by dividing DAI system clock
9	FM	R/W	0	DAI Frame Clock Master Select 0 : Set that DAI frame clock is come from external pin 1 : Set that DAI frame clock is generated by dividing DAI bit clock
8	CC	R/W	0	CDIF Clock Select. BLCK and LRCK would be from CDIF under master mode. This is usually associated with CDIF Monitor Mode 0 : Disable CDIF Clock master mode 1 : Enable CDIF Clock master mode
7-6	BD	R/W	0	DAI Bit Clock Divider Select 0 : Select Div 4 (256fs->64fs) 1 : Select Div 6 (384fs->64fs) 2 : Select Div 8 (512fs->64fs, 384fs->48fs , 256fs->32fs) 3 : Select Div16 (512fs->32fs)
5-4	FD	R/W	0	DAI Frame Clock Divider Select 0 : Select Div 32 (32fs->fs) 1 : Select Div 48 (48fs->fs)

				2 : Select Div 64 (64fs->fs) 3 : Select Div use (xfs->fs) ← Only TDM Mode The combination of BD & FD determines that the ratio between main system clock and the sampling frequency. The multiplication between the division factor of BD and FD must be equal to this ratio. In TDM mode, FD should be set to 3.
3	BP	R/W	0	DAI Bit Clock Polarity 0 : Set that data is captured at positive edge of bit clock 1 : Set that data is captured at negative edge of bit clock
2	CM	R/W	0	CDIF Monitor Mode 0 : Disable 1 : Enable. Data bypass from CDIF
1	MM	R/W	0	DAI Monitor Mode 0 : Disable 1 : Enable. Transmitter should be enabled. (TE = 1)
0	LB	R/W	0	DAI Loop-back Mode 0 : Disable DAI Loop back mode 1 : Enable DAI Loop back mode

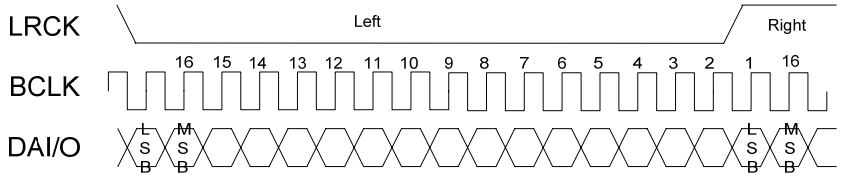
Digital Audio Volume Control Register (DAVC) 0xB0104044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R*											VC<4:0>				

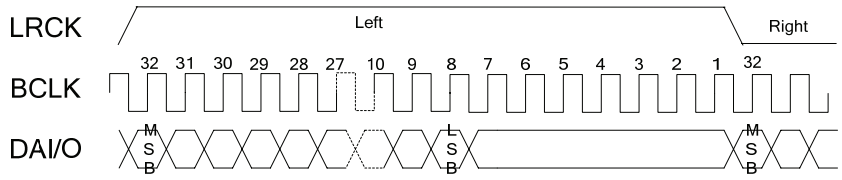
The volume of audio output can be manipulated by this register. It has –6dB unit so the output volume can be set from 0 dB to –90 dB as the following table.

Field	Name	RW	Reset	Description
31-5	R*	-	0	Reserved
4-0	VC	R/W	0	DAI Volume Control

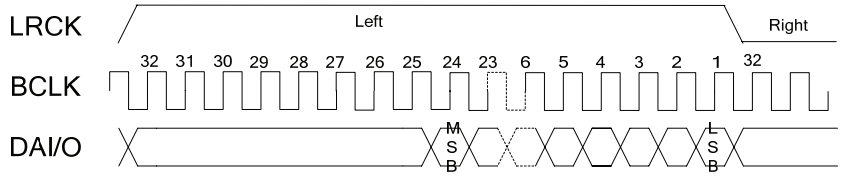
00000	0dB
00001	-6dB
00010	-12dB
00011	-18dB
00100	-24dB
00101	-30dB
00110	-36dB
00111	-42dB
01000	-48dB
01001	-54dB
01010	-60dB
01011	-66dB
01100	-72dB
01101	-78dB
01110	-84dB
01111	-90dB
10000	-96dB



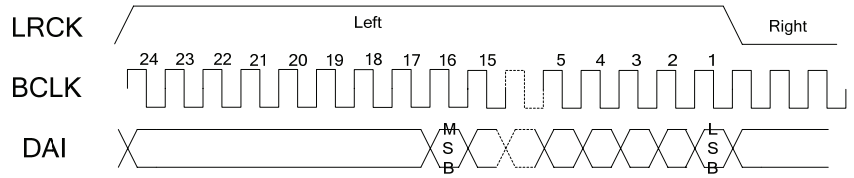
NMDR(T)=0, DSP=0, MD=0 (IIS mode), BP=0, BCLK = 32fs



NMDR(T)=0, DSP=0, MD=1(MSB justified mode), BP=0, BCLK=64fs



NMDR(T)=1, DSP=0, MD=1(LSB justified mode), BP=0, BCLK=64fs



NMDR(T)=1, DSP=0, MD=1(MSB justified mode), BP=1, BCLK=48fs

Figure 10.7 DAI Bus Timing Diagram

Multi Channel Control Register 0 (MCCR0) 0xB0104048

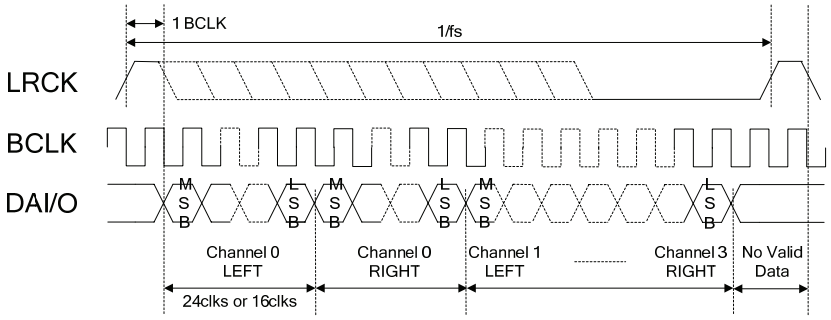
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DAO0M	DAO1M	DAO2M	DAO3M	R*		FBP	FEP								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TFI		TBD		FD		FI	FS								

Field	Name	RW	Reset	Description
31	DAO0M	R/W	0	DAO0 mask 0: Disable 1: Enable
30	DAO1M	R/W	0	DAO1 mask 0: Disable 1: Enable
29	DAO2M	R/W	0	DAO2 mask 0: Disable 1: Enable
28	DAO3M	R/W	0	DAO3 mask 0: Disable 1: Enable
27-26	R*	-	0	Reserved
25	FBP	R/W	0	Frame Begin Position(DSP & TDM) 0 : Frame Begin Position → Early Mode 1 : Frame Begin Position → Late Mode
24-16	FEP	R/W	0	Frame End Position(DSP & TDM) n : Frame pulse ends after n base clock has generated
15-14	TFI	R/W	0	Format Interface of TDM 00 : TDM Mode 0 (CIRRUSS, DSP Mode like) 01 : TDM Mode 1 (Wolfson, DSP Mode like) 10 : TDM Mode 2 (I2S Mode like)
13-12	TBD	R/W	0	Bit Clock Divider Select of TDM 00 : Disable 01 : Select Div 1 (256fs -> 256fs) 10 : Select Div 2 (512fs -> 256fs, 256fs -> 128fs)
11-10	FD	R/W	0	Frame Clock Divider Select (DSP & TDM) 00 : Select Div 32(32fs -> fs) 01 : Select Div 48 (48fs -> fs) 10 : Select Div 64 (64fs -> fs) 11 : Select Div use (xfs -> fs)
9	FI	R/W	0	Frame Invert(DSP & TDM) 0 : Disable 1 : Enable
8-0	FS	R/W	0	Frame Size(DSP & TDM) n : Frame pulse ends after n base clock has generated In DSP or TDM mode, the number of audio samples in a single LRCK duration

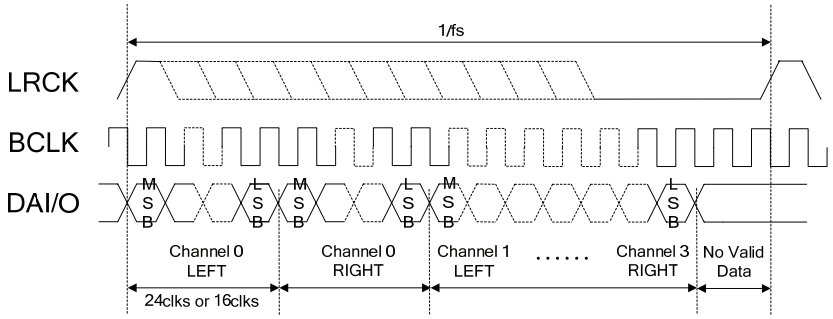
Multi Channel Control Register 1 (MCCR1) 0xB010404C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R*							VDE								

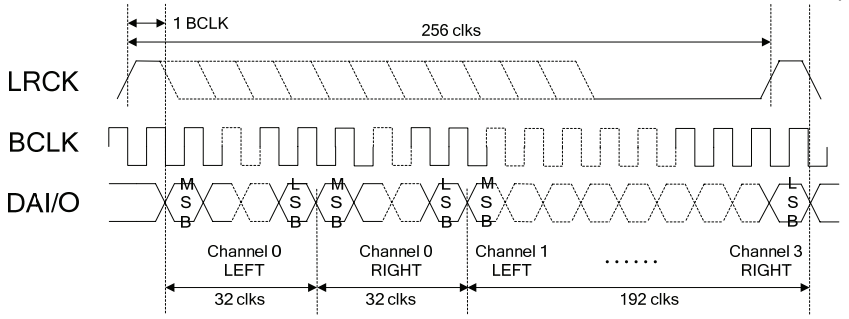
Field	Name	RW	Reset	Description
31-9	R*	R	0	Reserved
8-0	VDE	R/W	0	n : Valid Data ends after n base clock has generated When in TDM mode 1, VDE bits sets the valid data length out of 16bit or 24bit input data.



(a) TDM Mode Example 1 (Wolfson – late mode)
(DAMR → DSP=1, MD=0, FD=3, BP=1
MCCR0 → FBP=0, TFI=1(W), TBD=1, FD=11
MCLK1=BCLK=256fs)



(b) TDM Mode Example 2 (Wolfson – early mode)
(DAMR → DSP=1, MD=0, FD=3, BP=1
MCCR0 → FBP=1, TFI=1(W), TBD=1, FD=11
MCLK1=BCLK=256fs)



(c) TDM Mode Example 3 (Cirrus)
(DAMR → DSP=1, MD=0, FD=3, BP=1
MCCR0 → FBP=0, TFI=0(C), TBD=1, FD=11
MCLK1=BCLK=256fs)

Figure 10.8 DAI TDM Mode Timing Diagram

10.2.3 CDIF Registers

CD Data Input (CDDI0~CDDI7) **0xB0104080 ~ 0xB010409C**

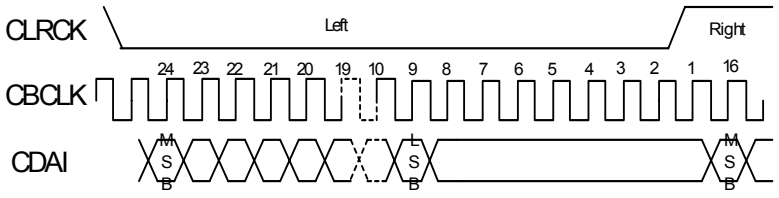
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Left Channel Data															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Right Channel Data															

Field	Name	RW	Reset	Description
31-16	LDATA	R	0	Left Channel Data
15-0	RDATA	R	0	Right Channel Data

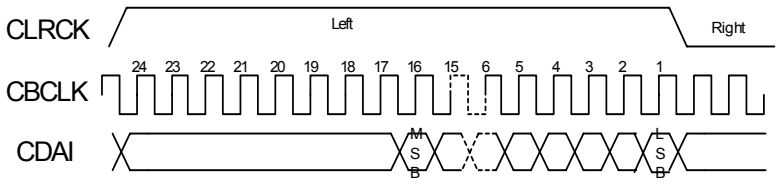
CD Interface Control Register (CICR) **0xB01040A0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*								EN	Rserved			BS		MD	BP

Field	Name	RW	Reset	Description
31-8	R*	R	0	Reserved
7	EN	R/W	0	0 : Disable CDIF 1 : Enable CDIF
6-4	R	R	0	Reserved
3-2	BS	R/W	0	CDIF Bit Clock Select 00 : 64fs 01 : 32fs 10 : 48fs
1	MD	R/W	0	Interface Mode Select 0 : Select IIS format 1 : Select LSB justified format
0	BP	R/W	0	CDIF Bit Clock Polarity 0 : Set that data is captured at positive edge of bit clock 1 : Set that data is captured at negative edge of bit clock



MD=0 (IIS mode), BP=0, CBCLK=48fs



MD=1(LSB justified mode), BP=0, CBCLK=48fs

Figure 10.9 CDIF Bus Timing Diagram

10.2.4 SPDIF Registers

SPDIF Transmitter Version Register (TxVersion) **0xB010B000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*			CSB	UDB	AW						DW	VER			

Field	Name	RW	Reset	Description
31-13	R*	R	0	Reserved
12	CSB	R	1	0 : Channel status buffer is not available 1 : Channel status buffer is available
11	UDB	R	1	0 : User data buffer is not available 1 : User data buffer is available
10-5	AW	R	1	n : Value of Address Width
4	DW	R	1	n : Value of Data Width
3-0	VER	R	1	n : Version Number

SPDIF Transmit Configuration Register (TxConfig) **0xB010B004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*								MODE				R*			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RATIO								UDATEN		CHSTEN		R*	IEN	TXD	TXEN

Field	Name	RW	Reset	Description
31-24	R*	R	0	Reserved
23-20	MODE	R/W	0	n : 16 + n Bits Transmit Sample Format 9 ~ 15 : Reserved
19-16	R*	R	0	Reserved
15-8	RATIO	R/W	0	Clock Divider Ratio n : Clock divider for the transmit frequency. The SPDIF clock is divided by a factor of (RATIO + 1) to generate the serial transmit clock.
7-6	UDATEN	R/W	0	User Data Enable Bits 0 : User data A & B set to 0. 1 : User data A & B generated from UserData bit 7-0 2 : User data A generated from UserData bit 7-0, B generated from UserData bit 15-8 3 : Reserved
5-4	CHSTEN	R/W	0	Channel Status Enable Bits 0 : Channel status A & B generated from TxChStat 1 : Channel status A & B generated from ChStat bit 7-0 2 : Channel status A generated from ChStat bit 7-0, B generated from ChStat bit 15-8 3 : Reserved
3	R*	R	0	Reserved
2	IEN	R/W	0	Interrupt Output Enable Bit n : ‘1’ for enabling the interrupt output.
1	TXD	R/W	0	Data Valid Bit n : ‘1’ for data being valid.
0	TXEN	R/W	0	Transmitter Enable Bit n : ‘1’ for enabling the transmission.

SPDIF Transmit Channel Status Control Register (TxChStat) 0xB010B008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*					SCS	SUD	SV	FREQ		Reserved		GSTS	PRE	CPY	AU

Field	Name	RW	Reset	Description
31-11	R*	R	0	Reserved
10	SCS	R/W	0	Store Channel Status Bit 0 : No Store 1 : Store
9	SUD	R/W	0	Store User Data Bit 0 : No Store 1 : Store
8	SV	R/W	0	Store Validity Bit 0 : No Store 1 : Store
7-6	FREQ	R/W	0	Sampling Frequency 0 : 44.1 KHz 1 : 48 KHz 2 : 32 KHz 3 : Sample Rate Converter
5-4	R*	R	0	Reserved
3	GSTS	R/W	0	Status Generation 0 : No indication 1 : Original/Commercially Pre-recorded data
2	PRE	R/W	0	Pre-emphasis 0 : None 1 : 50/15us
1	CPY	R/W	0	Copyright 0 : Copy inhibited 1 : Copy permitted
0	AU	R/W	0	Data Format 0 : Audio Format 1 : Data Format

SPDIF Transmit Interrupt Mask Register (TxIntMask) 0xB010B00C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*											HCSB	LCSB	HSB	LSB	R*

Field	Name	RW	Reset	Description
31-5	R*	R	0	Reserved
4	HCSB	R/W	0	Higher Channel Status/User Data Buffer Empty '1' for enable for higher channel status/user data buffer empty interrupt
3	LCSB	R/W	0	Lower Channel Status/User Data Buffer Empty '1' for enable for lower channel status/user data buffer empty interrupt
2	HSB	R/W	0	Higher Data Buffer Empty '1' for enable for higher data buffer empty interrupt
1	LSB	R/W	0	Lower Data Buffer Empty '1' for enable for lower data buffer empty interrupt
0	R	R	0	Reserved

SPDIF Transmit Interrupt Status Register (TxIntStat) 0xB010B010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*											HCSB	LCSB	HSB	LSB	R*

Field	Name	RW	Reset	Description
31-5	R*	R	0	Reserved
4	HCSB	R/W	0	Higher Channel Status/User Data Buffer Empty '1' for higher channel status/user data buffer empty interrupt activated
3	LCSB	R/W	0	Lower Channel Status/User Data Buffer Empty '1' for lower channel status/user data buffer empty interrupt activated
2	HSB	R/W	0	Higher Data Buffer Empty '1' for higher data buffer empty interrupt activated
1	LSB	R/W	0	Lower Data Buffer Empty '1' for lower data buffer empty interrupt activated
0	R*	R	0	Reserved

SPDIF Transmit User Data Buffer Register (UserData) 0xB010B080 ~ 0xB010B0DC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHBUD								CHAUD							

The user data is 24 bytes per sample block. Bit 0 of byte 0 is transmitted first. Bit 7 of byte 23 is transmitted last.

Field	Name	RW	Reset	Description
31-16	R*	R	0	Reserved
15-8	CHBUD	R/W	0	User Data for Channel B
7-0	CHAUD	R/W	0	User Data for Channel A

SPDIF Transmit Channel Status Buffer Register (ChStatus) 0xB010B100 ~ 0xB010B15C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHBCS								CHACS							

The channel status data is 24 bytes per sample block. Bit 0 of byte 0 is transmitted first. Bit 7 of byte 23 is transmitted last.

Field	Name	RW	Reset	Description
31-16	R*	R	0	Reserved
15-8	CHBCS	R/W	0	Channel Status for Channel B
7-0	CHACS	R/W	0	Channel Status for Channel A

SPDIF Transmit Sample Data Buffer Register (TxBuffer) 0xB010B200 ~ 0xB010B3FC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*								DATH							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATL															

The format of data words in transmit sample buffer. Channel A is transmitted first, and must be stored on even addresses, while channel B is stored on odd addresses in the sample buffer.

Field	Name	RW	Reset	Description
31-2	R*	R	0	Reserved
23-6	DATH	R/W	0	Upper 8 Bits for Sample Buffer Data Audio data if > 16 bits resolution. Unused bits are 0
15-0	DATL	R/W	0	Lower 16 Bits for Sample Buffer Data Audio data (16 bits mode). Bits 0 is LSB

DMA Configuration (DMACFG) 0xB010B400

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*											VCNT				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*	SS	RALR		DRQEN1	DRQEN0	AMODE1	AMODE0	FIFOCLR	R*			FIFOTH			

Field	Name	RW	Reset	Description
31-21	R*	R	0	Reserved
20-16	VCNT	R	X	FIFO Valid Entry Count
15	R*	R	0	Reserved
14	SS	R/W	0	Swap Sample 0 : Disable 1 : Enable
13-12	RALR	R/W	0	Read Address LR Mode 0 : Read Address 24bit Mode 1 : Read Address 24bit LR Mode 2 : Read Address 16bit Mode 3 : Read Address 16bit LR Mode
11	DRQEN1	R/W	0	DMA Request Enable for User Data Buffer
10	DRQEN0	R/W	0	DMA Request Enable for Sample Data Buffer
9	AMODE1	R/W	0	Sample Data Buffer Address Mode. Buffer would be written with sequence below. (0, 2, 4, 6, 1, 3, 5, 7)
8	AMODE0	R/W	0	0 : Ignore Address (FIFO Mode) 1 : Enable Address (16 Entries can be written with address)
7	FIFOCLR	R/W	0	Clear FIFO. Should be written with “0” for normal operation
6-4	R*	R	0	Reserved
3-0	FIFOTH	R/W	0x7	FIFO Threshold for DMA Request DMA request will be asserted if VCNT <= FIFOTH

SPDIF Receiver Version Register (RxVersion) 0xB010B800

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*												CSC			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R*				AW							DW	VER			

The version register allows the SW to read out all the parameter that was used to generate the receiver.

Field	Name	RW	Reset	Description
31-20	R*	R	0	Reserved
19-16	CSC	R	0x08	Channel Status Buffer Available Bit n : Value of Channel Status Capture
15-12	R*	R	0	Reserved
11-5	AW	R	0x08	Value of Address Width
4	DW	R	1	Value of Data Width 0 : Data Width is 16bit 1 : Data Width is 24bit
3-0	VER	R	1	Version Number

SPDIF Receiver Configuration Register (RxConfig) **0xB010B804**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*							BLKEN	MODE				PAREN	STATEN	USEREN	VALEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R*											VALID	CHAS	RINTEN	SAMPLE	RXEN

Field	Name	RW	Reset	Description
31-25	R*	R	0	Reserved
24	BLKEN	R/W	0	Block Boundary Marking 0 : Do not use block boundary marking 1 : Mark the first sample in each block with a 1 in bit 27
23-20	MODE	R/W	0	Store Samples as Bit 0 : Store samples as 16bit 1 : Store samples as 17bit 2~6 : Store samples as 18, 19, 20, 21, 22bits 7 : Store samples as 23bit 8 : Store samples as 24bit 9~15 : Reserved
19	PAREN	R/W	0	Store Parity Bit 0 : Do not use block boundary marking 1 : Store parity bit 31 in sample buffer
18	STATEN	R/W	0	Store Channel Status Bit 0 : Do not store channel status bit 1 : Store channel status bit in bit 30 in sample buffer
17	USEREN	R/W	0	Store User Data Bit 0 : Do not store user data bit 1 : Store user data bit in bit 29 in sample buffer
16	VALEN	R/W	0	Store Validity bit 0 : Do not store validity bit 1 : Store validity bit in bit 28 in sample buffer
15-5	R*	R	0	Reserved
4	VALID	R/W	0	Sample Dta Sore 0 : Sample data stored in buffers regardless of sub_frame Vlidity bit 1 : Sample data stored only when sub_frame validity bit is 0
3	CHAS	R/W	0	RxStatus Register Holds Channel 0 : RxStatus register holds status from channel B 1 : RxSatus register holds status from channel A
2	RINTEN	R/W	0	Interrupt Output 0 : Interrupt output is disabled 1 : Interrupt output is enabled
1	SAMPLE	R/W	0	Stored Data 0 : No data is stored in the sample buffer 1 : Data is stored in the sample buffer
0	RXEN	R/W	0	Receiver Disable Enable 0 : Receiver is disabled 1 : Receiver is enabled

SPDIF Receiver Status Register (RxStatus) **0xB010B808**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*									COPY	EMPH			AUDIO	PRO	LOCK

The signal status register holds information about the received signal. When the receiver is disable, all bits read 0. When lock bit is 0, all other bits are set to 0.

Field	Name	RW	Reset	Description
31-17	R*	R	0	Reserved
6	COPY	R	0	Copy Information 0 : Copy inhibited 1 : Copy permitted (copy bit only applicable in consumer mode)
5-3	EMPH*	R	0	Emphasis Code - Emphasis code from the channel status block. Note that the interpretation of these bits is different in consumer and professional mode.
2	AUDIO	R	0	Signal Data 0 : Signal is non-audio 1 : Signal is audio
1	PRO	R	0	Signal format 0 : Signal format is consumer 1 : Signal format is professional
0	LOCK	R	0	Lock to SPDIF Signal 0 : Receiver has no valid input signal 1 : Receiver is locked to SPDIF signal

SPDIF Receiver Interrupt Mask Register (RxIntMask) 0xB010B80C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*					DBS		CRD	CAP7	CAP6	CAP5	CAP4	CAP3	CAP2	CAP1	CAP0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R*											PEB	PEA	HSBF	LSBF	LOCK

Field	Name	RW	Reset	Description
31-27	R*	R	0	Reserved
26-25	DBS	R/W	0	DMA Burst Select 0 : DMA 1 Burst 1 : DMA 2 Burst 2 : DMA 4 Burst 3 : DMA 8 Burst
24	CRD	R/W	0	Capture Register DMA Interrupt Mask 0 : Disable 1 : Enable
23	CAP7	R/W	0	Capture Register 7 Interrupt Mask 0 : Disable 1 : Enable
22	CAP6	R/W	0	Capture Register 6 Interrupt Mask 0 : Disable 1 : Enable
21	CAP5	R/W	0	Capture Register 5 Interrupt Mask 0 : Disable 1 : Enable
20	CAP4	R/W	0	Capture Register 4 Interrupt Mask 0 : Disable 1 : Enable
19	CAP3	R/W	0	Capture Register 3 Interrupt Mask 0 : Disable 1 : Enable
18	CAP2	R/W	0	Capture Register 2 Interrupt Mask 0 : Disable 1 : Enable
17	CAP1	R/W	0	Capture Register 1 Interrupt Mask 0 : Disable 1 : Enable
16	CAP0	R/W	0	Capture Register 0 Interrupt Mask 0 : Disable 1 : Enable
15-5	R*	R	0	Reserved
4	PEB	R/W	0	Parity Error Channel B Interrupt Mask 0 : Disable 1 : Enable
3	PEA	R/W	0	Parity Error Channel A Interrupt Mask 0 : Disable 1 : Enable
2	HSBF	R/W	0	Higher Sample Buffer Full Interrupt Mask 0 : Disable 1 : Enable
1	LSBF	R/W	0	Lower Sample Buffer Full Interrupt Mask 0 : Disable 1 : Enable
0	LOCK	R/W	0	Change in Lock Bit Interrupt Mask 0 : Disable 1 : Enable

SPDIF Receiver Interrupt Status Register (RxIntStat) **0xB010B810**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*							CRD	CAP7	CAP6	CAP5	CAP4	CAP3	CAP2	CAP1	CAP0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R*											PEB	PEA	HSBF	LSBF	LOCK

Field	Name	RW	Reset	Description
31-25	R*	R	0	Reserved
24	CRD	R/W	0	Capture Register DMA Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
23	CAP7	R/W	0	Capture Register 7 Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
22	CAP6	R/W	0	Capture Register 6 Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
21	CAP5	R/W	0	Capture Register 5 Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
20	CAP4	R/W	0	Capture Register 4 Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
19	CAP3	R/W	0	Capture Register 3 Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
18	CAP2	R/W	0	Capture Register 2 Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
17	CAP1	R/W	0	Capture Register 1 Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
16	CAP0	R/W	0	Capture Register 0 Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
15-5	R*	R	0	Reserved
4	PEB	R/W	0	Parity Error Channel B Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
3	PEA	R/W	0	Parity Error Channel A Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
2	HSBF	R/W	0	Higher Sample Buffer Full Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
1	LSBF	R/W	0	Lower Sample Buffer Full Interrupt 0 : Disable 1 : Enable A bit in this register is set to '1' when an event occurs
0	LOCK	R/W	0	Change in Lock Bit Interrupt 0 : Disable 1 : Enable

				A bit in this register is set to '1' when an event occurs

If the corresponding bit in RxIntMask is set to 1, an interrupt is generated (if enabled). Write 1 to a bit to clear the event. The interrupt signal goes inactive when all events been cleared

SPDIF Receiver Channel Status Capture [n] Register (RxCapCtl [n]) 0xB010B840 ~ 0xB010B87C(even)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITPOS								CDATA	CHID	BITLEN					

Field	Name	RW	Reset	Description
31-16	R*	R	0	Reserved
15-8	BITPOS	R/W	0	First Bit Position of Bt-Filed to be Captured n : Valid Range 0 to 191
7	CDATA	R/W	0	DATA Captured 0 : User data is captured 1 : Channel status data is captured
6	CHID	R/W	0	Source Channel 0 : Source is channel A 1 : Source is channel B
5-0	BITLEN	R/W	0	Bit Length 0 : Capture function disable 1~32 : Length of bit-field to be captured

The channel status capture register can be set up to capture a specified bit-field of the channel status frame or user data frame, and also generate an interrupt when the bit-field changes. The bit-field can be from 1 to 32 bits long. There can be up to eight sets of bit-field capture registers

SPDIF Receiver Channel Status Data [n] Register (RxCap [n]) 0xB010B840~0xB010B87C(odd)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA															

Field	Name	RW	Reset	Description
31-0	DATA	R	0	Captured Channel Status n : First bit captured is stored in bit 0 Captured channel status/user data bits

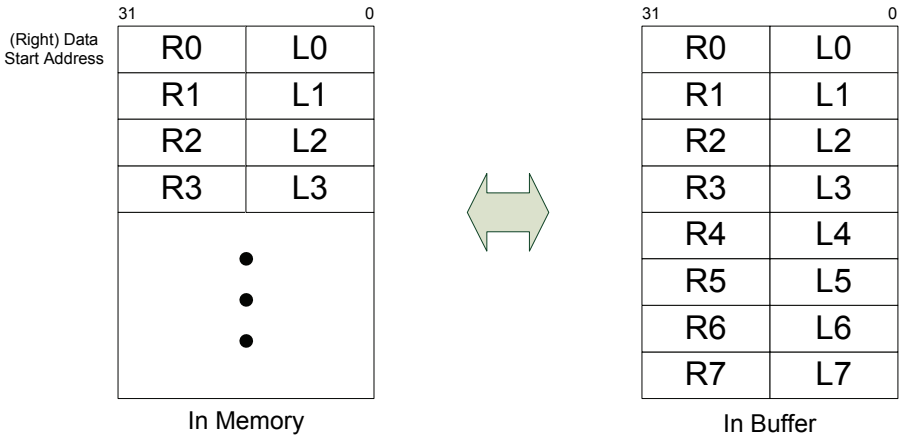
SPDIF Receiver Sample Data Buffer (RxBuffer) **0xB010BA00 ~ 0xB010BA1C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PARITY	CHSTAT	USRDAT	VALID	BLKS	R*			DATAH							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATAL															

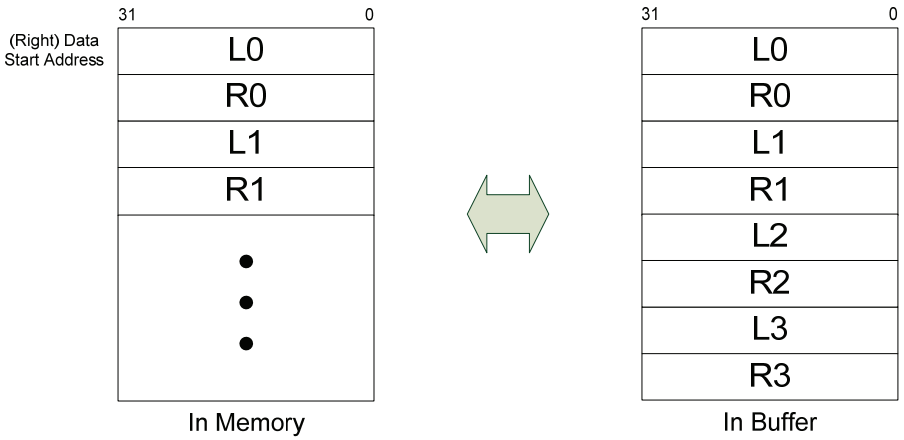
Format of data words in receive sample buffer

Field	Name	RW	Reset	Description
31	PARITY	R	0	Parity Bit If enable, otherwise 0
30	CHSTAT	R	0	Channel Status Bit If enable, otherwise 0
29	USRDAT	R	0	User Data Bit If enable, otherwise 0
28	VALID	R	0	Validity Bit If enable, otherwise 0
27	BLKS	R	0	Start of Sample Block Bit If enable, otherwise 0
26-24	R*	R	0	Reserved
23-16	DATAH	R	0	DATA High n : Audio Data(if > 16bit Resolution) - Unused bit are 0
15-0	DATAL	R	0	DATA Low n : Audio Data - Bit 0 is LSB

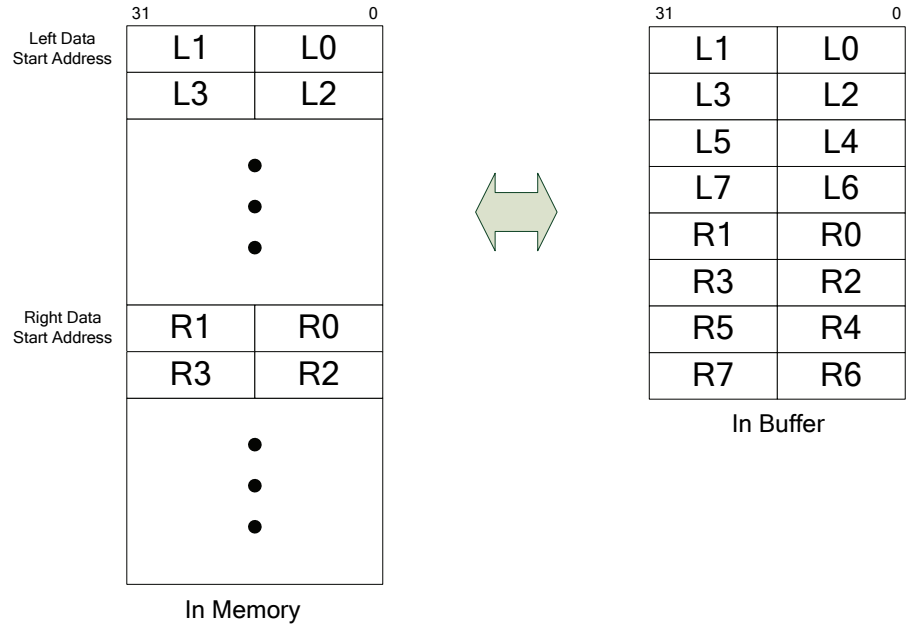
10.2.5 Audio Data Format



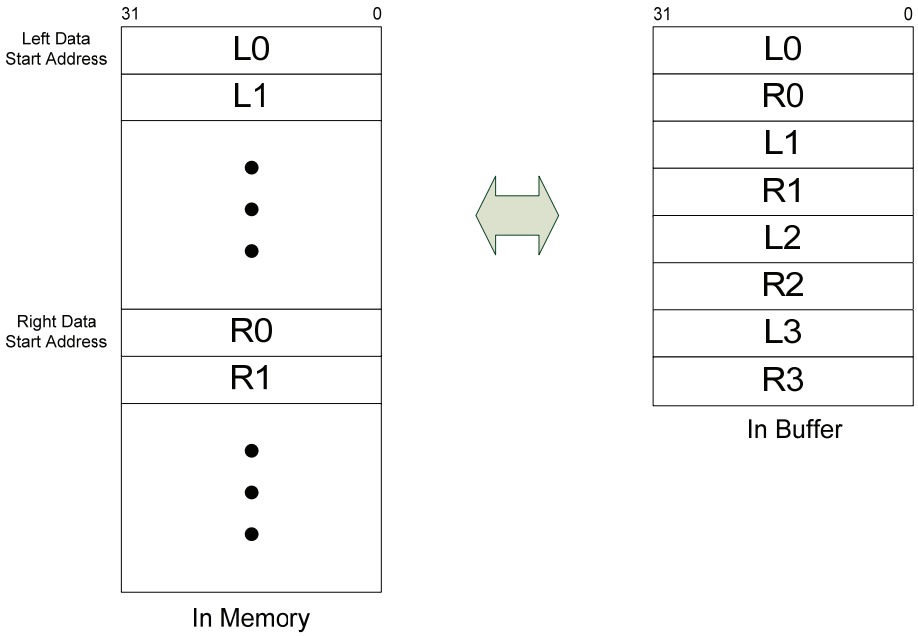
LRMode=0, Data_Width=16 bit



LRMode=0, Data_Width=24 bit



LRMode=1, Data_Width=16 bit



LRMode=1, Data_Width=24 bit

Figure 10.10 Data Format between Memory and Buffer

11 Audio1 (Stereo)

11.1 Overview

11.1.1 AUDIO DMA

AUDIO DMA is a specialized DMA for Audio Interfaces such as DAI, CDIF and SPDIF. A general DMA is also available. When a general DMA is used for transferring audio data, it has to read from memory and write to Audio Interface registers. It seems to have redundant bus cycle time from the viewpoint of time-critical applications. So we applied a dedicated DMA to Audio Interface to reduce redundant bus cycle time. AUDIO DMAs enable you to utilize audio data for more flexible purpose. The structure of AUDIO DMA is similar to that of general DMA. You may use a general DMA instead of AUDIO DMA. You are not recommended to use AUDIO DMA and general DMA at the same time.

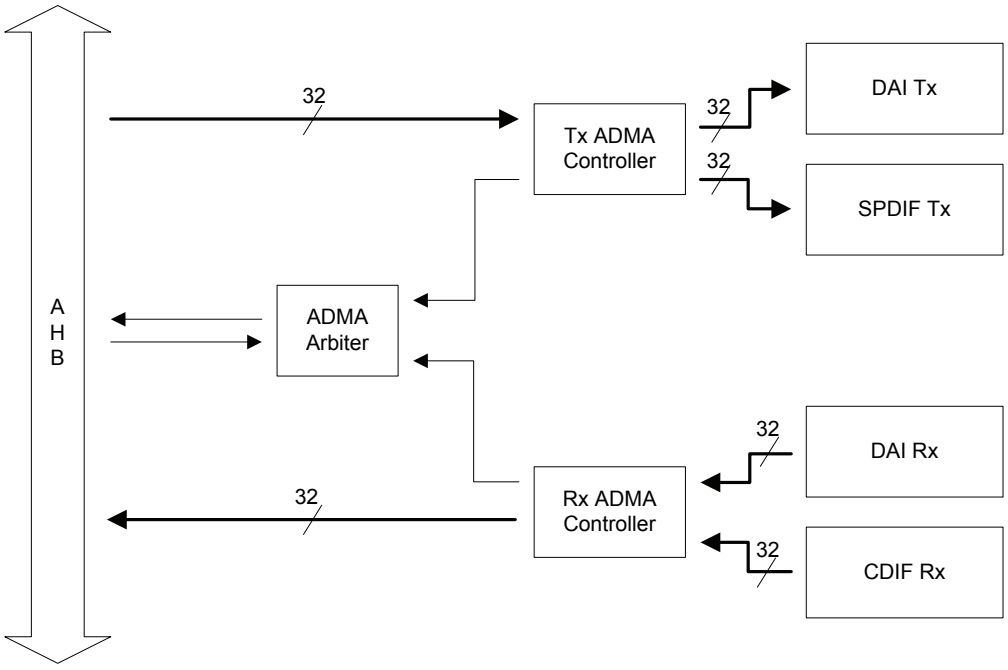


Figure 11.1 AUDIO DMA Top Block Diagram

11.1.2 DAI & CDIF

The block diagram of DAI is shown in below figure.

The NVS2310 provides digital audio interface that complies with IIS (Inter-IC Sound). The DAI has five input/output pins for IIS interface; MCLK, BCLK, LRCK, DAI, DAO. All DAI input/output pins are multiplexed with GPIO pins; GPIO_E[4:0].

The MCLK is the system clock pin that is used for CODEC system clock. In master mode, the MCLK can be generated from clock generator in which that is known as a DCLK, or fed from the outside of chip in slave mode. The DAI can process 256fs, 384fs and 512fs as a system clock. 256fs means that the system clock has 256 times of sampling frequency (fs).

The BCLK is the serial bit clock for IIS data exchange. The DAI can generate 64fs, 48fs and 32fs by dividing a system clock. The polarity of BCLK can be programmed. That is, the serial bit can be stable either rising edge of BCLK or falling edge of BCLK.

The LRCK is the frame clock for the stereo audio channel Left and Right. The frequency of LRCK is known as the “fs” – sampling frequency. Generally, for audio application – such as MP3 Player, CD player, the fs can be set to 8kHz, 16kHz, 11.05kHz, 24kHz, 32kHz, 44.1kHz and 48kHz. For supporting the wide range of sampling frequency in audio application, the DCO function is very useful to generate a system clock. Refer the chapter of clock generator for detail information.

All three clocks (MCLK, BCLK, LRCK) are selectable as master or slave.

The DAI, DAO are the serial data input output pins respectively.

The DAI has two 8-word input/output buffers. It has a banked buffer structure so that one side of buffer is receiving/transmitting data while the other side of that can be read/written through the DADI_XX/DADO_XX registers. The

maximum data word size is 24 bit. Data is justified to MSB of 32bits and zeros are padded to LSB.

There are 2 types of interrupt from IIS; transmit done interrupt, receive done interrupt. The transmit-done interrupt is generated when the 8 words are transferred successfully in the output buffer. At this interrupt, user should fill another 8 more words into the other part of the output buffer in the interrupt service routine (ISR). In this ISR routine, 8 consecutive stores of word data to the DADO registers are needed. The receive-done interrupt is generated when the 8 words are received successfully in the input buffer. At this interrupt, user should read 8 received words from the input buffer using 8 consecutive load instructions from the DADI registers.

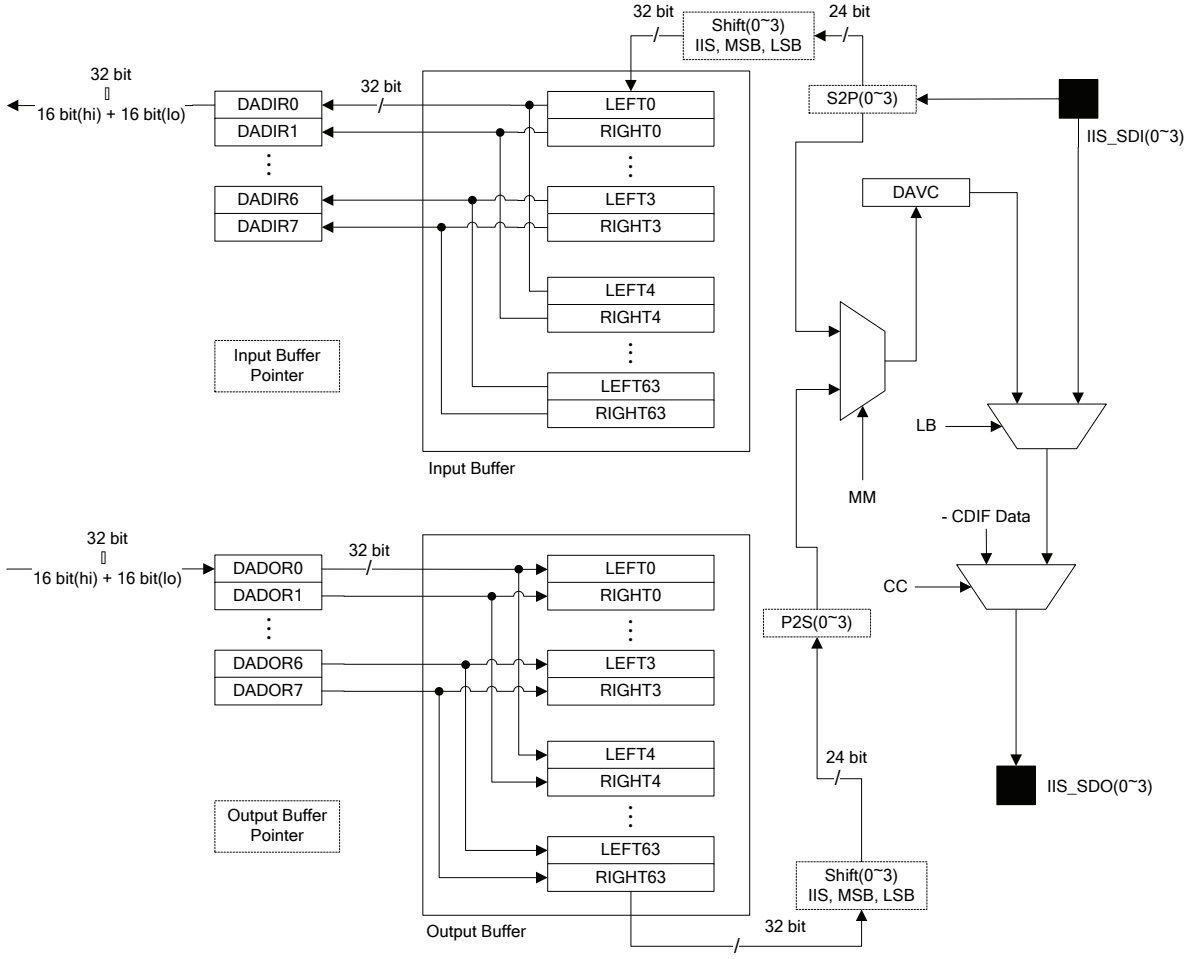


Figure 11.2 DAI Block Diagram

The block diagram of CDIF is illustrated in below figure.

The NVS2310 provides CD-ROM interface for feasible implementation of CD-ROM application such as CD-MP3 player. The CDIF supports the industry standard IIS format and the LSB justified format that is used as the most popular format for CD-ROM interface by Sony and Samsung.

The CDIF has three pins for interface; CBCLK, CLRCK, CDAI. These are multiplexed with GPIO_B[13], GPIO_B[14] and GPIO_B[12]. The CBCLK is the bit clock input pins of which frequency can be programmed by CICR for selection of 48fs and 32fs. The CLRCK is the frame clock input pin that indicates the channel of CD stereo digital audio data. The CDAI is the input data pin.

The CDIF has nine registers; CDDI_0 to CDDI_3 and CICR. The CDDI_0 to the CDDI_3 are the banked read only registers for access of data input buffer. The data input buffer is composed of sixteen 32 bit wide registers of which upper 16 bit is left channel data and lower is right channel data.

The CDIF receive the serial data from CDAI pin and store the data into the buffer through the serial to parallel register. Whenever the half of buffer is filled, the receive interrupt is generated. Only the half of input buffer can be accessible through the CDDI_0 to the CDDI_3.

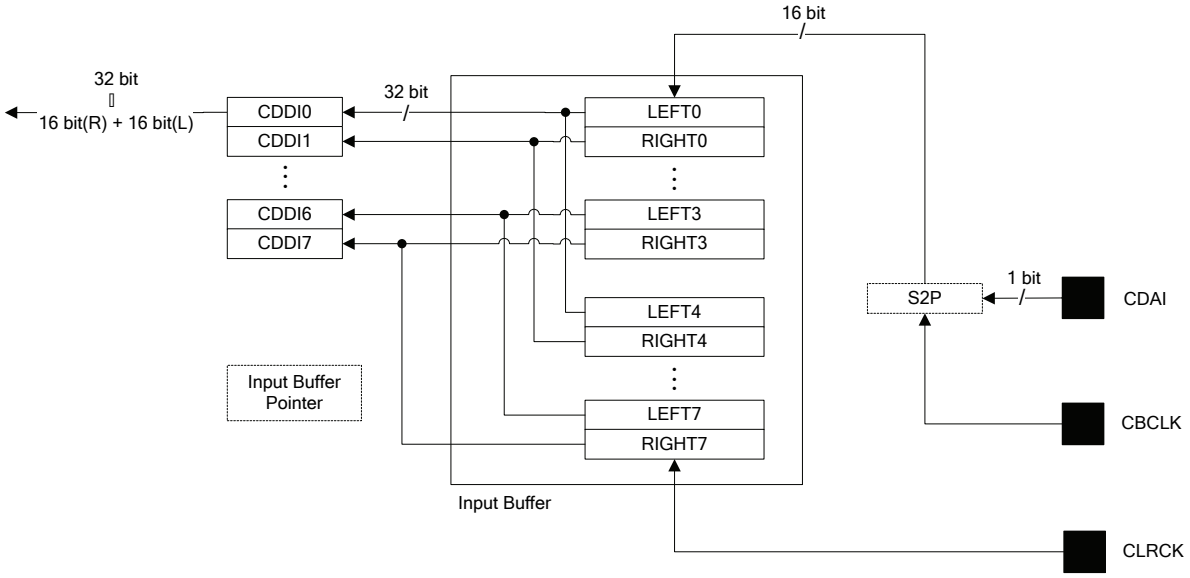


Figure 11.3 CDIF Block Diagram

11.1.3 SPDIF

The SPDIF (or AES/EBU, IEC950 standards) is a point-to-point protocol for serial transmission of digital audio through a single transmission line. The transmission medium can be either electrical or optical (e.g. TosLink). It provides two channels for audio data, a method for communicating control information, and some error detection capabilities. The control information is transmitted as one bit per sample and accumulates in a block structure. The data is by-phase encoded, which enables the receiver to extract a clock from the data. Coding violations, defined as preambles, are used to identify sample and block boundaries.

The SPDIF interfaces are found on most CD/DVD players, audio equipment and computer sound cards.

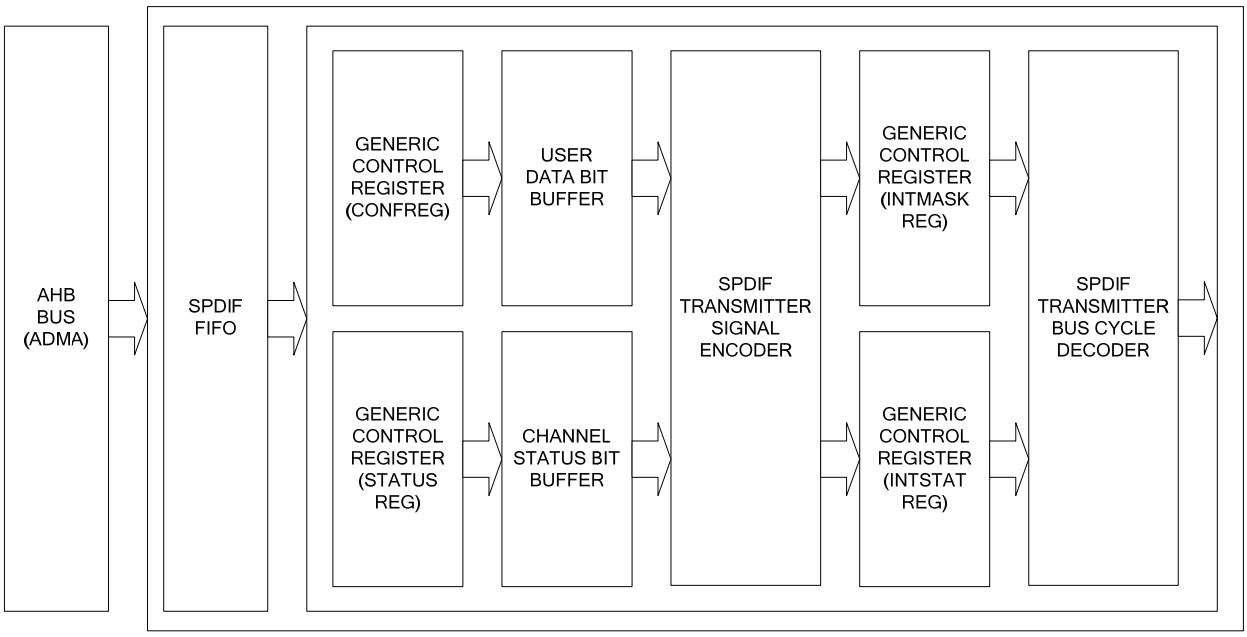


Figure 11.4 SPDIF Tx Block Diagram

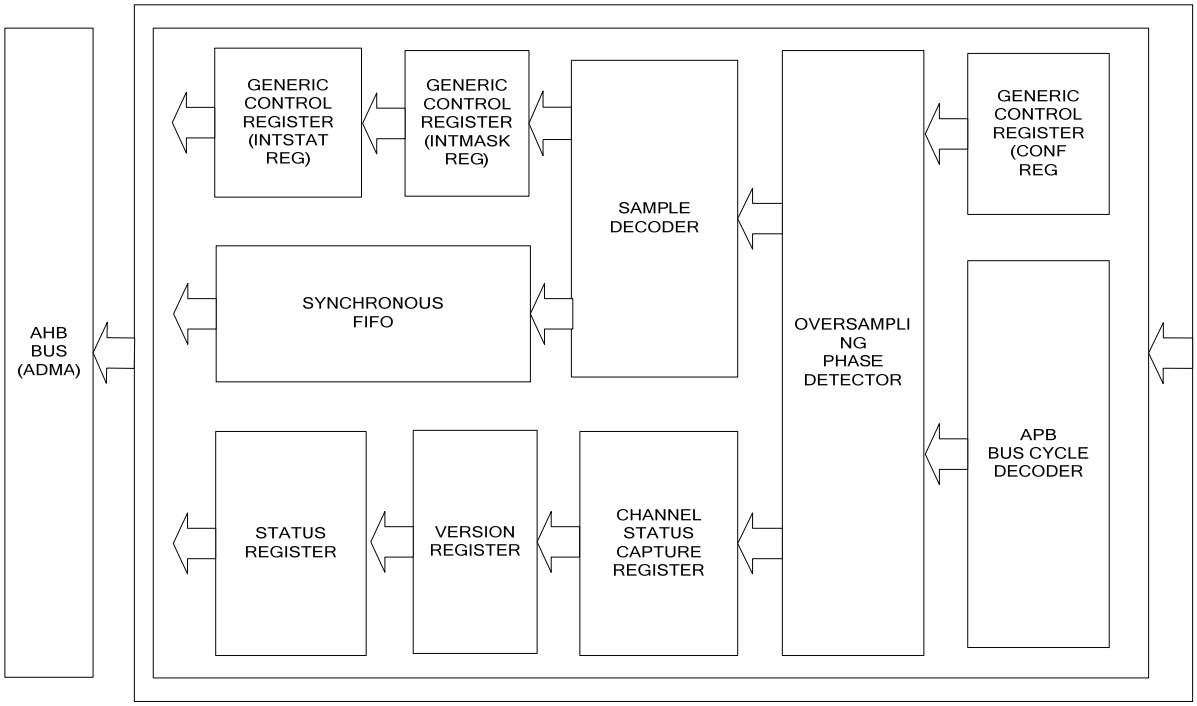


Figure 11.5 SPDIF Rx Block Diagram

11.2 Register Description

Table 11.1 AUDIO DMA Register Map (Base Address = 0xB0105000)

Name	Address	Type	Reset	Description
RxDaDar	0x00	R/W	0x00000000	DAI Rx (Right) Data Destination Address
RxDaParam	0x04	R/W	0x00000000	DAI Rx Parameters
RxDaTCnt	0x08	R/W	0x00000000	DAI Rx Transmission Counter Register
RxDaCdar	0x0C	R	0x00000000	DAI Rx (Right) Data Current Destination Address
RxCdDar	0x10	R/W	0x00000000	CDIF Rx (Right) Data Destination Address
RxCdParam	0x14	R/W	0x00000000	CDIF Rx Parameters
RxCdTCnt	0x18	R/W	0x00000000	CDIF Rx Transmission Counter Register
RxCdCdar	0x1C	R	0x00000000	CDIF Rx (Right) Data Current Destination Address
RxDaDarL	0x28	R/W	0x00000000	DAI Rx Left Data Destination Address
RxDaCdarL	0x2C	R	0x00000000	DAI Rx Left Data Current Destination Address
RxCdDarL	0x30	R/W	0x00000000	CDIF Rx Left Data Destination Address
RxCdCdarL	0x34	R	0x00000000	CDIF Rx Left Data Current Destination Address
TransCtrl	0x38	R/W	0x0000AA00	DMA Transfer Control Register
RptCtrl	0x3C	R/W	0x00000000	DMA Repeat Control Register
TxDaSar	0x40	R/W	0x00000000	DAI Tx (Right) Data Source Address
TxDaParam	0x44	R/W	0x00000000	DAI Tx Parameters
TxDaTCnt	0x48	R/W	0x00000000	DAI Tx Transmission Counter Register
TxDaCsar	0x4C	R	0x00000000	DAI Tx (Right) Data Current Source Address
TxSpSar	0x50	R/W	0x00000000	SPDIF Tx (Right) Data Source Address
TxSpParam	0x54	R/W	0x00000000	SPDIF Tx Parameters
TxSpTCnt	0x58	R/W	0x00000000	SPDIF Tx Transmission Counter Register
TxSpCsar	0x5C	R	0x00000000	SPDIF Tx (Right) Data Current Source Address
TxDaSarL	0x68	R/W	0x00000000	DAI Tx Left Data Source Address
TxDaCsarL	0x6C	R	0x00000000	DAI Tx Left Data Current Source Address
TxSpSarL	0x70	R/W	0x00000000	SPDIF Tx Left Data Source Address
TxSpCsarL	0x74	R	0x00000000	SPDIF Tx Left Data Current Source address
ChCtrl	0x78	R/W	0x00008000	DMA Channel Control Register
IntStatus	0x7C	R/W	0x00000000	DMA Interrupt Status Register

Table 11.2 DAI Register Map (Base Address = 0xB0106000)

Name	Address	Type	Reset	Description
DADIR0	0x00	R	-	Digital Audio Input Register 0
DADIR1	0x04	R	-	Digital Audio Input Register 1
DADIR2	0x08	R	-	Digital Audio Input Register 2
DADIR3	0x0C	R	-	Digital Audio Input Register 3
DADIR4	0x10	R	-	Digital Audio Input Register 4
DADIR5	0x14	R	-	Digital Audio Input Register 5
DADIR6	0x18	R	-	Digital Audio Input Register 6
DADIR7	0x1C	R	-	Digital Audio Input Register 7
DADOR0	0x20	R/W	-	Digital Audio Output Register 0
DADOR1	0x24	R/W	-	Digital Audio Output Register 1
DADOR2	0x28	R/W	-	Digital Audio Output Register 2
DADOR3	0x2C	R/W	-	Digital Audio Output Register 3
DADOR4	0x30	R/W	-	Digital Audio Output Register 4
DADOR5	0x34	R/W	-	Digital Audio Output Register 5
DADOR6	0x38	R/W	-	Digital Audio Output Register 6
DADOR7	0x3C	R/W	-	Digital Audio Output Register 7
DAMR	0x40	R/W	0x00000000	Digital Audio Mode Register
DAVC	0x44	R/W	0x0000	Digital Audio Volume Control Register

Table 11.3 CDIF Register Map (Base Address = 0xB0106000)

Name	Address	Type	Reset	Description
CDDI_0	0x80	R		CD Digital Audio Input Register 0
CDDI_1	0x84	R		CD Digital Audio Input Register 1
CDDI_2	0x88	R		CD Digital Audio Input Register 2
CDDI_3	0x8C	R		CD Digital Audio Input Register 3
CDDI_4	0x90	R		CD Digital Audio Input Register 4
CDDI_5	0x94	R		CD Digital Audio Input Register 5
CDDI_6	0x98	R		CD Digital Audio Input Register 6
CDDI_7	0x9C	R		CD Digital Audio Input Register 7
CICR	0xA0	R/W	0x0000	CD Interface Control Register

Table 11.4 SPDIF Tx Register Map (Base Address = 0xB010D000)

Name	Address	Type	Reset	Description
TxVersion	0x00	R	0x00003111	Version Register
TxConfig	0x04	R/W	0x00000000	Configuration Register
TxChStat	0x08	R/W	0x00000000	Channel Status Control Register
TxIntMask	0x0C	R/W	0x00000000	Interrupt Mask Register
TxIntStat	0x10	R/W	0x00000000	Interrupt Status Register
UserData	0x80~0xDC	W	-	User Data Buffer
ChStatus	0x100~0x15C	W	-	Channel Status Buffer
TxBuffer	0x200~0x23C	W	-	Transmit Data Buffer
DMACFG	0x400	R/W	0x00000007	Additional Configuration for DMA
CSBUDB	0x680~0x6DC	W	-	Merged Window for CSB/UDB

11.2.1 Audio DMA Register

DAI0(CDIF)Rx / DAI0(SPDIF)Tx (Right) Data Destination Address Register

0xB0105000(0x10)/0x40(0x50)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RxDaDar															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RxDaDar															

Field	Name	RW	Reset	Description
31-0	RxDaDar	R/W	0	This register contains the start address of target memory block for DAI0 Rx DMA transfer. The transfer begins reading data from this address. When LRMode is asserted, this is used for Right Data of DAI0 Rx.

DAI(CDIF) Rx / DAI(SPDIF) Tx Parameters Register

0xB0105004(0x14)/0x44(0x54)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SMASK															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMASK								SINC							

Field	Name	RW	Reset	Description
31-8	SMASK	R/W	0	0 : non-masked N : Masked so that source address bit doesn't be changed during DMA transfer Each bit field controls the dedicated bit of source address field. That is, if SMASK[23] is set to 1, the 28th bit of source address is masked, and if SMASK[22] is set to 1, the 27th bit of source address is masked, and so on. If a bit is masked, a corresponding bit of address bus is not changed during DMA transfer. This function can be used to generate circular buffer address.
7-0	SINC	R/W	0	N : Source address is added by amount of sinc at every write cycles. sinc is represented as 2's complement, so if SINC[7] is 1, the source address is decremented. The addresses of DMA transfer have 32bit wide, but the upper 4bit of them are not affected during DMA transfer. If the source or destination address reaches its maximum address space like 0x7FFFFFFF or 0x2FFFFFFF, the next transfer is starting from 0x70000000 or 0x20000000 not from 0x80000000 or 0x30000000.

DAI(CDIF) Rx / DAI(SPDIF) Tx Transmission Counter Register

0x B0105008(0x18)/0x48(0x58)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C_TCOUNT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST_TCOUNT															

Field	Name	RW	Reset	Description
31-16	C_TCOUNT	R	0	N : Represent cn number of Hop transfer remains
15-0	ST_TCOUNT	R/W	0	N : DMA transfers data by amount of sn Hop transfers

At the beginning of transfer, the C_TCNT is updated by ST_TCNT register. At the end of every hop transfer, this is decremented by 1 until it reaches to zero. When this reaches to zero, the DMA finishes its transfer and may or may not generate its interrupt according to IEN flag of CHCTRL register.

DAI0(CDIF)Rx / DAI0(SPDIF)Tx (Right) Data Current Destination Address Register

0x B010500C(0x1C)/0x4C(0x5C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RxDaCdar															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RxDaCdar															

Field	Name	RW	Reset	Description
31-0	RxDaCdar	R	0	This register contains the current destination address of DAI0 Rx DMA transfer. It represents that the current transfer read data from this address. This is read only register. When LRMode is asserted, this is used for Right Data of DAI0 Rx.

DAI0(CDIF)Rx / DAI0(SPDIF)Tx Left Data Destination Address Register

0x B0105028(0x30)/0x68(0x70)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RxDaDarL															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RxDaDarL															

Field	Name	RW	Reset	Description
31-0	RxDaDarL	R/W	0	This register contains the start address of target memory block for DAI0 Rx DMA transfer. The transfer begins reading data from this address. When LR Mode is asserted, this is used for Left Data of DAI0 Rx, otherwise this is not available.

DAI(CDIF) Rx / DAI(SPDIF) Left Data Current Destination Address Register 0x B010502C(0x34)/0x6C(0x74)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RxDaCdarL															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RxDaCdarL															

Field	Name	RW	Reset	Description
31-0	RxDaCdarL	R	0	This register contains the current destination address of DAI0 Rx DMA transfer. It represents that the current transfer read data from this address. This is read only register. When LRMode is asserted, this is used for Left Data of DAI0 Rx, otherwise this is not available.

Transfer Control Register

0x B0105038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*		RCN	TCN	CRLCK	DRLCK	STLCK	DTLCK	CRTRG	DRTRG	STTRG	DTTRG	CRRPT	DRRPT	STRPT	DTRPT
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRBSIZE		DRBSIZE		STBSIZE		DTBSIZE		CRWSIZE		DRWSIZE		STWSIZE		DTWSIZE	

Field	Name	RW	Reset	Description
31-30	R*	R	0	Reserved
29	RCN	R/W	0	Issue Continuous Transfer of Rx DMA 0 : Rx DMA transfer begins from source/destination address 1 : Rx DMA transfer begins from current source/destination address It must be used after the former transfer has been executed, so that current source/destination address registers contain a meaningful value.
28	TCN	R/W	0	Issue Continuous Transfer of Tx DMA 0 : Tx DMA transfer begins from source/destination address 1 : Tx DMA transfer begins from current source/destination address

				It must be used after the former transfer has been executed, so that current source/destination address registers contain a meaningful value.
27	CRLCK	R/W	0	Issue Locked Transfer of CDIF Rx DMA 0 : CDIF Rx DMA transfer is done without lock transfer 1 : CDIF Rx DMA transfer is done with lock transfer Lock field controls the LOCK signal (refer to AHB specification). When the LOCK is set to 1, the DMA transfer doesn't be bothered by other AHB masters like LCD controller, ARM etc.
26	DRLCK	R/W	0	Issue Locked Transfer of DAI Rx DMA 0 : DAI Rx DMA transfer is done without lock transfer 1 : DAI Rx DMA transfer is done with lock transfer
25	STLCK	R/W	0	Issue Locked Transfer of SPDIF Tx DMA 0 : SPDIF Tx DMA transfer is done without lock transfer 1 : SPDIF Tx DMA transfer is done with lock transfer
24	DTLCK	R/W	0	Issue Locked Transfer of DAI Tx DMA 0 : DAI Tx DMA transfer is done without lock transfer 1 : DAI Tx DMA transfer is done with lock transfer
23	CRTRG	R/W	0	Trigger Type of Rx DMA 0 : SINGLE edge-triggered detection of CDIF Rx DMA 1 : SINGLE level-sensitive detection of CDIF Rx DMA In SINGLE Type, After one Hop data transferring DMA checks External DMA Request (DREQ) and then if its bit is active , DMA transfers next hop data . DREQ is detected level-sensitive or edge-triggered by SINGLE transfer TYPE.
22	DRTRG	R/W	0	Trigger Type of Rx DMA 0 : SINGLE edge-triggered detection of DAI Rx DMA 1 : SINGLE level-sensitive detection of DAI Rx DMA
21	STTRG	R/W	0	Trigger Type of Tx DMA 0 : SINGLE edge-triggered detection of SPDIF Tx DMA 1 : SINGLE level-sensitive detection of SPDIF Tx DMA
20	DTTRG	R/W	0	Trigger Type of Tx DMA 0 : SINGLE edge-triggered detection of DAI Tx DMA 1 : SINGLE level-sensitive detection of DAI Tx DMA
19	CRRPT	R/W	0	Repeat Mode Control of CDIF Rx DMA 0 : After all of hop transfer has executed, CDIF Rx DMA channel is disabled. 1 : CDIF Rx DMA channel remains enabled. When another DMA request has occurred, CDIF Rx DMA channel start transfer data again with the same manner (type, address, increment, mask) as the latest transfer of that channel.
18	DRRPT	R/W	0	Repeat Mode Control of DAI Rx DMA 0 : After all of hop transfer has executed, DAI Rx DMA channel is disabled. 1 : DAI Rx DMA channel remains enabled. When another DMA request has occurred, DAI Rx DMA channel start transfer data again with the same manner (type, address, increment, mask) as the latest transfer of that channel.
17	STRPT	R/W	0	Repeat Mode Control of SPDIF Tx DMA 0 : After all of hop transfer has executed, SPDIF Tx DMA channel is disabled.

				1 : SPDIF Tx DMA channel remains enabled. When another DMA request has occurred, SPDIF Tx DMA channel start transfer data again with the same manner (type, address, increment, mask) as the latest transfer of that channel.
16	DTRPT	R/W	0	Repeat Mode Control of DAI Tx DMA 0 : After all of hop transfer has executed, DAI Tx DMA channel is disabled. 1 : DAI Tx DMA channel remains enabled. When another DMA request has occurred, DAI Tx DMA channel start transfer data again with the same manner (type, address, increment, mask) as the latest transfer of that channel. The 1 Hop of transfer means 1 burst of read followed by 1 burst of write. 1 burst means 1, 2 or 4 consecutive read or write cycles defined by BSIZE field of CHCTRL register.
15-14	CRBSIZE	R/W	2	Burst Size of CDIF Rx DMA 0 : 1 Burst transfer of CDIF Rx DMA consists of 1 read or write cycle. 1 : 1 Burst transfer of CDIF Rx DMA consists of 2 read or write cycles 2 : 1 Burst transfer of CDIF Rx DMA consists of 4 read or write cycles 3 : 1 Burst transfer of CDIF Rx DMA consists of 8 read or write cycles When LRMode is enabled, 8 Read/Write is not available.
13-12	DRBSIZE	R/W	2	Burst Size of DAI Rx DMA 0 : 1 Burst transfer of DAI Rx DMA consists of 1 read or write cycle. 1 : 1 Burst transfer of DAI Rx DMA consists of 2 read or write cycles 2 : 1 Burst transfer of DAI Rx DMA consists of 4 read or write cycles 3 : 1 Burst transfer of DAI Rx DMA consists of 8 read or write cycles When LRMode is enabled, 8 Read/Write is not available.
11-10	STBSIZE	R/W	2	Burst Size of SPDIF Tx DMA 0 : 1 Burst transfer of SPDIF Tx DMA consists of 1 read or write cycle. 1 : 1 Burst transfer of SPDIF Tx DMA consists of 2 read or write cycles 2 : 1 Burst transfer of SPDIF Tx DMA consists of 4 read or write cycles 3 : 1 Burst transfer of SPDIF Tx DMA consists of 8 read or write cycles When LRMode is enabled, 8 Read/Write is not available.
9-8	DTBSIZE	R/W	2	Burst Size of DAI Tx DMA 0 : 1 Burst transfer of DAI Tx DMA consists of 1 read or write cycle. 1 : 1 Burst transfer of DAI Tx DMA consists of 2 read or write cycles 2 : 1 Burst transfer of DAI Tx DMA consists of 4 read or write cycles 3 : 1 Burst transfer of DAI Tx DMA consists of 8 read or write cycles When LRMode is enabled, 8 Read/Write is not available.
7-6	CRWSIZE	R/W	0	Word Size of CDIF Rx DMA 0 : Each cycle read or write 8bit data of CDIF Rx DMA 1 : Each cycle read or write 16bit data of CDIF Rx DMA 2, 3 : Each cycle read or write 32bit data of CDIF

				Rx DMA
5-4	DRWSIZE	R/W	0	Word Size of DAI Rx DMA 0 : Each cycle read or write 8bit data of DAI Rx DMA 1 : Each cycle read or write 16bit data of DAI Rx DMA 2, 3 : Each cycle read or write 32bit data of DAI Rx DMA
3-2	STWSIZE	R/W	0	Word Size of SPDIF Tx DMA 0 : Each cycle read or write 8bit data of SPDIF Tx DMA 1 : Each cycle read or write 16bit data of SPDIF Tx DMA 2, 3 : Each cycle read or write 32bit data of SPDIF Tx DMA
1-0	DTWSIZE	R/W	0	Word Size of DAI Tx DMA 0 : Each cycle read or write 8bit data of DAI Tx DMA 1 : Each cycle read or write 16bit data of DAI Tx DMA 2, 3 : Each cycle read or write 32bit data of DAI Tx DMA

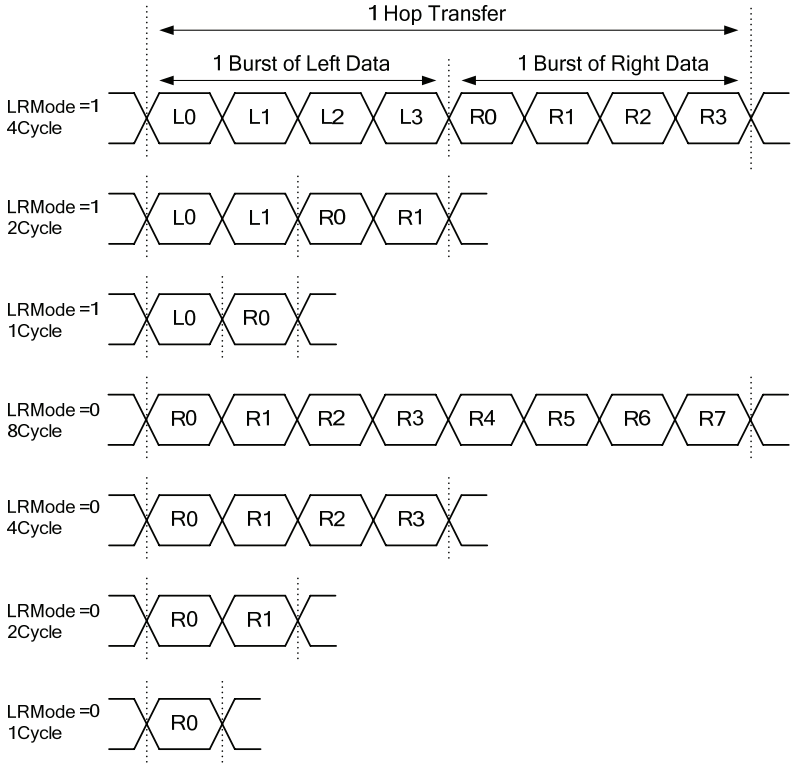


Figure 11.6 Relation between Hop and Burst Transfers

Repeat Control Register

0x B010503C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DRI	Reserved							RPTCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RPTCNT															

Field	Name	RW	Reset	Description
31	DRI	R/W	0	Disable Repeat Interrupt 0 : DMA Interrupt is occurred when the end of each Repeated DMA operation 1 : DMA Interrupt occur is occurred when the last DMA Repeated DMA operation This bit is meaningful when Repeat Mode is enabled.
30-28	R*	R	0	Reserved
27-24	DBTH	RW	0	DAI Buffer Threshold n : DMA transfer buffer threshold determines DMA transfer trigger condition This field sets the threshold condition of internal 64-depth buffer for DAI DMA. If audio data increases more than (DBTH+1)*2^DTBSIZE, the internal DMA starts the transfer. When MPE bit is enabled, DBTH should be fixed to 7 and also DTBSIZE should be 2.
23- 0	RPTCNT	R/W	0	Repeat Count 0 : DMA transfer will be repeated endlessly. n : DMA transfer will be repeated (n + 1) * TCOUNT times. This bit is meaningful when Repeat Mode is enabled. When this bit is cleared in repeat mode, DMA transfer will be repeated endlessly. To exit endless repeat mode, clear EN bit of ChCtrl or disable Repeat Mode. It's possible to circular transfer using repeat count.

Channel Control Register

0xB0105078

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CREN	DREN	STEN	DTEN	R								CRLR	DRLR	STLR	DTLR
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CRDW	DREW	STDW	DTDW	CRSEN	DRSEN	STSEN	DTSEN	CRWB	DRWB	STWB	DTWB	CREN	DREN	STEN	DTEN

Field	Name	RW	Reset	Description
31	CREN	R/W	0	DMA Channel Enable of CDIF Rx 0 : CDIF Rx DMA channel is terminated and disabled. It does not affect the TCOUNT register, so if the current hop counter is not zero when channel is disabled, it is possible that the transfer illegally starts right after channel is re-enabled. Make sure that TCOUNT is zero not to continue transfer after channel is re-enabled. 1 : CDIF Rx DMA channel is enabled.
30	DREN	R/W	0	DMA Channel Enable of DAI Rx 0 : DAI Rx DMA channel is terminated and disabled. 1 : DAI Rx DMA channel is enabled.
29	STEN	R/W	0	DMA Channel Enable of SPDIF Tx 0 : SPDIF Tx DMA channel is terminated and disabled. 1 : SPDIF Tx DMA channel is enabled.
28	DTEN	R/W	0	DMA Channel Enable of DAI Tx 0 : DAI Tx DMA channel is terminated and disabled. 1 : DAI Tx DMA channel is enabled.
27	DSSC	R/W	0	DAI and SPDIF are have the same clock 0 : Disable 1 : Enable (DAI Block CLK → SPDIF Block CLK)

26	CSS	R/W	0	CDIF or SPDIF Select 0: CDIF Rx Mode 1: SPDIF Rx Mode
25	DRMCM	R/W	0	DMA Multi Channel Mode of DAI Rx 0 : Single Channel Mode. 1 : Multi Channel Mode.
24	DTMCM	R/W	0	DMA Multi Channel Mode of DAI Tx 0 : Single Channel Mode. 1 : Multi Channel Mode.
23-22	DMRSEL	R/W	0	DMA Multi Channel Select of DAI Rx 00 : DAI0 & DAI1 Channel Select (3.1Ch) 01 : DAI0 & DAI1 & DAI2 Channel Select (5.1Ch) 10 : DAI0 & DAI1 & DAI2 & DAI3 channel select (7.1Ch)
21-20	DMTSEL	R/W	0	DMA Multi Channel Select of DAI Tx 00 : DAI0 & DAI1 Channel Select (3.1Ch) 01 : DAI0 & DAI1 & DAI2 Channel Select (5.1Ch) 10 : DAI0 & DAI1 & DAI2 & DAI3 channel select (7.1Ch)
19	CRLR	R/W	0	Left/Right Data Mode of CDIF Rx 0 : Disable LRMode of CDIF Rx DMA. 1 : Enable LRMode of CDIF Rx DMA. When LRMode is enabled, audio data will be written to memory separately according to whether it is left or right one.
18	DRLR	R/W	0	Left/Right Data Mode of DAI Rx 0 : Disable LRMode of DAI Rx DMA. 1 : Enable LRMode of DAI Rx DMA.
17	STLR	R/W	0	Left/Right Data Mode of SPDIF Tx 0 : Disable LRMode of SPDIF Tx DMA. 1 : Enable LRMode of SPDIF Tx DMA.
16	DTLR	R/W	0	Left/Right Data Mode of DAI Tx 0 : Disable LRMode of DAI Tx DMA. 1 : Enable LRMode of DAI Tx DMA.
15	CRDW	R/W	0	Width of Audio Data of CDIF Rx 0 : Assume width of CDIF Rx Data is 24bits. 1 : Assume width of CDIF Rx Data is 16bits. AUDIO DMA manipulates audio data as 16bit-length one. If it gets 24bit-length audio data from audio devices, it will cut out lsb-16bits and put those into DMA FIFO. For more details, refer to attached pictures.
14	DRDW	R/W	0	Width of Audio Data of DAI Rx 0 : Assume width of DAI Rx Data is 24bits. 1 : Assume width of DAI Rx Data is 16bits.
13	STDW	R/W	0	Width of Audio Data of SPDIF TX 0 : Assume width of SPDIF Tx Data is 24bits. 1 : Assume width of SPDIF Tx Data is 16bits.
12	DTDW	R/W	0	Width of Audio Data of DAI Tx 0 : Assume width of DAI Tx Data is 24bits. 1 : Assume width of DAI Tx Data is 16bits.
11	CRSEN	R/W	0	Swapping Half-word/Byte align of CDIF Rx 0 : Disable swapping of CDIF Rx DMA 1 : Swaps half-word or byte align according to WB bits of CDIF Rx DMA
10	DRSEN	R/W	0	Swapping Half-word/Byte align of DAI Rx 0 : Disable swapping of DAI Rx DMA 1 : Swaps half-word or byte align according to WB bits of DAI Rx DMA
9	STSEN	R/W	0	Swapping Half-word/Byte align of SPDIF Tx 0 : Disable swapping of SPDIF Tx DMA 1 : Swaps half-word or byte align according to WB bits of SPDIF Tx DMA
8	DTSEN	R/W	0	Swapping Half-word/Byte align of DAI Tx 0 : Disable swapping of DAI Tx DMA 1 : Swaps half-word or byte align according to WB bits of DAI Tx DMA

7	CRWB	R/W	0	Choose Half-word/Byte of CDIF Rx 0 : Byte swapping of CDIF Rx DMA. This is available only when SWP is enabled. 1 : Half-word swapping of CDIF Rx DMA. This is available only when SWP is enabled.
6	DRWB	R/W	0	Choose Half-word/Byte of DAI Rx 0 : Byte swapping of DAI Rx DMA. This is available only when SWP is enabled. 1 : Half-word swapping of DAI Rx DMA. This is available only when SWP is enabled.
5	STWB	R/W	0	Choose Half-word/Byte of SPDF Tx 0 : Byte swapping of SPDIF Tx DMA. This is available only when SWP is enabled. 1 : Half-word swapping of SPDIF Tx DMA. This is available only when SWP is enabled.
4	DTWB	R/W	0	Choose Half-word/Byte of DAI Tx 0 : Byte swapping of DAI Tx DMA. This is available only when SWP is enabled. 1 : Half-word swapping of DAI Tx DMA. This is available only when SWP is enabled.
3	CRIEN	R/W	0	Interrupt Enable of CDIF Rx 1 : At the same time the CRI goes to 1, CDIF Rx DMA interrupt request is generated. To generate IRQ or FIQ interrupt, the DMA flag of IEN register in the interrupt controller must be set to 1 ahead.
2	DRIEN	R/W	0	Interrupt Enable of DAI Rx 1 : At the same time the CRI goes to 1, DAI Rx DMA interrupt request is generated.
1	STIEN	R/W	0	Interrupt Enable of SPDIF Tx 1 : At the same time the CRI goes to 1, SPDIF Tx DMA interrupt request is generated.
0	DTIEN	R/W	0	Interrupt Enable of DAI Tx 1 : At the same time the CRI goes to 1, DAI Tx DMA interrupt request is generated.

DMA Interrupt Status Register **0x B010507C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CRI	DRI	STI	DTI	CRMI	DRMI	STMI	DTMI

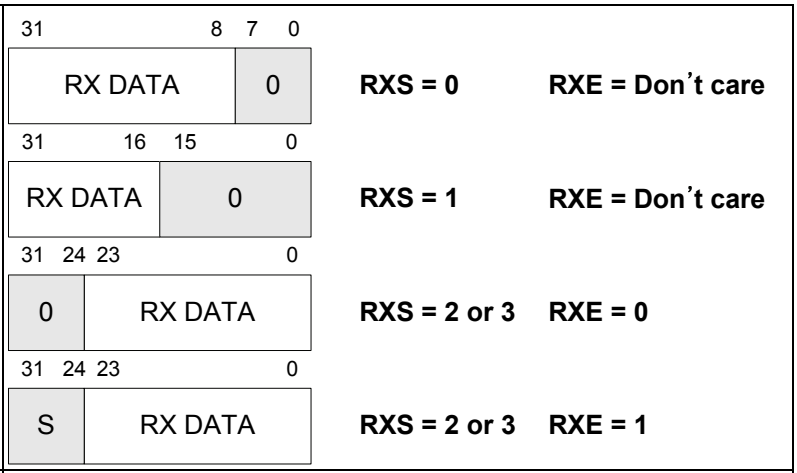
Field	Name	RW	Reset	Description
31-8	R*	R	0	Reserved
7	CRI	R/W	0	DMA Interrupt Status of CDIF(SPDIF) Rx 0 : No interrupt occurred while CDIF Rx DMA transfer. 1 : Interrupt occurred while CDIF Rx DMA transfer. Without regard to Interrupt enable bit(CRIEN) of ChCtrl, this bit indicates the CDIF Rx DMA interrupt status. This bit is automatically cleared when FLAG bit of ChCtrl is cleared. This bit is read only.
6	DRI	R/W	0	DMA Interrupt Status of DAI Rx 0 : No interrupt occurred while DAI Rx DMA transfer. 1 : Interrupt occurred while DAI Rx DMA transfer. Without regard to Interrupt enable bit(DRIEN) of ChCtrl, this bit indicates the DAI Rx DMA interrupt status. This bit is automatically cleared when FLAG bit of ChCtrl is cleared. This bit is read only.
5	STI	R/W	0	DMA Interrupt Status of SPDIF Tx 0 : No interrupt occurred while SPDIF Tx DMA transfer. 1 : Interrupt occurred while SPDIF Tx DMA transfer. Without regard to Interrupt enable bit(STIEN) of ChCtrl, this bit indicates the SPDIF Tx DMA interrupt status. This bit is automatically cleared when FLAG bit of ChCtrl is cleared. This bit is read only.
4	DTI	R/W	0	DMA Interrupt Status of DAI Tx 0 : No interrupt occurred while DAI Tx DMA transfer. 1 : Interrupt occurred while DAI Tx DMA transfer. Without regard to Interrupt enable bit(DTIEN) of ChCtrl, this bit indicates the DAI Tx DMA interrupt status. This bit is automatically cleared when FLAG bit of ChCtrl is cleared. This bit is read only.
3	CRMI	R/W	0	DMA Masked Interrupt Status of CDIF(SPDIF) Rx 1 : Interrupt with enable asserted occurred while CDIF Rx DMA transfer. This bit indicates the CDIF Rx DMA interrupt status when CRIEN is asserted.
2	DRMI	R/W	0	DMA Masked Interrupt Status of DAI Rx 1 : Interrupt with enable asserted occurred while DAI Rx DMA transfer. This bit indicates the DAI Rx DMA interrupt status when DRIEN is asserted.
1	STMI	R/W	0	DMA Masked Interrupt Status of SPDIF Tx 1 : Interrupt with enable asserted occurred while SPDIF Tx DMA transfer. This bit indicates the SPDIF Tx DMA interrupt status when STIEN is asserted.
0	DTMI	R/W	0	DMA Masked Interrupt Status of DAI Tx 1 : Interrupt with enable asserted occurred while DAI Tx DMA transfer. This bit indicates the DAI Tx DMA interrupt status when DTIEN is asserted.

11.2.2 DAI Register

Digital Audio Mode Register (DAMR) **0xB0106040**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BPS	LPS	R		AFE	R		NMDR	NMDT	RXE	RXS		TXS		R	SP
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EN	TE	RE	MD	SM	BM	FM	CC	BD		FD		BP	CM	MM	LB

Field	Name	RW	Reset	Description
31	BPS	R/W	0	BCLK Pad Select 0 : BCLK source select BCLK Pad 1 : BCLK direct at master mode
30	LPS	R/W	0	LRCK Pad Select 0 : LRCK source select LRCK Pad 1 : LRCK direct at master mode
29	DBTEN	R/W	0	DAI Buffer Threshodl Enable 0 : Disable 1 : Enable This bit enables DAI buffer threshold (DBTH) control register. If it is set to low, DAI DMA transfer starts right after new data is stored to DAI buffer. When MPE bit is set to high, this field should be set to high not to get wrong transfer in multi-port condition.
28	MPE	R/W	0	Multi-Port Enable 0 : Disable 1 : Enable In multi-port condition, DAI DMA transfer mode is fixed to one mode. That is, DBTEN, DBTH, DTBSIZE should be fixed to 1, 7 and 2 each.
27	AFE	R/W	0	Audio Filter Enable 1 : Audio filters will be enabled. When CD BCLK, CD LRCK, DAI MCLK, DAI BCLK and DAI LRCK are exposed to noises, you may suppress them by enable this bit. Audio Filter clock has to be enabled before enabling this bit and be at least four times faster than CD BCLK or DAI MCLK. 0 : Audio filters will be disabled.
26	DDL	R/W	0	DSP Mode Word Length 0 : Data Length 24bit 1 : Data Length 16bit This selects the number of bit width in Wolfson DSP & TDM mode. In Wolfson TDM mode, MCCR1 must be set additionally.
25	DSP	R/W	0	DSP Mode 0 : IIS 1 : DSP or TDM Mode MD ='0' & DSP='1'→ DSP Mode MD ='0' & DSP='1'& FD= "11" → TDM Mode
24	NMDR	R/W	0	Rx Justfied Mode 0 : Rx Left-justified Mode 1 : Rx Right-justified Mode MD ='1' & DSP='0' & NMDR='0' → Left-justified Rx Mode MD ='1' & DSP='0' & NMDR='1' → Right-justified Rx Mode
23	NMDT	R/W	0	Tx Justified Mode 0 : Tx Left-justified Mode 1 : Tx Right-justified Mode MD ='1' & DSP='0' & NMDT='0' → Left-justified Tx Mode MD ='1' & DSP='0' & NMDR='1' → Right-justified Tx Mode
22	RXE	R/W	0	DAI RX Data Sign Extension 0 : Disable (zero extension) 1 : Enable (sign bit extension)
21-20	RXS	R/W	0	DAI Rx Shift 0 : Bit-pack MSB and 24bit mode. 1 : Bit-pack MSB and 16bit mode. 2 : Bit-pack LSB and 24bit mode. 3 : Bit-pack LSB and 16bit mode.

				
19-18	TXS	R/W	0	DAI Tx Shift 0 : Bit-pack MSB mode. 1 : Bit-pack MSB mode. 2 : Bit-pack LSB and 24bit mode. 3 : Bit-pack LSB and 16bit mode. If DMA transfer bit width is 16, DAI TX Shift register should be set to "Bit-pack LSB and 16bit mode"(TXS=3). When DAI TX Shift register is "Bit-pack LSB and 24bit mode", Most Significant Byte of the second 16bit is filled with 0.
17	LBT	R/W	0	Loop Back Test(Tx to Rx) 0 : Disable 1 : Enable
16	SP	R/W	0	DAI System Clock Polatity 0 : Disable 1 : Enable
15	EN	R/W	0	DAI Master Enable 0 : Disable 1 : Enable
14	TE	R/W	0	DAI Transmitter Enable 0 : Disable 1 : Enable
13	RE	R/W	0	DAI Receiver Enable 0 : Disable 1 : Enable
12	MD	R/W	0	DAI Sync Mode 0 : Set DAI sync as IIS, DSP or TDM sync mode 1 : Set DAI sync as Left/Right-justified mode
11	SM	R/W	0	DAI System Clock Master Select 0 : Set that DAI system clock is come from external pin 1 : Set that DAI system clock is generated by the clock generator block The DAI system clock in clock generator is known as DCLK. Its frequency can be determined by PCK_DAI from the CKC.
10	BM	R/W	0	DAI Bit Clock Master Select 0 : Set that DAI bit clock is from external pin 1 : Set that DAI bit clock is generated by dividing DAI system clock
9	FM	R/W	0	DAI Frame Clock Master Select 0 : Set that DAI frame clock is come from external pin 1 : Set that DAI frame clock is generated by dividing DAI bit clock
8	CC	R/W	0	CDIF Clock Select. BLCK and LRCK would be from CDIF under master mode. This is usually associated with CDIF Monitor Mode 0 : Disable CDIF Clock master mode 1 : Enable CDIF Clock master mode
7-6	BD	R/W	0	DAI Bit Clock Divider Select 0 : Select Div 4 (256fs->64fs) 1 : Select Div 6 (384fs->64fs) 2 : Select Div 8 (512fs->64fs, 384fs->48fs , 256fs->32fs) 3 : Select Div16 (512fs->32fs)
5-4	FD	R/W	0	DAI Frame Clock Divider Select 0 : Select Div 32 (32fs->fs) 1 : Select Div 48 (48fs->fs)

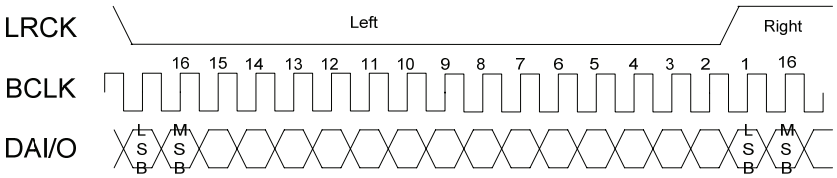
				2 : Select Div 64 (64fs->fs) 3 : Select Div use (xfs->fs) ← Only TDM Mode The combination of BD & FD determines that the ratio between main system clock and the sampling frequency. The multiplication between the division factor of BD and FD must be equal to this ratio. In TDM mode, FD should be set to 3.
3	BP	R/W	0	DAI Bit Clock Polarity 0 : Set that data is captured at positive edge of bit clock 1 : Set that data is captured at negative edge of bit clock
2	CM	R/W	0	CDIF Monitor Mode 0 : Disable 1 : Enable. Data bypass from CDIF
1	MM	R/W	0	DAI Monitor Mode 0 : Disable 1 : Enable. Transmitter should be enabled. (TE = 1)
0	LB	R/W	0	DAI Loop-back Mode 0 : Disable DAI Loop back mode 1 : Enable DAI Loop back mode

Digital Audio Volume Control Register (DAVC) 0x B0105044

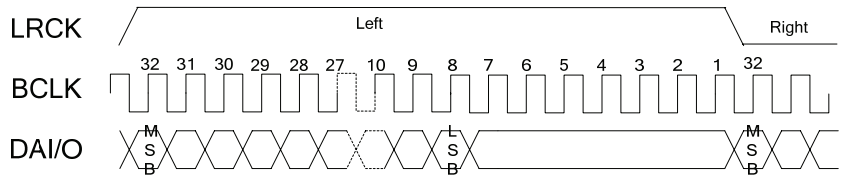
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved											VC<4:0>				

The volume of audio output can be manipulated by this register. It has –6dB unit so the output volume can be set from 0 dB to –90 dB as the following table.

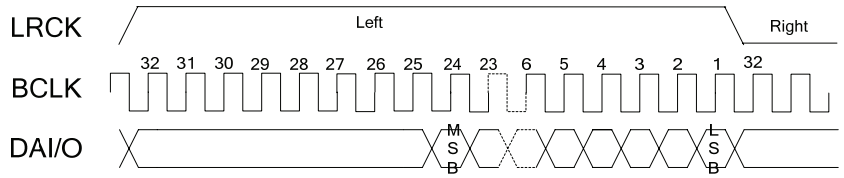
Field	Name	RW	Reset	Description
31-5	R*	-	0	Reserved
4-0	VC	R/W	0	DAI Volume Control 00000 0dB 00001 -6dB 00010 -12dB 00011 -18dB 00100 -24dB 00101 -30dB 00110 -36dB 00111 -42dB 01000 -48dB 01001 -54dB 01010 -60dB 01011 -66dB 01100 -72dB 01101 -78dB 01110 -84dB 01111 -90dB 10000 -96dB



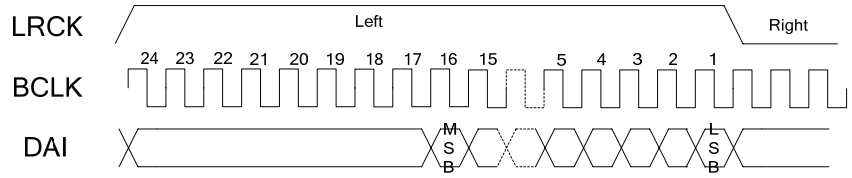
NMDR(T)=0, DSP=0, MD=0 (IIS mode), BP=0, BCLK = 32fs



NMDR(T)=0, DSP=0, MD=1(MSB justified mode), BP=0, BCLK=64fs



NMDR(T)=1, DSP=0, MD=1(LSB justified mode), BP=0, BCLK=64fs



NMDR(T)=1, DSP=0, MD=1(MSB justified mode), BP=1, BCLK=48fs

Figure 11.7 DAI Bus Timing Diagram

11.2.3 CDIF Registers

CD Data Input (CDDI0~CDDI7) **0x B0105080 ~ 0x B010509C**

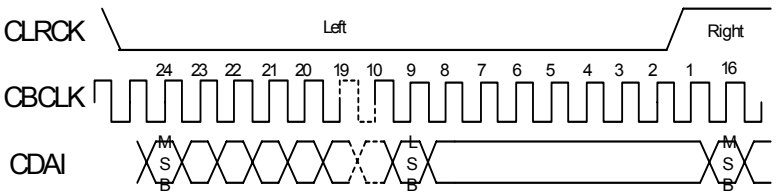
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LDATA															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA															

Field	Name	RW	Reset	Description
31-16	LDATA	R	0	Left Channel Data
15-0	RDATA	R	0	Right Channel Data

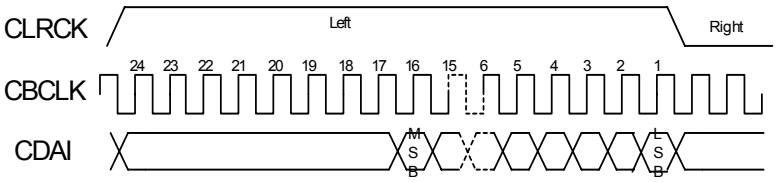
CD Interface Control Register (CICR) **0x B01050A0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								EN	Rserved			BS		MD	BP

Field	Name	RW	Reset	Description
31-8	R*	R	0	Reserved
7	EN	R/W	0	0 : Disable CDIF 1 : Enable CDIF
6-4	R	R	0	Reserved
3-2	BS	R/W	0	CDIF Bit Clock Select 00 : 64fs 01 : 32fs 10 : 48fs
1	MD	R/W	0	Interface Mode Select 0 : Select IIS format 1 : Select LSB justified format
0	BP	R/W	0	CDIF Bit Clock Polarity 0 : Set that data is captured at positive edge of bit clock 1 : Set that data is captured at negative edge of bit clock



MD=0 (IIS mode), BP=0, CBCLK=48fs



MD=1(LSB justified mode), BP=0, CBCLK=48fs

Figure 11.8 CDIF Bus Timing Diagram

11.2.4 SPDIF Registers

SPDIF Transmitter Version Register **0xB010D000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			CSB	UDB	AW						DW	VER			

Field	Name	RW	Reset	Description
31-13	R*	R	0	Reserved
12	CSB	R	1	0 : Channel status buffer is not available 1 : Channel status buffer is available
11	UDB	R	1	0 : User data buffer is not available 1 : User data buffer is available
10-5	AW	R	1	n : Value of Address Width
4	DW	R	1	n : Value of Data Width
3:0	VER	R	1	n : Version Number

SPDIF Transmit Configuration Register **0x B010D004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								MODE				Reserved			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RATIO								UDATEN		CHSTEN		R*	IEN	TXD	TXEN

Field	Name	RW	Reset	Description
31-24	R*	R	0	Reserved
23-20	MODE	R/W	0	n : 16 + n Bits Transmit Sample Format 9 ~ 15 : Reserved
19-16	R*	R	0	Reserved
15-8	RATIO	R/W	0	Clock Divider Ratio n : Clock divider for the transmit frequency. The SPDIF clock is divided by a factor of (RATIO + 1) to generate the serial transmit clock.
7-6	UDATEN	R/W	0	User Data Enable Bits 0 : User data A & B set to 0. 1 : User data A & B generated from UserData bit 7-0 2 : User data A generated from UserData bit 7-0, B generated from UserData bit 15-8 3 : Reserved
5-4	CHSTEN	R/W	0	Channel Status Enable Bits 0 : Channel status A & B generated from TxChStat 1 : Channel status A & B generated from ChStat bit 7-0 2 : Channel status A generated from ChStat bit 7-0, B generated from ChStat bit 15-8 3 : Reserved
3	R*	R	0	Reserved
2	IEN	R/W	0	Interrupt Output Enable Bit n : '1' for enabling the interrupt output.
1	TXD	R/W	0	Data Valid Bit n : '1' for data being valid.
0	TXEN	R/W	0	Transmitter Enable Bit n : '1' for enabling the transmission.

SPDIF Transmit Channel Status Control Register 0xB010D008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					SCS	SUD	SV	FREQ		Reserved		GSTS	PRE	CPY	AU

Field	Name	RW	Reset	Description
31-11	R*	R	0	Reserved
10	SCS	R/W	0	Store Channel Status Bit 0 : No Store 1 : Store
9	SUD	R/W	0	Store User Data Bit 0 : No Store 1 : Store
8	SV	R/W	0	Store Validity Bit 0 : No Store 1 : Store
7-6	FREQ	R/W	0	Sampling Frequency 0 : 44.1 KHz 1 : 48 KHz 2 : 32 KHz 3 : Sample Rate Converter
5-4	R*	R	0	Reserved
3	GSTS	R/W	0	Status Generation 0 : No indication 1 : Original/Commercially Pre-recorded data
2	PRE	R/W	0	Pre-emphasis 0 : None 1 : 50/15us
1	CPY	R/W	0	Copyright 0 : Copy inhibited 1 : Copy permitted
0	AU	R/W	0	Data Format 0 : Audio Format 1 : Data Format

SPDIF Transmit Interrupt Mask Register 0xB010D00C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											HCSB	LCSB	HSB	LSB	R*

Field	Name	RW	Reset	Description
31-5	R*	R	0	Reserved
4	HCSB	R/W	0	Higher Channel Status/User Data Buffer Empty '1' for enable for higher channel status/user data buffer empty interrupt
3	LCSB	R/W	0	Lower Channel Status/User Data Buffer Empty '1' for enable for lower channel status/user data buffer empty interrupt
2	HSB	R/W	0	Higher Data Buffer Empty '1' for enable for higher data buffer empty interrupt
1	LSB	R/W	0	Lower Data Buffer Empty '1' for enable for lower data buffer empty interrupt
0	R	R	0	Reserved

SPDIF Transmit Interrupt Status Register 0xB010D010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											HCSB	LCSB	HSB	LSB	R*

Field	Name	RW	Reset	Description
31-5	R*	R	0	Reserved
4	HCSB	R/W	0	Higher Channel Status/User Data Buffer Empty '1' for higher channel status/user data buffer empty interrupt activated
3	LCSB	R/W	0	Lower Channel Status/User Data Buffer Empty '1' for lower channel status/user data buffer empty interrupt activated
2	HSB	R/W	0	Higher Data Buffer Empty '1' for higher data buffer empty interrupt activated
1	LSB	R/W	0	Lower Data Buffer Empty '1' for lower data buffer empty interrupt activated
0	R*	R	0	Reserved

SPDIF Transmit User Data Buffer Register 0xB010D080 ~ 0xB010D0DC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHBUD								CHAUD							

The user data is 24 bytes per sample block. Bit 0 of byte 0 is transmitted first. Bit 7 of byte 23 is transmitted last.

Field	Name	RW	Reset	Description
31-16	R*	R	0	Reserved
15-8	CHBUD	R/W	0	User Data for Channel B
7-0	CHAUD	R/W	0	User Data for Channel A

SPDIF Transmit Channel Status Buffer Register 0xB010D100 ~ 0xB010D15C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHBCS								CHACS							

The channel status data is 24 bytes per sample block. Bit 0 of byte 0 is transmitted first. Bit 7 of byte 23 is transmitted last.

Field	Name	RW	Reset	Description
31-16	R*	R	0	Reserved
15-8	CHBCS	R/W	0	Channel Status for Channel B
7-0	CHACS	R/W	0	Channel Status for Channel A

SPDIF Transmit Sample Data Buffer Register 0xB010D200 ~ 0xB010D3FC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								DATH							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATL															

The format of data words in transmit sample buffer. Channel A is transmitted first, and must be stored on even addresses, while channel B is stored on odd addresses in the sample buffer.

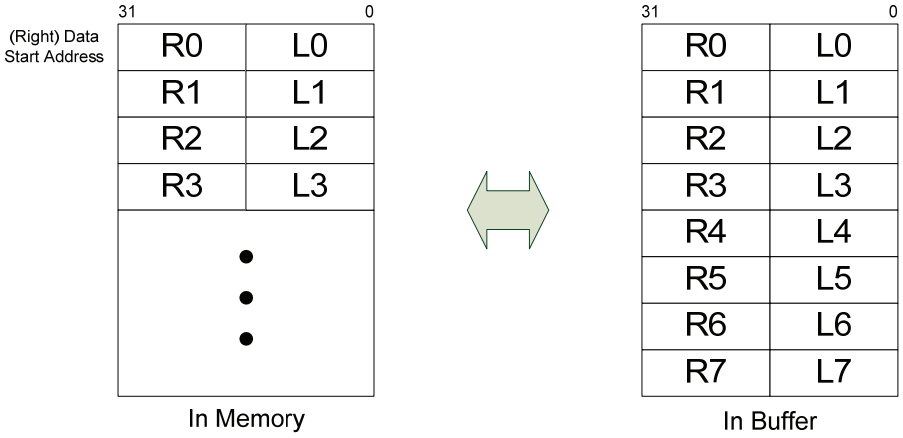
Field	Name	RW	Reset	Description
31-24	R*	R	0	Reserved
23-16	DATH	R/W	0	Upper 8 Bits for Sample Buffer Data Audio data if > 16 bits resolution. Unused bits are 0
15-0	DATL	R/W	0	Lower 16 Bits for Sample Buffer Data Audio data (16 bits mode). Bits 0 is LSB

DMA Configuration (DMACFG) 0xB010D400

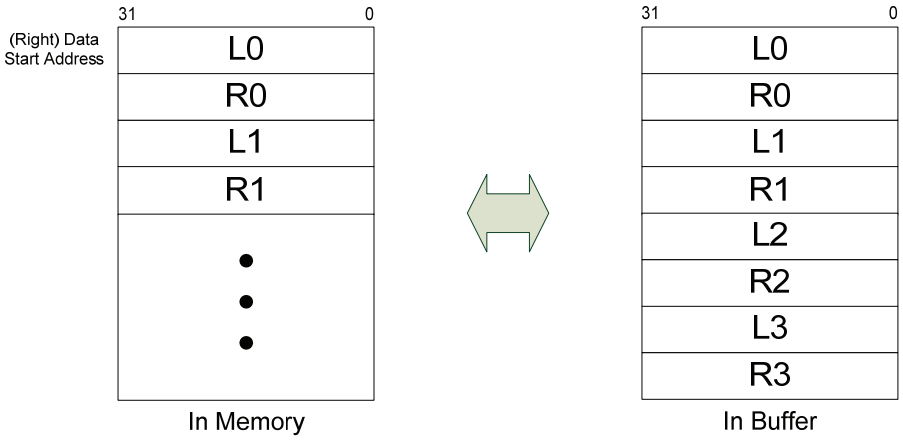
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											VCNT				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SS	RALR		DRQEN1	DRQEN0	AMODE1	AMODE0	FIFOCLR	Reseved			FIFOTH			

Field	Name	RW	Reset	Description
31-21	R*	R	0	Reserved
20-16	VCNT	R	X	FIFO Valid Entry Count
15	R*	R	0	Reserved
14	SS	R/W	0	Swap Sample 0 : Disable 1 : Enable
13-12	RALR	R/W	0	Read Address LR Mode 0 : Read Address 24bit Mode 1 : Read Address 24bit LR Mode 2 : Read Address 16bit Mode 3 : Read Address 16bit LR Mode
11	DRQEN1	R/W	0	DMA Request Enable for User Data Buffer
10	DRQEN0	R/W	0	DMA Request Enable for Sample Data Buffer
9	AMODE1	R/W	0	Sample Data Buffer Address Mode. Buffer would be written with sequence below. (0, 2, 4, 6, 1, 3, 5, 7)
8	AMODE0	R/W	0	0 : Ignore Address (FIFO Mode) 1 : Enable Address (16 Entries can be written with address)
7	FIFOCLR	R/W	0	Clear FIFO. Should be written with “0” for normal operation
6-4	R*	R	0	Reserved
3-0	FIFOTH	R/W	0x7	FIFO Threshold for DMA Request DMA request will be asserted if VCNT <= FIFOTH

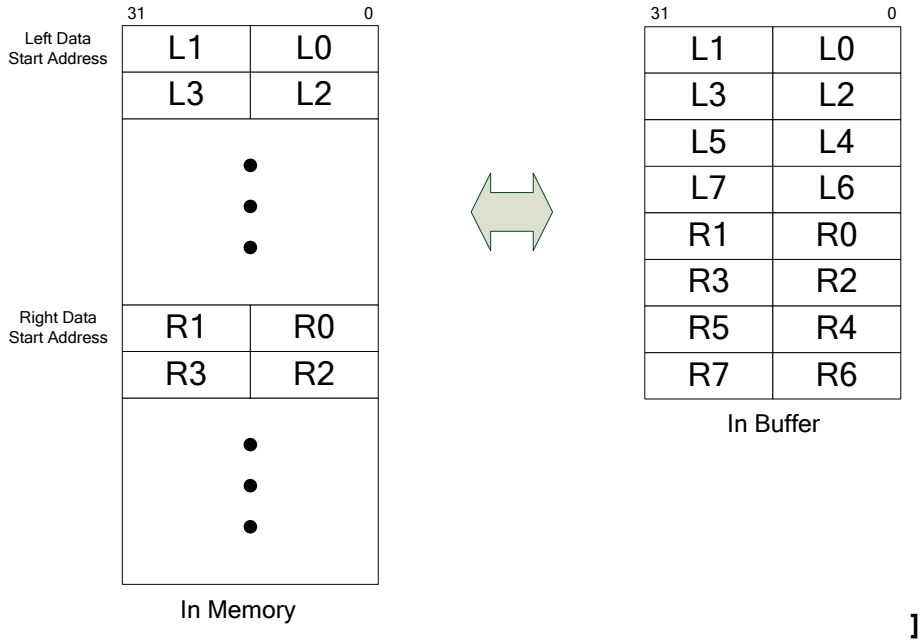
11.2.5 Audio Data Format



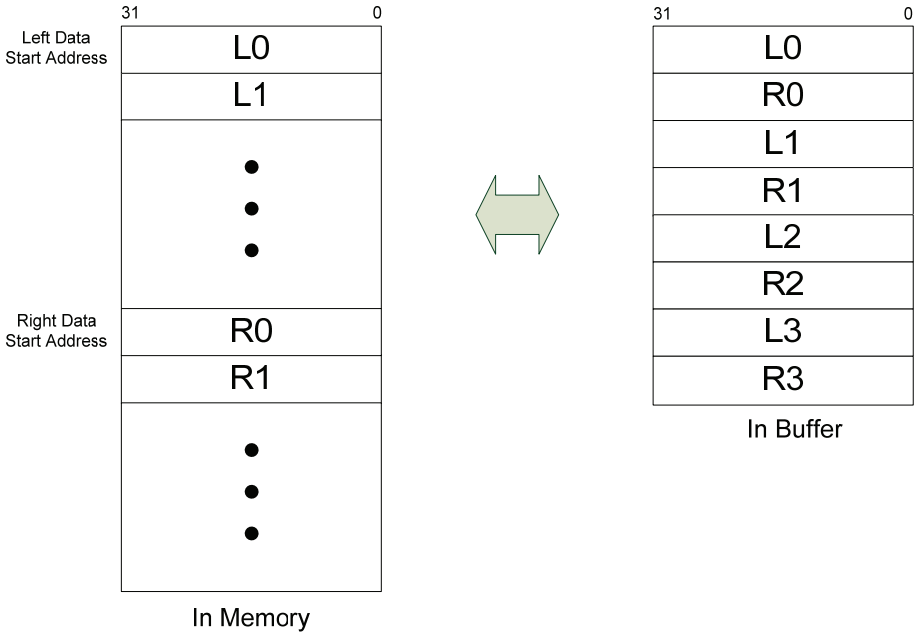
LRMode=0, Data_Width=16 bit



LRMode=0, Data_Width=24 bit



LRMode=1, Data_Width=16 bit



LRMode=1, Data_Width=24 bit

Figure 11.9 Data Format between Memory and Buffer

12 NFC

12.1 Function Description

NVS2310 has a NAND flash controller which can support various type of NAND flash such as 532(512 + 16)bytes/page, 2112(2048+64)bytes/page, 4314(4096 + 218)bytes/page NAND, 8644(8192 +448) bytes/page NAND and so on.

The supportable configuration for external NAND flash is as follows.

Case1) single NAND flash of 8-bit bus-width. (8bit I/O BUS cofiguration)
Case2) single NAND flash of 16-bit bus-width. (16bit I/O BUS configuration)
Case3) double NAND flashes of 8-bit bus-width. (16bit I/O BUS configuration)
Case4) double NAND flashes of 16-bit bus-width. (32bit I/O BUS configuration)
Case5) quad NAND flashes of 8-bit bus-width. (32bit I/O BUS configuration)

The supportable main features are as follows

- 8 bit, 16bit, 32bit I/O bus width is supportable
- AHB Master I/F with DMA Controller for burst data transfer
- AHB Slave I/F for command , address and single/burst data transfer.
- 1 page burst program and read (MAX page size : 8192Bytes + ECC size)
- Configurable read data capture timming to reduce read cycle period
- NFC dedicated memory(2048Bytes) for spare data(ECC or system related data)
- NAND 1 Ready or 4 Ready is supportable
- NFC dedicated ECC encoding/decoding capability
  - SLC ECC
  - MLC 4/6/12/16/24 bit ECC
  - MLC ECC Error Correction Pass

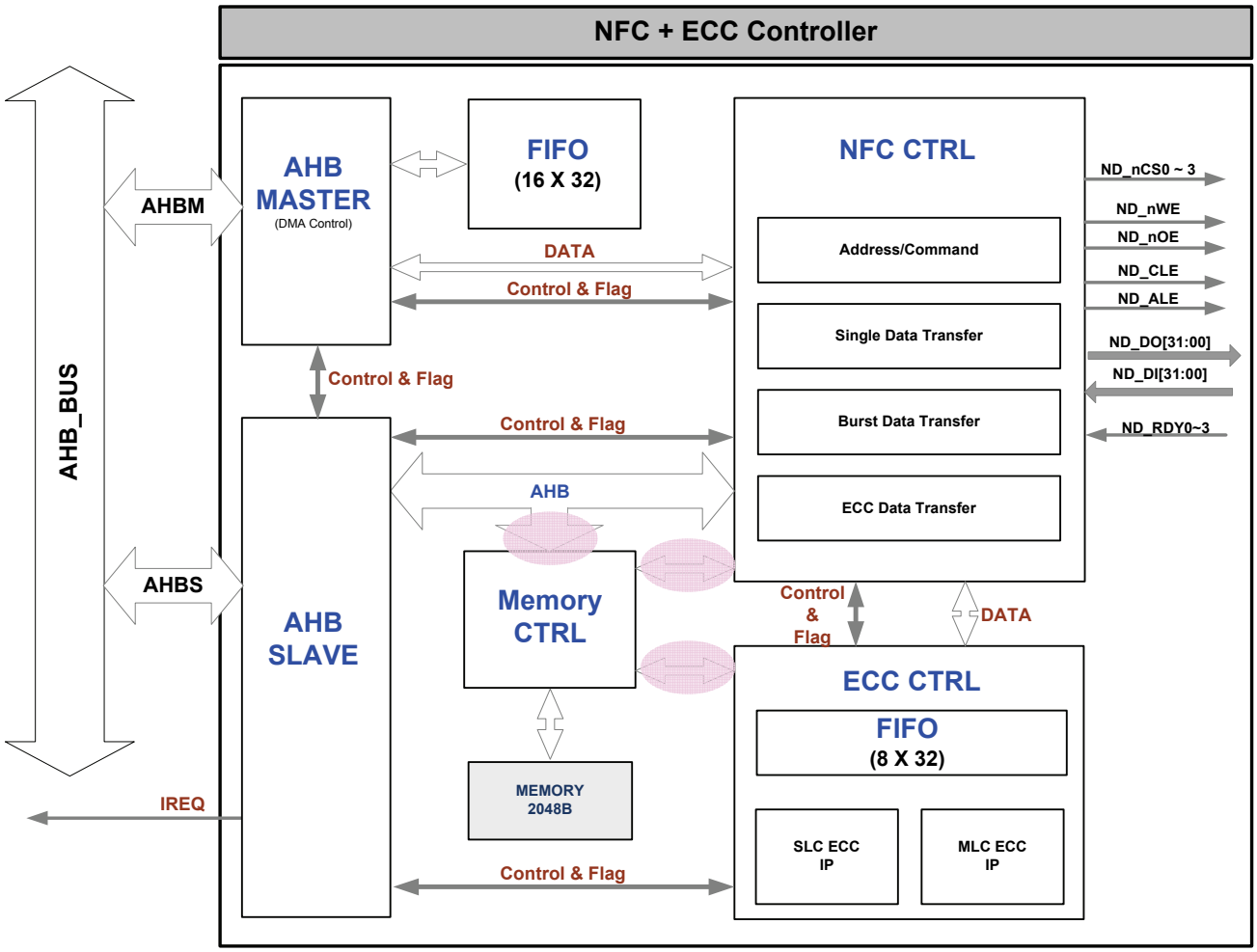


Figure 12.1 NAND Flash Controller Block Diagram

12.1.1 NAND Burst Program / Read By DMA

NVS2310 NFC(NAND Flash Controller) provides two modes of NAND Burst Program / Read. One is General DMA(GDMA) and another is NFC dedicated DMA (NDMA).

1) Burst Program / Read by GDMA(General DMA) with ECC
In this case, NAND burst program and NAND burst read operation can be executed by access to NAND Linear Data Register(NFC_LDATA) through GDMA. Maximum transfer data size is limited to 1024 Bytes (MAX ECC Data Size). If 1 page data size of NAND is larger than 1024 Bytes, the data is divided into less than 1024 Bytes in order to transmit DMA and to execute ECC.

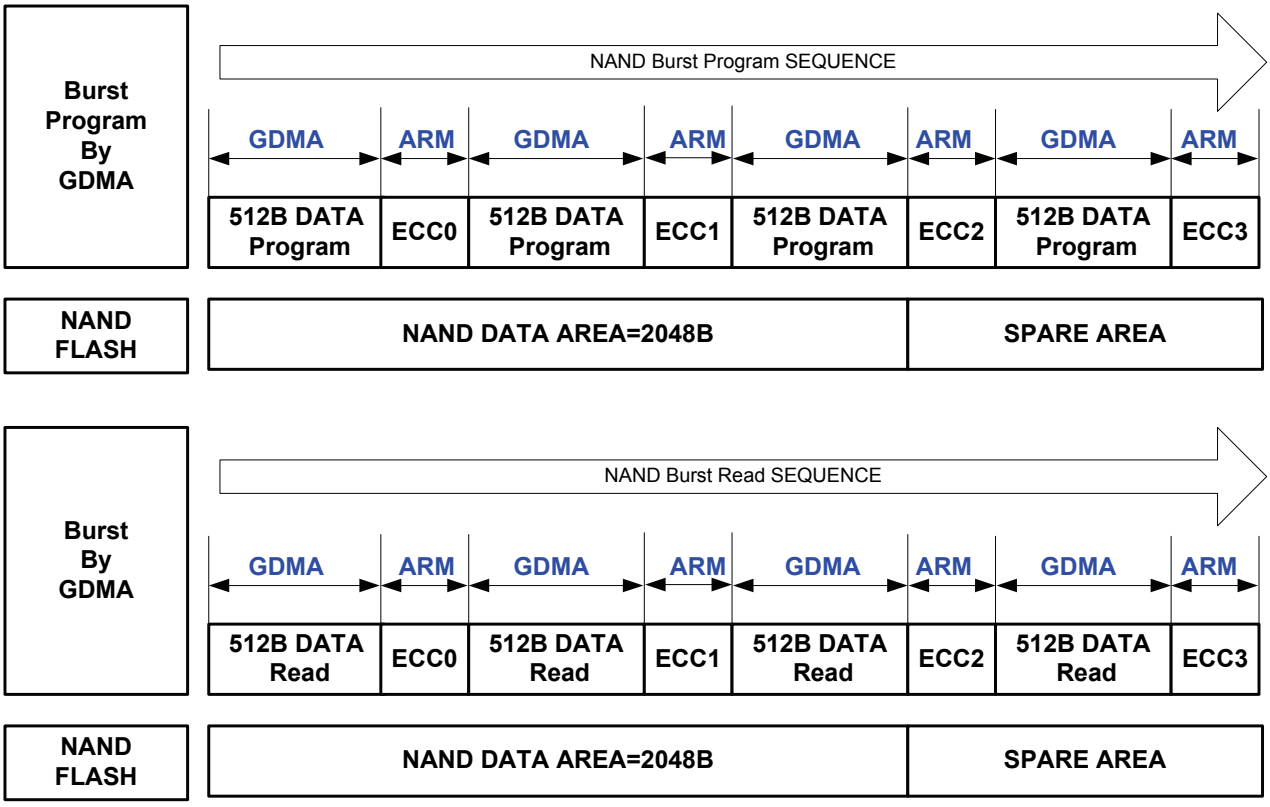


Figure 12.2 A Structure of NAND Flash for Burst Program/Read By GDMA

2) Burst Program / Read by NDMA(NFC DMA) with ECC
NAND burst program and read can be possible by NFC dedicated DMA(NDMA). In this case, even though a page size is larger than 1024 Bytes, 1 page burst program and read operation can be possible. However, data transfer and ECC are internally executed per each subpage and Each subpage should be configured by related register.
The number of subpage can be set up to 32. The max data size of the subpage is 1024 bytes.

Encoded/decoded ECC of each subpage data is orderly saved to NFC dedicated memory. For the saved ECC data, burst transfer is automatically executed after all of the subpaged data(subpage X N) is transferred.(NFC_CTRL[BMODE] == 1)
For example, if 1 page data area of NAND is 2048 bytes and ECC requirement is 12bit/512bytes, the number of sub-pages should be set to 4 and the data size of each subpage should be set 512 bytes in order to execute 1 page burst program and read operation via NDMA. The following figure is the page structure of NAND flash when burst read or program by NDMA is excuted.

As a result, NDMA can provide the better bandwidth than GDMA and reduce bus traffic.

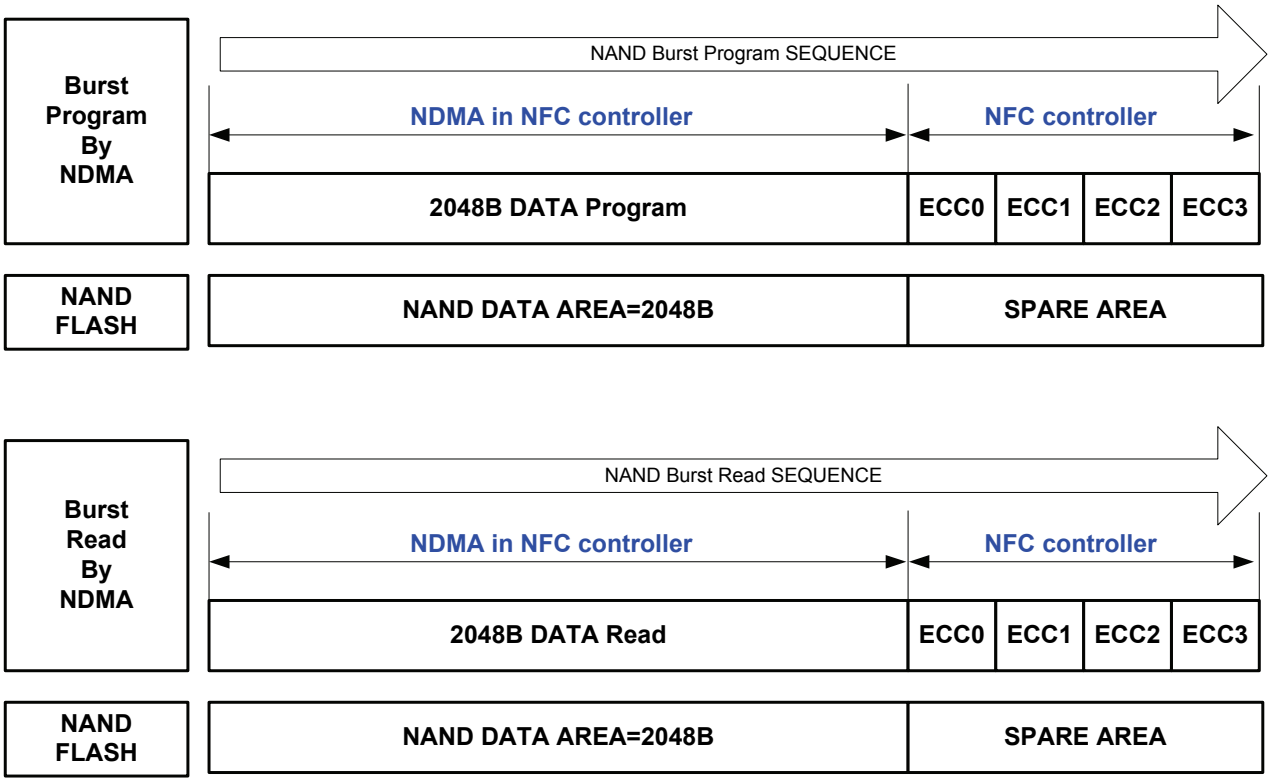


Figure 12.3 A Structure of NAND Flash for Burst Program/Read By NDMA

12.1.2 NAND Ready Configuration.

In NVS2310, READY input can be configured to 1 ready or 4 ready input of NAND Flash by IO_MCFG[RDYCFG]. In the case of 4 ready, ND_nCS[N](N=0,…,3) can mask its own ready input.

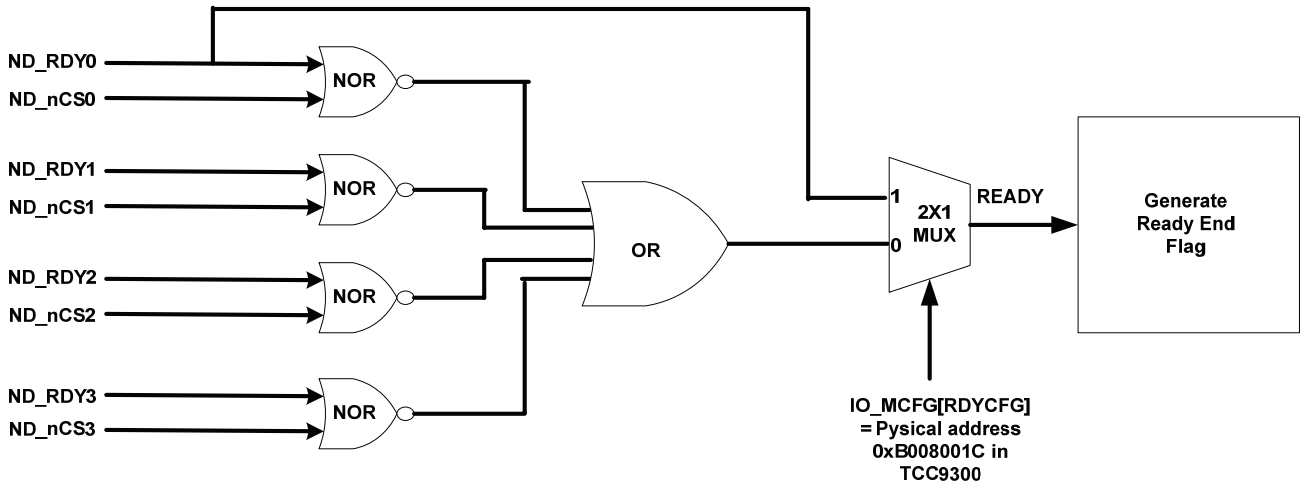


Figure 12.4 Ready

12.2 Register Description

Table 12.1 NAND Flash Controller Register Map (Base Address=0xB0050000)

Name	Address	Type	Reset	Description
NFC_CMD	0x00	W	-	NAND Flash Command Register
NFC_LADDR	0x04	W	-	NAND Flash Linear Address Register
NFC_SADDR	0x08	W	-	NAND Flash Signal Address Register
NFC_SDATA	0x0C	R/W	Unknown	NAND Flash Single Data Register
NFC_WDATA	0x1x	R/W	Unknown	NAND Flash Word Data Register
NFC_LDATA	0x2x/3x	R/W	Unknown	NAND Flash Linear Data Register
NFC_MDATA	0x40	R/W	Unknown	NAND Flash Multiple Data Register
NFC_CACYC	0x44	R/W	0x00010101	NAND Flash Command & Address Cycle Register
NFC_WRCYC	0x48	R/W	0x00010101	NAND Flash Write Cycle Register
NFC_RECYC	0x4C	R/W	0x00010101	NAND Flash Read Cycle Register
NFC_CTRL	0x50	R/W	0xF006C800	NFC Control Register
NFC_DSIZE	0x54	R/W	0x0000FFFF	NFC Data Size Register
NFC_SPARE	0x58	R/W	0x00000200	NFC Spare Burst Transfer Control Register
NFC_PRSTART	0x5C	R/W	0x00000000	NFC Burst Program/Read Start
NFC_RST	0X60	R/W	0x00000000	NFC Reset Register
NFC_IRQCFG	0x64	R/W	0x00000000	NFC Interrupt Request Control Register
NFC_IRQ	0x68	R/W	0x00000000	NFC Interrupt Request Register
NFC_STATUS	0x6C	R/W		NFC Status Register
NFC_RFWBASE	0x70	R/W	0x00000000	NFC Read Flag Wait Base Word Register
NDMA_ADDR	0X100	R/W	0x00000000	NFC DMA Source/Destination Register
NDMA_PARAM	0X104	R/W	0x00000000	NFC DMA Parameter Register
NDMA_CADDR	0X108	R	0x00000000	NFC DMA Current Address Register
NDMA_CTRL	0x10C	R/W	0x00000000	NFC DMA Control Register
NDMA_SPCTRL	0x110	R/W	0x00000000	NFC DMA SUBPAGE Control Regiter
NDMA_STATUS	0x114	R/W	0x00000000	NFC DMA Status Register
SP_CFG0	0x120	R/W	0x00000000	NFC Sub Page Configuration Register 0
SP_CFG1	0x124	R/W	0x00000000	NFC Sub Page Configuration Register 1
SP_CFG2	0x128	R/W	0x00000000	NFC Sub Page Configuration Register 2
SP_CFG3	0x12C	R/W	0x00000000	NFC Sub Page Configuration Register 3
SP_CFG4	0x130	R/W	0x00000000	NFC Sub Page Configuration Register 4
SP_CFG5	0x134	R/W	0x00000000	NFC Sub Page Configuration Register 5
SP_CFG6	0x138	R/W	0x00000000	NFC Sub Page Configuration Register 6
SP_CFG7	0x13C	R/W	0x00000000	NFC Sub Page Configuration Register 7
SP_CFG8	0x140	R/W	0x00000000	NFC Sub Page Configuration Register 8
SP_CFG9	0x144	R/W	0x00000000	NFC Sub Page Configuration Register 9
SP_CFG10	0x148	R/W	0x00000000	NFC Sub Page Configuration Register 10
SP_CFG11	0x14C	R/W	0x00000000	NFC Sub Page Configuration Register 11
SP_CFG12	0x150	R/W	0x00000000	NFC Sub Page Configuration Register 12
SP_CFG13	0x154	R/W	0x00000000	NFC Sub Page Configuration Register 13
SP_CFG14	0x158	R/W	0x00000000	NFC Sub Page Configuration Register 14
SP_CFG15	0x15C	R/W	0x00000000	NFC Sub Page Configuration Register 15
SP_CFG16	0x160	R/W	0x00000000	NFC Sub Page Configuration Register 16
SP_CFG17	0x164	R/W	0x00000000	NFC Sub Page Configuration Register 17
SP_CFG18	0x168	R/W	0x00000000	NFC Sub Page Configuration Register 18
SP_CFG19	0x16C	R/W	0x00000000	NFC Sub Page Configuration Register 19
SP_CFG20	0x170	R/W	0x00000000	NFC Sub Page Configuration Register 20
SP_CFG21	0x174	R/W	0x00000000	NFC Sub Page Configuration Register 21
SP_CFG22	0x178	R/W	0x00000000	NFC Sub Page Configuration Register 22
SP_CFG23	0x17C	R/W	0x00000000	NFC Sub Page Configuration Register 23
SP_CFG24	0x180	R/W	0x00000000	NFC Sub Page Configuration Register 24
SP_CFG25	0x184	R/W	0x00000000	NFC Sub Page Configuration Register 25
SP_CFG26	0x188	R/W	0x00000000	NFC Sub Page Configuration Register 26
SP_CFG27	0x18C	R/W	0x00000000	NFC Sub Page Configuration Register 27
SP_CFG28	0x190	R/W	0x00000000	NFC Sub Page Configuration Register 28
SP_CFG29	0x194	R/W	0x00000000	NFC Sub Page Configuration Register 29
SP_CFG30	0x198	R/W	0x00000000	NFC Sub Page Configuration Register 30

SP_CFG31	0x19C	R/W	0x00000000	NFC Sub Page Configuration Register 31
ECC_CTRL	0x200	R/W	0x00000000	ECC Control Register
SECC_ERRNUM	0x204	R/W	0x00000000	SLC ECC Error Number Register
MECC_ERRNUM	0x204	R/W	0x00000000	MLC ECC Error Number Register
ECC_BASE	0x208	R/W	-	Reserved
ECC_MASK	0x20C	R/W	-	Reserved
ECC_CLEAR	0x210	R/W	0x00000000	NFC ECC Register Clear
ECC_RSTART	0x214	R/W	-	NFC ECC Decoding Start
ECC_STLDCTL	0x218	R/W	0x00000000	NFC ECC Store Status and Load Control Register
ECC_STATUS	0x21C	R/W	0x00000001	NFC ECC STATUS Register
SECC_0	0x220	R/W	0x00000000	1st SLC ECC Code Register
SECC_1	0x224	R/W	0x00000000	2nd SLC ECC Code Register
SECC_2	0x228	R/W	0x00000000	3rd SLC ECC Code Register
SECC_3	0x22C	R/W	0x00000000	4th SLC ECC Code Register
SECC_4	0x230	R/W	0x00000000	5th SLC ECC Code Register
SECC_5	0x234	R/W	0x00000000	6th SLC ECC Code Register
SECC_6	0x238	R/W	0x00000000	7th SLC ECC Code Register
SECC_7	0x23C	R/W	0x00000000	8th SLC ECC Code Register
SECC_8	0x240	R/W	0x00000000	9th SLC ECC Code Register
SECC_9	0x244	R/W	0x00000000	10th SLC ECC Code Register
SECC_10	0x248	R/W	0x00000000	11th SLC ECC Code Register
SECC_11	0x24C	R/W	0x00000000	12th SLC ECC Code Register
SECC_12	0x250	R/W	0x00000000	13th SLC ECC Code Register
SECC_13	0x254	R/W	0x00000000	14th SLC ECC Code Register
SECC_14	0x258	R/W	0x00000000	15th SLC ECC Code Register
SECC_15	0x25C	R/W	0x00000000	16th SLC ECC Code Register
MECC4_0	0x220	R/W	0x00000000	1st MLC ECC4 Code Register
MECC4_1	0x224	R/W	0x00000000	2nd MLC ECC4 Code Register
MECC6_0	0x220	R/W	0x00000000	1st MLC ECC6 Code Register
MECC6_1	0x224	R/W	0x00000000	2nd MLC ECC6 Code Register
MECC6_2	0x228	R/W	0x00000000	3rd MLC ECC6 Code Register
MECC12_0	0x220	R/W	0x00000000	1st MLC ECC12 Code Register
MECC12_1	0x224	R/W	0x00000000	2nd MLC ECC12 Code Register
MECC12_2	0x228	R/W	0x00000000	3rd MLC ECC12 Code Register
MECC12_3	0x22C	R/W	0x00000000	4th MLC ECC12 Code Register
MECC12_4	0x230	R/W	0x00000000	5^{th} MLC ECC12 Code Register
MECC12_5	0x234	R/W	0x00000000	6^{th} MLC ECC12 Code Register
MECC16_0	0x220	R/W	0x00000000	1st MLC ECC14 Code Register
MECC16_1	0x224	R/W	0x00000000	2nd MLC ECC14 Code Register
MECC16_2	0x228	R/W	0x00000000	3rd MLC ECC14 Code Register
MECC16_3	0x22C	R/W	0x00000000	4th MLC ECC14 Code Register
MECC16_4	0x230	R/W	0x00000000	5th MLC ECC14 Code Register
MECC16_5	0x234	R/W	0x00000000	6th MLC ECC14 Code Register
MECC16_6	0x238	R/W	0x00000000	7th MLC ECC14 Code Register
MECC16_7	0x23C	R/W	0x00000000	8th MLC ECC14 Code Register
MECC24_0	0x220	R/W	0x00000000	1st MLC ECC16 Code Register
MECC24_1	0x224	R/W	0x00000000	2nd MLC ECC16 Code Register
MECC24_2	0x228	R/W	0x00000000	3rd MLC ECC16 Code Register
MECC24_3	0x22C	R/W	0x00000000	4th MLC ECC16 Code Register
MECC24_4	0x230	R/W	0x00000000	5th MLC ECC16 Code Register
MECC24_5	0x234	R/W	0x00000000	6th MLC ECC16 Code Register
MECC24_6	0x238	R/W	0x00000000	7th MLC ECC16 Code Register
MECC24_7	0x23C	R/W	0x00000000	8th MLC ECC16 Code Register
MECC24_8	0x240	R/W	0x00000000	9th MLC ECC16 Code Register
MECC24_9	0x244	R/W	0x00000000	10th MLC ECC16 Code Register
MECC24_10	0x248	R/W	0x00000000	11th MLC ECC16 Code Register

SECC_EADDR0	0x260	R	0x00000000	SLC ECC Error Address Register 0
SECC_EADDR1	0x264	R	0x00000000	SLC ECC Error Address Register 1
SECC_EADDR2	0x268	R	0x00000000	SLC ECC Error Address Register 2
SECC_EADDR3	0x26C	R	0x00000000	SLC ECC Error Address Register 3
SECC_EADDR4	0x270	R	0x00000000	SLC ECC Error Address Register 4
SECC_EADDR5	0x274	R	0x00000000	SLC ECC Error Address Register 5
SECC_EADDR6	0x278	R	0x00000000	SLC ECC Error Address Register 6
SECC_EADDR7	0x27C	R	0x00000000	SLC ECC Error Address Register 7
SECC_EADDR8	0x280	R	0x00000000	SLC ECC Error Address Register 8
SECC_EADDR9	0x284	R	0x00000000	SLC ECC Error Address Register 9
SECC_EADDR10	0x288	R	0x00000000	SLC ECC Error Address Register 10
SECC_EADDR11	0x28C	R	0x00000000	SLC ECC Error Address Register 11
SECC_EADDR12	0x290	R	0x00000000	SLC ECC Error Address Register 12
SECC_EADDR13	0x294	R	0x00000000	SLC ECC Error Address Register 13
SECC_EADDR14	0x298	R	0x00000000	SLC ECC Error Address Register 14
SECC_EADDR15	0x29C	R	0x00000000	SLC ECC Error Address Register 15
MECC4_EADDR0	0x260	R	0x00000000	MLC ECC4 Error Address Register 0
MECC4_EADDR1	0x264	R	0x00000000	MLC ECC4 Error Address Register 1
MECC4_EADDR2	0x268	R	0x00000000	MLC ECC4 Error Address Register 2
MECC4_EADDR3	0x26C	R	0x00000000	MLC ECC4 Error Address Register 3
MECC6_EADDR0	0x260	R	0x00000000	MLC ECC6 Error Address Register 0
MECC6_EADDR1	0x264	R	0x00000000	MLC ECC6 Error Address Register 1
MECC6_EADDR2	0x268	R	0x00000000	MLC ECC6 Error Address Register 2
MECC6_EADDR3	0x26C	R	0x00000000	MLC ECC6 Error Address Register 3
MECC6_EADDR4	0x270	R	0x00000000	MLC ECC6 Error Address Register 4
MECC6_EADDR5	0x27C	R	0x00000000	MLC ECC6 Error Address Register 5
MECC12_EADDR0	0x260	R	0x00000000	MLC ECC12 Error Address Register 0
MECC12_EADDR1	0x264	R	0x00000000	MLC ECC12 Error Address Register 1
MECC12_EADDR2	0x268	R	0x00000000	MLC ECC12 Error Address Register 2
MECC12_EADDR3	0x26C	R	0x00000000	MLC ECC12 Error Address Register 3
MECC12_EADDR4	0x270	R	0x00000000	MLC ECC12 Error Address Register 4
MECC12_EADDR5	0x274	R	0x00000000	MLC ECC12 Error Address Register 5
MECC12_EADDR6	0x278	R	0x00000000	MLC ECC12 Error Address Register 6
MECC12_EADDR7	0x27C	R	0x00000000	MLC ECC12 Error Address Register 7
MECC12_EADDR8	0x280	R	0x00000000	MLC ECC12 Error Address Register 8
MECC12_EADDR9	0x284	R	0x00000000	MLC ECC12 Error Address Register 9
MECC12_EADDR10	0x288	R	0x00000000	MLC ECC12 Error Address Register 10
MECC12_EADDR11	0x28C	R	0x00000000	MLC ECC12 Error Address Register 11
MECC16_EADDR0	0x260	R	0x00000000	MLC ECC16 Error Address Register 0
MECC16_EADDR1	0x264	R	0x00000000	MLC ECC16 Error Address Register 1
MECC16_EADDR2	0x268	R	0x00000000	MLC ECC16 Error Address Register 2
MECC16_EADDR3	0x26C	R	0x00000000	MLC ECC16 Error Address Register 3
MECC16_EADDR4	0x270	R	0x00000000	MLC ECC16 Error Address Register 4
MECC16_EADDR5	0x274	R	0x00000000	MLC ECC16 Error Address Register 5
MECC16_EADDR6	0x278	R	0x00000000	MLC ECC16 Error Address Register 6
MECC16_EADDR7	0x27C	R	0x00000000	MLC ECC16 Error Address Register 7
MECC16_EADDR8	0x280	R	0x00000000	MLC ECC16 Error Address Register 8
MECC16_EADDR9	0x284	R	0x00000000	MLC ECC16 Error Address Register 9
MECC16_EADDR10	0x288	R	0x00000000	MLC ECC16 Error Address Register 10
MECC16_EADDR11	0x28C	R	0x00000000	MLC ECC16 Error Address Register 11
MECC16_EADDR12	0x290	R	0x00000000	MLC ECC16 Error Address Register 12
MECC16_EADDR13	0x294	R	0x00000000	MLC ECC16 Error Address Register 13
MECC16_EADDR14	0x298	R	0x00000000	MLC ECC16 Error Address Register 14
MECC16_EADDR15	0x29C	R	0x00000000	MLC ECC16 Error Address Register 15
MECC24_EADDR0	0x260	R	0x00000000	MLC ECC24 Error Address Register 0
MECC24_EADDR1	0x264	R	0x00000000	MLC ECC24 Error Address Register 1
MECC24_EADDR2	0x268	R	0x00000000	MLC ECC24 Error Address Register 2
MECC24_EADDR3	0x26C	R	0x00000000	MLC ECC24 Error Address Register 3
MECC24_EADDR4	0x270	R	0x00000000	MLC ECC24 Error Address Register 4

MECC24_EADDR5	0x274	R	0x00000000	MLC ECC24 Error Address Register 5
MECC24_EADDR6	0x278	R	0x00000000	MLC ECC24 Error Address Register 6
MECC24_EADDR7	0x27C	R	0x00000000	MLC ECC24 Error Address Register 7
MECC24_EADDR8	0x280	R	0x00000000	MLC ECC24 Error Address Register 8
MECC24_EADDR9	0x284	R	0x00000000	MLC ECC24 Error Address Register 9
MECC24_EADDR10	0x288	R	0x00000000	MLC ECC24 Error Address Register 10
MECC24_EADDR11	0x28C	R	0x00000000	MLC ECC24 Error Address Register 11
MECC24_EADDR12	0x290	R	0x00000000	MLC ECC24 Error Address Register 12
MECC24_EADDR13	0x294	R	0x00000000	MLC ECC24 Error Address Register 13
MECC24_EADDR14	0x298	R	0x00000000	MLC ECC24 Error Address Register 14
MECC24_EADDR15	0x29C	R	0x00000000	MLC ECC24 Error Address Register 15
MECC24_EADDR16	0x2A0	R	0x00000000	MLC ECC24 Error Address Register 16
MECC24_EADDR17	0x2A4	R	0x00000000	MLC ECC24 Error Address Register 17
MECC24_EADDR18	0x2A8	R	0x00000000	MLC ECC24 Error Address Register 18
MECC24_EADDR19	0x2AC	R	0x00000000	MLC ECC24 Error Address Register 19
MECC24_EADDR20	0x2B0	R	0x00000000	MLC ECC24 Error Address Register 20
MECC24_EADDR21	0x2B4	R	0x00000000	MLC ECC24 Error Address Register 21
MECC24_EADDR22	0x2B8	R	0x00000000	MLC ECC24 Error Address Register 22
MECC24_EADDR23	0x2BC	R	0x00000000	MLC ECC24 Error Address Register 23
NFC_MEMORY	0x400~0x9FF	R/W		NFC MEMORY for SPARE Area

Command Register (NFC_CMD) **0xB0050000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CMD3[7:0]								CMD2[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMD1[7:0]								CMD0[7:0]							

For 32bit bus width configuration of 8 bit NAND flash or 16 bit NAND flash, NFC_CMD[CMD3] downto NFC_CMD[CMD0] is used as command register. (NFC_CTRL[BW] = 2'b10)

For 16bit bus width configuration of 8 bit NAND flash or 16 bit NAND flash, the NFC_CMD[CMD1] downto NFC_CMD[CMD0] is used as command register. NFC_CMD[CMD3] and NFC_CMD[CMD2] is ignored.(NFC_CTRL[BW] = 2'b01)

For 8bit bus width of NAND flash, the NFC_CMD[CMD0] is used as command register. NFC_CMD[CMD3] downto NFC_CMD[CMD1] is ignored. (NFC_CTRL[BW] = 2'b00)
If MSK of NFC_CTRL is high, the NFC_CMD[CMD3] downto NFC_CMD[CMD1] is masked.

1. NFC_CTRL[BW]=2'01(16bit BW)

NFC_CMD	MSK=1	MSK=0	I/O[15:00]	
			MSK=1	MSK=0
0x00000000	0x00000000	0x00000000	0x0000	0x0000
0x80808080	0x00000080	0x00008080	0x0080	0x8080
0x60606060	0x00000060	0x00006060	0x0060	0x6060
0x70707070	0x00000070	0x00007070	0x0070	0x7070
…	…	…	…	…

2. NFC_CTRL[BW]=2'10(32bit BW)

NFC_CMD	MSK=1	MSK=0	I/O[31:00]	
			MSK=1	MSK=0
0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x80808080	0x00000080	0x80808080	0x00000080	0x80808080
0x60606060	0x00000060	0x60606060	0x00000060	0x60606060
0x70707070	0x00000070	0x70707070	0x00000070	0x70707070
…	…	…	…	…

Linear Address Register (NFC_LADDR) 0xB0050004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LADDR[15:0]															

By writing to this register, memory controller generates linear address shown in Table 12.2

For NFC_LADDR = 0x12345600, CADDR NFC_CTRL[14:12]) = 2 and PSIZE(NFC_CTRL[17:16]) = 1 (512 bytes)

- MSK(NFC_CTRL[15]) = 1 : ND_IO[15:0] = 0x0000 – 0x002B – 0x001A
- MSK(NFC_CTRL[15]) = 0 : ND_IO[15:0] = 0x0000 – 0x2B2B – 0x1A1A

Single Address Cycle Register (NFC_SADDR) 0xB0050008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SADDR3[7:0]								SADDR2[7:0]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SADDR1[7:0]								SADDR0[7:0]							

For 32bit bus width configuration of 8 bit NAND flash or 16 bit NAND flash, NFC_SADDR[SADDR3] downto NFC_SADDR[SADDR0] is used as single address register. (NFC_CTRL[BW] = 2'b10)

For 16bit bus width configuration of 8 bit NAND flash or 16 bit NAND flash, the NFC_SADDR[SADDR1] downto NFC_SADDR[SADDR0] is used as single address register. NFC_SADDR[SADDR3] and NFC_SADDR[SADDR2] is ignored.(NFC_CTRL[BW] = 2'b01)

For 8bit bus width of NAND flash, the NFC_SADDR[SADDR0] is used as single address register. NFC_SADDR[SADDR3] downto NFC_SADDR[SADDR1] is ignored. (NFC_CTRL[BW] = 2'b00)

If MSK of NFC_CTRL is high, the NFC_SADDR[SADDR3] downto NFC_SADDR[SADDR1] is masked.

For example)
1. NFC_CTRL[BW]=2'01(16bit BW)

NFC_SADDR	MSK=1	MSK=0	I/O[15:00]	
			MSK=1	MSK=0
0xFFFFFFFF	0x000000FF	0x0000FFFF	0x00FF	0xFFFF
0xAAAAAAA	0x000000AA	0x0000AAAA	0x00AA	0xAAAA
…	…	…	…	…

2. NFC_CTRL[BW]=2'10(32bit BW)

NFC_SADDR	MSK=1	MSK=0	I/O[31:00]	
			MSK=1	MSK=0
0xFFFFFFFF	0x000000FF	0xFFFFFFFF	0x000000FF	0xFFFFFFFF
0xAAAAAAA	0x000000AA	0xAAAAAAAA	0x000000AA	0xAAAAAAAA
…	…	…	…	…

Table 12.2 Page Size of NAND Flash

# of Cycle (CADDR)	PSIZE = 0	PSIZE = 1	PSIZE = 2	PSIZE = 3	PSIZE > 3
1st	ADDR[7:0]	ADDR[7:0]	ADDR[7:0]	ADDR[7:0]	ADDR[7:0]
2nd	ADDR[16:9]	ADDR[16:9]	ADDR[10:8]	ADDR[11:8]	ADDR[12:8]
3rd	ADDR[24:17]	ADDR[24:17]	ADDR[18:11]	ADDR[19:12]	ADDR[20:13]
4th	ADDR[31:25]	ADDR[31:25]	ADDR[26:19]	ADDR[27:20]	ADDR[28:21]
5th	-	-	ADDR[31:27]	ADDR[31:28]	ADDR[31:29]

The Figure 12.5 represents the relation between each cycle and address generation.

User must set this information appropriately to PSIZE and CADDR field of NFC_CTRL register ahead of accessing NAND data.

ADDR means address value that is written to NFC_LADDR or NFC_BADDR register. The shaded cycles represent Block address cycles. That is, NAND address cycles start from there when NFC_BADDR register is accessed.

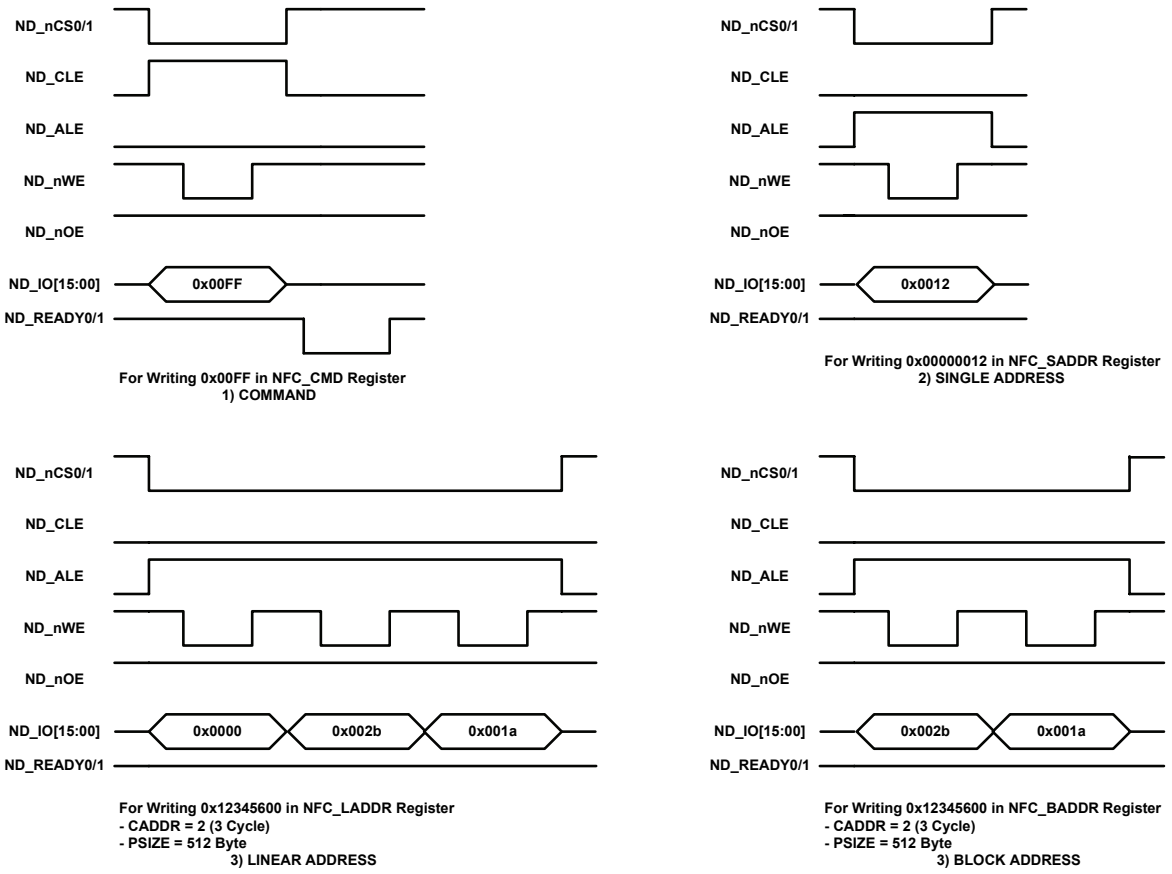


Figure 12.5 Example of Address/Command Writing Operation

Word Data Register (NFC_WDATA) 0xB005001x

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFC_WDATA[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NFC_WDATA[15:0]															

This register is used for single word data transfer by CPU.

This Register is useful in reading and writing NAND Flash data of spare area.

Linear Data Register (NFC_LDATA) 0xB005002x/0xB005003x

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFC_LDATA[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NFC_LDATA[15:00]															

This register is used for burst data transfer by DMA/CPU.

To start burst data transfer, User must write any value to NFC_PSTART or NFC_RSTART

Multiple Data Register (NFC_MDATA) 0xB0050040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NFC_MDATA[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NFC_MDATA[15:0]															

This register is used for reading and writing multiple 8bit or 16bit data according to NAND Flash Bus Size.
NFC_CTRL.DNUM[01:00] register define the number of multiple 8 bit or 16 bit data for

CMD/ADDRESS CYCLE **0xB0050044**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								STP							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PW								HLD[7:0]							

Note) Command / Address Cycle Configuration Register.

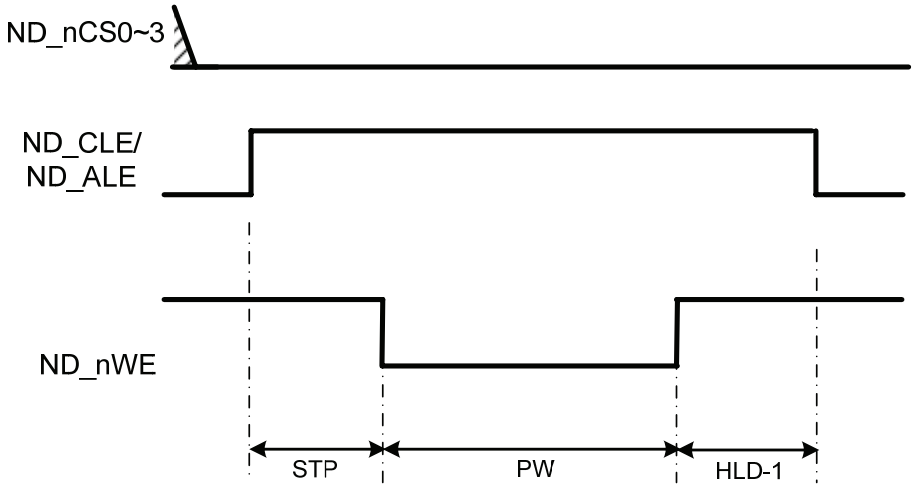


Figure 12.6 Timing Diagram of Read/Write Enable Signal for Command or Address Cycle

WRITE CYCLE **0xB0050048**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								STP							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PW								HLD[7:0]							

Note) Write Cycle Configuration Register. Refer to CMD/ADDR Cycle Configuration Register

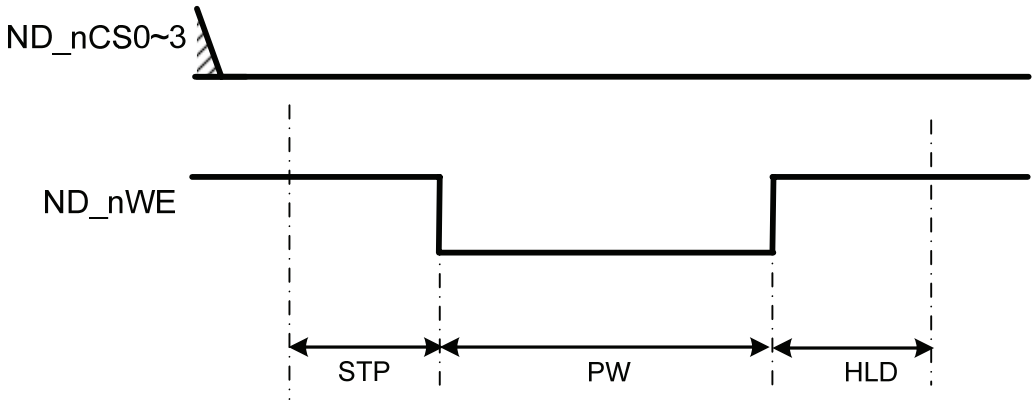


Figure 12.7 Timing Diagram of Write Enable Signal for Write cycle

READ CYCLE **0xB005004C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved								STP							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PW								HLD[7:0]							

Note) Read Cycle Configuration Register.

For Single / Word data read (SDATA/WDATA/MDATA) : The pulse width is expanded as much as NFC_CTRL[RDNDLY] value.

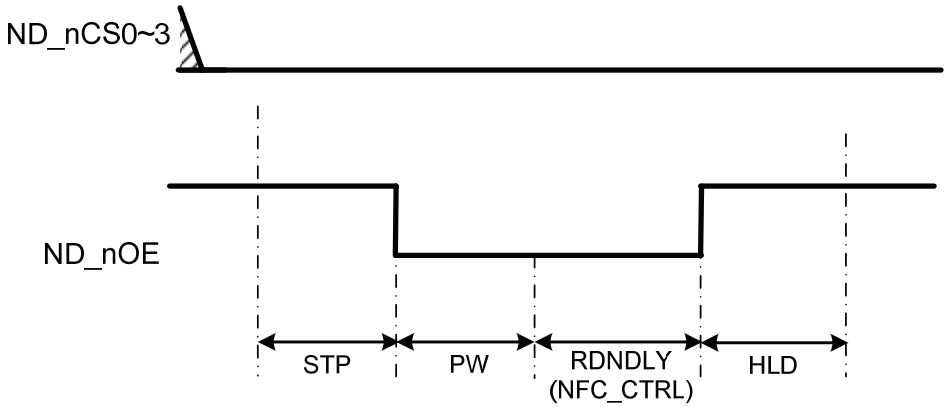


Figure 12.8 Timing Diagram of Read Enable Signal for Single Read cycle

For Burst data read (LDATA or NDMA)

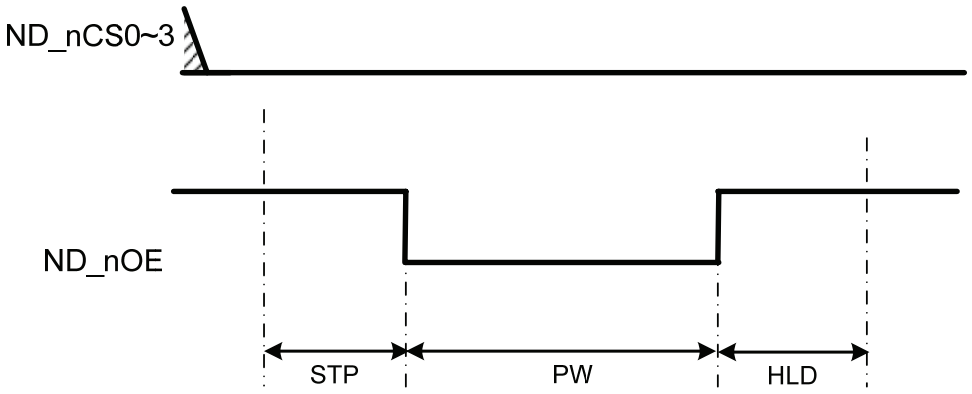


Figure 12.9 Timing Diagram of Read Enable Signal for Burst Read cycle

NFC CTRL **0xB0050050**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CS3	CS2	CS1	CS0	RDNDLY			ARFW				DACK	DMODE	BMODE	ECCON	PRSEL
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DNUM[01:00]				RM	MSK	BW		BSIZE		PSIZE			CADDR[2:0]		

Field	Name	RW	Reset	Description
31	CS3SEL	R/W	0x1	NAND Flash CS3 Selection 0 : NAND Flash nCS3 is Enable 1 : NAND Flash nCS3 is Disable
30	CS2SEL	R/W	0x1	NAND Flash CS2 Selection 0 : NAND Flash nCS2 is Enable 1 : NAND Flash nCS2 is Disable
29	CS1SEL	R/W	0x1	NAND Flash CS1 Selection 0 : NAND Flash nCS1 is Enable 1 : NAND Flash nCS1 is Disable
28	CS0SEL	R/W	0x1	NAND Flash CS0 Selection 0 : NAND Flash nCS0 is Enable 1 : NAND Flash nCS0 is Disable

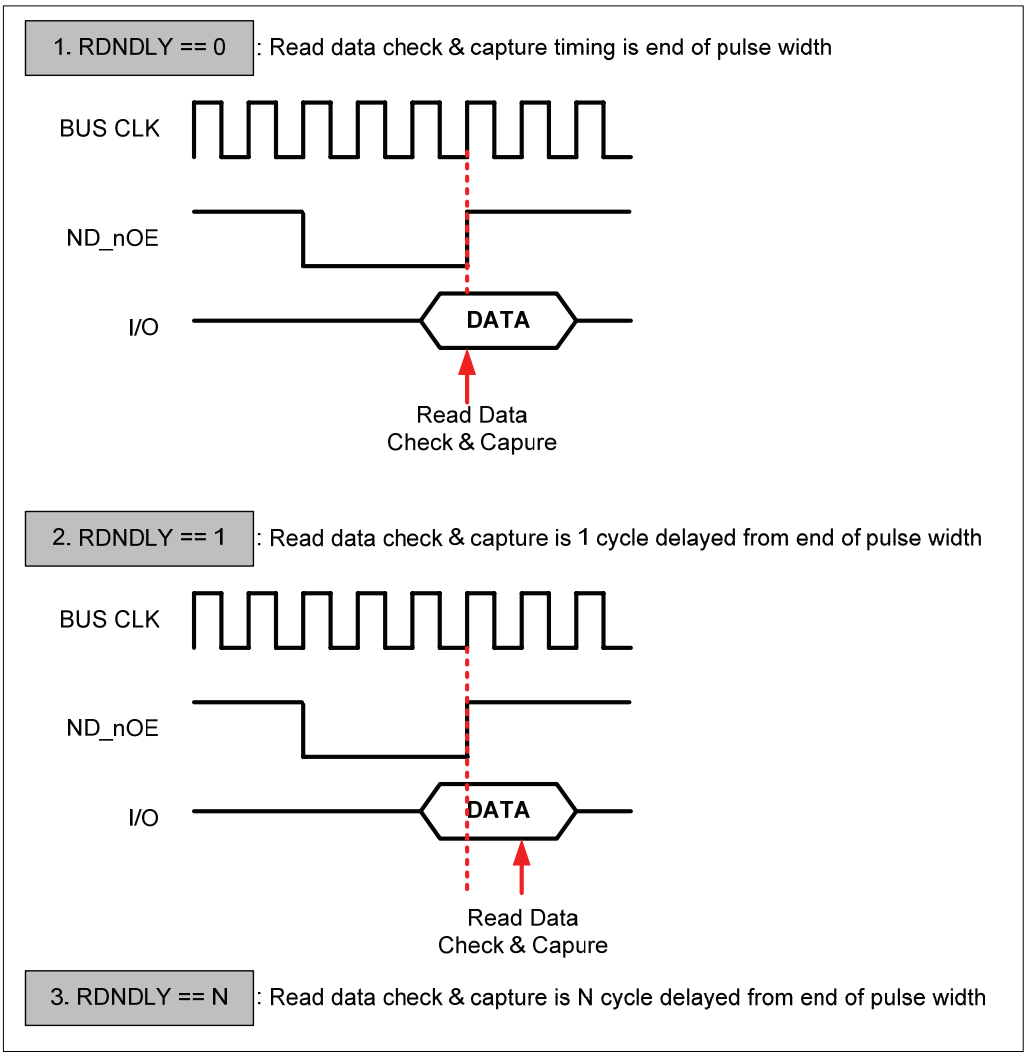
Field	Name	RW	Reset	Description
27-25	RDNDLY	R/W	0x0	NAND Read Data Capture Delay Control 000 : NONE 001 : 1 bus clock delay 010 : 2 bus clock delay 011 : 3 bus clock delay 100 : 4 bus clock delay 101 : 5 bus clock delay 110 : 6 bus clock delay 111 : 7 bus clock delay

*) Read data capture timing is adjusted in burst data read. If RDNDLY is 0, at the end of the pulse width, read data is checked and captured.

If RDNDLY is N, the timing of checking and capturing read data is as below.

Available read data timing = end of pulse width + (Bus clock X N)

RDNDLY is useful for compensating read data cycle increase caused by I/O delay.



Field	Name	RW	Reset	Description
24	ARFW	R/W	0x0	Automatically Ready Flag Wait Enable 0 : Disable 1 : Enable

*) Once ARFW is set, in burst data read mode ready flag is automatically checked after Read Command. After this, the next procedure (burst data read) can be automatically executed. However, this is available only when there is 2nd read command.

For example : NAND Flash of SAMSUNG
1st READ_CMD (ex. 0x00) – ADDRESS –2nd READ_CMD(ex. 0x30)

In order to automatically check ready flag, NFC_RFWBASE Register value should be the same as 2nd Read Command value.

For example)
If 2nd READ_CMD (0x00000030) → set ARFWBASE(0x00000030)
If 2nd READ_CMD (0x30303030) → set ARFWBASE(0x30303030)
…

Field	Name	RW	Reset	Description
20	DACK	R/W	0x0	DMA Acknowledge Enable 0 : Disable 1 : Enable

*) In the case of burst transfer with CPU used, it should be set to ‘0’.

Field	Name	RW	Reset	Description
19	DMODE	R/W	0x0	DMA Mode Selection 0 : General DMA(GDMA) 1 : NFC Dedicate DMA(NDMA)

Field	Name	RW	Reset	Description
18	BMODE	R/W	0x1	Burst Operation Mode Selection for NDMA 0 : User data is only burst-transferred between NVS2310 and NAND Flash. 1 : User data + spare(ECC and etc.) is burst-transferred between NVS2310 and NAND Flash.

*) This function is only available for burst data transfer by NDMA.

The detail descriptions are the follwing.

1) If BMODE is '1'

- Burst Program :
In this mode, both user data and spare data are burst-written to the NAND flash.
1st STEP, Burst program is orderly executed in NAND as much as data size set in SPCFGn register. At this time, ECC data is generated whenever transmitting data of each subpage is completed. The generated ECC data is automatically saved to a designated address of SPCFGn[ECC_ADDR] in NFC dedicated memory.
2nd STEP, If transmitting all subpage data as many as subpages which have been set is completed, data as much as the data size which has been set in NFC_SPARE is read from NFC dedicated memory and transmitted to the NAND flash.
3rd STEP, If the transmission is completed, program interrupt or program end flag is occurred.

- Burst Read :
In this mode, both user data and spare data are burst-read from the NAND flash.
1st STEP, Data size as much as set in SPCFGn register is orderly burst-read from the NAND.
2nd STEP, If all subpage data read is completed, spare data as big as data size of NFC_SPARE is read from the NAND flash and saved to NFC dedicated memory.
3rd STEP, Once spare data read is completed, by referring to SP_CFGn register per subpage ECC decoding is executed. ECC data of each subpage is read from SP_CFGn[ECC_ADDR] of NFC dedicated memory for error detecting. If an error occurs, an error flag with a read flag or read interrupt occurs. If ECC error does not occur in all subpages, the error flag does not occur and only read flag or read interrupt occurs.
If an error flag occurs, A user can be possible to correct the error by referring to an error number and error address register. Once error correction is completed, the number of subpage where and error occurs must be checked by referring to MECC_STATUS[C_SUBPAGE] register. If it is not the last subpage, ECC decoding should be restarted by setting ECC Decoding Register(ECC_RSTART) to decode ecc for the rest of subpage.

2) If BMODE is '0'
1. Program : Not supportable.

2. Read :
1st STEP, Data size as much as set in SPCFGn register is orderly burst-read from the NAND.
Once all subpage data is read, a flag or read interrupt occurs. In order for ECC decoding, ECC data of each subpage should be single or burst-read from the spare area of the NAND Flash and after this ECC decoding should be manually executed per each subpage by using ECC Store Status & Load Control Register.

Field	Name	RW	Reset	Description
17	ECCON	R/W	0x0	NAND Flash ECC Encoding/Decoding Enable 0 : Disable 1 : Enable

If this bit is set, ECC encoding/decoding can be possible for burst transfer, single word data(WDATA) and multiple data.

Field	Name	RW	Reset	Description
16	PRSEL	R/W	0x0	Program/Read Mode Selection for Burst Transfer 0 : Read 1 : Program This register should be set befor burst transfer start.

Field	Name	RW	Reset	Description
15-14	DNUM	R/W	0x0	The Number of Multiple Data Configuration N : NFC MDATA Access conut

The number of transfer byte/half words of multiple bytes/half word data transfer (NFC_MDATA) is set.

For 8 bit bus width of NAND flash, transferred bytes = DNUM + 1

For 16 bit bus width of NAND flash, transferred half word = DNUM + 1 (must be DNUM<2)

If the 32 bit bus width of NAND flash is only used, ignore NFC_CRTL[DNUM].

Field	Name	RW	Reset	Description
11	RM	R/W	0x1	NAND Flash Ready Mask Disable 0 : READY input from NAND Flash is masked. If it is '0', READY flag or interrupt is not generated. 1 : READY input from NAND Flash is unmasked.

Field	Name	RW	Reset	Description
10	MSK	R/W	0x0	NAND Flash IO Mask Enable 0 : NAND Flash High Bytes(MSB 31bit downto 8bit) is not masked 1 : NAND Flash High Bytes(MSB 31bit downto 8bit) is masked

Field	Name	RW	Reset	Description
9-8	BW	R/W	0x0	NAND Flash Bus Width Select 00 : Bus width = 8bit 01 : Bus width = 16bit 10 : Bus width = 32bit 11 : Reserved.

Field	Name	RW	Reset	Description
7-6	BSIZE	R/W	0x0	NAND Flash Burst Size 00 : 1 burst 01 : 2 burst 10 : 4 burst 11 : 8 burst

*) This function is only available for burst transfer by GDMA. or CPU. A burst size must be the same as that of GDMA or that of CPU

Field	Name	RW	Reset	Description
5-3	PSIZE	R/W	0x0	Page Size of NAND Flash 000 : 1 page = 256 Half-Word 001 : 1 page = 512 Bytes 010 : 1 page = 1024 Half-Word 011 : 1 page = 2048 Bytes 100 : 1 page = 4096 Bytes 101~111 : 1 page = 8192 Bytes

A Linear address for external NAND is affected by CTRL[PSIZE]. If the single address mode is only used, ignore CRTL[PSIZE].

Field	Name	RW	Reset	Description
2-0	CADDR	R/W	0x0	The Number of Address Cycles N : The number of address cycle for NAND flash. (N+1) cycle is used for generating address cycle command

NFC DSIZE **0xB0050054**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_SIZE[15:00]															

Field	Name	RW	Reset	Description
15-0	DATA_SIZE	R/W	0x0000	NAND Flash Data Size N : N is transferred byte data size This is only available in burst transfer by GDMA or CPU. In the case of burst-transfer by NDMA, ignore NFC_DSIZE register. For Samsung 258 Half-word/512 Byte small block, N is 512(byte). For Samsung 1024 Half-Word/2048 Byte big block, N is 2048(byte).

NFC_SPARE **0xB0050058**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
														SPW	SPR
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				SPARE_DSIZE[11:00]											

Field	Name	RW	Reset	Description
17	SPW	R/W	0x0	0 : Read-only 1 : Spare Area Burst Program If this bit is set, the data of NFC dedicated memory is burst-transferred to NAND Flash as many as SPARE_DSIZE. This bit is automatically cleared and Spare Area Burst Write Flag(NFC_IRQ[SPWF]) occurs.
16	SPR	R/W	0x0	0 : Read-only 1 : Spare Area Burst Read If this bit is set, data as many as SPARE_DSIZE is burst-read from the NAND Flash and saved to NFC dedicated memory. Once this operation is completed, this bit is automatically cleared and Spare Area Burst Read Flag occure
11-0	SPARE_DSIZE	R/W	0x000	NAND Flash Spare Area Data Size N : The byte size of burst transfer data between NFC dedicated memory and NAND Flash It is available for burst transfer by NDMA(BMODE==1) or spare area burst program/read(by SPW, SPR).

NFC Burst Program/Read Start Register (NFC_BSTART) **0xB005005C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Don't care															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Don't care															

When this register is written by any value, data transfer is started. That is, Programming operation to NAND flash or Reading Operation from NAND is started as NFC_CTRL[PRSEL]

For burst data transfer by GDMA, you must set EN flag of DMA_CHCTRL register first, and then write any value to NFC_BSTART ahead of accessing NFC_LDATA

For Burst data transfer by NDMA, you must set EN flag of NDMA_CHCTRL register first, and then write any value to NFC_BSTART.

NFC_RESET 0xB0050060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Don't care															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Don't care															

If any value is written, NFC controller is internally reset.

NFC_IRQCFG 0xB0050064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDYIEN	PIEN	RDIEN		SPWIEN	SPRIEN			MEIEN							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Field	Name	RW	Reset	Description
31	RDYIEN	R/W	0x0	NAND Flash Ready Interrupt Enable 0 : Ready Interrupt Disable 1 : Ready Interrupt Enable
30	PIEN	R/W	0x0	NAND Flash Program Interrupt Enable 0 : Program Interrupt Disable 1 : Program Interrupt Enable Program interrupt occurs when all of data is programmed to NAND flash by burst transfer. The data size or burst-transfer is decided by NFC_DSIZE ane NFC_SPCFGn[DSIZE]
29	RDIEN	R/W	0x0	NAND Flash Read Interrupt Enable 0 : Read Interrupt Disable 1 : Read Interrupt Enable Read interrupt occurs when all of data is read from NAND flash by burst transfer. The data size or burst-transfer is decided by NFC_DSIZE or NFC_SPCFGn[DSIZE]
27	SPWIEN	R/W	0x0	NAND Flash Spare Area Burst Write Interrupt Enable 0 : Disable 1 : Enable
26	SPRIEN	R/W	0x0	NAND Flash Spare Area Burst Read Interrupt Enable 0 : Disable 1 : Enable
23	MEIEN	R/W	0x0	MLC ECC Decoding Interrupt Enable 0 : Disable 1 : Enable

NFC_IRQ 0xB0050068

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDYIRQ	PIRQ	RDIRQ		SPWIRQ	SPRIRQ			MEDIRQ							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDYFG	PFG	RDFG		SPWFG	SPRFG	Reserve	Reserve	MEDFG	MEEFG	SEDFG	SEEFG	ERRFG			

Field	Name	RW	Reset	Description
31	RDYIRQ	R/W	0x0	NAND Flash Ready End Interrupt 0 (Read) : NONE 1 (Read) : Ready interrupt occurred 1 (Write) : Ready Interrupt clear
30	PIRQ	R/W	0x0	NAND Flash Program End Interrupt 0 (Read) : NONE 1 (Read) : Program interrupt occurred 1 (Write) : Program Interrupt clear

29	RDIRQ	R/W	0x0	NAND Flash Read End Interrupt 0 (Read) : NONE 1 (Read) : Read interrupt occurred 1 (Write) : Read Interrupt clear
27	SPWIRQ	R/W	0x0	NAND Flash Spare Burst Write End Interrupt 0 (Read) : NONE 1 (Read) : Spare burst write interrupt occurred 1 (Write) : Spare burst write Interrupt clear
26	SPRIRQ	R/W	0x0	NAND Flash Spare Burst Read End Interrupt 0 (Read) : NONE 1 (Read) : Spare burst read interrupt occurred 1 (Write) : Spare burst read Interrupt clear
23	MEDIRQ	R/W	0x0	MLC ECC Decoding End Interrupt 0 (Read) : NONE 1 (Read): MLC ECC decoding interrupt occurred 1 (Write) : MLC ECC decoding Interrupt clear
15	RDYFG	R/W	0x0	NAND Flash Ready End Flag 0 (Read) : NONE 1 (Read) : Ready flag occurred 1 (Write) : Ready flag clear
14	PFG	R/W	0x0	NAND Flash Program End Flag 0 (Read) : NONE 1 (Read) : Program flag occurred 1 (Write) : Program flag clear
13	RDFG	R/W	0x0	NAND Flash Read End Flag 0 (Read) : NONE 1 (Read) : Read flag occurred 1 (Write) : Read flag clear
11	SPWFG	R/W	0x0	NAND Flash Spare Burst Write End Flag 0 (Read) : NONE 1 (Read) : Spare burst write flag occurred 1 (Write) : Spare burst write flag clear
10	SPRFG	R/W	0x0	NAND Flash Spare Burst Read End Flag 0 (Read) : NONE 1 (Read) : Spare burst read flag occurred 1 (Write) : Spare burst read flag clear
7	MEDFG	R/W	0x0	MLC ECC Decoding End Flag 0 (Read) : NONE 1 (Read) : MLC ECC decoding flag occurred 1 (Write) : MLC ECC decoding flag clear
6	MEEFG	R/W	0x0	MLC ECC Encoding End Flag 0 (Read) : NONE 1 (Read) : MLC ECC encoding flag occurred 1 (Write) : MLC ECC encoding flag clear
5	SEDFG	R/W	0x0	SLC ECC Decoding End Flag 0 (Read) : NONE 1 (Read) : SLC ECC decoding flag occurred 1 (Write) : SLC ECC decoding flag clear
4	SEEFG	R/W	0x0	SLC ECC Encoding End Flag 0 (Read) : NONE 1 (Read) : SLC ECC encoding flag occurred 1 (Write) : SLC ECC encoding flag clear
3	ERRFG	R/W	0x0	MLC/SLC Error Flag 0 (Read) : NONE 1 (Read) : Error flag occurred 1 (Write) : Error flag clear

NFC_STATUS 0xB005006C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDY	FS										CURR_SUBPAGE[4:0]				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
										NFC_STATE					

Field	Name	RW	Reset	Description
31	RDY	R	0x0	NAND Flash Ready Flag Status 0 : External ready flag of dedicated NAND Flash is low 1 : External ready flag of dedicated NAND Flash is high
30	FS	R	0x0	NAND Flash FIFO Status 0 : FIFO is not ready to write and read for burst transfer 1 : FIFO is ready to write and read for burst transfer For burst data transfer by ARM, user must check that this bit is high, ahead of access NFC_LDATA.
20-16	CURR_SUBPAGE	R	0x00	Current Sub-Page Indicator N : Nth Sub-page data is transferred from/to NAND Flash
5-0	NFC_STATE	R	0x00	NFC FSM Current State N : Reserved

NFC_RFWBASE 0xB0050070

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RFWBASE[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RFWBASE[15:00]															

*) refer to NFC_CTRL[RFWEN]

NFC DMA Source/Destination Address (NDMA_ADDR) 0xB0050100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ST_ADDR[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ST_ADDR[15:0]															

This register contains the start address of source memory block for NFC program operation or the start address of destination memory block for NFC Read operation. This register field is only available in NDMA mode enable of NFC_CTRL

NFC DMA Parameter Register (NDMA_PARAM) 0xB0050104

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MASK[23:8]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK[7:0]								INC[7:0]							

Field	Name	RW	Reset	Description
31-8	MASK	R/W	0x0	Source/Destination Address mask Register 0 : non-masked 1 : Masked so that source address bit doesn't be changed during DMA transfer

Each bit field controls the dedicated bit of source/destination address field. That is, if MASK[23] is set to 1, the 28th bit of source/destination address is masked, and if MASK[22] is set to 1, the 27th bit of source/destination address is masked, and so on. If a bit is masked, a corresponding bit of address bus is not changed during DMA transfer. This function can be used to generate circular buffer address.

Field	Name	RW	Reset	Description
7-0	INC	R/W	0x0	Source/Destination address is added by amount of sinc at every write cycles. sinc is represented as 2's complement, so if INC[7] is 1, the source address is decremented

The addresses of DMA transfer are 32bit wide, but the upper 4bit of them are not affected during DMA transfer. If the source or destination address reaches its maximum address space like 0x7FFFFFFF or 0x2FFFFFFF, the next transfer is starting from 0x70000000 or 0x20000000 not from 0x80000000 or 0x30000000.

NFC DMA Current Address (NDMA_CADDR) 0xB0050108

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
C_ADDR[15:0]															

This register contains the current source/destination address of DMA transfer. It represents that the current transfer read data from this address. This is read only register.

NFC DMA Control Register(NDMA_CTRL) 0xB005010C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Reserve		LOCK	ARB			BSIZE		WSIZE		FLAG		SEQ	EN

Field	Name	RW	Reset	Description
11	LOCK	R/W	0x0	Issue Locked Transfer 0 : Non-locked transfer mode 1 : NDMA transfer executed with lock transfer.

Lock field controls the LOCK signal (refer to AHB specification). When the LOCK is set to 1, the DMA transfer is not bothered by other AHB masters like LCD controller, ARM etc. This field is only meaningful in case of non-burst type transfers.

Field	Name	RW	Reset	Description
10	ARB	R/W	0x0	Arbitration Transfer 0 : NDMA transfer executed with arbitration 1 : NDMA transfer executed with no arbitration..

Arbitration means that at the end of every HOP transfer, the AHB bus is released from DMA channel so other master can occupy the bus when that master has requested the bus. Burst means that once the DMA request occurs, all of transfers are executed without further DMA requests.

Field	Name	RW	Reset	Description
7-6	BSIZE	R/W	0x0	Busrt Size 00 : 1 burst consists of 1 read or write cycle 01 : 1 burst consists of 2 read or write cycle 10 : 1 burst consists of 4 read or write cycle 11 : 1 burst consists of 8 read or write cycle
5-4	WSIZE	R/W	0x0	Word Size 00 : Read or write 8bit data for each cycle 01 : Read or write 16bit data for each cycle 10,11 : Read or write 32bit data for each cycle
3	FLAG	R/W	0x0	DMA Done Flag Represents that all hop of transfers are fulfilled. When writing 1 to this bit, it is cleared to 0
2	SEQ	R/W	0x0	Sequential mode control If the number of subpage is equal and above ‘2’, this bit must be set to ‘1’ 0 : After all of hop transfer has executed, the DMA channel is disabled 1 : The DMA channel remains enabled. When another DMA request has occurred, the DMA channel start transfer data again with the same manner (address, increment, mask) as the latest transfer of that channel.
1	EN	R/W	0x0	NFC DMA Enable 0 : NDMA Channel is disabled 1 : NDMA Channel enable, Before NDMA transfer start by NFC_CTRL[BSTART] , this bit must be set to ‘1’

NFC DMA SUBPAGE Control(NDMA_SPCTRL) 0xB0050110

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
											SUBPAGE CNT[04:00]				

Field	Name	RW	Reset	Description
4-2	SUBPAGE_CNT	R/W	0x0	Set the Number of Subpage N : The number of subpage

NFC DMA STATUS(NDMA_STATUS) 0xB0050114

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		C_HOP_COUNT[13:00]													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										C_STATE[05:00]					

Field	Name	RW	Reset	Description
29-16	C_HOP_COUNT	R	0x0000	Current Hop Count
5-0	C_STATE	R	0x00	Current NDMA State

NFC SUB-PAGE Configuration Register (SP_CFGn, n=0,…,31) 0xB0050120+n*4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					ECC_ADDR[10:00]										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSIZE[15:00]															

Field	Name	RW	Reset	Description
26-16	ECC_ADDR	R/W	0x0	ECC offset address in NFC dedicate memory

It represents the offset address of ECC data in NFC dedicated memory. n(n=0,…,31) is the number of subpage.
In the case of burst program by NDMA, after ECC encoding of each subpage is completed. the generated ECC data is automatically saved to the NFC dedicated memory where is ECC_ADDR.
In the case of burst read by NDMA, If NFC_CTRL[BMODE] equal '1', the controller reads ECC data from the address and ecc deconding for each subpage is automatically processed after all of the data is transferred from NAND Flash to the destination memory(payload data) and NFC dedicated memory(ECC or information data as much as NFC_SPARE[SPARE_DSIZE])

Field	Name	RW	Reset	Description
15*0	DSIZE	R/W	0x0	The number of data for each subpage[n]

Note) In the case of burst transfer by NDMA, it represents ECC encoding / decoding data size and transferred data size from/to NAND Flash. Its size must set.

NFC ECC Control Register(ECC_CTRL) 0xB0050200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					ECC_DSIZE[26:16]										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved										CP	MODE	ECC_EN[03:00]			

Field	Name	RW	Reset	Description
26-16	ECC_DSIZE	R/W	0x000	N : ECC data size It is only useful for the following case 1) NAND Flash burst transfer by GDMA & CPU and ECC_CTRL[MODE] = 0 2) ECC_CTRL[MODE] = 1
5	CP	R/W	0x0	MLC ECC Correction Pass Enable 0 : Correction pass disable 1 : Correction pass enable
4	MODE	R/W	0x0	ECC Mode Selection 0 : ECC is calculated for I/O data transferred to NAND Flash. 1 : ECC is calculated for internal data written to NFC dedicated memory or read from NFC dedicated memory * To access NFC memory by CPU, MODE should be '1'
3-0	ECC_EN	R/W	0x0	ECC Enable Configuration 00xx : Disable 0100 : SLC ECC encoding enable 0101 : SLC ECC decoding enable 0110 : MLC ECC4 encoding enable 0111 : MLC ECC4 decoding enable 1000 : MLC ECC6 encoding enable 1001 : MLC ECC6 decoding enable 1010 : MLC ECC12 encoding enable 1011 : MLC ECC12 decoding enable 1100 : MLC ECC16 encoding enable 1101 : MLC ECC16 decoding enable 1110 : MLC ECC24 encoding enable 1111 : MLC ECC24 decoding enable

NFC SECC Error Number Register (SECC_STATUS) 0xB0050204

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENUM15		ENUM14		ENUM13		ENUM12		ENUM11		ENUM10		ENUM09		ENUM08	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENUM07		ENUM06		ENUM05		ENUM04		ENUM03		ENUM02		ENUM01		ENUM00	

(n = 0,…, 15)

Field	Name	RW	Reset	Description
[2*n+1:2*n]	ENUM[n]	R/W	0x0	SLC ECC Error Status 00 : No error 01 : 2-bit ECC error is detect(incorrectable) 10 : 1-bit ECC error is detect(correctable) 11 : Correctable impossible error

NFC MLC ECC Error Status Register(MECC_STATUS) 0xB0050204

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
											C_SUBPAGE				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
											ERRNUM[04:00]				

Table 12.3 Error Status Table for MLC ECC4/6/12/16/24 bit

ERRNUM [4:0]	MLC ECC4 DECODE	MLC ECC6 DECODE	MLC ECC12 DECODE	MLC ECC16 DECODE	MLC ECC24 DECODE
00000	No Error	No Error	No Error	No Error	No Error
00001	1 BIT Error	1 BIT Error	1 BIT Error	1 BIT Error	1 BIT Error
00010	2 BIT Error	2 BIT Error	2 BIT Error	2 BIT Error	2 BIT Error
00011	3 BIT Error	3 BIT Error	3 BIT Error	3 BIT Error	3 BIT Error
00100	4 BIT Error	4 BIT Error	4 BIT Error	4 BIT Error	4 BIT Error
00101	-unknown	5 BIT Error	5 BIT Error	5 BIT Error	5 BIT Error
00110		6 BIT Error	6 BIT Error	6 BIT Error	6 BIT Error
00111		-unknown	7 BIT Error	7 BIT Error	7 BIT Error
01000			8 BIT Error	8 BIT Error	8 BIT Error
01001			9 BIT Error	9 BIT Error	9 BIT Error
01010			10 BIT Error	10 BIT Error	10 BIT Error
01011			11 BIT Error	11 BIT Error	11 BIT Error
01100			12 BIT Error	12 BIT Error	12 BIT Error
01101			-unknown	13 BIT Error	13 BIT Error
01110				14 BIT Error	14 BIT Error
01111				15 BIT Error	15 BIT Error
10000				16 BIT Error	16 BIT Error
10001				-unknown	17 BIT Error
10010					18 BIT Error
10011					19 BIT Error
10100					20 BIT Error
10101					21 BIT Error
10110					22 BIT Error
10111					23 BIT Error
11000					24 BIT Error
11001~					unknown
11111	correction impossible error				

NFC ECC Clear Register (ECC_CLEAR) 0xB0050210

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Don't care															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Don't care															

If any bit is written, some of internal logic is cleared. For ECC encoding or decoding, This register must be set before and after ECC_CTRL[ECC_EN] is set.

NFC ECC Decoding Restart (ECC_RSTART) 0xB0050214

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Don't care															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Don't care															

This register is used to restart ecc decoding procedure for the next subpage. In the case of burst read by NDMA, ECC decoding of each subpage is subsequently processed by the controller. when ecc error occurs during ecc decoding for some of subpages. the controller generates an error flag and holds the ecc decoding procedure for the rest of subpages and ECC Current State is
. To start ecc [decoding] decoding for the next subpages, this register should be written with any value.

NFC ECC Store Status & Load Control Register (ECC_STLDCTL) 0xB0050218

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STLD31	STLD30	STLD29	STLD28	STLD27	STLD26	STLD25	STLD24	STLD23	STLD22	STLD21	STLD20	STLD19	STLD18	STLD17	STLD16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STLD15	STLD14	STLD13	STLD12	STLD11	STLD10	STLD9	STLD8	STLD7	STLD6	STLD5	STLD4	STLD3	STLD2	STLD1	STLD0

Field	Name	RW	Reset	Description
31-0	STLD0~ STLD31	R/W	0x00000000	ECC Store Status and Load Enable Control Each bit filed is synchronized with subpage N. 0 (read) : o/w 1 (read) : ecc of subpage N is stored in internal store register. 1 (write) : ecc of subpage N is loaded in internal LFSR(Linear Feedback Shift Register) for ecc decoding of the subpage when NFC_CTRL[BMODE] is '0. Once any of bit filed which value is '1' is written to '1', it is cleared(1->0). If all of the bit field is cleared(32'b0), there is no subpage can be decoding.

NFC ECC Current State Register (ECC_STATUS) 0xB005021C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		ECC_STATE[29:16]													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ECC_STATE[15:00]															

Field	Name	RW	Reset	Description
29-0	ECC_STATE	R/W	0x00000001	0x00000001:ECC_IDLE. 0x00000002:SLC ECC data wait 0x00000004:SLC ECC encode or decode 0x00000008:SLC ECC End 0x00000010:MLC ECC encode data wait 0x00000020:MLE ECC decode data wait 0x00000040:MLC ECC encode 0x00000080:MLC ECC decode 0x00000100:MLC ECC4 encoded ecc input wait 0x00000200:MLC ECC4 decoded ecc input wait 0x00000400:MLC ECC4 error calculation is on process. 0x00000800:MLC ECC6 encoded ecc input wait 0x00001000:MLC ECC6 decoded ecc input wait 0x00002000:MLC ECC6 error calculation is on process. 0x00004000:MLC ECC12 encoded ecc input wait 0x00008000:MLC ECC12 decoded ecc input wait 0x00010000:MLC ECC12 error calculation is on process. 0x00020000:MLC ECC16 encoded ecc input wait 0x00040000:MLC ECC16 decoded ecc input wait 0x00080000:MLC ECC16 error calculation is on process. 0x00020000:MLC ECC24 encoded ecc input wait 0x00040000:MLC ECC24 decoded ecc input wait 0x00400000:MLC ECC24 error calculation is

				on process. 0x08000000:MLC ECC Erorr Occure in NDMA mode 0x08000000:MLC ECC Erorr and Holt State (MECC_CORR) 0x20000000:MLC ECC next subpage decode wait (MECC_PWAIT) o/w : reserved If burst read by NDMA and BMODE=0, should check if ECC_STATE is the followings at the end of every subpage decoding because MECC decoding end flag do not ocuure. 1) ECC_IDLE : represent all of the subpage decoding completed and last subpage decoding has no error. Controller is ready. 2) MECC_CORR : Error occure in current subpage decoding. 3) MECC_PWAIT : there is no error in cuurent subpage decoding and controller is ready for next subpage decoding.

MLC ECC4 Code Register 0(MECC4_0) **0xB0050220**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC4_CODE1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC4_CODE0[15:00]															

MLC ECC4 Code Register 1(MECC4_1) **0xB0050224**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								MECC4_CODE4[56:48]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC4_CODE3[47:32]															

Note: Real ECC Code Size for MLC ECC4 mode is 52 bits, For the remainder bits of MLC ECC4 Code Register 1, zeros are padded.
Note: MLC ECC4 Code Register must be written and read by word access.

MLC ECC6 Code Register 0(MECC6_0) **0xB0050220**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC6_CODE1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC6_CODE0[15:00]															

MLC ECC6 Code Register 1(MECC6_1) **0xB0050224**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC6_CODE3[63:48]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC6_CODE2[47:32]															

MLC ECC6 Code Register 2(MECC6_2) **0xB0050228**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												MECC6_CODE5[83:80]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC6_CODE4[79:64]															

Note: Real ECC Code Size for MLC ECC8 mode is 104 bits, For the remainder bits of MLC ECC6 Code Register 3, zeros are padded.
Noe: MLC ECC8 Code Register must be written and read by word access.

MLC ECC12 Code Register 0(MECC12_0) 0xB0050220

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC12_CODE1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC12_CODE0[15:00]															

MLC ECC12 Code Register 1(MECC12_1) 0xB0050224

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC12_CODE3[63:48]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC12_CODE2[47:32]															

MLC ECC12 Code Register 2(MECC12_2) 0xB0050228

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC12_CODE5[95:80]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC12_CODE4[79:64]															

MLC ECC12 Code Register 3(MECC12_3) 0xB005022C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC12_CODE7[127:112]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC12_CODE6[111:96]															

MLC ECC12 Code Register 4(MECC12_4) 0xB0050230

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC12_CODE9[159:144]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC12_CODE8[143:128]															

MLC ECC12 Code Register 4(MECC12_5) 0xB0050234

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								MECC12_CODE10[168:160]							

Note: Real ECC Code Size for MLC ECC12 mode is 156 bits, For the remainder bits of MLC ECC12 Code Register 4, zeros are padded.
Note: MLC ECC12 Code Register must be written and read by word access.

MLC ECC16 Code Register 0(MECC16_0) 0xB0050220

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC16_CODE1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC16_CODE0[15:00]															

MLC ECC16 Code Register 1(MECC16_1) 0xB0050224

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC16_CODE3[63:48]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC16_CODE2[47:32]															

MLC ECC16 Code Register 2(MECC16_2) **0xB0050228**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC16_CODE5[95:80]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MECC16_CODE4[79:64]															

MLC ECC16 Code Register 3(MECC16_3) **0xB005022C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC16_CODE7[127:112]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC16_CODE6[111:96]															

MLC ECC16 Code Register 4(MECC16_4) **0xB0050230**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC16_CODE9[159:144]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC16_CODE8[143:128]															

MLC ECC16 Code Register 5(MECC16_5) **0xB0050234**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC16_CODE11[191:176]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC16_CODE10[175:160]															

MLC ECC16 Code Register 6(MECC16_6) **0xB0050238**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC16_CODE13[223:208]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC16_CODE12[207:192]															

Note: Real ECC Code Size for MLC ECC16 mode is 207 bits, For the remainder bits of MLC ECC16 Code Register 6, zeros are padded.
Note: MLC ECC16 Code Register must be written and read by word access.

MLC ECC24 Code Register 0(MECC24_0) **0xB0050220**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC24_CODE1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE0[15:00]															

MLC ECC24 Code Register 1(MECC24_1) **0xB0050224**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC24_CODE3[63:48]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE2[47:32]															

MLC ECC24 Code Register 2(MECC24_2) **0xB0050228**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC24_CODE5[95:80]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE4[79:64]															

MLC ECC24 Code Register 3(MECC24_3) **0xB005022C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC24_CODE7[127:112]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE6[111:96]															

MLC ECC24 Code Register 4(MECC24_4) **0xB0050230**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC24_CODE9[159:144]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE8[143:128]															

MLC ECC24 Code Register 5(MECC24_5) **0xB0050234**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC24_CODE11[191:176]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE10[175:160]															

MLC ECC24 Code Register 6(MECC24_6) **0xB0050238**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC24_CODE13[223:208]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE12[207:192]															

MLC ECC24 Code Register 7(MECC24_7) **0xB005023C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC24_CODE15[255:240]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE14[239:224]															

MLC ECC24 Code Register 8(MECC24_8) **0xB0050240**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC24_CODE17[287:272]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE16[271:256]															

MLC ECC24 Code Register 9(MECC24_9) **0xB0050244**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MECC24_CODE19[319:304]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE18[303:288]															

MLC ECC24 Code Register 10(MECC24_10) **0xB0050248**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MECC24_CODE20[335:320]															

SLC ECC Error Address (SECC_EADDRx, x = 0,…,15) **0xB0050260+n*4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SECC_EADDRx[14:0]														

Note:To correct errors for SLC ECC
- SECC_EADDR[14:3] : Byte Error Address
- SECC_EADDR[2:0] : Error Bit Plane

Corrected DATA = ADDR[SECC_EADDR[14:3]] ^ (1 << SECC_EADDR[2:0])
(Byte) (Byte)

MLC ECC4 Error Address Register (MECC4_EADDRn, n = 0,1,2,3) **0xB0050260+n*4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		EADDR[13:00]													

Field	Name	RW	Reset	Description
13-0	EADDR	R/W	0x0000	N is bit address of ECC ERROR

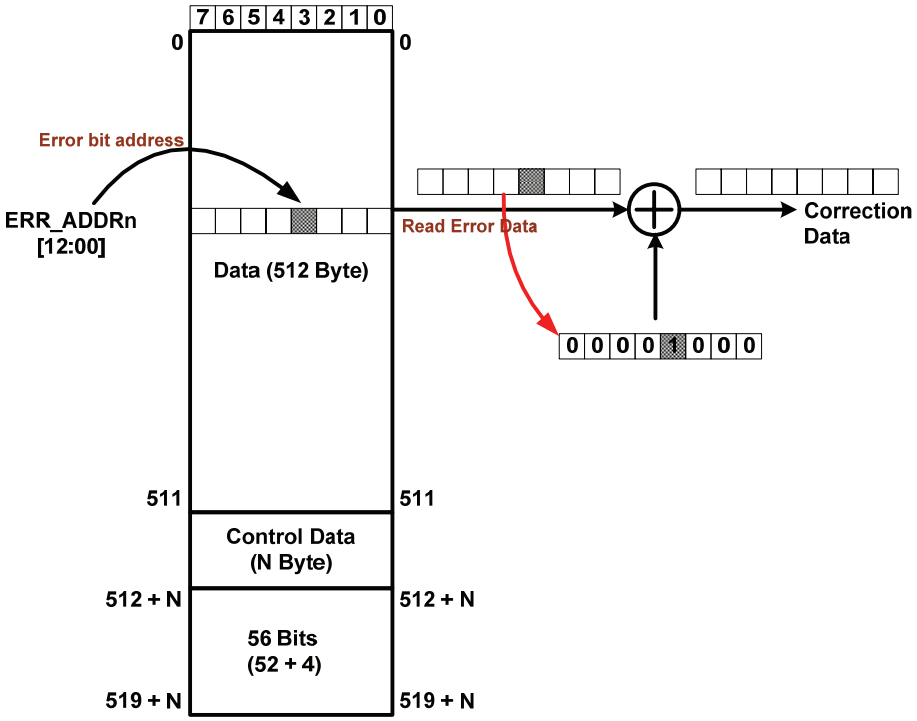


Figure 12.10 Example of MLC ECC4 error correction

MLC ECC6 Error Address Register (MECC8_EADDRn, n = 0,…,5) **0xB0050260+n*4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		EADDR[13:00]													

Note: Refer to MECC4_EADDRn

MLC ECC12 Error Address Register (MECC12_EADDRn, n = 0,…,11) **0xB0050260+n*4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		EADDR[13:00]													

Note: Refer to MECC4_EADDRn

MLC ECC16 Error Address Register (MECC16_EADDRn, n = 0,…,15) **0xB0050260+n*4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		EADDR[13:00]													

Note: Refer to MECC4_EADDRn

MLC ECC24 Error Address Register (MECC24_EADDRn, n = 0,…,23) **0xB0050260+ n*4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		EADDR[13:00]													

Note: Refer to MECC4_EADDRn

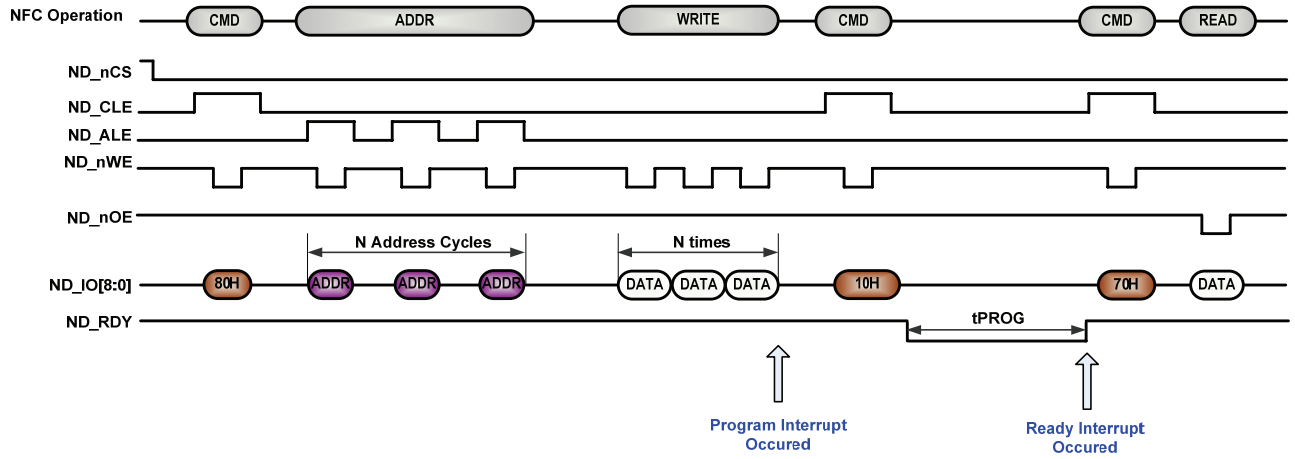


Figure 12.11 Timing Diagram of NAND Page Program

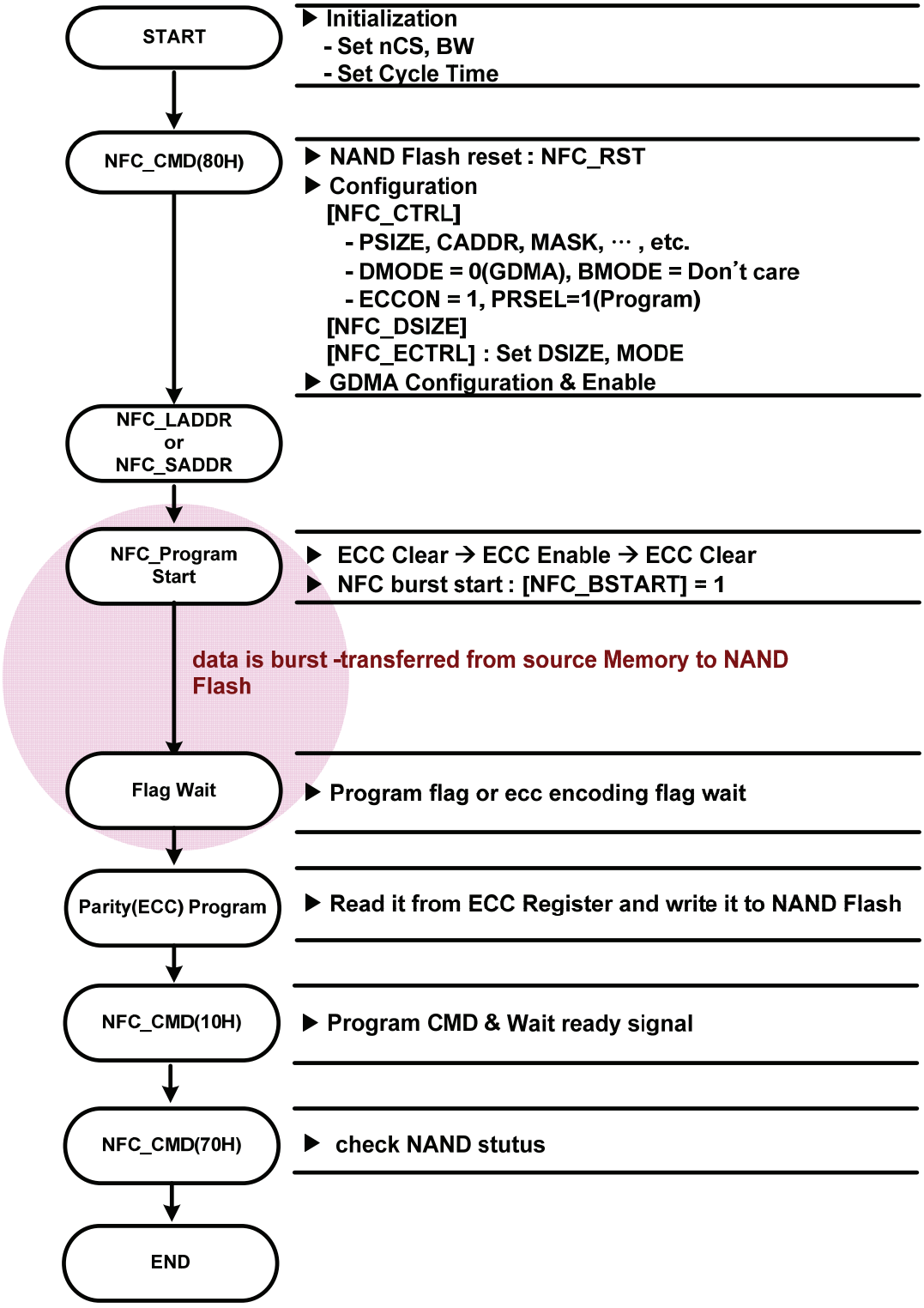


Figure 12.12 NAND Burst Program Sequence by GDMA

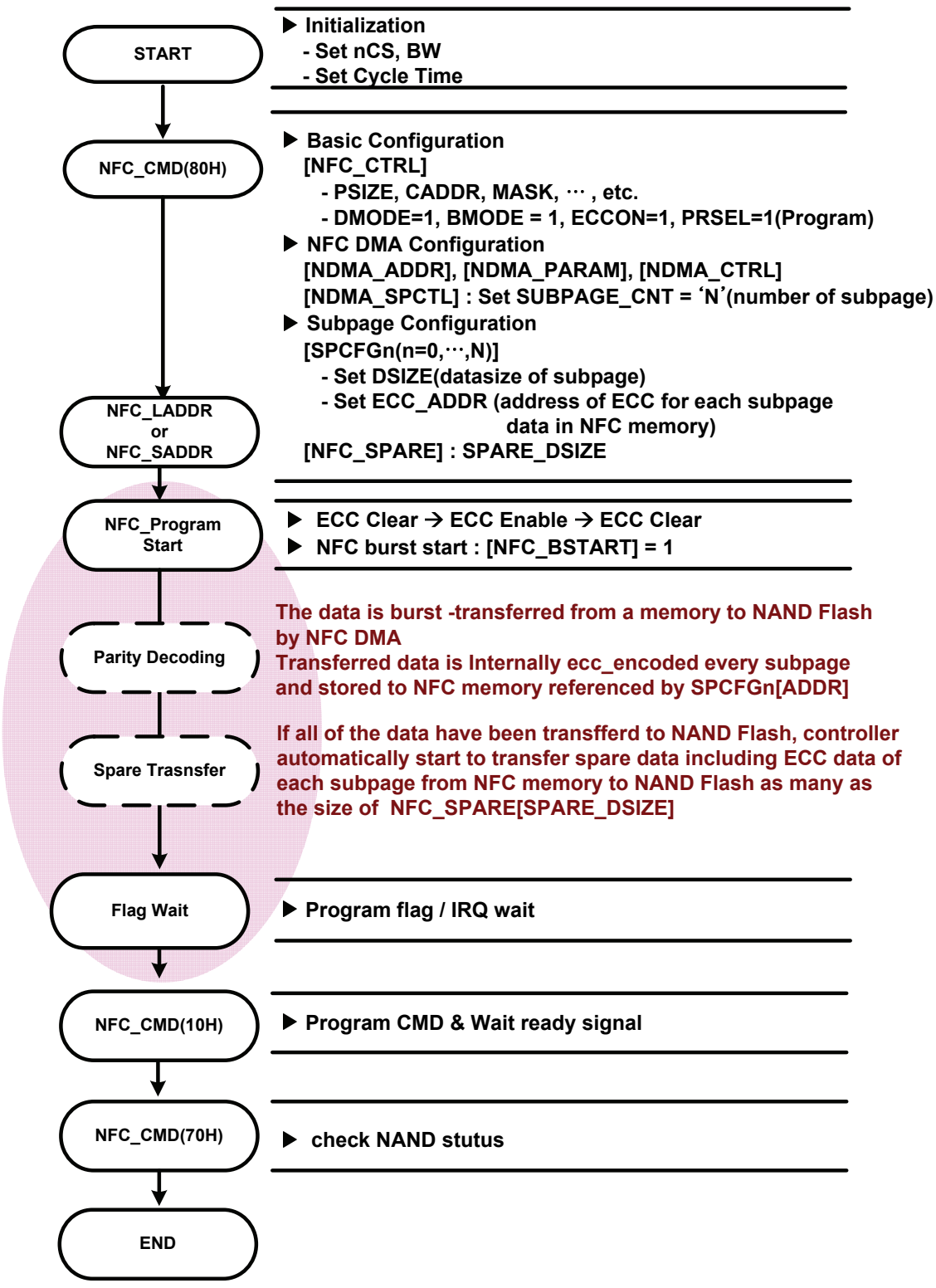


Figure 12.13 NAND Burst Program Sequence by NDMA

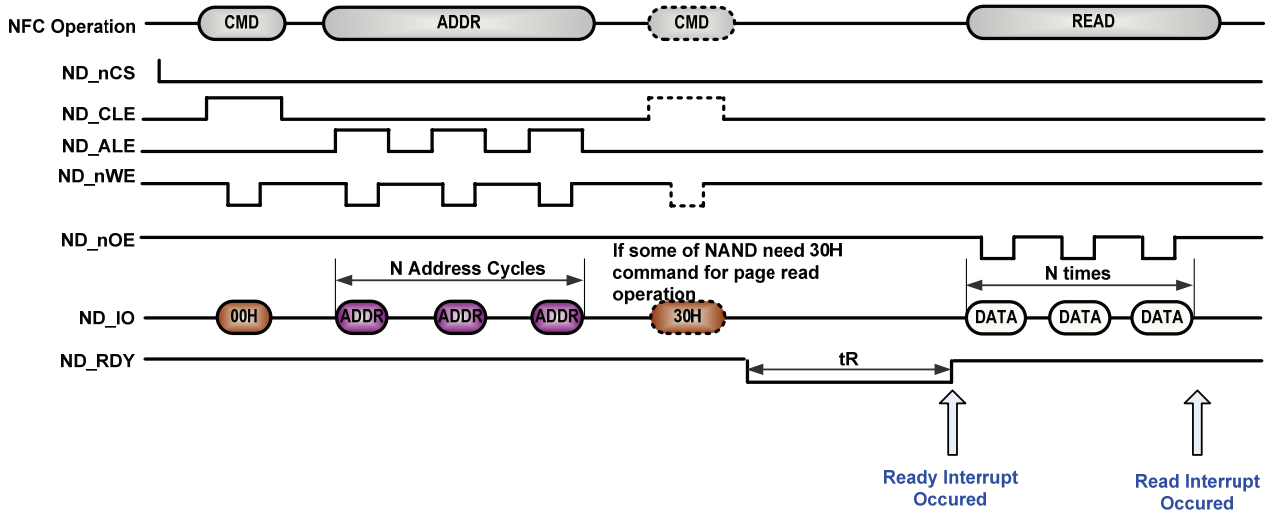


Figure 12.14 Timing Diagram of NAND Page Read

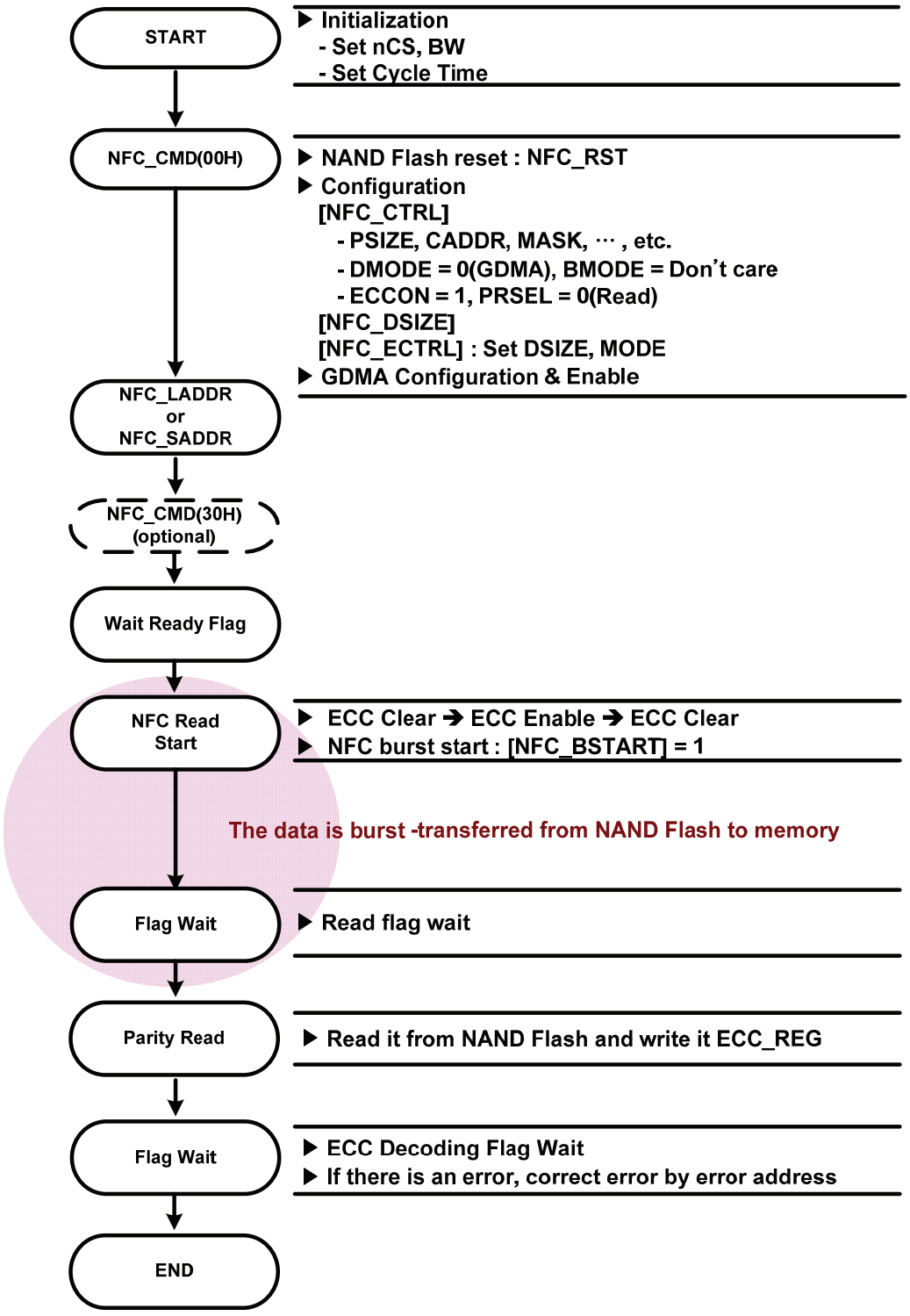


Figure 12.15 NAND Burst Read Sequence by GDMA

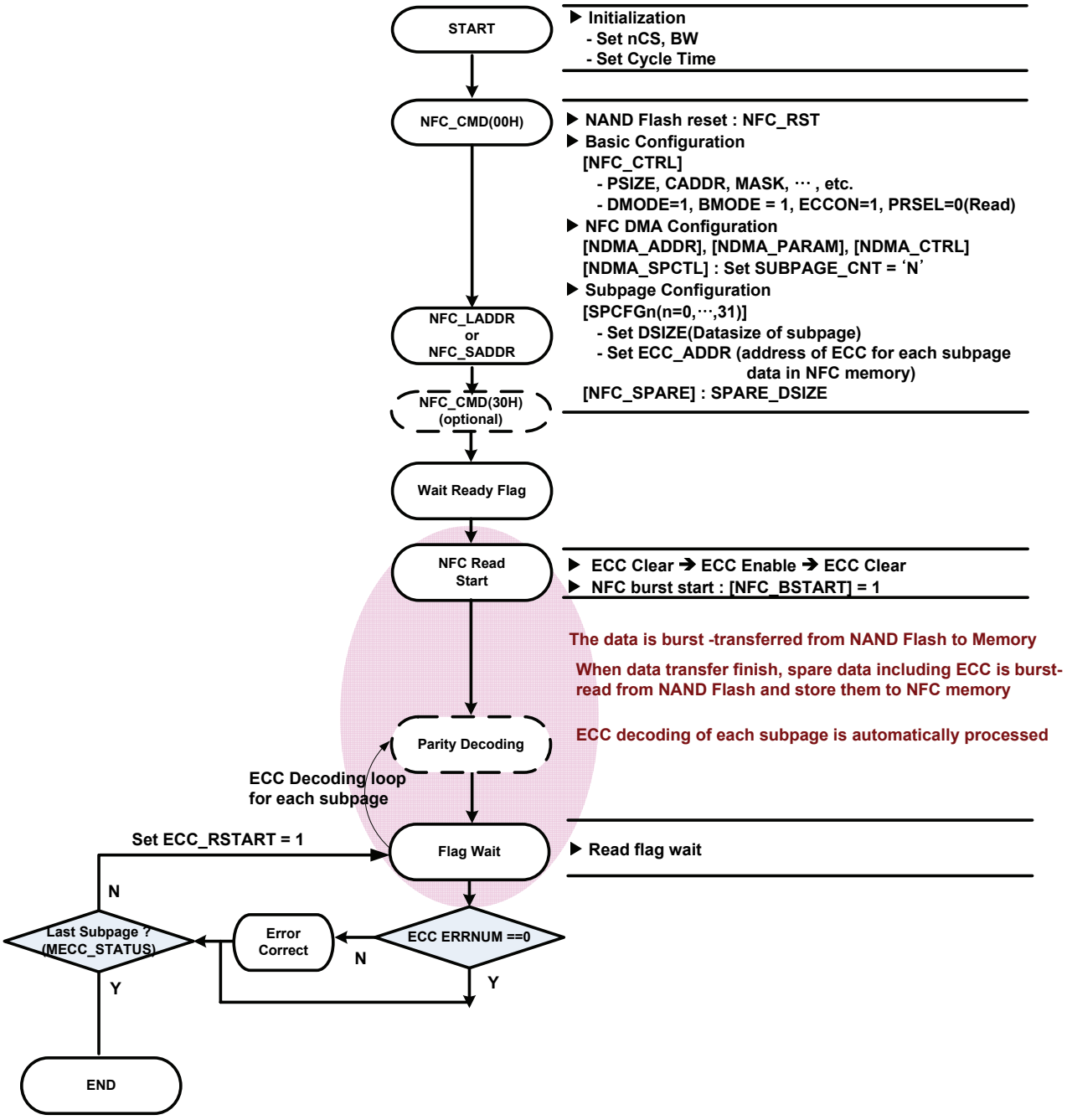


Figure 12.16 NAND Burst Read Sequence by NDMA(BMODE=1)

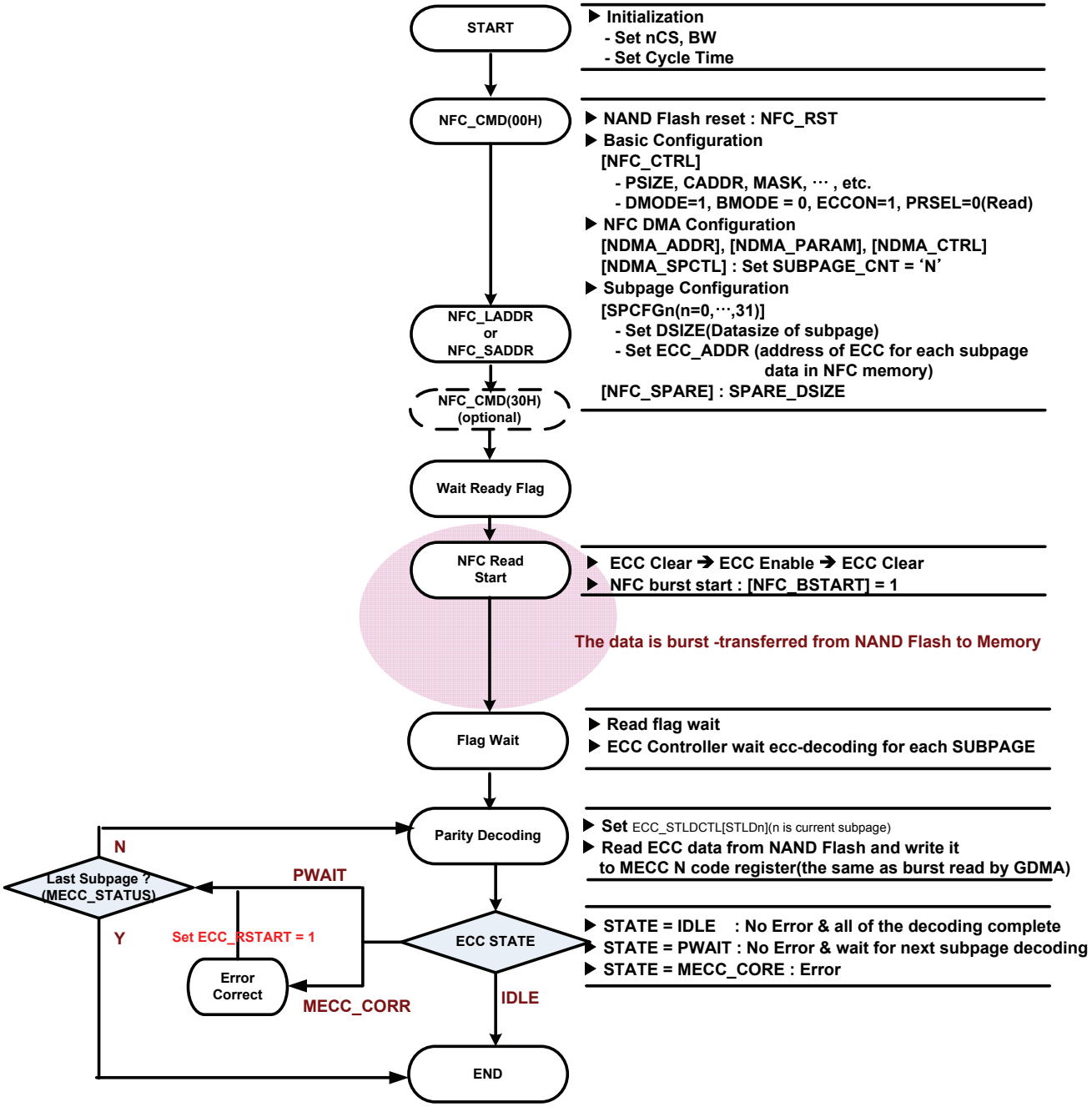


Figure 12.17 NAND Burst Read Sequence by NDMA(BMODE=0)

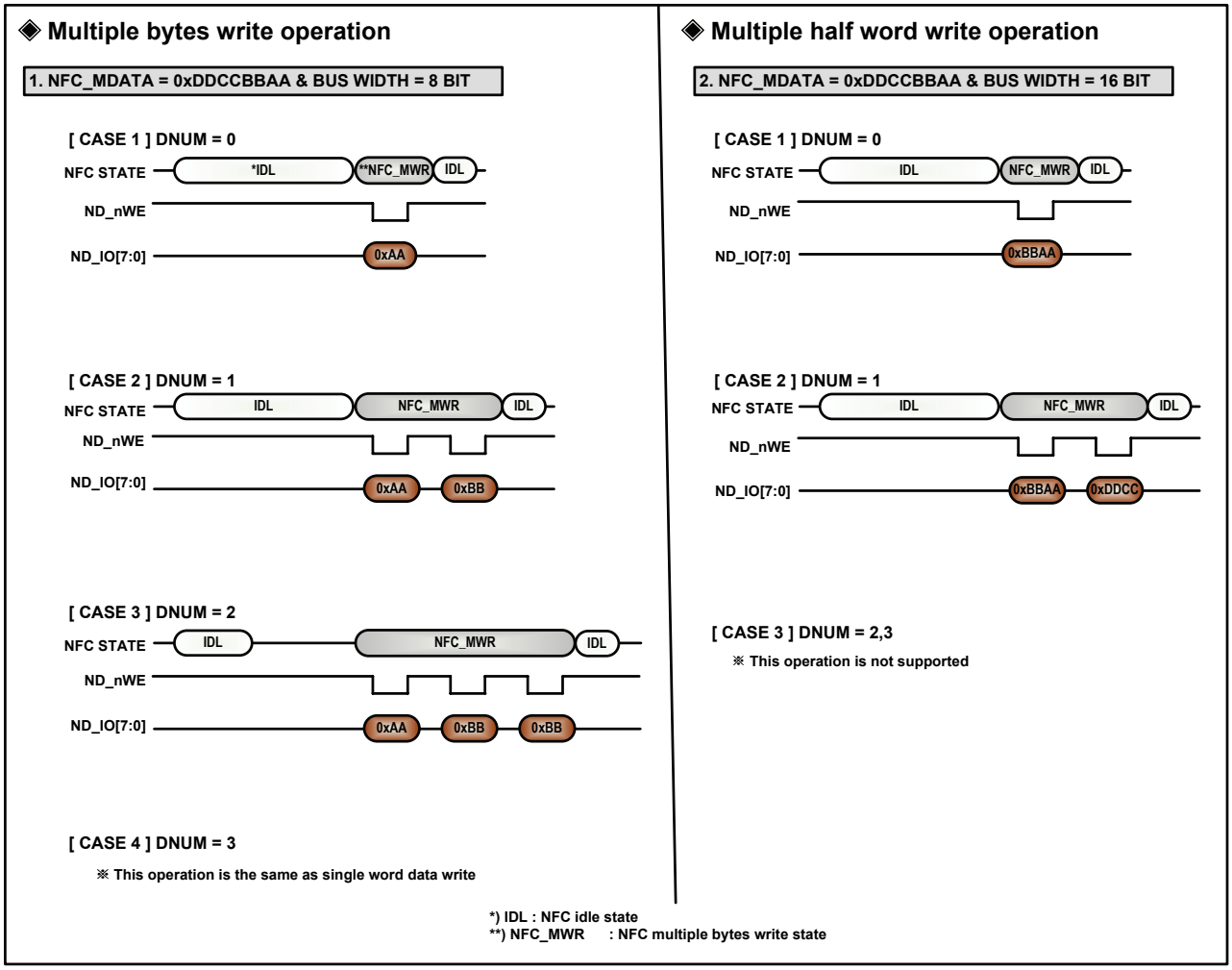


Figure 12.18 Example of Multiple bytes/half Word Write Operation

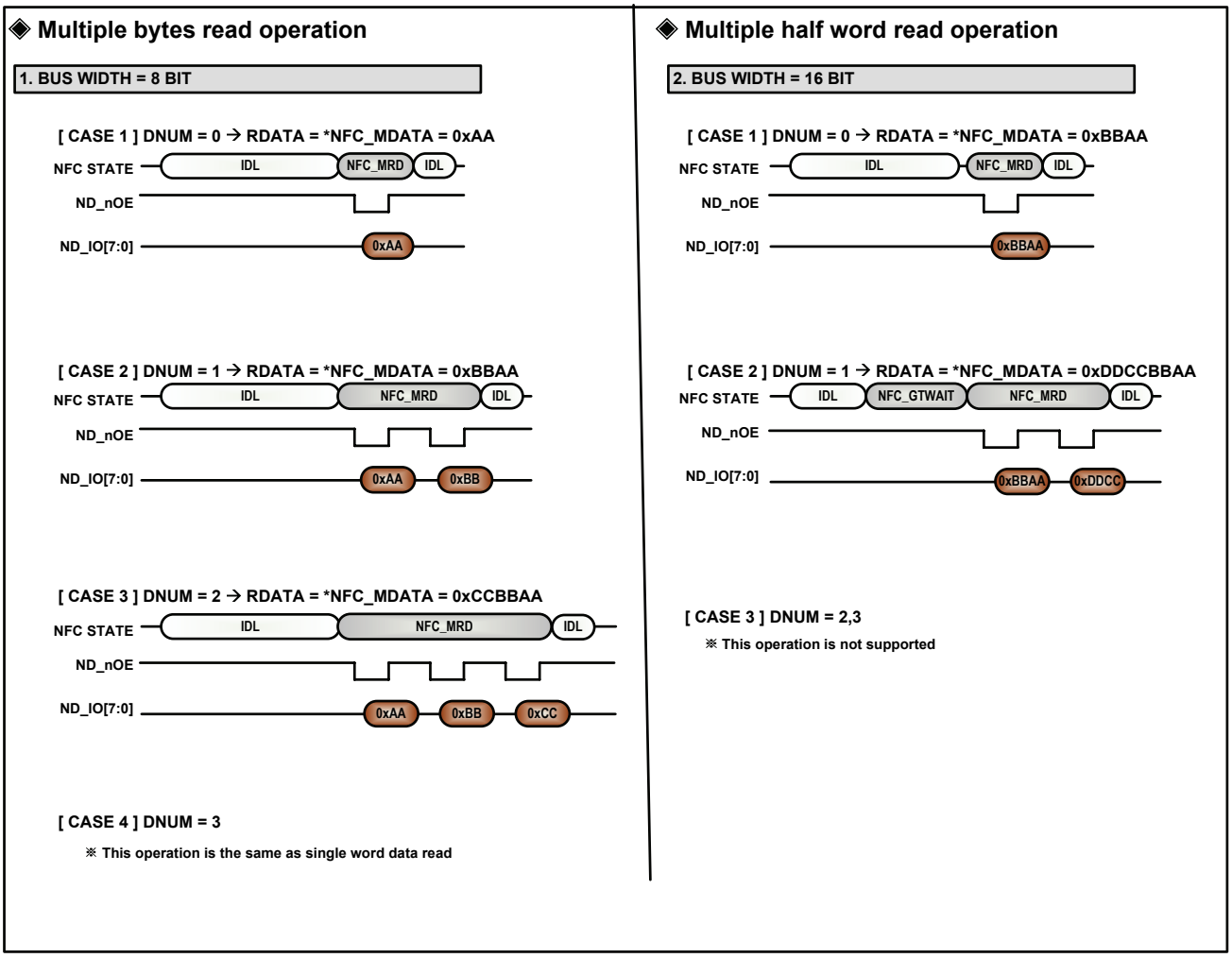


Figure 12.19 Example of Multiple bytes/half Word Read Operation

13 USB 2.0 OTG Controller

13.1 Overview

The NVS2310 supports Dual-Role Device (DRD) controller, which supports both device and host functions and is fully compliant with the On-The-Go Supplement to the USB 2.0 Specification, Revision 1.0a.

It can also be configured as a Host-only or Device-only controller, fully compliant with the USB 2.0 Specification. The USB 2.0 configurations support high-speed (HS, 480-Mbps), full-speed (FS, 12-Mbps), and low-speed (LS, 1.5-Mbps) transfers. Additionally, it can be configured as a USB 1.1 full-speed/low-speed DRD.

The main features of NVS2310 USB 2.0 OTG controller are as follows.

[GENERAL FEATURES]

- Supports Slave, External DMA Controller Interface
- Includes USB power management features
- Includes power-saving features (clock gating, two power rails for advanced power management)
- Supports packet-based, Dynamic FIFO memory allocation for endpoints for small FIFOs and flexible, efficient use of RAM
- Uses single-port RAM
- Provides support to change an endpoint's FIFO memory size during transfers
- Supports endpoint FIFO sizes that are not powers of 2, to allow the use of contiguous memory locations
- Supports the Keep-Alive in Low-Speed mode and SOFs in High/Full-Speed modes
- Optional support for Transmit and Receive thresholding in DMA mode when dedicated Tx FIFO is selected in Device mode. Thresholding and threshold length selectable through global registers. For supporting thresholding, the AHB must be run at 60 MHz or higher.

[SOFTWARE FEATURES]

- Software handles USB commands (SETUP transactions are detected and their command payloads are forwarded to the application for decoding).
- Software handles USB errors.

[APPLICATION FEATURES]

- Interfaces for the application via the AHB:
- AHB Slave interface for accessing Control and Status Registers (CSRs), the Data FIFO, and queues
- Optional AHB Master interface for Data FIFO access when Internal DMA is enabled
- Supports all AHB burst types in AHB Slave interface
- Software-selectable AHB burst type on AHB Master interface
- Takes care of the 1KB boundary breakup.
- Optional support for a dedicated transmit FIFO for each of the device IN endpoints in Slave and DMA modes. Each FIFO can hold multiple packets.

[USB 2.0 SUPPORTED FEATURES]

- Complies with the On-The-Go Supplement to the USB 2.0 Specification (Revision 1.0a)
- Operates in High-Speed (HS, 480-Mbps), Full-Speed (FS, 12-Mbps) and Low-Speed (LS, 1.5-Mbps) modes
  Caution : In host role controller, Low-Speed device connection through Full-Speed Hub is not supported.
- Supports the UTMI+ Level 3 interface (Revision 1.0, February 25th, 2004). 8-, 16-, and 8/16-bit data buses are supported.
- Supports Session Request Protocol (SRP)

- Supports Host Negotiation Protocol (HNP)
- Supports up to 16 bidirectional endpoints, including control endpoint 0
- Supports up to 16 host channels. In Host mode, when the number of device endpoints to be supported is more than the number of host channels, software can reprogram the channels to support up to 127 devices, each having 32 endpoints (IN + OUT), for a maximum of 4, 064 endpoints.
- Supports a generic root hub
- Includes automatic ping capabilities

[POWER FEATURES]

- PHY clock gating support during USB Suspend mode and Session-Off mode
- AHB clock gating support during USB Suspend mode and Session-Off mode

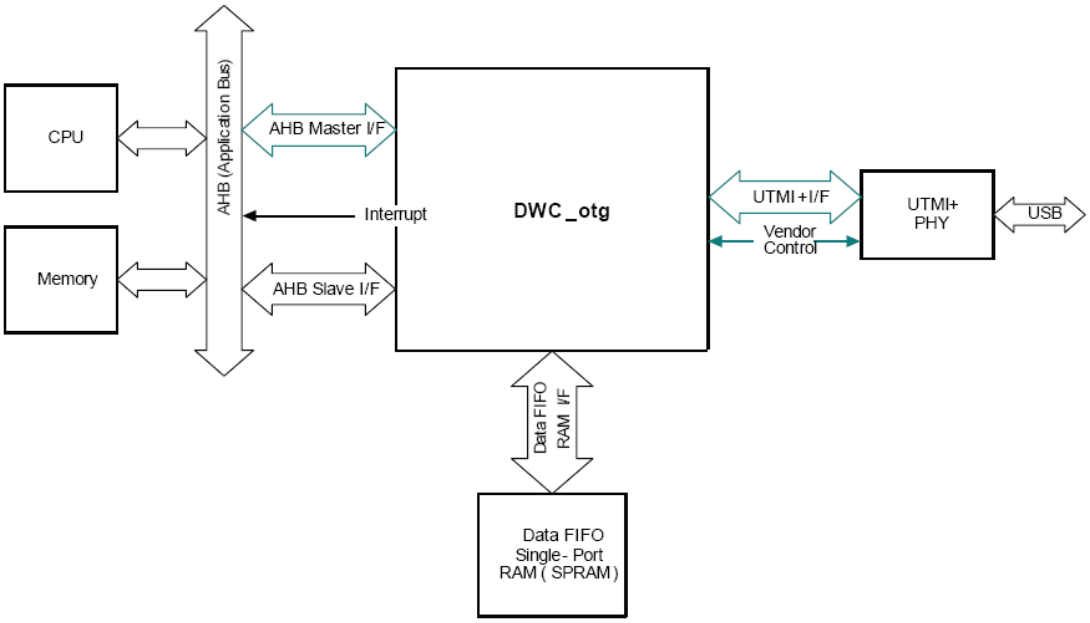


Figure 13.1 USB Controller Block Diagram

13.2 Register Description for USB 2.0 OTG Controller

By reading from and writing to the Control and Status Registers (CSRs) through the AHB Slave interface. CSRs are classified as follows:

- Core Global Registers
- Host Mode Registers
- -Host Global Registers
- -Host Port CSRs
- -Host Channel-Specific Registers
- Device Mode Registers
- -Device Global Registers
- -Device Endpoint-Specific Registers
- Power and Clock-Gating Registers
- Data FIFO (DFIFO) Access Registers

Only the Core Global, Power and Clock Gating, Data FIFO Access, and Host Port registers can be accessed in both Host and Device modes. When operating in one mode, either Device or Host, the application must not access registers from the other mode. If an illegal access occurs, a Mode Mismatch interrupt is generated and reflected in the Core Interrupt register (GINTSTS.ModeMis).

When the core switches from one mode to another, the registers in the new mode of operation must be reprogrammed as they would be after a power-on reset.

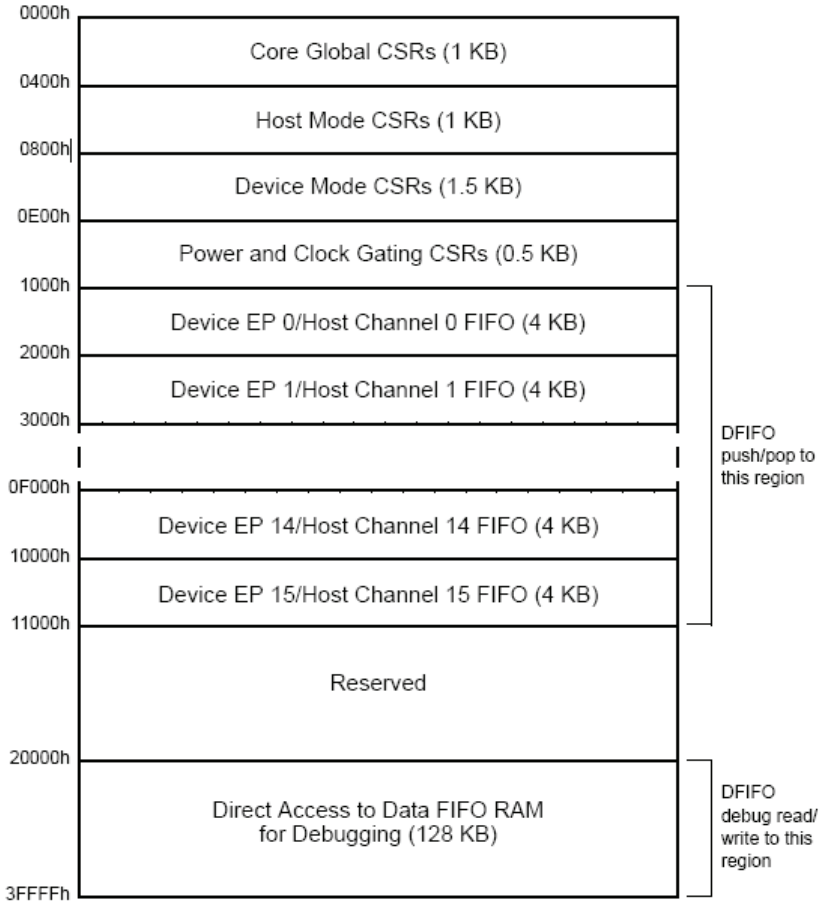


Figure 13.2 USB OTG CSR Memory Map

Table 13.1 USB Register Map (Base Address = 0xB0040000)

Core Global CSR Map			
Name	**Address**	**Reset**	**Description**
GOTGCTL	0x000	0x00010000	OTG Control and Status Register
GOTGINT	0x004	0x00000000	OTG Interrupt Register
GAHBCFG	0x008	0x00000000	Core AHB Configuration Register
GUSBCFG	0x00C	0x00001400	Core USB Configuration Register
GRSTCTL	0x010	0x80000000	Core Reset Register
GINTSTS	0x014	0x04000000	Core Interrupt Register
GINTMSK	0x018	0x00000000	Core Interrupt Mask Register
GRXSTSR	0x01C	0x00000000	Receive Status Debug Read Register (Read Only)
GRXSTSP	0x020	0x00000000	Receive Status Read /Pop Register (Read Only)
GRXFSIZ	0x024	0x00000000	Receive FIFO Size Register
GNPTXFSIZ	0x028	0x00000000	Non-periodic Transmit FIFO Size Register
GNPTXSTS	0x02C	0x00000000	Non-periodic Transmit FIFO/Queue Status Register
	0x030-0x038		Reserved
GUID	0x03C	0x01234567	User ID Register
	0x040		Reserved
GHWCFG1	0x044	0x00000000	User HW Config1 Register(Read Only)
GHWCFG2	0x048	0x228FFCF0	User HW Config2 Register(Read Only)
GHWCFG3	0x04C	0x800000E9	User HW Config3 Register(Read Only)
GHWCFG4	0x050	0x1FF08030	User HW Config4 Register(Read Only)
	0x054-0x0FF		Reserved
HPTXFSIZ	0x100	0x00000000	Host Periodic Transmit FIFO Size Register
DIEPTXFn	0x104-0x13C	0x00000000	Device IN Endpoint Transmit FIFO Size Register
	0x140-0x3FF		Reserved
Host Mode CSR Map			
Name	**Address**	**Reset**	**Description**
HCFG	0x400	0x00000000	Host Configuration Register
HFIR	0x404	0x0000EA60	Host Frame Interval Register
HFNUM	0x408	0x00003FFF	Host Frame Number/Frame Time Remaining Register
	0x40C		Reserved
HPTXSTS	0x410	0x00000000	Host Periodic Transmit FIFO/Queue Status Register
HAINT	0x414	0x00000000	Host All Channels Interrupt Register
HAINTMSK	0x418	0x00000000	Host All Channels Interrupt Mask Register
HPRT	0x440	0x00000000	Host Port Control and Status Register
	0x444-0x4FC		Reserved
HCCHARn	0x500 + n * 0x20	0x00000000	Host Channel n Characteristics Register (n=0~15)
HCSPLTn	0x504 + n * 0x20	0x00000000	Host Channel n Split Control Register (n=0~15)
HCINTn	0x508 + n * 0x20	0x00000000	Host Channel n Interrupt Register (n=0~15)
HCINTMSKn	0x50C + n * 0x20	0x00000000	Host Channel n Interrupt Mask Register (n=0~15)
HCTSIZn	0x510 + n * 0x20	0x00000000	Host Channel n Transfer Size Register (n=0~15)
HCDMAn	0x514 + n * 0x20	0x00000000	Host Channel n DMA Address Register (n=0~15)
	0x518 + n * 0x20 - 0x51C + n * 0x20		Reserved
	0x6FD-0x7FF		Reserved
Device Mode CSR Map			
Name	**Address**	**Reset**	**Description**
DCFG	0x800	0x00000000	Device Configuration Register
DCTL	0x804	0x00000000	Device Control Register
DSTS	0x808	0x00000002	Device Status Register (Read Only)
	0x80C		Reserved
DIEPMSK	0x810	0x00000000	Device IN Endpoint Common Interrupt Mask Register
DOEPMSK	0x814	0x00000000	Device OUT Endpoint Common Interrupt Mask Register
DAINT	0x818	0x00000000	Device All Endpoints Interrupt Register
DAINTMSK	0x81C	0x00000000	Device All Endpoints Interrupt Mask Register
	0x820-0x824		Reserved
	0x830-0x834		Reserved
DVBUSDIS	0x828	0x00000B8F	Device VBUS Discharge Time Register
DVBUSPULSE	0x82C	0x000002C6	Device VBUS Pulsing Time Register
DTHRCTL	0x830	0x01000000	Device Threshold Control Register
DIEPEMPMSK	0x834	0x00000000	Device IN Endpoint FIFO Empty Interrupt Mask Register
	0x838-0x8FF		Reserved

DIEPCTL0	0x900	0x00001000	Device Control IN Endpoint 0 Control Register
DIEPCTLn	0x900 + (n * 0x20)	0x00000000	Device Control IN Endpoint n Control Register (n=1~15)
	0x904 + (n * 0x20)		Reserved (n=0~15)
DIEPINTn	0x908 + (n * 0x20)	0x00000080	Device IN Endpoint n Interrupt Register (n=0~15)
	0x90C + (n * 0x20)		Reserved (n=0~15)
DIEPTSIZ0	0x910	0x00000000	Device IN Endpoint 0 Transfer Size Register
DIEPTSIZn	0x910 + (n * 0x20)	0x00000000	Device IN Endpoint n Transfer Size Register (n=1~15)
DIEPDMAn	0x914 + (n * 0x20)	0x00000000	Device IN Endpoint n DMA Address Register (n=0~15)
DTXFSTSn	0x918 + (n * 0x20)	0x00000000	Device IN Endpoint Transmit FIFO Status Register (n=0~15)
	0x918 + (n * 0x20) - 0x91C + (n * 0x20)		Reserved (n=0~15)
DOEPCTL0	0xB00	0x00001000	Device Control OUT Endpoint 0 Control Register
DOEPCTLn	0xB00 + (n * 0x20)	0x00000000	Device Control OUT Endpoint n Control Register (n=1~15)
	0xB04 + (n * 0x20)		Reserved (n=0~15)
DOEPINTn	0xB08 + (n * 0x20)	0x00000080	Device OUT Endpoint n Interrupt Register (n=0~15)
	0xB0C + (n * 0x20)		Reserved (n=0~15)
DOEPTSIZ0	0xB10	0x00000000	Device OUT Endpoint 0 Transfer Size Register
DOEPTSIZn	0xB10 + (n * 0x20)	0x00000000	Device OUT Endpoint 0 Transfer Size Register (n=1~15)
DOEPDMAn	0xB14 + (n * 0x20)	0x00000000	Device OUT Endpoint 0 DMA Address Register (n=0~15)
	0xB18 + (n * 0x20) - 0xB1C + (n * 0x20)		Reserved (n=0~15)
	0xCFD-0xDFF		Reserved

Power and Clock Gating CSR Map

Name	Address	Reset	Description
PCGCR	0xE00	0x00000000	Power and Clock Gating Control Register
	0xE04-0xFFF		Reserved

Data FIFO(DFIFO) Access Register Map

Name	Address	Type	Reset	Description
DFIFO0W	0x1000-0x1FFC	W	-	Device IN Endpoint 0/Host OUT Channel 0 : DFIFO Write Access
DFIFO0R		R	-	Device OUT Endpoint 0/Host IN Channel 0 : DFIFO Read Access
DFIFO1W	0x2000-0x1FFC	W	-	Device IN Endpoint 1/Host OUT Channel 1 : DFIFO Write Access
DFIFO1R		R	-	Device OUT Endpoint 1/Host IN Channel 1 : DFIFO Read Access
~	~			~
DFIFO14W	0xF000-0xFFFC	W	-	Device IN Endpoint 14/Host OUT Channel 14 : DFIFO Write Access
DFIFO14R		R	-	Device OUT Endpoint 14/Host IN Channel 14 : DFIFO Read Access
DFIFO15W	0x10000-0x10FFC	W	-	Device IN Endpoint 15/Host OUT Channel 15 : DFIFO Write Access
DFIFO15R		R	-	Device OUT Endpoint 15/Host IN Channel 15 : DFIFO Read Access

Table 13.2 USB OTG Register (Base Address = 0xB0080020)

Name	Addr.	Description
OTGCR	0x0	**USBOTG Configuration Register**

OTG Control and Status Register (GOTGCTL) 0xB0040000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
												BSesVld	ASesVld	DbncTime	ConIDSts
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				DevHNPEn	HstSetHNPEn	HNPReq	HstNegScs							SesReq	SesReqScs

The OTG Control and Status register controls the behavior and reflects the status of the OTG function of the core.

Field	Name	Mode	RW	Reset	Description
31-20	-	-	-	-	Reserved
19	BSesVld	Device	R	0x0	B-Session Valid Indicates the Device mode transceiver status. • 0: B-session is not valid. • 1: B-session is valid. In OTG mode, you can use this bit to determine if the device is connected or disconnected.
18	ASesVld	Host	R	0x0	A-Session Valid Indicates the Host mode transceiver status. • 0: A-session is not valid • 1: A-session is valid
17	DbncTime	Host	R	0x0	Long/Short Debounce Time Indicates the debounce time of a detected connection. • 0: Long debounce time, used for physical connections (100 ms + 2.5 μs) • 1: Short debounce time, used for soft connections (2.5 μs)
16	ConIDSts	Host Device	R	0x1	Connector ID Status Indicates the connector ID status on a connect event. • 0: The OTG core is in A-Device mode • 1: The OTG core is in B-Device mode
15-12	-	-	-	0x0	Reserved
11	DevHNPEn	Device	R/W	0x0	Device HNP Enabled The application sets this bit when it successfully receives a SetFeature.SetHNPEnable command from the connected USB host. • 0: HNP is not enabled in the application • 1: HNP is enabled in the application
10	HstSetHNPEn	Host	R/W	0x0	Host Set HNP Enable The application sets this bit when it has successfully enabled HNP (using the SetFeature. SetHNPEnable command) on the connected device. • 0: Host Set HNP is not enabled • 1: Host Set HNP is enabled
9	HNPReq	Device	R/W	0x0	HNP Request The application sets this bit to initiate an HNP request to the connected USB host. The application can clear this bit by writing a 0 when the Host Negotiation Success Status Change bit in the OTG Interrupt register (GOTGINT.HstNegSucStsChng) is set. The core clears this bit when the HstNegSucStsChng bit is cleared. • 0: No HNP request • 1: HNP request
8	HstNegScs	Device	R	0x0	Host Negotiation Success The core sets this bit when host negotiation is successful. The core clears this bit when the HNP Request (HNPReq) bit in this register is set. • 0: Host negotiation failure • 1: Host negotiation success
7-2	-	-	-	0x0	Reserved
1	SesReq	Device	R/W	0x0	Session Request The application sets this bit to initiate a session request on the USB. The application can clear this bit by writing a 0 when the Host Negotiation Success Status Change bit in the OTG Interrupt register (GOTGINT.HstNegSucStsChng) is set. The core clears this

					bit when the HstNegSucStsChng bit is cleared. If you use the USB 1.1 Full-Speed Serial Transceiver interface to initiate the session request, the application must wait until the VBUS discharges to 0.2 V, after the B-Session Valid bit in this register (GOTGCTL.BSesVld) is cleared. This discharge time varies between different PHYs and can be obtained from the PHY vendor. • 0: No session request • 1: Session request
0	SesReqScs	Device	R	0x0	Session Request Success The core sets this bit when a session request initiation is successful. • 0: Session request failure • 1: Session request success

OTG Interrupt Register (GOTGINT) 0xB0040004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
												DbnceDone	ADevTOUTChg	HstNegDet	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						HstNegSucStsChng	SesReqSucStsChng						SesEndDet		

The application reads this register whenever there is an OTG interrupt and clears the bits in this register to clear the OTG interrupt.

Field	Name	Mode	RW	Reset	Description
31-20	-	-	-	-	Reserved
19	DbnceDone	Host	R/W	0x0	Debounce Done The core sets this bit when the debounce is completed after the device connect. The application can start driving USB reset after seeing this interrupt. This bit is only valid when the HNP Capable or SRP Capable bit is set in the Core USB Configuration register (GUSBCFG.HNPCap or GUSBCFG.SRPCap, respectively).
18	ADevTOUTChg	Host Device	R/W	0x0	A-Device Timeout Change () The core sets this bit to indicate that the A-device has timed out while waiting for the B-device to connect.
17	HstNegDet	Host Device	R/W	0x0	Host Negotiation Detected The core sets this bit when it detects a host negotiation request on the USB.
16-10	-	-	-	0x0	Reserved
9	HstNegSucStsChng	Host Device	R/W	0x0	Host Negotiation Success Status Change The core sets this bit on the success or failure of a USB host negotiation request. The application must read the Host Negotiation Success bit of the OTG Control and Status register (GOTGCTL.HstNegScs) to check for success or failure.
8	SesReqSucStsChng	Host Device	R/W	0x0	Session Request Success Status Change The core sets this bit on the success or failure of a session request. The application must read the Session Request Success bit in the OTG Control and Status register (GOTGCTL.SesReqScs) to check for success or failure.
7-3	-	-	-	0x0	Reserved
2	SesEndDet	Host Device	R/W	0x0	Session End Detected The core sets this bit when the utmiotg_bvalid signal is deasserted.
1-0	-	-	-	0x0	Reserved

Core AHB Configuration Register (GAHBCFG) 0xB0040008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							PTxFEmpLvl	NPTxFEmpLvl		DMAEn	HBstLen				GlblIntrMsk

This register can be used to configure the core after power-on or a change in mode of operation. This register mainly contains AHB system-related configuration parameters. Do not change this register after the initial programming. The application must program this register before starting any transactions on either the AHB or the USB..

Field	Name	Mode	RW	Reset	Description
31-9	-	-	-	0	Reserved
8	PTxFEmpLvl	Host	R/W	0x0	Periodic TxFIFO Empty Level Indicates when the Periodic TxFIFO Empty Interrupt bit in the Core Interrupt register (GINTSTS.PTxFEmp) is triggered. This bit is used only in Slave mode. • 1’b0: GINTSTS.PTxFEmp interrupt indicates that the Periodic TxFIFO is half empty • 1’b1: GINTSTS.PTxFEmp interrupt indicates that the Periodic TxFIFO is completely empty
7	NPTxFEmpLvl	Host Device	R/W	0x0	Non-Periodic TxFIFO Empty Level This bit is used only in Slave mode. In host mode and with Shared FIFO with device mode, this bit indicates when the Non-Periodic TxFIFO Empty Interrupt bit in the Core Interrupt register (GINTSTS.NPTxFEmp) is triggered. With dedicated FIFO in device mode, this bit indicates when IN endpoint Transmit FIFO empty interrupt (DIEPINTn.TxFEmp) is triggered. • 1’b0: DIEPINTn.TxFEmp interrupt indicates that the IN Endpoint TxFIFO is half empty • 1’b1: DIEPINTn.TxFEmp interrupt indicates that the IN Endpoint TxFIFO is completely empty
6	-	-	-	0x0	Reserved
5	DMAEn	Host Device	R/W	0x0	DMA Enable • 1’b0: Core operates in Slave mode • 1’b1: Core operates in a DMA mode
4-1	HBstLen	Host Device	R/W	0x0	Burst Length/Type () AHB Master burst type: • 4’b0000 Single • 4’b0001 INCR • 4’b0011 INCR4 • 4’b0101 INCR8 • 4’b0111 INCR16 • Others: Reserved
0	GlblIntrMsk	Host Device	R/W	0x0	Global Interrupt Mask The application uses this bit to mask or unmask the interrupt line assertion to itself. Irrespective of this bit’s setting, the interrupt status registers are updated by the core. • 1’b0: Mask the interrupt assertion to the application. • 1’b1: Unmask the interrupt assertion to the application.

Core USB Configuration Register (GUSBCFG) 0xB004000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CorTxP	ForceDevMode	ForceHstMode							TermSelDLPulse						OtgI2CSel
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PhyLPwrClkSel		USBTrdTim				HNPCap	SRPCap		PHYSel	FSIntf	ULPI_UTMI_Sel	PHYIf	TOutCal		

This register can be used to configure the core after power-on or a changing to Host mode or Device mode. It contains USB and USB-PHY related configuration parameters. The application must program this register before starting any transactions on either the AHB or the USB. Do not make changes to this register after the initial programming.

Field	Name	Mode	RW	Reset	Description
31	CorTXP	Host Device	R	0x0	Corrupt Tx packet This bit is for debug purposes only. Never set this bit to 1.
30	ForceDevMode	Host Device	R/W	0x0	Force Device Mode Writing a 1 to this bit will force the core to device mode irrespective of utmiotg_iddig input pin. • 1’b0 : Normal Mode. • 1’b1 : Force Device Mode. After setting the force bit, the application must wait at least 25ms before the change to take effect.
29	ForceHstMode	Host Device	R/W	0x0	Force Host Mode Writing a 1 to this bit will force the core to host mode irrespective of utmiotg_iddig input pin. • 1’b0 : Normal Mode. • 1’b1 : Force Host Mode. After setting the force bit, the application must wait at least 25ms before the change to take effect.
28-23	-	-	-	0x0	Reserved
22	TermSelDLPulse	Device	R/W	0x0	TermSel DLine Pulsing Selection This bit selects utmi_termselect to drive data line pulse during SRP. • 1’b0: Data line pulsing using utmi_txvalid (default). • 1’b1: Data line pulsing using utmi_termsel.
21-17	-	-	-	0x0	Reserved
16	OtgI2CSel	Host Device	R	0x0	UTMIFS or I2C Interface Select The application uses this bit to select the I2C interface. • 1’b0: UTMI USB 1.1 Full-Speed interface for OTG signals • 1’b1: I2C interface for OTG signals(not supported)
15	PhyLPwrClkSel	Host Device	R/W	0x0	PHY Low-Power Clock Select Selects either 480-MHz or 48-MHz (low-power) PHY mode. In FS and LS modes, the PHY can usually operate on a 48-MHz clock to save power. • 1’b0: 480-MHz Internal PLL clock • 1’b1: 48-MHz External Clock In 480 MHz mode, the UTMI interface operates at either 60 or 30-MHz, depending upon whether 8- or 16-bit data width is selected. In 48-MHz mode, the UTMI interface operates at 48MHz in FS mode and at either 48 or 6 MHz in LS mode (depending on the PHY vendor). This bit drives the utmi_fsls_low_power core output signal, and is valid only for UTMI+ PHYs.
14	-	-	-	0x0	Reserved
13-10	USBTrdTim	Device	R/W	0x5	USB Turnaround Time Sets the turnaround time in PHY clocks. Specifies the response time for a MAC request to the Packet FIFO Controller (PFC) to fetch data from the DFIFO (SPRAM). This must be programmed to • 4’h5: When the MAC interface is 16-bit UTMI+ . • 4’h9: When the MAC interface is 8-bit UTMI+ . Note: The values above are calculated for the minimum AHB

					frequency of 30MHz. USB turnaround time is critical for certification where long cables and 5-Hubs are used, so if you need the AHB to run at less than 30 MHz, and if USB turnaround time is not critical, these bits can be programmed to a larger value.
9	HNPCap	Host Device	R/W, R	0x0	HNP-Capable The application uses this bit to control the OTG core's HNP capabilities. • 1'b0: HNP capability is not enabled. • 1'b1: HNP capability is enabled..
8	SRPCap	Host Device	R/W, R	0x0	SRP-Capable The application uses this bit to control SRP capabilities. If the core operates as a non-SRP-capable B-device, it cannot request the connected A-device (host) to activate VBUS and start a session. • 1'b0: SRP capability is not enabled. • 1'b1: SRP capability is enabled.
7	-	-	-	0x0	Reserved
6	PHYSel	Host Device	R/W, R	0x0	USB 2.0 High-Speed PHY or USB 1.1 Full-Speed Serial Transceiver Select The application uses this bit to select either a high-speed UTMI+, or a full-speed transceiver. • 1'b0: USB 2.0 high-speed UTMI+ • 1'b1: USB 1.1 full-speed serial transceiver
5	FSIntf	Host Device	R/W, W	0x0	Full-Speed Serial Interface Select The application uses this bit to select either a unidirectional or bidirectional USB 1.1 full-speed serial transceiver interface. • 1'b0: 6-pin unidirectional full-speed serial interface • 1'b1: 3-pin bidirectional full-speed serial interface
4	ULPI_UTMI_Sel	Host Device	R/W, R	0x0	ULPI or UTMI+ Select The application uses this bit to select either a UTMI+ interface or ULPI Interface. • 1'b0: UTMI+ Interface • 1'b1: ULPI Interface(not supported)
3	PHYIf	Host Device	R/W, R	0x0	PHY Interface The application uses this bit to configure the core to support a UTMI+ PHY with an 8- or 16-bit interface. • 1'b0: 8 bits (not supported) • 1'b1: 16 bits
2-0	TOutCal	Host Device	R/W	0x0	HS/FS Timeout Calibration The number of PHY clocks that the application programs in this field is added to the high-speed/full-speed interpacket timeout duration in the core to account for any additional delays introduced by the PHY. This can be required, because the delay introduced by the PHY in generating the linestate condition can vary from one PHY to another. The USB standard timeout value for high-speed operation is 736 to 816 (inclusive) bit times. The USB standard timeout value for full-speed operation is 16 to 18 (inclusive) bit times. The application must program this field based on the speed of enumeration. The number of bit times added per PHY clock are: High-speed operation: • One 30-MHz PHY clock = 16 bit times • One 60-MHz PHY clock = 8 bit times Full-speed operation: • One 30-MHz PHY clock = 0.4 bit times • One 60-MHz PHY clock = 0.2 bit times • One 48-MHz PHY clock = 0.25 bit times

Core Reset Register (GRSTCTL) 0xB0040010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AHBIdle	DMAReq														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					TxFNum					TxFFlsh	RxFFlsh	INTknQFlsh	FrmCntrRst	HSftRst	CSftRst

The application uses this register to reset various hardware features inside the core..

Field	Name	Mode	RW	Reset	Description
31	AHBIdle	Host Device	R	0x1	AHB Master Idle Indicates that the AHB Master State Machine is in the IDLE condition.
30	DMAReq	Host Device	R	0x0	DMA Request Signal Indicates that the DMA request is in progress. Used for debug.
29-11	-	-	-	0x0	Reserved
10-6	TxFNum	Host Device	R/W	0x0	TxFIFO Number This is the FIFO number that must be flushed using the TxFIFO Flush bit. This field must not be changed until the core clears the TxFIFO Flush bit. • 5’h0: - Non-periodic TxFIFO flush in Host mode - Tx FIFO 0 flush in device mode • 5’h1: - Periodic TxFIFO flush in Host mode - TXFIFO 1 flush in device mode • 5’h2: - TXFIFO 2 flush in device mode ... • 5’hF: - TXFIFO 15 flush in device mode • 5’h10: Flush all the transmit FIFOs in device or host mode.
5	TxFFlsh	Host Device	R/W	0x0	TxFIFO Flush This bit selectively flushes a single or all transmit FIFOs, but cannot do so if the core is in the midst of a transaction. The application must write this bit only after checking that the core is neither writing to the TxFIFO nor reading from the TxFIFO. Verify using these registers: • Read–NAK Effective Interrupt ensures the core is not reading from the FIFO • Write–GRSTCTL.AHBIdle ensures the core is not writing anything to the FIFO. Flushing is normally recommended when FIFOs are reconfigured. FIFO flushing is also recommended during device endpoint disable. The application must wait until the core clears this bit before performing any operations. This bit takes eight clocks to clear, using the slower clock of phy_clk or hclk.
4	RxFFlsh	Host Device	R/W	0x0	RxFIFO Flush The application can flush the entire RxFIFO using this bit, but must first ensure that the core is not in the middle of a transaction. The application must only write to this bit after checking that the core is neither reading from the RxFIFO nor writing to the RxFIFO. The application must wait until the bit is cleared before performing any other operations. This bit requires 8 clocks (slowest of PHY or AHB clock) to clear.
3	INTknQFlsh	Device	R/W	0x0	IN Token Sequence Learning Queue Flush The application writes this bit to flush the IN Token Sequence Learning Queue.
2	FrmCntrRst	Host	R/W	0x0	Host Frame Counter Reset The application writes this bit to reset the (micro)frame number counter inside the core. When the (micro)frame counter is reset, the subsequent SOF sent out by the core has a (micro)frame number of 0.
1	HSftRst	Host	R/W	0x0	HClk Soft Reset

		Device			The application uses this bit to flush the control logic in the AHB Clock domain. Only AHB Clock Domain pipelines are reset. • FIFOs are not flushed with this bit. • All state machines in the AHB clock domain are reset to the Idle state after terminating the transactions on the AHB, following the protocol. • CSR control bits used by the AHB clock domain state machines are cleared. • To clear this interrupt, status mask bits that control the interrupt status and are generated by the AHB clock domain state machine are cleared. • Because interrupt status bits are not cleared, the application can get the status of any core events that occurred after it set this bit. This is a self-clearing bit that the core clears after all necessary logic is reset in the core. This can take several clocks, depending on the core’ s current state.
0	CSftRst	Host Device	R/W	0x0	Core Soft Reset Resets the hclk and phy_clock domains as follows: • Clears the interrupts and all the CSR registers except the following register bits: - PCGCCTL.RstPdwnModule - PCGCCTL.GateHclk - PCGCCTL.PwrClmp - PCGCCTL.StopPPhyLPwrClkSelclk - GUSBCFG.PhyLPwrClkSel - GUSBCFG.DDRSel - GUSBCFG.PHYSel - GUSBCFG.FSIntf - GUSBCFG.ULPI_UTMI_Sel - GUSBCFG.PHYIf - HCFG.FSLSPclkSel - DCFG.DevSpd - GGPIO • All module state machines (except the AHB Slave Unit) are reset to the IDLE state, and all the transmit FIFOs and the receive FIFO are flushed. • Any transactions on the AHB Master are terminated as soon as possible, after gracefully completing the last data phase of an AHB transfer. Any transactions on the USB are terminated immediately. The application can write to this bit any time it wants to reset the core. This is a self-clearing bit and the core clears this bit after all the necessary logic is reset in the core, which can take several clocks, depending on the current state of the core. Once this bit is cleared software must wait at least 3 PHY clocks before doing any access to the PHY domain (synchronization delay). Software must also must check that bit 31 of this register is 1 (AHB Master is IDLE) before starting any operation. Typically software reset is used during software development and also when you dynamically change the PHY selection bits in the USB configuration registers listed above. When you change the PHY, the corresponding clock for the PHY is selected and used in the PHY domain. Once a new clock is selected, the PHY domain has to be reset for proper operation.

Core Interrupt Register (GINTSTS) 0xB0040014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WkUpInt	SessReqInt	DisconnInt	ConIDStsChng		PTxFEmp	HChInt	PrtInt		FetSusp	incompIP/ incompISOOUT	incompISOIN	OEPInt	IEPInt	EPMis	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EOPF	ISOOutDrop	EnumDone	USBRst	USBSusp	ErlySusp			GOUTNakEff	GINNakEff		RxFLvl	Sof	OTGInt	ModeMis	CurMod

This register interrupts the application for system-level events in the current mode of operation (Device mode or Host mode). Some of the bits in this register are valid only in Host mode, while others are valid in Device mode only. This register also indicates the current mode of operation. In order to clear the interrupt status bits of type R/W, the application must write 1'b1 into the bit.

The FIFO status interrupts are read only; once software reads from or writes to the FIFO while servicing these interrupts, FIFO interrupt conditions are cleared automatically.

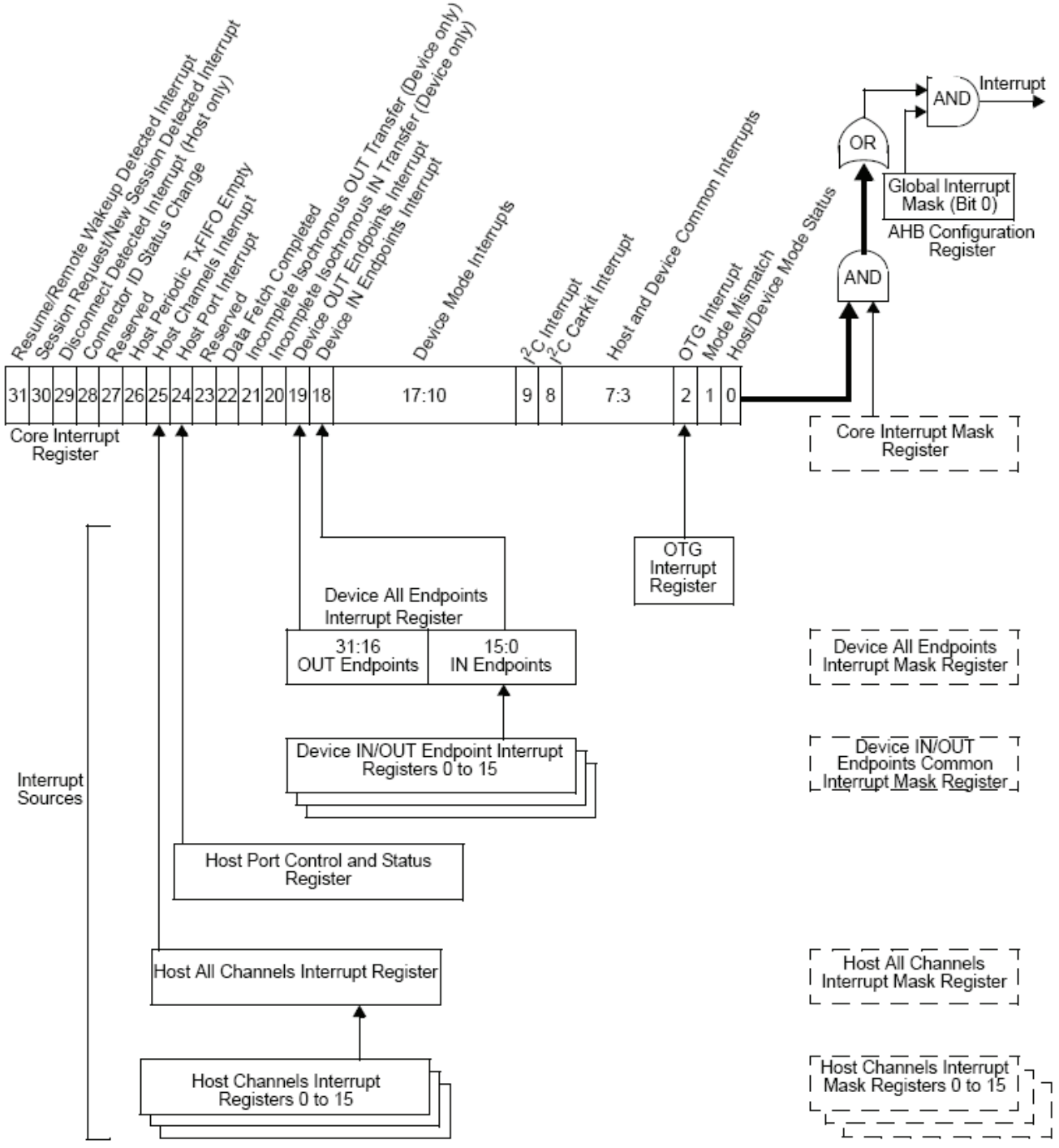
The Application must clear the GINTSTS register at initialization before unmasking the interrupt bit to avoid any interrupts generated prior to initialization.

Field	Name	Mode	RW	Reset	Description
31	WkUpInt	Host Device	R/W	0x0	Resume/Remote Wakeup Detected Interrupt In Device mode, this interrupt is asserted when a resume is detected on the USB. In Host mode, this interrupt is asserted when a remote wakeup is detected on the USB.
30	SessReqInt	Host Device	R/W	0x0	Session Request/New Session Detected Interrupt In Host mode, this interrupt is asserted when a session request is detected from the device. In Device mode, this interrupt is asserted when the utmiotg_bvalid signal goes high.
29	DisconnInt	Host	R/W	0x0	Disconnect Detected Interrupt Asserted when a device disconnect is detected.
28	ConIDStsChng	Host Device	R/W	0x0	Connector ID Status Change The core sets this bit when there is a change in connector ID status.
27	-	-	-	0x0	Reserved
26	PTxFEmp	Host	R	0x1	Periodic TxFIFO Empty Asserted when the Periodic Transmit FIFO is either half or completely empty and there is space for at least one entry to be written in the Periodic Request Queue. The half or completely empty status is determined by the Periodic TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.PTxFEmpLvl).
25	HChInt	Host	R	0x0	Host Channels Interrupt The core sets this bit to indicate that an interrupt is pending on one of the channels of the core (in Host mode). The application must read the Host All Channels Interrupt (HAINT) register to determine the exact number of the channel on which the interrupt occurred, and then read the corresponding Host Channel-n Interrupt (HCINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the HCINTn register to clear this bit.
24	PrtInt	Host	R	0x0	Host Port Interrupt The core sets this bit to indicate a change in port status of one of ports in Host mode. The application must read the Host Port Control and Status (HPRT) register to determine the exact event that caused this interrupt. The application must clear the appropriate status bit in the Host Port Control and Status register to clear this bit.
23	-	-	-	0x0	Reserved
22	FetSusp	Device	R/W	0x0	Data Fetch Suspended This interrupt is valid only in DMA mode. This interrupt indicates that the core has stopped fetching data for IN endpoints due to the unavailability of TxFIFO space or Request Queue space.

					This interrupt is used by the application for an endpoint mismatch algorithm. For example, after detecting an endpoint mismatch, the application: • Sets a global non-periodic IN NAK handshake • Disables In endpoints • Flushes the FIFO • Determines the token sequence from the IN Token Sequence Learning Queue • Re-enables the endpoints • Clears the global non-periodic IN NAK handshake If the global non-periodic IN NAK is cleared, the core has not yet fetched data for the IN endpoint, and the IN token is received: the core generates an “IN token received when FIFO empty” interrupt. The OTG then sends the host a NAK response. To avoid this scenario, the application can check the GINTSTS.FetSusp interrupt, which ensures that the FIFO is full before clearing a global NAK handshake. Alternatively, the application can mask the “IN token received when FIFO empty” interrupt when clearing a global IN NAK handshake.
21	incomplP	Host	R/W	0x0	Incomplete Periodic Transfer In Host mode, the core sets this interrupt bit when there are incomplete periodic transactions still pending which are scheduled for the current microframe.
	incompISOOUT	Device			Incomplete Isochronous OUT Transfer The Device mode, the core sets this interrupt to indicate that there is at least one isochronous OUT endpoint on which the transfer is not completed in the current microframe. This interrupt is asserted along with the End of Periodic Frame Interrupt (EOPF) bit in this register.
20	incompISOIN	Device	R/W	0x0	Incomplete Isochronous IN Transfer The core sets this interrupt to indicate that there is at least one isochronous IN endpoint on which the transfer is not completed in the current microframe. This interrupt is asserted along with the End of Periodic Frame Interrupt (EOPF) bit in this register. Note: This interrupt is not asserted in Scatter/Gather DMA mode.
19	OEPInt	Device	R	0x0	OUT Endpoints Interrupt The core sets this bit to indicate that an interrupt is pending on one of the OUT endpoints of the core (in Device mode). The application must read the Device All Endpoints Interrupt (DAINT) register to determine the exact number of the OUT endpoint on which the interrupt occurred, and then read the corresponding Device OUT Endpoint-n Interrupt (DOEPINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the corresponding DOEPINTn register to clear this bit.
18	IEPInt	Device	R	0x0	IN Endpoints Interrupt The core sets this bit to indicate that an interrupt is pending on one of the IN endpoints of the core (in Device mode). The application must read the Device All Endpoints Interrupt (DAINT) register to determine the exact number of the IN endpoint on which the interrupt occurred, and then read the corresponding Device IN Endpoint-n Interrupt (DIEPINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the corresponding DIEPINTn register to clear this bit.
17	EPMis	Device	R	0x0	Endpoint Mismatch Interrupt
16	-	-	-	0x0	Reserved
15	EOPF	Device	R/W	0x0	End of Periodic Frame Interrupt Indicates that the period specified in the Periodic Frame Interval field of the Device Configuration register (DCFG.PerFrInt) has been reached in the current microframe.

14	ISOOutDrop	Device	R/W	0x0	Isochronous OUT Packet Dropped Interrupt The core sets this bit when it fails to write an isochronous OUT packet into the RxFIFO because the RxFIFO doesn't have enough space to accommodate a maximum packet size packet for the isochronous OUT endpoint.
13	EnumDone	Device	R/W	0x0	Enumeration Done The core sets this bit to indicate that speed enumeration is complete. The application must read the Device Status (DSTS) register to obtain the enumerated speed.
12	USBRst	Device	R/W	0x0	USB Reset The core sets this bit to indicate that a reset is detected on the USB.
11	USBSusp	Device	R/W	0x0	USB Suspend The core sets this bit to indicate that a suspend was detected on the USB. The core enters the Suspended state when there is no activity on the phy_line_state_i signal for an extended period of time.
10	ErlySusp	Device	R/W	0x0	Early Suspend The core sets this bit to indicate that an Idle state has been detected on the USB for 3 ms.
9-8	-	-	-	0x0	Reserved
7	GOUTNakEff	Device	R/W	0x0	Global OUT NAK Effective Indicates that the Set Global OUT NAK bit in the Device Control register (DCTL.SGOUTNak), set by the application, has taken effect in the core. This bit can be cleared by writing the Clear Global OUT NAK bit in the Device Control register (DCTL.CGOUTNak).
6	GINNakEff	Device	R/W	0x0	Global IN Non-periodic NAK Effective Indicates that the Set Global Non-periodic IN NAK bit in the Device Control register (DCTL.SGNPInNak), set by the application, has been taken effect in the core. That is, the core has sampled the Global IN NAK bit set by the application. This bit can be cleared by clearing the Clear Global Non-periodic IN NAK bit in the Device Control register (DCTL.CGNPInNak). This interrupt does not necessarily mean that a NAK handshake is sent out on the USB. The STALL bit takes precedence over the NAK bit.
5	-	-	-	0x0	Reserved
4	RxFLvl	Host Device	R	0x0	RxFIFO Non-Empty Indicates that there is at least one packet pending to be read from the RxFIFO.
3	Sof	Host Device	R/W	0x0	Start of (micro)Frame In Host mode, the core sets this bit to indicate that an SOF (FS), micro-SOF (HS), or Keep-Alive (LS) is transmitted on the USB. The application must write a 1 to this bit to clear the interrupt. In Device mode, in the core sets this bit to indicate that an SOF token has been received on the USB. The application can read the Device Status register to get the current (micro)frame number. This interrupt is seen only when the core is operating at either HS or FS.
2	OTGInt	Host Device	R	0x0	OTG Interrupt The core sets this bit to indicate an OTG protocol event. The application must read the OTG Interrupt Status (GOTGINT) register to determine the exact event that caused this interrupt. The application must clear the appropriate status bit in the GOTGINT register to clear this bit.
1	ModeMis	Host Device	R/W	0x0	Mode Mismatch Interrupt The core sets this bit when the application is trying to access: • A Host mode register, when the core is operating in Device mode • A Device mode register, when the core is operating in Host mode The register access is completed on the AHB with an OKAY response, but is ignored by the core internally and doesn't affect the operation of the core.

0	CurMod	Host Device	R	0x0	Current Mode of Operation Indicates the current mode of operation. • 1’b0: Device mode • 1’b1: Host mode



Note: Because an interrupt mask only masks an interrupt, software must clear an interrupt before unmasking it, to avoid servicing an old interrupt.

Figure 13.3 USB OTG Controller Interrupt Hierarachy

Core Interrupt Mask Register (GINTMSK) 0xB0040018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WkUpIntMsk	SessReqIntMsk	DisconnIntMsk	ConIDStsChngMsk		PTxFEmpMsk	HChIntMsk	PrtIntMsk		FetSuspMsk	incomplPMsk/ incompISOOUTMsk	incompISOINMsk	OEPIntMsk	IEPIntMsk	EPMisMsk	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EOPFMsk	ISOOutDropMsk	EnumDoneMsk	USBRstMsk	USBSuspMsk	ErlySuspMsk			GOUTNakEffMsk	GINNakEffMsk		RxFLvlMsk	SofMsk	OTGIntMsk	ModeMisMsk	

This register works with the Core Interrupt register to interrupt the application. When an interrupt bit is masked, the interrupt associated with that bit is not generated. However, the Core Interrupt (GINTSTS) register bit corresponding to that interrupt is still set.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	Mode	RW	Reset	Description
31	WkUpIntMsk	Host Device	R/W	0x0	Resume/Remote Wakeup Detected Interrupt Mask
30	SessReqIntMsk	Host Device	R/W	0x0	Session Request/New Session Detected Interrupt Mask()
29	DisconnIntMsk	Host Device	R/W	0x0	Disconnect Detected Interrupt Mask
28	ConIDStsChngMsk	Host Device	R/W	0x0	Connector ID Status Change Mask
27	-	-	-	0x0	Reserved
26	PTxFEmpMsk	Host	R/W	0x0	Periodic TxFIFO Empty Mask
25	HChIntMsk	Host	R/W	0x0	Host Channels Interrupt Mask
24	PrtIntMsk	Host	R/W	0x0	Host Port Interrupt Mask
23	-	-	-	0x0	Reserved
22	FetSuspMsk	Device	R/W	0x0	Data Fetch Suspended Mask
21	incomplPMsk	Host	R/W	0x0	Incomplete Periodic Transfer Mask
	incomplSOOUTMsk	Device			Incomplete Isochronous OUT Transfer Mask
20	incomplSOINMsk	Device	R/W	0x0	Incomplete Isochronous IN Transfer Mask
19	OEPIntMsk	Device	R/W	0x0	OUT Endpoints Interrupt Mask
18	IEPIntMsk	Device	R/W	0x0	IN Endpoints Interrupt Mask
17	EPMisMsk	Device	R/W	0x0	Endpoint Mismatch Interrupt Mask
16	-	-	-	-	Reserved
15	EOPFMsk	Device	R/W	0x0	End of Periodic Frame Interrupt Mask
14	ISOOutDropMsk	Device	R/W	0x0	Isochronous OUT Packet Dropped Interrupt Mask
13	EnumDoneMsk	Device	R/W	0x0	Enumeration Done Mask
12	USBRstMsk	Device	R/W	0x0	USB Reset Mask
11	USBSuspMsk	Device	R/W	0x0	USB Suspend Mask
10	ErlySuspMsk	Device	R/W	0x0	Early Suspend Mask
9-8	-	-	-	0x0	Reserved
7	GOUTNakEffMsk	Device	R/W	0x0	Global OUT NAK Effective Mask
6	GINNakEffMsk	Device	R/W	0x0	Global Non-periodic IN NAK Effective Mask
5	-	-	-	0x0	Reserved
4	RxFLvlMsk	Host Device	R/W	0x0	Receive FIFO Non-Empty Mask
3	SofMsk	Host Device	R/W	0x0	Start of (micro)Frame Mask ()
2	OTGIntMsk	Host Device	R/W	0x0	OTG Interrupt Mask
1	ModeMisMsk	Host Device	R/W	0x0	Mode Mismatch Interrupt Mask
0	-	-	-	0x0	Reserved

Receive Status Debug Read/Status Read and Pop Registers (GRXSTSR/GRXSTSP)

0xB004001C(Read)/0xB0040020(Pop)

A read to the Receive Status Debug Read register returns the contents of the top of theReceive FIFO. A read to the Receive Status Read and Pop register additionally pops the top data entry out of the RxFIFO.

The receive status contents must be interpreted differently in Host and Device modes. The core ignores the receive status pop/read when the receive FIFO is empty and returns a value of 32'h0000_0000. The application must only pop the Receive Status FIFO when the Receive FIFO Non-Empty bit of the Core Interrupt register (GINTSTS.RxFLvl) is asserted.

The following table shows the use of these registers in Host mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
											PktSts				DPID[1]
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DPID[0]	BCnt											ChNum			

Field	Name	Mode	RW	Reset	Description
31-21	-	-	-	0x0	Reserved
20-17	PktSts	Host	R	0x0	Packet Status Indicates the status of the received packet • 4'b0010: IN data packet received • 4'b0011: IN transfer completed (triggers an interrupt) • 4'b0101: Data toggle error (triggers an interrupt) • 4'b0111: Channel halted (triggers an interrupt) • Others: Reserved
16-15	DPID	Host	R	0x0	Data PID Indicates the Data PID of the received packet • 2'b00: DATA0 • 2'b10: DATA1 • 2'b01: DATA2 • 2'b11: MDATA
14-4	BCnt	Host	R	0x0	Byte Count Indicates the byte count of the received IN data packet.
3-0	ChNum	Host	R	0x0	Channel Number () Indicates the channel number to which the current received packet belongs.

The following table shows the use of these registers in Device mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							FN				PktSts				DPID[1]
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DPID[0]	BCnt											EPNum			

Field	Name	Mode	RW	Reset	Description
31-25	-	-	-	0x0	Reserved
24-21	FN	Device	R	0x0	Frame Number This is the least significant 4 bits of the (micro)frame number in which the packet is received on the USB. This field is supported only when isochronous OUT endpoints are supported.
20-17	PktSts	Device	R	0x0	Packet Status () Indicates the status of the received packet • 4’b0001: Global OUT NAK (triggers an interrupt) • 4’b0010: OUT data packet received • 4’b0011: OUT transfer completed (triggers an interrupt) • 4’b0100: SETUP transaction completed (triggers an interrupt) • 4’b0110: SETUP data packet received • Others: Reserved
16-15	DPID	Device	R	0x0	Data PID Indicates the Data PID of the received OUT data packet • 2’b00: DATA0 • 2’b10: DATA1 • 2’b01: DATA2 • 2’b11: MDATA
14-4	BCnt	Device	R	0x0	Byte Count Indicates the byte count of the received IN data packet.

Receive FIFO Size Register (GRXFSIZ) 0xB0040024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RxFDep															

The application can program the RAM size and the memory start address for the Nonperiodic TxFIFO.

Field	Name	Mode	RW	Reset	Description
31-16	-	-	-	0x0	Reserved
15-0	RxFDep	Host Device	R/W, R	0x0	RxFIFO Depth This value is in terms of 32-bit words. • Minimum value is 16 • Maximum value is 4160 The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth (parameter OTG_RX_DFIFO_DEPTH) during coreConsultant configuration. If Enable Dynamic FIFO Sizing? was deselected in coreConsultant (parameter OTG_DFIFO_DYNAMIC = 0), these flops are optimized, and reads return the power-on value. If Enable Dynamic FIFO Sizing? was selected in coreConsultant (parameter OTG_DFIFO_DYNAMIC = 1), you can write a new value in this field. You can write a new value in this field. Programmed values must not exceed the power-on value set in coreConsultant.

Non-Periodic Transmit FIFO Size Register (GNPTXFSIZ) 0xB0040028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NPTxFDep/ INEPTxF0Dep															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NPTxFStAddr/ INEPTxF0StAddr															

The application can program the RAM size that must be allocated to the RxFIFO.

Field	Name	Mode	RW	Reset	Description
31-16	NPTxFDep	Host	R/W, R	0x0	Non-periodic TxFIFO Depth This value is in terms of 32-bit words. Minimum value is 16. Maximum value is 4160.
	INEPTxF0Dep	Device			IN Endpoint TxFIFO 0 Depth This value is in terms of 32-bit words. Minimum value is 16. Maximum value is 4160.
15-0	NPTxFStAddr	Host	R/W, R	0x0	Non-periodic Transmit RAM Start Address This field contains the memory start address for Non-periodic Transmit FIFO RAM.
	INEPTxF0StAddr	Device			IN Endpoint FIFO0 Transmit RAM Start Address This field contains the memory start address for IN Endpoint Transmit FIFO# 0.

Non-Periodic Transmit FIFO/Queue Status Register (GNPTXSTS) 0xB004002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	NPTxQTop							NPTxQSpcAvail							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NPTxFSpcAvail															

In Device mode, this register is invalid.

This read-only register contains the free space information for the Non-periodic TxFIFO and the Non-periodic Transmit Request Queue.

Field	Name	Mode	RW	Reset	Description
31	-	-	-	0x0	Reserved
30-24	NPTxQTop	Host	R	0x0	Top of the Non-periodic Transmit Request Queue Entry in the Non-periodic Tx Request Queue that is currently being processed by the MAC. • Bits [30:27]: Channel/endpoint number • Bits [26:25]: - 2'b00: IN/OUT token - 2'b01: Zero-length transmit packet (device IN/host OUT) - 2'b10: PING/CSPLIT token - 2'b11: Channel halt command • Bit [24]: Terminate (last entry for selected channel/endpoint)
23-16	NPTxQSpcAvail	Host	R	0x0	Non-periodic Transmit Request Queue Space Available Indicates the amount of free space available in the Non-periodic Transmit Request Queue. This queue holds both IN and OUT requests in Host mode. • 8'h0: Non-periodic Transmit Request Queue is full • 8'h1: 1 location available • 8'h2: 2 locations available • n: n locations available ($0 \le n \le 8$) • Others: Reserved
15-0	NPTxFSpcAvail	Host	R	0x0	Non-periodic TxFIFO Space Avail Indicates the amount of free space available in the Non-periodic TxFIFO. Values are in terms of 32-bit words. • 16'h0: Non-periodic TxFIFO is full • 16'h1: 1 word available • 16'h2: 2 words available • 16'hn: n words available (where $0 \le n \le$ 32,768) • 16'h8000: 32,768 words available • Others: Reserved

User ID Register (GUID) 0xB004003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UserID[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UserID[15:0]															

This is a read/write register containing the User ID.

This register can be used in the following ways:

- To store the version or revision of your system
- To store hardware configurations
- As a scratch register

Field	Name	Mode	RW	Reset	Description
31-0	UserID	Host Device	R	0x1234567	User ID Application-programmable ID field

User HW Config1 Register (GHWCFG1) 0xB0040044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epdir [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir [15:0]															

Field	Name	Mode	RW	Reset	Description
31-0	epdir	Host Device	R	0x0	Endpoint Direction Two bits per endpoint represent the direction. • 2’b00: BIDIR (IN and OUT) endpoint • 2’b01: IN endpoint • 2’b10: OUT endpoint • 2’b11: Reserved Bits [31:30]: Endpoint 15 direction Bits [29:28]: Endpoint 14 direction ... Bits [3:2]: Endpoint 1 direction Bits[1:0]: Endpoint 0 direction (always BIDIR)

User HW Config2 Register (GHWCFG2) 0xB0040048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TknQDepth					PTxQDepth		NPTxQDepth				DynFifo Sizing	PerioS upport	NumHstChnl[3:2]	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NumHstChnl[1:0]		NumDevEps				FSPhyType		HSPhyType		SingPnt	OtgArch				

Field	Name	RW	Reset	Description
31	-	-	0x0	Reserved
30-26	TknQDepth	R	0x8	Device Mode IN Token Sequence Learning Queue Depth: Range: 0‑30
25-24	PTxQDepth	R	0x2	Host Mode Periodic Request Queue Depth • 2’b00: 2 • 2’b01: 4 • 2’b10: 8 • Others: Reserved
23-22	NPTxQDepth	R	0x2	Non-periodic Request Queue Depth • 2’b00: 2 • 2’b01: 4 • 2’b10: 8 • Others: Reserved
21-20	-	-	0x0	Reserved
19	DynFifoSizing	R	0x1	Dynamic FIFO Sizing Enabled • 1’b0: No

				• 1’b1: Yes
18	PerioSupport	R	0x1	Periodic OUT Channels Supported in Host Mode • 1’b0: No • 1’b1: Yes
17-14	NumHstChnl	R	0xF	Number of Host Channels Indicates the number of host channels supported by the core in Host mode. The range of this field is 0-15: 0 specifies 1 channel, 15 specifies 16 channels.
13-10	NumDevEps	R	0xF	Number of Device Endpoints Indicates the number of device endpoints supported by the core in Device mode in addition to control endpoint 0. The range of this field is 1-15.
9-8	FSPhyType	R	0x0	Full-Speed PHY Interface Type • 2’b00: Full-speed interface not supported • 2’b01: Dedicated full-speed interface • 2’b10: FS pins shared with UTMI+ pins • 2’b11: FS pins shared with ULPI pins
7-6	HSPhyType	R	0x3	High-Speed PHY Interface Type • 2’b00: High-Speed interface not supported • 2’b01: UTMI+ • 2’b10: ULPI • 2’b11: UTMI+ and ULPI
5	SingPnt	R	0x1	Point-to-Point • 1’b0: Multi-point application • 1’b1: Single-point application
4-3	OtgArch	R	0x2	Architecture • 2’b00: Slave-Only • 2’b01: External DMA • 2’b10: Internal DMA
2-0	OtgMode	R	0x0	Mode of Operation • 3’b000: HNP- and SRP-Capable OTG (Host & Device) • 3’b001: SRP-Capable OTG (Host & Device) • 3’b010: Non-HNP and Non-SRP Capable OTG (Host & Device) • 3’b011: SRP-Capable Device • 3’b100: Non-OTG Device • 3’b101: SRP-Capable Host • 3’b110: Non-OTG Host • Others: Reserved

User HW Config3 Register (GHWCFG3) 0xB004004C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DfifoDepth															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				RstType	OptFeature			OtgEn	PktSizeWidth			XferSizeWidth			

Field	Name	RW	Reset	Description
31-16	DfifoDepth	R	0x8000	DFIFO Depth This value is in terms of 32-bit words. • Minimum value is 32 • Maximum value is 4160
15-12	-	-	0x0	Reserved
11	RstType	R	0x0	Reset Style for Clocked always Blocks in RTL • 1’b0: Asynchronous reset is used in the core • 1’b1: Synchronous reset is used in the core
10	OptFeature	R	0x0	Optional Features Removed Indicates whether the User ID register, GPIO interface ports, and SOF toggle and counter ports were removed for gate count optimization by enabling Remove Optional Features during coreConsultant configuration. • 1’b0: No • 1’b1: Yes
9-8	-	-	0x0	Reserved
7	OtgEn	R	0x1	OTG Function Enabled The application uses this bit to indicate core’s OTG capabilities. • 1’b0: Not OTG capable • 1’b1: OTG Capable
6-4	PktSizeWidth	R	0x6	Width of Packet Size Counters • 3’b000: 4 bits • 3’b001: 5 bits • 3’b010: 6 bits • 3’b011: 7 bits • 3’b100: 8 bits • 3’b101: 9 bits • 3’b110: 10 bits • Others: Reserved
3-0	XferSizeWidth	R	0x9	Width of Transfer Size Counters • 4’b0000: 11 bits • 4’b0001: 12 bits ... • 4’b1000: 19 bits • Others: Reserved

User HW Config4 Register (GHWCFG4) 0xB0040050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Scatter/Gather DMA	Scatter/Gather DMA configuration	INEps				DedFifoMode	SessEndFltr	BValidFltr	AValidFltr	VBusValidFltr	IddgFltr	NumCtlEps			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PhyDataWidth										AhbFreq	EnablePwrOpt	NumDevPerioEps			

Field	Name	RW	Reset	Description
31	Scatter/Gather DMA	R	0x0	•1'b0: Non Dynamic configuration • 1'b1: Dynamic configuration
30	Scatter/Gather DMA configuration	R	0x1	•1'b0: Non-Scatter/Gather DMA configuration • 1'b1: Scatter/Gather DMA configuration
29-26	INEps	R	0x7	Endpoints Range 0 -15 • 0 : 1 IN Endpoint • 1 : 2 IN Endpoints • .... 15 : 16 IN Endpoints
25	DedFifoMode	R	0x1	Enable Dedicated Transmit FIFO for device IN Endpoints • 1’ b0 : Dedicated Transmit FIFO Operation not enabled. • 1’ b1 : Dedicated Transmit FIFO Operation enabled.
24	SessEndFltr	R	0x1	“session_end” Filter Enabled () • 1’ b0: No filter • 1’ b1: Filter
23	BValidFltr	R	0x1	“b_valid” Filter Enabled • 1’ b0: No filter • 1’ b1: Filter
22	AValidFltr	R	0x1	“a_valid” Filter Enabled • 1’ b0: No filter • 1’ b1: Filter
21	VBusValidFltr	R	0x1	“vbus_valid” Filter Enabled • 1’ b0: No filter • 1’ b1: Filter
20	IddgFltr	R	0x1	“iddig” Filter Enable • 1’ b0: No filter • 1’ b1: Filter
19-16	NumCtlEps	R	0x0	Number of Device Mode Control Endpoints in Addition to Endpoint 0 Range: 0-15
15-14	PhyDataWidth	R	0x2	UTMI+ PHY/ULPI-to-Internal UTMI+ Wrapper Data Width When a ULPI PHY is used, an internal wrapper converts ULPI to UTMI+ . • 2’ b00: 8 bits • 2’ b01: 16 bits • 2’ b10: 8/16 bits, software selectable • Others: Reserved
13-6	-	-	0x0	Reserved
5	AhbFreq	R	0x1	Minimum AHB Frequency Less Than 60 MHz • 1’ b0: No • 1’ b1: Yes
4	EnablePwrOpt	R	0x1	Enable Power Optimization • 1’ b0: No • 1’ b1: Yes
3-0	NumDevPerioEps	R	0x0	Number of Device Mode Periodic IN Endpoints Range: 0‑15

Host Periodic Transmit FIFO Size Register (HPTXFSIZ) 0xB0040100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTxFSize															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTxFStAddr															

This register holds the size and the memory start address of the Periodic TxFIFO.

Field	Name	RW	Reset	Description
31-0	PTxFSize	R/W, R	0	Host Periodic TxFIFO Depth This value is in terms of 32-bit words. • Minimum value is 16 • Maximum value is 4160
15-0	PTxFStAddr	R/W, R	0	Host Periodic TxFIFO Start Address In shared FIFO operation : • OTG_RX_DFIFO_DEPTH + OTG_TX_HNPERIO_DFIFO_DEPTH In dedicated FIFO mode : • OTG_RX_DFIFO_DEPTH + OTG_TX_HNPERIO_DFIFO_DEPTH

Device IN Endpoint Transmit Fifo Size Register: (DIEPTXFn) 0xB0040104+(FIFO_number-1)*0x4

FIFO_number:1≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INEPnTxFDep															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INEPnTxFStAddr															

This register holds the size and the memory start address of IN endpoint TxFIFOs implemented in Device mode. Each FIFO holds the data for one IN endpoint. This register is repeated for instantiated IN endpoint FIFOs 1 to 15. For IN endpoint FIFO 0 use GNPTXFSIZ register for programming the size and memory start address.

Field	Name	RW	Reset	Description
31-0	INEPnTxFDep	R/W, R	0	IN Endpoint TxFIFO Depth This value is in terms of 32-bit words. Minimum value is 16. Maximum value is 4160.
15-0	INEPnTxFStAddr	R/W, R	0	IN Endpoint FIFOn Transmit RAM Start Address This field contains the memory start address for IN endpoint Transmit FIFOn (0<n< = 15). OTG_RX_DFIFO_DEPTH + SUM 0 to n – 1 (OTG_DINEP_TXFIFO_DEPTH_n) For example start address of IN endpoint FIFO 1 is OTG_RX_DFIFO_DEPTH + OTG_DINEP_TXFIFO_DEPTH_0 The start address of IN endpoint FIFO 2 is OTG_RX_DFIFO_DEPTH + OTG_DINEP_TXFIFO_DEPTH_0 + OTG_DINEP_TXFIFO_DEPTH_1

Host Configuration Register (HCFG) 0xB0040400

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													FSLSSupp	FSLSPclkSel	

This register configures the core after power-on. Do not make changes to this register after initializing the host.

Field	Name	RW	Reset	Description
31-3	-	-	0	Reserved
2	FSLSSupp	R/W	0	FS- and LS-Only Support The application uses this bit to control the core's enumeration speed. Using this bit, the application can make the core enumerate as a FS host, even if the connected device supports HS traffic. Do not make changes to this field after initial programming. • 1'b0: HS/FS/LS, based on the maximum speed supported by the connected device • 1'b1: FS/LS-only, even if the connected device can support HS
1-0	FSLSPclkSel	R/W	0	FS/LS PHY Clock Select When the core is in FS Host mode • 2'b00: PHY clock is running at 30/60 MHz • 2'b01: PHY clock is running at 48 MHz • Others: Reserved When the core is in LS Host mode • 2'b00: PHY clock is running at 30/60 MHz. When the UTMI+/ULPI PHY Low Power mode is not selected, use 30/60 MHz. • 2'b01: PHY clock is running at 48 MHz. When the UTMI+ PHY Low Power mode is selected, use 48MHz if the PHY supplies a 48 MHz clock during LS mode. • 2'b10: PHY clock is running at 6 MHz. In USB 1.1 FS mode, use 6 MHz when the UTMI+ PHY Low Power mode is selected and the PHY supplies a 6 MHz clock during LS mode. If you select a 6 MHz clock during LS mode, you must do a soft reset. • 2'b11: Reserved

Host Frame Interval Register (HFIR) 0xB0040404

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FrInt															

This register stores the frame interval information for the current speed to which the otg core has enumerated.

Field	Name	RW	Reset	Description
31-16	-	-	0	Reserved
15-0	FrInt	R/W	60000 (decimal)	Frame Interval The value that the application programs to this field specifies the interval between two consecutive SOFs (FS) or micro-SOFs (HS) or Keep-Alive tokens (HS). This field contains the number of PHY clocks that constitute the required frame interval. The default value set in this field for a FS operation when the PHY clock frequency is 60 MHz. The application can write a value to this register only after the Port Enable bit of the Host Port Control and Status register (HPRT.PrtEnaPort) has been set. If no value is programmed, the core calculates the value based on the PHY clock specified in the FS/LS PHY Clock Select field of the Host Configuration register (HCFG.FSLSPclkSel). Do not change the value of this field after the initial configuration. • 125 μs * (PHY clock frequency for HS) • 1 ms * (PHY clock frequency for FS/LS)

Host Frame Number/Frame Time Remaining Register (HFNUM) 0xB0040408

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FrRem															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FrNum															

This register indicates the current frame number. It also indicates the time remaining (interms of the number of PHY clocks) in the current (micro)frame.

Field	Name	RW	Reset	Description
31-16	FrRem	R	0x0	Frame Time Remaining Indicates the amount of time remaining in the current microframe (HS) or frame (FS/LS), in terms of PHY clocks. This field decrements on each PHY clock. When it reaches zero, this field is reloaded with the value in the Frame Interval register and a new SOF is transmitted on the USB.
15-0	FrNum	R/W	0x3FFF	Frame Number This field increments when a new SOF is transmitted on the USB, and is reset to 0 when it reaches 16’h3FFF.

Host Periodic Transmit FIFO/Queue Status Register (HPTXSTS) 0xB0040410

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTxQTop								PTxQSpcAvail							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTxFSpcAvail															

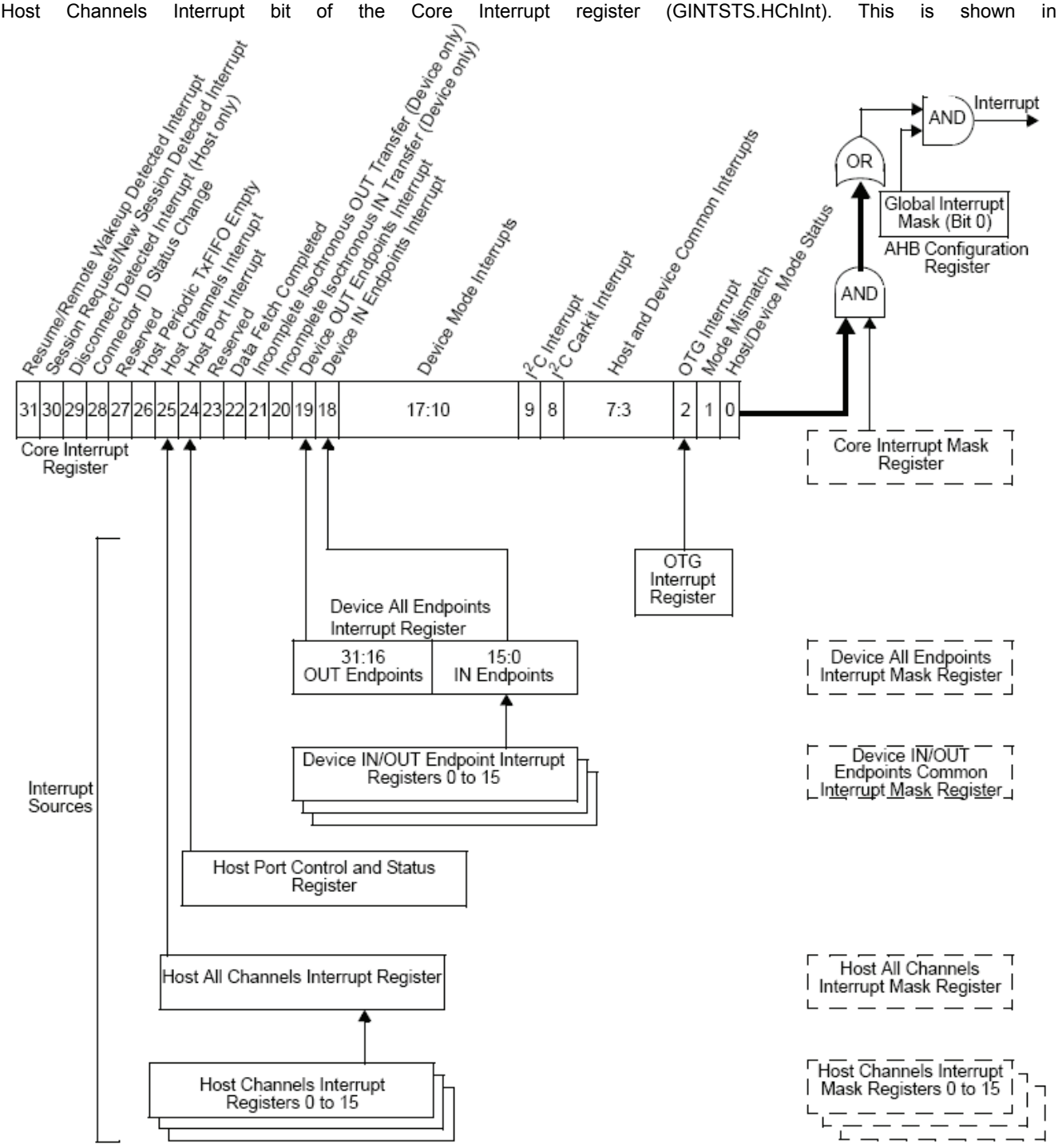
This read-only register contains the free space information for the Periodic TxFIFO and the Periodic Transmit Request Queue.

Field	Name	RW	Reset	Description
31-24	PTxQTop	R	0x0	Top of the Periodic Transmit Request Queue This indicates the entry in the Periodic Tx Request Queue that is currently being processes by the MAC. This register is used for debugging. • Bit [31]: Odd/Even (micro)frame - 1’b0: send in even (micro)frame - 1’b1: send in odd (micro)frame • Bits [30:27]: Channel/endpoint number • Bits [26:25]: Type - 2’b00: IN/OUT - 2’b01: Zero-length packet - 2’b10: CSPLIT - 2’b11: Disable channel command • Bit [24]: Terminate (last entry for the selected channel/endpoint)
23-16	PTxQSpcAvail	R	0x0	Periodic Transmit Request Queue Space Available Indicates the number of free locations available to be written in the Periodic Transmit Request Queue. This queue holds both IN and OUT requests. • 8’h0: Periodic Transmit Request Queue is full • 8’h1: 1 location available • 8’h2: 2 locations available • n: n locations available (0≤n≤8) • Others: Reserved
15-0	PTxFSpcAvail	R/W	0x0	Periodic Transmit Data FIFO Space Available Indicates the number of free locations available to be written to in the Periodic TxFIFO. Values are in terms of 32-bit words • 16’h0: Periodic TxFIFO is full • 16’h1: 1 word available • 16’h2: 2 words available • 16’hn: n words available (where 0≤n≤ 32,768) • 16’h8000: 32,768 words available • Others: Reserved

Host All Channels Interrupt Register (HAINT) **0xB0040414**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HAINT															

When a significant event occurs on a channel, the Host All Channels Interrupt register interrupts the application using the Host Channels Interrupt bit of the Core Interrupt register (GINTSTS.HChInt). This is shown in



Note: Because an interrupt mask only masks an interrupt, software must clear an interrupt before unmasking it, to avoid servicing an old interrupt.

Figure 13.3. There is one interrupt bit per channel, up to a maximum of 16 bits. Bits in this register are set and cleared when the application sets and clears bits in the corresponding Host Channel-n Interrupt register.

Field	Name	RW	Reset	Description
31-16	-	-	0x0	Reserved
15-0	HAINT	R	0x0	Channel Interrupts One bit per channel: Bit 0 for Channel 0, bit 15 for Channel 15

Host All Channels Interrupt Mask Register (HAINTMSK) 0xB0040418

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HAINTMsk															

The Host All Channel Interrupt Mask register works with the Host All Channel Interrupt register to interrupt the application when an event occurs on a channel. There is one interrupt mask bit per channel, up to a maximum of 16 bits.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

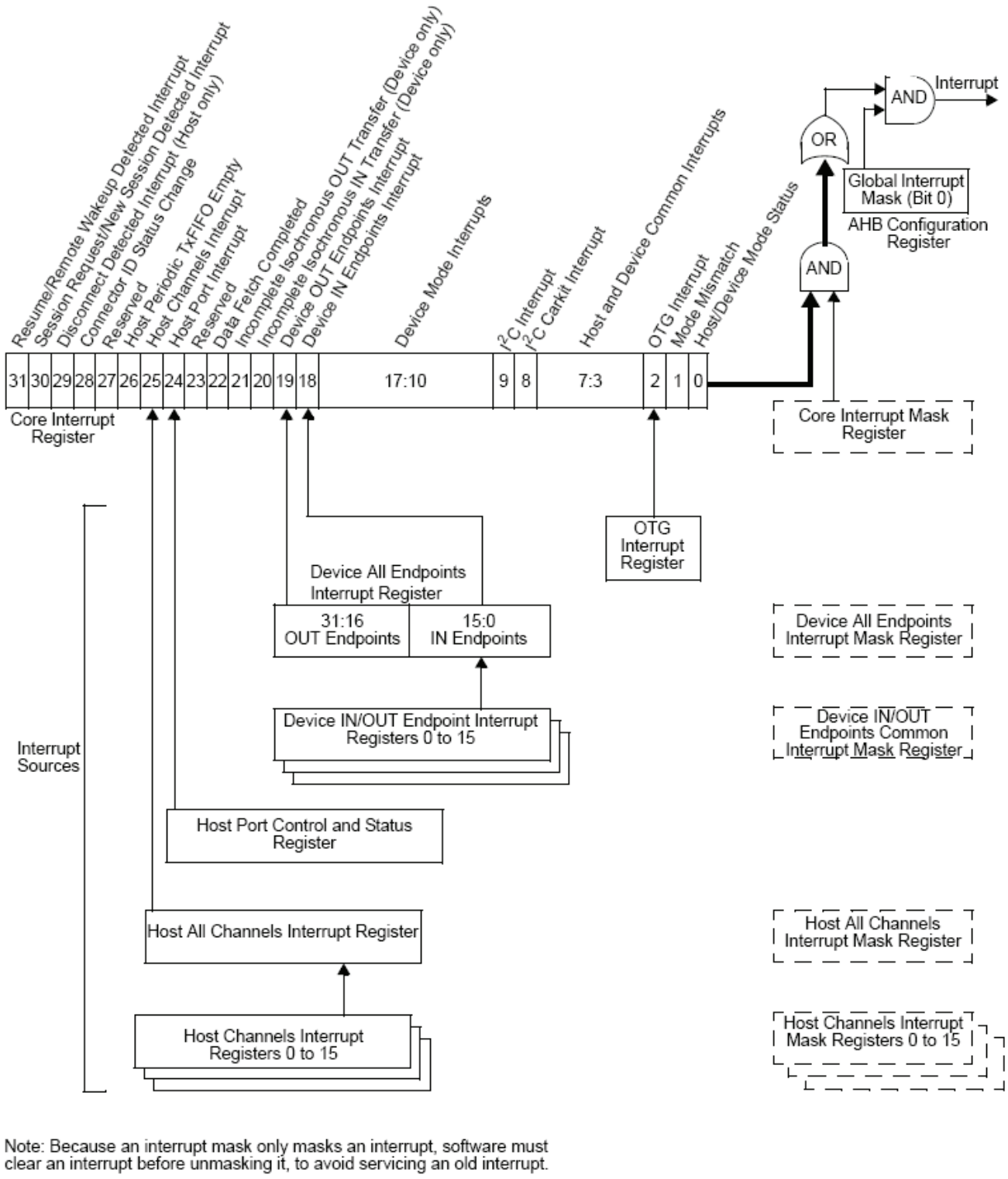
Field	Name	RW	Reset	Description
31-16	-	-	0x0	Reserved
15-0	HAINTMsk	R/W	0x0	Channel Interrupts Mask One bit per channel: Bit 0 for Channel 0, bit 15 for Channel 15

Host Port Control and Status Register (HPRT) 0xB0040440

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
													PrtSpd		PrtTstCtl[3]
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PrtTstCtl[2:0]			PrtPwr	PrtLnSts			PrtRst	PrtSusp	PrtRes	PrtOvrCurrChng	PrtOvrCurrAct	PrtEnChng	PrtEna	PrtConnDet	PrtConnSts

This register is available only in Host mode. Currently, the OTG Host supports only one port.

A single register holds USB port-related information such as USB reset, enable, suspend, resume, connect status, and test mode for each port. It is shown in



Note: Because an interrupt mask only masks an interrupt, software must clear an interrupt before unmasking it, to avoid servicing an old interrupt.

Figure 13.3. The PrtOvrCurrChng and PrtEnChng in this register can trigger an interrupt to the application through the Host Port Interrupt bit of Core Interrupt register (GINTSTS.PrtInt). On a Port Interrupt, the application must read this register and clear the bit that caused the interrupt.

Field	Name	RW	Reset	Description
31-19	-	-	0x0	Reserved
18-17	PrtSpd	R	0x0	Port Speed () Indicates the speed of the device attached to this port. • 2’ b00: High speed • 2’ b01: Full speed • 2’ b10: Low speed

				• 2’b11: Reserved
16-13	PrtTstCtl	R/W	0x0	Port Test Control The application writes a nonzero value to this field to put the port into a Test mode, and the corresponding pattern is signaled on the port. • 4’b0000: Test mode disabled • 4’b0001: Test_J mode • 4’b0010: Test_K mode • 4’b0011: Test_SE0_NAK mode • 4’b0100: Test_Packet mode • 4’b0101: Test_Force_Enable • Others: Reserved
12	PrtPwr	R/W	0x0	Port Power The application uses this field to control power to this port, and the core clears this bit on an overcurrent condition. • 1’b0: Power off • 1’b1: Power on
11-10	PrtLnSts	R	0x0	Port Line Status Indicates the current logic level USB data lines • Bit [10]: Logic level of D- • Bit [11]: Logic level of D +
9	-	-	0x0	Reserved
8	PrtRst	R/W	0x0	Port Reset When the application sets this bit, a reset sequence is started on this port. The application must time the reset period and clear this bit after the reset sequence is complete. • 1’b0: Port not in reset • 1’b1: Port in reset The application must leave this bit set for at least a minimum duration mentioned below to start a reset on the port. The application can leave it set for another 10 ms in addition to the required minimum duration, before clearing the bit, even though there is no maximum limit set by the USB standard. • High speed: 50 ms • Full speed/Low speed: 10 ms
7	PrtSusp	R/W	0x0	Port Suspend The application sets this bit to put this port in Suspend mode. The core only stops sending SOFs when this is set. To stop the PHY clock, the application must set the Port Clock Stop bit, which asserts the suspend input pin of the PHY. The read value of this bit reflects the current suspend status of the port. This bit is cleared by the core after a remote wakeup signal is detected or the application sets the Port Reset bit or Port Resume bit in this register or the Resume/Remote Wakeup Detected Interrupt bit or Disconnect Detected Interrupt bit in the Core Interrupt register (GINTSTS.WkUpInt or GINTSTS.DisconnInt, respectively). • 1’b0: Port not in Suspend mode • 1’b1: Port in Suspend mode
6	PrtRes	R/W	0x0	Port Resume The application sets this bit to drive resume signaling on the port. The core continues to drive the resume signal until the application clears this bit. If the core detects a USB remote wakeup sequence, as indicated by the Port Resume/Remote Wakeup Detected Interrupt bit of the Core Interrupt register (GINTSTS.WkUpInt), the core starts driving resume signaling without application intervention and clears this bit when it detects a disconnect condition. The read value of this bit indicates whether the core is currently driving resume signaling. • 1’b0: No resume driven • 1’b1: Resume driven
5	PrtOvrCurrChng	R/W	0x0	Port Overcurrent Change The core sets this bit when the status of the Port Overcurrent Active bit (bit 4) in this register changes. The application must write a 1 to the bit to clear the interrupt
4	PrtOvrCurrAct	R	0x0	Port Overcurrent Active Indicates the overcurrent condition of the port. • 1’b0: No overcurrent condition

				• 1’b1: Overcurrent condition
3	PrtEnChng	R/W	0x0	Port Enable/Disable Change The core sets this bit when the status of the Port Enable bit [2] of this register changes. The application must write a 1 to the bit to clear the interrupt
2	PrtEna	R/W	0x0	Port Enable A port is enabled only by the core after a reset sequence, and is disabled by an overcurrent condition, a disconnect condition, or by the application clearing this bit. The application cannot set this bit by a register write. It can only clear it to disable the port. This bit does not trigger any interrupt to the application. • 1’b0: Port disabled • 1’b1: Port enabled
1	PrtConnDet	R/W	0x0	Port Connect Detected The core sets this bit when a device connection is detected to trigger an interrupt to the application using the Host Port Interrupt bit of the Core Interrupt register (GINTSTS.PrtInt). The application must write a 1 to this bit to clear the interrupt.
0	PrtConnSts	R	0x0	Port Connect Status • 0: No device is attached to the port. • 1: A device is attached to the port.

Host Channel-n Characteristics Register (HCCHARn) **0xB0040500+(Channel_number*0x20)**
Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ChEna	ChDis	OddFrm	DevAddr							MC/ EC		EPType		LSpdDev	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPDir	EPNum				MPS										

Field	Name	RW	Reset	Description
31	ChEna	R/W	0x0	Channel Enable This field is set by the application and cleared by the OTG host. • 1’b0: Channel disabled • 1’b1: Channel enabled
30	ChDis	R/W	0x0	Channel Disable The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer for that channel is complete. The application must wait for the Channel Disabled interrupt before treating the channel as disabled.
29	OddFrm	R/W	0x0	Odd Frame This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. • 1’b0: Even (micro)frame • 1’b1: Odd (micro)frame
28-22	DevAddr	R/W	0x0	Device Address This field selects the specific device serving as the data source or sink.
21-20	MC/ EC	R/W	0x0	Multi Count / Error Count When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1’b0), this field indicates to the host the number of transactions that must be executed per microframe for this endpoint. • 2’b00: Reserved This field yields undefined results. • 2’b01: 1 transaction • 2’b10: 2 transactions to be issued for this endpoint per microframe • 2’b11: 3 transactions to be issued for this endpoint per microframe When HCSPLTn.SpltEna is set (1’b1), this field indicates the number of immediate retries to be performed for a periodic split transactions on transaction errors. This field must be set to at least 2’b01.
19-18	EPType	R/W	0x0	Endpoint Type Indicates the transfer type selected. • 2’b00: Control • 2’b01: Isochronous • 2’b10: Bulk • 2’b11: Interrupt
17	LSpdDev	R/W	0x0	Low-Speed Device This field is set by the application to indicate that this channel is communicating to a low-speed device.
16	-	-	0x0	Reserved
15	EPDir	R/W	0x0	Endpoint Direction Indicates whether the transaction is IN or OUT. • 1’b0: OUT • 1’b1: IN
14-11	EPNum	R/W	0x0	Endpoint Number Indicates the endpoint number on the device serving as the data source or sink.
10-0	MPS	R/W	0x0	Maximum Packet Size Indicates the maximum packet size of the associated endpoint.

Host Channel-n Split Control Register (HCSPLTn) **0xB0040504+(Channel_number*0x20)**

Channel_number:0≤n≤15

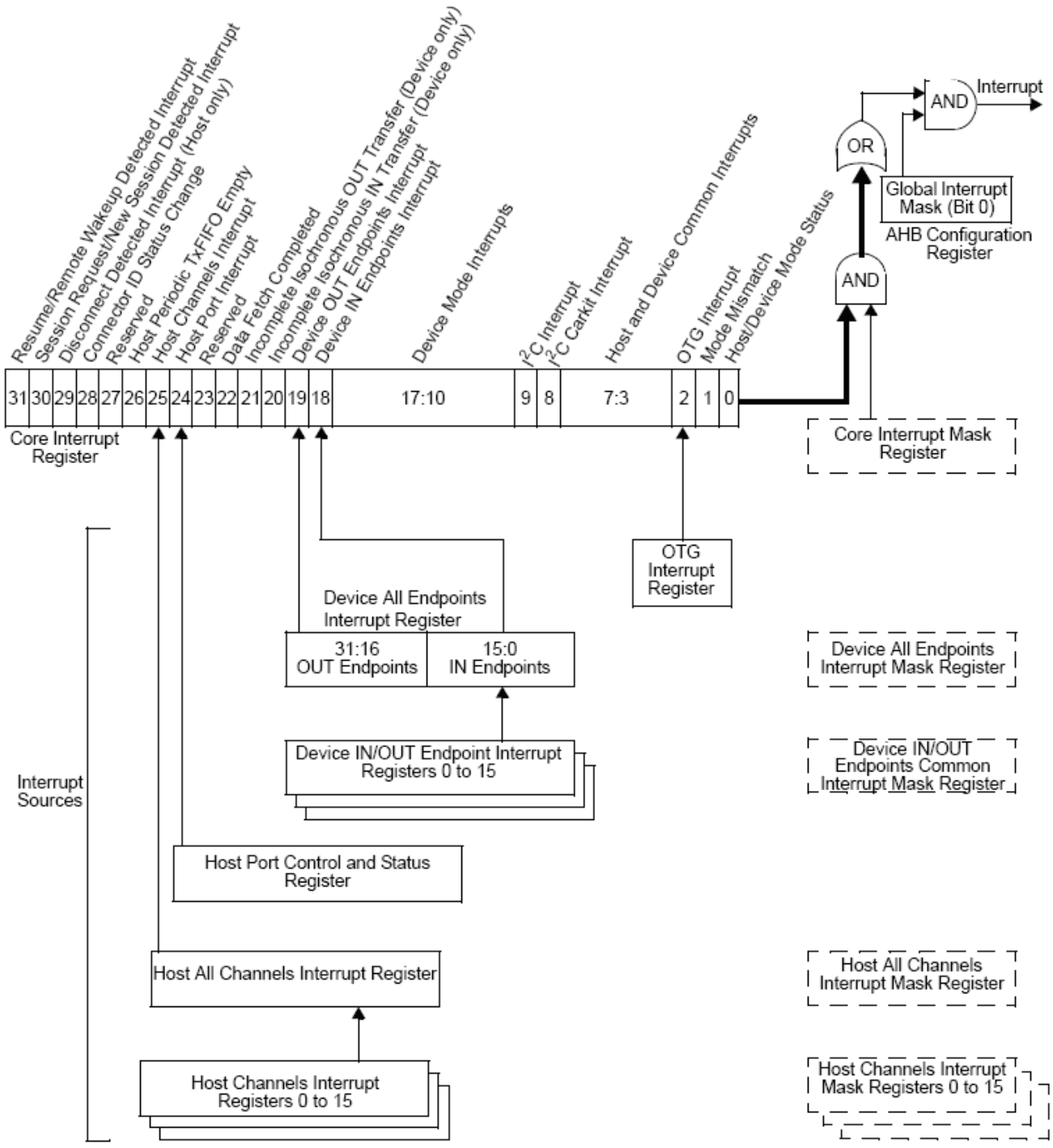
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SpltEna															CompSplt
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
XactPos		HubAddr							PrtAddr						

Field	Name	RW	Reset	Description
31	SpltEna	R/W	0x0	Split Enable The application sets this field to indicate that this channel is enabled to perform split transactions.
30-17	-	-	0x0	Reserved
16	CompSplt	R/W	0x0	Do Complete Split The application sets this field to request the OTG host to perform a complete split transaction.
15-14	XactPos	R/W	0x0	Transaction Position This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. • 2’b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). • 2’b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). • 2’b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). • 2’b01: End. This is the last payload of this transaction (which is larger than 188 bytes).
13-7	HubAddr	R/W	0x0	Hub Address This field holds the device address of the transaction translator’s hub.
6-0	PrtAddr	R/W	0x0	Port Address This field is the port number of the recipient transaction translator.

Host Channel-n Interrupt Register (HCINTn) **0xB0040508+(Channel_number*0x20)**
Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					DataTglErr	FrmOvrun	BblErr	XactErr	NYET	ACK	NAK	STALL	AHBErr	ChHltd	XferCompl

This register indicates the status of a channel with respect to USB- and AHB-related events. It is shown in



Note: Because an interrupt mask only masks an interrupt, software must clear an interrupt before unmasking it, to avoid servicing an old interrupt.

Figure 13.3. The application must read this register when the Host Channels Interrupt bit of the Core Interrupt register (GINTSTS.HChInt) is set. Before the application can read this register, it must first read the Host All Channels Interrupt (HAINT) register to get the exact channel number for the Host Channel-n Interrupt register. The application must clear the appropriate bit in this register to clear the corresponding bits in the HAINT and GINTSTS registers.

Field	Name	RW	Reset	Description
31-11	-	-	0x0	Reserved
10	DataTglErr	R/W	0x0	Data Toggle Error
9	FrmOvrun	R/W	0x0	Frame Overrun
8	BblErr	R/W	0x0	Babble Error
7	XactErr	R/W	0x0	Transaction Error Indicates one of the following errors occurred on the USB. • CRC check failure • Timeout • Bit stuff error • False EOP
6	NYET	R/W	0x0	NYET Response Received Interrupt
5	ACK	R/W	0x0	ACK Response Received/Transmitted Interrupt
4	NAK	R/W	0x0	NAK Response Received Interrupt
3	STALL	R/W	0x0	STALL Response Received Interrupt
2	AHBErr	R/W	0x0	AHB Error This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel’s DMA address register to get the error address.
1	ChHltd	R/W	0x0	Channel Halted Indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application.
0	XferCompl	R/W	0x0	Transfer Completed Transfer completed normally without any errors.

Host Channel-n Interrupt Mask Register (HCINTMSKn) 0xB004050C+(Channel_number*0x20)

Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					DataTglErrMsk	FrmOvrunMsk	BblErrMsk	XactErrMsk	NyetMsk	AckMsk	NakMsk	StallMsk	AHBErrMsk	ChHltdMsk	XferComplMsk

This register reflects the mask for each channel status described in the previous section.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-11	-	-	0x0	Reserved
10	DataTglErrMsk	R/W	0x0	Data Toggle Error Mask
9	FrmOvrunMsk	R/W	0x0	Frame Overrun Mask
8	BblErrMsk	R/W	0x0	Babble Error Mask
7	XactErrMsk	R/W	0x0	Transaction Error Mask
6	NyetMsk	R/W	0x0	NYET Response Received Interrupt Mask
5	AckMsk	R/W	0x0	ACK Response Received/Transmitted Interrupt Mask
4	NakMsk	R/W	0x0	NAK Response Received Interrupt Mask
3	StallMsk	R/W	0x0	STALL Response Received Interrupt Mask
2	AHBErrMsk	R/W	0x0	AHB Error Mask
1	ChHltdMsk	R/W	0x0	Channel Halted Mask
0	XferComplMsk	R/W	0x0	Transfer Completed Mask

Host Channel-n Transfer Size Register (HCTSIZn) 0xB0040510+(Channel_number*0x20)

Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DoPng	Pid		PktCnt										XferSize[18:16]		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
XferSize[15:0]															

Field	Name	RW	Reset	Description
31	DoPng	R/W	0x0	Do Ping () Setting this field to 1 directs the host to do PING protocol.
30-29	Pid	R/W	0x0	PID The application programs this field with the type of PID to use for the initial transaction. The host maintains this field for the rest of the transfer. • 2'b00: DATA0 • 2'b01: DATA2 • 2'b10: DATA1 • 2'b11: MDATA (non-control)/SETUP (control)
28-19	PktCnt	R/W	0x0	Packet Count This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decreme ts this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion.
18-0	XferSize	R/W	0x0	Transfer Size For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved for the transfer. The application is expected to program this field as an integer multiple of the maximum packet size for IN transactions (periodic and non-periodic).

Host Channel-n DMA Address Register (HCDMAn)

0xB0040514+(Channel_number*0x20)
Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAAddr [31:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAAddr [15:0]															

This register is used by the OTG host in the internal DMA mode to maintain the current buffer pointer for IN/OUT transactions. The starting DMA address must be DWORD-aligned.

Field	Name	RW	Reset	Description
31-0	DMAAddr	R/W	0x0	DMA Address This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transaction.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ResValid						PerSchIntvl		Desc DMA	EPMisCnt						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			PerFrInt		DevAddr							Ena32 KHzS	NZSts OUTH Shk	DevSpd	

This register configures the core in Device mode after power-on or after certain control commands or enumeration. Do not make changes to this register after initial programming.

Field	Name	Mode	Reset	RW	Description
31-26	ResValid	Device	6'd2	R/W	Resume Validation Period This field controls the period when the core resumes from a suspend. When this bit is set, the core counts for the ResValid number of clock cycles to detect a valid resume. This field is effective only when DCFG.Ena32KHzSusp is set.
25-24	PerSchIntvl	Device	2'b0	R/W	Periodic Scheduling Interval PerSchIntvl must be programmed only for Scatter/Gather DMA mode. This field specifies the amount of time the Internal DMA engine must allocate for fetching periodic IN endpoint data. Based on the number of periodic endpoints, this value must be specified as 25,50 or 75% of (micro)frame. • When any periodic endpoints are active, the internal DMA engine allocates the specified amount of time in fetching periodic IN endpoint data. • When no periodic endpoints are active, then the internal DMA engine services non-periodic endpoints, ignoring this field. • After the specified time within a (micro)frame, the DMA switches to fetching for non-periodic endpoints. • 2'b00: 25% of (micro)frame. • 2'b01: 50% of (micro)frame. • 2'b10: 75% of (micro)frame. • 2'b11: Reserved.
23	DescDMA	Device	1'b0	R/W	Enable Scatter/Gather DMA in Device mode When the Scatter/Gather DMA option selected during configuration of the RTL, the application can set this bit during initialization to enable the Scatter/Gather DMA operation. •GAHBCFG.DMAEn=0,DCFG.DescDMA=0 => Slave mode •GAHBCFG.DMAEn=0,DCFG.DescDMA=1 => Invalid

					•GAHBCFG.DMAEn=1,DCFG.DescDMA=0 => Buffered DMA mode •GAHBCFG.DMAEn=1,DCFG.DescDMA=1 => Scatter/Gather DMA mode
22-18	EPMisCnt	Device	0	R/W	IN Endpoint Mismatch Count This field is valid only in shared FIFO operation. The application programs this filed with a count that determines when the core generates an Endpoint Mismatch interrupt (GINTSTS.EPMis). The core loads this value into an internal counter and decrements it. The counter is reloaded whenever there is a match or when the counter expires. The width of this counter depends on the depth of the Token Queue.
17-13	-		0		Reserved
12-11	PerFrInt	Device	0	R/W	Periodic Frame Interval Indicates the time within a (micro)frame at which the application must be notified using the End Of Periodic Frame Interrupt. This can be used to determine if all the isochronous traffic for that (micro)frame is complete. • 2'b00: 80% of the (micro)frame interval • 2'b01: 85% • 2'b10: 90% • 2'b11: 95%
10-4	DevAddr	Device	0	R/W	Device Address The application must program this field after every SetAddress control command.
3	Ena32KHzS	Device	0	R/W	Enable 32-KHz Suspend Mode When the FS PHY interface is chosen and this bit is set, the core expects the 48-KHz PHY to be switched to 32 KHz during a suspend. This bit can only be set if FS PHY interface has been selected. If FS PHY has not been selected, this bit must be zero.
2	NZStsOUTHShk	Device	0	R/W	Non-Zero-Length Status OUT Handshake The application can use this field to select the handshake the core sends on receiving a nonzero-length data packet during the OUT transaction of a control transfer's Status stage. • 1'b1: Send a STALL handshake on a nonzero-length status OUT transaction and do not send the received OUT packet to the application. • 1'b0: Send the received OUT packet to the application (zerolength or nonzero-length) and send a handshake based on the NAK and STALL bits for the endpoint in the Device Endpoint Control register.
1-0	DevSpd	Device	0	R/W	Device Speed Indicates the speed at which the application requires the core to enumerate, or the maximum speed the application can support. However, the actual bus speed is determined only after the chirp sequence is completed, and is based on the speed of the USB host to which the core is connected. See "Device Initialization" on page 210 for details. • 2'b00: High speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) • 2'b01: Full speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) • 2'b10: Low speed (USB 1.1 transceiver

					clock is 6 MHz). If you select 6 MHz LS mode, you must do a soft reset. • 2'b11: Full speed (USB 1.1 transceiver clock is 48 MHz)

Device Control Register (DCTL) **0xB0040804**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
															NakOnBble
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**		**2**	**1**	**0**
IgnrFrmNum	GMC			PWRONPrgDone	CGOUTNak	SGOUTNak	CGNPInNak	SGNPInNak	TstCtl			GOUTNakSts	GNPINNakSts	SftDiscon	RmtWkUpSig

Field	Name	Mode	Reset	RW	Description
31-17	-		0		Reserved
16	NakOnBble	Device	0	R/W	Set NAK automatically on babble. The core sets NAK automatically for the endpoint on which babble is received.
15	IgnrFrmNum	Device	0	R/W	Ignore frame number for isochronous endpoints in case of Scatter/Gather DMA. Do NOT program IgnrFrmNum bit to 1'b1 when the core is operating in Threshold mode. Note: When Scatter/Gather DMA mode is enabled this feature is not applicable to high-speed, high-bandwidth transfers. When this bit is enabled, there must be only one packet per descriptor. • 0: The core transmits the packets only in the frame number in which they are intended to be transmitted. • 1: The core ignores the frame number, sending packets immediately as the packets are ready. Scatter/Gather: • In Scatter/Gather DMA mode, when this bit I s enabled, the packets are not flushed when an ISOC IN token is received for an elapsed frame. • When Scatter/Gather DMA mode is disabled, this field is used by the application to enable periodic transfer interrupt. The application can program periodic endpoint transfers for multiple (micro)frames. - 0: Periodic transfer interrupt feature is d isabled; the application must program transfers for periodic endpoints every (micro)frame - 1: Periodic transfer interrupt feature is enabled; the application can program transfers for multiple (micro)frames for periodic endpoints. In non-Scatter/Gather DMA mode, the application receives transfer complete interrupt after transfers for multiple (micro)frames are completed.
14-13	GMC	Device	1	R//W	Global Multi Count GMC must be programmed only once after initialization. Applicable only for Scatter/Gather DMA mode. This indicates the number of packets to be serviced for that end point before moving to the next end point. It is only for non-periodic end points. • 2'b00: Invalid. • 2'b01: 1 packet. • 2'b10: 2 packets. • 2'b11: 3 packets. When Scatter/Gather DMA mode is disabled, this field is reserved. and

					reads 2'b00.
12	Reserved	-	-	-	-
11	PWROnPrgDone	Device	0	R/W	Power-On Programming Done The application uses this bit to indicate that register programming is completed after a wake-up from Power Down Mode.
10	CGOUTNak	Device	0	W	Clear Global OUT NAK A write to this field clears the Global OUT NAK.
9	SGOUTNak	Device	0	W	Set Global OUT NAK A write to this field sets the Global OUT NAK. The application uses this bit to send a NAK handshake on all OUT endpoints. The application must set the this bit only after making sure that the Global OUT NAK Effective bit in the Core Interrupt Register (GINTSTS.GOUTNakEff) is cleared.
8	CGNPInNak	Device	0	W	Clear Global Non-periodic IN NAK A write to this field clears the Global Non-periodic IN NAK.
7	SGNPInNak	Device	0	W	Set Global Non-periodic IN NAK A write to this field sets the Global Non-periodic IN NAK.The application uses this bit to send a NAK handshake on all nonperiodic IN endpoints. The application must set this bit only after making sure that the Global IN NAK Effective bit in the Core Interrupt Register (GINTSTS.GINNakEff) is cleared.
6-4	TstCtl	Device	0	R/W	Test Control • 3'b000: Test mode disabled • 3'b001: Test_J mode • 3'b010: Test_K mode • 3'b011: Test_SE0_NAK mode • 3'b100: Test_Packet mode • 3'b101: Test_Force_Enable • Others: Reserved
3	GOUTNakSts	Device	0	R	Global OUT NAK Status • 1'b0: A handshake is sent based on the FIFO Status and the NAK and STALL bit settings. • 1'b1: No data is written to the RxFIFO, irrespective of space availability. Sends a NAK handshake on all packets, except on SETUP transactions. All isochronous OUT packets are dropped.
2	GNPINNakSts	Device	0	R	Global Non-periodic IN NAK Status • 1'b0: A handshake is sent out based on the data availability in the transmit FIFO. • 1'b1: A NAK handshake is sent out on all non-periodic IN endpoints, irrespective of the data availability in the transmit FIFO.
1	SftDiscon	Device	0	R/W	Soft Disconnect The application uses this bit to signal the otg core to do a soft disconnect. As long as this bit is set, the host does not see that the device is connected, and the device does not receive signals on the USB. The core stays in the disconnected state until the application clears this bit. The minimum duration for which the core must keep this bit set is specified in Table below. • 1'b0: Normal operation. When this bit is cleared after a soft disconnect, the core drives the phy_opmode_o signal on the

					UTMI+ to 2'b00, which generates a device connect event to the USB host. When the device is reconnected, the USB host restarts device enumeration. • 1'b1: The core drives the phy_opmode_o signal on the UTMI+ to 2'b01, which generates a device disconnect event to the USB host.
0	RmtWkUpSig	Device	0	R/W	Remote Wakeup Signaling When the application sets this bit, the core initiates remote signaling to wake up the USB host. The application must set this bit to instruct the core to exit the Suspend state. As specified in the USB 2.0 specification, the application must clear this bit 1–15 ms after setting it.

Table below lists the minimum duration under various conditions for which the SoftDisconnect (SftDiscon) bit must be set for the USB host to detect a device disconnect. To accommodate clock jitter, it is recommended that the application add some extra delay to the specified minimum duration.

Table 13.3 Minimum Duration for Soft Disconnect

Operating Speed	Device Status	Minimum Duration
High Speed	Suspended	1 ms + 2.5 us
High Speed	Idle	3 ms + 2.5 us
High Speed	Not Idle or Suspended (Performing transactions)	125 us
Full speed/Low speed	Suspended	1 ms + 2.5 us
Full speed/Low speed	Idle	2.5 us
Full speed/Low speed	Not Idle or Suspended (Performing transactions)	2.5 us

Device Status Register (DSTS) 0xB0040808

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
										SOFFN[13:8]					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOFFN[7:0]												ErrticErr	EnumSpd		SuspSts

This register indicates the status of the core with respect to USB-related events. It must be read on interrupts from Device All Interrupts (DAINT) register.

Field	Name	RW	Reset	Description
31-22	-	-	0x0	Reserved
21-8	SOFFN	R	0x0	Frame or Microframe Number of the Received SOF When the core is operating at high speed, this field contains a microframe number. When the core is operating at full or low speed, this field contains a frame number.
7-4	-	-	0x0	Reserved
3	ErrticErr	R	0x0	Erratic Error The core sets this bit to report any erratic errors (phy_rxvalid_i/phy_rxvldh_i or phy_rxactive_i is asserted for at least 2 ms, due to PHY error) seen on the UTMI+ . Due to erratic errors, the OTG core goes into Suspended state and an interrupt is generated to the application with Early Suspend bit of the Core Interrupt register (GINTSTS.ErlySusp). If the early suspend is asserted due to an erratic error, the application can only perform a soft disconnect recover.
2-1	EnumSpd	R	0x1	Enumerated Speed Indicates the speed at which the OTG core has come up after speed detection through a chirp sequence. • 2’b00: High speed (PHY clock is running at 30 or 60 MHz) • 2’b01: Full speed (PHY clock is running at 30 or 60 MHz) • 2’b10: Low speed (PHY clock is running at 6 MHz) • 2’b11: Full speed (PHY clock is running at 48 MHz) Low speed is not supported for devices using a UTMI+ PHY.
0	SuspSts	R	0x0	Suspend Status In Device mode, this bit is set as long as a Suspend condition is detected on the USB. The core enters the Suspended state when there is no activity on the phy_line_state_i signal for an extended period of time. The core comes out of the suspend: • When there is any activity on the phy_line_state_i signal • When the application writes to the Remote Wakeup Signaling bit in the Device Control register (DCTL.RmtWkUpSig).

Device IN Endpoint Common Interrupt Mask Register (DIEPMSK) 0xB0040810

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							TxfifoUndrnMsk		INEPNakEffMsk	INTknEPMisMsk	INTknTXFEmpMsk	TimeOUTMsk	AHBErrMsk	EPDisbldMsk	XferComplMsk

This register works with each of the Device IN Endpoint Interrupt (DIEPINTn) registers for all endpoints to generate an interrupt per IN endpoint. The IN endpoint interrupt for a specific status in the DIEPINTn register can be masked by writing to the corresponding bit in this register. Status bits are masked by default.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-9	-	-	0x0	Reserved
8	TxfifoUndrnMsk	R/W	0x0	Fifo Underrun Mask
7	-	-	0x0	Reserved
6	INEPNakEffMsk	R/W	0x0	IN Endpoint NAK Effective Mask
5	INTknEPMisMsk	R/W	0x0	IN Token received with EP Mismatch Mask
4	INTknTXFEmpMsk	R/W	0x0	IN Token Received When TxFIFO Empty Mask
3	TimeOUTMsk	R/W	0x0	Timeout Condition Mask (Non-isochronous endpoints)
2	AHBErrMsk	R/W	0x0	AHB Error Mask
1	EPDisbldMsk	R/W	0x0	Endpoint Disabled Interrupt Mask
0	XferComplMsk	R/W	0x0	Transfer Completed Mask

Device OUT Endpoint Common Interrupt Mask Register (DOEPMSK) 0xB0040814

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							OutPktErrMsk		Back2BackSETup		OUTTknEPdisMsk	SetUPMsk	AHBErrMsk	EPDisbldMsk	XferComplMsk

This register works with each of the Device OUT Endpoint Interrupt (DOEPINTn) registers for all endpoints to generate an interrupt per OUT endpoint. The OUT endpoint interrupt for a specific status in the DOEPINTn register can be masked by writing into the corresponding bit in this register. Status bits are masked by default.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-9	-	-	0x0	Reserved
8	OutPktErrMsk	R/W	0x0	OUT Packet Error Mask
7	-	-	0x0	Reserved
6	Back2BackSETup	R/W	0x0	Back-to-Back SETUP Packets Received Mask Applies to control OUT endpoints only.
5	-	-	0x0	Reserved
4	OUTTknEPdisMsk	R/W	0x0	OUT Token Received when Endpoint Disabled Mask Applies to control OUT endpoints only.
3	SetUPMsk	R/W	0x0	SETUP Phase Done Mask Applies to control endpoints only.
2	AHBErrMsk	R/W	0x0	AHB Error Mask
1	EPDisbldMsk	R/W	0x0	Endpoint Disabled Interrupt Mask
0	XferComplMsk	R/W	0x0	Transfer Completed Interrupt Mask

Device All Endpoints Interrupt Register (DAINT) 0xB0040818

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OutEPInt															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
InEpInt															

When a significant event occurs on an endpoint, a Device All Endpoints Interrupt register interrupts the application using the Device OUT Endpoints Interrupt bit or Device IN Endpoints Interrupt bit of the Core Interrupt register (GINTSTS.OEPInt or GINTSTS.IEPInt, respectively). This is shown in Figure 4-2. There is one interrupt bit per endpoint, up to a maximum of 16 bits for OUT endpoints and 16 bits for IN endpoints. For a bidirectional endpoint, the corresponding IN and OUT interrupt bits are used. Bits in this register are set and cleared when the application sets and clears bits in the corresponding Device Endpointn Interrupt register (DIEPINTn/DOEPINTn).

Field	Name	RW	Reset	Description
31-16	OutEPInt	R	0x0	OUT Endpoint Interrupt Bits One bit per OUT endpoint: Bit 16 for OUT endpoint 0, bit 31 for OUT endpoint 15
15-0	InEpInt	R	0x0	IN Endpoint Interrupt Bits One bit per IN Endpoint: Bit 0 for IN endpoint 0, bit 15 for endpoint 15

Device All Endpoints Interrupt Mask Register (DAINTMSK) 0xB004081C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OutEPInt															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
InEpInt															

The Device Endpoint Interrupt Mask register works with the Device Endpoint Interrupt register to interrupt the application when an event occurs on a device endpoint. However, the Device All Endpoints Interrupt (DAINT) register bit corresponding to that interrupt is still set.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-16	OutEPMsk	R	0x0	OUT Endpoint Interrupt Mask Bits One bit per OUT endpoint: Bit 16 for OUT endpoint 0, bit 31 for OUT endpoint 15
15-0	InEpMsk	R	0x0	IN Endpoint Interrupt Mask Bits One bit per IN Endpoint: Bit 0 for IN endpoint 0, bit 15 for IN endpoint 15

Device VBUS Discharge Time Register (DVBUSDIS) 0xB0040828

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVBUSDis															

This register specifies the VBUS discharge time after VBUS pulsing during SRP.

Field	Name	RW	Reset	Description
31-16	-	-	0x0	Reserved
15-0	DVBUSDis	R/W	30 MHz: 0x0B8F 60 MHz: 0x17D7	Device VBUS Discharge Time Specifies the VBUS discharge time after VBUS pulsing during SRP. This value equals: VBUS discharge time in PHY clocks / 1, 024 The value you use depends whether the PHY is operating at 30 MHz (16-bit data width) or 60 MHz (8-bit data width). Depending on your VBUS load, this value can need adjustment.

Device VBUS Pulsing Time Register (DVBUSPULSE) 0xB004082C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				DVBUSPulse											

This register specifies the VBUS pulsing time during SRP.

Field	Name	RW	Reset	Description
31-12	-	-	0x0	Reserved
11-0	DVBUSPulse	R/W	30 MHz: 12'h2C6 60 MHz: 12'h5B8	Device VBUS Pulsing Time Specifies the VBUS pulsing time during SRP. This value equals: VBUS pulsing time in PHY clocks / 1, 024 The value you use depends whether the PHY is operating at 30 MHz (16-bit data width) or 60 MHz (8-bit data width).

Device Threshold Control Register (DTHRCTL) 0xB0040830

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				ArbPrkEn		RxThrLen									RxThrEn
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					TxThrLen									ISOThrEn	NonISOThrEn

Thresholding is not supported in Slave mode and so this register should not be programmed in Slave mode. For threshold support, the AHB needs to be run at 60MHz or higher.

Field	Name	RW	Reset	Description
31-28	-	-	0x0	Reserved
27	ArbPrkEn	R/W	0x1	Arbiter Parking Enable This bit controls internal DMA arbiter parking for IN endpoints. When thresholding is enabled and this bit is set to one, then the arbiter parks on the IN endpoint for which there is a token received on the USB. This is done to avoid getting into underrun conditions. By default the parking is enabled.
26	-	-	0x0	Reserved
25-17	RxThrLen	R/W	0x0	Receive Threshold Length This field specifies Receive thresholding size in DWORDS. This field also specifies the amount of data received on the USB before the core can start transmitting on the AHB. The threshold length has to be at least eight DWORDS. The recommended value for ThrLen is to be the same as the programmed AHB Burst Length (GAHBCFG.HBstLen).
16	RxThrEn	R/W	0x0	Receive Threshold Enable When this bit is set, the core enables thresholding in the receive direction.
15-11	-	-	0x0	Reserved
10-2	TxThrLen	R/W	0x0	Transmit Threshold Length This field specifies Transmit thresholding size in DWORDS. This field specifies the amount of data in bytes to be in the corresponding endpoint transmit FIFO, before the core can start transmit on the USB. The threshold length has to be at least eight DWORDS. This field controls both isochronous and non-isochronous IN endpoint thresholds. The recommended value for ThrLen is to be the same as the programmed AHB Burst Length (GAHBCFG.HBstLen).
1	ISOThrEn	R/W	0x0	ISO IN Endpoints Threshold Enable. When this bit is set, the core enables thresholding for isochronous IN endpoints.
0	NonISOThrEn	R/W	0x0	Non-ISO IN Endpoints Threshold Enable When this bit is set, the core enables thresholding for Non Isochronous IN endpoints.

Device IN Endpoint FIFO Empty Interrupt Mask Register: (DIEPEMPMSK) 0xB0040834

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
InEpTxfEmpMsk															

This register is used to control the IN endpoint FIFO empty interrupt generation (DIEPINTn.TxfEmp).

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-16	-	-	0x0	Reserved
15-0	InEpTxfEmpMsk	R/W	0x0	IN EP Tx FIFO Empty Interrupt Mask Bits These bits acts as mask bits for DIEPINTn. TxFEmp interrupt One bit per IN Endpoint: Bit 0 for IN EP 0, bit 15 for IN EP 15

Device Control IN Endpoint 0 Control Register (DIEPCTL0) 0xB0040900

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPEna	EPDis			SNAK	CNAK	TxFNum				Stall		EPType		NAKSts	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBA ctEP														MPS	

This section describes the Control IN Endpoint 0 Control register. Nonzero control endpoints use registers for endpoints 1–15.

Field	Name	RW	Reset	Description
31	EPEna	R/W	0x0	Endpoint Enable • When Scatter/Gather DMA mode is enabled, for IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. • When Scatter/Gather DMA mode is disabled—such as in buffer-pointer based DMA mode—this bit indicates that data is ready to be transmitted on the endpoint. The core clears this bit before setting the following interrupts on this endpoint: • Endpoint Disabled • Transfer Completed
30	EPDis	R/W	0x0	Endpoint Disable The application sets this bit to stop transmitting data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled Interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint.
29-28	-	-	0x0	Reserved
27	SNAK	W	0x0	Set NAK A write to this bit sets the NAK bit for the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also set this bit for an endpoint after a SETUP packet is received on that endpoint.
26	CNAK	R/W	0x0	Clear NAK A write to this bit clears the NAK bit for the endpoint.
25-22	TxFNum	R/W	0x0	TxFIFO Number This value is set to the FIFO number that is assigned to IN Endpoint 0. ********** Warning TxFIFO Number should be same as the Device IN Endpoint Number
21	Stall	R/W	0x0	STALL Handshake The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Nonperiodic IN NAK, or Global OUT NAK is set along with this bit,

				the STALL bit takes priority.
20	-	-	0x0	Reserved
19-18	EPType	R	0x0	Endpoint Type Hardcoded to 00 for control.
17	NAKSts	R	0x0	NAK Status Indicates the following: • 1’b0: The core is transmitting non-NAK handshakes based on the FIFO status • 1’b1: The core is transmitting NAK handshakes on this endpoint. When this bit is set, either by the application or core, the core stops transmitting data, even if there is data available in the TxFIFO. Irrespective of this bit’s setting, the core always responds to SETUP data packets with an ACK handshake.
16	-	-	0x0	Reserved
15	USBActEP	R	0x1	USB Active Endpoint This bit is always set to 1, indicating that control endpoint 0 is always active in all configurations and interfaces.
14-2	-	-	0x0	Reserved
1-0	MPS	R/W	0x0	Maximum Packet Size Applies to IN and OUT endpoints. The application must program this field with the maximum packet size for the current logical endpoint. • 2’b00: 64 bytes • 2’b01: 32 bytes • 2’b10: 16 bytes • 2’b11: 8 bytes

Device Control OUT Endpoint 0 Control Register (DOEPCTL0) **0xB0040B00**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPEna	EPDis			SNAK	CNAK					Stall	Snp	EPType		NAKSts	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBA ctEP														MPS	

This section describes the Control OUT Endpoint 0 Control register. Nonzero control endpoints use registers for endpoints 1–15.

Field	Name	RW	Reset	Description
31	EPEna	R/W	0x0	Endpoint Enable When Scatter/Gather DMA mode is enabled, for OUT endpoints this bit indicates that the descriptor structure and data buffer to receive data is setup. • When Scatter/Gather DMA mode is disabled—(such as for buffer-pointer based DMA mode)—this bit indicates that the application has allocated the memory to start receiving data from the USB. The core clears this bit before setting any of the following interrupts on this endpoint: • SETUP Phase Done • Endpoint Disabled • Transfer Completed Note: In DMA mode, this bit must be set for the core to transfer SETUP data packets into memory.
30	EPDis	R	0x0	Endpoint Disable The application cannot disable control OUT endpoint 0.
29-28	-	-	0x0	Reserved
27	SNAK	W	0x0	Set NAK A write to this bit sets the NAK bit for the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also set this bit on a Transfer Completed interrupt, or after a SETUP packet is received on the endpoint.
26	CNAK	R/W	0x0	Clear NAK A write to this bit clears the NAK bit for the endpoint.
25-22	-	-	0x0	Reserved

21	Stall	R/W	0x0	STALL Handshake The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake.
20	Snp	R/W	0x0	Snoop Mode This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory.
19-18	EPType	R	0x0	Endpoint Type Hardcoded to 00 for control.
17	NAKSts	R	0x0	NAK Status Indicates the following: • 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. • 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit, the core stops receiving data, even if there is space in the RxFIFO to accommodate the incoming packet. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake.
16	-	-	0x0	Reserved
15	USBActEP	R	0x1	USB Active Endpoint This bit is always set to 1, indicating that control endpoint 0 is always active in all configurations and interfaces.
14-2	-	-	0x0	Reserved
1-0	MPS	R	0x0	Maximum Packet Size The maximum packet size for control OUT endpoint 0 is the same as what is programmed in control IN Endpoint 0. • 2'b00: 64 bytes • 2'b01: 32 bytes • 2'b10: 16 bytes • 2'b11: 8 bytes

Device Endpoint-n Control Register (DIEPCTLn/DOEPCTLn)

0xB0040900+(Endpoint_number*0x20) for IN endpoints
0xB0040B00+(Endpoint_number*0x20) for OUT endpoints
Endpoint_number:1≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPEna	EPDis	SetD1PID/SetOddFr	SetD0PID/SetEvenFr	SNAK	CNAK	TxFNum				Stall	Snp	EPType		NAKSts	DPID/EO_FrNum
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBActEP					MPS										

The application uses this register to control the behavior of each logical endpoint other than endpoint 0.

Field	Name	RW	Reset	Description
31	EPEna	R/W	0x0	Endpoint Enable Applies to IN and OUT endpoints. For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. The core clears this bit before setting any of the following interrupts on this endpoint: • SETUP Phase Done (OUT only) • Endpoint Disabled • Transfer Completed Note: For control OUT endpoints in DMA mode, this bit must be

				set to be able to transfer SETUP data packets in memory.
30	EPDis	R/W	0x0	Endpoint Disable Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint.
29	SetD1PID	W	0x0	Set DATA1 PID Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1.
	SetOddFr			Set Odd (micro)frame Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame.
28	SetD0PID	W	0x0	Set DATA0 PID Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0.
	SetEvenFr			Set Even (micro)frame Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro)frame.
27	SNAK	W	0x0	Set NAK Applies to IN and OUT endpoints. A write to this bit sets the NAK bit for the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also set this bit for OUT endpoints on a Transfer Completed interrupt, or after a SETUP is received on the endpoint.
26	CNAK	W	0x0	Clear NAK A write to this bit clears the NAK bit for the endpoint.
25-22	TxFNum	R/W	0x0	TxFIFO Number These bits specify the FIFO number associated with this endpoint. Each active IN endpoint should be programmed to a separate FIFO number. This field is valid only for IN endpoints. ********** Warning TxFIFO Number should be same as the Device IN Endpoint Number
21	Stall	R/W	0x0	STALL Handshake Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit’ s setting, the core always responds to SETUP data packets with an ACK handshake.
20	Snp	R/W	0x0	Snoop Mode Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory.
19-18	EPType	R/W	0x0	Endpoint Type Applies to IN and OUT endpoints. This is the transfer type supported by this logical endpoint. • 2’ b00: Control

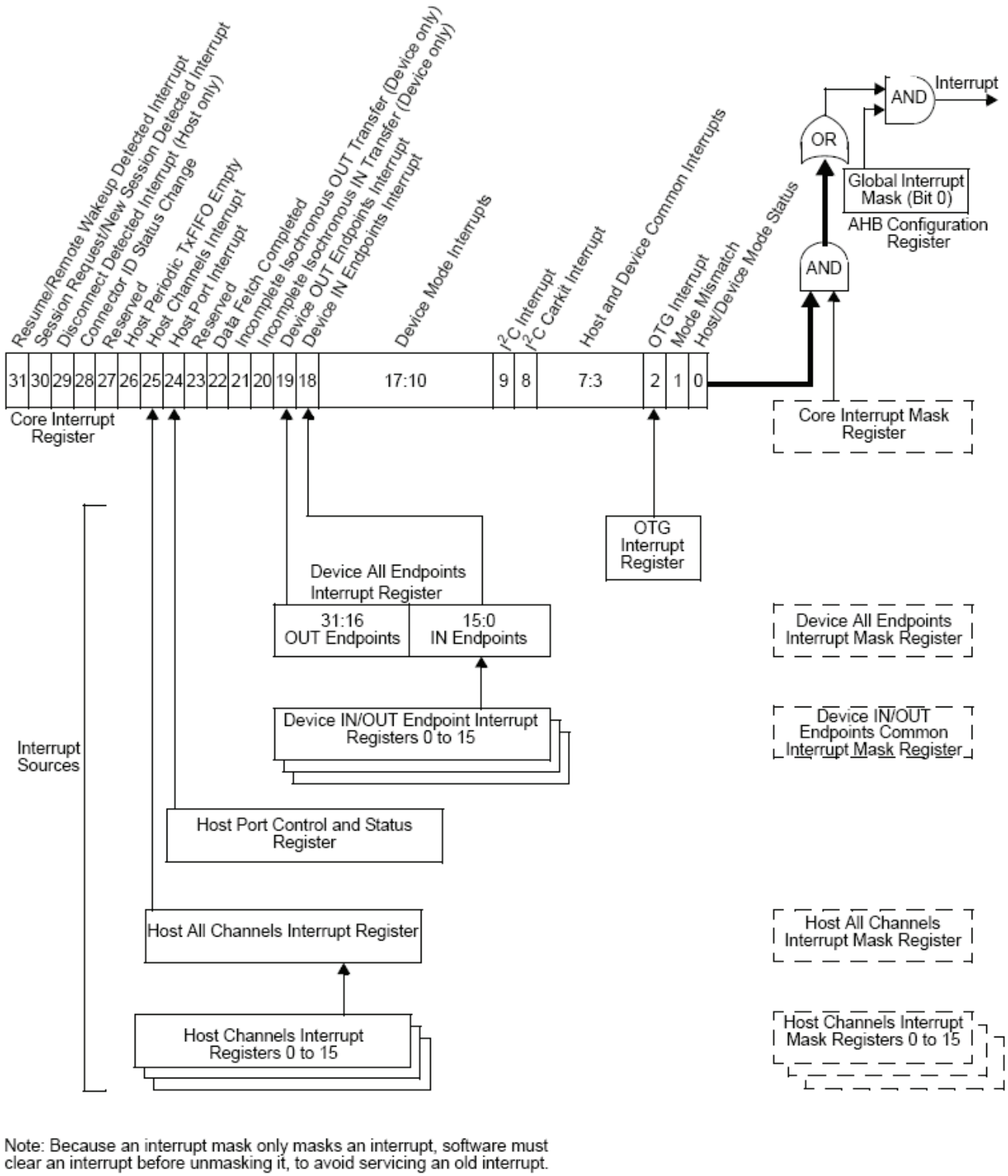
				• 2’b01: Isochronous • 2’b10: Bulk • 2’b11: Interrupt
17	NAKSts	R	0x0	NAK Status Applies to IN and OUT endpoints. Indicates the following: • 1’b0: The core is transmitting non-NAK handshakes based on the FIFO status. • 1’b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: • The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. • For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. • For isochronous IN endpoints: The core sends out a zerolength data packet, even if there data is available in the TxFIFO. Irrespective of this bit’s setting, the core always responds to SETUP data packets with an ACK handshake.
16	DPID	R	0x0	Endpoint Data PID Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. Applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. • 1’b0: DATA0 • 1’b1: DATA1
	EO_FrNum			Even/Odd (Micro)Frame Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. • 1’b0: Even (micro)frame • 1’b1: Odd (micro)frame
15	USBActEP	R/W	0x0	USB Active Endpoint Applies to IN and OUT endpoints. Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program endpoint registers accordingly and set this bit.
14-11	-	-	0x0	Reserved
10-0	MPS	R/W	0x0	Maximum Packet Size Applies to IN and OUT endpoints. The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.

Device Endpoint-n Interrupt Register (DIEPINTn/DOEPINTn)

0xB0040908+(Channel_number*0x20) for IN Endpoints
0xB0040B08+(Channel_number*0x20) for OUT Endpoints
Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							TxfifoUndrn/OutPktErr	TxFEmp	INEPNakEff/Back2BackSETup	INTknEPMis	INTknTXFEmp/OUTTknEPdis	TimeOUT/SetUp	AHBErr	EPDisbld	XferCompl

This register indicates the status of an endpoint with respect to USB- and AHB-related events. It is shown in



Note: Because an interrupt mask only masks an interrupt, software must clear an interrupt before unmasking it, to avoid servicing an old interrupt.

Figure 13.3. The application must read this register when the OUT Endpoints Interrupt bit or IN Endpoints Interrupt bit of the Core Interrupt register (GINTSTS.OEPInt or GINTSTS.IEPInt, respectively) is set. Before the application can read this register, it must first read the Device All Endpoints Interrupt (DAINT) register to get the exact endpoint number for the Device Endpoint-n Interrupt register. The application must clear the appropriate bit in this register to clear the corresponding bits in the DAINT and GINTSTS registers.

Field	Name	RW	Reset	Description
31-9	-	-	0x0	Reserved
8	TxfifoUndrn	R/W	0x0	Fifo Underrun Applies to IN endpoints Only This bit is valid only if thresholding is enabled. Core generates this interrupt when it see a transmit FIFO underrun condition for this

				endpoint.
	OutPktErr			OUT Packet Error Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core sees an overflow or a CRC error for non-ISOC OUT packet.
7	TxFEmp	R	0x1	Transmit FIFO Empty This bit is valid only for IN Endpoints This interrupt is asserted when the TxFIFO for this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)).
6	INEPNakEff	R/W	0x0	IN Endpoint NAK Effective Applies to periodic IN endpoints only. Indicates that the IN endpoint NAK bit set by the application has taken effect in the core. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit set (either by the application or by the core). This interrupt does not necessarily mean that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit.
	Back2BackSETup			Back-to-Back SETUP Packets Received Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets for this particular endpoint. For information about handling this interrupt, see page 13-366.
5	INTknEPMis	R/W	0x0	IN Token Received with EP Mismatch Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one for which the IN token was received. This interrupt is asserted on the endpoint for which the IN token was received. For OUT endpoints, this bit is Reserved
4	INTknTXFEmp	R/W	0x0	IN Token Received When TxFIFO is Empty Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is asserted on the endpoint for which the IN token was received.
	OUTTknEPdis			OUT Token Received When Endpoint Disabled Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint for which the OUT token was received.
3	TimeOUT	R/W	0x0	Timeout Condition Applies only to Control IN endpoints. Indicates that the core has detected a timeout condition on the USB for the last IN token on this endpoint.
	SetUp			SETUP Phase Done Applies to control OUT endpoints only. Indicates that the SETUP phase for the control endpoint is complete and no more back-to-back SETUP packets were received for the current control transfer. On this interrupt, the application can decode the received SETUP data packet.
2	AHBErr	R/W	0x0	AHB Error Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address.
1	EPDisbld	R/W	0x0	Endpoint Disabled Interrupt Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request.
0	XferCompl	R/W	0x0	Transfer Completed Interrupt Applies to IN and OUT endpoints. Indicates that the programmed transfer is complete on the AHB as

				well as on the USB, for this endpoint.

Device Endpoint 0 Transfer Size Register (DIEPTSIZ0/DOEPTSIZ0)

0xB0040910 for IN Endpoints
0xB0040B10 for OUT Endpoints

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SUPCnt										PktCnt				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
									XferSize						

The application must modify this register before enabling endpoint 0. Once endpoint 0 is enabled using Endpoint Enable bit of the Device Control Endpoint 0 Control registers (DIEPCTL0.EPEna/DOEPCTL0.EPEna), the core modifies this register. The application can only read this register once the core has cleared the Endpoint Enable bit. SUPCnt field is available only in DOEPTSIZ0

Nonzero endpoints use the registers for endpoints 1–15.

[Device IN Endpoint 0 Transfer Size Register: DIEPTSIZ0]

Field	Name	RW	Reset	Description
31-21	-	-	0x0	Reserved
20-19	PktCnt	R/W	0x0	Packet Count Indicates the total number of USB packets that constitute the Transfer Size amount of data for endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.
18-7	-	-	0x0	Reserved
6-0	XferSize	R/W	0x0	Transfer Size Indicates the transfer size in bytes for endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.

[Device OUT Endpoint 0 Transfer Size Register: DOEPTSIZ0]

Field	Name	RW	Reset	Description
31	-	-	0x0	Reserved
30-29	SUPCnt	R/W	0x0	SETUP Packet Count This field specifies the number of back-to-back SETUP data packets the endpoint can receive. • 2’b01: 1 packet • 2’b10: 2 packets • 2’b11: 3 packets
28-20	-	-	0x0	Reserved
20-19	PktCnt	R/W	0x0	Packet Count This field is decremented to zero after a packet is written into the RxFIFO.
18-7	-	-	0x0	Reserved
6-0	XferSize	R/W	0x0	Transfer Size Indicates the transfer size in bytes for endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.

Device Endpoint-n Transfer Size Register (DIEPTSIZn/DOEPTSIZn)

0xB0040910+(Endpoint_number*0x20) for IN Endpoints
0xB0040B10+(Endpoint_number*0x20) for OUT Endpoints
Endpoint_number:1≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MC/ RxDPID/ SUPCnt		PktCnt										XferSize[18:16]		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
XferSize[15:0]															

The application must modify this register before enabling the endpoint. Once the endpoint is enabled using Endpoint Enable bit of the Device Endpoint-n Control registers (DIEPCTLn.EPEna/DOEPCTLn.EPEna), the core modifies this register. The application can only read this register once the core has cleared the Endpoint Enable bit.
This register is used only for endpoints other than Endpoint 0.

Field	Name	RW	Reset	Description
31	-	-	0x0	Reserved
30-29	MC	R/W	0x0	Multi Count Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. • 2’b01: 1 packet • 2’b10: 2 packets • 2’b11: 3 packets
	-	R		For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core should fetch for an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp).
	RxDPID	R		Received Data PID Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. • 2’b00: DATA0 • 2’b01: DATA2 • 2’b10: DATA1 • 2’b11: MDATA
	SUPCnt	R/W		SETUP Packet Count Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. • 2’b01: 1 packet • 2’b10: 2 packets • 2’b11: 3 packets
28-19	PktCnt	R/W	0	Packet Count Indicates the total number of USB packets that constitute the Transfer Size amount of data for this endpoint. • IN Endpoints: This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO. • OUT Endpoints: This field is decremented every time a packet (maximum size or short packet) is written to the RxFIFO.
18-0	XferSize	R/W	0	Transfer Size This field contains the transfer size in bytes for the current endpoint. The core only interrupts the application after it has exhausted the transfer size amount of data. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. • IN Endpoints: The core decrements this field every time a packet from the external memory is written to the TxFIFO. • OUT Endpoints: The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.

Device Endpoint-n DMA Address Register (DIEPDMAn/DOEPDMAn)

0xB0040914+(Endpoint_number*0x20) for IN Endpoints
0xB0040B14+(Endpoint_number*0x20) for OUT Endpoints
Endpoint_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAAddr[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMAAddr[15:0]															

Field	Name	RW	Reset	Description
31-0	DMAAddr	R/W	0x0	DMA Address Holds the start address of the external memory for storing or fetching endpoint data. This register is incremented on every AHB transaction. Application can give only a DWORD-aligned address. Note: For control endpoints, this address stores control OUT data packets as well as SETUP transaction data packets. If more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten.

Device IN Endpoint Transmit FIFO Status Register (DTXFSTSn)

0xB0040918+(Endpoint_number*0x20) for IN Endpoints
Endpoint_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INEPTxFSpcAvail															

This read-only register contains the free space information for the Device IN endpoint TxFIFO.

Field	Name	RW	Reset	Description
31-16	-	-	0x0	Reserved
15-0	INEPTxFSpcAvail	R	0x0	IN Endpoint TxFIFO Space Avail Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. • 16’h0: Endpoint TxFIFO is full • 16’h1: 1 word available • 16’h2: 2 words available • 16’hn: n words available (where 0≤n≤ 32,768) • 16’h8000: 32,768 words available • Others: Reserved

Power and Clock Gating Control Register (PCGCCTL) 0xB0040E00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							Reset After Susp	L1 Suspended	Phy Sleep	Enbl_L1Gating		Enbl_L1Gating	RstPdwnModule	Pwr Clmp	Stop Pclk

This register is available in Host and Device modes. The PwrClmp bit is available only if the OTG_EN_PWROPT parameter is set to 1 during core configuration. The application can use this register to control the core's power-down and clock gating features.
Because the CSR module is turned off during power-down, this registers is implemented in the AHB Slave BIU module.

Field	Name	RW	Reset	Description
31-9	-	-	0x0	Reserved
8	ResetAfterSusp	RW	0x0	Reset After Suspend In partial power-down mode of operation, this bit needs to be set in host mode before clamp is removed if the host needs to issue reset after suspend. If this bit is not set, then the host issues resume after suspend. This bit is not applicable in device mode and non-partial power-down mode
7	L1 Suspended	R	0x0	Deep Sleep This bit indicates that the PHY is in deep sleep when in L1 state.
6	PhySleep	R	0x0	PHY in Sleep This bit indicates that the PHY is in the Sleep state.
5	Enbl_L1Gating	RW	0x0	Enable Sleep Clock Gating When this bit is set, core internal clock gating is enabled in Sleep state if the core cannot assert utmi_l1_suspend_n. When this bit is not set, the PHY clock is not gated in Sleep state.
4	-	-	-	Reserved
3	RstPdwnModule	RW	0x0	Reset Power-Down Modules This bit is valid only in Partial Power-Down mode. The application sets this bit when the power is turned off. The application clears this bit after the power is turned on and the PHY clock is up.
2	PwrClmp	RW	0x0	Power Clamp This bit is valid only in Partial Power-Down mode (OTG_EN_PWROPT = 1). The application sets this bit before the power is turned off to clamp the signals between the power-on modules and the power-off modules. The application clears the bit to disable the clamping before the power is turned on.
1	GateHclk	RW	0x0	Gate Hclk The application sets this bit to gate hclk to modules other than the AHB Slave and Master and wakeup logic when the USB is suspended or the session is not valid. The application clears this bit when the USB is resumed or a new session starts.
0	StopPclk	RW	0x0	Stop Pclk The application sets this bit to stop the PHY clock (phy_clk) when the USB is suspended, the session is not valid, or the device is disconnected. The application clears this bit when the USB is resumed or a new session starts

USB OTG Configuration Register (OTGCR) 0xB0080020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												OTGAMAP		ENDS	ENDM

BIT	Name	RW	Reset	Description
3-2	OTGAMAP	R/W	2'b00	Fixed value of address[17:16] into USB OTG link slave interface
1	ENDS	R/W	0	Endian of slave interface of USB OTG link 0: little, 1: big
0	ENDM	R/W	0	Endian of master interface of USB OTG link 0: little, 1: big

13.3 Register Description for UTMI (USB PHY)

USB PHY Configuration Register0 (UPCR0) 0xB0080024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	PR	CM	RCS		RCD		SDI	FO	VBDS	DMPD	DPPD	TBSH	TBS	VBD	LBE

Field	Name	RW	Reset	Description
14	PR	R/W	0	When asserted, this customer-specific signal resets the corresponding port's transmit and receive logic without disabling the clocks within the USB 2.0 nanoPHY. 0: The transmit and receive FSMs are operational, and the line_state logic becomes sequential after 11 PHYCLOCK cycles. 1: The transmit and receive finite state machines are reset, and the line_state logic combinationally reflects the state of the single-ended receivers.
13	CM	R/W	1	This signal controls the power-down signals in the XO, Bias, and PLL blocks when the USB 2.0 nanoPHY is suspended. 0: The XO, Bias, and PLL Blocks remains powered in suspend mode. 1: The XO, Bias, and PLL Blocks are powered down in suspend mode.
12-11	RCS	R/W	1	This signal selects the reference clock source for the PLL block. **RCS** / **Common Block Power Down Control** 11 — The PLL uses CLKCORE as reference 10 — The PLL uses CLKCORE as reference 01 — The XO block uses an external clock supplied on the XO pin(default) 00 — The XO block uses the clock from a crystal
10-9	RCD	R/W	0	This signal selects the reference clock source for the PLL block. **RCD** / **Reference Clock Frequency Select** 11 — Reserved 10 — 48 MHz 01 — 24 MHz 00 — 12 MHz
8	SDI	R/W	1	This test signal enables you to perform IDDQ testing by powering down all analog blocks. **SID** / **IDDQ Test Enable** 1 — The analog blocks are powered down 0 — The analog blocks are not powered down.
7	FO	R/W	0	This controller signal enables the UTMI or serial interface. ** It is effective only when UPCR0.MODE is set and you should not clear this bit when UPCR0.MODE is set **FO** / **UTMI/Serial Interface Select** 1 — The TXENABLEN, FSDATAEXT, and FSSE0EXT inputs drive USB 2.0 nanoPHY data output onto the D+ and D- lines 0 — Data on the D+ and D- lines is transmitted and received through the UTMI.
6	VBDS	R/W	0	This controller signal enables the UTMI or serial interface. This signal selects the VBUSVLDEXT input or the internal Session Valid comparator to indicate when the VBUS signal on the USB cable is valid

<table>
<tr><td></td><td></td><td></td><td></td><td><table><tr><th>VBDS</th><th>External VBUS Valid Select</th></tr><tr><td>1</td><td>The VBUSVLDEXT input is used</td></tr><tr><td>0</td><td>The internal Session Valid comparator is used</td></tr></table></td></tr>
<tr><td>5</td><td>DMPD</td><td>R/W</td><td>1</td><td>This controller signal enables or disables the pull-down resistance on the D-line.
When an A/B device is acting as a host (Downstream-facing port), DPPULLDOWN and DMPULLDOWN are enabled. UTMI+/OTG-compliant controllers are not allowed to toggle DPPULLDOWN and DMPULLDOWN during normal operation. USB PHY Configuration Register2 (UPCR2).OPMODE = 2’b01(non-driving) has precedence over the DPPULLDOWN and DMPULLDOWN controls.
** It is effective only when UPCR0.MODE is set.<table><tr><th>DMPD</th><th>D- Pull-Down Resistor Enable</th></tr><tr><td>1</td><td>The pull-down resistance on D- is enabled.</td></tr><tr><td>0</td><td>The pull-down resistance on D- is disabled.</td></tr></table></td></tr>
<tr><td>4</td><td>DPPD</td><td>R/W</td><td>1</td><td>This controller signal enables or disables the pull-down resistance on the D+line.
** It is effective only when UPCR0.MODE is set.<table><tr><th>DPPD</th><th>D+ Pull-Down Resistor Enable</th></tr><tr><td>1</td><td>The pull-down resistance on D+ is enabled</td></tr><tr><td>0</td><td>The pull-down resistance on D+ is disabled</td></tr></table></td></tr>
<tr><td>3</td><td>TBSH</td><td>R/W</td><td>0</td><td>This controller signal enables or disables bits stuffing on DATAIN[7:0] when OPMODE[1:0] =2’b11<table><tr><th>TBSH</th><th>Low-Byte Transmit Bit-Stuffing Enable</th></tr><tr><td>1</td><td>Bit stuffing is enabled</td></tr><tr><td>0</td><td>Bit stuffing is disabled.</td></tr></table></td></tr>
<tr><td>2</td><td>TBS</td><td>R/W</td><td>0</td><td>This controller signal enables or disables bits stuffing on DATAIN[7:0] when OPMODE[1:0] =2’b11<table><tr><th>TBS</th><th>Low-Byte Transmit Bit-Stuffing Enable</th></tr><tr><td>1</td><td>Bit stuffing is enabled</td></tr><tr><td>0</td><td>Bit stuffing is disabled</td></tr></table></td></tr>
<tr><td>1</td><td>VBD</td><td>R/W</td><td>0</td><td>This signal is valid in device mode only when the VBUSVLDEXTSEL signal is high. VBUSVLDEXTSEL indicates whether the VBUS signal on the USB cable is valid. In addition, VBUSVLDEXTSEL enables the pull-up resistor on the D+ line.<table><tr><th>VBD</th><th>External VBUS Valid Indicator</th></tr><tr><td>1</td><td>The VBUS signal is valid, and the pull-up resistor on D+ is enabled</td></tr><tr><td>0</td><td>The VBUS signal is not valid, and the pull-up resistor on D+ is disabled.</td></tr></table></td></tr>
<tr><td>0</td><td>LBE</td><td>R/W</td><td>0</td><td>This controller signal places the USB 2.0 nanoPHY in Loopback mode, which enables the receive and transmit logic concurrently.<table><tr><th>LBE</th><th>Loopback Test Enable</th></tr><tr><td>1</td><td>During data transmission, the receive logic is enabled.</td></tr><tr><td>0</td><td>During data transmission, the receive logic is disabled.</td></tr></table></td></tr>
</table>

[VBUS Valid and End Interrupt]

VBUS is the USB power supply pin. When in device mode, an off-chip charge pump must provide power to VBUS pin. Alternatively VBUS can be set internally by VBUSVLDEXT and VBUSVLDEXTSEL in USB PHY Configuration Register 0(UPCR0). If VBUSVLDEXTSEL is set to 1, VBUSVLDEXT value can be supplied as VBUS data of USB PHY.

There are two interrupts related to VBUS and these interrupts, which indicate the state of VBUS, are generated from B_VALID and SESSEND signals. B_VALID signal is set to 1 when VBUS is 1 and SESSEND signal is set to 1 when VBUS is 0. Therefore, when VBUS value changes, one of two interrupts definitely occurs

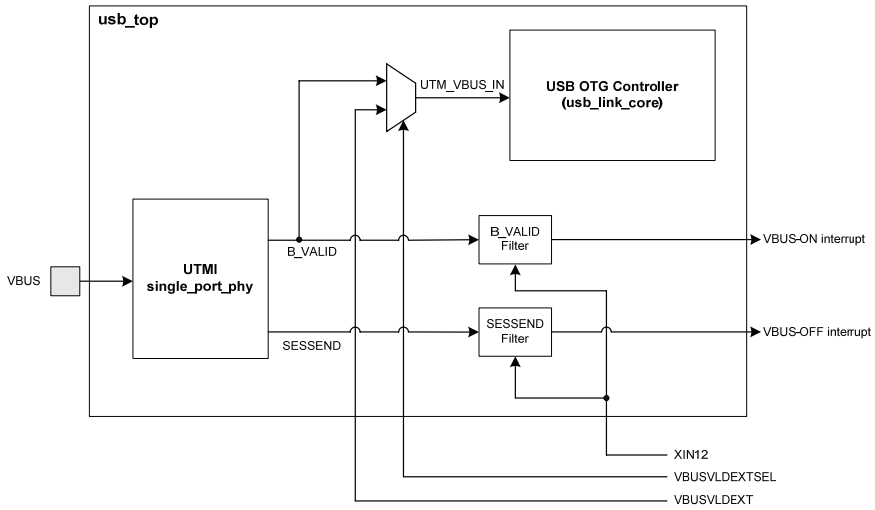


Figure 13.4 VBUS Control and Interrupts

USB PHY Configuration Register1 (UPCR1) 0xB0080028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXFSLST					SQRXT				OTGT				CDT		

Field	Name	RW	Reset	Description
15-12	TXFSLST	R/W	0x7	FS/LS Pull-Up Resistance Adjustment This bus adjusts the low- and full-speed pull-up resistance, based on nominal power, voltage, and temperature. 1111 -2.5% 0111 Design default (default) 0011 +2.5% 0001 +5% 0000 +7.5% Others Reserved
10-8	SQRXT	R/W	0x3	Squelch Threshold Tune This bus adjusts the voltage level for the threshold used to detect valid high-speed data 111 -20% 110 -15% 101 -10% 100 -5% 011 Design default (default) 010 +5% 001 +10% 000 +15% Others Reserved
6-4	OTGT	R/W	0x4	VBUS Valid Threshold Adjustment This bus adjusts the voltage level for the threshold used to detect valid high-speed data. 111 +9% 110 +6% 101 +3% 100 Design default(default) 011 -3% 010 -6% 001 -7% 000 -12% Others Reserved
2-0	CDT	R/W	0x3	Disconnect Threshold Adjustment This bus adjusts the voltage level for the threshold used to detect a disconnect event at the host. 111 +6% 110 +4.5% 101 +3% 100 +1.5% 011 Design default (default) 010 -1.5% 001 -3% 000 -4.5%

USB PHY Configuration Register2 (UPCR2) 0xB008002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM	XCVRSEL			OPMODE		TXVRT					TXRT			TP

Field	Name	RW	Reset	Description
14	TM	R/W	0	USB Termination Select 1 Full Speed termination is enabled (default). 0 High Speed termination is enabled. This controller signal selects FS or HS termination. ** It is effective only when UPCR0.MODE is set
13-12	XCVRSEL	R/W	0	FS/LS Pull-Up Resistance Adjustment 11 Sends an LS packet on an FS bus or receives an Ls packet 10 LS transceiver 01 FS transceiver (default) 00 HS transceiver This controller bus selects the HS, FS, or LS transceiver. ** It is effective only when UPCR0.MODE is set
10-9	OPMODE	R/W	0x3	UTMI Operational Mode 11 Normal operation SYNC or EOP generation 10 Disable bit stuffing and NRZI encoding 01 Non driving 00 Normal(default) This controller bus selects the UTMI operational mode. ** It is effective only when UPCR0.MODE is set.
8-5	TXVRT	R/W	0x8	HS DC Voltage Level Adjustment 1111 +8.75% 1110 +7.5% 1101 +6.25% 1100 +5% 1011 +3.75% 1010 +2.5% 1001 +1.25% 1000 Design default (default) 0111 -1.25% 0110 -2.5% 0101 -3.75% 0100 -5% 0011 -6.25% 0010 -7.5% 0001 -8.75% 0000 -10% This bus adjusts the voltage to which the high speed DC level is tuned.
3-2	TXRT	R/W	0	HS Transmitter Rise/Fall Time Adjustment 01 -8% 00 Design default (default) Others Reserved This bus adjusts the rise/fall times the high speed waveform
0	TP	R/W	0	HS transmitter Pre-Emphasis Enable 1 The HS Transmitter pre-emphasis is enabled 0 The HS transmitter pre-emphasis is disabled. (default) This signal enables or disables the pre-emphasis for a J-K or K-J state transition in HS mode

USB PHY Configuration Register3 (UPCR3) **0xB0080030**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SCALE		OTG									TXHSXVT		TXF	ABE

Field	Name	RW	Reset	Description
14-13	SCALE	R/W	0	Scale down mode 0 This is enabled during simulation only. This controller signal powers down the OTG block, including the VBUS Valid comparator. If the application does not use OTG functionality, you can set this input high to save power.
12	OTG	R/W	0	OTG Disable 1 The OTG block is powered down. 0 The OTG block is not powered down(default) This controller signal powers down the OTG block, including the VBUS Valid comparator. If the application does not use OTG functionality, you can set this input high to save power.
3-2	TXHSXVT	R/W	1	Transmitter High Speed Crossover Adjustment 3 The crossover voltage is increased by 15mV 2 The crossover voltage is increased by 30mV 1 Default setting 0 Reserved TXHSXVT adjusts the voltage at which the DP and DM signals cross while transmitting in HS mode
1	TXF	R/W	0	TXF Mode 1 Synchronize txf empty signal 0 Do not synchronize txf empty signal
0	ABE	R/W	0	TX Jitter Adjustment 1 TX Jitter Fix enable (default) 0 TX Jitter Fix disable

13.4 Programming Model

13.4.1 Overview

This chapter describes the programming requirements for the otg core in Host and Device modes. Each significant programming feature of the otg core is discussed in a separate section.

13.4.2 Core Initialization

The application must perform the core initialization sequence following the configuration parameters the otg core. If the cable is connected during powerup, the Current Mode of Operation bit in the Core Interrupt register (GINTSTS.CurMod) reflects the mode. The otg core enters Host mode when an “A” plug is connected, or Device mode when a “B” plug is connected.

This section explains the initialization of the otg core after power-on. The application must follow the initialization sequence irrespective of Host or Device mode operation. All core global registers are initialized according to the core’s configuration.

1. Read the User Hardware Configuration registers (GHWCFG1, 2, 3, and 4) to find the configuration parameters selected for otg core.

2. Program the following fields in the Global AHB Configuration (GAHBCFG) register.

- DMA Mode bit
- AHB Burst Length field
- Global Interrupt Mask bit = 1

- RxFIFO Non-Empty (GINTSTS.RxFLvl) (applicable only when the core is operating in Slave mode)
- non-periodic TxFIFO Empty Level (can be enabled only when the core is operating in Slave mode as a host)
- Periodic TxFIFO Empty Level (can be enabled only when the core is operating in Slave mode)

3. Program the following fields in GUSBCFG register.

- HNP Capable bit
- SRP Capable bit
- External HS PHY
- UTMI+ Selection bit
- PHY Interface bit
- HS/FS TimeOUT Time-Out Calibration field
- USB Turnaround Time field

4. The software must unmask the following bits in the GINTMSK register.

- OTG Interrupt Mask
- Mode Mismatch Interrupt Mask

5. If the GUID register is selected for implementation, the software has the option of programming this register.

6. The software can read the GINTSTS.CurMod bit to determine whether the otg core is operating in Host or Device mode. The software the follows either the “Host Initialization” or “Device Initialization” sequence.

13.4.3 Host Initialization

To initialize the core as host, the application must perform the following steps.

1. Program GINTMSK.PrtInt to unmask.

2. Program the HCFG register to select full-speed host or high-speed host.

3. Program the HPRT.PrtPwr bit to 1’b1. This drives VBUS on the USB.

4. Wait for the HPRT0.PrtConnDet interrupt. This indicates that a device is connect to the port.

5. Program the HPRT.PrtRst bit to 1’b1. This starts the reset process.

6. Wait at least 10 ms for the reset process to complete.

7. Program the HPRT.PrtRst bit to 1’b0.

8. Wait for the HPRT.PrtEnChng interrupt.

9. Read the HPRT.PrtSpd field to get the enumerated speed.

10. Program the HFIR register with a value corresponding to the selected PHY clock [29].

11. Program the RXFSIZE register to select the size of the receive.

12. Program the NPTXFSIZE register to select the size and the start address of the Nonperiodic Transmit FIFO for non-periodic.

13. Program the HPTXFSIZ register to select the size and start address of the Periodic.

Transmit FIFO for periodic transactions.

To communicate with devices, the system software must initialize and enable at least one channel.

[29] At this point, the host is up and running and the port register begins to report device disconnects, etc. The port is active with SOFs occurring down the enabled port.

13.4.4 Device Initialization

The application must perform the following steps to initialize the core as a device on powerup or after a mode change from Host to Device.

1. Program the following fields in the DCFG register.

- Device Speed
- Non-Zero-Length Status OUT Handshake
- Periodic Frame Interval (when periodic endpoints are supported)

2. Program the Device threshold control register. This is required only if you are using DMA mode and you are planning to enable thresholding.

3. Program the GINTMSK register to unmask the following interrupts.

- USB Reset
- Enumeration Done
- Early Suspend
- USB Suspend
- SOF

4. Wait for the GINTSTS.USBReset interrupt, which indicates a reset has been detected on the USB and lasts for about 10 ms. On receiving this interrupt, the application must perform the steps listed in “Initialization on USB Reset”.

5. Wait for the GINTSTS.EnumerationDone interrupt. This interrupt indicates the end of reset on the USB. On receiving this interrupt, the application must read the DSTS register to determine the enumeration speed and perform the steps listed in “Initialization on Enumeration Completion”. At this point, the device is ready to accept SOF packets and perform control transfers on control endpoint 0.

13.5 Modes of Operation

The application can operate the core either in DMA mode, where the core fetches the data to be transmitted or updates the received data on the AHB, or in Slave mode, where the application initiates transfers for data fetch and store. The application cannot operate the core using DMA and Slave modes simultaneously.

13.5.1 DMA Mode

With an internal DMA option, the otg host uses the AHB Master interface for transmit packet data fetch (AHB to USB) and receive data update (USB to AHB). The AHB Master uses the programmed DMA address (HCDMAn register in Host mode and DIEPWMAn/DOEPWMAn register in Device mode) to access the data buffers.
When Scatter/Gather DMA is enabled in device mode DIEPDMAn/DOEPDMAn registers are used to access the base descriptor.

13.5.1.1 Transfer-Level Operation

In DMA mode, the application is interrupted only after the programmed transfer size istransmitted or received (provided the otg core detects no NAK/NYET/Timeout/Error response in Host mode, or Timeout/CRC Error in Device mode). The application must handle all transaction errors. In Device mode, all the USB errors are handled by the core itself

13.5.1.2 Transaction-Level Operation

This mode is similar to transfer-level operation, with the programmed transfer size equal to one packet size (maximum size or short packet). When Scatter/Gather DMA is enabled, the transfer size is extracted from the descriptors.

13.5.2 Slave Mode

In Slave mode, the application can operate the otg core either in transaction-level (packet-level) operation or in pipelined transaction-level operation.

13.5.2.1 Transaction-Level Operation

The application handles one data packet at a time per channel/endpoint in transaction-level operations. Based on the handshake response received on the USB, the application determines whether to retry the transaction or proceed with the next, until the end of the transfer. The application is interrupted on completion of every packet. The application performs transaction-level operations for a channel/endpoint for a transmission (host: OUT/device: IN) or reception (host: IN/device: OUT).

[Host Mode]

For an OUT transaction, the application enables the channel and writes the data packet into the corresponding (Periodic or Non-periodic) transmit FIFO. The otg core automatically writes the channel number into the corresponding (Periodic or Non-periodic) Request Queue, along with the last DWORD write of the packet.

For an IN transaction, the application enables the channel and the otg core automatically writes the channel number into the corresponding Request queue. The application must wait for the packet received interrupt, then empty the packet from the receive FIFO.

[Device Mode]

For an IN transaction, the application enables the endpoint, writes the data packet into the corresponding transmit FIFO, and waits for the packet completion interrupt from the core.

For an OUT transaction, the application enables the endpoint, waits for the packet received interrupt from the core, then empties the packet from the receive FIFO.

Note

The application has to finish writing one complete packet before switching to a different channel/endpoint FIFO. Violating this rule results in an error.

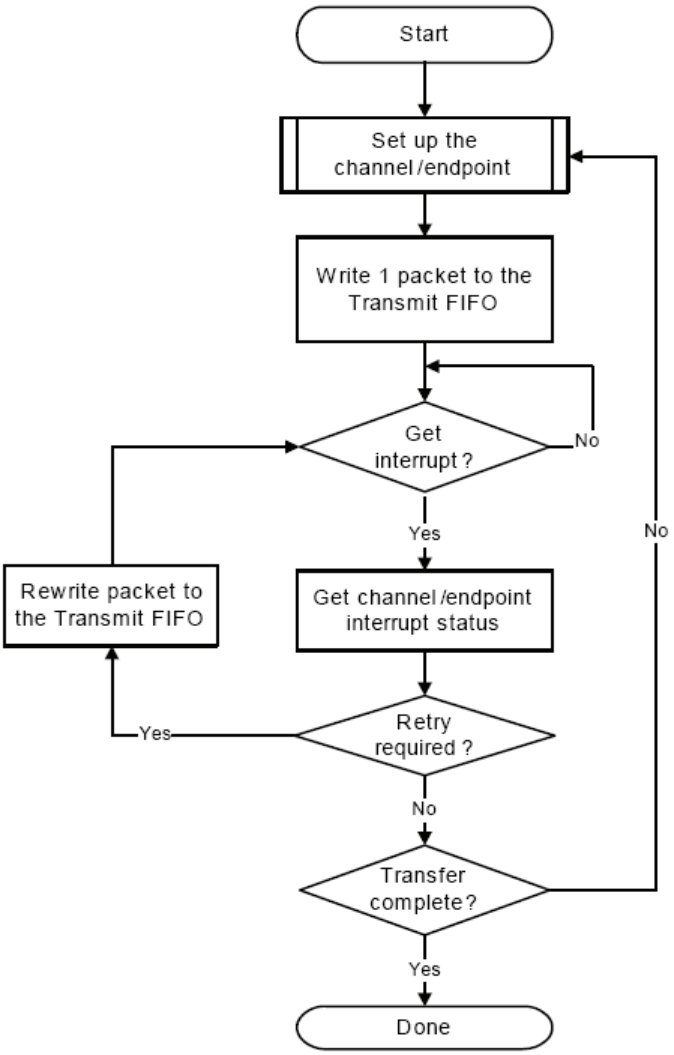


Figure 13.5 Transmit Transaction-Level Operation in Slave Mode

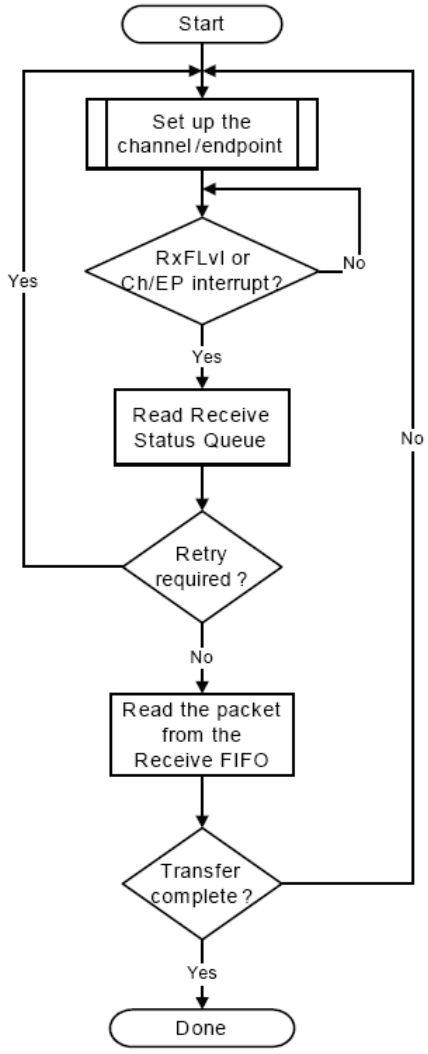


Figure 13.6 Receive Transaction-Level Operation in Slave Mode

13.5.2.2 Pipelined Transaction-Level Operation

The application can pipeline more than one transaction (IN or OUT) with pipelined transaction-level operation, which is analogous to Transfer mode in DMA mode. In pipelined transaction-level operation, the application can program the core to perform multiple transactions. The advantage of this mode of operation compared to transactionlevel operation is that the application is not interrupted on packet basis.

[Host Mode]

For an OUT transaction, the application sets up a transfer and enables the channel. The application can write multiple packets back-to-back for the same channel into the transmit FIFO, based on the space availability. It can also pipeline OUT transactions for multiple channels by writing into the HCHARn register, followed by a packet write to that channel. The core writes the channel number, along with the last DWORD write for the packet, into the Request queue and schedules transactions on the USB in the same order.
For an IN transaction, the application sets up a transfer and enables the channel, and the otg core writes the channel number into the Request queue. The application can schedule IN transactions on multiple channels, provided space is available in the Request queue. The core initiates an IN token on the USB only when there is enough space to receive at least of one maximum-packet-size packet of the channel in the top of the Request queue.

[Device Mode]

For an IN transaction, the application sets up a transfer and enables the endpoint. The application can write multiple packets back-to-back for the same endpoint into the transmit FIFO, based on available space. It can also pipeline IN transactions for multiple channels by writing into the DIEPCTLn register followed by a packet write to that endpoint. The core writes the endpoint number, along with the last DWORD write for the packet into the Request queue. The core transmits the data in the

transmit FIFO when an IN token is received on the USB.

For an OUT transaction, the application sets up a transfer and enables the endpoint. The core receives the OUT data into the receive FIFO, when it has available space. As the packets are received into the FIFO, the application must empty data from it.

From this point on in this chapter, the terms “Pipelined Transaction mode” and “Transfe mode” are used interchangeably.

13.5.3 Thresholding in DMA Mode

The application can program the core to do FIFO thresholding when operating as a device in DMA mode. With threshold support, the core can be configured to operate with less than maximum packet size FIFOs for a particular endpoint. This results in a smaller FIFO requirement when compared to non-thresholding mode.
When Scatter/Gather DMA is enabled and the application is setting up multiple transfers with one packet data scattered in more than one buffer, the application disables thresholding, to avoid reverting updated pointers in more than one descriptor on an underrun.
FIFO thresholding is supported only in DMA mode. FIFO thresholding is not supported when the core is operating as a host, even in DMA mode.

- The core allows both receive and transmit FIFO thresholding.
- Device Threshold Control Register bit DTHRCTL.RxThrRn must be set to enable receive thresholding. DTHRCTL.RxThrLen specifies the receive threshold size.
- Transmit uses separate Threshold Enable controls for isochronous and non-isochronous endpoints . Bits DTHRCTL.NonISOThrEn and DTHRCTL.ISOThrEn specify these Threshold Enable controls.
- The register filed DTHRCTL.TxThrLen specifies the transmit threshold length and is common for isochronous and non-isochronous endpoints. The minimum threshold length supported by the core is four DWORDs.
- Threshold enable controls cannot be changed randomly. The application can set or reset the threshold enable bits only after ensuring that the core is not programmed to do any transfers (FIFOs are flushed, NAK bits are set, all endpoints are disabled).
- One of the limitation of thresholding mode is in ping protocol, and could violate PING protocol. A PING token will be responded with an ACK handshake and the following OUT token could result in a NAK handshake. This behavior is a result of receive fifo overflow, and cannot be avoided in thresholding mode. This scenario will not occur ifthere are no overflows.

When transmit thresholding is enabled, the core starts transmitting data on the USB for a particular endpoint when there is threshold amount of data available in the corresponding transmit FIFO.

When receive thresholding is enabled, the core starts transferring data from the receive FIFO to the system memory as soon as there is threshold amount of data available in the receive FIFO. Any underrun or overflow conditions are handled by the core internally.

13.6 Host Programming Model

13.6.1 Channel Initialization

The application must initialize one or more channels before it can communicate with connected devices. To initialize and enable a channel, the application must perform the following steps.

1. Program the GINTMSK register to unmask the following:

- Channel Interrupt
- Non-periodic Transmit FIFO Empty for OUT transactions (applicable for Slave mode that operates in pipelined transaction-level with the Packet Count field programmed with more than one).
- Non-periodic Transmit FIFO Half-Empty for OUT transactions (applicable for Slave mode that operates in pipelined transaction-level with the Packet Count field programmed with more than one).

2. Program the HAINTMSK register to unmask the selected channels' interrupts.

3. Program the HCINTMSK register to unmask the transaction-related interrupts of interest given in the Host Channel Interrupt register.

4. Program the selected channel's HCTSIZn register with the total transfer size, in bytes, and the expected number of packets, including short packets. The application must program the PID field with the initial data PID (to be used on the first OUT transaction or to be expected from the first IN transaction).

5. Program the selected channels' HCSPLTn register(s) with the hub and port addresses (split transactions only).

6. Program the selected channels' HCDMAn register(s) with the buffer start address.

7. Program the HCCHARn register of the selected channel with the device's endpoint characteristics, such as type, speed, direction, and so forth. (The channel can be enabled by setting the Channel Enable bit to 1'b1 only when the application is ready to transmit or receive any packet).

Repeat steps 1–7 for other channels.

13.6.2 Halting a Channel

The application can disable any channel by programming the HCCHARn register with the HCCHARn.ChDis and HCCHARn.ChEna bits set to 1'b1. This enables the OTG host to flush the posted requests (if any) and generates a Channel Halted interrupt. The application must wait for the HCINTn.ChHltd interrupt before reallocating the channel for other transactions. The OTG host does not interrupt the transaction that has been already started on USB.
In Slave mode operation, before disabling a channel, the application must ensure that there is at least one free space available in the Non-periodic Request Queue (when disabling a nonperiodic channel) or the Periodic Request Queue (when disabling a periodic channel). The application can simply flush the posted requests when the Request queue is full (before disabling the channel), by programming the HCCHARn register with the HCCHARn.ChDis bit set to 1'b1, and the HCCHARn.ChEna bit reset to 1'b0.

To disable a channel in DMA mode operation, the application need not check for space in the Request queue. The OTG host checks for space in which to write the Disable request on the disabled channel's turn during arbitration. Meanwhile, all posted requests are dropped from the Request queue when the HCCHARn.ChDis bit is set to 1'b1.

The application is expected to disable a channel on any of the following conditions:

1. When a HCINTn.XferCompl interrupt is received during a non-periodic IN transfer or
high-bandwidth interrupt IN transfer (Slave mode only)

2. When a HCINTn.STALL, HCINTn.XactErr, HCINTn.BblErr, or HCINTn.DataTglErr interrupt is received for an IN or OUT channel (Slave mode only). For high-bandwidth interrupt INs in Slave mode, once the application has received a DataTglErr interrupt it must disable the channel and wait for a Channel Halted interrupt. The application must be able to receive other interrupts (DataTglErr, Nak, Data, XactErr, BabbleErr) for the same channel before receiving the halt.

3. When a GINTSTS.DisconnInt (Disconnect Device) interrupt is received. (The application is expected to disable all enabled channels in Slave and DMA modes)

4. When the application aborts a transfer before normal completion (Slave and DMA modes).

13.6.3 Ping Protocol

When the OTG host operates in high speed, the application must initiate the ping protocol when communicating with high-speed bulk or control (Data and Status stage) OUT endpoints. The application must initiate the ping protocol when it receives a NAK/NYET/XactErr interrupt. When the otg host receives one of the above responses, it does not continue any transaction for a specific endpoint, drops all posted or fetched OUT requests (from the Request queue), and flushes the corresponding data (from the transmit FIFO).

In Slave mode, the application can send a ping token either by setting the HCTSIZn.DoPng bit before enabling the channel or by just writing the HCTSIZn register with DoPng bit set when the channel is already enabled. This enables the otg host to write a ping request entry to the Request queue. The application must wait for the response to the ping token (a NAK, ACK, or XactErr interrupt) before continuing the transaction or sending another ping token. The application can continue the data transaction only after receiving an ACK from the OUT endpoint for the requested ping. The channel-specific interrupt service routine for the ping protocol in Slave mode is shown in table below.

In DMA mode operation, the application can start a ping protocol transfer by setting the HCTSIZn.DoPng bit before enabling the channel. The otg host continues sending ping tokens until receiving an ACK, then switches automatically to the data transaction.

Table 13.4 Interrupt Service Routine for Ping Protocol in Slave Mode

Ping Protocol for High Speed Bulk/Control OUT Endpoints

Unmask (ACK/NAK/XactErr/ChHltd/STALL)
if (ACK)
{
Reset Error Count
Re-initialize Channel (Do data transactions)
{
else if (NAK)
{
Reset Error Count
Send Ping
}
else if (STALL)
{
Disable Channel
}
else if (XactErr)
{
Increment Error Count
if (Error_count < 3)
{
Send Ping
}
else
{
Disable Channel
}
}
else if (ChHltd)
{
De-allocate Channel
}

13.6.4 Sending a Zero-Length Packet

To send a zero-length data packet, the application must initialize an OUT channel as follows.

1. Program the HCTSIZn register of the selected channel with a correct PID, XferSize = 0, and PktCnt = 1.

2. Program the HCCHARn register of the selected channel with ChEna = 1 and the device's endpoint characteristics, such as type, speed, and direction.

The application must treat a zero-length data packet as a separate transfer, and cannot combine it with a non-zero-length transfer.

13.6.5 Operational Model

The application must initialize a channel as described in “Channel Initialization” before communicating to the connected device. This section explains the sequence of operation to be performed for different types of USB transactions.

13.6.5.1 Writing the Transmit FIFO in Slave Mode

The figure below shows the flow diagram for writing to the transmit FIFO in Slave mode. The otg host automatically writes an entry (OUT request) to the Periodic/Non-periodic Request Queue, along with the last DWORD write of a packet. The application must ensure that at least one free space is available in the Periodic/Non-periodic Request Queue before starting to write to the transmit FIFO. The application must always write to the transmit FIFO in DWORDs. If the packet size is non-DWORD aligned, the application must use padding. The otg host determines the actual packet size based on the programmed maximum packet size and transfer size.

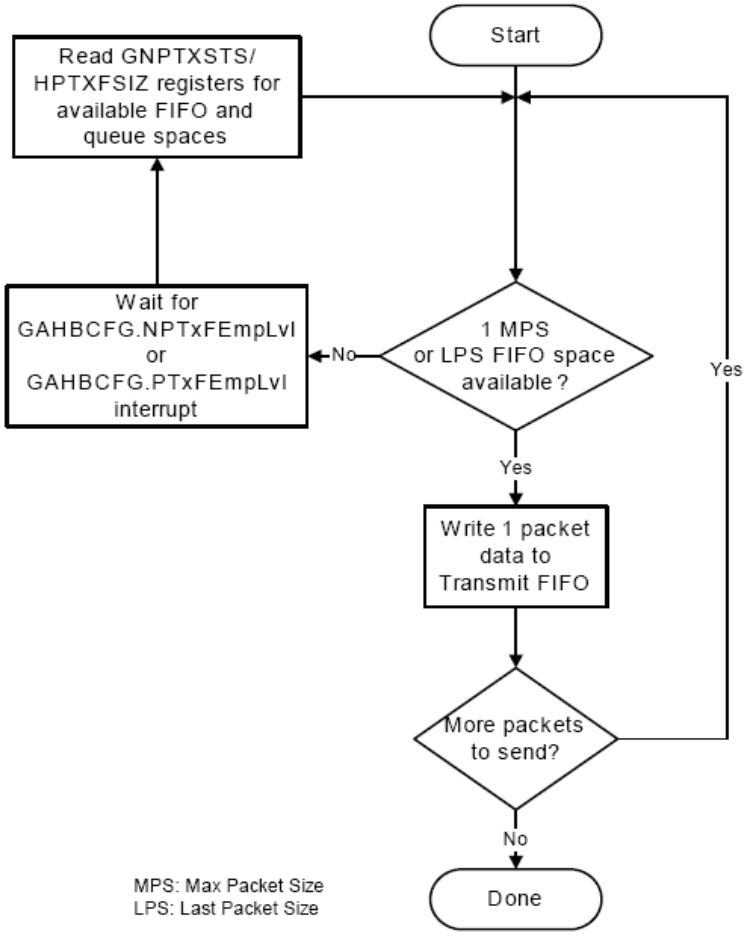


Figure 13.7 Transmit FIFO Write Task in Slave Mode

13.6.5.2 Reading the Receive FIFO in Slave Mode

The figure below shows the flow diagram for reading the receive FIFO in Slave mode. The application must ignore all packet statuses other than IN Data Packet (4'b0010).

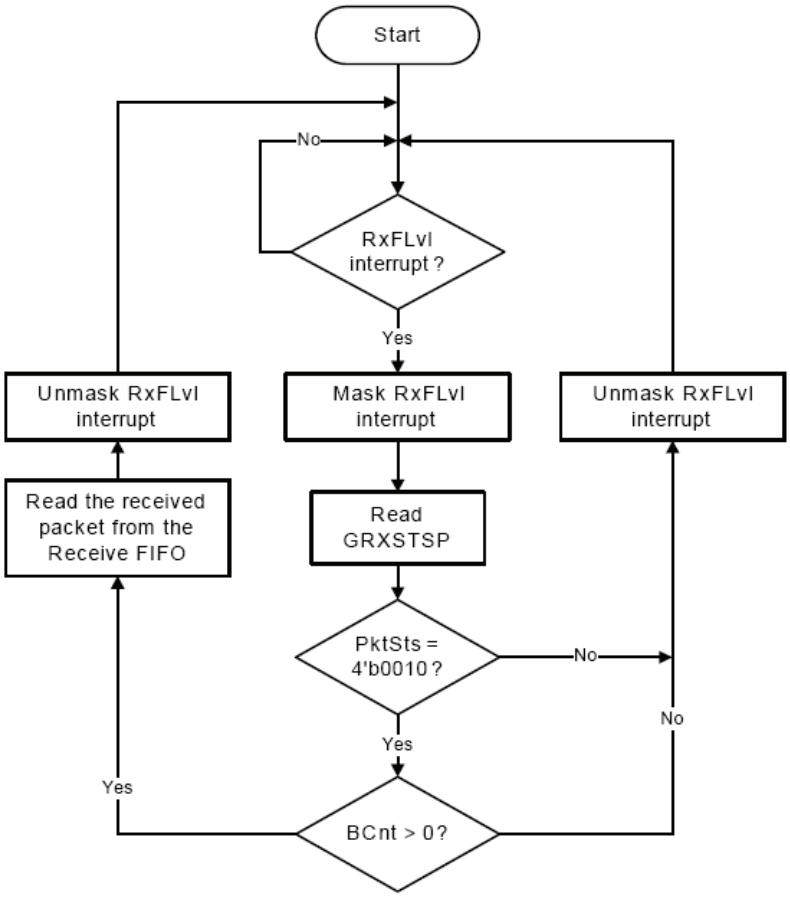


Figure 13.8 Receive FIFO Read Task in Slave Mode

13.6.5.3 Bulk and Control OUT/SETUP Transactions in Slave Mode

A typical bulk or control OUT/SETUP pipelined transaction-level operation in Slave mode is shown in figure below. See channel 1 (ch_1). Two bulk OUT packets are transmitted. A control SETUP transaction operates the same way but has only one packet. The assumptions are:

- The application is attempting to send two maximum-packet-size packets (transfer size = 1, 024 bytes).
- The Non-periodic Transmit FIFO can hold two packets (1 KB for HS or 128 KB for FS).
- The Non-periodic Request Queue depth = 4.

[Normal Bulk and Control OUT/SETUP Operations]

The sequence of operations in figure below(channel 1) is as follows:

1. Initialize channel 1 as explained in "Channel Initialization".

2. Write the first packet for channel 1.

3. Along with the last DWORD write, the core writes an entry to the Non-periodic Request Queue.

4. As soon as the non-periodic queue becomes non-empty, the core attempts to send an OUT token in the current frame/microframe.

5. Write the second (last) packet for channel 1.

6. The core generates the XferCompl interrupt as soon as the last transaction is completed successfully.

7. In response to the XferCompl interrupt, de-allocate the channel for other transfers.

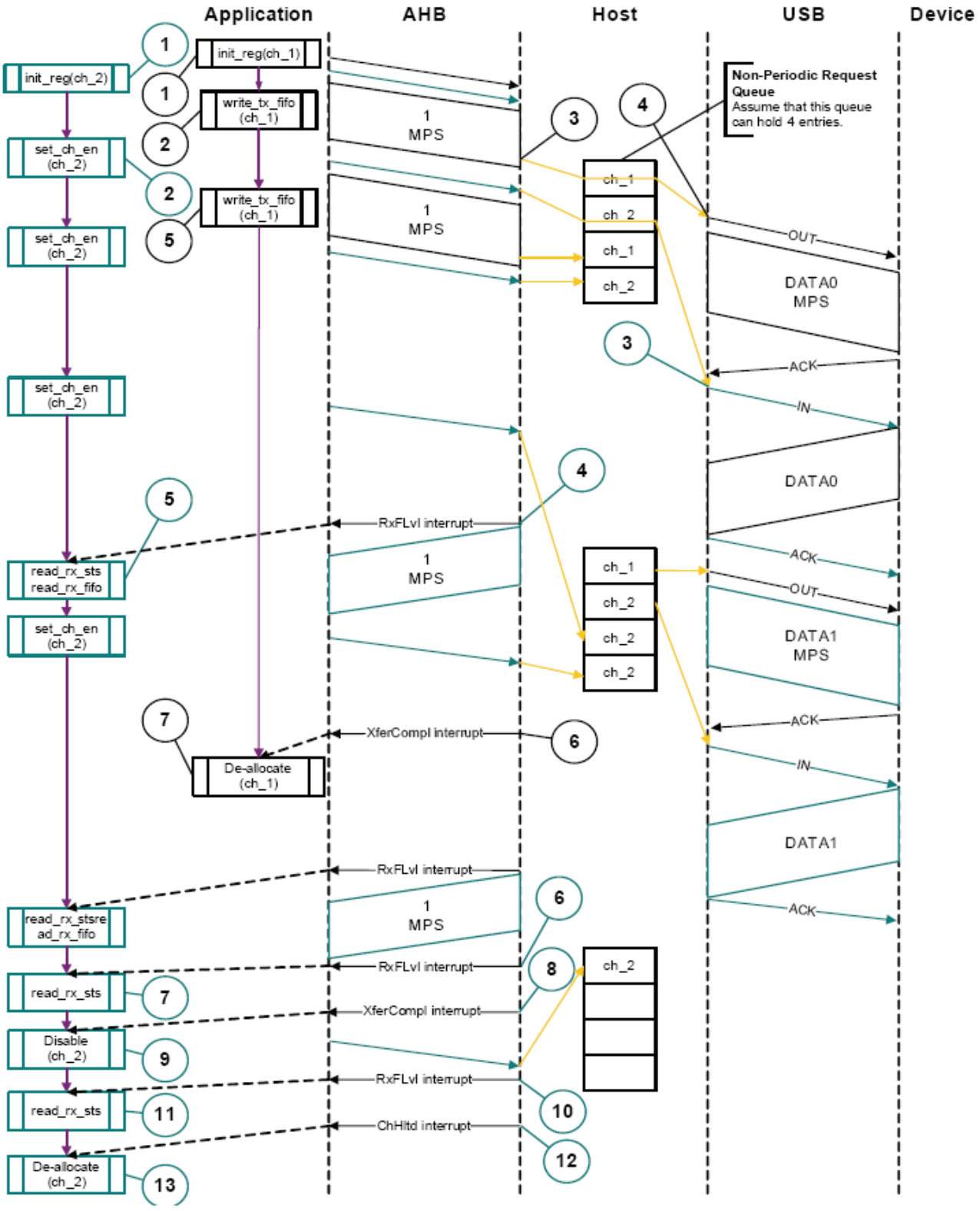


Figure 13.9 Normal Bulk/Control OUT/SETUP and Bulk/Control IN Transactions in Slave Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control OUT/SETUP transactions in Slave mode is shown in table below.

Table 13.5 Interrupt Service Routine for Bulk/Control OUT/SETUP and Bulk/Control IN Transactions in Slave Mode

Bulk/Control OUT/SETUP (Non-Split)	Bulk/Control IN (Non-Split)

Unmask (NAK/XactErr/NYET/STALL/XferCompl)
if (XferCompl)
{
Reset Error Count
Mask ACK
De-allocate Channel
}
else if (STALL)
{
Transfer Done = 1
Unmask ChHltd
Disable Channel
}
else if (NAK or XactErr or NYET)
{
Rewind Buffer Pointers
Unmask ChHltd
Disable Channel
if (XactErr)
{
Increment Error Count
Unmask ACK
}
else
{
Reset Error Count
}
}
else if (ChHltd)
{
Mask ChHltd
if (Transfer Done or (Error_count == 3))
{
De-allocate Channel
}
else
{
Re-initialize Channel (Do ping protocol for HS)
}
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
	Unmask (XactErr/XferCompl/BblErr/STALL/DataTglErr)
	if (XferCompl)
	{
	Reset Error Count
	Unmask ChHltd
	Disable Channel
	Reset Error Count
	Mask ACK
	}
	else if (XactErr or BblErr or STALL)
	{
	Unmask ChHltd
	Disable Channel
	if (XactErr)
	{
	Increment Error Count
	Unmask ACK
	}
	}
	else if (ChHltd)
	{
	Mask ChHltd
	if (Transfer Done or (Error_count == 3))
	{
	De-allocate Channel
	}
	else
	{
	Re-initialize Channel
	}
	}
	else if (ACK)
	{
	Reset Error Count
	Mask ACK
	}
	else if (DataTglErr)
	{
	Reset Error Count
	}

1. The application is expected to write the data packets into the transmit FIFO as and when the space is available in the transmit FIFO and the Request queue. The application can make use of	1. The application is expected to write the requests as and when the Request queue space is available and until the XferCompl interrupt is received.

GINTSTS.NPTxFEmp interrupt to find the transmit FIFO space.	2. The application must clear and never modify the DoPing bit after enabling the channel and until the XferCompl or ChHltd interrupt is received. The core uses the DoPing bit to flush the excessive IN requests after receiving the last or short packet.

13.6.5.4 Bulk and Control IN Transactions in Slave Mode

A typical bulk or control IN pipelined transaction-level operation in Slave mode is shown in figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive two maximum-packet-size packets (transfer size = 1, 024 bytes).
- The receive FIFO can contain at least one maximum-packet-size packet and two status DWORDs per packet (520 bytes for HS or 72 bytes for FS).
- The Non-periodic Request Queue depth = 4.

[Normal Bulk and Control IN Operations]

The sequence of operations in figure above (channel 2) is as follows:

1. Initialize channel 2 as explained in “Channel Initialization”.

2. Set the HCCHAR2.ChEna bit to write an IN request to the Non-periodic Request Queue.

3. The core attempts to send an IN token after completing the current OUT transaction.

4. The core generates an RxFLvl interrupt as soon as the received packet is written to the receive FIFO.

5. In response to the RxFLvl interrupt, mask the RxFLvl interrupt and read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. Following this, unmask the RxFLvl interrupt.

6. The core generates the RxFLvl interrupt for the transfer completion status entry in the receive FIFO.

7. The application must read and ignore the receive packet status when the receive packet status is not an IN data packet (GRXSTSR.PktSts != 4’b0010).

8. The core generates the XferCompl interrupt as soon as the receive packet status is read.

9. In response to the XferCompl interrupt, disable the channel (as explained in “Halting a Channel”) and stop writing the HCCHAR2 register for further requests. The core writes a channel disable request to the Non-periodic Request Queue as soon as the HCCHAR2 register is written.

10. The core generates the RxFLvl interrupt as soon as the halt status is written to the receive FIFO.

11. Read and ignore the receive packet status.

12. The core generates a ChHltd interrupt as soon as the halt status is popped from the receive FIFO.

13. In response to the ChHltd interrupt, de-allocate the channel for other transfers.

Note

For Bulk/Control IN transfer, the application is expected to write the requests when the Request queue space is available, and until the XferCompl interrupt is received.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control IN transactions in Slave mode is shown in the table above.

13.6.5.5 Bulk and Control OUT/SETUP Transactions IN DMA Mode

A typical bulk or control OUT/SETUP operation in DMA mode is shown in figure below. See channel 1 (ch_1). Two bulk OUT packets are transmitted. A control SETUP transaction operates the same way but has only one packet. The assumptions are:

- The application is attempting to send two maximum-packet-size packets (transfer size = 1, 024 bytes).
- The Non-periodic Transmit FIFO can hold two packets (1 KB for HS or 128 bytes for FS).
- The Non-periodic Request Queue depth = 4.

[Normal Bulk and Control OUT/SETUP Operations]

The sequence of operations in Figure 13.9 (channel 1) is as follows:

1. Initialize and enable channel 1 as explained in “Channel Initialization”.

2. The otg host starts fetching the first packet as soon as the channel is enabled. For internal DMA mode, the otg host uses the programmed DMA address to fetch the packet.

3. After fetching the last DWORD of the second (last) packet, the otg host masks channel 1 internally for further arbitration.

4. The otg host generates a ChHltd interrupt as soon as the last packet is received.

5. In response to the ChHltd interrupt, de-allocate the channel for other transfers.

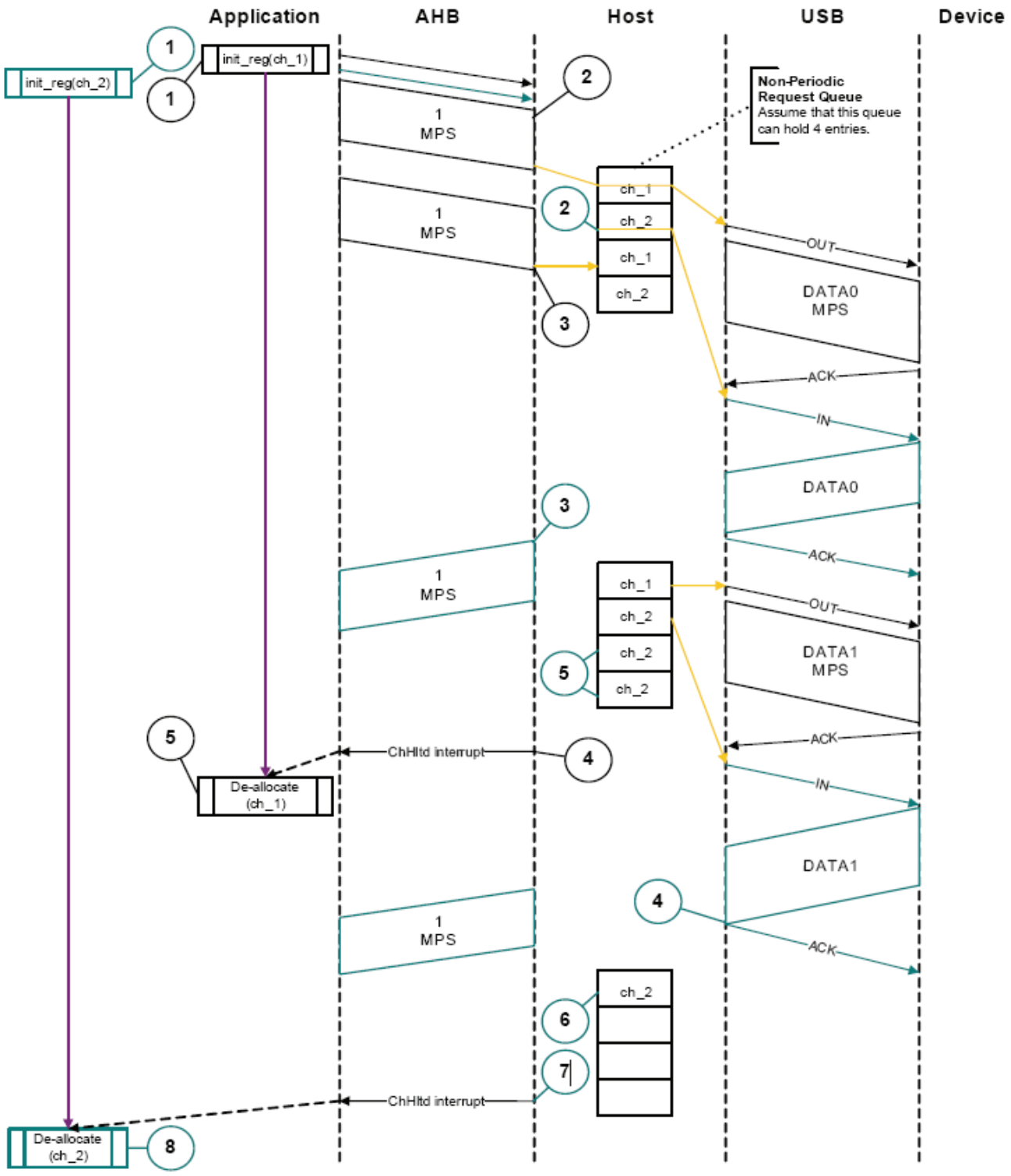


Figure 13.10 Normal Bulk/Control OUT/SETUP and Bulk/Control IN Transactions in DMA Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control OUT/SETUP transactions in DMA mode is shown in the table below.

Table 13.6 Interrupt Service Routines for Bulk/Control OUT/SETUP and Bulk/Control IN Transactions in DMA Mode

Bulk/Control OUT/SETUP (Non-Split)	Bulk/Control IN (Non-Split)

Unmask (ChHltd)
if (ChHltd)
{
if (XferCompl or STALL)
{
Reset Error Count
Mask ACK
De-allocate Channel
}
else if (NAK or XactErr or NYET)
{
Rewind Buffer Pointers
if (XactErr)
{
if (Error_count == 2)
{
De-allocate Channel
}
else
{
Increment Error Count
Re-initialize Channel
}
}
else
{
Reset Error Count
Mask ACK
}
Re-initialize Channel
}
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
	Unmask (ChHltd)
	if (ChHltd)
	{
	if (XferCompl or STALL or BblErr)
	{
	Reset Error Count
	Mask ACK
	De-allocate Channel
	}
	else if (XactErr)
	{
	if (Error_count == 2)
	{
	De-allocate Channel
	}
	else
	{
	Unmask ACK
	Unmask NAK
	Unmask DataTglErr
	Increment Error Count
	Re-initialize Channel
	}
	}
	}
	else if (ACK or NAK or DataTglErr)
	{
	Reset Error Count
	Mask ACK
	Mask NAK
	Mask DataTglErr
	}

1. As soon as the channel is enabled, the core attempts to fetch and write data packets, in multiples of the maximum packet size, to the transmit FIFO when space is available in the transmit FIFO and the Request queue. The core stops fetching as soon as the last packet is fetched. 2. While continuing the transfer to a high-speed device, the application must set the DoPing bit before enabling the channel if the previous transaction ended with NAK, NYET, or XacrErr	1. The application must clear and never modify the DoPing bit after enabling the channel and until the ChHltd interrupt is received. The core uses the DoPing bit to flush the excessive IN requests after receiving the last or short packet.

response. In this case, the core starts with the ping protocol, then automatically switches to Data Transfer mode.	

13.6.5.6 Bulk and Control IN Transactions in DMA Mode

A typical bulk or control IN operation in DMA mode is shown in Figure 5-6. See channel 2 (ch_2). The assumptions are:

- The application is attempting to send two maximum-packet-size packets (transfer size = 1, 024 bytes).
- The application is attempting to receive two maximum-packet-size packets (transfer size = 1, 024 bytes).
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet (520 bytes for HS or 72 bytes for FS).
- The Non-periodic Request Queue depth = 4.

[Normal Bulk and Control IN Operations]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”.

2. The otg host writes an IN request to the Request queue as soon as channel 2 receives the grant from the arbiter. (Arbitration is performed in a round-robin fashion, with fairness.).

3. The otg host starts writing the received data to the system memory as soon as the last byte is received with no errors.

4. When the last packet is received, the otg host sets an internal flag to remove any extra IN requests from the Request queue.

5. The otg host flushes the extra requests.

6. The final request to disable channel 2 is written to the Request queue. At this point, channel 2 is internally masked for further arbitration.

7. The otg host generates the ChHltd interrupt as soon as the disable request comes to the top of the queue.

8. In response to the ChHltd interrupt, de-allocate the channel for other transfers.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control IN transactions in DMA mode is shown in the table above.

13.6.5.7 Control Transactions in Slave Mode

Setup, Data, and Status stages of a control transfer must be performed as three separate transfers. Setup- Data- or Status-stage OUT transactions are performed similarly to the bulk OUT transactions explained in “Bulk and Control OUT/SETUP Transactions in Slave Mode”. Data- or Status-stage IN transactions are performed similarly to the bulk IN transactions explained in “Bulk and Control IN Transactions in Slave Mode”.

For all three stages, the application is expected to set the HCCHAR1.EPType field to Control. During the Setup stage, the application is expected to set the HCTSIZ1.PID field to SETUP.

13.6.5.8 Control Transactions in DMA Mode

Setup, Data, and Status stages of a control transfer must be performed as three separate transfers. Setup- and Data- or Status-stage OUT transactions are performed similarly to the bulk OUT transactions explained in “Bulk and Control OUT/SETUP Transactions in DMA Mode”. Data- or Status-stage IN transactions are performed similarly to the bulk IN transactions explained in “Bulk and Control IN Transactions in DMA Mode”.

For all three stages, the application is expected to set the HCCHAR1.EPType field to Control. During the Setup stage, the application is expected to set the HCTSIZ1.PID field to SETUP.

13.6.5.9 Interrupt OUT Transactions in Slave Mode

A typical interrupt OUT operation in Slave mode is shown in the figure below. See channel 1 (ch_1). The assumptions are:

- The application is attempting to send one packet in every frame/microframe (up to 1 maximum packet size), starting with the odd frame/microframe (transfer size = 1, 024 bytes).
- The Periodic Transmit FIFO can hold one packet (1 KB for HS or FS).
- Periodic Request Queue depth = 4.

[Normal Interrupt OUT Operation]

The sequence of operations in the figure below (channel 1) is as follows:

1. Initialize and enable channel 1 as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit.

2. Write the first packet for channel 1. For a high-bandwidth interrupt transfer, the application must write the subsequent packets up to MC (maximum number of packets to be transmitted in the next frame/microframe times before switching to another channel.

3. Along with the last DWORD write of each packet, the otg host writes an entry to the Periodic Request Queue.

4. The otg host attempts to send an OUT token in the next (odd) frame/microframe.

5. The otg host generates an XferCompl interrupt as soon as the last packet is transmitted successfully.

6. In response to the XferCompl interrupt, reinitialize the channel for the next transfer.

[Handling Non-ACK Responses]

The table below shows the channel-specific ISR for an interrupt OUT transaction in Slave mode.

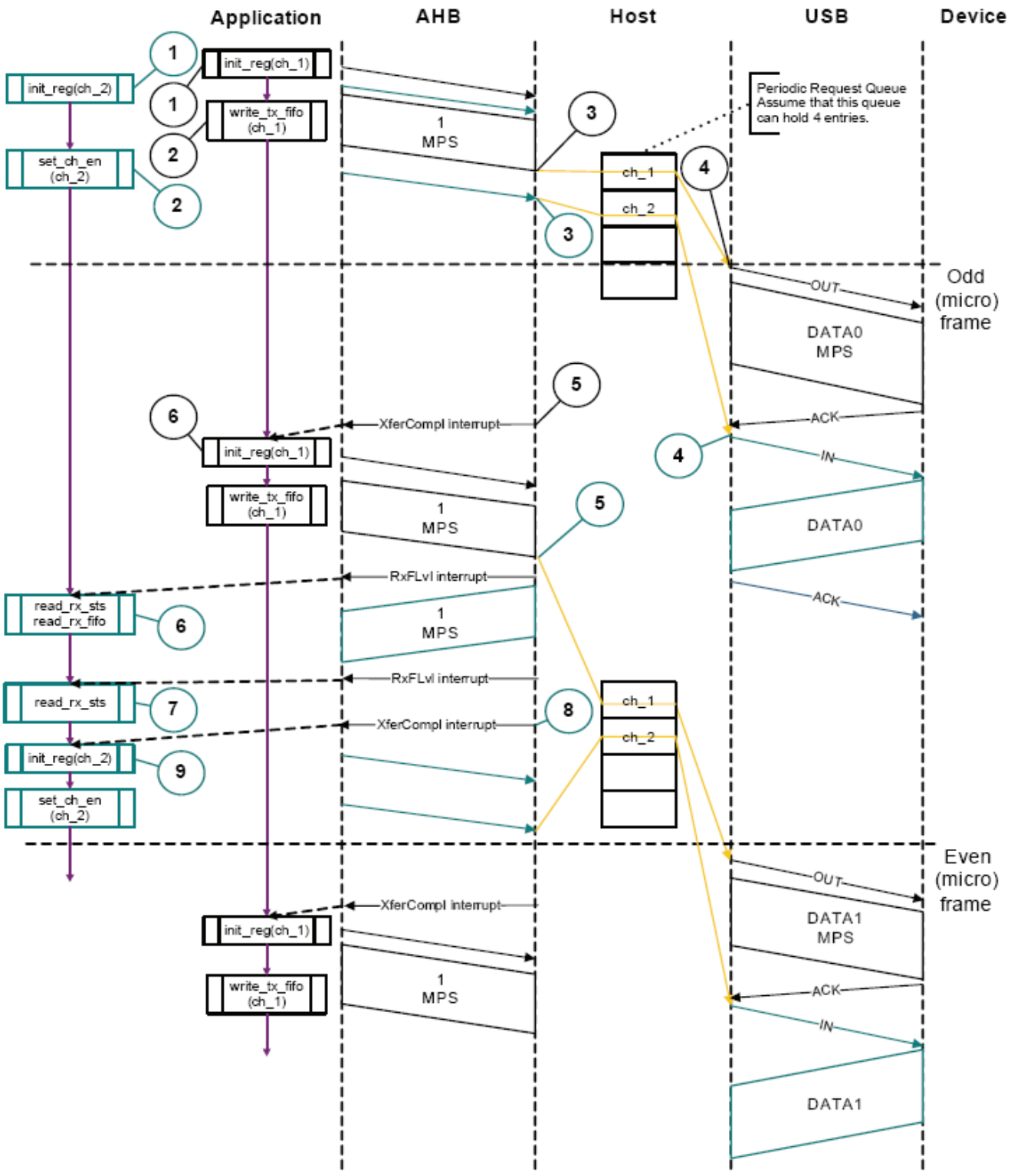


Figure 13.11 Normal Interrupt OUT/IN Transactions in Slave Mode

Table 13.7 Interrupt Service Routine for Interrupt OUT/IN Transactions in Slave Mode

Interrupt OUT (Non-Split)	Interrupt IN (Non-Split)

Unmask
(NAK/XactErr/STALL/XferCompl/FrmOvrun)
if (XferCompl)
{
Reset Error Count
Mask ACK
De-allocate Channel
}
else if (STALL or FrmOvrun)
{
Mask ACK
Unmask ChHltd
Disable Channel
if (STALL)
{
Transfer Done = 1
}
}
else if (NAK or XactErr)
{
Rewind Buffer Pointers
Reset Error Count
Mask ACK
Unmask ChHltd
Disable Channel
}
else if (ChHltd)
{
Mask ChHltd
if (Transfer Done or (Error_count == 3))
{
De-allocate Channel
}
else
{
Re-initialize Channel (in next b_interval - 1 uF/F)
}
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
	Unmask
	(NAK/XactErr/XferCompl/BblErr/STALL/FrmOvrun/DataTglErr)
	if (XferCompl)
	{
	Reset Error Count
	Mask ACK
	if (HCTSIZn.PktCnt == 0)
	{
	De-allocate Channel
	}
	else
	{
	Transfer Done = 1
	Unmask ChHltd
	Disable Channel
	}
	}
	else if (STALL or FrmOvrun or NAK or DataTglErr or BblErr)
	{
	Mask ACK
	Unmask ChHltd
	Disable Channel
	if (STALL or BblErr)
	{
	Reset Error Count
	Transfer Done = 1
	}
	else if (!FrmOvrun)
	{
	Reset Error Count
	}
	}
	else if (XactErr)
	{
	Increment Error Count
	Unmask ACK
	Unmask ChHltd
	Disable Channel
	}
	else if (ChHltd)
	{
	Mask ChHltd
	if (Transfer Done or (Error_count == 3))
	{
	De-allocate Channel
	}
	else
	{
	Re-initialize Channel (in next b_interval - 1 uF/F)
	}
	}
	else if (ACK)
	{
	Reset Error Count
	Mask ACK
	}

1. The application is expected to write the data packets into the transmit FIFO when the space is available in the transmit FIFO and the Request queue up to the count specified in the MC field before switching to another channel. The application uses the GINTSTS.NPTxFEmp interrupt to find the transmit FIFO space.	1. The application is expected to write the requests for the same channel when the Request queue space is available up to the count specified in the MC field before switching to another channel (if any).

13.6.5.10 Interrupt IN Transactions in Slave Mode

A typical interrupt IN operation in Slave mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one packet (up to 1 maximum packet size) in every frame/microframe, starting with odd. (transfer size = 1, 024 bytes).
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet (1, 032 bytes for HS or 1, 031 bytes for FS).
- Periodic Request Queue depth = 4.

[Normal Interrupt IN Operation]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize channel 2 as explained in “Channel Initialization”. The application must set the HCCHAR2.OddFrm bit.

2. Set the HCCHAR2.ChEna bit to write an IN request to the Periodic Request Queue. For a high-bandwidth interrupt transfer, the application must write the HCCHAR2 register MC (maximum number of expected packets in the next frame/microframe) times before switching to another channel.

3. The otg host writes an IN request to the Periodic Request Queue for each HCCHAR2 register write with a ChEna bit set.

4. The otg host attempts to send an IN token in the next (odd) frame/microframe.

5. As soon as the IN packet is received and written to the receive FIFO, the otg host generates an RxFLvl interrupt.

6. In response to the RxFLvl interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. The application must mask the RxFLvl interrupt before reading the receive FIFO, and unmask after reading the entire packet.

7. The core generates the RxFLvl interrupt for the transfer completion status entry in the receive FIFO. The application must read and ignore the receive packet status when the receive packet status is not an IN data packet (GRXSTSR.PktSts != 4’b0010).

8. The core generates an XferCompl interrupt as soon as the receive packet status is read.

9. In response to the XferCompl interrupt, read the HCTSIZ2.PktCnt field. If HCTSIZ2.PktCnt != 0, disable the channel (as explained in “Halting a Channel”) before re-initializing the channel for the next transfer, if any). If HCTSIZ2.PktCnt == 0, reinitialize the channel for the next transfer. This time, the application must reset the HCCHAR2.OddFrm bit.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for an interrupt IN transaction in Slave mode is shown in the Table above.

13.6.5.11 Interrupt OUT Transactions in DMA Mode

A typical interrupt OUT operation in DMA mode is shown in figure below. See channel 1 (ch_1). The assumptions are:

- The application is attempting to transmit one packet in every frame/microframe (up to 1 maximum packet size of 1, 024 bytes).
- The Periodic Transmit FIFO can hold one packet (1 KB for HS/FS).
- Periodic Request Queue depth = 4.

[Normal Interrupt OUT Operation]

1. Initialize and enable channel 1 as explained in “Channel Initialization”.

2. The otg host starts fetching the first packet as soon the channel is enabled and writes the OUT request along with the last DWORD fetch. In high-bandwidth transfers, the otg host continues fetching the next packet (up to the value specified in the MC field) before switching to the next channel.

3. The otg host attempts to send the OUT token in the beginning of the next odd frame/microframe.

4. After successfully transmitting the packet, the otg host generates a ChHltd interrupt.

5. In response to the ChHltd interrupt, reinitialize the channel for the next transfer.

[Handling Non-ACK Responses]

Table below shows the channel-specific ISR for an interrupt OUT transaction in DMA mode.

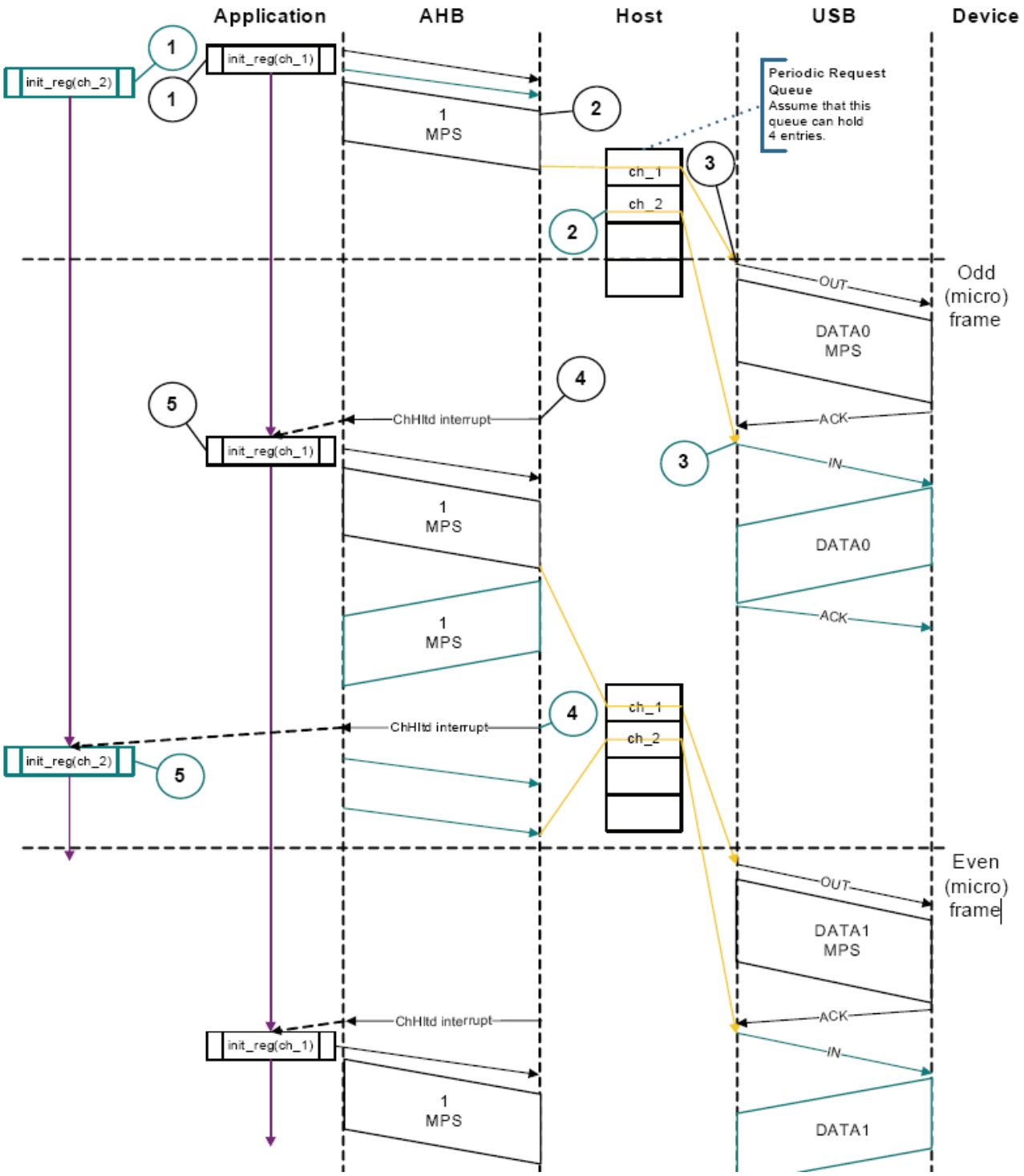


Figure 13.12 Normal Interrupt OUT/IN Transactions in DMA Mode

Table 13.8 Interrupt Service Routine for Interrupt OUT/IN Transactions in DMA Mode

Interrupt OUT (Non-Split)	Interrupt IN (Non-Split)

Unmask (ChHltd)
if (ChHltd)
{
if (XferCompl)
{
Reset Error Count
Mask ACK
if (Transfer Done)
{
De-allocate Channel
}
else
{
Re-initialize Channel (in next b_interval - 1 uF/F)
}
}
else if (STALL)
{
Transfer Done = 1
Reset Error Count
Mask ACK
De-allocate Channel
}
else if (NAK or FrmOvrun)
{
Mask ACK
Rewind Buffer Pointers
Re-initialize Channel (in next b_interval - 1 uF/F)
if (NAK)
{
Reset Error Count
}
}
else if (XactErr)
{
if (Error_count == 2)
{
De-allocate Channel
}
else
{
Increment Error Count
Rewind Buffer Pointers
Unmask ACK
Re-initialize Channel (in next b_interval - 1 uF/F)
}
}
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
	Unmask (ChHltd)
	if (ChHltd)
	{
	if (XferCompl)
	{
	Reset Error Count
	Mask ACK
	if (Transfer Done)
	{
	De-allocate Channel
	}
	else
	{
	Re-initialize Channel (in next b_interval - 1 uF/F)
	}
	}
	else if (STALL or BblErr)
	{
	Reset Error Count
	Mask ACK
	De-allocate Channel
	}
	else if (NAK or DataTglErr or FrmOvrun)
	{
	Mask ACK
	Re-initialize Channel (in next b_interval - 1 uF/F)
	if (DataTglErr or NAK)
	{
	Reset Error Count
	}
	}
	else if (XactErr)
	{
	if (Error_count == 2)
	{
	De-allocate Channel
	}
	else
	{
	Increment Error Count
	Unmask ACK
	Re-initialize Channel (in next b_interval - 1 uF/F)
	}
	}
	}
	else if (ACK)
	{
	Reset Error Count
	Mask ACK
	}

1. As soon as the channel is enabled, the core attempts to fetch and write data packets, in maximum packet size multiples, to the transmit FIFO when the space is available in the transmit FIFO and the Request queue. The core stops fetching as soon as the last packet is fetched (the number of packets is determined by the MC field of the HCCHARn register).	1. As soon as the channel is enabled, the core attempts to write the requests into the Request queue when the space is available up to the count specified in the MC field.

13.6.5.12 Interrupt IN Transactions in DMA Mode

A typical isochronous IN operation in DMA mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one packet in every frame/microframe (up to 1 maximum packet size of 1, 024 bytes).
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet (1, 032 bytes for HS/FS).
- Periodic Request Queue depth = 4.

[Normal Interrupt IN Operation]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”.

2. The otg host writes an IN request to the Request queue as soon as the channel 2 gets the grant from the arbiter (round-robin with fairness). In high-bandwidth transfers, the otg host writes consecutive writes up to MC times.

3. The otg host attempts to send an IN token at the beginning of the next (odd) frame/microframe.

4. As soon the packet is received and written to the receive FIFO, the otg host generates a ChHltd interrupt.

5. In response to the ChHltd interrupt, reinitialize the channel for the next transfer.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for Interrupt IN transaction in DMA mode is shown in the table above.

13.6.5.13 Isochronous OUT Transactions in Slave Mode

A typical isochronous OUT operation in Slave mode is shown in figure below. See channel 1 (ch_1). The assumptions are: The application is attempting to send one packet every frame/microframe (up to 1 maximum packet size), starting with an odd frame/microframe. (transfer size = 1, 024 bytes).

- The Periodic Transmit FIFO can hold one packet (1 KB for HS/FS).
- Periodic Request Queue depth = 4.

[Normal Isochronous OUT Operation]

The sequence of operations in the figure below (channel 1) is as follows:

1. Initialize and enable channel 1 as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit.

2. Write the first packet for channel 1. For a high-bandwidth isochronous transfer, the application must write the subsequent packets up to MC (maximum number of packets to be transmitted in the next frame/microframe) times before switching to another channel.

3. Along with the last DWORD write of each packet, the otg host writes an entry to the Periodic Request Queue.

4. The otg host attempts to send the OUT token in the next frame/microframe (odd).

5. The otg host generates the XferCompl interrupt as soon as the last packet is transmitted successfully.

6. In response to the XferCompl interrupt, reinitialize the channel for the next transfer.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for an isochronous OUT transaction in Slave mode is shown in table below.

Table 13.9 Interrupt Service Routine for Isochronous OUT/IN Transactions in Slave Mode

Isochronous OUT (Non-Split)	Isochronous IN (Non-Split)

Unmask (FrmOvrun/XferCompl)
if (XferCompl)
{
De-allocate Channel
}
else if (FrmOvrun)
{
Unmask ChHltd
Disable Channel
}
else if (ChHltd)
{
Mask ChHltd
De-allocate Channel
}
	Unmask (XactErr/XferCompl/FrmOvrun/BblErr)
	if (XferCompl or FrmOvrun)
	{
	if (XferCompl and (HCTSIZn.PktCnt == 0))
	{
	Reset Error Count
	De-allocate Channel
	}
	else
	{
	Unmask ChHltd
	Disable Channel
	}
	}
	else if (XactErr or BblErr)
	{
	Increment Error Count
	Unmask ChHltd
	Disable Channel
	}
	else if (ChHltd)
	{
	Mask ChHltd
	if (Transfer Done or (Error_count == 3))
	{
	De-allocate Channel
	}
	else
	{
	Re-initialize Channel
	}
	}

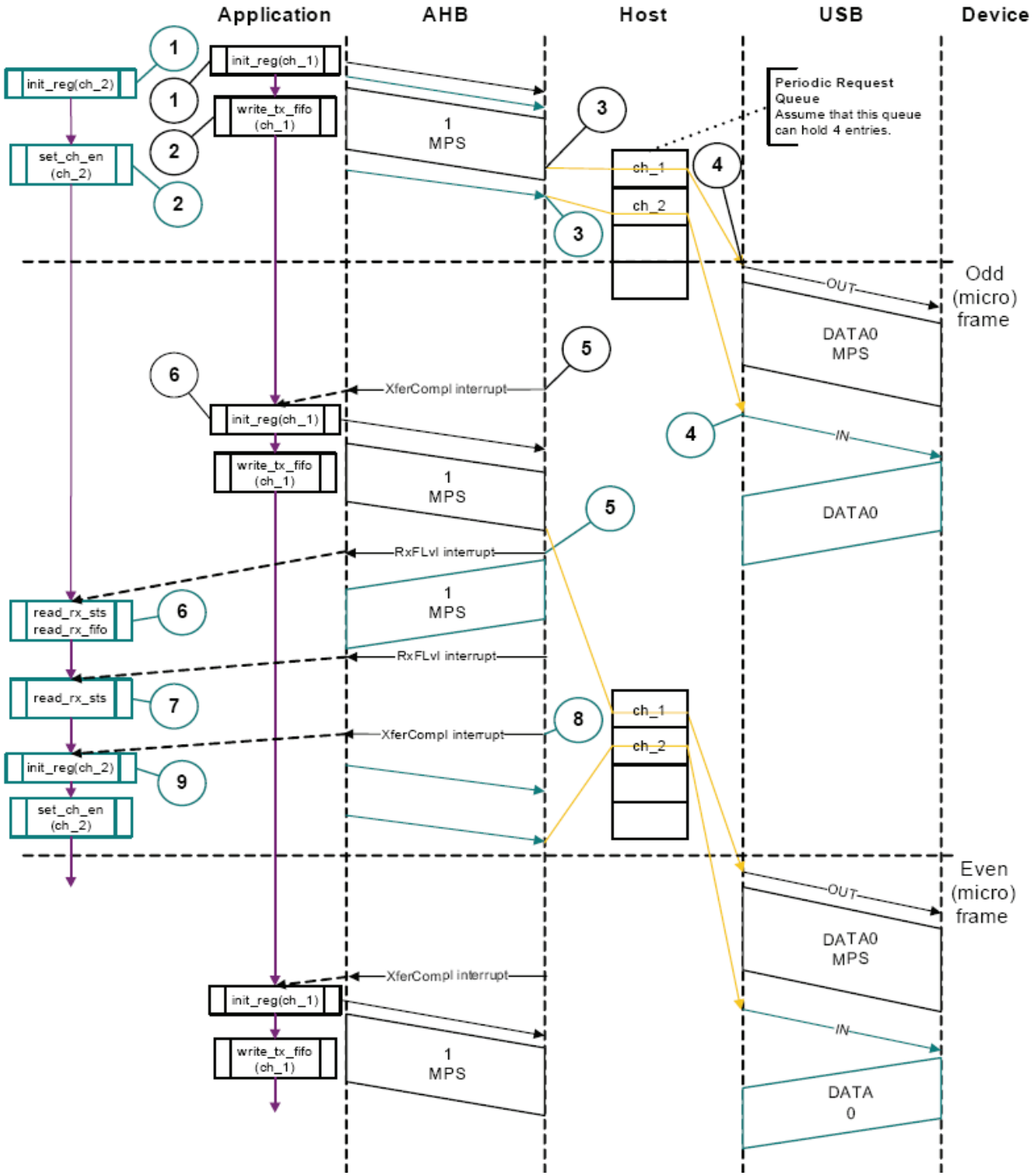


Figure 13.13 Normal Isochronous OUT/IN Transactions in Slave Mode

13.6.5.14 Isochronous IN Transactions in DMA Mode

A typical isochronous IN operation in DMA mode is shown in figure below. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one packet in every frame/microframe (up to 1 maximum packet size of 1, 024 bytes).
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDS per packet (1, 032 bytes for HS/FS).
- Periodic Request Queue depth = 4.

[Normal Isochronous IN Operation]

The sequence of operations in the figure above(channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”.

2. The otg host writes an IN request to the Request queue as soon as the channel 2 gets the grant from the arbiter (round-robin with fairness). In high-bandwidth transfers, the otg host performs consecutive writes up to MC times.

3. The otg host attempts to send an IN token at the beginning of the next (odd) frame/microframe.

4. As soon the packet is received and written to the receive FIFO, the OTG host generates a ChHltd interrupt.

5. In response to the ChHltd interrupt, reinitialize the channel for the next transfer.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for an isochronous IN transaction in DMA

mode is shown in the table above.

13.6.5.15 Bulk and Control OUT/SETUP Split Transactions in Slave Mode

A typical bulk or control SETUP/OUT operation in Slave mode is shown in figure below. See channel 1 (ch_1). Two bulk OUT packets are transmitted. A control SETUP transaction operates the same way but has only one packet. The assumptions are:

- The application is attempting to transmit one packet.

[Normal Bulk and Control OUT/SETUP Split Operations]

The sequence of operations in figure below (channel 1) is as follows:

1. Initialize and enable channel 1 as explained in “Channel Initialization”.

2. Write the packet for channel 1. Along with the last DWORD write of the packet, the otg host writes an entry to the Periodic Request Queue.

3. The otg host sends an OUT token.

4. The otg host generates an ACK interrupt as soon as the start split transaction completes successfully.

5. In response to the ACK interrupt, set the HCSPLT1.ComplSplt to send the complete split.

6. The otg host sends out the complete split transaction.

7. The otg host generates the XferCompl interrupt after successfully completing the complete split transaction.

8. In response to XferCompl interrupt, de-allocate the channel.

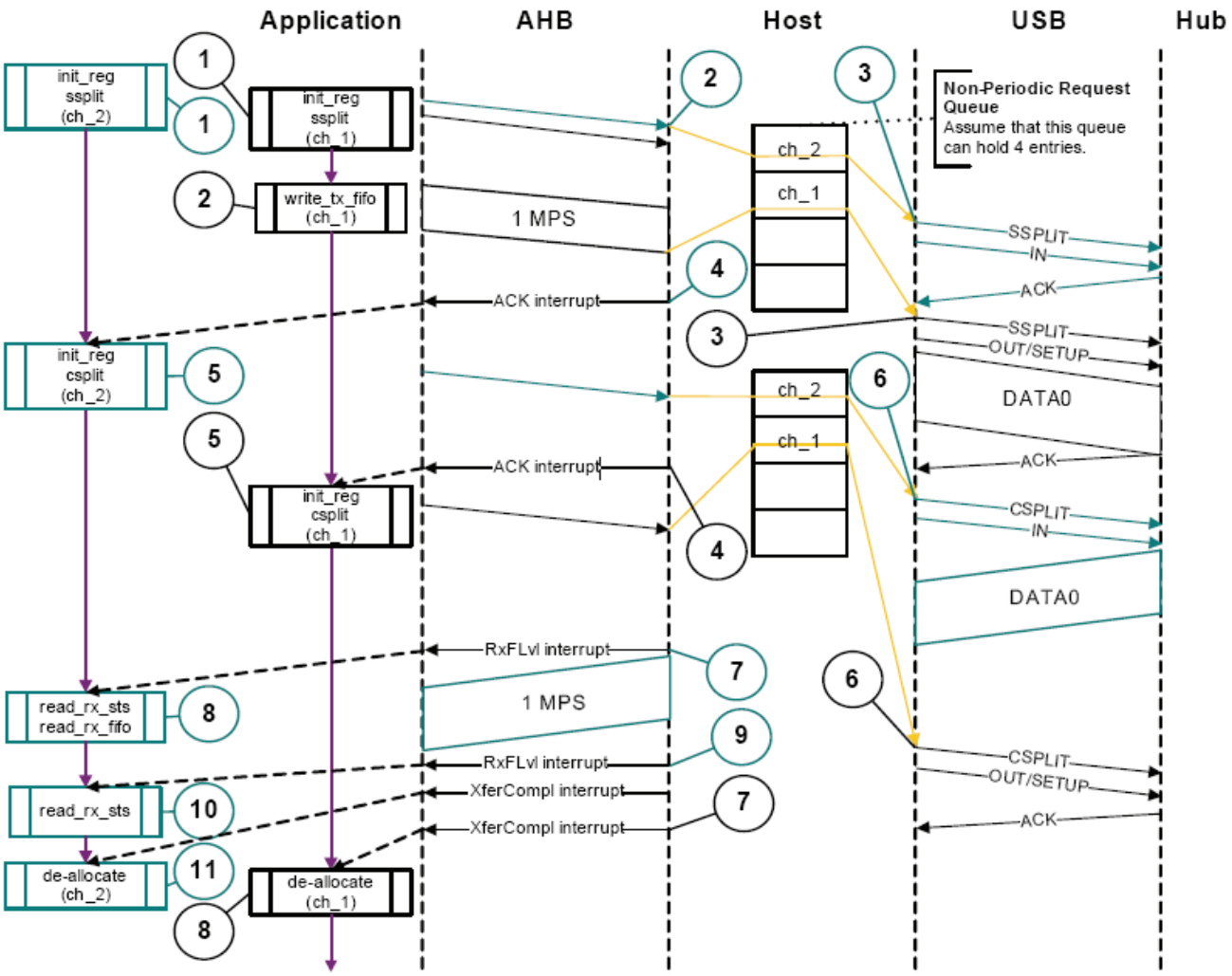


Figure 13.14 Normal Bulk/Control OUT/SETUP and Bulk/Control IN Split Transactions in Slave Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control OUT/SETUP split transactions in Slave mode is shown in table below.

Table 13.10 Interrupt Service Routine for Bulk/Control OUT/SETUP and Bulk/Control IN Split Transactions in Slave Mode

	Bulk/Control OUT/SETUP (Split)	Bulk/Control IN (Split)

Start Split	if (ACK)
	{
	Reset Error Count
	Do Complete Split
	}
	else if (NAK)
	{
	Rewind Buffer Pointers
	Retry Start Split
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	Increment Error Count
	if (Error_count < 3)
	{
	Retry Start Split
	}
	else
	{
	Unmask ChHltd
	Disable Channel
	}
	}
	else if (ChHltd)
	{
	Mask ChHltd
	De-allocate Channel
	}
		Unmask (ACK/NAK/XactErr/DataTglErr)
		if (ACK)
		{
		Reset Error Count
		Do Complete Split
		}
		else if (NAK)
		{
		Retry Start Split
		}
		else if (XactErr/DataTglErr)
		{
		Increment Error Count
		if (Error_count < 3)
		{
		Retry Start Split
		}
		else
		{
		Unmask ChHltd
		Disable Channel
		}
		}
		else if (ChHltd)
		{
		Mask ChHltd
		De-allocate Channel
		}

Complete Split	Unmask
	(NAK/XactErr/NYET/STALL/XferCompl)
	if (XferCompl)
	{
	De-allocate Channel
	}
	else if (NAK)
	{
	Rewind Buffer Pointers
	Retry Start Split
	}
	else if (NYET)
	{
	Retry Complete Split
	}
	else if (STALL)
	{
	Unmask ChHltd
	Disable Channel
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	Increment Error Count
	if (Error_count < 3)
	{
	Retry Start Split
	}
	else
	{
	Unmask ChHltd
	Disable Channel
	}
	}
		Unmask
		(NAK/XactErr/NYET/STALL/XferCompl)
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NAK)
		{
		Retry Start Split
		}
		else if (NYET)
		{
		Retry Complete Split
		}
		else if (STALL or BblErr)
		{
		Unmask ChHltd
		Disable Channel
		}
		else if (XactErr)
		{
		Increment Error Count
		if (Error_count < 3)
		{
		Retry Start Split
		}
		else
		{
		Unmask ChHltd
		Disable Channel
		}
		}
		else if (ChHltd)
		{

	else if (ChHltd)
	{
	Mask ChHltd
	De-allocate Channel
	}
		Mask ChHltd
		De-allocate Channel
		}

13.6.5.16 Bulk and Control IN Split Transactions in Slave Mode

A typical bulk or control IN operation in Slave mode is shown in the figure above (see channel 2 [ch_2). The assumptions are:

- The application is attempting to receive one packet.

[Normal Bulk and Control IN Split Operations]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”.

2. The otg host writes the Start Split request to the Periodic Request Queue as soon as the HCCHAR2 register is written.

3. The otg host sends the Start Split IN token.

4. As soon as the IN token is transmitted, the otg host generates an ACK interrupt.

5. In response to the ACK interrupt, set the HCSPLT2.ComplSplt bit to send the complete split token.

6. The otg host sends the complete split token.

7. As soon as the received packet is written to the receive FIFO, the otg host generates the RxFLvl interrupt.

8. In response to the RxFLvl interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. The application must mask the RxFLvl interrupt before reading the receive FIFO and unmask it after reading the entire packet.

9. The core generates the RxFLvl interrupt for the transfer completion status entry in the receive FIFO.

10. The application must read the receive packet status and, when the receive packet status is not an IN data packet (GRXSTSR.PktSts != 4’b0010), ignore it.

11. The core generates the XferCompl interrupt as soon as the receive packet status is read. In response to the XferCompl interrupt, read the HCTSIZ2.PktCnt field. If HCTSIZ2.PktCnt != 0, disable the channel (as explained in “Halting a Channel”) before re-initializing the channel for the next transfer, if any.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control IN split transactions inSlave mode is shown in the table above.

13.6.5.17 Bulk and Control OUT/SETUP Split Transactions in DMA Mode

A typical bulk or control OUT/SETUP operation in DMA mode is shown in figure below. See channel 1 (ch_1). Two bulk OUT packets are transmitted. A control SETUP transaction operates the same way but has only one packet. It is assumed that the application is attempting to transmit one packet.

[Normal Bulk and Control OUT/SETUP Split Operations]

The sequence of operations in figure below (channel 1) is as follows:

Initialize and enable channel 1 for start split as explained in “Channel Initialization”.
The host starts fetching the first packet as soon the channel is enabled and writes the OUT request along with the last DWORD fetch.

After successfully transmitting start split, the OTG host generates the ChHltd interrupt.

In response to the ChHltd interrupt, set the HCSPLT1.ComplSplt bit to send the complete split.

After successfully transmitting complete split, the OTG host generates the ChHltd interrupt.

In response to the ChHltd interrupt, de-allocate the channel.

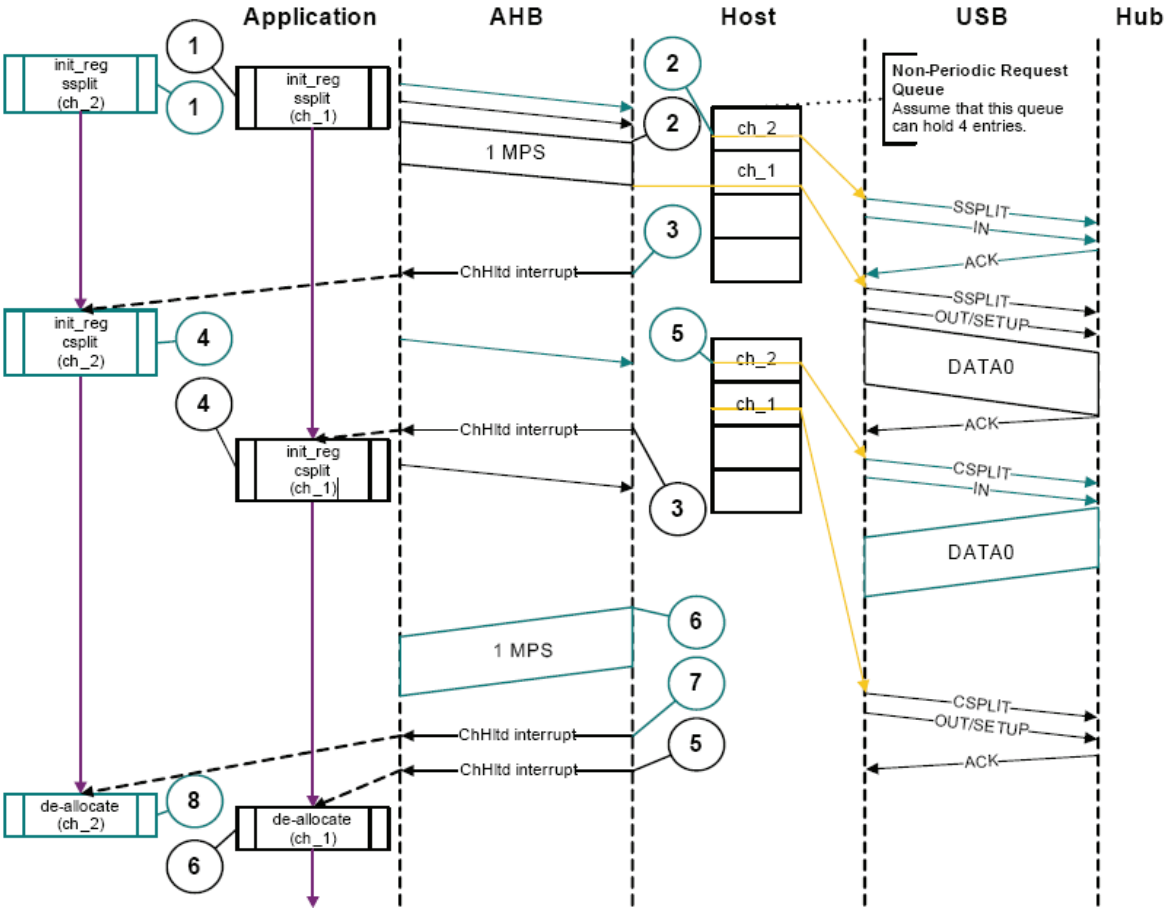


Figure 13.15 Normal Bulk/Control OUT/SETUP and Bulk/Control IN Split Transactions in DMA Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control OUT/SETUP split transactions in DMA mode is shown in table below.

Table 13.11 Interrupt Service Routine for Bulk/Control OUT/SETUP and Bulk/Control IN Split Transactions in DMA Mode

	Bulk/Control OUT/SETUP (Split)	Bulk/Control IN (Split)

Start Split	if (ChHltd)
	{
	if (ACK)
	{
	Reset Error Count
	Do Complete Split
	}
	else if (NAK)
	{
	Rewind Buffer Pointers
	Retry Start Split
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	Increment Error Count
	if (error_count < 3)
	{
	Retry Start Split
	}
	else
	{
	De-allocate Channel
	}
	}
	}
		Unmask (ChHltd)
		if (ChHltd)
		{
		if (ACK)
		{
		Reset Error Count
		Do Complete Split
		}
		else if (NAK)
		{
		Retry Start Split
		}
		else if (XactErr)
		{
		Increment Error Count
		if (error_count < 3)
		{
		Retry Start Split
		}
		else
		{
		De-allocate Channel
		}
		}
		}

Complete Split	Unmask (ChHltd)
	if (ChHltd)
	{
	if (XferCompl)
	{
	De-allocate Channel
	}
	else if (NAK)
	{
	Rewind Buffer Pointers
	Retry Start Split
	}
	else if (NYET)
	{
	Retry Complete Split
	}
	else if (STALL)
	{
	De-allocate Channel
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	Increment Error Count
	if (error_count < 3)
	{
	Retry Start Split
	}
	else
	{
	De-allocate Channel
	}
	}
	}
		Unmask (ChHltd)
		if (ChHltd)
		{
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NAK)
		{
		Retry Start Split
		}
		else if (NYET)
		{
		Retry Complete Split
		}
		else if (STALL/BblErr)
		{
		De-allocate Channel
		}
		else if (XactErr)
		{
		Increment Error Count
		if (error_count < 3)
		{
		Retry Start Split
		}
		else
		{
		De-allocate Channel
		}
		}
		}

13.6.5.18 Bulk/Control IN Split Transactions in DMA Mode

A typical bulk or control IN operation in DMA mode is shown in the figure above. See channel 1 (ch_1). The assumptions are:

- The application is attempting to receive one packet.

[Normal Bulk and Control IN Split Operations]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”.

2. The otg host writes the start split request to the non-periodic request after getting the grant from the arbiter. The otg host masks the channel 2 internally for the arbitration after writing the request.

3. As soon as the IN token is transmitted, the otg host generates the ChHltd interrupt.

4. In response to the ChHltd interrupt, set the HCSPLT2.ComplSplt bit and re-enable the channel to send the complete split token. This unmasks channel 2 for arbitration.

5. The otg host writes the complete split request to the non-periodic request after receiving the grant from the arbiter.

6. The otg host starts writing the packet to the system memory after receiving the packet successfully.

7. As soon as the received packet is written to the system memory, the otg host generates a ChHltd interrupt.

8. In response to the ChHltd interrupt, de-allocate the channel.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control IN split transactions in DMA mode is shown in the table above.

13.6.5.19 Interrupt OUT Split Transactions in Slave Mode

A typical interrupt OUT split operation in Slave mode is shown in figure below. See channel 1 (ch_1). The assumptions are:

- The application is attempting to transmit one maximum-packet-size packet in an odd microframe.

[Normal Interrupt OUT Split Operation]

The sequence of operations in figure below (channel 1) is as follows:

1. Initialize and enable channel 1 as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit.

2. Write the packet for channel 1.

3. Along with the last DWORD write of the packet, the otg host writes an entry to the Periodic Request Queue.

4. The otg host attempts to send an OUT token in the next odd microframe.

5. The otg host generates an ACK interrupt as soon as the packet is transmitted successfully.

6. In response to the ACK interrupt, set the HCSPLT1.ComplSplt to send the complete split.

7. The otg host generates the XferCompl interrupt after successfully completing the complete split transaction.

8. In response to the XferCompl interrupt, de-allocate the channel.

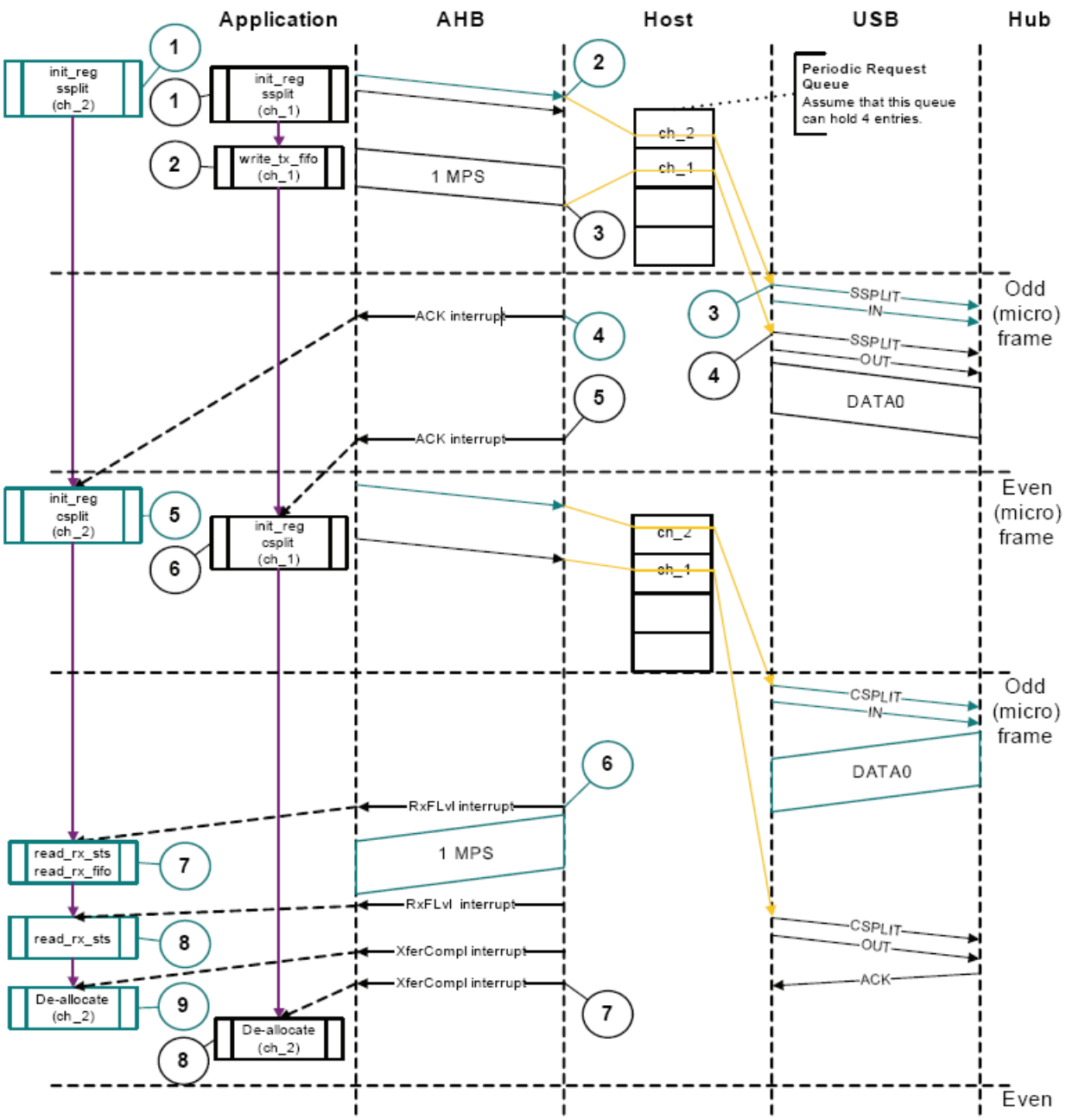


Figure 13.16 Normal Interrupt OUT/IN Split Transactions in Slave Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for interrupt OUT and IN split transactions in Slave mode is shown in table below.

	Interrupt OUT (Split)	Interrupt IN (Split)

Start Split	Unmask (ACK/FrmOvrun)
	if (ACK)
	{
	Do Complete Split
	}
	else if (FrmOvrun)
	{
	Rewind Buffer Pointers
	Unmask ChHltd
	Disable Channel
	(Retry Start Split (in next b_interval - 1 uF))
	}
	else if (ChHltd)
	{
	Mask ChHltd
	De-allocate Channel
	}
		Unmask (ACK/FrmOvrun/
		DataTglErr)
		if (ACK)
		{
		Do Complete Split
		}
		else if (FrmOvrun/DataTglErr)
		{
		Rewind Buffer Pointers
		Unmask ChHltd
		Disable Channel
		(Retry Start Split (in next b_interval - 1 uF))
		}
		else if (ChHltd)
		{
		Mask ChHltd
		De-allocate Channel
		}

Complete Split	Unmask
	(NAK/XactErr/NYET/STALL/XferCompl/FrmOvrun)
	if (XferCompl)
	{
	De-allocate Channel
	}
	else if (NAK)
	{
	Unmask ChHltd
	Disable Channel
	(Retry Start Split (in next b_interval - 1 uF))
	}
	else if (NYET)
	{
	Retry Complete Split
	}
	else if (STALL or FrmOvrun)
	{
	Unmask ChHltd
	Disable Channel
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	Unmask ChHltd
	Disable Channel
	(If (HCCHARn.EC == 3), Retry Start Split (in next b_interval - 1 uF))
	}
	else if (ChHltd)
	{
	Mask ChHltd
	De-allocate Channel
	}
		Unmask
		(NAK/XactErr/NYET/STALL/XferCompl/FrmOvrun/BblErr)
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NAK)
		{
		Unmask ChHltd
		Disable Channel
		(Retry Start Split (in next b_interval - 1 uF))
		}
		else if (NYET)
		{
		Retry Complete Split
		}
		else if (STALL or FrmOvrun or BblErr)
		{
		Unmask ChHltd
		Disable Channel
		}
		else if (XactErr)
		{
		Rewind Buffer Pointers
		Unmask ChHltd
		Disable Channel
		(If (HCCHARn.EC == 3), Retry Start Split (in next b_interval - 1 uF))
		}
		else if (ChHltd)
		{
		Mask ChHltd
		De-allocate Channel
		}

Note

The otg host tracks the error count in EC field for periodic split transactions. If the EC field matches the original programmed error count after XactErr interrupt, the application must treat the XactErr as “ERR response received.” The EC field indicates the number of immediate retries that the OTG host has performed before generating the XactErr interrupt.

13.6.5.20 Interrupt IN Split Transactions in Slave Mode

A typical interrupt IN split operation in Slave mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one maximum-packet-size packet in an odd microframe.
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet.

[Normal Interrupt IN Split Operation]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”. The application must set the HCCHAR2.OddFrm bit.

2. The otg host writes the start split request to the Periodic Request Queue as soon as the HCCHAR2 register is written.

3. The otg host attempts to send a start split IN token in the next odd microframe.

4. As soon as the IN packet is transmitted, the otg host generates an ACK interrupt.

5. In response to the ACK interrupt, set the HCSPLT2.CompSplt bit in the next microframe to send the complete split token in the next odd microframe.

6. As soon as the received packet is written to the receive FIFO, the otg host generates the RxFLvl interrupt.

7. In response to the RxFLvl interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. The application must mask the RxFLvl interrupt before reading the receive FIFO and unmask after reading the entire packet.

8. The core generates the RxFLvl interrupt for the transfer completion status entry in the receive FIFO. The application must read the receive packet status and, if the receive packet status is not an IN data packet (GRXSTSR.PktSts != 4’b0010), ignore it.

9. The core generates the XferCompl interrupt as soon as the receive packet status is read. In response to the XferCompl interrupt, read the HCTSIZ2.PktCnt field. If HCTSIZ2.PktCnt != 0, disable the channel (as explained in “Halting a Channel” on page 216) before re-initializing the channel for the next transfer, if any. If HCTSIZ2.PktCnt == 0, reinitialize the channel for the next transfer. This time, the application must reset the HCCHAR2.OddFrm bit.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for an interrupt IN split transaction in Slave mode is shown in the table above.

13.6.5.21 Interrupt OUT Split Transactions in DMA Mode

A typical interrupt OUT split operation in DMA mode is shown in figure below (see channel 1 [ch_1]). It is assumed that the application is attempting to transmit one packet (1 maximum packet size) in an odd microframe.

[Normal Interrupt OUT Split Operation]

The sequence of operations in figure below (channel 1) is as follows:

1. Initialize and enable channel 1 for start split as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit.

2. The otg host starts reading the packet.

3. The otg host attempts to send the start split transaction.

4. After successfully transmitting the start split, the otg host generates the ChHltd interrupt.

5. In response to the ChHltd interrupt, set the HCSPLT1.ComplSplt bit to send the complete split.

6. After successfully completing the complete split transaction, the otg host generates the ChHltd interrupt.

7. In response to ChHltd interrupt, de-allocate the channel.

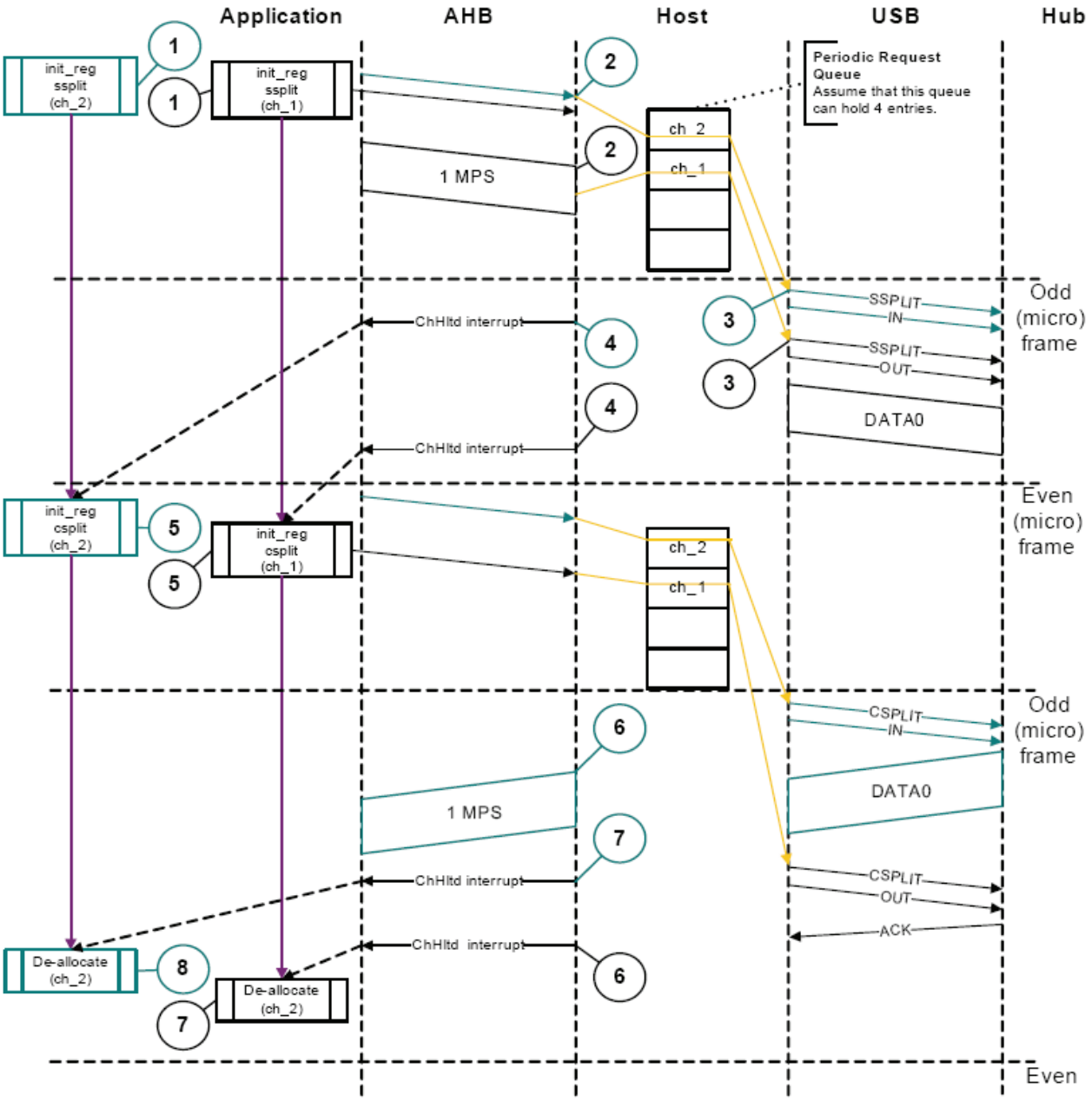


Figure 13.17 Normal Interrupt OUT/IN Split Transactions in DMA Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for interrupt OUT split transaction in DMA mode is shown in table below.

Table 13.12 Interrupt Service Routine for Interrupt OUT/IN Split Transactions in DMA Mode

	Interrupt OUT (Split)	Interrupt IN (Split)

Start Split	Unmask (ChHltd)
	if (ChHltd)
	{
	if (ACK)
	{
	Do Complete Split
	}
	else if (FrmOvrun)
	{
	Rewind Buffer Pointers
	Retry Start Split (in next b_interval - 1 uF)
	}
	}
		Unmask (ChHltd)
		if (ChHltd)
		{
		if (ACK)
		{
		Do Complete Split
		}
		else if (FrmOvrun)
		{
		Rewind Buffer Pointers
		Retry Start Split (in next b_interval - 1 uF)
		}
		}

Complete Split	Unmask (ChHltd)
	if (ChHltd)
	{
	if (XferCompl)
	{
	De-allocate Channel
	}
	else if (NAK)
	{
	Retry Start Split (in next b_interval - 1 uF)
	}
	else if (NYET)
	{
	Retry Complete Split
	}
	else if (STALL or FrmOvrun)
	{
	De-allocate Channel
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	if (HCCHARn.EC == 3) // ERR response received
	{
	Retry Start Split (in next b_interval - 1 uF)
	}
	else
	{
	De-allocate Channel
	}
	}
	}
		Unmask (ChHltd)
		if (ChHltd)
		{
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NAK)
		{
		Retry Start Split (in next b_interval - 1 uF)
		}
		else if (NYET)
		{
		Retry Complete Split
		}
		else if (STALL or FrmOvrun or BblErr)
		{
		De-allocate Channel
		}
		else if (XactErr)
		{
		Rewind Buffer Pointers
		if (HCCHARn.EC == 3) // ERR response received
		{
		Retry Start Split (in next b_interval - 1 uF)
		}
		else
		{
		De-allocate Channel
		}
		}
		}

Note

The otg host tracks the error count in the EC field for periodic split transactions. If the EC field matches the original programmed error count after XactErr interrupt, the application must treat the XactErr as an ERR Response Received. The EC field indicates the number of immediate retries the OTG host has performed before generating the XactErr interrupt.

13.6.5.22 Interrupt IN Split Transactions in DMA Mode

A typical interrupt IN split operation in DMA mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one maximum-packet-size packet in an odd microframe.
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet.

[Normal Interrupt IN Split Operation]

The sequence of operations in the figure above (channel 2 {ch_2]) is as follows:

1. Initialize and enable channel 2 for start split as explained in “Channel Initialization”.

2. The otg host writes an IN request to the Request queue as soon as channel 2 receives the grant from the arbiter.

3. The otg host attempts to send the start split IN token in the beginning of the next odd microframe.

4. The otg host generates the ChHltd interrupt after successfully transmitting the start split IN token.

5. In response to the ChHltd interrupt, set the HCSPLT2.ComplSplt bit to send the complete split.

6. As soon the packet is received successfully, the otg host starts writing the data to the system memory.

7. The otg host generates the ChHltd interrupt after transferring the received data to the system memory.

8. In response to the ChHltd interrupt, de-allocate or reinitialize the channel for the next start split.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for interrupt IN split transaction in DMA mode is shown in the table above .

13.6.5.23 Isochronous OUT Split Transactions in Slave Mode

A typical isochronous OUT split operation in Slave mode is shown in the figure below. See channel 1 (ch_1). The assumptions are:

- The application is attempting to transmit a 376-byte packet in an odd microframe.

[Normal Isochronous OUT Split Operation]

The sequence of operations in the figure below (channel 1 [ch_1]) is as follows:

1. Initialize and enable channel 1 for start split (begin) as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit. Program the MPS field with 188 bytes.

2. Write the packet for channel 1.

3. Along with the last DWORD write of the packet, the otg host writes an entry to the Periodic Request Queue.

4. The otg host attempts to send an OUT token in the next odd microframe.

5. The otg host generates an ACK interrupt as soon as the packet is transmitted successfully.

6. In response to the ACK interrupt, reinitialize the registers to send the start split (end).

7. The otg host generates an ACK interrupt after successfully completing the start split (end) transaction.

8. In response to the ACK interrupt, de-allocate the channel.

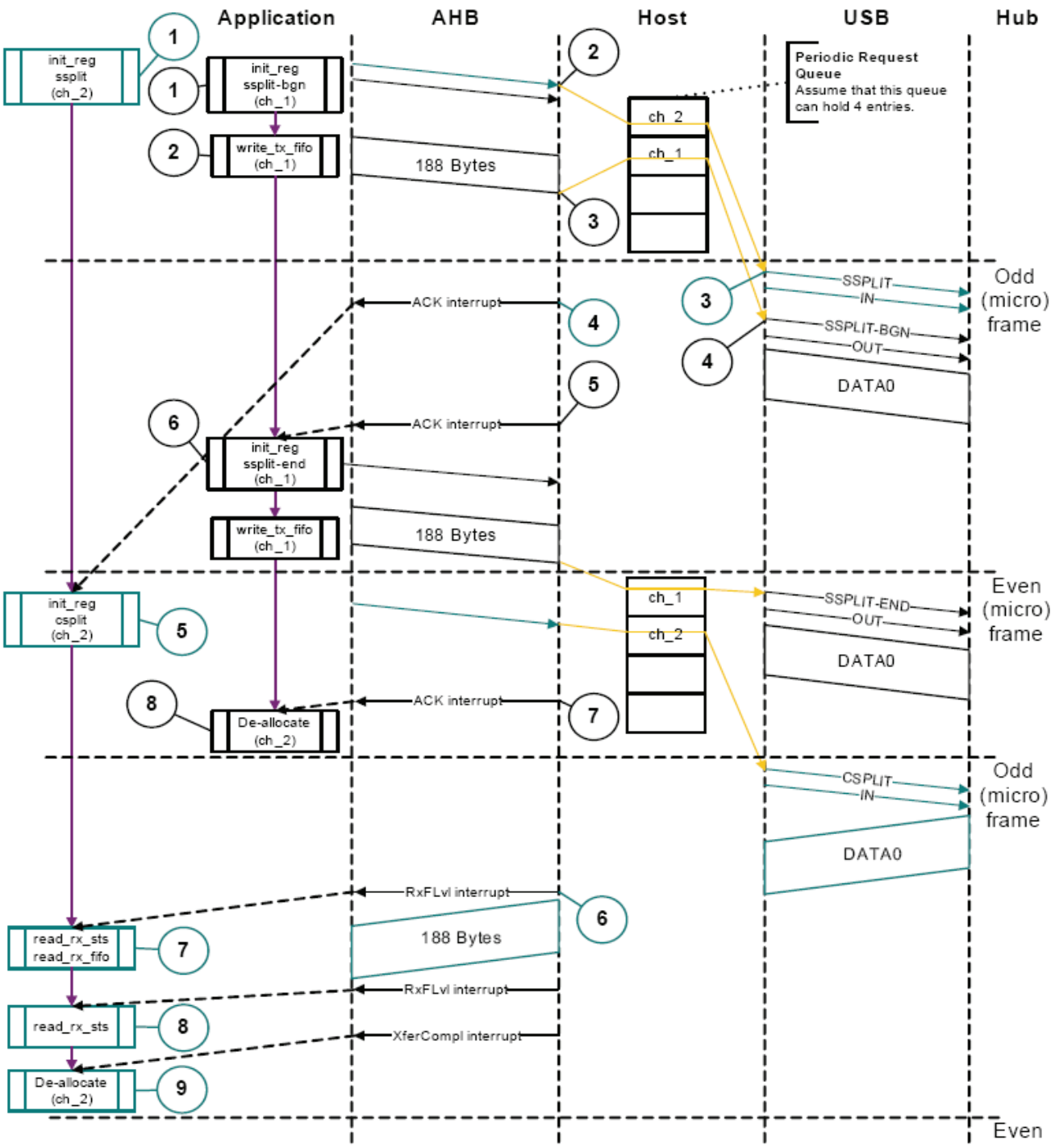


Figure 13.18 Normal Isochronous OUT/IN Split Transactions in Slave Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for isochronous OUT split transaction in Slave mode is shown in table below.

Table 13.13 Interrupt Service Routine for Isochronous OUT/IN Split Transactions in Slave Mode

	Isochronous OUT (Split)	Isochronous IN (Split)

Start Split	Unmask (XferCompl)
	if (XferCompl)
	{
	Do Next Start Split (in next b_interval - 1 uF)
	}
	else if (FrmOvrun)
	{
	Do Next Transaction in next frame.
	}
		Unmask (ACK)
		if (ACK)
		{
		Do Complete Split
		}
		else if (FrmOvrun)
		{
		Do Next Transaction in next frame.
		}

Complete Split	Not Applicable	Unmask (XactErr/NYET/STALL/XferCompl/FrmOvrun/BblErr)
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NYET)
		{
		Do Next Complete Split
		}
		else if (STALL or FrmOvrun or BblErr)
		{
		Unmask ChHltd
		Disable Channel
		}
		else if (XactErr)
		{
		Rewind Buffer Pointers
		if (HCCHARn.EC == 3) // ERR response received
		{
		Record ERR error
		Do Next Start Split (in next frame)
		}
		else
		{
		Unmask ChHltd
		Disable Channel
		}
		}
		else if (ChHltd)
		{
		Mask ChHltd
		De-allocate Channel
		}

Note

The otg host keeps track of error count in EC field for periodic split transactions. If the EC field matches the original programmed error count after XactErr interrupt, the application must treat the XactErr as “ERR response received”. The EC field indicates the number of immediate retries that the otg host has performed before generating the XactErr interrupt.

13.6.5.24 Isochronous IN Split Transactions in Slave Mode

A typical isochronous IN split operation in Slave mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one maximum-packet-size packet in an odd microframe.
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet.

[Normal Isochronous IN Split Operation]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 for start split as explained in “Channel Initialization”. The application must set the HCCHAR2.OddFrm bit.

2. The otg host writes the start split request to the Periodic Request Queue as soon as the HCCHAR2 register is written.

3. The otg host attempts to send the start split IN token in the next odd microframe.

4. As soon as the IN packet is transmitted, the otg host generates an ACK interrupt.

5. In response to the ACK interrupt, set the Do Complete Split bit (HCSPLT2.CompSplt) in the next microframe to send the complete split token in the next odd microframe.

6. As soon as the received packet is written to the receive FIFO, the otg host generates the RxFLvl interrupt.

7. In response to the RxFLvl interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. The application must mask the RxFLvl interrupt before reading the receive FIFO and unmask it after reading the entire packet.

8. The core generates the RxFLvl interrupt for the transfer completion status entry in the receive FIFO. The application must read the receive packet status and if the receive packet status is not an IN data packet (GRXSTSR.PktSts != 4’b0010), ignore it.

9. The core generates the XferCompl interrupt as soon as the receive packet status is read. In response to the XferCompl interrupt, read the HCTSIZ2.PktCnt field. If HCTSIZ2.PktCnt != 0, disable the channel (as explained in “Halting a Channel” on page 216) before re-initializing the channel for the next transfer, if any. If HCTSIZ2.PktCnt == 0, reinitialize the channel for the next transfer. This time, the application must reset the HCCHAR2.OddFrm bit.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for isochronous IN split transaction in Slave mode is shown in the table above.

13.6.5.25 Isochronous OUT Split Transactions in DMA Mode

A typical isochronous OUT split operation in DMA mode is shown in figure below. See channel 1 (ch_1). The assumption is that the application is attempting to transmit a 376-byte packet in an odd microframe.

[Normal Isochronous OUT Split Operation]

The sequence of operations in figure below (channel 1) is as follows:

1. Initialize and enable channel 1 for start split (begin) as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit. Program the MPS field with 188 bytes.

2. The otg host starts reading the packet.

3. After successfully transmitting the start split (begin), the otg host generates the ChHltd interrupt.

4. In response to the ChHltd interrupt, reinitialize the registers to send the start split (end).

5. After successfully transmitting the start split (end), the _otg host generates a ChHltd interrupt.

6. In response to the ChHltd interrupt, de-allocate the channel.

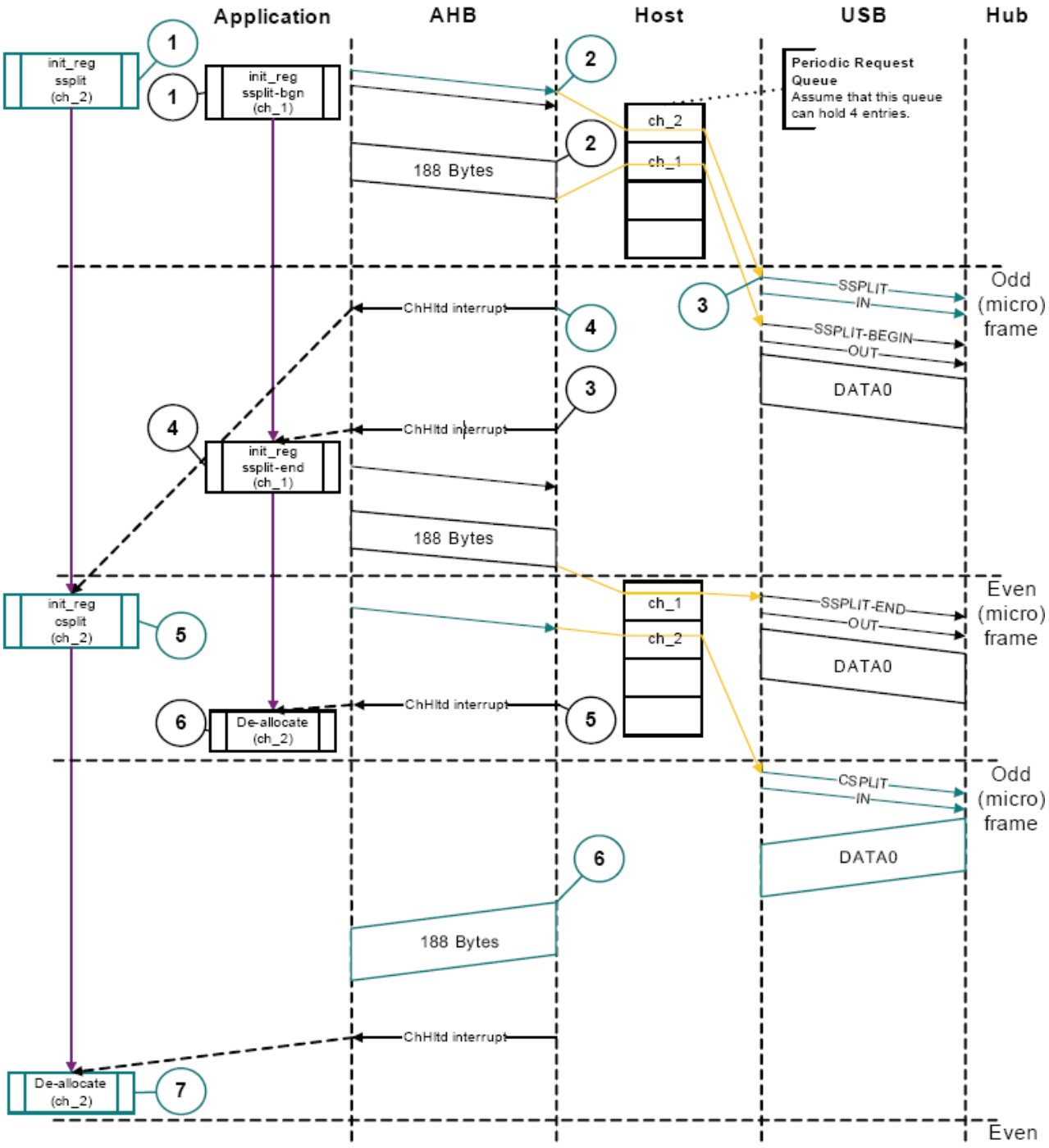


Figure 13.19 Normal Isochronous OUT/IN Split Transactions in DMA Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for an isochronous OUT split transaction in DMA mode is shown in table below.

	Isochronous OUT (Split)	Isochronous IN (Split)

Start Split	Unmask (ChHltd)
	if (ChHltd)
	{
	if (ACK)
	{
	Do Next Start Split (in next b_interval – 1 uF)
	}
	else if (FrmOvrun)
	{
	Do Next Transaction in next frame.
	}
	}
		Unmask (ChHltd)
		if (ChHltd)
		{
		if (ACK)
		{
		Do Complete Split
		}
		else if (FrmOvrun)
		{
		Rewind Buffer Pointers
		Retry Start Split (in next b_interval – 1 uF)
		}
		}

Complete Split	Not Applicable	Unmask (ChHltd)
		if (ChHltd)
		{
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NAK)
		{
		Retry Start Split (in next b_interval – 1 uF)
		}
		else if (NYET)
		{
		Do Next Complete Split
		}
		else if (STALL or FrmOvrun or BblErr)
		{
		De-allocate Channel
		}
		else if (XactErr)
		{
		Rewind Buffer Pointers
		if (HCCHARn.EC = = 3) // ERR response received
		{
		Record ERR error
		Do Next Start Split (in next frame)
		}
		else
		{
		De-allocate Channel
		}
		}
		}

Note

The otg host keeps track of error count in EC field for periodic split transactions. If the EC field matches the original programmed error count after XactErr interrupt, the application must treat the XactErr as “ERR response received”. The EC field indicates the number of immediate retries that the OTG host has performed before generating the XactErr interrupt.

13.6.5.26 Isochronous IN Split Transactions in DMA Mode

A typical isochronous IN split operation in DMA mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one maximum-packet-size packet in an odd microframe.
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet.

[Normal Isochronous IN Split Operation]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 for start split as explained in “Channel Initialization”.

2. The otg host writes an IN request to the Request queue as soon as channel 2 receives the grant from the arbiter.

3. The otg host attempts to send the start split IN token in the beginning of the next odd microframe.

4. The otg host generates the ChHltd interrupt after successfully transmitting the start split IN token.

5. In response to the ChHltd interrupt, set the HCSPLT2.ComplSplt bit to send the complete split.

6. As soon the packet is received successfully, the otg host starts writing the data to the system memory.

7. The otg host generates the ChHltd interrupt after transferring the received data to the system memory. In response to the ChHltd interrupt, de-allocate the channel or reinitialize the channel for the next start split.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for isochronous IN split transaction in DMA mode is shown in the table above.

13.6.6 Selecting the Queue Depth

Choose the Periodic and Non-periodic Request Queue depths carefully to match the number of periodic/non-periodic endpoints accessed.

The Non-periodic Request Queue depth affects the performance of non-periodic transfers. The deeper the queue (along with sufficient FIFO size), the more often the core is able to pipeline non-periodic transfers. If the queue size is small, the core is able to put in new requests only when the queue space is freed up.

The core’s Periodic Request Queue depth is critical to performing periodic transfers as scheduled. Select the periodic queue depth, based on the number of periodic transfers scheduled in a microframe. In Slave mode, however, the application must also take into account the disable entry that must be put into the queue. So, if there are two non-high bandwidth periodic endpoints, the Periodic Request Queue depth must be at least 4. If at least one high-bandwidth endpoint supported, the queue depth must be 8. If the Periodic Request Queue depth is smaller than the periodic transfers scheduled in a microframe, a frame overrun condition results.

13.6.7 Handling Babble Conditions

The otg handles two cases of babble: packet babble and port babble. Packet babble occurs if the device sends more data than the maximum packet size for the channel. Port babble occurs if the core continues to receive data from the device at EOF2 (the end of frame 2, which is very close to SOF).

When OTG detects a packet babble, it stops writing data into the Rx buffer and waits for the end of packet (EOP). When it detects an EOP, it flushes already-written data in the Rx buffer and generates a Babble interrupt to the application.

When OTG detects a port babble, it flushes the RxFIFO and disables the port. The core then generates a Port Disabled Interrupt (GINTSTS.PrtInt, HPRT.PrtEnChng). On receiving this interrupt, the application must determine that this is not due to an overcurrent condition (another cause of the Port Disabled interrupt) by checking HPRT.PrtOvrCurrAct, then perform a soft reset. The core does not send any more tokens after it has detected a port babble condition.

13.7 Device Programming Model

13.7.1 Endpoint Initialization

13.7.1.1 Initialization on USB Reset

1. Set the NAK bit for all OUT endpoints

- DOEPCTLn.SNAK = 1 (for all OUT endpoints)

2. Unmask the following interrupt bits

- DAINTMSK.INEP0 = 1 (control 0 IN endpoint)
- DAINTMSK.OUTEP0 = 1 (control 0 OUT endpoint)
- DOEPMSK.SETUP = 1
- DOEPMSK.XferCompl = 1
- DIEPMSK.XferCompl = 1
- DIEPMSK.TimeOut = 1

3. To transmit or receive data, the device must initialize more registers as specified in "Device DMA/Slave Mode Initialization".

4. Set up the Data FIFO RAM for each of the FIFOs

- Program the GRXFSIZ Register, to be able to receive control OUT data and setup data. If thresholding is not enabled, at a minimum, this must be equal to 1 max packet size of control endpoint 0 + 2 DWORDs (for the status of the control OUT data packet) + 10 DWORDs (for setup packets). If thresholding is enabled, at a minimum, this must be equal to 2* (Rx_threshold_length/4 +1)+ 2 DWORDs (for the status of the control OUT data packet) + 10 DWORDs (for setup packets)
- Program the GNPTXFSIZ Register in Shared FIFO operation or dedicated FIFO size register (depending on the FIFO number chosen) in Dedicated FIFO operation, to be able to transmit control IN data. At a minimum, this must be equal to 1 max packet size of control endpoint 0.If thresholding is enabled, this can be programmed to less than one max packet size.

5. (This step is not required if you are using Scatter/Gather DMA mode.) Program the following fields in the endpoint-specific registers for control OUT endpoint 0 to receive a SETUP packet

- DOEPTSIZ0.SetUP Count = 3 (to receive up to 3 back-to-back SETUP packets)
- In DMA mode, DOEPWMA0 register with a memory address to store any SETUP packets received

At this point, all initialization required to receive SETUP packets is done, except for enabling control OUT endpoint 0 in DMA mode.

13.7.1.2 Initialization on Enumeration Completion

1. On the Enumeration Done interrupt (GINTSTS.EnumDone, read the DSTS register to determine the enumeration speed.

2. Program the DIEPCTL0.MPS field to set the maximum packet size. This step configures control endpoint 0. The maximum packet size for a control endpoint depends on the enumeration speed.

3. In DMA mode, program the DOEPCTL0 register to enable control OUT endpoint 0, in order to receive a SETUP packet. In Scatter/Gather DMA mode, the descriptors must be set up in memory before enabling the endpoint.

- DOEPCTL0.EPEna = 1

At this point, the device is ready to receive SOF packets and is configured to perform control transfers on control endpoint 0.

13.7.1.3 Initialization on SetAddress Command

This section describes what the application must do when it receives a SetAddress command in a SETUP packet.

1. Program the DCFG register with the device address received in the SetAddress command

2. Program the core to send out a status IN packet.

13.7.1.4 Initialization on SetConfiguration/SetInterface Command

This section describes what the application must do when it receives a SetConfiguration or SetInterface command in a SETUP packet.

1. When a SetConfiguration command is received, the application must program the endpoint registers to configure them with the characteristics of the valid endpoints in the new configuration.

2. When a SetInterface command is received, the application must program the endpoint registers of the endpoints affected by this command.

3. Some endpoints that were active in the prior configuration or alternate setting are not valid in the new configuration or alternate setting. These invalid endpoints must be deactivated.

4. For details on a particular endpoint's activation or deactivation, see "Endpoint Activation" and "Endpoint Deactivation".

5. Unmask the interrupt for each active endpoint and mask the interrupts for all inactive endpoints in the DAINTMSK register.

6. Set up the Data FIFO RAM for each FIFO (only if Dynamic FIFO Sizing is enabled). See "Data FIFO RAM Allocation" for more detail.

7. After all required endpoints are configured, the application must program the core to send a status IN packet.
At this point, the device core is configured to receive and transmit any type of data packet.

13.7.1.5 Endpoint Activation

This section describes the steps required to activate a device endpoint or to configure an existing device endpoint to a new type.

1. Program the characteristics of the required endpoint into the following fields of theDIEPCTLn register (for IN or bidirectional endpoints) or the DOEPCTLn register (for OUT or bidirectional endpoints).

- Maximum Packet Size
- USB Active Endpoint = 1
- Endpoint Start Data Toggle (for interrupt and bulk endpoints)
- Endpoint Type
- TxFIFO Number1

2. Once the endpoint is activated, the core starts decoding the tokens addressed to that endpoint and sends out a valid handshake for each valid token received for the endpoint.

13.7.1.6 Endpoint Deactivation

This section describes the steps required to deactivate an existing endpoint.

1. In the endpoint to be deactivated, clear the USB Active Endpoint bit in the DIEPCTLn register (for IN or bidirectional endpoints) or the DOEPCTLn register (for OUT or bidirectional endpoints).

2. Once the endpoint is deactivated, the core ignores tokens addressed to that endpoint, resulting in a timeout on the USB.

13.7.1.7 Device DMA/Slave Mode Initialization

The application must meed the following conditions to set up the device core to handle traffic.

- In Slave mode, GINTMSK.NPTxFEmpMsk, and GINTMSK.RxFLvlMsk must be unset.
- In DMA mode, the aforementioned interrupts must be masked.

13.7.2 Operational Model

13.7.2.1 SETUP and OUT Data Transfers

This section describes the internal data flow and application-level operations during data OUT transfers and SETUP transactions.

[Packet Read in Slave Mode]

This section describes how to read packets (OUT data and SETUP packets) from the receive FIFO in Slave mode.

1. On catching a GINTSTS.RxFLvl interrupt, the application must read the Receive Status Pop register (GRXSTSP).

2. The application can mask the GINTSTS.RxFLvl interrupt by writing to GINTMSK.RxFLvl = 1'b0, until it has read the packet from the receive FIFO.

3. If the received packet's byte count is not 0, the byte count amount of data is popped from the receive Data FIFO and stored in memory. If the received packet byte count is 0, no data is popped from the Receive Data FIFO.

4. The receive FIFO's packet status readout indicates one of the following.

- Global OUT NAK Pattern: PktSts = Global OUT NAK, BCnt = 11'h000, EPNum = Dont Care (4'h0), DPID = Dont Care (2'b00). This data indicates that the global OUT NAK bit has taken effect.
- SETUP Packet Pattern: PktSts = SETUP, BCnt = 11'h008, EPNum = Control EPNum, DPID = D0. This data indicates that a SETUP packet for the specified endpoint is now available for reading from the receive FIFO.
- Setup Stage Done Pattern: PktSts = Setup Stage Done, BCnt = 11'h0, EPNum = Control EP Num, DPID = Don't Care (2'b00). This data indicates that the Setup stage for the specified endpoint has completed and the Data stage has started. After this entry is popped from the receive FIFO, the core asserts a Setup interrupt on the specified control OUT endpoint.
- Data OUT Packet Pattern: PktSts = DataOUT, BCnt = size of the Received data OUT packet ($0 \leq$ BCnt $\leq 1,024$), EPNum = EPNum on which the packet was received, DPID = Actual Data PID.
- Data Transfer Completed Pattern: PktSts = Data OUT Transfer Done, BCnt = 11'h0, EPNum = OUT EP Num on which the data transfer is complete, DPID = Dont Care (2'b00). This data indicates that a OUT data transfer for the specified OUT endpoint has completed. After this entry is popped from the receive FIFO, the core asserts a Transfer Completed interrupt on the specified OUT endpoint.

The encoding for the PktSts is listed in "Receive Status Debug Read/Status Read and Pop Registers (GRXSTSR/GRXSTSP)".

5. After the data payload is popped from the receive FIFO, the GINTSTS.RxFLvl interrupt must be unmasked.

6. Steps 1–5 are repeated every time the application detects assertion of the interrupt line due to GINTSTS.RxFLvl. Reading an empty receive FIFO can result in undefined core behavior.

Below figure provides a flow chart of the above procedure.

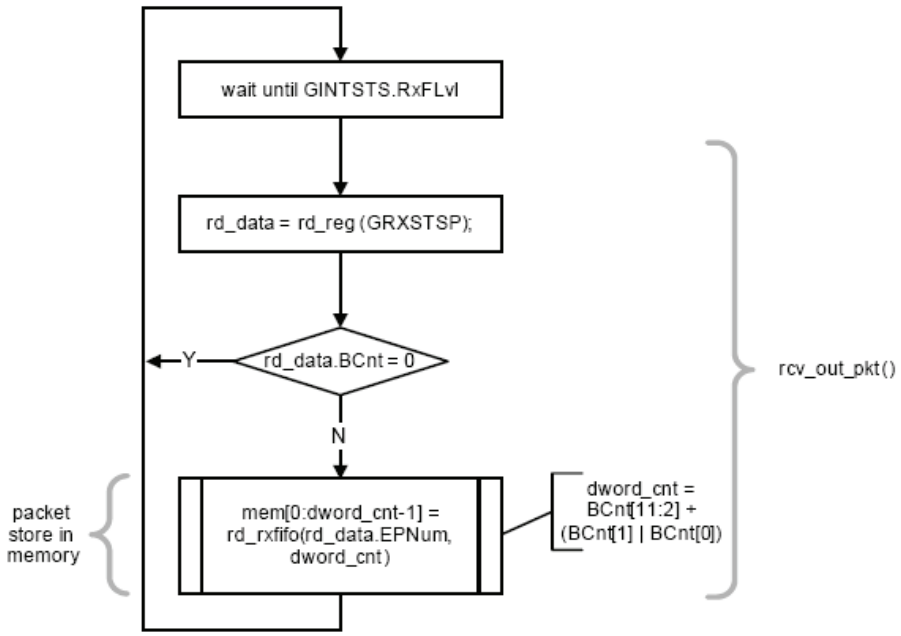


Figure 13.20 Receive FIFO Packet Read in Slave Mode

[SETUP Transactions]

This section describes how the core handles SETUP packets and the application's sequence for handling SETUP transactions.

Application Requirements

1. To receive a SETUP packet, the DOEPTSIZn.SUPCnt field in a control OUT endpoint must be programmed to a non-zero value. When the application programs the SUPCnt field to a non-zero value, the core receives SETUP packets and writes them to the receive FIFO, irrespective of the DOEPCTLn.NAK status and DOEPCTLn.EPEna bit setting. The SUPCnt field is decremented every time the control endpoint receives a SETUP packet. If the SUPCnt field is not programmed to a proper value before receiving a SETUP packet, the core still receives the SETUP packet and decrements the SUPCnt field, but the application possibly is not be able to determine the correct number of SETUP packets received in the Setup stage of a control transfer.

- DOEPTSIZn.SUPCnt = 3

2. In DMA mode, the OUT endpoint must also be enabled, to transfer the received SETUP packet data from the internal receive FIFO to the external memory.

- DOEPCTLn.EPEna = 1'b1

3. The application must always allocate some extra space in the Receive Data FIFO, to be able to receive up to three SETUP packets on a control endpoint.

- The space to be Reserved is (4 * n) + 6 DWORDs, where n is the number of control endpoints supported by the device. Three DWORDs are required for the first SETUP packet, 1 DWORD is required for the Setup Stage Done DWORD, and 6 DWORDs are required to store two extra SETUP packets among all control endpoints.
- 3 DWORDs per SETUP packet are required to store 8 bytes of SETUP data and 4 bytes of SETUP status (Setup Packet Pattern). The core reserves this space in the receive data.
- FIFO to write SETUP data only, and never uses this space for data packets.

4. In Slave mode, the application must read the 2 DWORDs of the SETUP packet from the receive FIFO. In DMA mode, the core writes the 2 DWORDs of SETUP data to the memory.

5. The application must read and discard the Setup Stage Done DWORD from the receive FIFO.

Internal Data Flow

1. When a SETUP packet is received, the core writes the received data to the receive FIFO, without checking for available space in the receive FIFO and irrespective of the endpoint's NAK and Stall bit settings.

- The core internally sets the IN NAK and OUT NAK bits for the control IN/OUT endpoints on which the SETUP packet was received.

2. For every SETUP packet received on the USB, 3 DWORDs of data is written to the receive FIFO, and the SUPCnt field is decremented by 1.

- The first DWORD contains control information used internally by the core
- The second DWORD contains the first 4 bytes of the SETUP command
- The third DWORD contains the last 4 bytes of the SETUP command

3. When the Setup stage changes to a Data IN/OUT stage, the core writes an entry (Setup Stage Done DWORD) to the receive FIFO, indicating the completion of the Setup stage.

4. On the AHB side, SETUP packets are emptied either by the DMA or the application. In DMA mode, the SETUP packets (2 DWORDs) are written to the memory location programmed in the DOEPWMAn register, only if the endpoint is enabled. If the endpoint is not enabled, the data remains in the receive FIFO until the enable bit is set.

5. When either the DMA or the application pops the Setup Stage Done DWORD from the receive FIFO, the core interrupts the application with a DOEPINTn.SETUP interrupt, indicating it can process the received SETUP packet.

- The core clears the endpoint enable bit for control OUT endpoints.

Application Programming Sequence

1. Program the DOEPTSIZn register.

- DOEPTSIZn.SUPCnt = 3

2. In DMA mode, program the DOEPWMAn register and DOEPCTLn register with the endpoint characteristics and set the Endpoint Enable bit (DOEPCTLn.EPEna).

- Endpoint Enable = 1

3. In Slave mode, wait for the GINTSTS.RxFLvl interrupt and empty the data packets from the receive FIFO, as explained in "Packet Read in Slave Mode". This step can be repeated many times.

4. Assertion of the DOEPINTn.SETUP interrupt marks a successful completion of the SETUP Data Transfer.

- On this interrupt, the application must read the DOEPTSIZn register to determine the number of SETUP packets received and process the last received SETUP packet.
- In DMA mode, the application must also determine if the interrupt bit DOEPINTn.Back2BackSETup is set. This bit is set if the core has received more than three back-to-back SETUP packets. If this is the case, the application must ignore the DOEPTSIZn.SUPCnt value and use the DOEPWMAn directly to read out the last SETUP packet received. DOEPWMAn-8 provides the pointer to the last valid SETUP data.

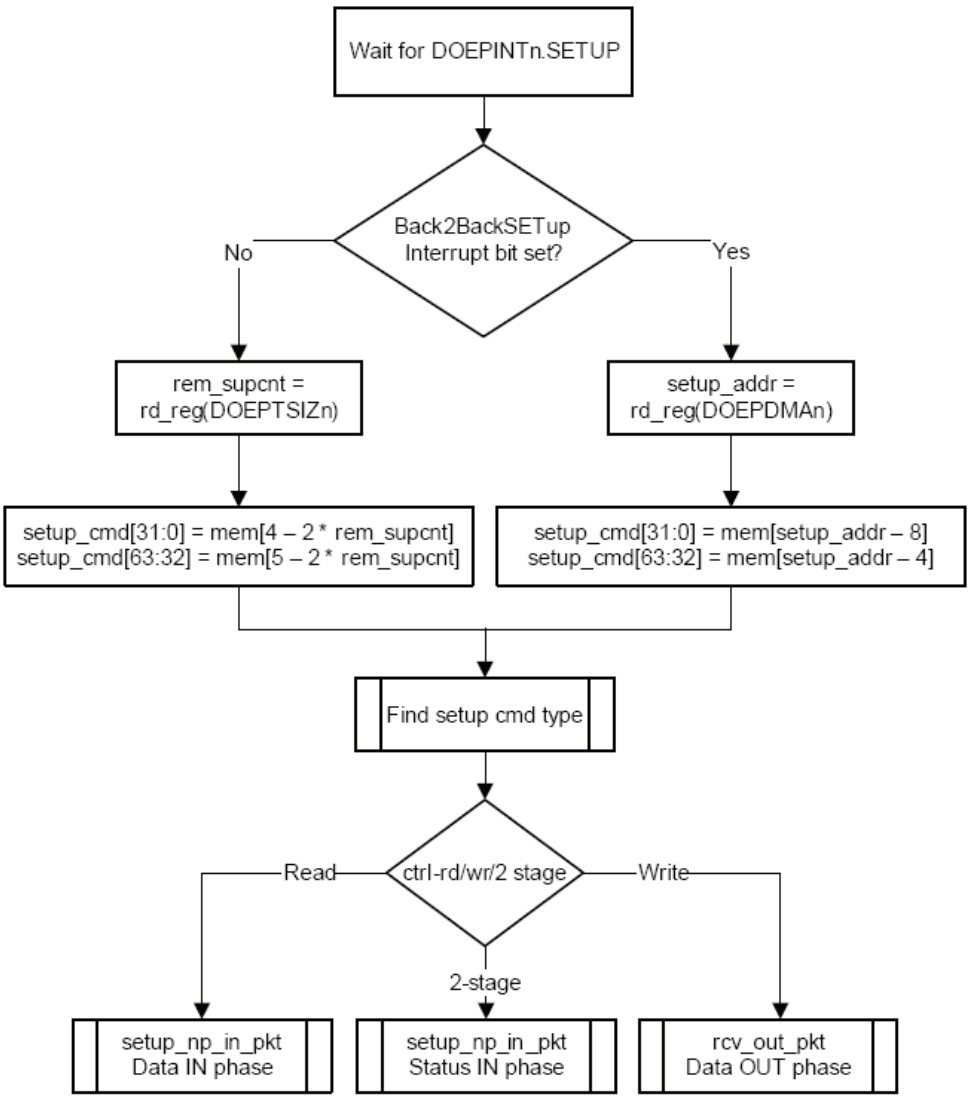


Figure 13.21 Processing a SETUP Packet

[Handling More Than Three Back-to-Back SETUP Packets]

Per the USB 2.0 specification, normally, during a SETUP packet error, a host does not send more than three back-to-back SETUP packets to the same endpoint. However, the USB 2.0 specification does not limit the number of back-to-back SETUP packets a host can send to the same endpoint. When this condition occurs, the OTG core generates an interrupt (DOEPINTn.Back2BackSETup). In DMA mode, the core also rewinds the DMA address for that endpoint (DOEPWMAn) and overwrites the first SETUP packet in system memory with the fourth, second with the fifth, and so on. If the Back2BackSETup interrupt is asserted, the application must read the OUT endpoint DMA register (DOEPWMAn) to determine the final SETUP data in system memory.

In DMA mode, the application can mask the Back2BackSETup interrupt, but after receiving the DOEPINT.SETUP interrupt, the application can read the DOEPINT.Back2BackSETup interrupt bit. In Slave mode, the application can use the GINTSTS.RxFLvl interrupt to read out the SETUP packets from the FIFO whenever the core receives the SETUP packet.

[Setting the Global OUT NAK]

Internal Data Flow

1. When the application sets the Global OUT NAK (DCTL.SGOUTNak), the core stops writing data, except SETUP packets, to the receive FIFO. Irrespective of the space availability in the receive FIFO, non-isochronous OUT tokens receive a NAK handshake response, and the core ignores isochronous OUT data packets

2. The core writes the Global OUT NAK pattern to the receive FIFO. The application must reserve enough receive FIFO

space to write this data pattern. See “Data FIFO RAM Allocation”.

3. When either the core (in DMA mode) or the application (in Slave mode) pops the Global OUT NAK pattern DWORD from the receive FIFO, the core sets the GINTSTS.GOUTNakEff interrupt.

4. Once the application detects this interrupt, it can assume that the core is in Global OUT NAK mode. The application can clear this interrupt by clearing the DCTL.SGOUTNak bit.

Application Programming Sequence

1. To stop receiving any kind of data in the receive FIFO, the application must set the Global OUT NAK bit by programming the following field.

- DCTL.SGOUTNak = 1’b1

2. Wait for the assertion of the interrupt GINTSTS.GOUTNakEff. When asserted, this interrupt indicates that the core has stopped receiving any type of data except SETUP packets.

3. The application can receive valid OUT packets after it has set DCTL.SGOUTNak and before the core asserts the GINTSTS.GOUTNakEff interrupt.

4. The application can temporarily mask this interrupt by writing to the GINTMSK.GINNakEffMsk bit.

- GINTMSK.GINNakEffMsk = 1’b0

5. Whenever the application is ready to exit the Global OUT NAK mode, it must clear the DCTL.SGOUTNak bit. This also clears the GINTSTS.GOUTNakEff interrupt.

- DCTL.CGOUTNak = 1’b1

6. If the application has masked this interrupt earlier, it must be unmasked as follows:

- GINTMSK.GINNakEffMsk = 1’b1

[Disabling an OUT Endpoint]

The application must use this sequence to disable an OUT endpoint that it has enabled.

Application Programming Sequence

1. Before disabling any OUT endpoint, the application must enable Global OUT NAK mode in the core, as described in “Setting the Global OUT NAK”.

- DCTL.DCTL.SGOUTNak = 1’b1

2. Wait for the GINTSTS.GOUTNakEff interrupt

3. Disable the required OUT endpoint by programming the following fields.

- DOEPCTLn.EPDisable = 1’b1
- DOEPCTLn.SNAK = 1’b1

4. Wait for the DOEPINTn.EPDisabled interrupt, which indicates that the OUT endpoint is completely disabled. When the EPDisabled interrupt is asserted, the core also clears the following bits.

- DOEPCTLn.EPDisable = 1’b0
- DOEPCTLn.EPEnable = 1’b0

5. The application must clear the Global OUT NAK bit to start receiving data from other non-disabled OUT endpoints.

- DCTL.SGOUTNak = 1’b0

[Generic Non-Isochronous OUT Data Transfers without Thresholding]

This section describes a regular non-isochronous OUT data transfer (control, bulk, or interrupt).

Application Requirements

1. Before setting up an OUT transfer, the application must allocate a buffer in the memory to accommodate all data to be received as part of the OUT transfer, then program that buffer's size and start address (in DMA mode) in the endpoint-specific registers.

2. For OUT transfers, the Transfer Size field in the endpoint's Transfer Size register must be a multiple of the maximum packet size of the endpoint, adjusted to the DWORD boundary.

- transfer size[epnum] = n * (mps[epnum] + 4 – (mps[epnum] mod 4))
- packet count[epnum] = n
- n > 0

3. In DMA mode, the core stores a received data packet in the memory, always starting on a DWORD boundary. If the maximum packet size of the endpoint is not a multiple of 4, the core inserts byte pads at end of a maximum-packet-size packet up to the end of the DWORD.

4. On any OUT endpoint interrupt, the application must read the endpoint's Transfer Size register to calculate the size of the payload in the memory. The received payload size can be less than the programmed transfer size.

- Payload size in memory = application-programmed initial transfer size – core updated final transfer size
- Number of USB packets in which this payload was received = applicationprogrammed initial packet count – core updated final packet count

Internal Data Flow

1. The application must set the Transfer Size and Packet Count fields in the endpointspecific registers, clear the NAK bit, and enable the endpoint to receive the data.

2. Once the NAK bit is cleared, the core starts receiving data and writes it to the receive FIFO, as long as there is space in the receive FIFO. For every data packet received on the USB, the data packet and its status are written to the receive FIFO. Every packet (maximum packet size or short packet) written to the receive FIFO decrements the Packet Count field for that endpoint by 1.

- OUT data packets received with Bad Data CRC are flushed from the receive FIFO automatically.
- After sending an ACK for the packet on the USB, the core discards non-isochronous OUT data packets that the host, which cannot detect the ACK, re-sends. The application does not detect multiple back-to-back data OUT packets on the same endpoint with the same data PID. In this case the packet count is not decremented.
- If there is no space in the receive FIFO, isochronous or non-isochronous data packets are ignored and not written to the receive FIFO. Additionally, non-isochronous OUT tokens receive a NAK handshake reply.
- In all the above three cases, the packet count is not decremented because no data is written to the receive FIFO.

3. When the packet count becomes 0 or when a short packet is received on the endpoint, the NAK bit for that endpoint is set. Once the NAK bit is set, the isochronous or nonisochronous data packets are ignored and not written to the receive FIFO, and nonisochronous OUT tokens receive a NAK handshake reply.

4. After the data is written to the receive FIFO, either the application (in Slave mode) or the core's DMA engine (in External or Internal DMA mode), reads the data from the receive FIFO and writes it to external memory, one packet at a time per endpoint.

5. At the end of every packet write on the AHB to external memory, the transfer size for the endpoint is decremented by the size of the written packet.

6. The OUT Data Transfer Completed pattern for an OUT endpoint is written to the receive FIFO on one of the following conditions.

- The transfer size is 0 and the packet count is 0
- The last OUT data packet written to the receive FIFO is a short packet (0≤packet size< maximum packet size)

7. When either the application or the DMA pops this entry (OUT Data Transfer Completed),

- Transfer Completed interrupt is generated for the endpoint and the endpoint enable is cleared.

Application Programming Sequence

1. Program the DOEPTSIZn register for the transfer size and the corresponding packet count. Additionally, in DMA mode, program the DOEPWMAn register.

2. Program the DOEPCTLn register with the endpoint characteristics, and set the Endpoint Enable and ClearNAK bits.

- DOEPCTLn.EPEna = 1
- DOEPCTLn.CNAK = 1

3. In Slave mode, wait for the GINTSTS.Rx StsQ level interrupt and empty the data packets from the receive FIFO as explained in “Packet Read in Slave Mode”.

- This step can be repeated many times, depending on the transfer size.

4. Asserting the DOEPINTn.XferCompl interrupt marks a successful completion of the nonisochronous OUT data transfer.

5. Read the DOEPTSIZn register to determine the size of the received data payload.

[Generic non-Isochronous OUT Data Transfer with Thresholding]

This section describes a regular non-ISO OUT data transfer (Control/Bulk/Intr) when thresholding is enabled.

Application Requirements

Application requirements is the same as without thresholding.

Internal Data Flow

1. The application should set the transfer size and packet count fields in the endpoint specific registers, clear the NAK bit and enable the endpoint to receive the data.

2. Once the NAK bit is cleared, the core starts receiving the data and writes it into the receive FIFO, as long as there is threshold amount of space in the receive FIFO. For every threshold amount of data received on the USB, the data packet and the threshold status are written into the receive FIFO. At the end of a packet, a last threshold status and also a data update status is written into the receive FIFO.If it was the last packet of the transfer, then a transfer complete status is also written into the receive FIFO.On every packet (mps sized or short packet) written into the receive FIFO, the packet count field for that endpoint is decremented by 1.

- OUT data packets received with Bad Data CRC are flushed out of the receive FIFO automatically.The core also rewind the DMA pointers internally.
- non-ISO OUT data packet re-sent by the host, because the ACK was not seen by the host, will be discarded by the core, after sending an ACK for the packet on the USB. The application will not see multiple back to back data OUT packets on the same endpoint, with the same data PID. In this case the packet count is not decremented.
- If there is no space for at least threshold amount of data in the receive FIFO, the ISO/non-ISO data packets are ignored and not written into the receive FIFO. In addition, the non-ISO OUT tokens are responded with NAK handshake.
- If the core sees an overflow case (no space in the fifo in the middle of a packet reception), then the core stops writing the remaining data into the fifo and sends a NAK handshake on the USB. The core rewinds the fifo pointer to the threshold boundary, so that the portion of the threshold data that is in the fifo is flushed out.The core also rewinds the

DMA pointers.The core also sets DOEPINTn.OutPktErr (This interrupt bit is mainly used for debug purpose).

In all the above cases, the packet count is not decremented because no data is written into the receive FIFO.

In High Speed, after the core has received a packet, the core sends a NYET handshake if the core does not find threshold amount of free space available in the FIFO.

3. When the packet count becomes 0 or when a short packet is received on the endpoint, the NAK bit for that endpoint is set. Once the NAK bit is set, the ISO/non-ISO data packets are ignored and not written into the receive FIFO and the non-ISO OUT tokens are responded with a NAK handshake.

4. The DMA engine will transfer data from the receive FIFO to the system memory as soon as it sees one threshold amount of data in the FIFO.

5. At the end of every packet write on the AHB into the external memory, the transfer size for the endpoint is decremented by the size of packet written into the memory.

6. The OUT Data Transfer Complete pattern for an OUT endpoint is written into the receive FIFO, on one of the following conditions.

- The last threshold is written into FIFO and the packet count is decremented to 0
- If it is a short packet and the core sees the end of packet within a threshold.

7. When this entry (OUT Data Transfer Complete) is popped out by the DMA engine, Transfer Complete interrupt for the endpoint is generated and the endpoint enable is cleared.

8. “Rewind OUT Data Transfer” pattern is written into the receive FIFO, on one of the following conditions

- On seeing Overflow condition.
- On seeing a CRC error.

9. When this entry (Rewind OUT Data Transfer) is popped out by the DMA engine, it does the DMA pointer rewind.

Application Programming Sequence

This sequence is the same as in non thresholding case.

[Generic Isochronous OUT Data Transfer without Thresholding]

This section describes a regular isochronous OUT data transfer.

Application Requirements

1. All the application requirements for non-isochronous OUT data transfers also apply to isochronous OUT data transfers

2. For isochronous OUT data transfers, the Transfer Size and Packet Count fields must always be set to the number of maximum-packet-size packets that can be received in a single microframe and no more. Isochronous OUT data transfers cannot span more than 1 microframe.

- $1 \leq$ packet count[epnum] ≤ 3

3. In Slave mode, when isochronous OUT endpoints are supported in the device, the application must read all isochronous OUT data packets from the receive FIFO (data and status) before the end of the periodic frame (GINTSTS.EOPF interrupt). In DMA mode, the application must guarantee enough bandwidth to allow emptying the isochronous OUT data packet from the receive FIFO before the end of each periodic frame.

4. To receive data in the following frame/microframe, an isochronous OUT endpoint must be enabled after the GINTSTS.EOPF and before the GINTSTS.SOF.

Internal Data Flow

1. The internal data flow for isochronous OUT endpoints is the same as that for nonisochronous OUT endpoints, but for a few differences.

2. When an isochronous OUT endpoint is enabled by setting the Endpoint Enable and clearing the NAK bits, the Even/Odd frame/microframe bit must also be set appropriately. The core receives data on a isochronous OUT endpoint in a particular microframe only if the following condition is met.

- DOEPCTLn.Even/Odd microframe = DSTS.SOFFN[0]

3. When either the application or the external/internal DMA completely reads an isochronous OUT data packet (data and status) from the receive FIFO, the core updates the DOEPTSIZn.Received DPID field with the data PID of the last isochronous OUT data packet read from the receive FIFO.

Application Programming Sequence

1. Program the DOEPTSIZn register for the transfer size and the corresponding packet count. When in DMA mode, also program the DOEPWMAn register.

2. Program the DOEPCTLn register with the endpoint characteristics and set the Endpoint Enable, ClearNAK, and Even/Odd frame/microframe bits.

- Endpoint Enable = 1
- CNAK = 1
- Even/Odd frame/microframe = (0: Even/1: Odd)

3. In Slave mode, wait for the GINTSTS.Rx StsQ level interrupt and empty the data packets from the receive FIFO as explained in “Packet Read in Slave Mode”.

- This step can be repeated many times, depending on the transfer size.

4. The assertion of the DOEPINTn.XferCompl interrupt marks the completion of the isochronous OUT data transfer. This interrupt does not necessarily mean that the data in memory is good.

- This interrupt can not always be detected for isochronous OUT transfers. Instead, the application can detect the GINTSTS.incomplete Isochronous OUT data interrupt. See “Incomplete Isochronous OUT Data Transfers”, for more details

5. Read the DOEPTSIZn register to determine the size of the received transfer and to determine the validity of the data received in the microframe. The application must treat the data received in memory as valid only if one of the following conditions is met.

- DOEPTSIZn.RxDPID = D0 and the number of USB packets in which this payload was received = 1
- DOEPTSIZn.RxDPID = D1 and the number of USB packets in which this payload was received = 2
- DOEPTSIZn.RxDPID = D2 and the number of USB packets in which this payload was received = 3
  - The number of USB packets in which this payload was received = App Programmed Initial Packet Count – Core Updated Final Packet Count

The application can discard invalid data packets.

[Generic Isochronous OUT Data Transfer with Thresholding]

This section describes a regular isochronous OUT data transfer.

Application Requirements

There is no change in this section from a non-thresholding mode.

Internal Data Flow

1. The internal data flow for isochronous OUT Endpoints when thresholding is enabled, is the same as that for the non isochronous OUT endpoints when thresholding is enabled, but for a few differences.

2. If MAC sees an overflow condition when writing a packet, it stops writing.The current threshold amount of data that is being written into the receive FIFO is flushed out at the end of packet.This will eventually result in GINTSTS.incomplete ISO OUT Data interrupt. Refer to the section Incomplete isochronous OUT Data Transfers, for more details.

3. If MAC sees a CRC error for the receiving packet, the last threshold being written into the receive FIFO is flushed out. This will eventually result in GINTSTS.incomplete isochronous OUT Data interrupt. Refer to the section Incomplete isochronous OUT Data Transfers, for more details.

4. Assertion of DOEPINTn.XferCompl interrupt marks a completion of the isochronous OUT Data Transfer. This interrupt may not necessarily mean that data in the memory is good data.

5. Read the DOEPTSIZn register, to find out the size of the received transfer and to find out the validity of the data received in the microframe. The application should treat the data received into the memory as valid only if one of the following conditions is met. Invalid data packets may be discarded by the application.

- DOEPTSIZn.RxDPID = D0 and Number of USB Packets in which this payload was received = 1
- DOEPTSIZn.RxDPID = D1 and Number of USB Packets in which this payload was received = 2
- DOEPTSIZn.RxDPID = D2 and Number of USB Packets in which this payload was received = 3

• Number of USB Packets in which this payload was received = App Programmed Initial Packet Count - Core Updated Final Packet Count

[Incomplete Isochronous OUT Data Transfers]

This section describes the application programming sequence when isochronous OUT data packets are dropped inside the core.

Internal Data Flow

1. For isochronous OUT endpoints, the DOEPINTn.XferCompl interrupt possibly is not always asserted. If the core drops isochronous OUT data packets, the application could fail to detect the DOEPINTn.XferCompl interrupt under the following circumstances.

- When the receive FIFO cannot accommodate the complete ISO OUT data packet, the core drops the received ISO OUT data. In thresholding this is same as overflow.
- When the isochronous OUT data packet is received with CRC errors
- When the isochronous OUT token received by the core is corrupted
- When the application is very slow in reading the data from the receive FIFO

2. When the core detects an end of periodic frame before transfer completion to all isochronous OUT endpoints, it asserts the GINTSTS.incomplete Isochronous OUT data interrupt, indicating that a DOEPINTn.XferCompl interrupt is not asserted on at least one of the isochronous OUT endpoints. At this point, the endpoint with the incomplete transfer remains enabled, but no active transfers remains in progress on this endpoint on the USB.

Application Programming Sequence

1. Asserting the GINTSTS.incomplete Isochronous OUT data interrupt indicates that in the current microframe, at least one isochronous OUT endpoint has an incomplete transfer.

- If this occurs because isochronous OUT data is not completely emptied from the endpoint, the application must ensure that the DMA or the application empties all isochronous OUT data (data and status) from the receive FIFO before proceeding.
- When all data is emptied from the receive FIFO, the application can detect the DOEPINTn.XferCompl interrupt. In this case, the application must re-enable the endpoint to receive isochronous OUT data in the next microframe, as described in “Generic Isochronous OUT Data Transfer without Thresholding”.

2. When it receives a GINTSTS.incomplete Isochronous OUT data interrupt, the application must read the control registers of all isochronous OUT endpoints (DOEPCTLn) to determine which endpoints had an incomplete transfer in the current microframe. An endpoint transfer is incomplete if both the following conditions are met.

- DOEPCTLn.Even/Odd microframe bit = DSTS.SOFFN[0]
- DOEPCTLn.Endpoint Enable = 1

3. The previous step must be performed before the GINTSTS.SOF interrupt is detected, to ensure that the current microframe number is not changed.

4. For isochronous OUT endpoints with incomplete transfers, the application must discard the data in the memory and disable the endpoint by setting the DOEPCTLn.Endpoint Disable bit.

5. Wait for the DOEPINTn.Endpoint Disabled interrupt and enable the endpoint to receive new data in the next microframe as explained in “Generic Isochronous OUT Data Transfer without Thresholding”.

- Because the core can take some time to disable the endpoint, the application possibly is not able to receive the data in the next microframe after receiving bad isochronous data.

[Stalling a Non-Isochronous OUT Endpoint]

This section describes how the application can stall a non-isochronous endpoint.

1. Put the core in the Global OUT NAK mode, as described in “Setting the Global OUT NAK”.

2. Disable the required endpoint, as described in “Disabling an OUT Endpoint”.

- When disabling the endpoint, instead of setting the DOEPCTL.SNAK bit, set DOEPCTL.STALL = 1.
  - The Stall bit always takes precedence over the NAK bit.

3. When the application is ready to end the STALL handshake for the endpoint, the DOEPCTLn.STALL bit must be cleared.

4. If the application is setting or clearing a STALL for an endpoint due to a SetFeature.Endpoint Halt or ClearFeature.Endpoint Halt command, the Stall bit must be set or cleared before the application sets up the Status stage transfer on the control endpoint.

[Examples]

This section describes and depicts some fundamental transfer types and scenarios.

Slave Mode Bulk OUT Transaction

Figure below depicts the reception of a single Bulk OUT Data packet from the USB to the AHB and describes the events involved in the process.

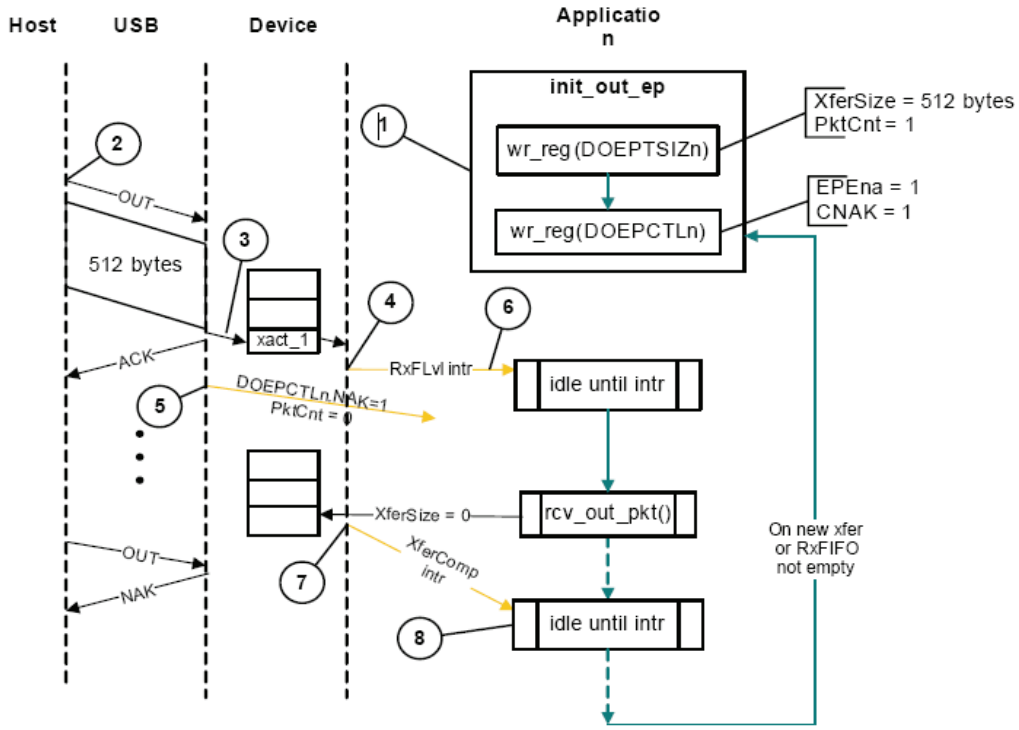


Figure 13.22 Slave Mode Bulk OUT Transaction

1. After a SetConfiguration/SetInterface command, the application initializes all OUT endpoints by setting DOEPCTLn.CNAK = 1 and DOEPCTLn.EPEna = 1, and setting a suitable XferSize and PktCnt in the DOEPTSIZn register.

2. Host attempts to send data (OUT token) to an endpoint.

3. When the core receives the OUT token on the USB, it stores the packet in the RxFIFO because space is available there.

4. After writing the complete packet in the RxFIFO, the core then asserts the GINTSTS.RxFLvl interrupt.

5. On receiving the PktCnt number of USB packets, the core sets the NAK bit for this endpoint internally to prevent it from receiving any more packets.

6. The application processes the interrupt and reads the data from the RxFIFO.

7. When the application has read all the data (equivalent to XferSize), the core generates a DOEPINTn.XferCompl interrupt.

8. The application processes the interrupt and uses the setting of the DOEPINTn.XferCompl interrupt bit to determine that the intended transfer is complete.

13.7.2.2 IN Data Transfers

This section describes IN data transfer details.

[Packet Write in Slave Mode]

This section describes how the application writes data packets to the endpoint FIFO in Slave mode.

1. The application can either choose polling or interrupt mode.

- In polling mode, application monitors the status of the endpoint transmit data FIFO, by reading the DTXFSTSn register, to determine, if there is enough space in the data FIFO.
- In interrupt mode, application waits for the DIEPINTn.TxFEmp interrupt and then reads the DTXFSTSn register, to determine, if there is enough space in the data FIFO.
- To write a single non-zero length data packet, there must be space to write the entire packet is the data FIFO.
- For writing zero length packet, application must not look for FIFO space.

2. Using one of the above mentioned methods, when the application determines that there is enough space to write a transmit packet, the application must first write into the endpoint control register, before writing the data into the data FIFO. The application, typically must do a read modify write on the DIEPCTLn, to avoid modifying the contents of the register, except for setting the Endpoint Enable bit.

The application can write multiple packets for the same endpoint, into the transmit FIFO, if space is available. For periodic IN endpoints, application must write packets only for one microframe. It can write packets for the next periodic transaction, only after getting transfer complete for the previous transaction.

[Setting IN Endpoint NAK]

Internal Data Flow

1. When the application sets the IN NAK for a particular endpoint, the core stops transmitting data on the endpoint, irrespective of data availability in the endpoint's transmit FIFO.

- Non-isochronous IN tokens receive a NAK handshake reply
- Isochronous IN tokens receive a zero-data-length packet reply

2. The core asserts the DIEPINTn.IN NAK Effective interrupt in response to the DIEPCTL.Set NAK bit.

3. Once this interrupt is seen by the application, the application can assume that the endpoint is in IN NAK mode. This interrupt can be cleared by the application by setting the DIEPCTLn. Clear NAK bit.

Application Programming Sequence

1. To stop transmitting any data on a particular IN endpoint, the application must set the IN NAK bit. To set this bit, the following field must be programmed.

- DIEPCTLn.SetNAK = 1'b1

2. Wait for assertion of the DIEPINTn.NAK Effective interrupt. This interrupt indicates the core has stopped transmitting data on the endpoint.

3. The core can transmit valid IN data on the endpoint after the application has set the NAK bit, but before the assertion of the NAK Effective interrupt.

4. The application can mask this interrupt temporarily by writing to the DIEPMSK.NAK Effective bit.

- DIEPMSK.NAK Effective = 1'b0

5. To exit Endpoint NAK mode, the application must clear the DIEPCTLn.NAK status. This also clears the DIEPINTn.NAK Effective interrupt.

- DIEPCTLn.ClearNAK = 1'b1

6. If the application masked this interrupt earlier, it must be unmasked as follows:

- DIEPMSK.NAK Effective = 1'b1

[IN Endpoint Disable]

Use the following sequence to disable a specific IN endpoint (periodic/non-periodic) that has been previously enabled.

Application Programming Sequence

1. In Slave mode, the application must stop writing data on the AHB, for the IN endpoint to be disabled.

2. The application must set the endpoint in NAK mode. Refer to the section Setting the Endpoint NAK

- DIEPCTLn.SetNAK = 1'b1

3. Wait for DIEPINTn.NAK Effective interrupt.

4. Set the following bits in the DIEPCTLn register for the endpoint that must be disabled.

- DIEPMSK.NAK Effective = 1'b1
- DIEPCTLn.Endpoint Disable = 1
- DIEPCTLn.SetNAK = 1

5. Assertion of DIEPINTn.Endpoint Disabled interrupt indicates that the core has completely disabled the specified endpoint. Along with the assertion of the interrupt, the core also clears the following bits.

- DIEPCTLn.EPEnable = 1'b0
- DIEPCTLn.EPDisable = 1'b0

6. The application must read the DIEPTSIZn register for the periodic IN EP, to calculate how much data on the endpoint was transmitted on the USB.

7. The application must flush the data in the Endpoint transmit FIFO, by setting the following fields in the GRSTCTL register.

- GRSTCTL.TxFIFONum = Endpoint Transmit FIFO Number
- GRSTCTL.TxFFlush = 1

The application must poll the GRSTCTL register, until the TxFFlush bit is cleared by the core, which indicates the end of flush operation. To transmit new data on this endpoint, the application can re-enable the endpoint at a later point.

[Generic Non-periodic IN Data Transfers without Thresholding]

This section describes a regular non periodic IN data transfer when transmit thresholding is not enabled.

Application Requirements

1. Before setting up an IN transfer, the application must ensure that all data to be transmitted as part of the IN transfer is part of a single buffer, and must program the size of that buffer and its start address (in DMA mode) to the endpoint-specific registers.

2. For IN transfers, the Transfer Size field in the Endpoint Transfer Size register denotes a payload that constitutes multiple maximum-packet-size packets and a single short packet. This short packet is transmitted at the end of the transfer.

- To transmit a few maximum-packet-size packets and a short packet at the end of the transfer:
  • Transfer size[epnum] = n * mps[epnum] + sp
  (where n is an integer $\geq$ 0, and 0 $\leq$ sp < mps[epnum])
  • If (sp > 0), then packet count[epnum] = n + 1.
  Otherwise, packet count[epnum] = n
- To transmit a single zero-length data packet:
  • Transfer size[epnum] = 0
  • Packet count[epnum] = 1
- To transmit a few maximum-packet-size packets and a zero-length data packet at the end of the transfer, the application must split the transfer in two parts. The first sends maximum-packet-size data packets and the second sends the zero-length data packet alone.
  • First transfer: transfer size[epnum] = n * mps[epnum]; packet count = n;
  • Second transfer: transfer size[epnum] = 0; packet count = 1;

3. In DMA mode, the core fetches an IN data packet from the memory, always starting at a DWORD boundary. If the maximum packet size of the IN endpoint is not a multiple of 4, the application must arrange the data in the memory with pads inserted at the end of a maximum-packet-size packet so that a new packet always starts on a DWORD boundary.

4. Once an endpoint is enabled for data transfers, the core updates the Transfer Size register. At the end of IN transfer or Endpoint Disabled interrupt, the application must read the Transfer Size register to determine how much data posted in the transmit FIFO was already sent on the USB.

- Data fetched into transmit FIFO = Application-programmed initial transfer size – core-updated final transfer size
- Data transmitted on USB = (application-programmed initial packet count – Core updated final packet count) * mps[epnum]
- Data yet to be transmitted on USB = (Application-programmed initial transfer size – data transmitted on USB)

Internal Data Flow

1. The application must set the Transfer Size and Packet Count fields in the endpointspecific registers and enable the endpoint to transmit the data.

2. In Slave mode, the application must also write the required data to the transmit FIFO for the endpoint. In DMA mode, the core fetches the data from memory according to the application setting for the endpoint.

3. Every time a packet is written into the transmit FIFO, either by the core's internal DMA (in DMA mode) or the application (in Slave Mode), the transfer size for that endpoint is decremented by the packet size. The data is fetched from the memory (DMA/
Application), until the transfer size for the endpoint becomes 0. After writing the data into the FIFO, the "number of packets in FIFO" count is incremented (this is a 3-bit count, internally maintained by the core for each IN endpoint transmit FIFO. The maximum number of packets maintained by the core at any time in an IN endpoint FIFO is eight). For zero-length packets, a separate flag is set for each FIFO, without any data in the FIFO.

4. Once the data is written to the transmit FIFO, the core reads it out upon receiving an IN token. For every non-isochronous IN data packet transmitted with an ACK handshake, the packet count for the endpoint is decremented by one, until the packet count is zero. The packet count is not decremented on a TIMEOUT.

5. For zero length packets (indicated by an internal zero length flag), the core sends out a zero-length packet for the IN token and decrements the Packet Count field.

6. If there is no data in the FIFO for a received IN token and the packet count field for that endpoint is zero, the core generates a IN Tkn Rcvd When FIFO Empty Interrupt for the endpoint, provided the endpoint NAK bit is not set. The core responds with a NAK handshake for non-isochronous endpoints on the USB.

7. When the transfer size is 0 and the packet count is 0, the transfer complete interrupt for the endpoint is generated and the endpoint enable is cleared.

Application Programming Sequence

1. Program the DIEPTSIZn register with the transfer size and corresponding packet count. In DMA mode, also program the DIEPWMAn register.

2. Program the DIEPCTLn register with the endpoint characteristics and set the CNAK and Endpoint Enable bits. In DMA mode, ensure that the NextEp field is programmed so that the core fetches the data for IN endpoints in the correct order. See "Non-periodic IN Endpoint Sequencing" for details.

3. When transmitting non-zero length data packet in slave mode, the application must poll the DTXFSTSn register (where n is the FIFO number associated with that endpoint) to determine whether there is enough space in the data FIFO. The application can optionally use DIEPINTn.TxFEmp before writing the data.

[Generic non-periodic IN Data Transfers with Thresholding]

This section describes a regular non periodic IN data transfer when transmit thresholding is enabled.

Application Requirements

Application requirements are the same as those for DMA mode with no thresholding.

Internal Data Flow

1. The application should set the transfer size and packet count fields in the Endpoint pecific registers, and enable the endpoint to transmit the data.

2. The core fetches threshold amount of data for one endpoint before switching to the next in a round robin fashion with equal fairness. The priority is given to periodic endpoints. non-periodic endpoint data is fetched only if data for the periodic endpoints has been fetched or if the currently active token on the USB is a non-periodic one.The core will not switch, and continue to fetch till the complete packet, if it finds that the currently active IN token on the USB is for that particular endpoint and the FIFO does not have the omplete packet.

3. In response to an IN token on the USB, the core starts transmitting data, if the MAC finds t least threshold amount of data in the FIFO for that particular endpoint.

4. With each threshold amount of data written into the FIFO, the transfer size for that endpoint is decremented by the threshold size, except for the last packet. For the last packet, the transfer size is not decremented by the core. After writing the first threshold amount of data into the FIFO, the “number of packets in FIFO” count is incremented. For zero-length packets, a separate flag is set for that endpoint FIFO, without any data in the FIFO. This count is internally maintained by the core and is decremented when a full packet has been read out of the FIFO.

5. If the MAC sees an underrun case, where there is not enough data in the FIFO, the core will corrupt the data (invert the CRC) on the USB. The core will internally flush the FIFO, rewinding the DMA pointers and re fetch the packet(s).Also DIEPINTn.TxfifoUndrn bit will be set as an indication to the application.

6. For every non-ISO data IN packet transmitted, with an ACK handshake, the packet count for the endpoint is decremented by 1, until the packet count becomes zero. The packet count is not decremented on a TIMEOUT or underrun condition.
7. If the zero length flag in the FIFO is set (internally set by the core, when the application enables and endpoint for zero length), then core sends out zero length packet for the IN token and decrements the packet count field.
8. If there is no data (or partial threshold data) in the FIFO for a received IN token, and the packet count field for that endpoint is 0, the core generates a IN Tkn Rcvd When FIFO Empty Interrupt for the endpoint. The core responds with a NAK handshake for the non-ISO endpoints on the USB.

9. If the core does not receive a handshake after sending the packet (TIMEOUT), the core nternally flush the FIFO, rewind the DMA pointers and re-fetch the packet(s).

10. When the packet count is zero, the transfer complete interrupt for the endpoint is generated, and then the endpoint enable and transfer size are both cleared by the core.

Application Programming Sequence

1. Program the DIEPTSIZn register, for the transfer size and the corresponding packet count and the DIEPWMAn register.

2. Program the DIEPCTLn register, with the Endpoint Characteristics and set the CNAK and the Endpoint Enable bit.Also specify the Tx FIFO number in the DIEPCTLn.TXFNum field.

3. Assertion of DIEPINTn.XferCompl interrupt marks the successful completion of the nonperiodic IN transfer. Read to DIEPTSIZn register should indicate, a transfer size = 0 and packet count = 0, indicating all the data is transmitted on the USB.

[Generic Periodic IN Data Transfers without Thresholding]

This section describes a typical Periodic IN data transfer when thresholding is not enabled. Application Requirements

1. Application requirements 1, 2, 3, and 4 of “Generic Non-periodic IN Data Transfers without Thresholding” also apply to periodic IN data transfers, except for a slight modification of Requirement 2.

- The application can only transmit multiples of maximum-packet-size data packets or multiples of maximum-packet-size packets, plus a short packet at the end. To transmit a few maximum-packet-size packets and a short packet at the end of the transfer, the following conditions must be met.
  - transfer size[epnum] = n * mps[epnum] + sp
  (where n is an integer $\geq$ 0, and 0 $\leq$ sp < mps[epnum])
  - If (sp > 0), packet count[epnum] = n + 1

Otherwise, packet count[epnum] = n;
- mc[epnum] = packet count[epnum]

- The application cannot transmit a zero-length data packet at the end of transfer. It can transmit a single zero-length data packet by it self. To transmit a single zero-length data packet.
  - transfer size[epnum] = 0
  - packet count[epnum] = 1
  - mc[epnum] = packet count[epnum]

2. The application can only schedule data transfers 1 microframe at a time.

- (DIEPTSIZn.MC−1) * DIEPCTLn.MPS ≤ DIEPTSIZn.XferSiz ≤ DIEPTSIZn.MC * DIEPCTLn.MPS
- DIEPTSIZn.PktCnt = DIEPTSIZn.MC
- If DIEPTSIZn.XferSiz < DIEPTSIZn.MC * DIEPCTLn.MPS, the last data packet of the transfer is a short packet.

3. The application can schedule data transfers for multiple microframes, only if multiples of max packet sizes (up to 3 packets), should be transmitted every microframe. This is can be done, only when the core is operating in DMA mode. This is not a recommended mode of operation though.

- ((n*DIEPTSIZn.MC) - 1)*DIEPCTLn.MPS <= DIEPTSIZn.Transfer Size <= n*DIEPTSIZn.MC*DIEPCTLn.MPS
- DIEPTSIZn.Packet Count = n*DIEPTSIZn.MC
- n is the number of microframes for which the data transfers are scheduled

Data Transmitted per microframe in this case would be DIEPTSIZn.MC*DIEPCTLn.MPS, in all the microframes except the last one. In the microframe “n”, the data transmitted would be (DIEPTSIZn.TransferSize - (n-1)*DIEPTSIZn.MC*DIEPCTLn.MPS)

4. For Periodic IN endpoints, the data must always be prefetched 1 microframe ahead for transmission in the next microframe. This can be done, by enabling the Periodic IN endpoint 1 (micro)frame ahead of the (micro)frame in which the data transfer is scheduled.

5. The complete data to be transmitted in the (micro)frame must be written into the transmit FIFO (either by the application or the DMA), before the Periodic IN token is received. Even when 1 DWORD of the data to be transmitted per microframe is missing in the transmit FIFO when the Periodic IN token is received, the core behaves as when the FIFO was empty. When the transmit FIFO is empty,

- A zero data length packet would be transmitted on the USB for ISO IN endpoints
- A NAK handshake would be transmitted on the USB for INTR IN endpoints

6. For a High Bandwidth IN endpoint with three packets in a microframe, the application can program the endpoint FIFO size to be 2*max_pkt_size and have the third packet load in after the first packet has been transmitted on the USB.

Internal Data Flow

1. The application must set the Transfer Size and Packet Count fields in the endpointspecific registers and enable the endpoint to transmit the data.

2. In Slave mode, the application must also write the required data to the associated transmit FIFO for the endpoint. In DMA mode, the core fetches the data for the endpoint from memory, according to the application setting.

3. Every time either the core’s internal DMA (in DMA mode) or the application (in Slave mode) writes a packet to the transmit FIFO, the transfer size for that endpoint is decremented by the packet size. The data is fetched from DMA or application memory until the transfer size for the endpoint becomes 0.

4. When an IN token is received for an periodic endpoint, the core transmits the data in the FIFO, if available. If the complete data payload (complete packet, in dedicated FIFO mode) for the microframe is not present in the FIFO, then the core generates an IN Tkn Rcvd When TxF Empty Interrupt for the endpoint.

- A zero-length data packet is transmitted on the USB for isochronous IN endpoints
- A NAK handshake is transmitted on the USB for interrupt IN endpoints

5. The packet count for the endpoint is decremented by 1 under the following conditions:

- For isochronous endpoints, when a zero- or non-zero-length data packet is transmitted
- For interrupt endpoints, when an ACK handshake is transmitted

6. When the transfer size and packet count are both 0, the Transfer Completed interrupt for the endpoint is generated and the endpoint enable is cleared.

7. At the “Periodic frame Interval” (controlled by DCFG.PerFrint), when the core finds nonempty any of the isochronous IN endpoint FIFOs scheduled for the current (micro)frame non-empty, the core generates a GINTSTS.incompISOIN interrupt.

Application Programming Sequence (Transfer Per Microframe)

1. Program the DIEPTSIZn register. In DMA mode, also program the DIEPWMAn register.

2. Program the DIEPCTLn register with the endpoint characteristics and set the CNAK and Endpoint Enable bits.

3. In Slave mode, write the data to be transmitted in the next microframe to the transmit FIFO as described in “Periodic Packet Write in Slave Mode: Shared FIFO”.

4. Asserting the DIEPINTn.In Token Rcvd When TxF Empty interrupt indicates that either the DMA or application has not yet written all data to be transmitted to the transmit FIFO.

- If the interrupt endpoint is already enabled when this interrupt is detected, ignore the interrupt. If it is not enabled, enable the endpoint so that the data can be transmitted on the next IN token attempt.
- If the isochronous endpoint is already enabled when this interrupt is detected, see “Incomplete Isochronous IN Data Transfers” for more details.

5. The core handles timeouts internally on interrupt IN endpoints programmed as periodic endpoints, or when Dedicated FIFO operation is used, without application intervention. The application, thus, never detects a DIEPINTn.TimeOUT interrupt for periodic interrupt IN endpoints.

6. Asserting the DIEPINTn.XferCompl interrupt with no DIEPINTn.In Tkn Rcvd When TxF Empty interrupt indicates the successful completion of an isochronous IN transfer. A read to the DIEPTSIZn register must indicate transfer size = 0 and packet count = 0, indicating all data is transmitted on the USB.

7. Asserting the DIEPINTn.XferCompl interrupt, with or without the DIEPINTn.In Tkn Rcvd When TxF Empty interrupt, indicates the successful completion of an interrupt IN transfer. A read to the DIEPTSIZn register must indicate transfer size = 0 and packet count = 0, indicating all data is transmitted on the USB.

8. Asserting the GINTSTS.incomplete Isochronous IN Transfer interrupt with none of the aforementioned interrupts indicates the core did not receive at least 1 periodic IN token in the current microframe.

- For isochronous IN endpoints, see “Incomplete Isochronous IN Data Transfers” for more details.

[Generic Periodic IN Data Transfers with Thresholding]

This section describes a typical Periodic IN data transfer when thresholding is enabled.

Application Requirements

Application requirements are the same as that for DMA mode with no thresholding.

Internal Data Flow

1. The application should set the transfer size and packet count fields in the Endpoint Specific registers, and enable the endpoint to transmit the data.

2. The core fetches threshold amount of data for one endpoint before switching to the next in a round robin fashion with equal fairness. The priority is given to periodic endpoints. The core will not switch, and continue to fetch till the complete packet, if it finds that the currently active IN token on the USB side is for that particular endpoint and the FIFO does not have the complete packet.

3. In response to an IN token on the USB, the core starts transmitting, if the MAC finds at least threshold amount of data in the FIFO for that particular endpoint.

4. After a full packet has been written into the FIFO, the transfer size for that endpoint is decremented by the packet size (or less). After writing the first threshold amount of data into the FIFO, the “number of packets in FIFO” count is incremented. For zero length packets, a separate flag will be set in the FIFO, without any data in the FIFO. This count is internally maintained b y the core and is decremented when a full packet has been read out of the FIFO.

5. If the MAC sees an underrun case, where there is not enough data in the FIFO, the core will corrupt the data (invert the CRC) on the USB. For isochronous endpoints, Underrun interrupt (DIEPINTn.TxFifoUndrn) is generated by the core for this endpoint. For interrupt endpoints, the core will flush the FIFO, rewind the DMA pointers and re-fetch the data.

6. If the core sees a timeout or an underrun condition for an interrupt endpoint, the core flushes the fifo, rewinds the DMA pointers and re-fetches the packet. No interrupt is generated to the application.

7. When an IN token is received for an periodic endpoint, the core transmits the data in the FIFO, if the core sees at least threshold amount of data. If the threshold amount of data for the packet is not present in the FIFO, the core generates a IN Tkn Rcvd When TxF Empty Interrupt for the endpoint.

- A zero data length packet would be transmitted on the USB for ISO IN endpoints
- A NAK handshake would be transmitted on the USB for INTR IN endpoints

8. The packet count for the endpoint is decremented by 1, on the following conditions

- When a zero or non zero data length for ISO endpoints
- When an ACK handshake is transmitted for INTR endpoints

9. The packet count is not decremented when the core has to corrupt a packet because of underrun condition.
10. When the transfer size is 0 and the packet count is 0, the transfer complete interrupt for the endpoint is generated and the endpoint enable is cleared.

Application Programming Sequence (Transfer Per Microframe)

1. Program the DIEPTSIZn register and in addition, in DMA mode program the DIEPWMAn register.

2. Program the DIEPCTLn register, with the Endpoint Characteristics and set the CNAK bit and the Endpoint Enable bit.Also specify the Tx FIFO number in the DIEPCTLn.TXFNum field.

3. Assertion of DIEPINTn.In Token Rcvd When TxF Empty interrupt indicates that the complete data to be transmitted is not written into the transmit FIFO.

- If the INTR endpoint is already enabled, when this interrupt is seen, ignore the interrupt. If the IN Endpoint in Dedicated Tx FIFO config is not enabled, enable the endpoint, so that the data could be transmitted on the next attempt of the IN token.
- If the ISO endpoint is already enabled, when this interrupt is seen, refer to the section Incomplete ISO IN Data Transfers, for more details.

4. TimeOUT on Interrupt IN endpoints, is handled by the core internally, without any application intervention. The application never sees any interrupt for timeout.

5. Assertion of DIEPINTn.XferCompl interrupt without any DIEPINTn.In Tkn Rcvd when TxF Empty interrupt, marks the successful completion of the ISO IN transfer. Read to DIEPTSIZn register should indicate, a transfer size = 0 and packet count = 0, indicating all the data is transmitted on the USB.

6. Assertion of DIEPINTn.XferCompl interrupt with/without any DIEPINTn.In Tkn Rcvd when TxF Empty interrupt, marks the successful completion of the INTR IN transfer. Read to DIEPTSIZn register should indicate, a transfer size = 0 and packet count = 0, indicating all the data is transmitted on the USB.

7. Assertion of GINTSTS.incomplete ISO IN Transfer interrupt with out any of the previously mentioned interrupts indicates that at least 1 periodic IN token was not received by the core, in the current microframe.

- For ISO IN endpoints, refer to the section Incomplete ISO IN Data Transfers, for more details.

8. Assertion of DIEPINTn.TxFifoUndrn interrupt indicate that there was an underrun condition for the isochronous transaction in the current microframe. The application may choose ignore this interrupt, as this will eventually result in GINTSTS.Incomplete isochronous IN interrupt at the end of periodic frame. The application can also choose to service this interrupt. If they choose to do so, then they can save some time in re-enabling the endpoint for the next microframe. In response to this interrupt,

- Disable the endpoint (Check section Disabling IN Endpoint in Dedicated Tx FIFO config).
- Application reads Endpoint DIEPSIZn register to see how much data is transferred.
- Re-enable the endpoint for the next microframe.

[Incomplete Isochronous IN Data Transfers]

This section describes what the application must do on an incomplete isochronous IN data transfer.

Internal Data Flow

1. An isochronous IN transfer is treated as incomplete in one of the following conditions.

- The core receives a corrupted isochronous IN token on at least one isochronous IN endpoint. In this case, the application detects a GINTSTS.incomplete Isochronous IN Transfer interrupt.
- The application or DMA is slow to write the complete data payload to the transmit FIFO and an IN token is received before the complete data payload is written to the FIFO. In this case, the application detects a DIEPINTn.IN Tkn Rcvd When TxFIFO Empty interrupt. The application can ignore this interrupt, as it eventually results in a GINTSTS.incomplete Isochronous IN Transfer interrupt at the end of periodic frame.
  • The core transmits a zero-length data packet on the USB in response to the received IN token.
- If thresholding is enabled and there was an underrun condition, the core also generates a DINEPINTn.TxfifoUndrn interrupt. The application can ignore this interrupt, which eventually results in a GINTSTS.incomplete ISO IN Transfer interrupt.

2. In either of the aforementioned cases, in Slave mode, the application must stop writing the data payload to the transmit FIFO as soon as possible.

3. The application must set the NAK bit and the disable bit for the endpoint. In DMA mode, the core automatically stops fetching the data payload when the endpoint disable bit is set.

4. The core disables the endpoint, clears the disable bit, and asserts the Endpoint Disable interrupt for the endpoint.

Application Programming Sequence

1. The application can ignore the DIEPINTn.IN Tkn Rcvd When TxFIFO empty interrupt on any isochronous IN endpoint, as it eventually results in a GINTSTS.incomplete Isochronous IN Transfer interrupt. The application can also ignore DIEPINTn.TxfifoUndrn interrupt when thresholding is enabled.

2. Assertion of the GINTSTS.incomplete Isochronous IN Transfer interrupt indicates an incomplete isochronous IN transfer on at least one of the isochronous IN endpoints.

3. The application must read the Endpoint Control register for all isochronous IN endpoints to detect endpoints with incomplete IN data transfers.

4. In Slave mode, the application must stop writing data to the Periodic Transmit FIFOs associated with these endpoints on the AHB.

5. In both modes of operation, program the following fields in the DIEPCTLn register to disable the endpoint.

- DIEPCTLn.SetNAK = 1
- DIEPCTLn.Endpoint Disable = 1

6. The DIEPINTn.Endpoint Disabled interrupt's assertion indicates that the core has disabled the endpoint.

- At this point, the application must flush the data in the associated transmit FIFO or overwrite the existing data in the FIFO by enabling the endpoint for a new transfer in the next microframe. To flush the data, the application must use the GRSTCTL register.

[Stalling Non-Isochronous IN Endpoints]

This section describes how the application can stall a non-isochronous endpoint.

Application Programming Sequence

1. Disable the IN endpoint to be stalled. Set the Stall bit as well.

- DIEPCTLn.Endpoint Disable = 1, when the endpoint is already enabled
- DIEPCTLn.STALL = 1
- The Stall bit always takes precedence over the NAK bit

2. Assertion of the DIEPINTn.Endpoint Disabled interrupt indicates to the application that the core has disabled the specified endpoint.

3. The application must flush the Non-periodic or Periodic Transmit FIFO, depending on the endpoint type. In case of a non-periodic endpoint, the application must re-enable the other non-periodic endpoints, which do not need to be stalled, to transmit data.

4. Whenever the application is ready to end the STALL handshake for the endpoint, the DIEPCTLn.STALL bit must be cleared.

5. If the application sets or clears a STALL for an endpoint due to a SetFeature.Endpoint Halt command or ClearFeature.Endpoint Halt command, the Stall bit must be set or cleared before the application sets up the Status stage transfer on the control endpoint.

Special Case: Stalling the Control OUT Endpoint

The core must stall IN/OUT tokens if, during the Data stage of a control transfer, the host sends more IN/OUT tokens than are specified in the SETUP packet. In this case, the application must to enable DIEPINTn.INTknTXFEmp and DOEPINTn.OUTTknEPdis interrupts during the Data stage of the control transfer, after the core has transferred the amount of data specified in the SETUP packet. Then, when the application receives this interrupt, it must set the STALL bit in the corresponding endpoint control register, and clear this interrupt.

[Examples]

Slave Mode Bulk IN Transaction

1. The host attempts to read data (IN token) from an endpoint.

2. On receiving the IN token on the USB, the core returns a NAK handshake, because no data is available in the transmit FIFO.

3. To indicate to the application that there was no data to send, the core generates a DIEPINTn.IN Token Rcvd When TxFIFO Empty interrupt.

4. When data is ready, the application sets up the DIEPTSIZn register with the Transfer Size and Packet Count fields.
5. The application writes one maximum packet size or less of data to the Non-periodic TxFIFO.

6. The host reattempts the IN token.

7. Because data is now ready in the FIFO, the core now responds with the data and the host ACKs it.

8. Because the XferSize is now zero, the intended transfer is complete. The device core generates a DIEPINTn.XferCompl interrupt.

9. The application processes the interrupt and uses the setting of the DIEPINTn.XferCompl interrupt bit to determine that the intended transfer is complete.

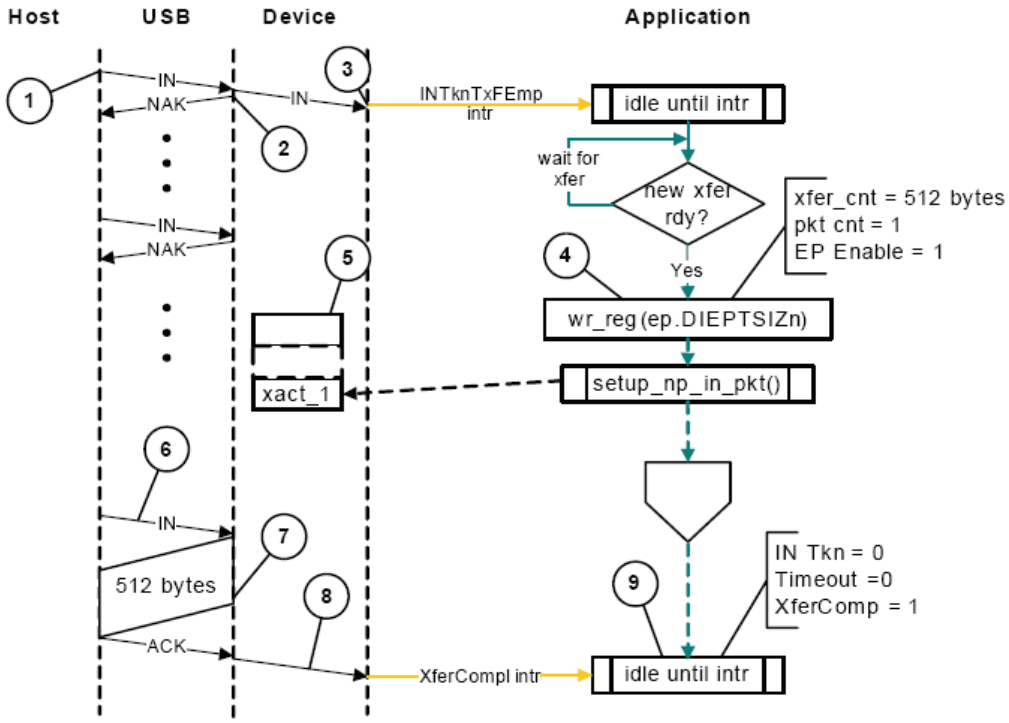


Figure 13.23 Slave Mode Bulk IN Transaction

Slave Mode Bulk IN Transfer (Pipelined Transaction)

1. The host attempts to read data (IN token) from an endpoint.

2. On receiving the IN token on the USB, the core returns a NAK handshake, because no data is available in the transmit FIFO.

3. To indicate that there was no data to send, the core generates an DIEPINTn.InTkn Rcvd When TxFIFO Empty interrupt.

4. When data is ready, the application sets up the DIEPTSIZn register with the transfer size and packet count.

5. The application writes one maximum packet size or less of data to the Non-periodic TxFIFO.

6. The host reattempts the IN token.

7. Because data is now ready in the FIFO, the core responds with the data, and the host ACKs it.

8. When the TxFIFO level falls below the halfway mark, the core generates a GINTSTS.NonPeriodic TxFIFO Empty interrupt. This triggers the application to start writing additional data packets to the FIFO.

9. A data packet for the second transaction is ready in the TxFIFO.

10. A data packet for third transaction is ready in the TxFIFO while the data for the second packet is being sent on the bus.
11. The second data packet is sent to the host.

12. The last short packet is sent to the host.

13. Because the last packet is sent and XferSize is now zero, the intended transfer is complete.

The core generates a DIEPINTn.XferCompl interrupt.

14. The application processes the interrupt and uses the setting of the DIEPINTn.XferCompl interrupt bit to determine that the intended transfer is complete.

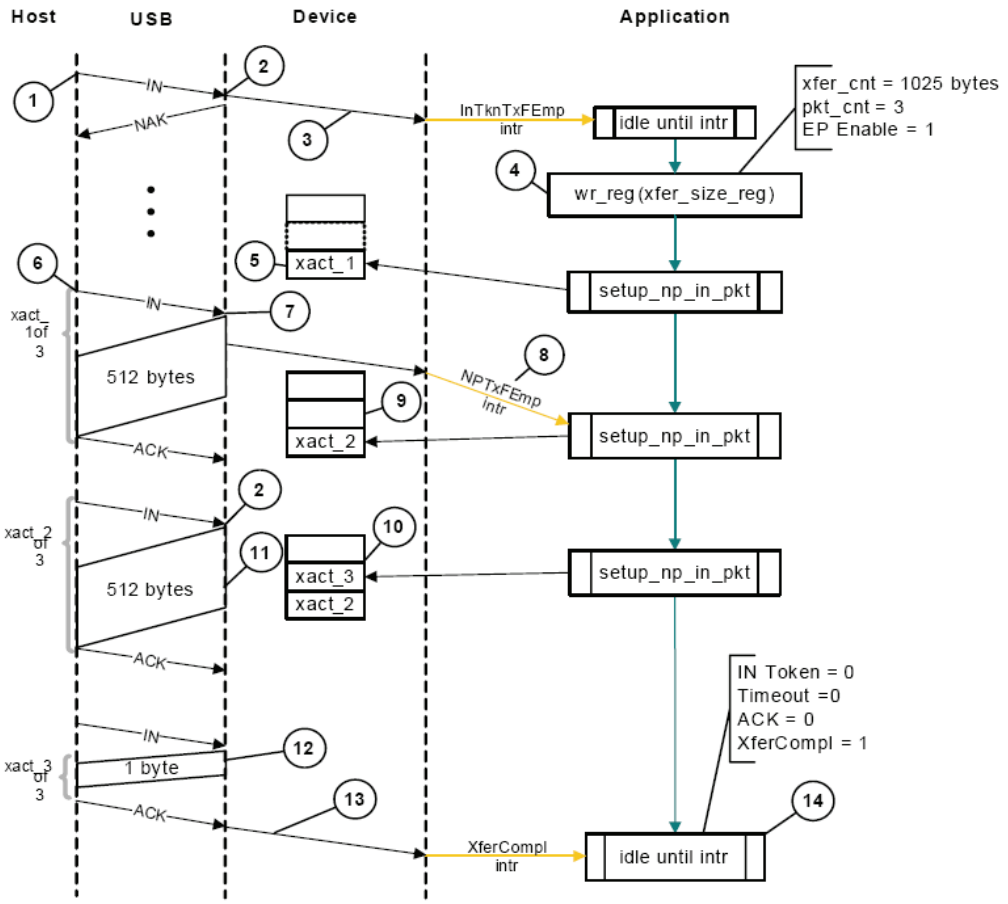


Figure 13.24 Slave Mode Bulk IN Transfer (Pipelined Transaction)

Slave Mode Bulk IN Two-Endpoint Transfer

1. The host attempts to read data (IN token) from endpoint 1.

2. On receiving the IN token on the USB, the core returns a NAK handshake, because no data is available in the transmit FIFO for endpoint 1, and generates a DIE PINT1.InTkn Rcvd When TxFIFO Empty interrupt.

3. The application processes the interrupt and initializes DIEPTSIZ1 register with the Transfer Size and Packet Count fields. The application starts writing the transaction data to the transmit FIFO.

4. The application writes one maximum packet size or less of data for endpoint 1 to the Non-periodic TxFIFO.

5. Meanwhile, the host attempts to read data (IN token) from endpoint 2.

6. On receiving the IN token on the USB, the core returns a NAK handshake, because no data is available in the transmit FIFO for endpoint 2, and the core generates a DIEPINT2.InTkn Rcvd When TxFIFO Empty interrupt.

7. Because the application has completed writing the packet for endpoint 1, it initializes the DIEPTSIZ2 register with the Transfer Size and Packet Count fields. The application starts writing the transaction data into the transmit FIFO for endpoint 2.

8. The host repeats its attempt to read data (IN token) from endpoint 1.

9. Because data is now ready in the TxFIFO, the core returns the data, which the host ACKs.

10. Meanwhile, the application has initialized the data for the next two packets in the TxFIFO (ep2.xact1 and ep1.xact2, in order).

11. The host repeats its attempt to read data (IN token) from endpoint 2.

12. Because endpoint 2's data is ready, the core responds with the data (ep2.xact_1), which the host ACKs.

13. Meanwhile, the application has initialized the data for the next two packets in the TxFIFO (ep2.xact2 and ep1.xact3, in order). The application has finished initializing data for the two endpoints involved in this scenario.

14. The host repeats its attempt to read data (IN token) from endpoint 1.

15. Because data is now ready in the FIFO, the core responds with the data, which the host ACKs.

16. The host repeats its attempt to read data (IN token) from endpoint 2.

17. With data now ready in the FIFO, the core responds with the data, which the host ACKs.

18. With the last packet for endpoint 2 sent and its XferSize now zero, the intended transfer is complete. The core generates a DIEPINT2.XferCompl interrupt for this endpoint.

19. The application processes the interrupt and uses the setting of the DIEPINT2.XferCompl interrupt bit to determine that the intended transfer on endpoint 2 is complete.

20. The host repeats its attempt to read data (IN token) from endpoint 1 (last transaction).

21. With data now ready in the FIFO, the core responds with the data, which the host ACKs.

22. Because the last endpoint one packet has been sent and XferSize is now zero, the intended transfer is complete. The core generates a DIEPINT1.XferCompl interrupt for this endpoint.

23. The application processes the interrupt and uses the setting of the DIEPINT1.XferCompl interrupt bit to determine that the intended transfer on endpoint 1 is complete.

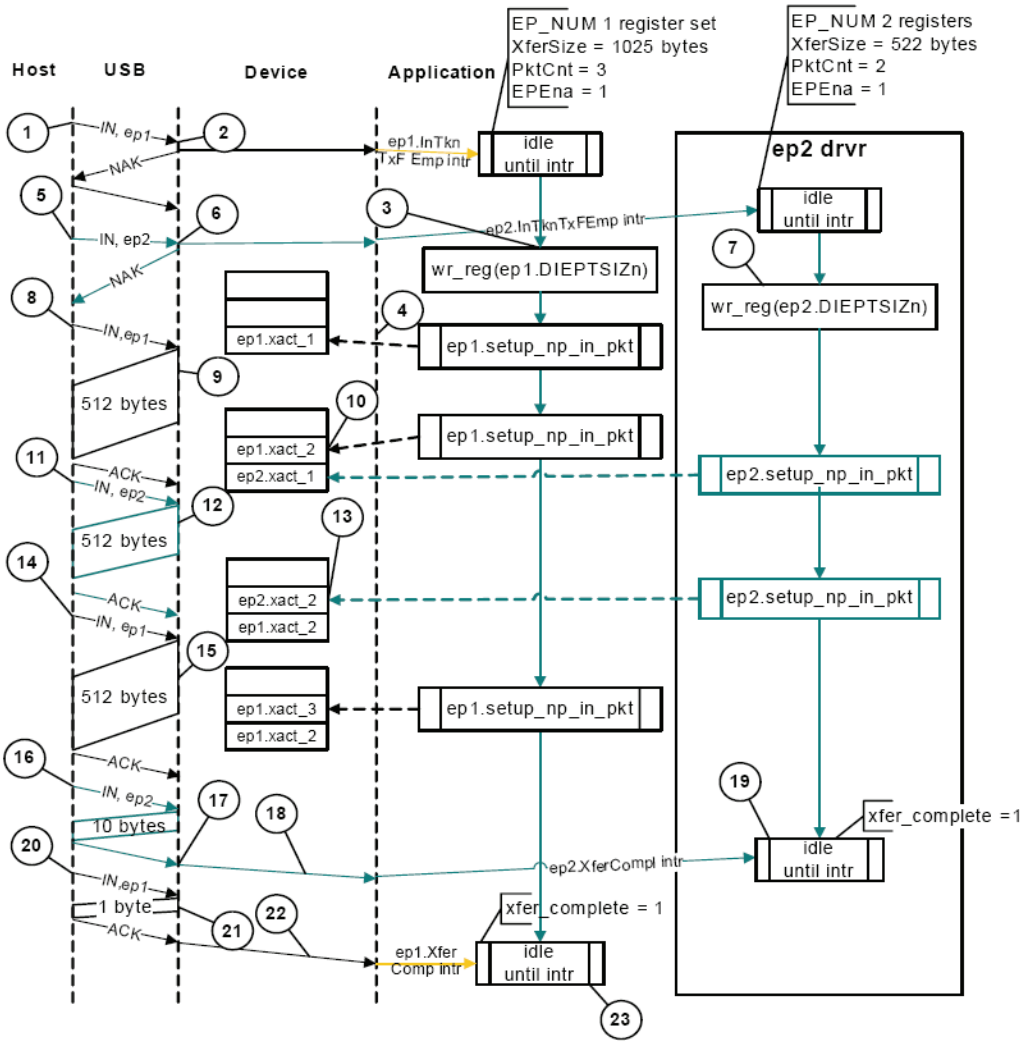


Figure 13.25 Slave Mode Bulk IN Two-Endpoint Transfer

Bulk IN Stall

1. The application has scheduled an IN transfer on receiving the DIEPINTn.InTknRcvd When TxFIFO Empty interrupt.

2. When the transfer is in progress, the application must force a STALL on the endpoint. This could be because the application has received a SetFeature.Endpoint Halt command. The application sets the Stall bit, disables the endpoint and waits for the DIEPINTn.Endpoint Disabled interrupt. This generates STALL handshakes for the endpoint on the USB.
3. On receiving the interrupt, the application flushes the Non-periodic Transmit FIFO and clears the DCTL.GlobalINNPNAK bit.

4. On receiving the ClearFeature.Endpoint Halt command, the application clears the Stall bit.

5. The endpoint behaves normally and the application can re-enable the endpoint for new transfers.

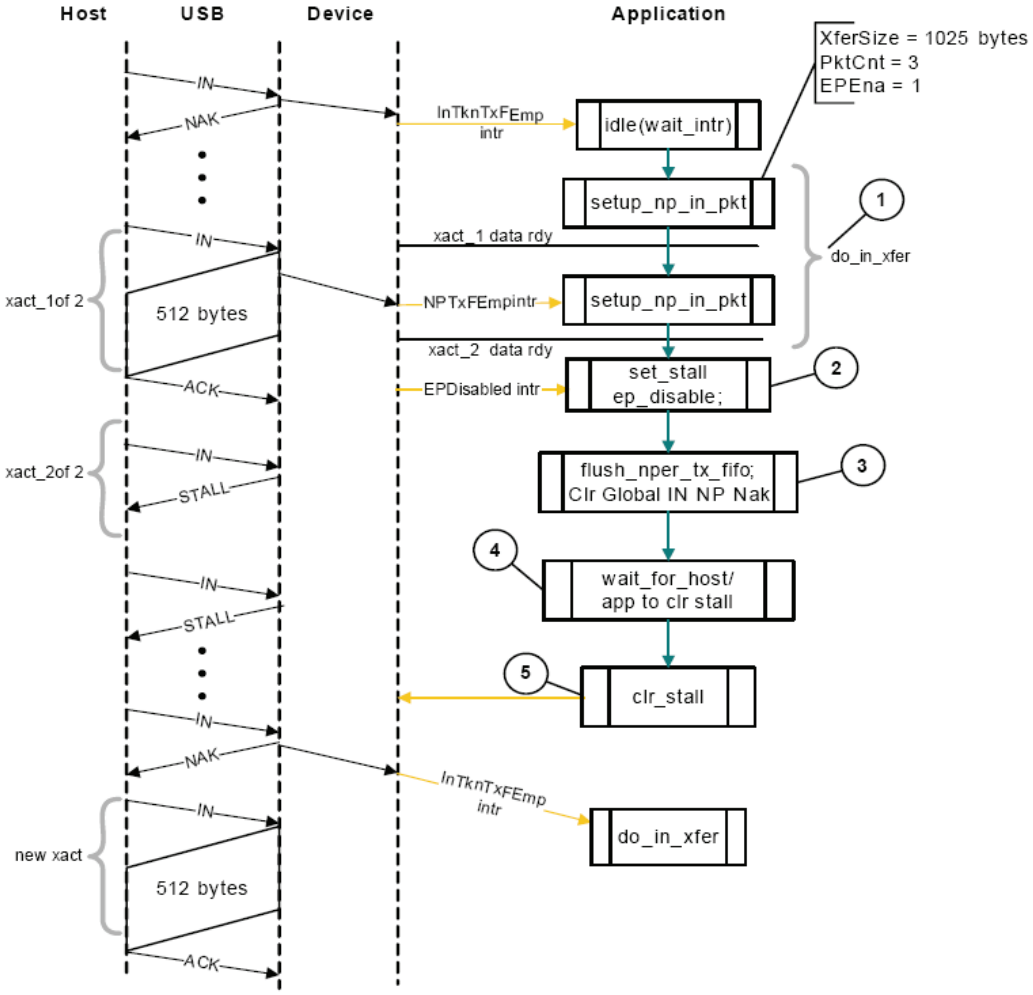


Figure 13.26 Bulk IN Stall

Bulk IN DMA mode with Thresholding

1. The host attempts to read data(IN token) from an endpoint

2. On receiving the IN token on the USB bus, the core sends back a NAK handshake because no data is available in the Transmit FIFO

3. To indicate to the application that there was no data to send, the core generates a DIEPINTn.IN Token Rcvd When TxF Empty interrupt.

4. When data is ready, the application sets up the DIEPTSIZn register with the TransferSize and Packet Count fields and enables the endpoint.

5. On seeing the endpoint enabled, the internal DMA engine start fetching the first threshold amount of data.

6. DMA engine fetching the second threshold.

7. The host re-attempts the IN token, and core start sending the data because it sees at least threshold amount of data in the FIFO.

8. The DMA engine starts fetching the third threshold after it sees threshold amount of space.

9. The MAC sees an underrun condition because of FIFO being empty in the middle of the packet, stops the packet and corrupt the CRC.

10. DMA engine starts to fetch the last threshold for that packet but it is late and will eventually result in underrun.

11. No handshake response from the host, because the packet was corrupted. Core rewinds the pointers and flush the FIFO.

12. The core starts to re-fetch the packet, staring with the first threshold.

13. The host re-attempts the IN token, and the core starts sending the data, since the core detects at least the threshold amount of data in the FIFO.

14. The core fetches the last threshold in time.

15. The core receives the handshake from host and Because the XferSize is now zero, the intended transfer is complete. Device core generates a DIEPINTn.XferCompl interrupt.

16. The application processes the interrupt and uses the setting of DIEPINTn.XferCompl interrupt bit to determine that the intended transfer is complete.

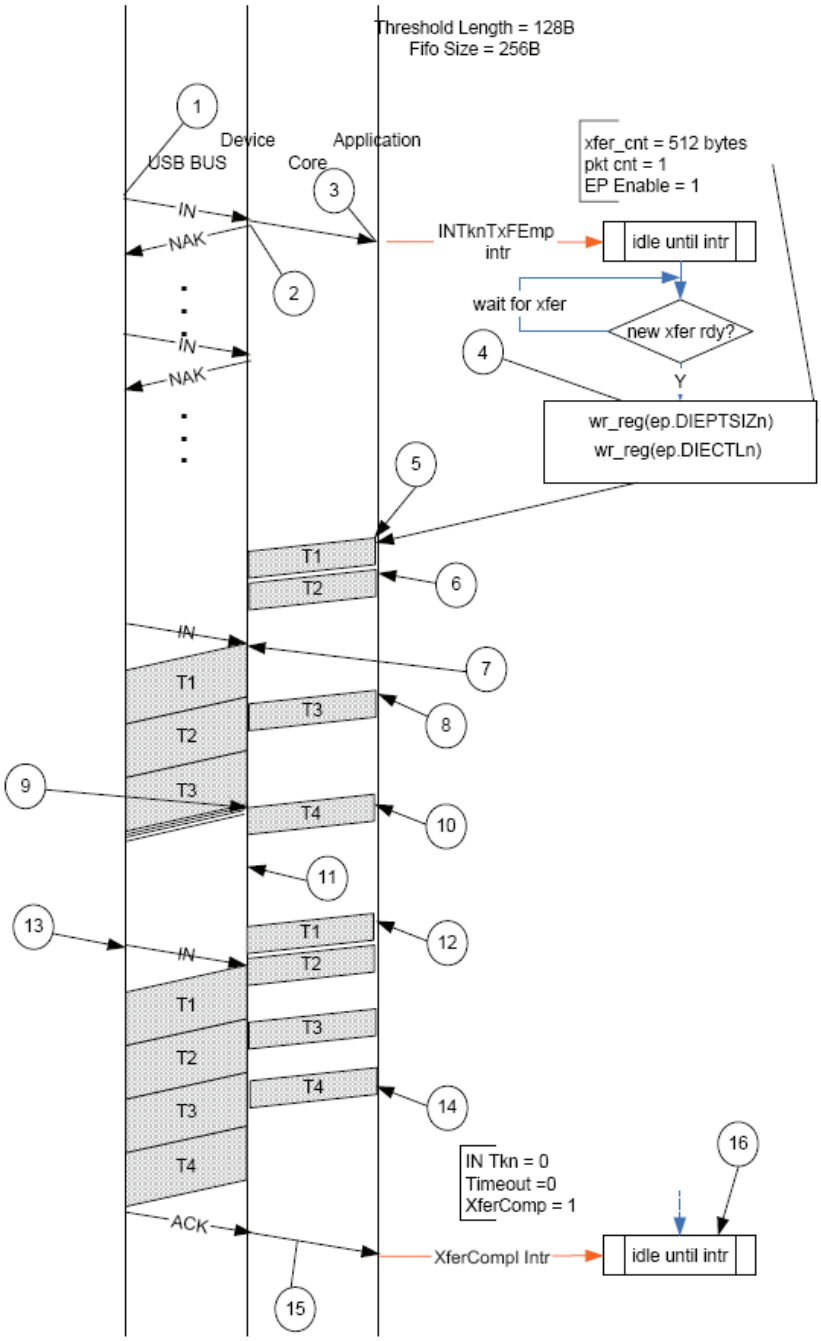


Figure 13.27 Bulk IN DMA mode with Thresholding

Isochronous IN DMA Mode with Thresholding

The FIFO size is assumed to be 512 Bytes and the threshold length is 128 Bytes.

1. Application enabled the isochronous IN endpoint for Odd microframe.

2. Core starts fetching the first 2 thresholds.

3. Core starts sending data out in response to the IN token.

4. Core sees an underrun condition, because the third threshold was not fetched in time.

5. Core generates DIEPINTn.TXFifoUnderrun interrupt (In this case, application ignores this interrupt).

6. At the End of Periodic Frame Interval, core generates GINTSTS.IncompISO interrupt.

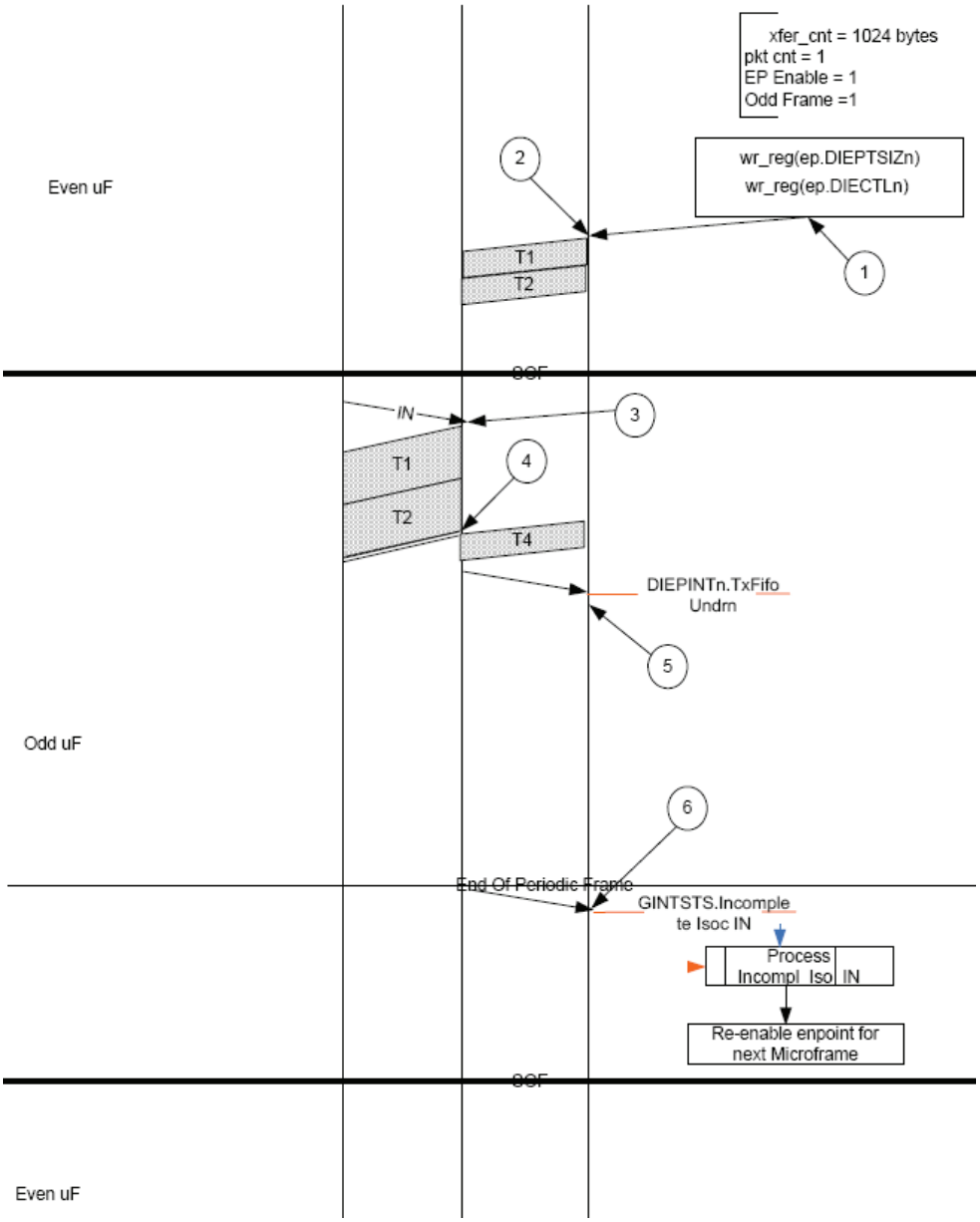


Figure 13.28 Isochronous IN DMA Mode with Thresholding

13.7.2.3 Control Transfers

This section describes the various types of control transfers.

[Control Write Transfers (SETUP, Data OUT, Status IN)]

This section describes control write transfers.

Application Programming Sequence

1. Assertion of the DOEPINTn.SETUP Packet interrupt indicates that a valid SETUP packet has been transferred to the application. See “SETUP Transactions” for more details. At the end of the Setup stage, the application must reprogram the DOEPTSIZn.SUPCnt field to 3 to receive the next SETUP packet.

2. If the last SETUP packet received before the assertion of the SETUP interrupt indicates a data OUT phase, program the core to perform a control OUT transfer as explained in “Generic Non-Isochronous OUT Data Transfers without Thresholding”. In DMA mode, the application must reprogram the DOEPWMAn register to receive a control OUT data packet to a different memory location.

3. In a single OUT data transfer on control endpoint 0, the application can receive up to 64 bytes. If the application is

expecting more than 64 bytes in the Data OUT stage, the application must re-enable the endpoint to receive another 64 bytes, and must continue to do so until it has received all the data in the Data stage.

4. Assertion of the DOEPINTn.Transfer Compl interrupt on the last data OUT transfer indicates the completion of the data OUT phase of the control transfer.

5. On completion of the data OUT phase, the application must do the following.

- To transfer a new SETUP packet in DMA mode, the application must re-enable the control OUT endpoint as explained in section in section “SETUP Transactions”.
  • DOEPCTLn.EPEna = 1’b1
- To execute the received Setup command, the application must program the required registers in the core. This step is optional, based on the type of Setup command received.

6. For the status IN phase, the application must program the core as described in “Generic Non-periodic IN Data Transfers without Thresholding” to perform a data IN transfer.

7. Assertion of the DIEPINTn.Transfer Compl interrupt indicates completion of the status IN phase of the control transfer.

8. The previous step must be repeated until the DIEPINTn.Transfer Compl interrupt is detected on the endpoint, marking the completion of the control write transfer.

[Control Read Transfers (SETUP, Data IN, Status OUT)]

This section describes control write transfers.

Application Programming Sequence

1. Assertion of the DOEPINTn.SETUP Packet interrupt indicates that a valid SETUP packet has been transferred to the application. See “SETUP Transactions” for more details. At the end of the Setup stage, the application must reprogram the DOEPTSIZn.SUPCnt field to 3 to receive the next SETUP packet.

2. If the last SETUP packet received before the assertion of the SETUP interrupt indicates a data IN phase, program the core to perform a control IN transfer as explained in “Generic Non-periodic IN Data Transfers without Thresholding”.

3. On a single IN data transfer on control endpoint 0, the application can transmit up to 64 bytes. To transmit more than 64 bytes in the Data IN stage, the application must re-enable the endpoint to transmit another 64 bytes, and must continue to do so, until it has transmitted all the data in the Data stage.

4. The previous step must be repeated until the DIEPINTn.Transfer Compl interrupt is detected for every IN transfer on the endpoint.

5. The DIEPINTn.Transfer Compl interrupt on the last IN data transfer marks the completion of the control transfer’s Data stage.

6. To perform a data OUT transfer in the status OUT phase, the application must program the core as described in “SETUP and OUT Data Transfers”.

- The application must program the DCFG.NZStsOUTHShk handshake field to a proper setting before transmitting an data OUT transfer for the Status stage.
- In DMA mode, the application must reprogram the DOEPWMAn register to receive the control OUT data packet to a different memory location.

7. Assertion of the DOEPINTn.Transfer Compl interrupt indicates completion of the status OUT phase of the control transfer. This marks the successful completion of the control read transfer.

- To transfer a new SETUP packet in DMA mode, the application must re-enable the control OUT endpoint as explained in “SETUP Transactions”.
  • DOEPCTLn.EPEna = 1’b1

[Two-Stage Control Transfers (SETUP/Status IN)]

This section describes two-stage control transfers.

Application Programming Sequence

1. Assertion of the DOEPINTn.SetUp interrupt indicates that a valid SETUP packet has been transferred to the application. See “SETUP Transactions” for more detail. To receive the next SETUP packet, the application must reprogram the DOEPTSIZn.SUPCnt field to 3 at the end of the Setup stage.

2. Decode the last SETUP packet received before the assertion of the SETUP interrupt. If the packet indicates a two-stage control command, the application must do the following.

- To transfer a new SETUP packet in DMA mode, the application must re-enable the control OUT endpoint. See “SETUP Transactions” for details.
  - DOEPCTLn.EPEna = 1’b1
- Depending on the type of Setup command received, the application can be required to program registers in the core to execute the received Setup command.

3. For the status IN phase, the application must program the core described in “Generic Non-periodic IN Data Transfers without Thresholding” to perform a data IN transfer.

4. Assertion of the DIEPINTn.Transfer Compl interrupt indicates the completion of the status IN phase of the control transfer.

5. The previous step must be repeated until the DIEPINTn.Transfer Compl interrupt is detected on the endpoint, marking the completion of the two-stage control transfer.

Example: Two-Stage Control Transfer

1. SETUP packet #1 is received on the USB and is written to the receive FIFO, and the core responds with an ACK handshake. This handshake is lost and the host detects a timeout.

2. The SETUP packet in the receive FIFO results in a GINTSTS.RxFLvl interrupt to the application, causing the application to empty the receive FIFO.

3. SETUP packet #2 on the USB is written to the receive FIFO, and the core responds with an ACK handshake.
4. The SETUP packet in the receive FIFO sends the application the GINTSTS.RxFLvl interrupt and the application empties the receive FIFO.

5. After the second SETUP packet, the host sends a control IN token for the status phase. The core issues a NAK response to this token, and writes a Setup Stage Done entry to the receive FIFO. This entry results in a GINTSTS.RxFLvl interrupt to the application, which empties the receive FIFO. After reading out the Setup Stage Done DWORD, the core asserts the DOEPINTn.SetUp packet interrupt to the application.

6. On this interrupt, the application processes SETUP Packet #2, decodes it to be a two-stage control command, and clears the control IN NAK bit.

- DIEPCTLn.CNAK = 1

7. When the application clears the IN NAK bit, the core interrupts the application with a DIEPINTn.INTknTXFEmp. On this interrupt, the application enables the control IN endpoint with a DIEPTSIZn.XferSize of 0 and a DIEPTSIZn.PktCnt of 1. This results in a zero-length data packet for the status IN token on the USB.

8. At the end of the status IN phase, the core interrupts the application with a DIEPINTn.XferCompl interrupt.

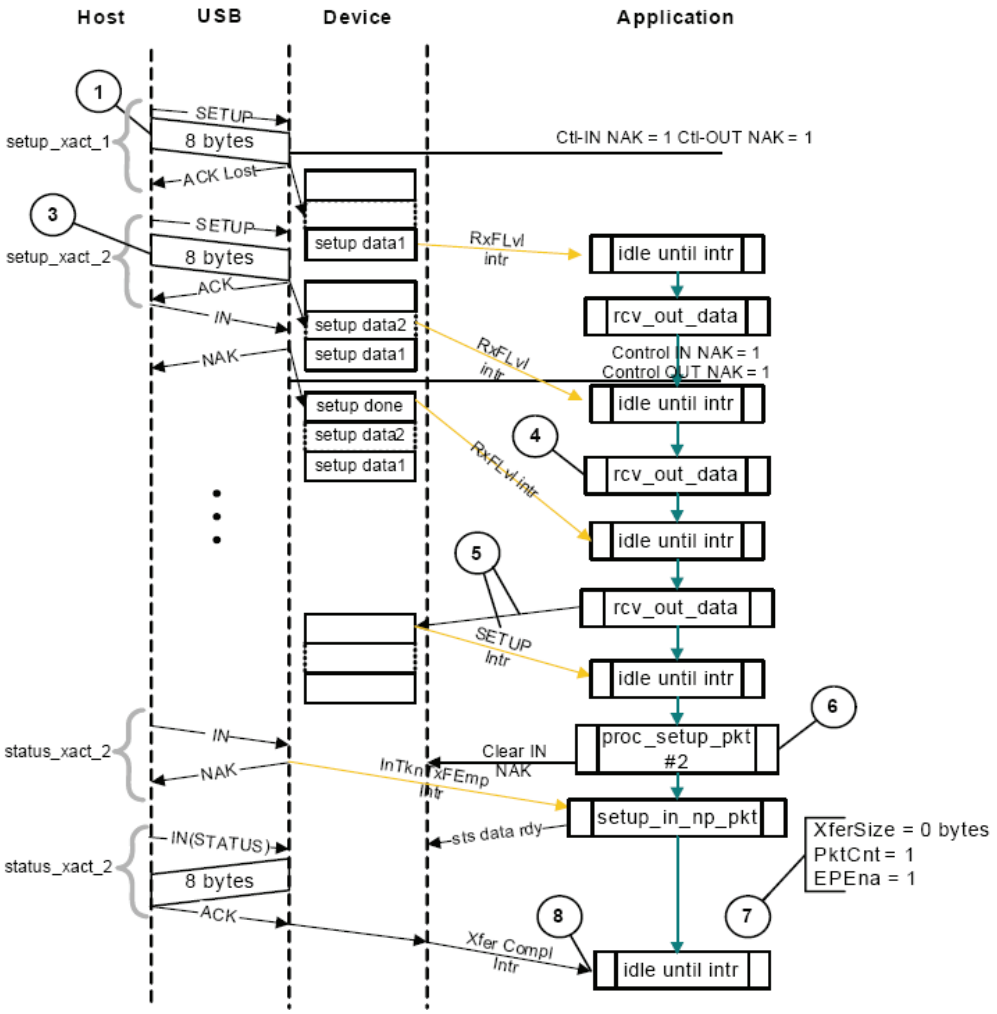


Figure 13.29 Two-Stage Control Transfer

13.7.3 Handling Babble Conditions

If the otg core receives a packet that is larger than the maximum packet size for that endpoint, the core stops writing data to the Rx buffer and waits for the end of packet (EOP). When the core detects the EOP, it flushes the packet in the Rx buffer and does not send any response to the host.

If the core continues to receive data at the EOF2 (the end of frame 2, which is very close to SOF), the core generates an early_suspend interrupt (GINTSTS.ErlySusp). On receiving this interrupt, the application must check the erratic_error status bit (DSTS.ErrticErr). If this bit is set, the application must take it as a long babble and perform a soft reset.

13.7.4 Worst Case Response Time

When the otg core acts as a device, there is a worst case response time for any tokens that follow an isochronous OUT. This worst case response time depends on the AHB clock frequency.

The core registers are in the AHB domain, and the core does not accept another token before updating these register values. The worst case is for any token following an isochronous OUT, because for an isochronous transaction, there is no handshake and the next token could come sooner. This worst case value is 7 PHY clocks when the AHB clock is the same as the PHY clock. When AHB clock is faster, this value is smaller.

If this worst case condition occurs, the core responds to bulk/interrupt tokens with a NAK and drops isochronous and SETUP tokens. The host interprets this as a timeout condition for SETUP and retries the SETUP packet. For isochronous transfers, the incomplISOCIN and incomplISOCOUT interrupts inform the application that isochronous IN/OUT packets were dropped.

13.7.5 Choosing the Value of GUSBCFG.USBTrdTim

The value in GUSBCFG.USBTrdTim is the time it takes for the MAC, in terms of PHY clocks after it has received an IN token, to get the FIFO status, and thus the first data from PFC (Packet FIFO Controller) block. This time involves the synchronization delay between the PHY and AHB clocks. The worst case delay for this is when the AHB clock is the same as the PHY clock. In this case, the delay is 5 clocks. If the PHY clock is running at 60 MHz and the AHB is running at 30 MHz, this value is 9 clocks.

Once the MAC receives an IN token, this information (token received) is synchronized to the AHB clock by the PFC (the PFC runs on the AHB clock). The PFC then reads the data from the SPRAM and writes it into the dual clock source buffer. The MAC then reads the data out of the source buffer (4 deep).

If the AHB is running at a higher frequency than the PHY, the application can use a smaller value for GUSBCFG.USBTrdTim. Next figure explains the 5-clock delay. This diagram has the following signals:

- tkn_rcvd: Token received information from MAC to PFC
- dynced_tkn_rcvd: Doubled sync tkn_rcvd, from pclk to hclk domain
- spr_read: Read to SPRAM
- spr_addr: Address to SPRAM
- spr_rdata: Read data from SPRAM
- srcbuf_push: Push to the source buffer
- srcbuf_rdata: Read data from the source buffer. Data seen by MAC

The application can use the following formula to calculate the value of GUSBCFG.USBTrdTim:

4 * AHB Clock + 1 PHY Clock = (2 clock sync + 1 clock memory address + 1 clock memory data from sync RAM) + (1 PHY Clock (next PHY clock MAC can sample the 2-clock FIFO output)

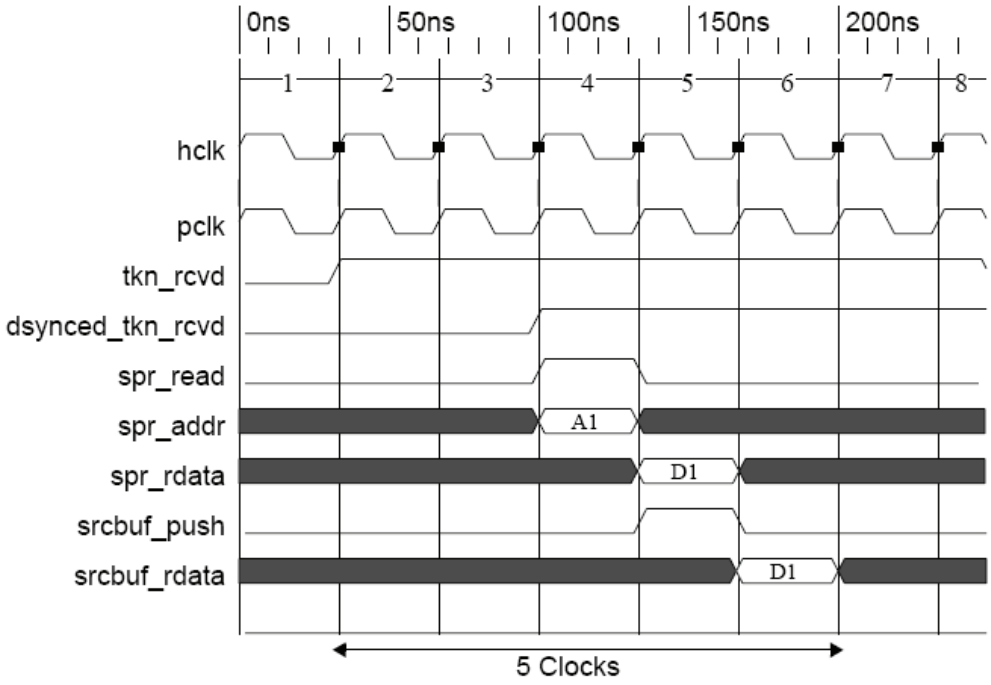


Figure 13.30 USBTrdTim Max Timing Case

13.8 Scatter-Gather DMA Mode

13.8.1 Overview

13.8.2 Scatter/Gather DMA Mode

When the Scatter/Gather DMA mode is enabled data buffers are presented through descriptor structures

1. The application prepares the descriptors, and sets the bit DIEPCTLn/DOEPCTLn.EPEna.

2. DMA fetches the corresponding descriptor (initially determined by DIEPDMAn/DOEPDMAn).

3. DMA internally sets the transfer size from descriptor back to DIEPTSIZn/DOEPTSIZn.

Note The registers DIEPTSIZn/DOEPTSIZn must not be written by the application in Scatter/Gather DMA mode

From this point, the current OTG flow executes. Once the transfer size data is moved by DMA, the DMA checks for further links in the descriptor chain. If this is the last descriptor, the DMA sets the DIOEPINTn.XferCompl interrupt. If there are further active links, the DMA continues to process them. In Scatter/Gather DMA mode, the core implements a true scatter-gather memory distribution in which data buffers are scattered over the system memory. Each endpoint memory structure is implemented as a contiguous list of descriptors, in which each descriptor points to a data buffer of predefined size. In addition to the buffer pointer (1 DWORD), the descriptor also has a status quadlet (1 DWORD). When the list is implemented as a ring buffer, the list processor switches to the first element of the list when it encounters last bit. All endpoints (control, bulk, interrupt, and isochronous) implement these structures in memory.

Note The descriptors are stored in continuos locations. For example descriptor 1 is stored in 32'h0000_0000, descriptor 2 is stored in 32'h0000_0008, descriptor 3 in 32'h0000_0010 and so on. The descriptors are always DWORD aligned.

13.8.3 SPRAM Requirements

For each endpoint the current descriptor pointer and descriptor status are cached to avoid additional requests to system memory. To save gates, these are stored in SPRAM. In addition DIEPDMAn/DOEPDMAn registers are implemented in SPRAM instead of flops.

13.8.3.1 Descriptor Memory Structures

The descriptor memory structures are displayed in Figure 1.31

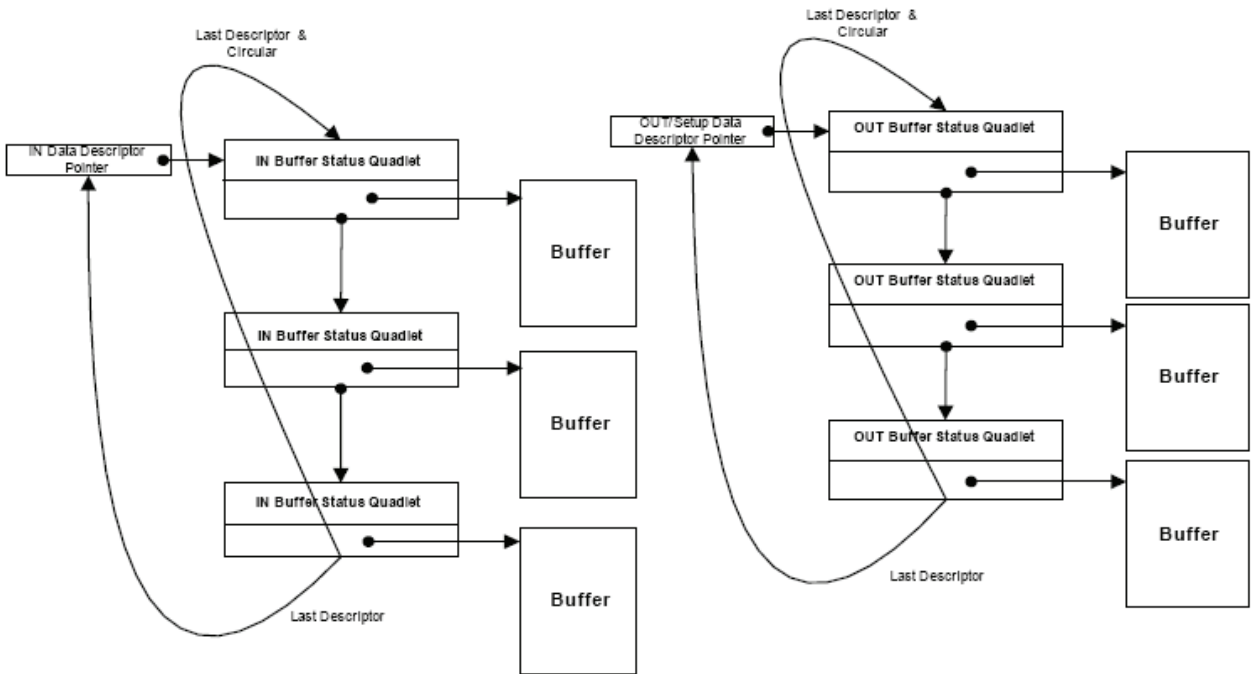


Figure 13.31 Descriptor Memory Structures

13.8.3.2 OUT Data Memory Structure

All endpoints that support OUT direction transactions (endpoints that receive data from the USB host), must implement a memory structure with the following characteristics:

- ❖ Each data buffer must have a descriptor associated with it to provide the status of the buffer. The buffer itself contains only raw data.
- ❖ Each buffer descriptor is two quadlets in length.

When the buffer status of the first descriptor is host Ready, the DMA fetches and processes its data buffer; otherwise the DMA optionally skips to the next descriptor until it reaches the end of the descriptor chain. The buffers to which the descriptor points hold packet data for non-isochronous endpoints and frame (FS)/μframe (HS) data for isochronous endpoints.

Host Ready -indicates that the descriptor is available for the DMA to process.

DMA Busy -indicates that the DMA is still processing the descriptor.

DMA Done -indicates that the buffer data transfer is complete.

Host Busy -indicates that the application is processing the descriptor.

The OUT data memory structure is shown in Figure 1.32, which shows the definition of status quadlet bits for non-ISO and ISO end points.

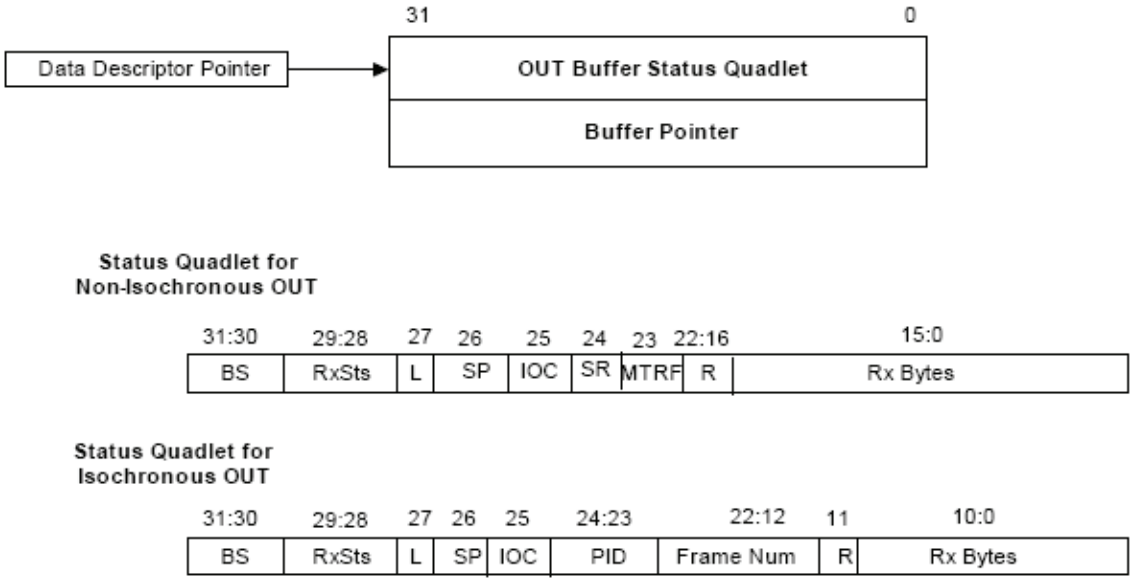


Figure 13.32 Out Data Memory Structure

The status quadlet interpretation depends on the end point type field (DOEPCTLn.EPType) for the corresponding end point. For example, if an end point is OUT and periodic, then the status quadlet is interpreted as Status Quadlet for Isochronous OUT. Table 6-3 displays the OUT Data Memory Structure fields.

Note Note that some fields change depending on the mode

Table 13.14 OUT Data Memory Structure Values

Bit	Name	Descriptiom.
BS[31:30]	Buffer Status	This 2-bit value describes data buffer status. Possible options are: • 2'b00: Host Ready • 2'b01: DMA Busy • 2'b10: DMA Done • 2'b11: Host Busy

<table>
<tr><td></td><td></td><td colspan="2">Application sets to Host Ready if the descriptor is ready or to Host Busy if the descriptor is not ready. Core sets to DMA busy if the descriptor is being serviced or to DMA Done if the transfer finished associated with the descriptor.

The application needs to make these bits as 2’b00 (Host Ready) as a last step after preparing the entire descriptor ready.Once the software makes these bits as Host Ready then it must not alter the descriptor until DMA completes</td></tr>
<tr><td>Rx Sts [29:28]</td><td>Receive Status</td><td colspan="2">This 2-bit value describes the status of the received data. Core updates this when the descriptor is closed. This reflects whether OUT data has been received correctly or with errors. BUFERR is set by the core when AHB error is encountered during buffer access. BUFERR is set by the core after asserting AHBErr for the corresponding end point.The possible combinations are:
• 2’b00: Success, No AHB errors
• 2’b01: Reserved
• 2’b10: Reserved
• 2’b11: BUFERR</td></tr>
<tr><td>L [27]</td><td>Last</td><td colspan="2">Set by the application, this bit indicates that this descriptor is the last one in the chain. Note - L Bit is interpreted by the core even when BS value is other than Host ready. For example, BNA is set, the core keeps traversing all the descriptors until it encounters a descriptor whose L bit is set after which the core disables the corresponding endpoint.</td></tr>
<tr><td>SP[26]</td><td>Short Packet</td><td colspan="2">Set by the Core, this bit indicates that this descriptor closed after short packet. When reset it indicates that the descriptor is closed after requested amount of data is received.</td></tr>
<tr><td>IOC[25]</td><td>Interrupt On complete</td><td colspan="2">Set by the application, this bit indicates that the core must generate a transfer complete interrupt(XferCompl) after this descriptor is finished.</td></tr>
<tr><td>[24]</td><td></td><td>Non Isochronous Out Bit: SR[24]
Bit ID: Setup Packet Received
Set by the Core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one setup packet per descriptor. On reception of a setup packet, the descriptor is closed and the corresponding endpoint is disabled after SETUP_COMPLETE status is seen in the Rx fifo. The core puts a SETUP_COMPLETE status into the Rx FIFO when it sees the first IN/OUT token after the SETUP packet for that particular endpoint.
However, if the L bit of the descriptor is set, the endpoint is disabled and the descriptor is closed irrespective of the SETUP_COMPLETE status. The application has to re-enable for receiving any OUT data for the control transfer. (It also need to reprogram the descriptor start address)

Note - Because of the above behavior, the core can receive any number of back to back setup packets and one descriptor for every setup packet is used.</td><td rowspan="2">Isochronous Out
Bit: PID [24:23]
Bit ID: ISO Received Data PID
Set by the Core. This field is for only high-speed isochronous transactions, and indicates the data PID for an isochronous receive packet.
• 2'b00: The packet contained in this descriptor is received with a data PID of DATA0
. • 2'b10: The packet contained in this descriptor is received with a data PID of DATA1.
• 2'b01: The packet contained in this descriptor is received with a data PID of DATA2.
• 2'b11: The packet contained in this descriptor is received with a data PID of MDATA.

For Full-Speed transactions, this field is reserved and the core writes 2’b00.</td></tr>
<tr><td>[23]</td><td></td><td>Non Isochronous Out
Bit: MTRF[23]
Bit ID: Multiple Transfer
Set by the application, this bit indicates the Core can continue processing the list after it encountered last descriptor.This is to support multiple transfers without application intervention. Reserved for ISO OUT and Control OUT endpoints.</td></tr>
</table>

[22:16]		Non Isochronous Out Bit: **[22:12]** Bit ID: **Reserved**	Isochronous Out Bit: **Frame Number [22:12]** Bit ID: **Frame number** **The 11-bit frame number in which the current ISO-OUT packet is received. For HS, the 11-bit is the concantenation of [7:0] of 1 ms frame number and [2:0] of uframe number, i.e {[7:0],[2:0]}.** **For FS, the 11-bit corresponds to full speed frame number.**
[15:12]		**Non Isochronous Out** **Bit:** Rx Bytes [15:0] **Bit ID:** Received number of bytes remaining This 16-bit value can take values from 0 to (64K-1) bytes, depending on the transfer size of data received from the USB host. The application programs the expected transfer size. When the descriptor is done this indicates remainder of the transfer size. Here, Rx Bytes must be in terms of multiple of MPS for the corresponding end point. The MPS for the various packet types are as follows: • Control • LS - 8 bytes - FS - 8,16,32,64 bytes - HS - 64 bytes • Bulk - FS - 8,16,32,64 bytes - HS - 512 bytes • Interrupt - LS - up to 8 bytes - FS - up to 64 bytes - HS - up to 1024 bytes **Note**: In case of Interrupt packets, the MPS may not be a multiple of 4. If the MPS in an interrupt packet is not a multiple of 4, then a single interrupt packet corresponds to a single descriptor. If MPS is a multiple of 4 for an interrupt packet, then a single descriptor can have multiple MPS packets.	
[11]			**Isochronous Out** **Bit: 11** **Bit ID**: Reserved
[10:0]			**Isochronous Out** **Bit**: Rx Bytes [10:0] **Bit ID**: Received number of bytes This 11-bit value can take values from 0 to (2K-1) bytes, depending on the packet size of data received from the USB host.Application programs the expected transfer size. When the descriptor is done this indicates remainder of the transfer size. The maximum payload size of each ISO packet as per USB specification 2.0 is as follows. • FS - up to 1023 bytes • HS - up to 1024 bytes **Note**: A Value of 0 indicates zero bytes of data, 1 indicates 1 byte of data and so on.

Table 13.15 displays the matrix of L bit and MTRF bit options.

Table 13.16 OUT - L Bit and MTRF Bit

L Bit	MTRF bit	Functionality
1	1	Continue to process the list after the last descriptor encountered. Use DOEPDMAn as next descriptor. The Endpoint is not disabled.
1	0	For non-Isochronous endpoints, Stop processing list after last descriptor encountered. The application intervenes and programs the list pointer into DOEPDMAn register when a list is created in a new location otherwise enables the endpoint. Start processing when the endpoint is enabled again with DOEPDMAn register pointing to start of list. For Isochronous endpoints, the DMA engine always goes back to the base descriptor address after the last descriptor.
0	1	After processing the current descriptor go to next descriptor. If a short packet OR zero length packet is received disable the endpoint and a transfer complete interrupt is generated irrespective of IOC bit setting for that descriptor.
0	0	After processing the current descriptor go to next descriptor. If a short packet OR zero length packet is received disable the endpoint and a transfer complete interrupt is generated irrespective of IOC bit setting for that descriptor.

Table 13.17 displays the out buffer pointer field description.

Note For Bulk and Interrupt End Points, if MTRF bit is set for the last descriptor in a list, then all the descriptors in that list

need to have their MTRF bit set.

Table 13.18 OUT Buffer Pointer

Bit	Name	Functionality
31:0	Buffer Address	The Buffer pointer field in the descriptor is 32 bits wide and contains the address where the received data is to be stored in the system memory. The starting buffer address must be DWORD aligned. The buffer size must be also DWORD aligned.

13.8.3.3 Isochronous OUT

- ❖ The application must create one descriptor per packet.
- ❖ End point is not disabled by the core based on L bit. The DMA always goes back to the base descriptor address after the last descriptor.
- ❖ The bit MTRF is not applicable.

13.8.3.4 Non-Isochronous OUT

- ❖ The core uses one descriptor per setup packet.
- ❖ The core closes the descriptor after receiving a short packet.
- ❖ Bit combinations for L and MTRF appear in Table 6-4.
- ❖ Multiple Interrupt packets in the same buffer is allowed only if the MPS is multiple of 4.

13.8.3.5 IN Data Memory Structure

All endpoints that support IN direction transactions (transmitting data to the USB host) must implement the following memory structure. Each buffer must have a descriptor associated with it. The application fills the data buffer, updates its status in the descriptor, and enables the endpoint. The DMA fetches this descriptor and processes it, moving on in this fashion until it reaches the end of the descriptor chain. The buffer to which the descriptor points to hold packet data for non-isochronous endpoints and frame (FS)/microframe (HS) data for isochronous endpoints. The definition of status quadlet bits for non-periodic and periodic end points are as shown in the figure 1.33. The status quadlet interpretation depends on the end point type field (DIEPCTLn.EPType) for the corresponding end point. For example, if an end point is IN and periodic, then the status quadlet is interpreted as “Status Quadlet for Isochronous IN”. The IN data memory structure is shown in Figure 1.33.

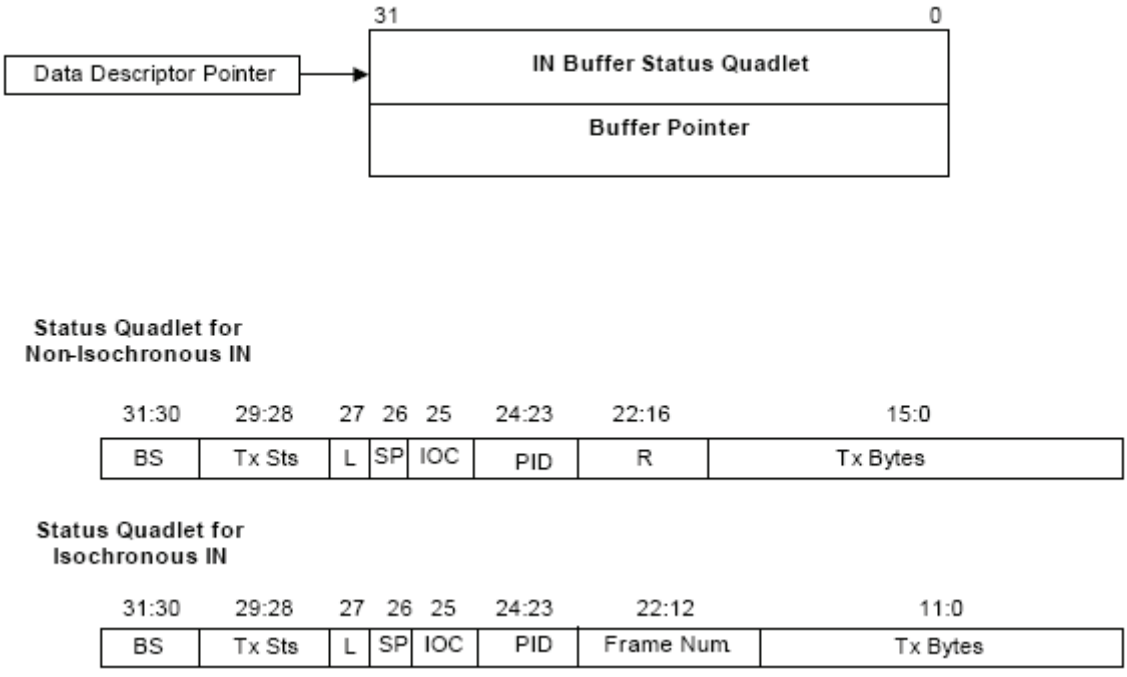


Figure 13.33 IN Data Memory Structure

Table 1.19 displays the IN Data Memory Structure fields.

Table 13.19 IN Data Memory Structure Values

Bit	Name	Descriptiom.	
BS[31:30]	Buffer Status	This 2-bit value describes data buffer status. Possible options are: • 2'b00: Host Ready • 2'b01: DMA Busy • 2'b10: DMA Done • 2'b11: Host Busy The application needs to make these bits as 2'b00 (Host Ready) as a last step after preparing the entire descriptor ready.Once the software makes these bits as HostReady then it must not alter the descriptor until DMA done	
Tx Sts [29:28]	Transmit Status	The status of the transmitted data. This reflects if the IN data has been transmitted correctly or with errors. BUFERR is set by core when there is a AHB error during buffer access. When iIgnrFrmNum is not set, BUFFLUSH is set by the core when • the core is fetching data pertaining to the current frame (N) and finds that the frame has incremented (N+1) during the data fetch • or • when it fetches a descriptor for which the frame number has already elapsed. The possible combinations are: • 2'b00: Success, No AHB errors • 2'b01: Reserved • 2'b10: Reserved • 2'b11: BUFERR Note: Also, irrespective of IgnrFrmNum value, in case of Isochronous IN endpoints, when threshold is enabled, if an underflow occurs when the corresponding buffer data is being accessed, the current descriptor is closed with a BUFFLUSH status.	
L [27]	Last	When set by the application, this bit indicates that this descriptor is the last one in the chain.	
SP[26]	Short Packet	When set, this bit indicates that this descriptor points to a short packet or a zero length packet. If there is more than one packet in the descriptor, it indicates that the last packet is a short packet or a zero length packet.	
IOC[25]	Interrupt On complete	When set by the application, this bit indicates that the core must generate a transfer complete interrupt after this descriptor is finished.	
[24]		**Non Isochronous Out Bit**: SR[24] **Bit** : Reserved[24:16] **Bit ID**: Reserved	**Isochronous Out** **Bit:** PID [24:23] **Bit ID**: Number of packets per frame This 2-bit value indicates the number of packets per µSOF (microframe) for isochronous IN transfers during high-speed operation. The application must program these bits in the descriptor (these bits must be the same for all descriptors of the same µSOF) such that the core returns an isochronous packet with an appropriate data PID per frame. • 2'b00: Reserved. The application should not program this value. • 2'b01: 1 packet per microframe, Data0 is the starting PID • 2'b10: 2 packets per microframe, Data1 is the starting PID • 2'b11: 3 packets per microframe, Data2 is the starting PID These bits are reserved for full-speed operation.
[24:23]		**Non Isochronous Out** **Bit**: MTRF[23] **Bit ID**: Multiple Transfer Set by the application, this bit indicates the Core can continue processing the list after it encountered last descriptor.This is to support multiple transfers without application intervention. Reserved for ISO OUT and Control OUT endpoints.	
[22:12]			

[15:12]		**Non Isochronous In** **Bit**: Tx bytes [15:0] **Bit ID**: Number of bytes to be transmitted This 16-bit value can take values from 0 to (64K-1) bytes, indicating the number of bytes of data to be transmitted to the USB host. **Note**: A Value of 0 indicates zero bytes of data, 1 indicates 1 byte of data and so on	**Isochronous In** **Bit:** Frame Number [22:12] **Bit ID**: Frame Number Frame number in which the current packet must be transmitted.The 11-bit is the concantenation of [7:0] of 1 ms frame number and [2:0] of uframe number i.e {[7:0],[2:0]}. For FS, this field must correspond to the 11-bit full speed frame number.
[11:0]			**Isochronous In Bit:** **Tx bytes [11:0]** **Bit ID**: Number of bytes to transmit Tx bytes [11:0] Number of bytes to be transmitted This 12-bit value can take values from 0 to (4K-1) bytes, indicating the number of bytes of data to be transmitted to the USB host. **Note**: A Value of 0 indicates zero bytes of data, 1 indicates 1 byte of data and so on.

Table 13.201 displays the matrix of L bit and MTRF bit options.

Table 13.21 IN - L Bit, SP Bit and MTRF Bit

L Bit	SP bit	Tx Bytes	Functionality
0	1	Multiple of endpoint maximum packet size	Transmit a zero length packet after the last packet
0	1	Not multiple of maximum packet size	Send short packet at the end after normal packets are sent out. Then move onto next descriptor
0	1	0	Transmit zero length packet. Then move on to next descriptor.
0	0	Multiple of endpoint maximum packet size	Send normal packets and then move to next descriptor.
0	0	Not a multiple of maximum packet size	Transmit the normal packets and concatenate the remaining bytes with next buffer from the next descriptor. This combination is valid only for bulk end points.
0	0	0	
1	1	Multiple of endpoint maximum packet size	Transmit a zero length packet after the last packet If this IN descriptor is for a ISO endpoint, then move onto the first descriptor in the list. If this IN descriptor is for a non-ISO endpoint, then stop processing this list and disable the corresponding end point.
1	1	Not multiple of maximum packet size	Send short packet after sending the normal packets If this IN descriptor is for a ISO endpoint, move onto the first descriptor in the list. If this IN descriptor is for a non-ISO endpoint, then stop processing this list and disable the corresponding end point.
1	1	0	Transmit zero length packet

			If this IN descriptor is for a ISO endpoint, move onto the first descriptor in the list. If this IN descriptor is for a non-ISO endpoint, then stop processing this list and disable the corresponding end point.
1	0	Multiple of endpoint maximum packet size	Send normal packets If this IN descriptor is for a ISO endpoint, Move onto the first descriptor in the list after current transfer done. If this IN descriptor is for a non-ISO endpoint, then stop processing the list and disable the corresponding end point.
1	0	Not multiple of maximum packet size.	Invalid. The behavior of the core is undefined for these values.
1	0	0	invalid. The behavior of the core is undefined for these values.

The descriptions provided for the different combinations in Table 6-7 depend on the previous descriptor L, SP, and Tx Bytes values. Consider table 1.22

Table 13.22 IN – Buffer Pointer

DESC NO	L bit	SP bit	Txbytes	Description
1	0	0	520	Send a normal packet of size 512, and concatenate the remaining 8 bytes with the next descriptor's buffer data
2	0	1	512	For this combination of L,SP and TxBytes, as per the above table, we need to send a zero length packet instead of a short packet. However, a normal packet followed by a short packet of length 8-bytes is sent. This is to illustrate the context dependency based on previous descriptor L,SP and TxByte combinations.

Table 1.23 displays the IN buffer pointer field description.

Table 13.23 IN Buffer Pointer

Bit	Bit ID	Description
31-0	Buffer Address	The Buffer pointer field in the descriptor is 32 bits wide and contains the address where the transmit data is stored in the system memory. The address can be non- DWORD aligned.

13.8.3.6 Descriptor Update Interrupt Enable Modes

If IOC bit is set for a descriptor and if the corresponding Transfer Completed Interrupt Mask (XferComplMask) is unmasked, this interrupt (DIOEPINTn.XferCompl) is asserted while closing that descriptor.

13.8.3.7 DMA Arbitration in Scatter/Gather DMA Mode

The arbiter grants receive higher priority than transmit. Within transmit, the priority is as follows.

- ❖ The highest priority is given to periodic endpoints. The periodic endpoints are serviced in a round robin fashion.
- ❖ The non periodic endpoints are serviced after the periodic scheduling interval has elapsed. The duration of the periodic scheduling interval is programmable, as specified by register bits DCFG[25:24]. When the periodic interval is active, the periodic endpoints are given priority.
- ❖ Amongst the periodic endpoints, the priority is round robin.
- ❖ Amongst the non periodic endpoints, the Global Multi Count field in the Device Control Register (DCTL) specifies the number of packets that need to be serviced for that end point before moving to the next endpoint.

The arbiter disables an endpoint and moves on to the next endpoint in the following scenarios as well, for all the endpoint types:

- ❖ Descriptor Fetch and AHB Error occurs.
- ❖ Buffer Not Available (BNA), such as when buffer status is Host busy.
- ❖ AHB Error during Descriptor update stage and Data transfer stage.

13.8.3.8 Buffer Data Access on AHB in Scatter/Gather DMA Mode

The buffer address whose data needs to be accessed in the system memory can be non DWORD aligned for transmit.
For buffer data read, the core arranges the buffer data to form a quadlet internally before populating the TXFIFO within the core as per the following scenarios

- ❖ The packet starts in a non DWORD aligned address, the core does two reads on AHB before appending the relevant bytes to form a quadlet internally. Hence the core stores the bytes before pushing to the TXFIFO.
- ❖ The packet ends in a non DWORD aligned address and it is not the end of the buffer or expected transfer, the core may switch to service another end point and come back to service the initial end point. In this case, the core reads the same DWORD location again and then samples only the relevant bytes. This eliminates the storage of the bytes for the initial end point.

For buffer data write, the core always performs DWORD accesses.

13.8.4 Control Transfer Handling

Control transfers (3-Stage Control R/WR or 2-Stage), can be handled effectively in the Descriptor-Based Scatter/Gather DMA mode by following the procedure explained in this section. By following this procedure the application is able to handle all normal control transfer flow and any of the following abnormal cases.

- ❖ More than one SETUP packet (back to back)□ Host could send any number of SETUP packets back to back, before sending any IN/OUT token. In this case, the application is suppose to take the last SETUP packet, and ignore the others.
- ❖ More OUT/IN tokens during data phase than what is specified in the wlength field□ If the host sends more OUT/IN data tokens than what is specified in the wlength field of the SETUP data, then the device must STALL.
- ❖ Premature SETUP packet during data/status phase□ Device application must be able to handle this SETUP packet and ignore the previous control transfer.
- ❖ Lost ACK for the last data packet of a Three-Stage Control Read Status Stage.

13.8.5 Interrupt Usage for Control Transfers

The application checks the following OUT interrupts status bits for the proper decoding of control transfers.

- ❖ DIEPINTn.XferCompl (Transfer complete, based on IOC bit in the descriptor)
- ❖ DIEPINTn.InTknTxfEmp (In token received when Tx FIFO is empty)
- ❖ DOEPINTn.XferCompl (Transfer complete, based on IOC bit in the descriptor)
- ❖ DOEPINTn.SetUp (Setup Complete interrupt, generated when the core receives IN/OUT token after a SETUP packet.
- ❖ DOEPINTn.StsPhseRcvd (Status phase received interrupt (Also called SI), generated when host has switched to status phase of a Control Write transfer).

The core performs some optimization of these interrupt settings, when it sees multiple interrupt bits need to be set for OUT endpoints. This reduces the number of valid combinations of interrupts and simplifies the application.

The core gives priority for DOEPINTn.XferCompl over DOEPINTn.SetUp and DOEPINTn.StsPhseRcvd (SI) interrupts. When setting the XferCompl interrupts, it clears the SetUP and SI interrupt bits.

- ❖ The core gives priority to DOEPINTn.SI interrupt over DOEPINTn.SetUp. When setting DOEPINTn.StsPhseRcvd (SI), the core clears DOEPINTn.SetUp interrupt bit.

Based on this, the application needs only to decode the combinations of interrupts for OUT endpoints shown in Table 1.24

Table 13.24 IN – Buffer Pointer

StsPhseRcvd (SI)	SetUp (SPD)	XferComl (IOC)	Description	Template Used
0	0	1	Core has updated the OUT descriptor. Check the “SR”(Setup Received) bit in the descriptor to see if the data is for a SETUP or OUT transaction.	Case A
0	1	0	Setup Phase Done Interrupt for the previously decoded SETUP packet.	Case B
0	1	1	The core has updated the OUT descriptor for a SETUP packet, and the core is indicating a SETUP complete status also.	Case C
1	0	0	Host has switched to Status phase of a Control OUT transfer	Case D
1	0	1	Core has updated the OUT descriptor. Check the SR”(Setup Received) bit in the descriptor to see if the data is for a SETUP or OUT transaction. Also, the host has already switched to Control Write Status phase.	Case E

13.8.6 Application Programming Sequence

This section describes the application programming sequence to take care of normal and abnormal Control transfer scenarios.
All the control transfer cases can be handled by five separate descriptor lists. The descriptor lists are shown in Figure 1.34

- ❖ Three lists are for SETUP. The SETUP descriptors also take data for the Status stage of Control Read.
- ❖ The first two (index 0 and 1) act in a ping-pong fashion.
- ❖ The third list is an empty list, linked to one of the OUT descriptors when premature SETUP comes during the data/status phase.
- ❖ Two lists are for IN and OUT data respectively.
- ❖ Figure 6-39 on page 445 displays setup_index 0, 1, and 2 as elements of array of pointers called setup_index. The first two elements of this array point to SETUP descriptors. The third element of this array is initially a NULL pointer, but is eventually linked to a SETUP descriptor. These array elements could also point to a descriptor for Control Read Status phase.

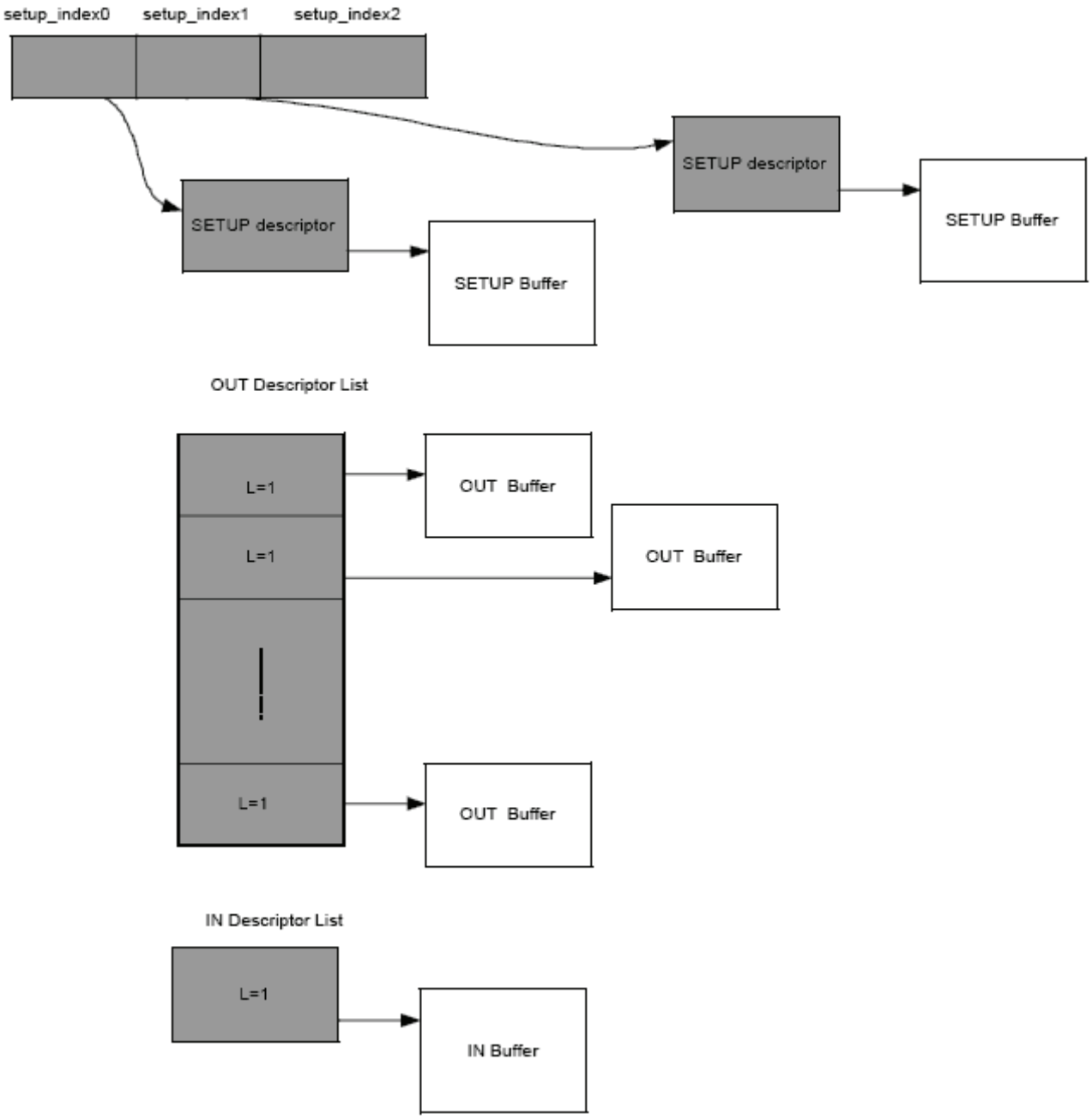


Figure 13.34 Descriptor Lists for Handling Control Transfers

The following are the steps that need to be followed by the application driver.

1. **Set up Desc for SETUP/Ctrl-Rd-Sts -**Setup2 descriptor lists in memory for taking in SETUP packets. Each of this list must have only one descriptor, with the descriptor fields set to the following

   - Rx_bytes Set it to Max packet size of the control endpoint.
   - IOC =1.
   - MTRF=0.
   - L=1.

2. **Enable DMA** If current setup_index =0, then setup_index=1. The application ping-pongs between these two descriptors. Program the address of the current setup descriptor (specified by setup_index) to DOEPDMAn. Write to DOEPCTLn with the following fields.

   - DOEPCTL.MPS Max Packet size of the endpoint
   - DOEPCTL.EPEna Set to 1 to enable the DMA for the endpoint.

3. **Wait for Interrupt-**Wait for OUT endpoint interrupt (GINTSTS.OEPInt). Then read the corresponding DOEPINT.

4. If Control Read Data Stage in progress

- Case A: Check SR bit (In this case SR bit is set, because the host cannot send OUT at this point. If it sends OUT it is NAKed. **GOTO** Step 24.
- Case B: **GOTO** Step 26.
- Case C:-Check SR bit (In this case SR bit is set because host cannot send OUT packets without SETUP at this stage). **GOTO** Step 24.
- Case D: Cannot happen at this stage because SI cannot come alone without a SETUP, at this stage.
- Case E: Indicates that host has switched to another SETUP (Three-Stage control write) and then has switched to status phase without and data phase (core clears SUP with SI in this case). Decode SETUP packet and if ok, **GOTO** Step 11. else If Ctrl Write Status Stage in progress OR Two-Stage Status Stage in progress
- Case A: Check SR bit (In this case SR bit is set, because the host cannot send OUT at this point. If it sends OUT it is NAKed.) **GOTO** Step 24.
- Case B: (Could happen for Two-Stage Ctrl Transfer.) **GOTO** Step 26.
- Case C: **GOTO** Step 24.
- Case D: Clear SI interrupt and wait Step 3.
- Case E: Cannot happen at this stage. *else*
- Case A: **GOTO** Check Desc.
- Case B: Normally, this does not occur at this stage. Either IOC comes first or IOC comes with SUP (Case C).
- Case C: **GOTO** Check Desc.
- Case D: Cannot happen at this point.
- Case E: If SR==1, Indicates Three stage control Transfer SETUP and that the host has switched to status phase. Decode the SETUP packet and **Goto** Step 11.
- Check Desc Read the Descriptor status quadlet corresponding to the setup_index and check the SR field. (Application might also want to check the BS and RxSts fields and take necessary actions if there is any abnormalities). If SR field is 1 **GOTO** Step 5 (If Step Step 20 is active, terminate it). If SR field is 0 **GOTO** Step 22 (Control Rd Status phase) (This must also terminate Step 20).

5. **Decode SETUP-**Decode the SETUP packet. If it is a Three-Stage Control Write, **GOTO** Step 20. If it is a Three-Stage Control Read, **GOTO** Step 15. If it is a Two-Stage Control transfer, **GOTO** Step 11 (Same as Status stage for 3-Stage Control Write).

6. **Desc list for Ctrl Wr data—**Setup descriptor list for Control write data phase. This must be based on the Wlength field in the SETUP data. The descriptors in the list must be setup such that there must be one descriptor per packet. Each of these descriptors must have the control fields set as follows.

- Rx_Bytes: Set to the Max Packet Size of the control Endpoint.
- IOC = 1
- MTRF = 0.
- L=1.
- At this point we are not enabling and clearing the NAK for the IN endpoint for status phase.This is because, the status phase for Control Write can be ACKed only after decoding the complete data for the data phase. **GOTO** Step 7.

7. **Enable DMA for Ctrl Wr Data:** Write the start address of this list to DOEPDMAn. Program the DOEPCTLn with the following bits set

- DOEPCTL.MPS: Max Packet size of the endpoint
- DOEPCTL.EPEna: Set to 1 to enable the DMA for the endpoint. **GOTO** Step 8.

8. **Wait for Ctrl Wr Data Interrupt:** Wait for OUT endpoint interrupt (GINTSTS.OEPInt). Then read the corresponding DOEPINTn.

- Case A: check the SR field. Also clear DOEPINTn.XferCompl by writing to DOEPINTn.(Application might also want to check the BS and RxSts fields and take necessary actions if there is any abnormalities). If SR field is 0 **GOTO** Step 9.If SR field is 1, **GOTO** Step 23**.** (This indicates that the host has switched to a new control transfer).
- Case B: **GOTO** Step 25.
- Case C: **GOTO** Step 23**.** (This indicates that the host has switched to a new control transfer).

- ✦ Case D: Host has switched to status phase. Decode the data received so far. **GOTO** Step 10.
- ✦ Case E: Check SR bit. If SR==0, decode the data received so far. GOTO Step 10. If SR==1, decode the SETUP packet and Goto Step10.

9. **Check Desc-**If it's not the last packet of data phase, Re-enable the endpoint and clear the Nak. This is because the core sets NAK after receiving each OUT packet for control write data phase. This is to allow application to STALL in case the host sends more data than what is specified in the Wlength field. **GOTO** Step 8. Re-enabling and clearing the NK involves the following steps.

- ✦ Write to DOEPDMA with the new descriptor address.
- ✦ Write to DOEPCTLn with the following fields.
  - ✧ DOEPCTL.MPS: Max Packet size of the endpoint
  - ✧ DOEPCTL.CNAK: Set to 1 to clear the NAK.
  - ✧ DOEPCTL.EPEna: Set to 1 to enable the DMA for the endpoint. If it is the last packet of the data phase, **GOTO** Step 10.

10. **STALL Extra Bytes—**Write to DOEPCTLn with Stall set so that the core could STALL any further OUT tokens from host.If the received Bytes so far is greater than what is specified in Wlength field OR is there were any unsupported commands in the data phase, then write to DIEPCTLn with the Stall bit set so that the Status phase could be Stalled.(The STALL bit is automatically cleared by the core with the next SETUP). **GOTO** Step 11.

11. **Disc list for Ctrl Wr Sts—**The following two process must run in parallel. This is because, we are preparing for the status phase (IN) of Control write but at the same time the host could send another SETUP. So IN and OUT descriptor list must be ready. a. Do Step 2— Step 5 (This is for handling SETUP or Ctrl Wr Status). If the OUT DMA is already enabled (OUT DMA was enabled for data phase of Three-Stage Control Write, but there was a premature status phase), GOTO Step 3. b. Setup descriptor list for Status phase IN, depending on the data in the status phase. Normally it is always a zero length packet. c. Tx_Bytes—Size of status phase, d. BS—Host Ready, e. L=1. f. IOC=1. g. SP=1 (Depending on the Tx_Bytes). h. Write to DIEPDMAn with the start address of the descriptor.Write to DIEPCTLn clear the NAK and enable the endpoint. Flush the corresponding TX FIFO. i. If SI has not been received in the data stage prior to the status stage, then wait for SI before clearing the NAK(DIEPCTLn.CNAK=1) j. DIEPCTLn.EpEna=1. k. **GOTO** Step 12.

12. **Wait for Interrupt—**Wait for IN endpoint interrupt (GINTSTS.IEPInt).

13. If IN endpoint INterrupt, and DIEPINTn.XferCompl, then **GOTO** Step 14.

14. **Check Desc—**Read the Status field of the descriptor. Check Tx_bytes in the descriptor.(Application might also want to check the BS and RxSts fields and take necessary actions if there is any abnormalities). This is end of Three-Stage Control Write OR Two-Stage Control transfer. We are now ready for the next control transfer (Already taken care by process "a" is Step 11**.**

15. **Desc for Ctrl Rd Data—**The following two steps must be run in parallel. This is because, we are preparing for Data phase of Control read, but at the same time, the host could abnormally abort this control transfer and send a SETUP, or switch to status phase.

16. Do Step 2— Step 5 (This is for handling SETUP and also Control Read Status phase.).

17. Setup descriptor list for Data phase IN, depending on the WLength field in the SETUP data. You could setup single descriptor OR multiple descriptors. If it is multiple descriptors, ensure that IOC for the last descriptor is set.

- ✧ Tx_Bytes: Size of data phase (Wlength field).
- ✧ BS: Host Ready ✧ L=1. ✧ IOC=1. It is mandatory to set the IOC when it is the last descriptor.
- ✧ SP=1 (Depending on the Tx_Bytes). ✧ Write to DIEPDMAn with the start address of the descriptor list.
- ✧ Write to DIEPCTLn clear the NAK and enable the endpoint.
- ✧ Flush the corresponding TX FIFO.
- ✧ DIEPCTLn.MPS = Max_packet size of the endpoint,
- ✧ DIEPCTLn.CNAK=1 only if SPD already set (Case C in Step 3).
- ✧ Also set the DOEPCTLn.CNAK for the corresponding OUT endpoint after SPD because a premature status stage (OUT) can come which must be acked.
- ✧ DIEPCTLn.EpEna=1.
- ✧ **GOTO** Step 18.

18. **Wait for Interrupt:** Wait for IN endpoint interrupt (GINTSTS.IEPInt)

19. If IN endpoint interrupt, read the corresponding DIEPINTn and if XferCompl is set **GOTO** Step 20.

20. **Check_Desc**—Wait for the DIEPINTn.IOC interrupt. Go to Step 21.

21. **Set_Stall—**Write to DIEPCTLn with STALL bit set. (The STALL bit is automatically cleared by the core with the next SETUP). The function of this process initiated in step Step 15 is over, and must be terminated. The next control transfer is already taken care by the process that is running from Step 2.

22. **Ctrl Rd Sts Desc Check—**Read the descriptor to check the Rxbytes and also check the SP field. The Three-Stage control Read is complete here. **GOTO** Step 2, in preparation for the next SETUP.

23. The unexpected SETUP packet now received during the control write data phase, is sitting in the descriptor allocated for Data. Link this to the setup descriptor pointer. setup_desc_index = 2. Point setup_desc_index to the current OUT descriptor (which has the SETUP). **GOTO** Step 5.

24. Disable IN Endpoint DMA. Core flushes the corresponding Tx FIFO in order to flush the data that was meant for Control Write Status phase OR Control Read data phase. If Step 12 or Step 18 is active, terminate it. **GOTO** Step .

25. Read Modify write DOEPCTLn to clear the NAK. Then **GOTO** Step 8 again.

- ✦ DOEPCTLn.CNAK Set to 1 to clear the NAK.

26. Read Modify write DIEPCTLn to clear the NAK. Then **GOTO** Step 3 again.

- ✦ DIEPCTLn.CNAK Set to 1 to clear the NAK.

27. Read Modify write DIEPCTLn to clear the NAK. Then Step 12 again

- ✦ DIEPCTLn.CNAK Set to 1 to clear the NAK.
- ✦ DOEPCTLn.CNAK:Set to 1 to clear the NAK for the out endpoint. This clears the NAK to accept status stage data in case of control read.

13.8.7 Internal Data Flow

This section explains the cores internal data flow for control transfers.

13.8.7.1 Three-Stage Control Write

Figure 6-40 on page 451 displays the core behavior for Three-Stage control write transfers.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint. Additionally, the clearing of the NAK bit is blocked by the core until the following SPD or SI is read by the application and cleared.

2. The DMA detects the RxFIFO as non-empty and does the following:

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Application clears NAK for the data phase, after receiving DOEPINTn.SetUp interrupt.

7. The core ACKs and the next OUT token because the NAK has been cleared (provided there is enough space in the RxFIFO.

8. DMA detects the OUT packet in RxFIFO and starts transferring the OUT packet to the system memory.

   - ✦ Fetch the descriptor pointed by DOEPMA.

- ✦ Transfer the OUT packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close descriptor with DMA_DONE status.

9. The core NAKs the next OUT token because the core internally sets the NAK after every control write data phase packets. This is to allow application to Stall any extra tokens.

10. The core generates DOEPINT.XferCompl after closing the OUT descriptor (Step 8).

11. Application clear NAK on receiving DOEPINTn.XferCompl interrupt.

12. Host starts the Status phase by sending the IN token which is NAKed by the core. The core push DATA_PHASE_DONE status into the RxFIFO.

13. The core generates DOEPINTn.XferCompl for the last OUT packet transfer to system memory.

14. The core generates DOEPINT.StsPhsRcvd interrupt after the DMA has popped the DATA_PHASE_DONE status from the RxFIFO.

15. Application clears the NAK and enables the IN endpoint for status phase.

16. The core starts fetching the data for the Status phase

- ✦ Fetch the descriptor pointed by DIEPDMA.
- ✦ Fetch the packet (if size >0) to Tx fifo.
- ✦ Close the descriptor with DMA_DONE status
- ✦ The core generates DIEPINTn.XferCompl interrupt after closing the descriptor.

17. The core sends out data in response to the Status Phase IN token.

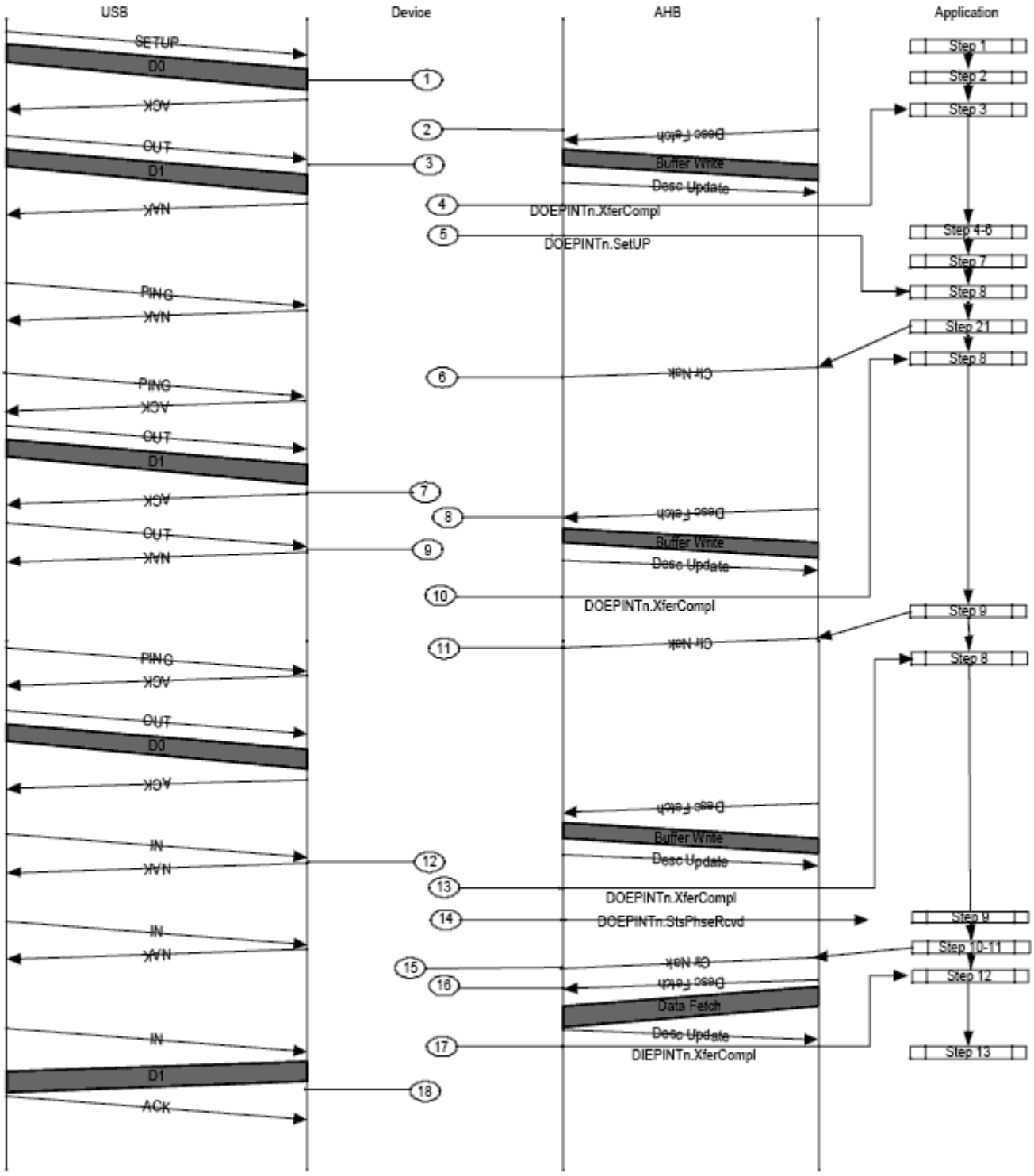


Figure 13.35 Three-Stage Control Write

In this example, it is assumed that the data phase consists of 2 packets, and the application allocates these two packets in a single buffer.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   ✦ Fetch the descriptor pointed by DOEPMA.

- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Data phase IN tokens are NAKed until this point because the NAK has not yet been cleared by the application.

7. The core starts fetching the IN data after the application enables IN DMA (In this example it is assumed that multiple packets are in the same buffer. But it could also be in different buffers). This involves the following steps

- ✦ Fetch the descriptor pointed by DIEPDMA.
- ✦ Fetch the data into the corresponding Tx FIFO.
- ✦ Close the descriptor with DMA_DONE status...s

8. The application clears the NAK after receiving the setup complete (DOEPINT.SetUp) interrupt. The application also clears NAK of the OUT End point to accept the status phase.

9. After all the data has been fetched for the descriptor (Step 7), core generates DIEPINT.XferCompl interrupt.

10. The core sends data in response to the IN token for the data phase.

11. The core sends out the last packet of the IN data phase.

12. The core ACKs the status phase.

13. The core generates DOEPINTn.XferCompl interrupt after transferring the data received for the status phase to system memory.

13.8.7.2 Two-Stage Control Transfer

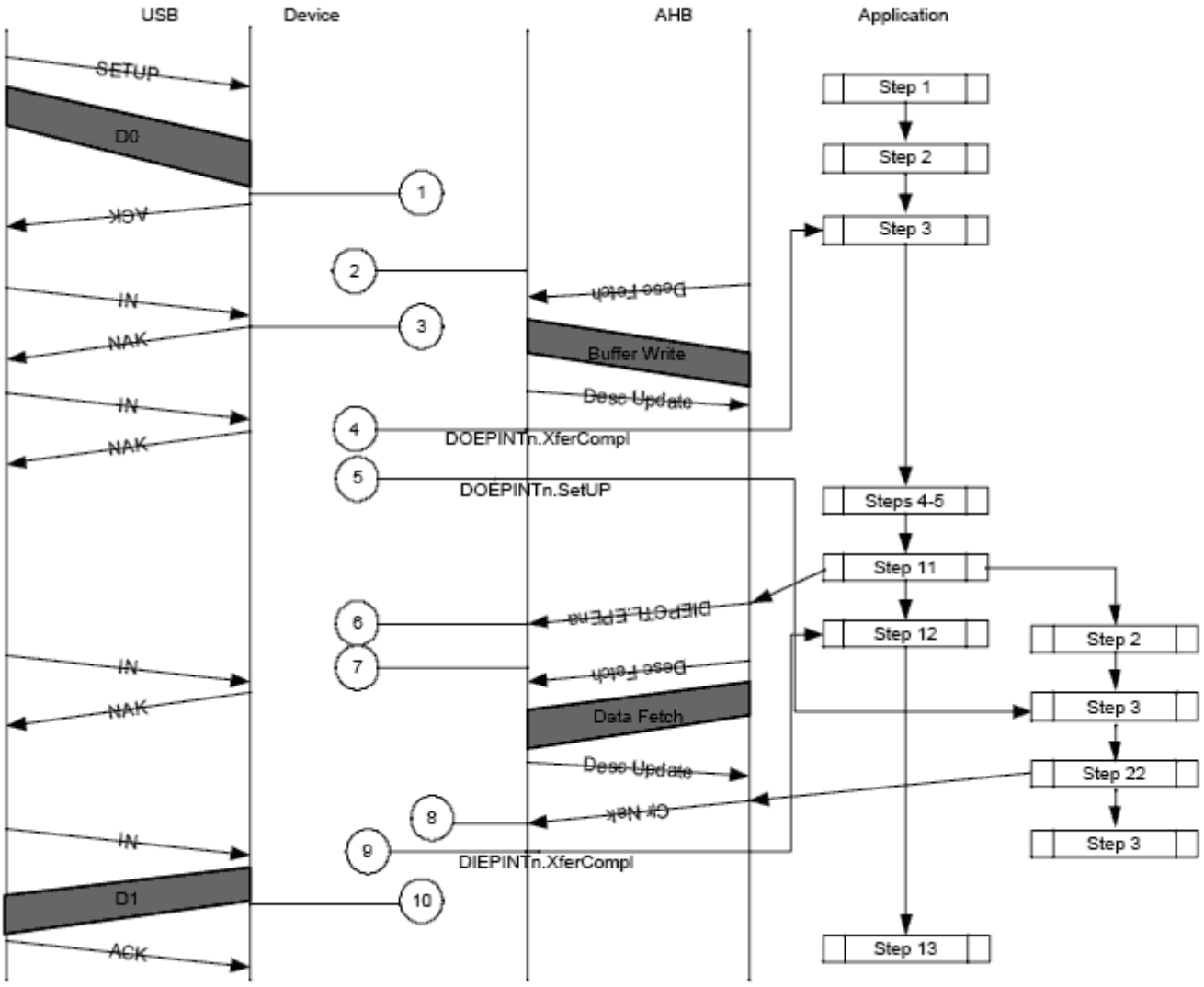


Figure 13.36 Two-Stage Control Write

This example shows the core behavior for a Two-Stage Control transfer.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.
   - ✦ The core receives the status phase IN token, which it NAKs. Core also pushes SETUP_COMPLETE status into the Rx FIFO.
   - ✦ The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2)
   - ✦ The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

3. Application enables the IN endpoint for status phase.

4. The core starts fetching the descriptor for IN endpoint.

5. Application clears the NAK for IN endpoint after getting the DOEPINTn.SetUP interrupt (Step 4).

6. The core generates DIEPINTn.XferCompl after updating the descriptor after IN data fetch.

7. The core sends out data for the status phase IN token from host.

13.8.7.3 Back to Back SETUP During Control Write

This example shows the core receiving 2 Back to Back SETUP tokens fo3 Three-Stage Control write.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

- ✦ Fetch the descriptor pointed by DOEPMA.
- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.
- ✦ The core receives another SETUP, and pushes the data into the Rx FIFO, also sets the NAK.
- ✦ The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The DMA detects the RxFIFO as non-empty (because of the 2nd SETUP packet) and does the following.

- ✦ Fetch the descriptor pointed by DOEPMA.
- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.
- ✦ The core generates DOEPINT.XferCompl interrupt after having transferred the second SETUP packet into memory (Step 6)
- ✦ The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

5. Application clears NAK for the data phase, after receiving DOEPINTn.SetUp interrupt.

6. The core ACKs the next OUT/Ping token after the NAK has been cleared by the application.

7. The DMA detects the RxFIFO as non-empty (because of the OUT packet) and does the following.

- ✦ Fetch the descriptor pointed by DOEPMA.
- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.
- ✦ The core generates DOEPINT.XferCompl interrupt after having transferred the OUT packet into memory (Step 11) and closing the descriptor.

The remaining steps are similar to Steps 11-18 of “Application Programming Sequence”. This example shows the core behavior for a Two-Stage Control transfer

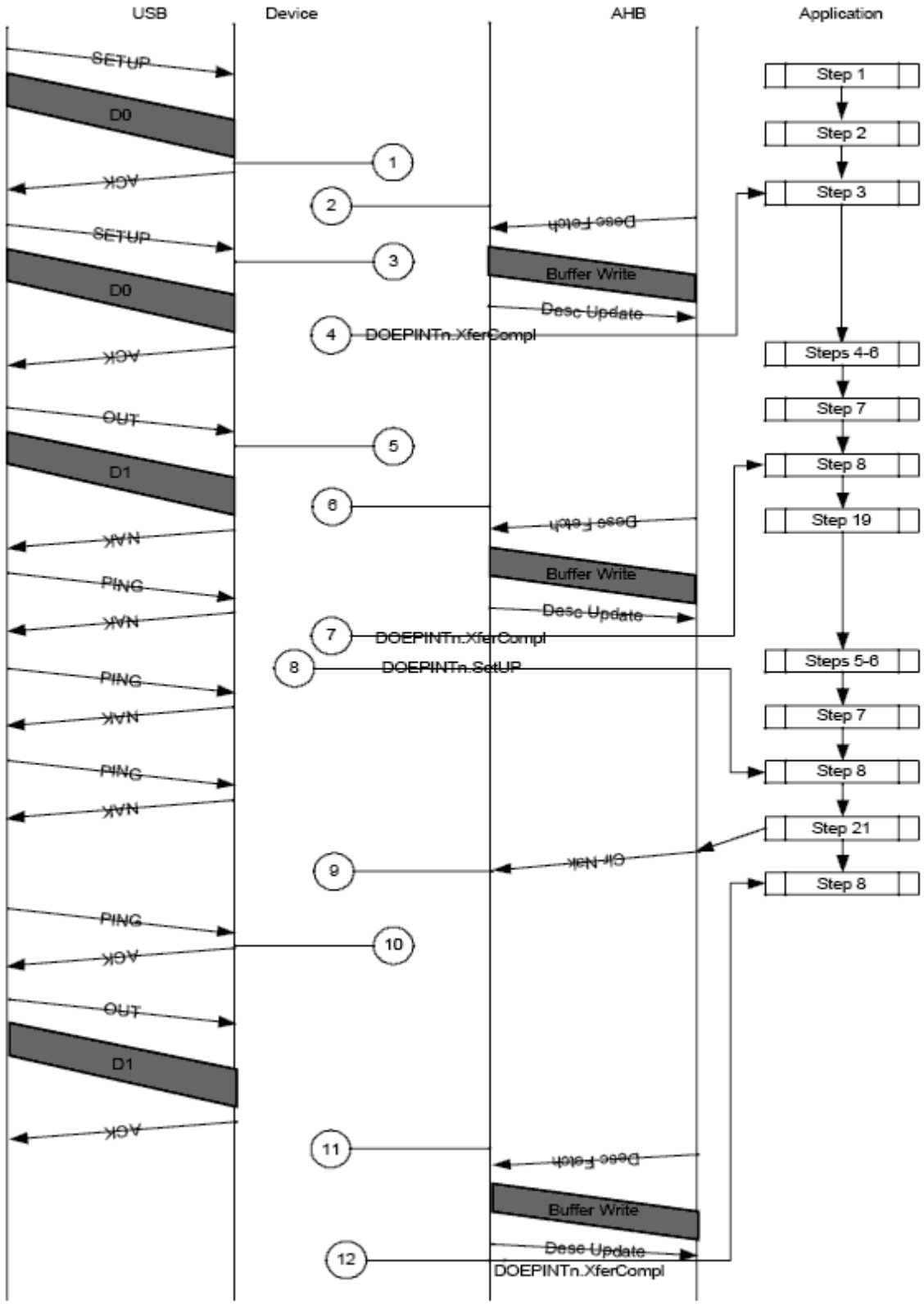


Figure 13.37 Back-to-Back SETUP Packet Handling During Control Write

13.8.7.4 Back-to-Back SETUPs During Control Read

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

- ✦ Fetch the descriptor pointed by DOEPMA.
- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.
- ✦ The core receives another SETUP, and pushes the data into the Rx FIFO, also sets the NAK.

3. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

4. Host sends IN token for the data phase which is NAKed by the core, because NAK is set in Setp3. The core pushes SETUP_COMPLETE status into RxFIFO.

5. After the application has re-enabled the OUT DMA (Application flow Step 2) core detects RxFIFO as non-empty because of the second SETUP packet and does the following.

- ✦ Fetch the descriptor pointed by DOEPMA.
- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.
- ✦ The core generates DOEPINTn.XferCompl interrupt after having transferred the SETUP packet into memory (Step6).
- ✦ The core starts fetching data for IN endpoint because the IN endpoint was enabled by application in Step-14.

6. On seeing DOEPINTn.XferCompl (Step 7) and finding that it is a SETUP packet, application disables the endpoint in Step 20.

7. The core generates DOEPINTn.SetUP (Setup complete) interrupt after popping the SETUP_COMPLETE status from the RxFIFO.\

8. The core generates endpoint disabled interrupt (as a result of application setting disable bit in step 9).

9. The core generates DIEPINTn.XferCompl after completing the IN data fetch and updating the descriptor.

10. application clears NAK after seeing setup_complete interrupt (generated in Step 10.

The flow after this is same as steps 9 - 13 of “Internal Data Flow”

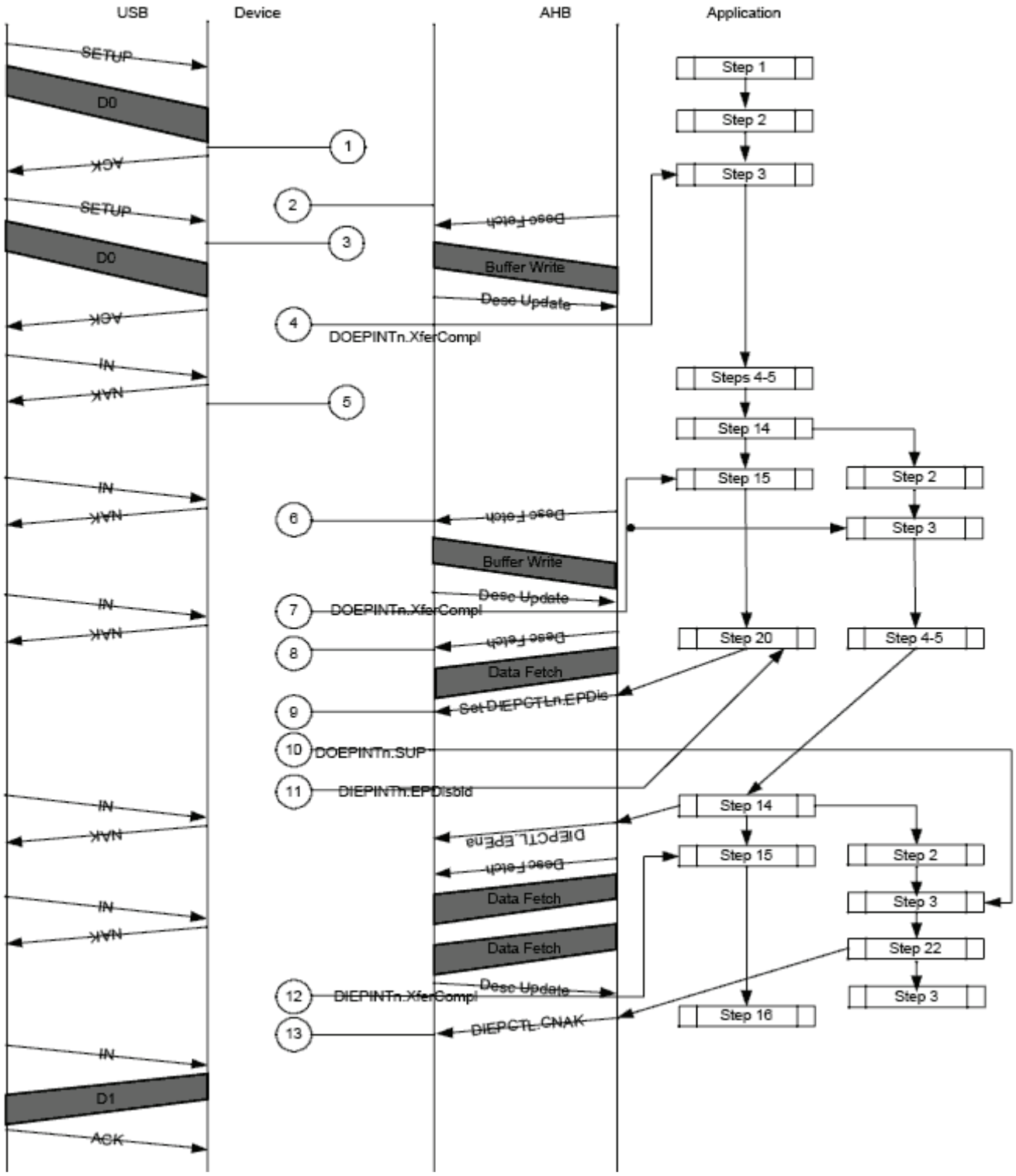


Figure 13.38 Back-to-Back SETUP During Control Read

13.8.7.5 Extra Tokens During Control Write Data Phase

This example assumes a three-stage control write transfer with only Wlength field in the SETUP indicating only 1 packet in the data phase. But the host sends an additional OUT packets which the core STALLs.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2)
.
5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Application clears NAK for the data phase, after receiving DOEPINTn.SetUp interrupt.

7. The core ACKs and the next OUT token because the NAK has been cleared (provided there is enough space in the RxFIFO.

8. DMA starts transferring the OUT packet to the system memory.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the OUT packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close descriptor with DMA_DONE status.
   - ✦ The core generates DOEPINTn.XferCompl interrupt after having transferred the OUT packet to the system memory. Since there were only one packet in the data phase, the data phase is complete here.
   - ✦ The core initially NAK s the extra tokens send by the host, because the core internally sets NAK after each OUT packet for the data phase of control write.

9. Application sets STALL to stall any extra tokens.

10. The core stalls the next OUT/PING token.

11. Host switches to next control transfer, core ACKs the SETUP. This SETUP packet is transferred to the system memory buffer originally allocated for Status phase.

12. The core generates DOEPINTn.XferCompl interrupt after transferring the SETUP packet to the system memory.

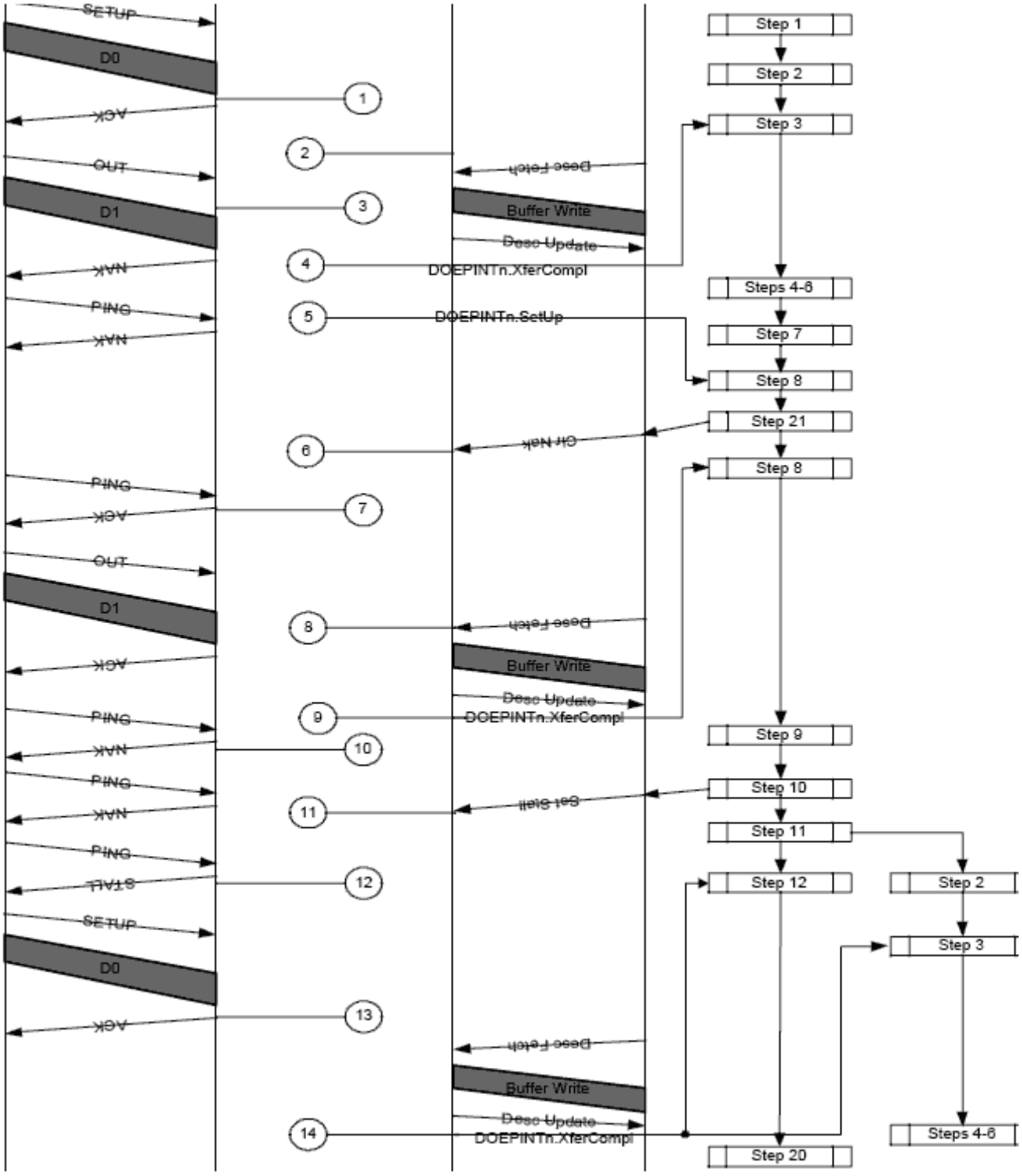


Figure 13.39 Extra Tokens During Control Write Data Phase

13.8.7.6 Extra Tokens During Control Read Data Phase

In this example, it is assumed that the data phase consists of 2 packets, and the application allocates these two packets in a single buffer. After the data phase is complete and the two packets have been transferred, the core sends an extra IN token and then the application sets Stall.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO. Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory(Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Data phase IN tokens are NAKed until this point because the NAK has not yet been cleared by the application.

7. The core starts fetching the IN data after the application enables IN DMA (In this example it is assumed that multiple packets are in the same buffer. But it could also be in different buffers). This involves the following steps

   - ✦ Fetch the descriptor pointed by DIEPDMA.
   - ✦ Fetch the data into the corresponding Tx FIFO.
   - ✦ Close the descriptor with DMA_DONE status.

8. The application clear the NAK after receiving the setup complete (DOEPINT.SetUp) interrupt. Set the Stall bit after all the Data has been pushed in the FIFO

9. After all the data has been fetched for the descriptor (Step 7), core generates DIEPINT.XferCompl interrupt.

10. The core sends data in response to the IN token for the data phase.

11. The core sends out the last packet of the IN data phase.

12. Host sends an extra token.

13. The core Stalls the IN token and also automatically Stalls the Status phase if the Host switches to the Status phase.

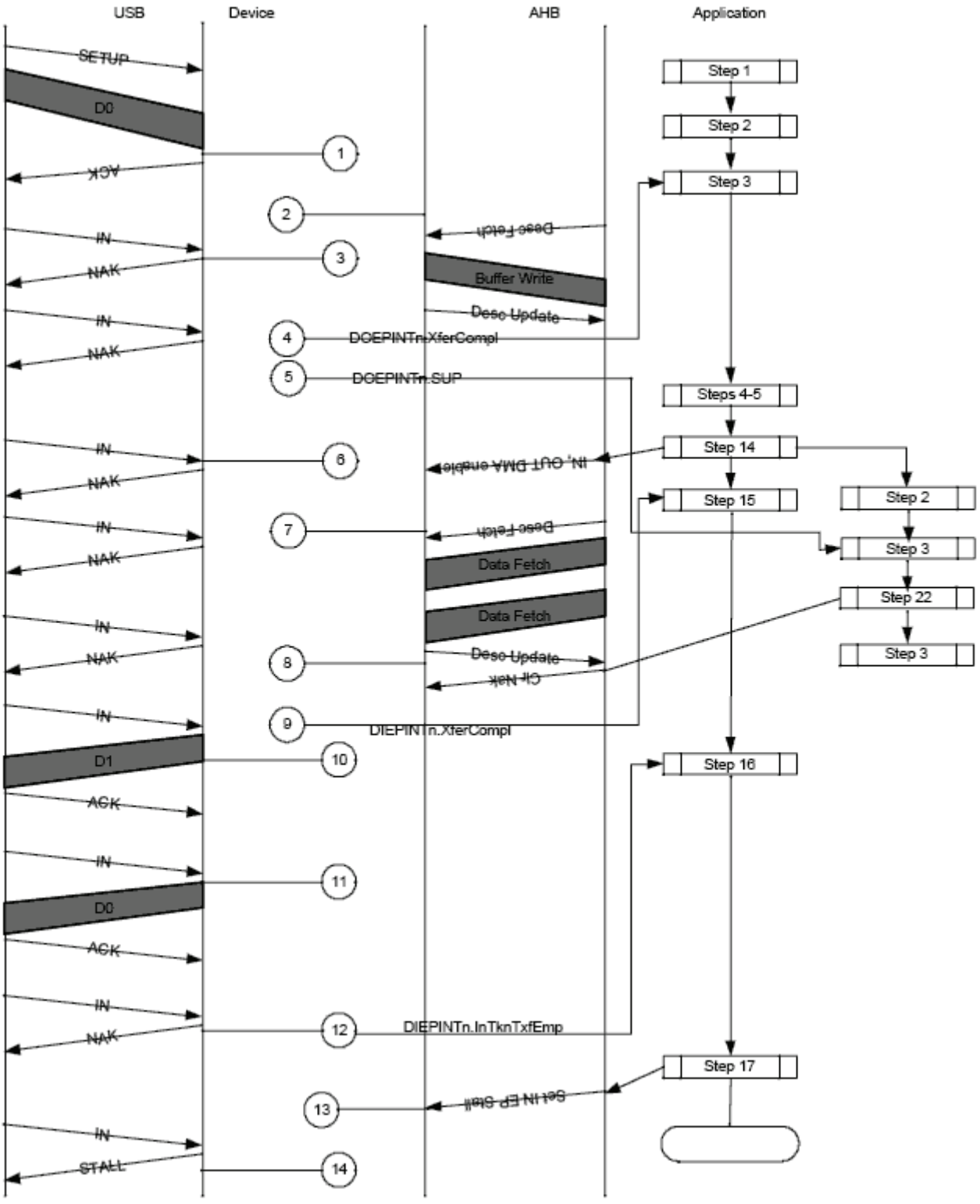


Figure 13.40 Extra IN Tokens During Control Read Data Phase

13.8.7.7 Premature SETUP During Control Write Data Phase

This example shows a Three-Stage Control Write transfer with host sending a premature Control Write SETUP packet during the data phase.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.
   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. The core receives a SETUP packet during the data phase. This is an unexpected SETUP packet.On receiving this SETUP, the SETUP data is pushed into the RxFIFO and the core again sets NAK on both IN and OUT endpoints of the control endpoint (NAK was already set because of the first SETUP packet received).

7. Application decodes the previous DOEPINT.SetUp interrupt and clears the NAK, unaware of the fact that there is another SETUP packet sitting in the RxFIFO for the same control endpoint. On seeing this condition, core does not allow clearing of the NAK bit, and masks the clearing of NAK. The core takes this decision based on the fact that a SETUP_COMPLETE status is pending in the RXFIFO.

8. The DMA detects the RxFIFO as non-empty (because of the unexpected SETUP) and does following

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.
   - ✦ The core NAKs the data phase OUT token because NAK bit clearing by the application did not take effect (as explained in Step 7).
   - ✦ The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 8).
   - ✦ The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status (for the unexpected SETUP packet received) out of the RxFIFO.

9. Application clears the NAK after decoding the latest SETUP packet. This time, the core does not mask the clearing of the NAK because there are no more SETUP_COMPLETE status sitting in the RxFIFO.

10. The core ACKs the next OUT/PING token of the data phase. 11. DMA starts transferring the OUT packet to the system memory.

    - ✦ Fetch the descriptor pointed by DOEPMA.
    - ✦ Transfer the OUT packet from the RxFIFO to the buffer pointed by the descriptor.
    - ✦ Close descriptor with DMA_DONE status.
    - ✦ The core generates DOEPINTn.XferCompl interrupt after having transferred the OUT packet to the system memory.
    - ✦ The remaining steps are similar to Steps 11-18 of “Application Programming Sequence”

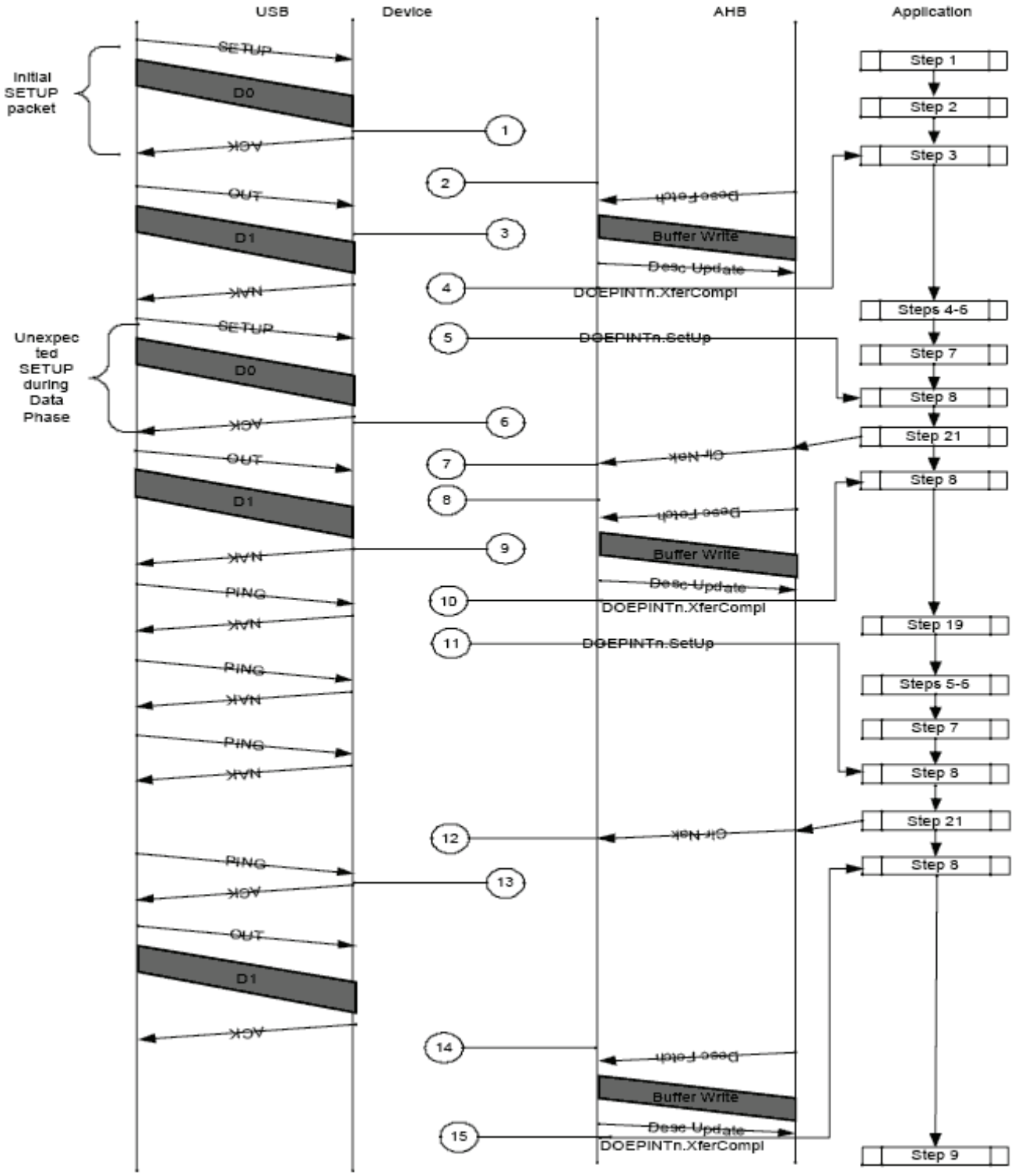


Figure 13.41 Premature SETUP During Control Write Data Phase

13.8.7.8 Premature SETUP During Control Read Data Phase

In this example, it is assumed that the data phase consists of 2 packets, and the application allocates these two packets in a single buffer. The host switches to a new control read command after having send two IN tokens during the data phase.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase IN token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase IN tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory(Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Host switches to a new Control transfer by sending a SETUP token. This is the premature SETUP packet. Core sets NAK on both IN and OUT control endpoints.

7. The core fetch the data for IN control endpoint after application enables the IN endpoint.

8. The core push SETUP_COMPLETE status into Rx FIFO on seeing the IN token for data phase.

9. Application clears NAK as a result of DOEPINT.SetUP (Setup complete) interrupt generated in Step 5. But core masks this clearing of setup_complete interrupt, because there is already one SETUP packet sitting in the Rx FIFO.

10. The core generates DIEPINTn.XFERCompl after closing the IN endpoint descriptor (for Step 7)

11. The core generates DOEPINTn.XferCompl after transferring the premature SETUP packet to system memory and closing the descriptor.

12. The core generates SETUP complete interrupt.

13. Application enables IN endpoint DMA for data phase.

14. The core fetches descriptor and data for IN endpoint.

15. Application clears IN endpoint NAK after receiving DOEPINTn.SetUP (Setup complete) interrupt. This time, the core does not mask the clearing of the Nak because NO SETUP packet is remaining in the Rx FIFO.

16. The core generates DIEPINTn.XferCompl interrupt after fetching the data and closing the descriptor. The remaining steps are same as steps 11 to 13 of “Internal Data Flow”.

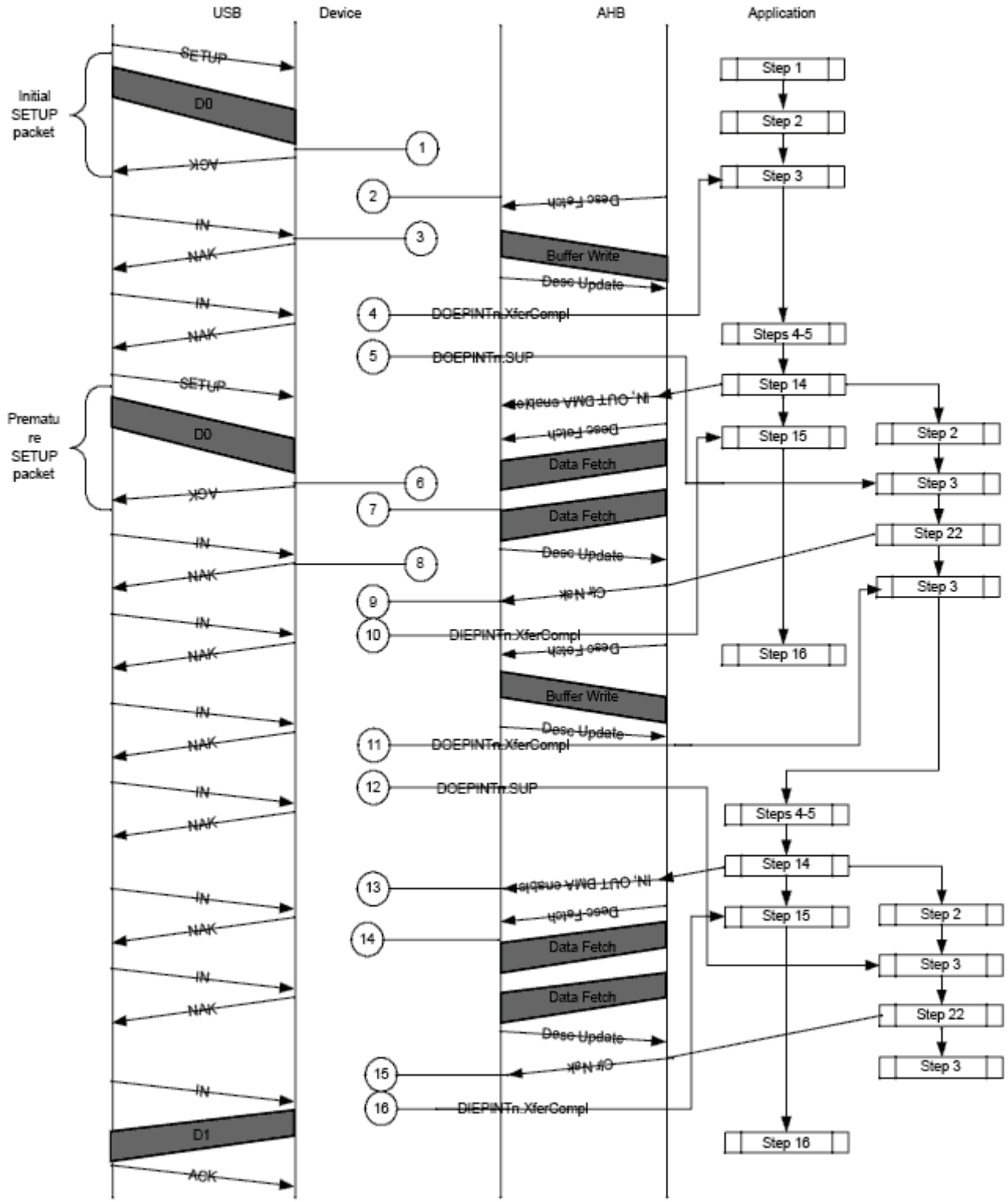


Figure 13.42 Premature SETUP During Control Read Data Phase

13.8.7.9 Premature Status During Control Write

This example assumes a Three-Stage control write transfer with only Wlength field in the SETUP indicating two packets in the data phase. But the host switch to data phase after the first packet of the data phase is complete.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Application clears NAK for the data phase, after receiving DOEPINTn.SetUp interrupt (Step 5).

7. The core ACKs and the next OUT token because the NAK has been cleared (provided there is enough space in the RxFIFO.

8. DMA sees TxFIFO non empty and starts transferring the OUT packet to the system memory.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the OUT packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close descriptor with DMA_DONE status.

9. Host switch to status phase (IN token) without completing the data phase.

10. The core generates DOEPINTn.XferComp after closing the descriptor after the data fetch.

11. Application sets up descriptor, enables IN endpoint and clear NAK.

12. The core starts to fetch the descriptor and data for the status phase once application has enabled the IN endpoint.

13. The core generates DIEPINTn.XferCompl after doing the data fetch and the descriptor update (step 12)

14. The core sends data out in response to status phase IN token.

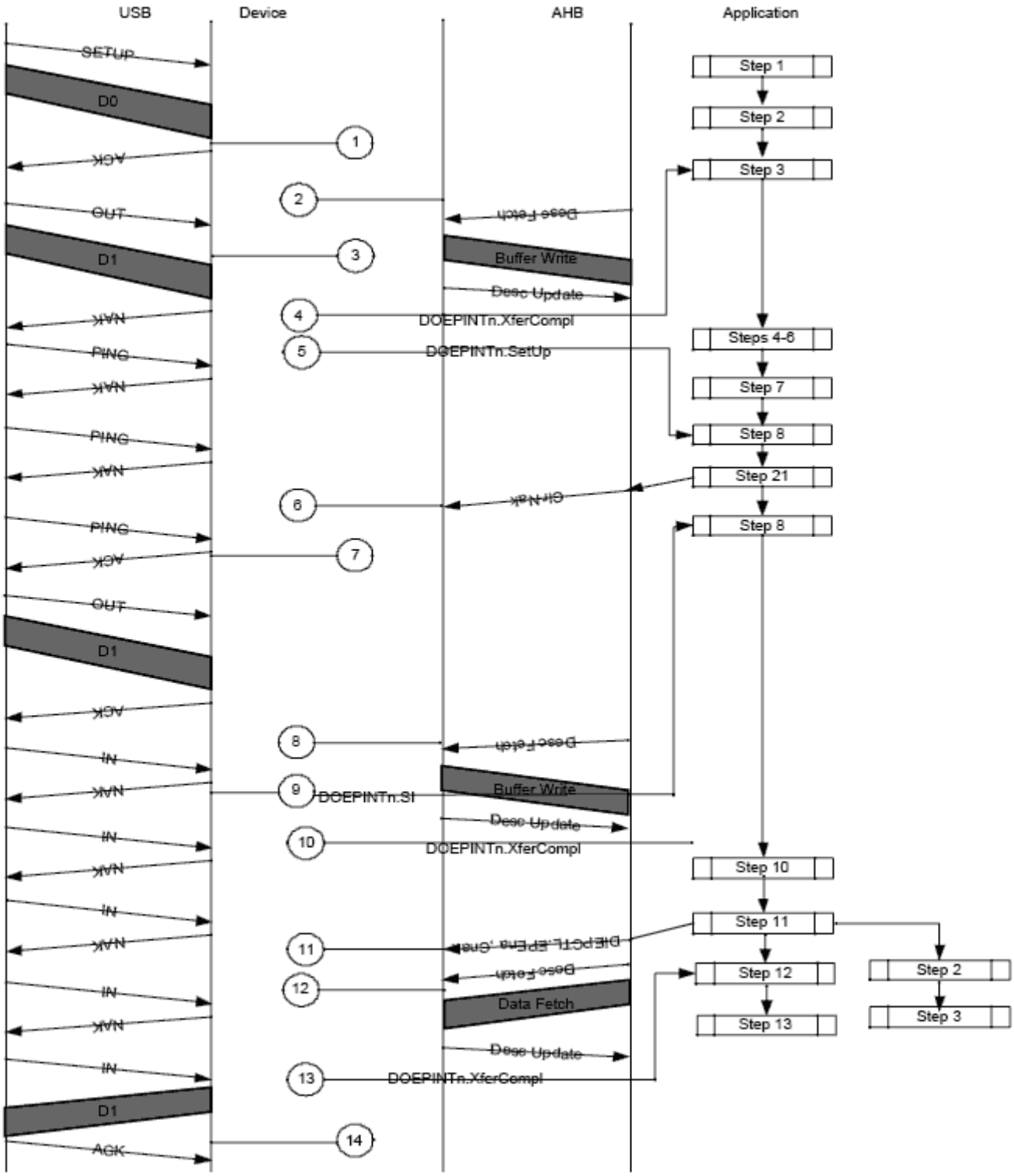


Figure 13.43 Premature Status Phase During Control Write

13.8.7.10 Premature Status During Control Read

In this example, it is assumed that the data phase consists of two packets, and the application allocates these two packets in a single buffer. After one packet in the data phase, host switches to status phase.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.
   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase IN token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase IN tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Data phase IN tokens are NAKed until this point because the NAK has not yet been cleared by the application.

7. The core starts fetching the IN data after the application enables IN DMA (In this example it is assumed that multiple packets are in the same buffer. But it could also be in different buffers). This involves the following steps

   - ✦ Fetch the descriptor pointed by DIEPDMA.
   - ✦ Fetch the data into the corresponding Tx FIFO.
   - ✦ Close the descriptor with DMA_DONE status.

8. Application clear the NAK after receiving the setup complete (DOEPINT.SetUp) interrupt.

9. After all the data has been fetched for the descriptor (Step 7), core generates DIEPINT.XferCompl interrupt.

10. The core sends data in response to the IN token for the data phase.

11. Host switches to status phase and sends the status phase OUT token. Core ACks the OUT packet because the NAK has already been cleared.

12. The DMA detects the RxFIFO as non-empty (because of the status phase data) and does following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.
   - ✦ The core generates DOEPINTn.XferCompl after transferring the status phase data to system memory and closing The descriptor.

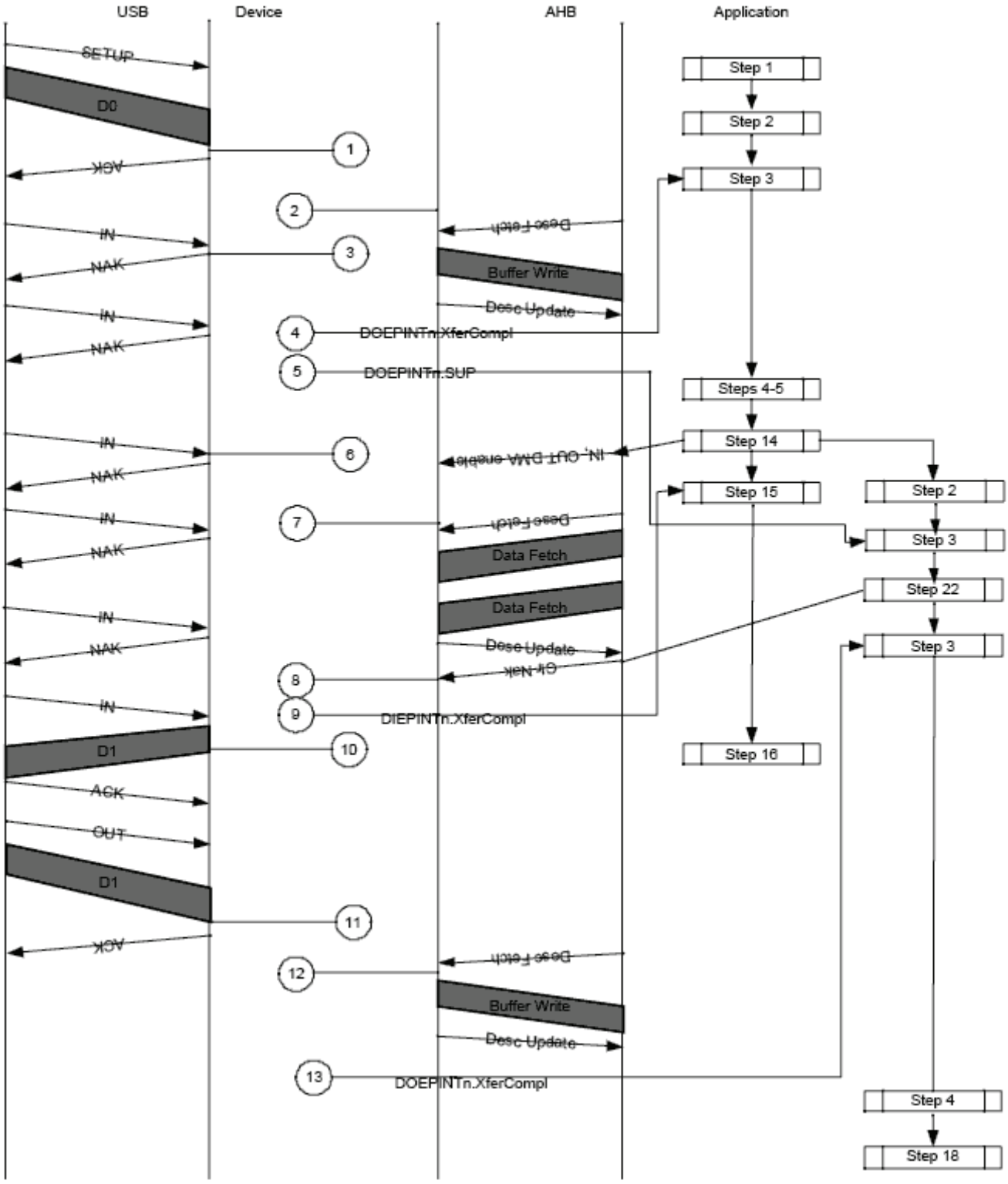


Figure 13.44 Premature Status Phase During Control Read

13.8.7.11 Lost ACK During Last Packet of Control Read

This is similar to the previous section. Figure 6-51 shows this.

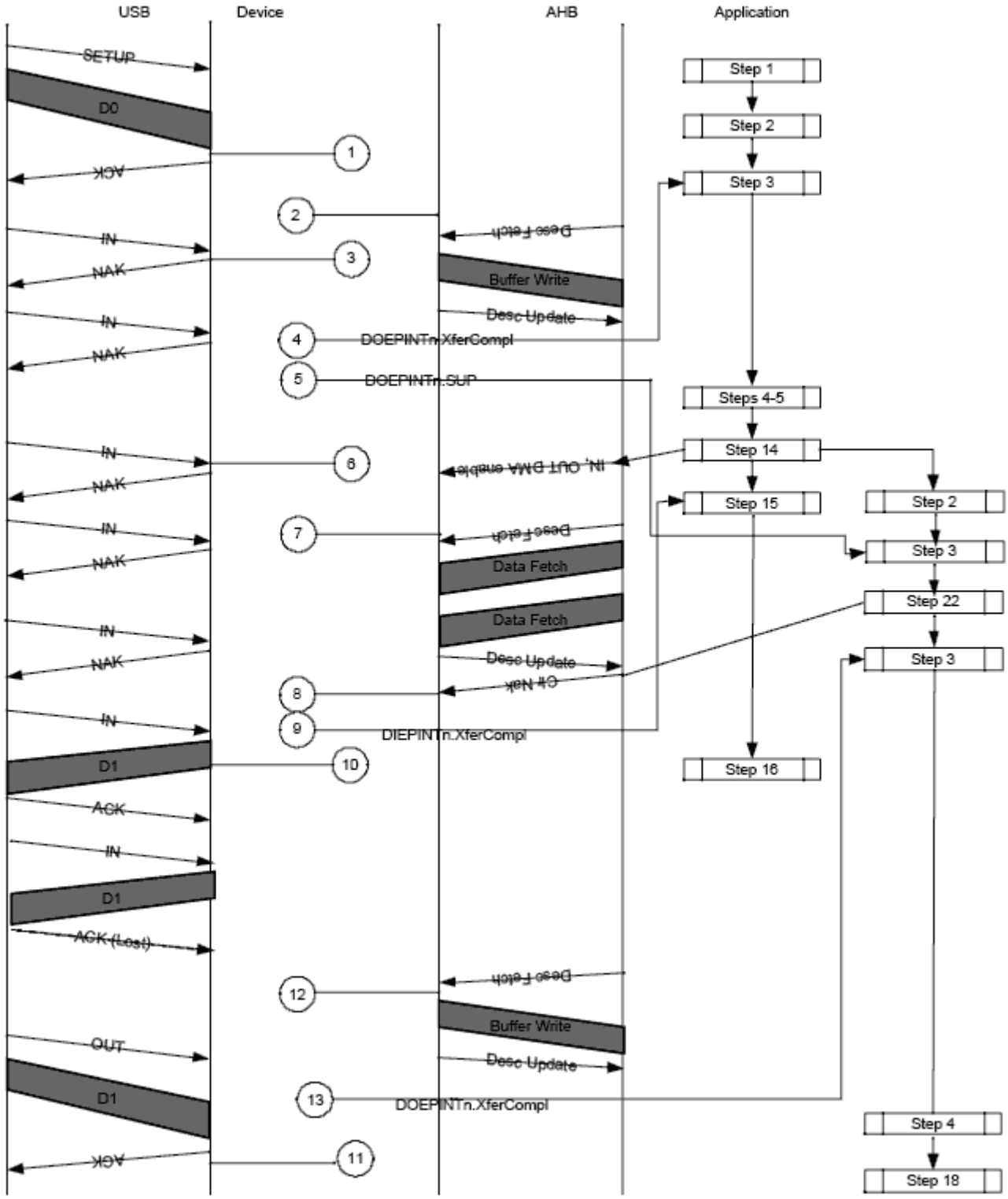


Figure 13.45 Lost ACK During Last Packet of Control Read

13.9 Bulk Transfer Handling in Scatter/Gather DMA Mode

The following section describes the Bulk transfer handling in Scatter/Gather DMA mode.

13.9.1 Bulk IN Transfer Data Transaction in Scatter-Gather DMA Mode

13.9.1.1 Interrupt usage

The following interrupts are of relevance.

1. DIEPINTn.XferCompl (Transfer complete, based on IOC bit in the descriptor)
2. DIEPINTn.BNA (Buffer Not Available)

13.9.1.2 Application Programming Sequence

This section describes the application programming sequence for Bulk IN transfer scenarios.

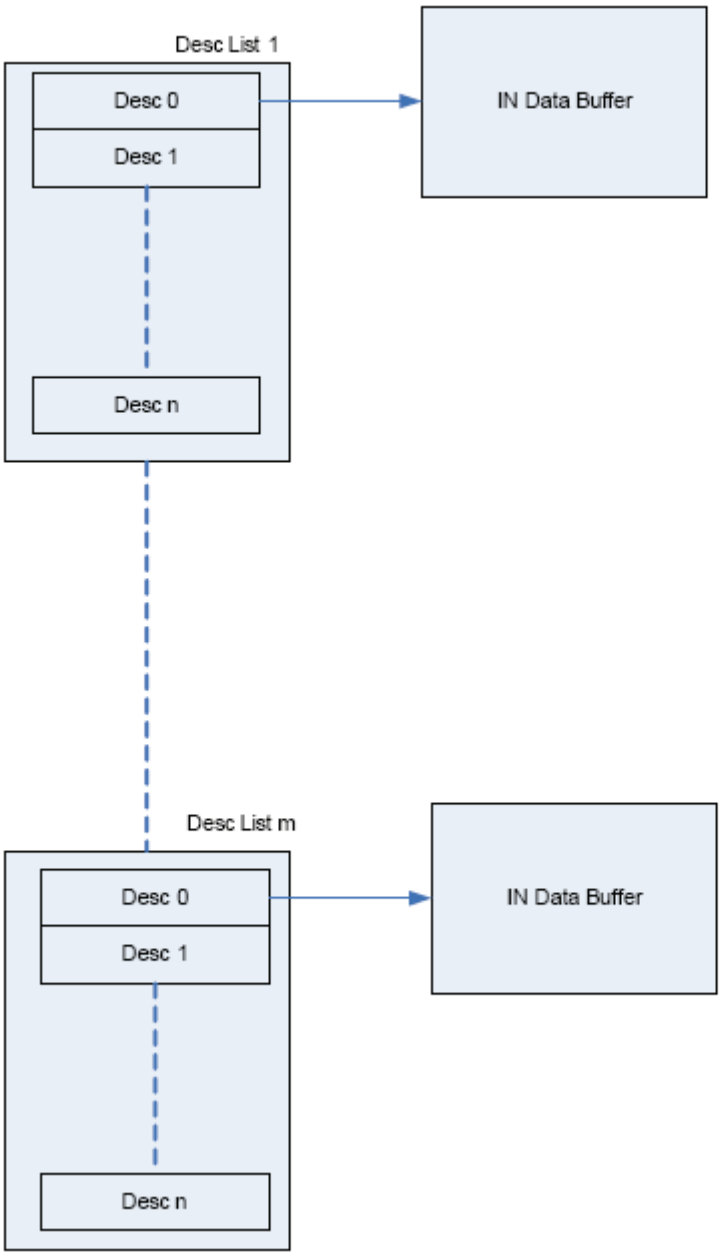


Figure 13.46 IN Descriptor List

1. Prepare Descriptor(s):

2. The application creates descriptor list(s) in the system memory pertaining to an Endpoint.

3. Each descriptor list may have up to *n* descriptors and there may be up to *m* descriptor lists.

4. Application may choose to set the IOC bit of the corresponding descriptor. If the IOC is set for the last descriptor of the list, the core generates DIEPINTn.XferCompl interrupt after the entire list is processed.

5. Program DIEPDMAn:

a. Application programs the base address of the descriptor in the corresponding IN Endpoint DIEPDMAn register.

6. Enable DMA:

a. Application programs the corresponding endpoint DIEPCTLn register with the following

i. DIEPCTLn.MPS—Max Packet size of the endpoint
ii. DIEPCTLn.CNAK—Set to 1 to clear the NAK
iii. DIEPCTLn.EPEna—Set to 1 to enable the DMA for the endpoint.

7. Wait for Interrupt:
a. On reception of DIEPINTn.XferCompl, application must check the Buffer status and Tx Status field of the descriptor to ascertain that the descriptor closed normally.

DIEPINTn.BNA interrupt gets generated by the core when it encounters a descriptor in the list whose Buffer Status field is not Host Ready. In this case, the application is suppose to read the DIEPDMAn register to ascertain the address for which the BNA interrupt is asserted to take corrective action.

13.9.1.3 Internal Flow

Bulk IN Transfers

The core handles Bulk IN transfers internally as functionally depicted in Figure 6-53 on page 475 (Non ISO IN Descriptor/Data Processing). Figure 6-54 depicts this flow.

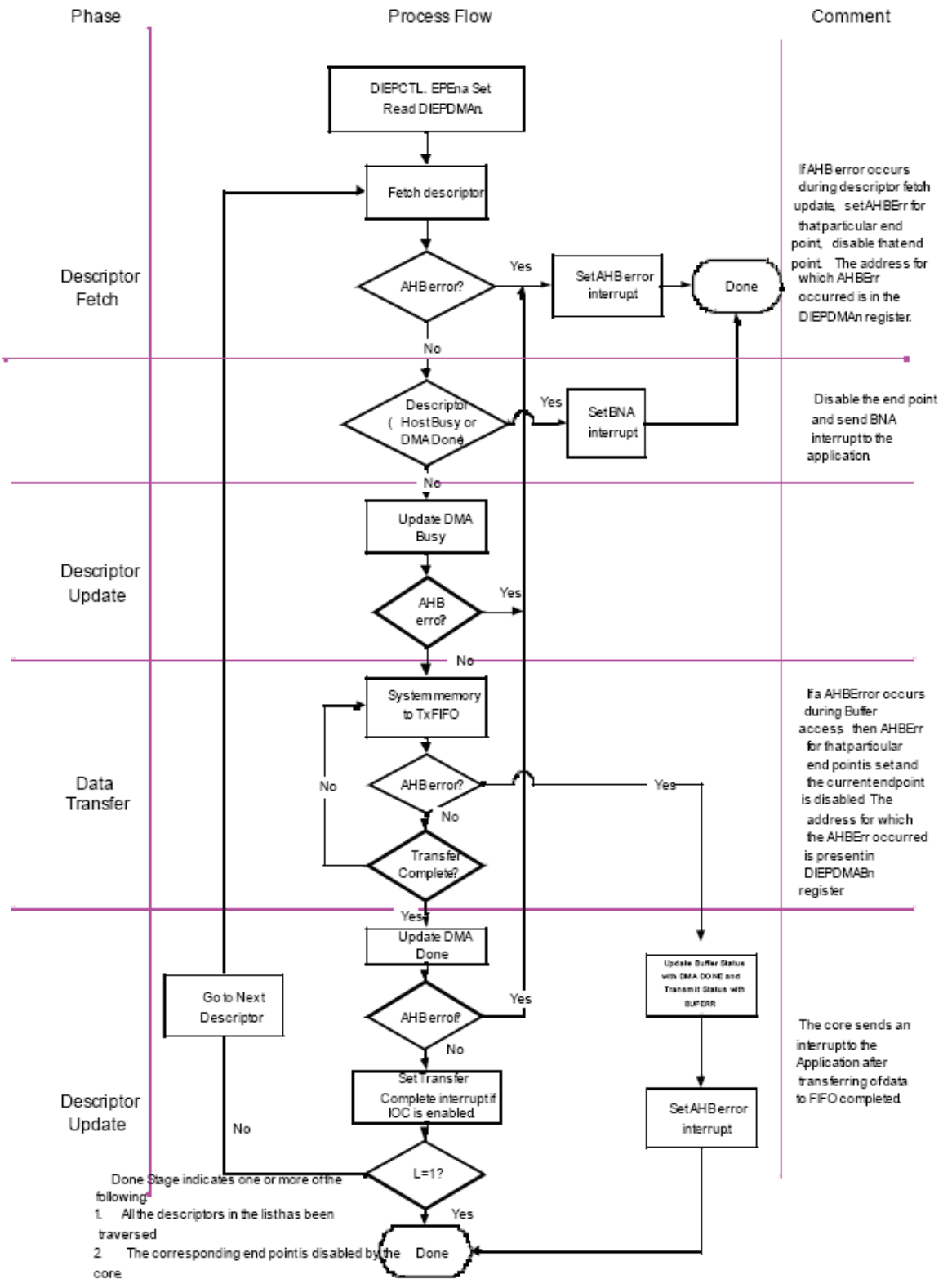
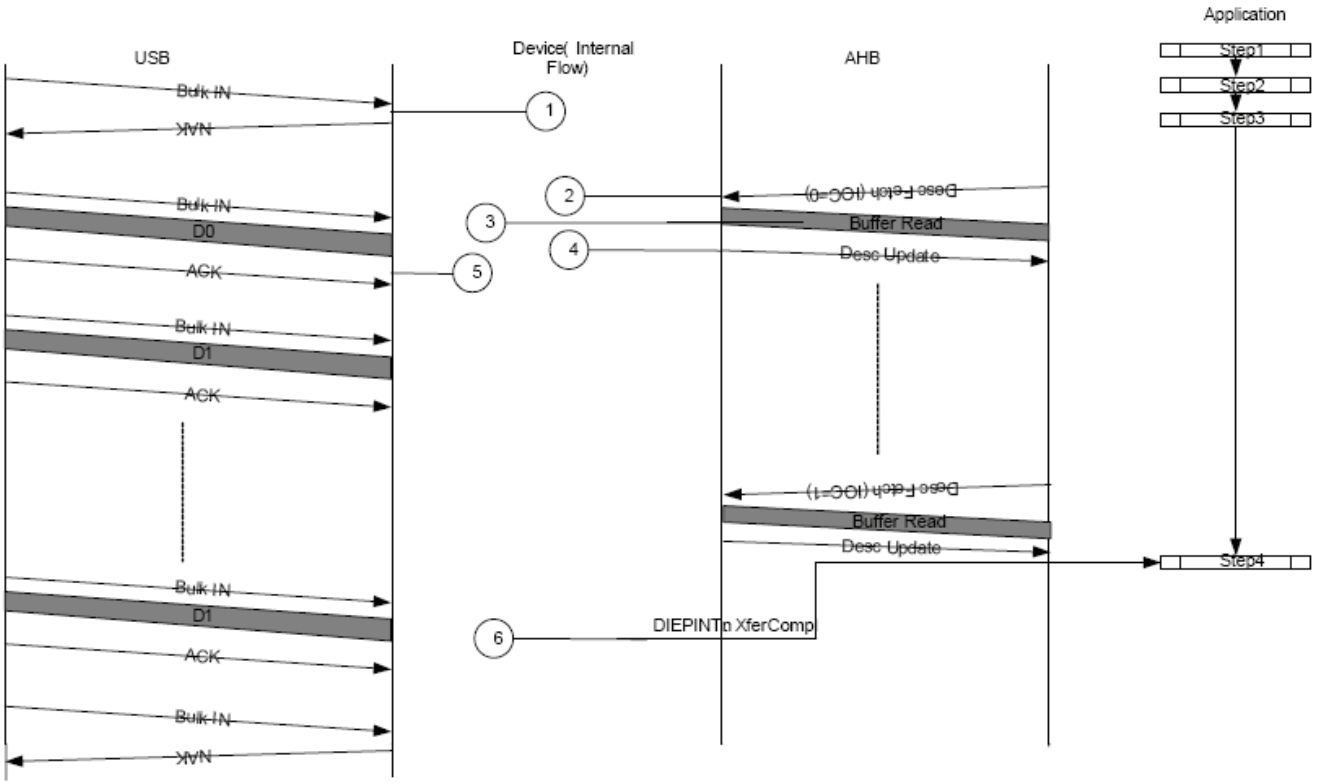


Figure 13.47 Non ISO IN Descriptor/Data Processing



1. When a BULK IN token is received on an end point before the corresponding DMA is enabled, (DIEPCTLn.EPEna = 1'b0), it is NAKed on USB.

2. As a result of application enabling the DMA for the corresponding end point (DIEPCTLn.EPEna=1), the core fetches the descriptor and processes it.

3. The DMA fetches the data from the system memory and populates its internal FIFO with this data.

4. After fetching all the data from a descriptor, the core closes the descriptor with a DMA_DONE status.

5. On reception of BULK IN tokens on USB, data is sent to the USB Host.

6. After the last descriptor in the chain is processed, the core generates DIEPINTn.XferCompl interrupt provided the IOC bit for the last descriptor is set.

13.9.2 Bulk OUT Data Transaction in Scatter-Gather Mode

13.9.2.1 Interrupt Usage

The following interrupts are of relevance.

1. DOEPINTn.XferCompl (Transfer complete, based on IOC bit in the descriptor)

2. DOEPINTn.BNA (Buffer Not Available)

13.9.2.2 Application Programming Sequence

This section describes the application programming sequence to take care of Bulk OUT transfer scenarios.

13.9.2.3 Application Programming Sequence

This section describes the application programming sequence to take care of Bulk OUT transfer scenarios.

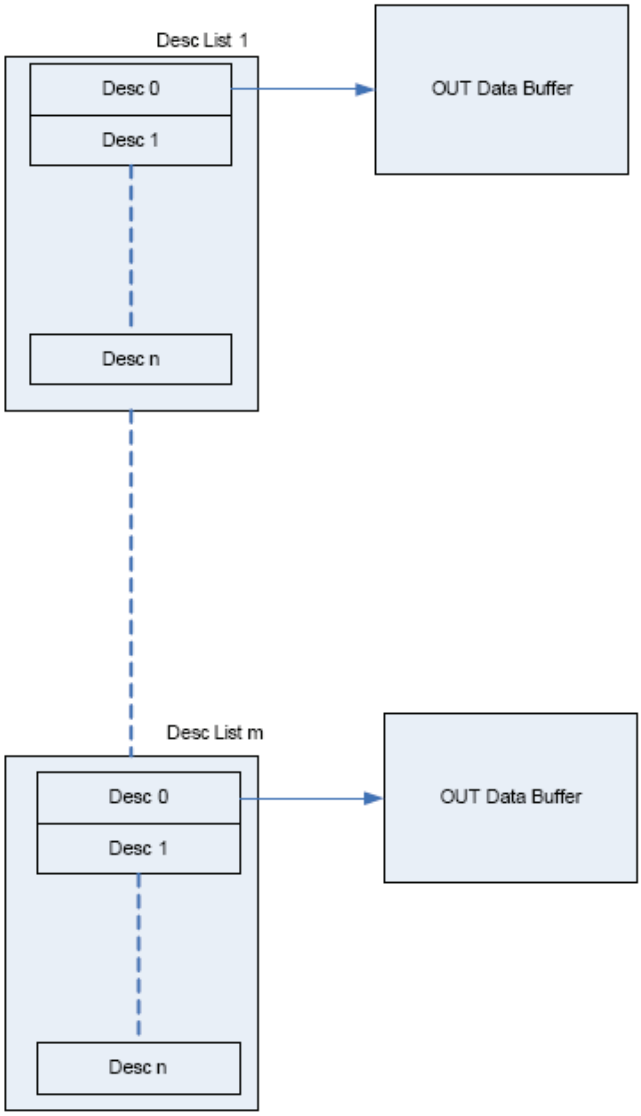


Figure 13.48 OUT Descriptor List

1. Prepare Descriptor(s):

2. The application creates descriptor list(s) in the system memory pertaining to an Endpoint.

3. Each descriptor list may have up to *n* descriptors and there may be up to *m* descriptor lists.

4. Application may choose to set the IOC bit of the corresponding descriptor. If the IOC is set for the last descriptor of the list, the core generates DOEPINTn.XferCompl interrupt after the entire list is processed.

    a. Based on L bit and MTRF bit combinations, the core may disable the end point. Refer to Table 6-3 on page 433 for bit field descriptions.

5. Program DOEPDMAn:

    a. Application programs the base address of the descriptor in the corresponding OUT Endpoint DOEPDMAn register.

6. Enable DMA:

    a. Application programs the corresponding endpoint DOEPCTLn register with the following

        i. DOEPCTL.MPS—Max Packet size of the endpoint
        ii. DOEPCTL.CNAK—Set to 1 to clear the NAK

iii. DOEPCTL.EPEna—Set to 1 to enable the DMA for the endpoint.

7. Wait for Interrupt:

a. On reception of DOEPINTn.XferCompl, application must check the Buffer status and Rx Status field of the descriptor to ascertain that the descriptor closed normally.

DOEPINTn.BNA interrupt gets generated by the core when it encounters a descriptor in the list whose Buffer Status field is not Host Ready. In this case, the application is suppose to read the DOEPDMAn register to ascertain the address for which the BNA interrupt is asserted to take corrective action.

13.9.2.4 Internal Flow

The core handles Bulk OUT transfers internally as depicted in Figure 6-56. Figure 6-57 also diagrams this flow.

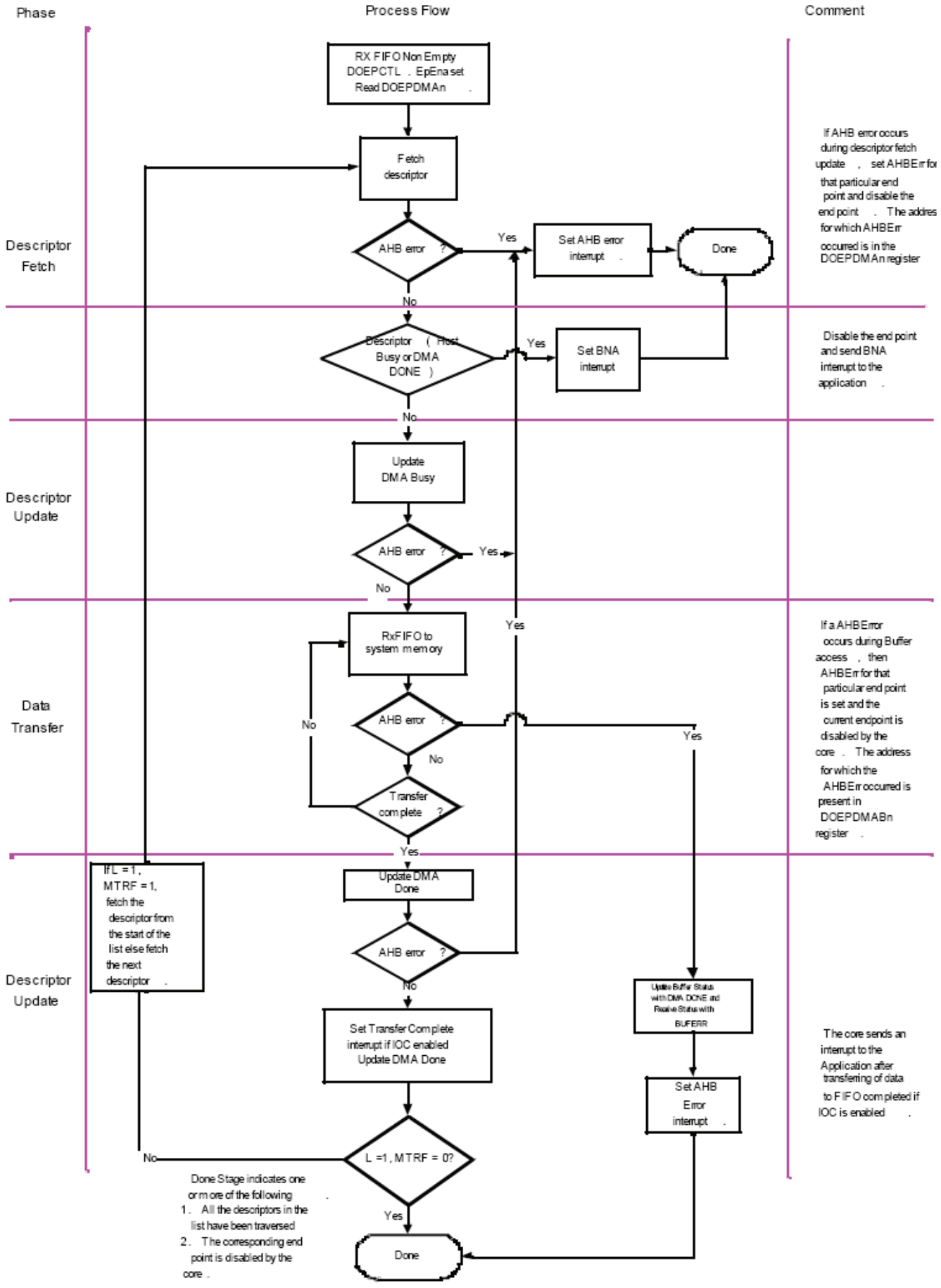

Phase
Process Flow
Comment
Descriptor Fetch
Descriptor Update
Data Transfer
Descriptor Update
RX FIFO Non Empty DOEPCTL . EpEna set Read DOEPDMAn .
Fetch descriptor
AHB error ?
Yes
Set AHB error interrupt .
Done
No
Descriptor (Host Busy or DMA DONE)
Yes
Set BNA interrupt
No
Update DMA Busy
AHB error ?
Yes
No
RxFIFO to system memory
AHB error ?
No
Yes
Transfer complete ?
Yes
Update DMA Done
AHB error ?
No
Set Transfer Complete interrupt if IOC enabled Update DMA Done
L =1 , MTRF = 0?
No
Yes
Done
If L = 1 , MTRF = 1, fetch the descriptor from the start of the list else fetch the next descriptor .
Update Buffer Status with DMA DONE and Receive Status with BUFERR
Set AHB Error interrupt .
Done Stage indicates one or more of the following .
1. All the descriptors in the list have been traversed
2. The corresponding end point is disabled by the core .
If AHB error occurs during descriptor fetch update , set AHBErr for that particular end point and disable the end point . The address for which AHBErr occured is in the DOEPDMAn register
Disable the end point and send BNA interrupt to the application .
If a AHBError occurs during Buffer access , then AHBErr for that particular end point is set and the current endpoint is disabled by the core . The address for which the AHBErr occurred is present in DOEPDMABn register .
The core sends an interrupt to the Application after transferring of data to FIFO completed if IOC is enabled .

Figure 13.49 Non ISO OUT Descriptor/Data Buffer Processing

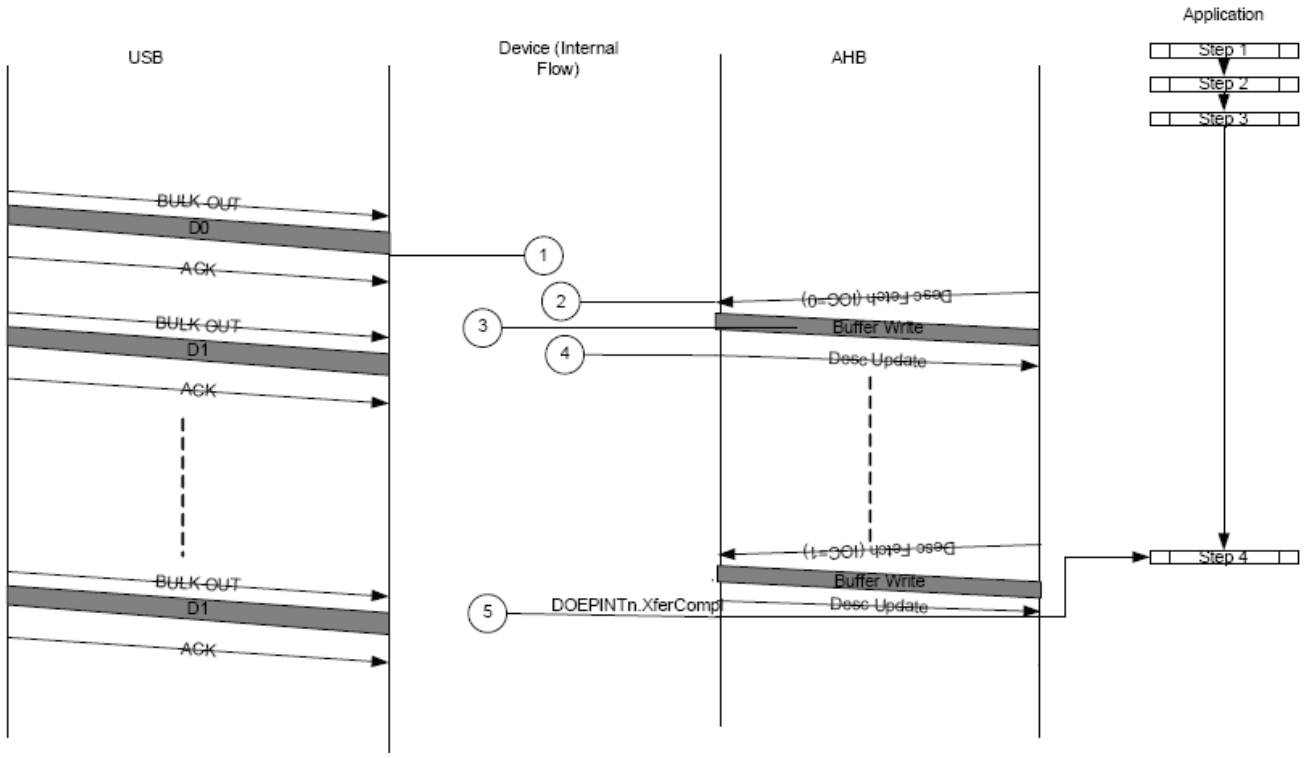


Figure 13.50 Bulk OUT Transfers

1. When a BULK OUT token is received on an end point, the core stores the received data internally in a FIFO.

2. As a result of application enabling the DMA for the corresponding end point (DOEPCTLn.EPEna=1), the core fetches the descriptor and processes it.

3. The DMA transfers the data from the internal FIFO to system memory.

4. After transferring all the data from the FIFO, the core closes the descriptor with a DMA_DONE status.

5. After the last descriptor in the chain is processed, the core generates DOEPINTn.XferCompl interrupt provided the IOC bit for the last descriptor is set.

13.10 Interrupt Transfer Handling in Descriptor-Based Scatter/Gather DMA Mode

13.10.1 Interrupt IN Data Transaction in Scatter/Gather DMA Mode

Application programming for Interrupt IN transfers is as with the Bulk IN transfer sequence.
The core handles Interrupt IN transfers internally in the same way it handles Bulk IN transfers

13.10.2 Interrupt OUT Transfer

Application programming for Interrupt OUT transfers is as with the Bulk OUT transfer sequence.
The core handles Interrupt OUT transfers internally in the same way it handles Bulk OUT Transfers

13.11 Isochronous Transfer Handling in Descriptor-Based Scatter/Gather DMA Mode

13.11.1 Isochronous IN Transfer

The application programming for Isochronous IN transfers is in the same manner as Bulk IN transfer sequence.
The following behavior is of importance while working with Isochronous IN end points

DCTL.IgnrFrmNum = 1'b1

The way the core handles Isochronous IN transfers internally in the same way as it handles Bulk IN Transfers.

DCTL.IgnrFrmNum = 1'b0

The core closes the descriptor and clears the corresponding fetched data in the FIFO if the USB (micro)frame number to which the descriptor belongs is elapsed.

13.11.1.1 Isochronous Transfers in Scatter/Gather (Descriptor DMA) Mode

This topic includes descriptions of both isochronous IN and OUT transfers

Isochronous IN

In the case of ISO IN After descriptor is fetched, the frame number field M is compared with current USB frame number N.

If the frame number in the fetched descriptor is already elapsed (M<N) then the descriptor is closed with status changed to DMA Done.

If the frame number in the fetched descriptor is for future (N>M+1) then the descriptor is left untouched. The Core suspends and re-look at this descriptor contents in the next frame/microframe.

- ❖ If the frame number in the fetched descriptor is for current or next frame (N=M or M+1) then the descriptor is further processed as per the flow chart. At the end of data transfer from memory to TxFIFO the above check must be performed. And if the data fetch finished in the subsequent frame, data must be flushed and descriptor must be closed (DMA Done) with BUFFLUSH status.

- ❖ For ISO IN, the application creates a series of descriptors (D,D+1,D+2.....) for a given periodic end point corresponding to successive frames (N,N+1,N+2.....).

Note The series of descriptors does not correspond to the series of frames in the same order.

For example, D and D + 1 may correspond to N, D + 2 may correspond to N + 1 and so on except in the case where the application can create more than one descriptor for the same microframe. The core fetches the descriptor and compares the frame/μframe number field with the current frame/μframe number.

If the fetched descriptor corresponds to a frame which has already elapsed, the core updates the descriptor with DMA Done Buffer status and proceeds to the next descriptor.

If the next descriptor fetched indicates that it corresponds to frame number N or N + 1, it services it. In the process of fetching the descriptors, if the core determines that the descriptor corresponds to a future frame/μframe (> N + 1), it does not service the descriptor in that frame/μframe. Instead, it moves on to the next periodic endpoint or non-periodic endpoint without disabling the current periodic endpoint. It revisits this endpoint in the next frame/μframe and repeats the process.

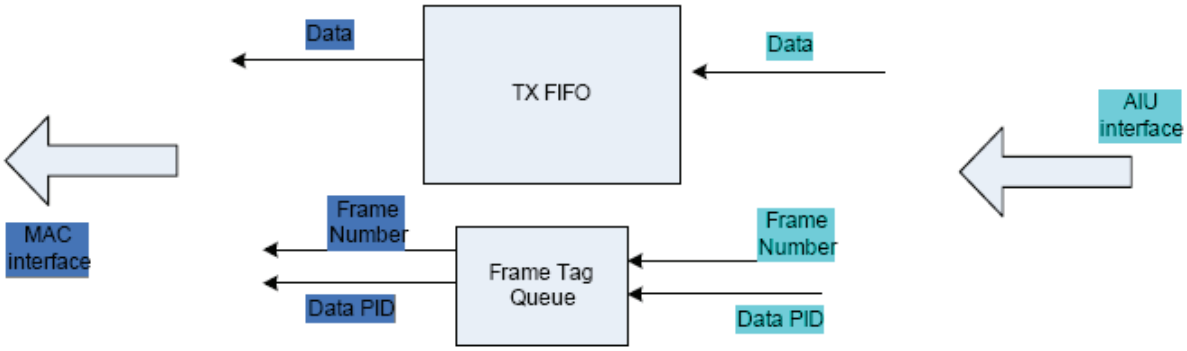


Figure 13.51 ISO IN Data Flow

Application Programming Sequence

This section describes the application programming sequence for Isochronous IN transfer scenarios.

Prepare Descriptor(s):

The application creates descriptor list(s) in the system memory pertaining to an Endpoint. Each descriptor list may have up to *n* descriptors and there may be up to *m* descriptor lists. Application may choose to set the IOC bit of the corresponding descriptor. If the IOC is set for the last descriptor of the list, the core generates DIEPINTn.XferCompl interrupt after the entire list is processed.

1. Program DIEPDMAn: a. Application programs the base address of the descriptor in the corresponding IN Endpoint DIEPDMAn register.

2. Enable DMA:

a. Application programs the corresponding endpoint DIEPCTLn register with the following
   i. DIEPCTLn.MPS—Max Packet size of the endpoint
   ii. DIEPCTLn.CNAK—Set to 1 to clear the NAK
   iii. DIEPCTLn.EPEna—Set to 1 to enable the DMA for the endpoint.

3. Wait for Interrupt:
   a. On reception of DIEPINTn.XferCompl, application must check the Buffer status and Tx Status field of the d escriptor to ascertain that the descriptor closed normally.

DIEPINTn.BNA interrupt gets generated by the core when it encounters a descriptor in the list whose Buffer Status field is not Host Ready. In this case, the application is suppose to read the DIEPDMAn register to ascertain the address for which the BNA interrupt is asserted to take corrective action.

13.11.1.2 Internal Flow

The core handles isochronous IN transfers internally as functionally depicted in Figure 6-59. Figure 6-60 on page 485 also diagrams this flow.

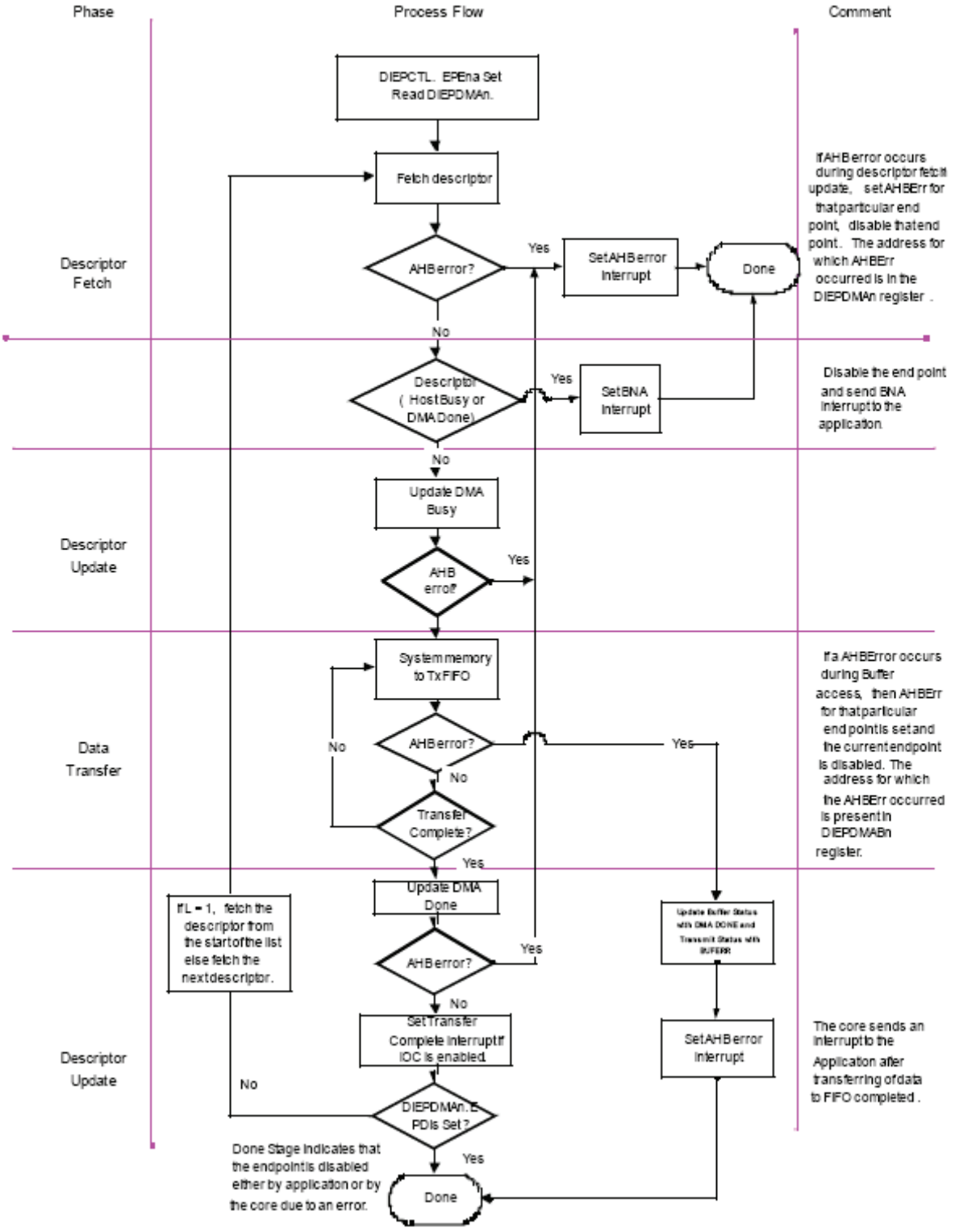


Figure 13.52 ISO IN Descriptor/Data Processing

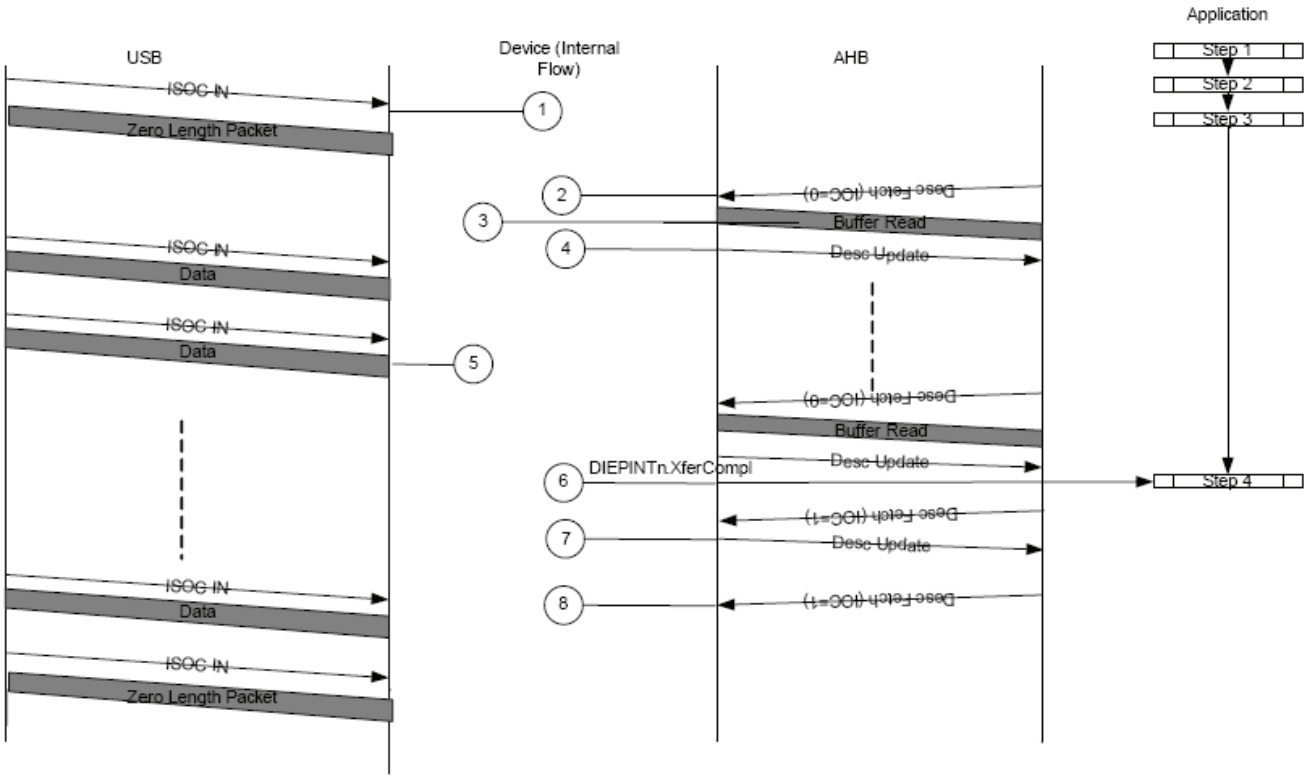


Figure 13.53 Isochronous IN Transfers

1. When an Isochronous IN token is received on an end point before the corresponding DMA is enabled, (DIEPCTLn.EPEna = 1'b0), zero length packet is sent on USB.

2. As a result of application enabling the DMA for the corresponding end point (DIEPCTLn.EPEna=1), the core fetches the descriptor. If the descriptor belongs to the current or the next USB frame number, the core processes it.

3. The DMA fetches the data pointed by the above descriptor from the system memory and populates its internal FIFO with this data.

4. After fetching all the data, the core closes the descriptor with a DMA_DONE status.

5. On reception of Isochronous IN tokens on USB, data is sent to the USB Host.

6. After the last descriptor in the chain is processed, the core generates DIEPINTn.XferCompl interrupt provided the IOC bit for the last descriptor is set.

7. When the DMA fetches a descriptor whose USB frame number has been already elapsed, it closes that descriptor with a DMA_DONE status without fetching the data for that descriptor.

8. When the DMA fetches a descriptor which has a future USB frame number, it does not service it in the current context. It services it in the future.

13.11.2 Isochronous OUT Transfer

The application programming for isochronous out transfers is in the same manner as Bulk OUT transfer sequence, except that the application creates only 1 packet per descriptor for an isochronous OUT endpoint. The core handles isochronous OUT transfers internally in the same way it handles Bulk OUT transfers, and as depicted in Figure 6-61.

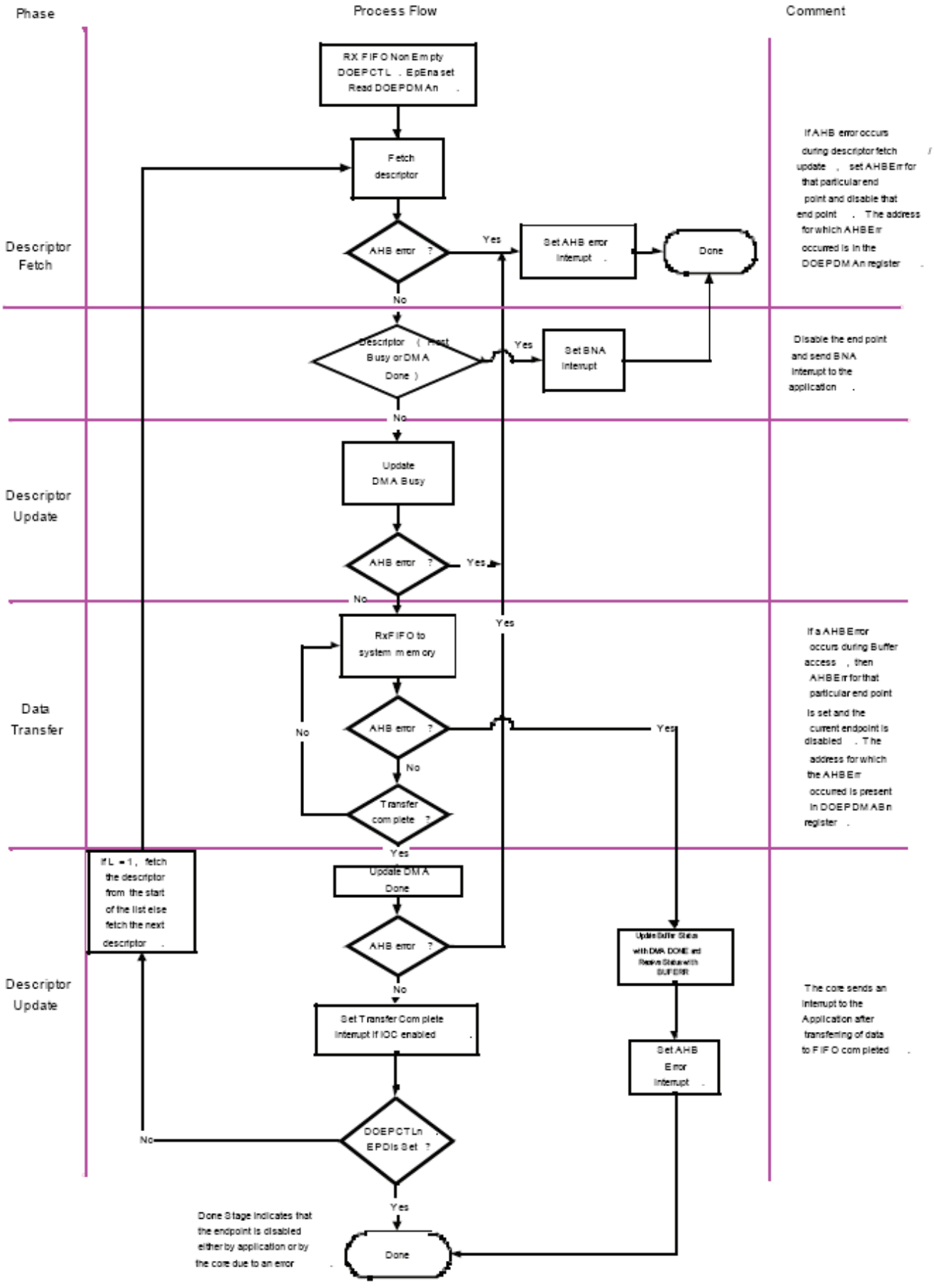


Figure 13.54 Isochronous OUT Descriptor/Data Buffer Processing

Isochronous OUT

For ISO OUT transactions, the core transfers the packets from the Rx FIFO to the system memory and updates the frame number field of the descriptor with the frame number in which the packet was received.The frame number for which data is received is extracted from the Receive Status queue and written back to the descriptor.

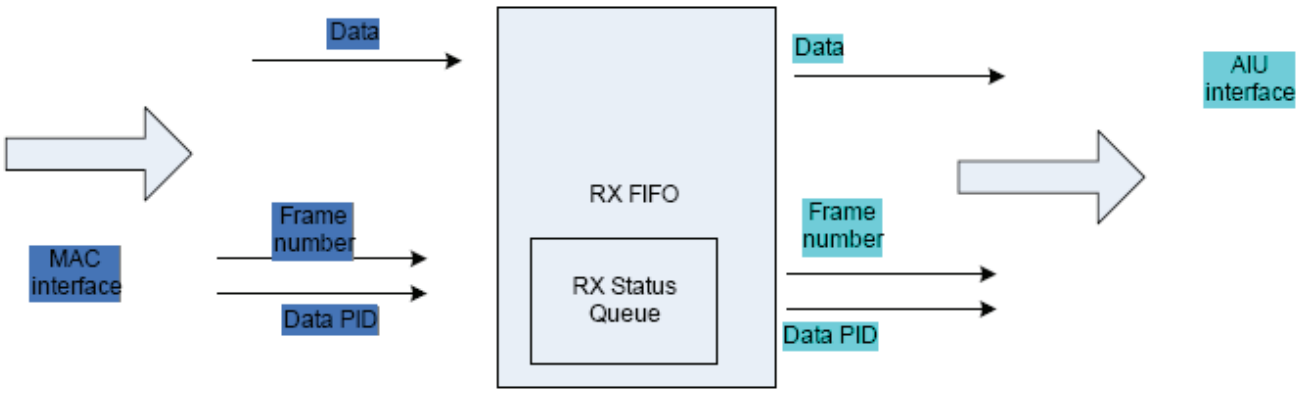


Figure 13.55 ISO Out Data Flow

Note Incomplete Isochronous Interrupt (GINTSTS.incomplete) is not generated in Scatter/Gather DMA mode. Received isochronous packets are sent unmodified to the application memory, with the corresponding frame number updated in the descriptor status.

13.12 OTG Programming Model

The otg core is an OTG device supporting HNP and SRP. When the core is connected to an “A” plug, it is referred to as an A-device. When the core is connected to a “B” plug it is referred to as a B-device. In Host mode, the otg core turns off VBUS to conserve power. SRP is a method by which the B-device signals the A-device to turn on VBUS power. A device must perform both data-line pulsing and VBUS pulsing, but a host can detect either data-line pulsing or VBUS pulsing for SRP. HNP is a method by which the B-device negotiates and switches to host role. In Negotiated mode after HNP, the B-device suspends the bus and reverts to the device role.

13.12.1 A-Device Session Request Protocol

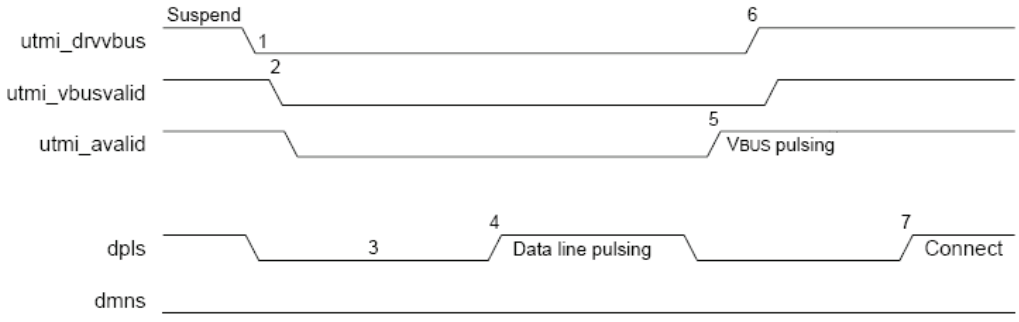


Figure 13.56 A-Device SRP

The application must set the SRP-Capable bit in the Core USB Configuration register. This enables the otg core to detect SRP as an A-device.

1. To save power, the application suspends and turns off port power when the bus is idle by writing the Port Suspend and Port Power bits in the Host Port Control and Status register.

2. PHY indicates port power off by deasserting the utmi_vbusvalid signal.

3. The device must detect SE0 for at least 2 ms to start SRP when VBUS power is off.

4. To initiate SRP, the device turns on its data line pull-up resistor for 5 to 10 ms. The otg core detects data-line pulsing.

5. The device drives VBUS above the A-device session valid (2.0 V minimum) for VBUS pulsing. The otg core interrupts the application on detecting SRP. The Session Request Detected bit is set in Global Interrupt Status register (GINTSTS.SessReqInt).

6. The application must service the Session Request Detected interrupt and turn on the Port Power bit by writing the Port Power bit in the Host Port Control and Status register. The PHY indicates port power-on by asserting utmi_vbusvalid signal.
7. When the USB is powered, the device connects, completing the SRP process.

13.12.2 B-Device Session Request Protocol

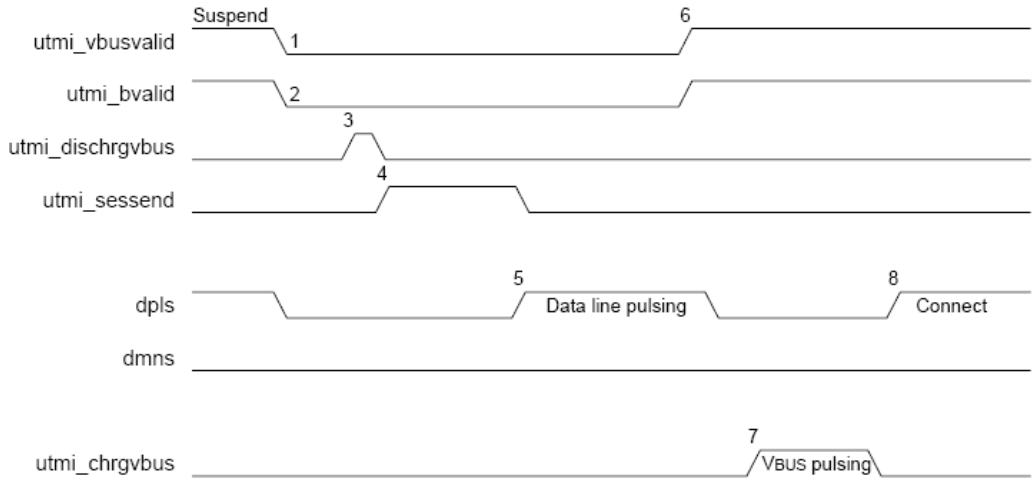


Figure 13.57 B-Device SRP

The application must set the SRP-Capable bit in the Core USB Configuration register. This enables the otg core to initiate SRP as a B-device. SRP is a means by which the otg core can request a new session from the host.

1. To save power, the host suspends and turns off port power when the bus is idle. PHY indicates port power off by deasserting the utmi_vbusvalid signal. The otg core sets the Early Suspend bit in the Core Interrupt register after 3 ms of bus idleness. Following this, the otg core sets the USB Suspend bit in the Core Interrupt register.

2. The PHY indicates the end of the B-device session by deasserting the utmi_bvalid signal.

3. The otg core asserts the utmi_dischrgvbus signal to indicate to the PHY to speed up VBUS discharge.

4. The PHY indicates the session's end by asserting the utmi_sessend signal. This is the initial condition for SRP. The otg core requires 2 ms of SE0 before initiating SRP. For a USB 1.1 full-speed serial transceiver, the application must wait until VBUS discharges to 0.2 V after GOTGCTL.BSesVld is deasserted. This discharge time can be obtained from the transceiver vendor and varies between different transceivers.

5. The application initiates SRP by writing the Session Request bit in the OTG Control and Status register. The otg core perform data-line pulsing followed by VBUS pulsing.

6. The host detects SRP from either the data-line or VBUS pulsing, and turns on VBUS. The PHY indicates VBUS power-on by asserting utmi_vbusvalid.

7. The otg core performs VBUS pulsing by asserting utmi_chrgvbus. The host starts a new session by turning on VBUS, indicating SRP success. The otg core interrupts the application by setting the Session Request Success Status Change bit in the OTG Interrupt Status register. The application reads the Session Request Success bit in the OTG Control and Status register.

8. When the USB is powered, the otg core connects, completing the SRP process.

13.12.3 A-Device Host Negotiation Protocol

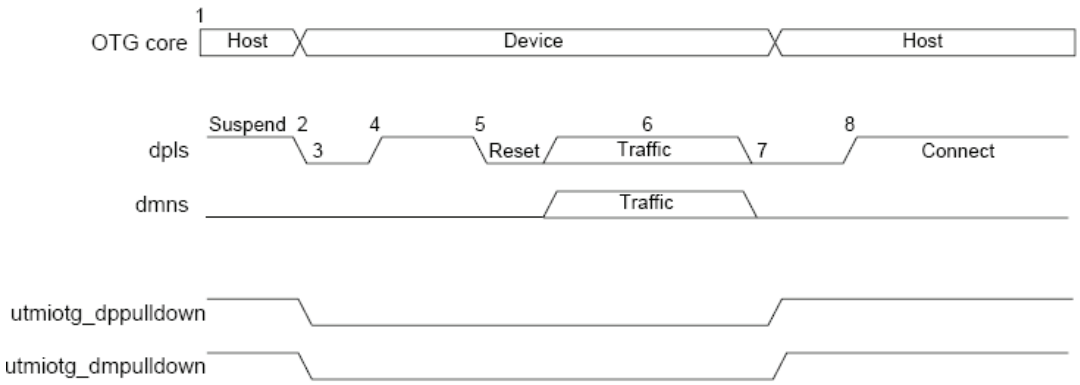


Figure 13.58 A-Device HNP

HNP switches the USB host role from the A-device to the B-device. The application must set the HNP-Capable bit in the Core USB Configuration register to enable the otg core to perform HNP as an A-device.

1. The otg core sends the B-device a SetFeature b_hnp_enable descriptor to enable HNP support. The B-device's ACK response indicates that the B-device supports HNP. The application must set Host Set HNP Enable bit in the OTG Control and Status register to indicate to the otg core that the B-device supports HNP.

2. When it has finished using the bus, the application suspends by writing the Port Suspend bit in the Host Port Control and Status register.

3. When the B-device observes a USB suspend, it disconnects, indicating the initial condition for HNP. The B-device initiates HNP only when it must switch to the host role; otherwise, the bus continues to be suspended. The otg core sets the Host Negotiation Detected interrupt in the OTG Interrupt Status register, indicating the start of HNP.

4. The otg core deasserts the utmiotg_dppulldown and utmiotg_dmpulldown signals to indicate a device role. The PHY enable the D + pull-up resistor indicates a connect for B-device. The application must read the Current Mode bit in the OTG Control and Status register to determine Device mode operation.

5. The B-device detects the connection, issues a USB reset, and enumerates the otg core for data traffic.

6. The B-device continues the host role, initiating traffic, and suspends the bus when done. The otg core sets the Early Suspend bit in the Core Interrupt register after 3 ms of bus idleness. Following this, the otg core sets the USB Suspend bit in the Core Interrupt register.

7. In Negotiated mode, the otg core detects the suspend, disconnects, and switches back to the host role. The otg core asserts the utmiotg_dppulldown and utmiotg_dmpulldown signals to indicate its assumption of the host role. The otg core sets the Connector ID Status Change interrupt in the OTG Interrupt Status register. The application must read the connector ID status in the OTG Control and Status register to determine the otg core's operation as an A-device. This indicates the completion of HNP to the application. The application must read the Current Mode bit in the OTG Control and Status register to determine Host mode operation.

8. The B-device connects, completing the HNP process.

13.12.4 B-Device Host Negotiation Protocol

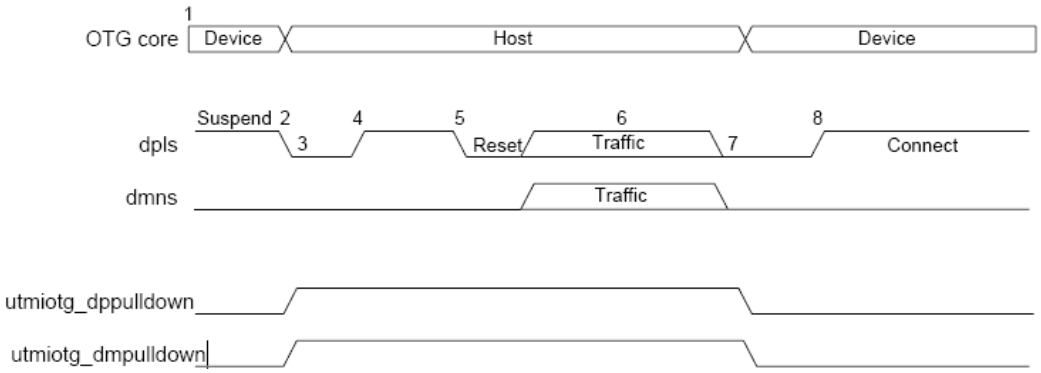


Figure 13.59 B-Device HNP

HNP switches the USB host role from B-device to A-device. The application must set the HNP-Capable bit in the Core USB Configuration register to enable the otg core to perform HNP as a B-device.

1. The A-device sends the SetFeature b_hnp_enable descriptor to enable HNP support. The otg core's ACK response indicates that it supports HNP. The application must set the Device HNP Enable bit in the OTG Control and Status register to indicate HNP support. The application sets the HNP Request bit in the OTG Control and Status register to indicate to the otg core to initiate HNP.

2. When it has finished using the bus, the A-device suspends by writing the Port Suspend bit in the Host Port Control and Status register. The otg core sets the Early Suspend bit in the Core Interrupt register after 3 ms of bus idleness. Following this, the otg core sets the USB Suspend bit in the Core Interrupt register.

3. The otg core disconnects and the A-device detects SE0 on the bus, indicating HNP. The otg core asserts the utmiotg_dppulldown and utmiotg_dmpulldown signals to indicate its assumption of the host role.

4. The A-device responds by activating its D + pull-up resistor within 3 ms of detecting SE0. The otg core detects this as a connect. The otg core sets the Host Negotiation Success Status Change interrupt in the OTG Interrupt Status register, indicating the HNP status. The application must read the Host Negotiation Success bit in the OTG Control and Status register to determine host negotiation success. The application must read the Current Mode bit in the Core Interrupt register (GINTSTS) to determine Host mode operation.

5. The otg core issues a USB reset and enumerates the A-device for data traffic.

6. The otg core continues the host role of initiating traffic, and when done, suspends the bus by writing the Port Suspend bit in the Host Port Control and Status register.

7. In Negotiated mode, when the A-device detects a suspend, it disconnects and switches back to the host role. The otg core deasserts the utmiotg_dppulldown and utmiotg_dmpulldown signals to indicate the assumption of the device role. The application must read the Current Mode bit in the Core Interrupt (GINTSTS) register to determine the Host mode operation.

8. The otg core connects, completing the HNP process.

13.12.5 Clock Gating

You can use clock gating to reduce power consumption when the USB is suspended or the session is not valid. The PHY turns off the PHY clock for as long as the core asserts the suspend_n signal to the PHY. The AHB clock to the HSOTG internal modules can also be gated by writing to the Gate hclk bit in the Power and Clock Gating Control register. The following sections show the procedures you must follow to use the clock gating feature.

13.12.5.1 Host Mode Suspend and Resume With Clock Gating

1. The application sets the Port Suspend bit in the Host Port CSR, and the core drives a USB suspend.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, the core gates the hclk (hclk_gated) to AHB-domain modules other than the BIU.

3. The core remains in Suspend mode.

4. The application clears the Gate hclk and Stop PHY Clock bits, and the PHY clock is generated.

5. The application sets the Port Resume bit, and the core starts driving Resume signaling.

6. The application clears the Port Resume bit after at least 20 ms.

7. The core is in normal operating mode.

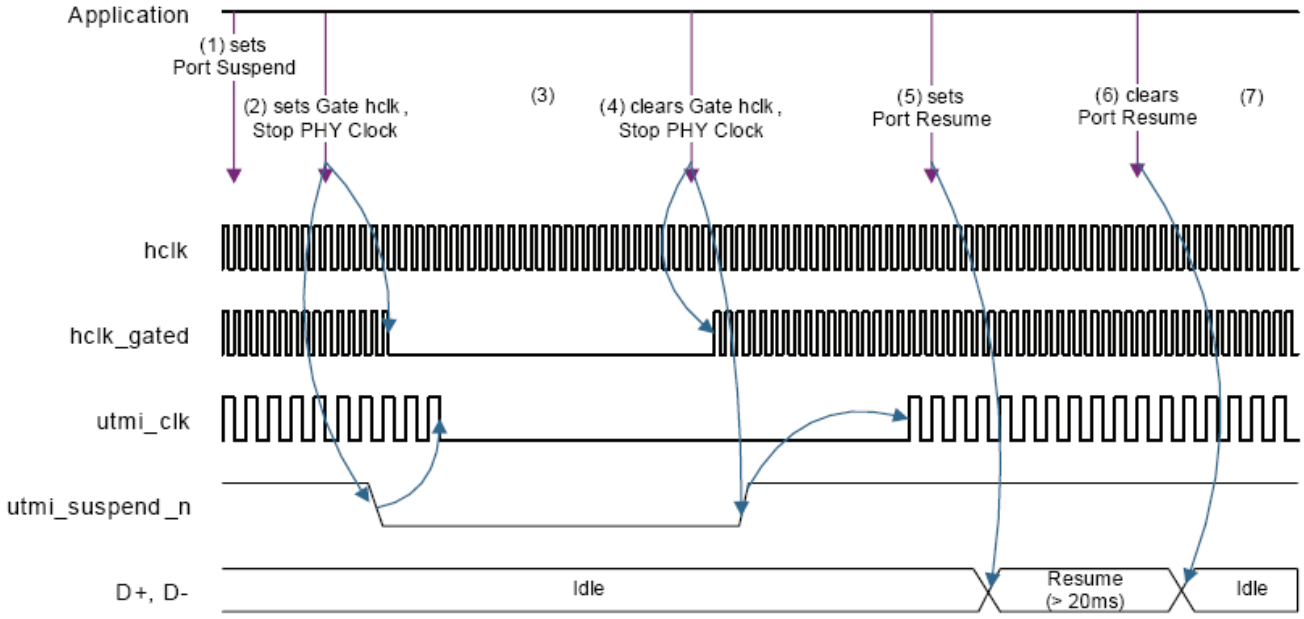


Figure 13.60Host Mode Suspend and Resume With Clock Gating

13.12.5.2 Host Mode Suspend and Remote Wakeup With Clock Gating

Sequence of operations

1. The application sets the Port Suspend bit in the Host Port CSR, and the core drives a USB suspend.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

3. The core remains in Suspend mode.

4. The Remote Wakeup signaling from the device is detected. The core deasserts the suspend_n signal to the PHY to generate the PHY clock. The core generates a Remote Wakeup Detected interrupt.

5. The application clears the Gate hclk and Stop PHY Clock bits. The core sets the Port Resume bit.

6. The application clears the Port Resume bit after at least 20 ms.

7. The core is in normal operating mode.

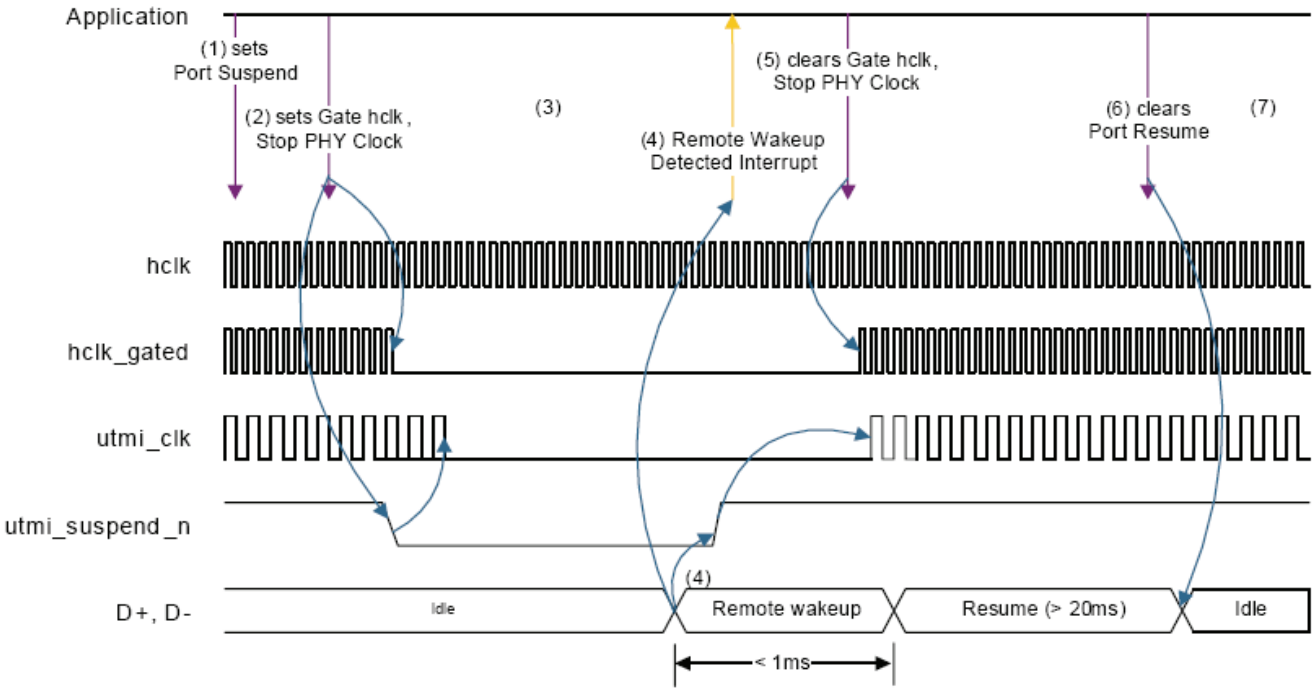


Figure 13.61Host Mode Suspend and Remote Wakeup With Clock Gating

13.12.5.3 Host Mode Session End and Start With Clock Gating

Sequence of operations

1. The application sets the Port Suspend bit in the Host Port CSR, and the core drives a USB suspend.

2. The application clears the Port Power bit. The core turns off VBUS.

3. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

4. The core remains in Low-Power mode.

5. The application clears the Gate hclk bit and the application clears the Stop PHY Clock bit to start the PHY clock.

6. The application sets the Port Power bit to turn on VBUS.

7. The core detects device connection and drives a USB reset.

8. The core is in normal operating mode.

13.12.5.4 Host Mode Session End and SRP With Clock Gating

Sequence of operations

1. The application sets the Port Suspend bit in the Host Port CSR, and the core drives a USB suspend.

2. The application clears the Port Power bit. The core turns off VBUS.

3. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating

Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

4. The core remains in Low-Power mode.

5. SRP (data line pulsing) from the device is detected. The core deasserts the suspend_n signal to the PHY to generate the PHY clock. An SRP Request Detected interrupt is generated.

6. The application clears the Gate hclk bit and the Stop PHY Clock bit.

7. The core sets the Port Power bit to turn on VBUS.

8. The core detects device connection and drives a USB reset.

9. The core is in normal operating mode.

13.12.5.5 Device Mode Suspend and Resume With Clock Gating

Sequence of operations

1. The core detects a USB suspend and generates a Suspend Detected interrupt.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

3. The core remains in Suspend mode.

4. The Resume signaling from the host is detected. The core deasserts the suspend_n signal to the PHY to generate the PHY clock. A Resume Detected interrupt is generated.

5. The application clears the Gate hclk bit and the Stop PHY Clock bit.

6. The host finishes Resume signaling.

7. The core is in normal operating mode.

13.12.5.6 Device Mode Suspend and Remote Wakeup With Clock Gating

Sequence of operations

1. The core detects a USB suspend and generates a Suspend Detected interrupt.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

3. The core remains in Suspend mode.

4. The application clears the Gate hclk bit and the Stop PHY Clock bit.

5. The application sets the Remote Wakeup bit in the Device Control register, the core starts driving Remote Wakeup signaling.

6. The host drives Resume signaling.

7. The core is in normal operating mode.

13.12.5.7 Device Mode Session End and Start With Clock Gating

Sequence of operations

1. The core detects a USB suspend, and generates a Suspend Detected interrupt. The host turns off VBUS.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

3. The core remains in Low-Power mode.

4. The new session is detected (bsessvld is high). The core deasserts the suspend_n signal to the PHY to generate the PHY clock. A New Session Detected interrupt is generated.

5. The application clears the Gate hclk and Stop PHY Clock bits.

6. The core detects USB reset.

7. The core is in normal operating mode

13.12.5.8 Device Mode Session End and SRP With Clock Gating

Sequence of operations

1. The core detects a USB suspend, and generates a Suspend Detected interrupt. The host turns off VBUS.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

3. The core remains in Low-Power mode.

4. The application clears the Gate hclk and Stop PHY Clock bits.

5. The application sets the SRP Request bit, and the core drives data line and VBUS pulsing.

6. The host turns on VBUS, detects device connection, and drives a USB reset.

7. The core is in normal operating mode.

13.13 Miscellaneous Topics

13.13.1 Data FIFO RAM Allocation

If Dynamic FIFO Sizing is enabled in the core, the External RAM must be allocated among different FIFOs in the core before any transactions can start. The application must follow this procedure every time it changes core FIFO RAM allocation.

The application must allocate data RAM per FIFO based on the AHB's operating frequency, the PHY Clock frequency, the available AHB bandwidth, and the performance required on the USB. Based on the above mentioned criteria, the application must provide a table as described below with RAM sizes for each FIFO in each mode.

13.13.1.1 Device Mode

[Dedicated Tx FIFO Operation]

When allocating data RAM for FIFOs in Device mode when dedicated TX FIFO is used, keep in mind these factors:

1. Receive FIFO RAM Allocation:

- RAM for SETUP Packets: 4 * n + 6 locations must be Reserved in the receive FIFO to receive up to n SETUP packets on control endpoints, where n is the number of control endpoints the device core supports. The core does not use these locations, which are Reserved for SETUP packets, to write any other data.
- 1 location for Global OUT NAK
- Status information is written to the FIFO along with each received packet. Therefore, a minimum space of (Largest Packet Size / 4) + 1 must be allotted to receive packets. If a high-bandwidth endpoint is enabled, or multiple isochronous endpoints are enabled, then at least two (Largest Packet Size / 4) + 1 spaces must be allotted to receive back-to-back packets. Typically, two (Largest Packet Size / 4) + 1 spaces are recommended so that when the previous packet is being transferred to AHB, the USB can receive the subsequent packet. If AHB latency is high, you must allocate enough space to receive multiple packets. This is critical to prevent dropping any isochronous packets.
- Along with each endpoint's last packet, transfer complete status information is also pushed to the FIFO. Typically, one location for each OUT endpoint is recommended.

2. Transmit FIFO RAM Allocation:

- The minimum RAM space required for each IN Endpoint Transmit FIFO is the maximum packet size for that particular IN endpoint.
- The more space allocated in the transmit IN Endpoint FIFO results in a better performance on the USB and can hide the latencies on the AHB.

FIFO Name	Data RAM Size
Receive dta FIFO	rx_fifo_size. This must include RAM for setup packets, OUT endpoint control information and data OUT packets, as mentioned earlier.
Transmit FIFO 0	Tx_fifo_size[0]
Transmit FIFO 1	Tx_fifo_size[1]
….	…
Transmit FIFO i	Tx_fifo_size[i]

With this information at hand, the following registers must be programmed as follows:

1. Receive FIFO Size Register (GRXFSIZ)

- GRXFSIZ.Receive FIFO Depth = rx_fifo_size;

2. Device IN Endpoint Transmit FIFO0 Size Register (GNPTXFSIZ)

- GNPTXFSIZ.non-periodic Transmit FIFO Depth = tx_fifo_size[0];
- GNPTXFSIZ.non-periodic Transmit RAM Start Address = rx_fifo_size;

3. Device IN Endpoint Transmit FIFO#1 Size Register (DIEPTXF1)

- DIEPTXF1. Transmit RAM Start Address = GNPTXFSIZ.FIFO0 Transmit RAM Start Address + tx_fifo_size[0];

4. Device IN Endpoint Transmit FIFO#2 Size Register (DIEPTXF2)

- DIEPTXF2.Transmit RAM Start Address = DIEPTXF1.Transmit RAM Start Address + tx_fifo_size[1];

5. Device IN Endpoint Transmit FIFO#i Size Register (DIEPTXFi)

- DIEPTXFm.Transmit RAM Start Address = DIEPTXFi-1.Transmit RAM Start Address + tx_fifo_size[i-1];

6. The transmit FIFOs and receive FIFO must be flushed after the RAM allocation is done, for the proper functioning of the FIFOs.

- GRSTCTL.TxFNum = 5'h10
- GRSTCTL.TxFFlush = 1'b1
- GRSTCTL.RxFFlush = 1'b1
- The application has to wait until the TxFFlush bit and the RxFFlush bits are cleared, before performing any operation on the core.

[Dedicated Tx FIFO operation with Thresholding]

1. Receive FIFO RAM allocation

- RAM for Setup Packets: 7*n + 6 locations have to be Reserved in the receive FIFO, to receive up to "n" setup packets on control endpoints, where "n" is the number of control endpoints supported by the device core. These locations Reserved for Setup Packets are not used by the core, to write any other data.
- 1 location for Global OUT NAK
- It is recommended to have an Rx FIFO space for two thresholds. Along with each received threshold, a status information is also written in to the FIFO. With the last threshold of a packet, two status DWORDs are written into the FIFO. With the last packet of a transfer, transfer complete status information is written into the FIFO. So worst case , a minimum of 2*(Rx_threshold length/4 + 4) space is needed to be allocated to receive a packet.

2. Transmit FIFO RAM Allocation:

- The minimum RAM space required for each IN Endpoint Transmit FIFO is min(2*Tx_threshold_length, endpoint_max_pkt_size).
- The more space allocated in the transmit IN Endpoint FIFO results in a better performance on the USB and can hide the latencies on the AHB.

If high AHB latencies results in underrun error very often, then there is a possibility that the Host might disable this endpoint because of Errors in the packet (typically when there is error in 3 consecutive packets). So thresholding must be enabled only when the AHB latency is not very high.

13.13.1.2 Host Mode

With this information at hand, the following registers must be programmed as follows:

1. Receive FIFO Size Register (GRXFSIZ)

- GRXFSIZ.RxFDep = rx_fifo_size;

2. Non-periodic Transmit FIFO Size Register (GNPTXFSIZ)

- GNPTXFSIZ.NPTxFDe = tx_fifo_size[0];
- GNPTXFSIZ.NPTxFStAddr = rx_fifo_size;

3. Host Periodic Transmit FIFO Size Register (HPTXFSIZ)

- HPTXFSIZ.PTxFSize = tx_fifo_size[1];
- HPTXFSIZ.PTxFStAddr = GNPTXFSIZ.NPTxFStAddr + tx_fifo_size[0];

4. The transmit FIFOs and receive FIFO must be flushed after RAM allocation for proper FIFO function.

- GRSTCTL.TxFNum = 5'h10
- GRSTCTL.TxFFlush = 1'b1
- GRSTCTL.RxFFlush = 1'b1
- The application must wait until the TxFFlush bit and the RxFFlush bits are cleared before performing any operation on the core.

13.13.1.3 Calculating the Total FIFO Size for OTG

[Dedicated FIFO Mode with No Thresholding]

The otg RxFIFO is shared between the host and device. The Host TxFIFOs are also shared with Device IN endpoint TxFIFOs 0 through n.

There are three ways to calculate the total FIFO size. The total FIFO size will depend on whether you support high-bandwidth transfers with high AHB latency.

Method 1

Use this method if you are using the following conditions:

- Minimum FIFO depth allocation
- No support for high-bandwidth endpoints
- The FIFO must equal at least one MaxPacketSize (MPS).

Device RxFIFO = (4 * number of control endpoints + 6) + ((largest USB packet used / 4) + 1 for status information) + (2 * number of OUT endpoints) + 1 for Global NAK

Note : Include the control OUT endpoint in the "number of OUT endpoints."

Host RxFIFO = (largest USB packet used / 4) + 1 for status information + 1 transfer complete

Host Non-Periodic TxFIFO = largest non-periodic USB packet used / 4

Host Periodic TxFIFO = largest periodic USB packet used / 4

Device IN Endpoint TxFIFOs (a separate FIFO is allocated to each IN endpoint) = IN Endpoints Max packet Size / 4

OTG Total RAM = (Device RxFIFO or Host RxFIFO; choose the largest one) + {((Host Non-Periodic TxFIFO + Host peiodic TxFIFO) OR Device IN Endpoint TxFIFO #0 + #1 + #2 + #n); choose the largest one}

Example

The maximum packet size (MPS) is 1,024 bytes for a periodic USB packet and 512 bytes for a non-periodic USB packet. There are three OUT endpoints, three IN endpoints, and one control endpoints. One of the IN endpoints is isochronous.

Device RxFIFO = (4 * 1 + 6) + ((1,024 / 4) +1) + (2 * 4) + 1 = 276

Host RxFIFO = ((1,024 / 4) + 1) + 1 = 258

Host Non-Periodic TxFIFO = (512 / 4) = 128

Host Periodic TxFIFO = (1,024 / 4) = 256

Device IN Endpoint TxFIFO:

FIFO #0= (64/ 4) = 16 (Assuming this is used for EP0)
FIFO #1= (512 / 4) = 128
FIFO #2 = (512/4) = 128
FIFO #3= (1,024 / 4) = 256 (Assuming this is used for Isochronous).

OTG Total RAM = max(276,258) + max(384,528) = 804.

Method 2

Use this method if you are using the recommended minimum FIFO depth allocation with support for high-bandwidth endpoints. This FIFO allocation enables the core to transfer a packet on the USB while the previous (next) packet is simultaneously transferred to the AHB.

This FIFO allocation improves the core's performance .

Device RxFIFO = (4 * number of control endpoints + 6) + 2 * ((largest USB packet used / 4) + 1) + (2 * number of OUT endpoints) + 1

Note: Include the control OUT endpoint in the "number of OUT endpoints."

Host RxFIFO = 2 * ((largest USB packet used / 4) + 1 + 1)

Host Non-Periodic TxFIFO = 2 * (largest non-periodic USB packet used / 4)

Host Periodic TxFIFO = 2 * (largest periodic USB packet used / 4)

Device IN Endpoint-Specific TxFIFOs (a separate FIFO is allocated to each endpoint) = 2*(max_pkt_size for the endpoint)/4.

OTG Total RAM = (Device RxFIFO or Host RxFIFO; choose the largest one) + {((Host Non-Periodic TxFIFO + Host peiodic TxFIFO) OR Device IN Endpoint TxFIFO #0 + #1 + #2 + #n); choose the largest one }

Example

The MPS is 1,024 bytes for a periodic USB packet and is 512 bytes for a non-periodic USB packet. There are three OUT endpoints, three IN endpoints, and one control endpoint. .One of the IN endpoint is Isochronous and the maximum number of periodic data packets per transfer for Endpoint 3 is 3.

Device RxFIFO = (4 * 1 + 6) + 2 * ((1,024 / 4) + 1) + (2 * 4) + 1 = 533

Host RxFIFO = 2 * ((1,024 / 4) + 1 + 1) = 516

Host Non-Periodic TxFIFO = 2 * (512 / 4) = 256

Host Periodic TxFIFO = 2 * (1,024 / 4) = 512

Device IN Endpoint TxFIFO:

FIFO #0 = 2*(64 / 4) = 32(Assuming used for Control Endpoint)

FIFO #1 = 2*(512 / 4) = 256
FIFO #2 = 2*(512 / 4) = 256
FIFO #3 = 2*(1024 / 4) = 256 * 2= 512 (Assuming used for High Bandwidth Endpoint)

OTG total RAM = max(533,516) + max(768,1056)= 1589

Method 3

Use this method if you are using the recommended FIFO depth allocation that supports high-bandwidth endpoints and high AHB latency.

Notes: x = (AHB latency + time to transfer largest packet on AHB) / time to transfer largest packet on USB. The value of x is an integer. Any fractional value is rounded to the nearest integer. For example: x = 20 µs / 17,039 µs = 1.17 µs = 2 µs.

Device RxFIFO = (4 * number of control endpoints + 6) + (x + 1) * ((largest USB packet used / 4) + 1) + (2 * number of OUT endpoints) + 1

Note : Include the control OUT endpoint in the “number of OUT endpoints.”

Host RxFIFO = (x + 1) * ((largest USB packet used / 4) + 1 + 1)

Host Non-Periodic TxFIFO = (x + 1) * (largest non-periodic USB packet used / 4)

Host Periodic TxFIFO = (x + 1) * (largest periodic USB packet used / 4)

Device IN Endpoint-Specific TxFIFOs (a separate FIFO is allocated to each endpoint) = (x+1)*(max_pkt_size for the endpoint)/4e

OTG Total RAM = (Device RxFIFO or Host RxFIFO; choose the largest one) + {((Host Non-Periodic TxFIFO + Host peiodic TxFIFO) OR Device IN Endpoint TxFIFO #0 + #1 + #2 + #n); choose the largest one}

For example, the otg core is operating as a device and receives three isochronous backto-back packets (high bandwidth). The RxFIFO is configured for only 2 MPS (2 kB). On the USB, it takes 2.08 ns (high speed) to transfer 1 bit of data. To transfer 1,024 bytes (8,192 bits), it takes approximately 17,039 µs (2.08 ns * 8,192) to complete the packet transfer from the USB to RxFIFO. After the full packet is in the RxFIFO (assume you are using DMA mode), the DMA begins to transfer the data packet from the RxFIFO to system memory. The AHB latency plus the time to transfer 1,024 bytes on the AHB is 20 µs.

A break-down of the sequence is as follows:

1. First isochronous packet from USB to RxFIFO = 17,039 µs.

2. DMA begins to fetch data from RxFIFO to memory; time = 0.

3. Second isochronous packet from USB to RxFIFO = 17,039 µs.

4. Because there is an extra 1 MPS RxFIFO, the core can receive the next packet regardless of whether the DMA has completed the transfer on the AHB.

5. Third isochronous packet from USB to RxFIFO = 17,039 µs.

If the DMA transfer on the AHB for the first packet is still not complete after 17,039 µs, no FIFO space is available for the third transaction. In this case, the AHB latency is 20 µs; therefore, the transfer of the first packet still has not completed. The OTG core does not issue an interrupt for the transfer completed for the third packet and the transfer times out on the USB side. To compensate for the high AHB latency, you can configure the RxFIFO for 3 more MPS.

Example

The MPS for a periodic USB packet is 1,024 bytes and 512 bytes for a non-periodic USB packet. There are three OUT endpoints, three IN endpoints, and one control endpoint. One of the IN Endpoint is Isochronous and the maximum number of periodic data packets per transfer for Endpoint 3 is 3. The AHB latency plus the time to transfer 1,024 bytes on the AHB is 20 µs.

Device RxFIFO = (4 * 1 Host Periodic TxFIFO = (2 + 1) * (1,024 / 4) = 768

Host RxFIFO = (2 + 1) * ((1,024 / 4) + 1 + 1) = 774

Host Non-Periodic TxFIFO = (2 + 1) * (512 / 4) = 384

Device IN ENdpoint TxFIFO:

FIFO #0 = (2+1) * (64/4) = 48 (Assuming to be used for Control Endpoint)
FIFO #1 = (2+1) * (512 / 4) = 384
FIFO #2 = (2+1) * (512 / 4) = 384
FIFO #3 = 3* (1,024 / 4) = 256 * 3= 768 (Assuming to be used for Isochronous Endpoint)

OTG total RAM = max(790,774) + max(1152, 1584) = 2374+ 6) + (2 + 1) * ((1,024 / 4) +1) + (2 * 4) + 1 =790

[Dedicated FIFO Mode with thresholding]

The main intention of threshold support are the following

- To have a smaller FIFO size
- To have faster DMA response.

Thresholding is supported only in device mode. So if your device does not have many IN endpoints (not more than 2) OR if your calculated total device FIFO depth without thresholding is not more than the required host FIFO depth, then you are not going to save FIFO space much by enabling thresholding.

It is strongly recommended that you enable dynamic FIFO sizing when thresholding is enabled.

Device RxFIFO = (7 * number of control endpoints + 6) + 2 * ((min(largest USB packet used, Rx_threshold_length) / 4) + 4) + 1 for Global NAK.

Host RxFIFO = (largest USB packet used / 4) + 1 for status information + 1 transfer complete

Host Non-Periodic TxFIFO = largest non-periodic USB packet used / 4

Host Periodic TxFIFO = largest periodic USB packet used / 4

Device IN Endpoint TxFIFOs (a separate FIFO is allocated to each IN endpoint) = min(2 x Transmit Threshold size, IN Endpoints Max packet Size) / 4

OTG Total RAM = (Device RxFIFO or Host RxFIFO; choose the largest one) + {((Host Non-Periodic TxFIFO + Host peiodic TxFIFO) OR Device IN Endpoint TxFIFO #0 + #1 + #2 + #n); choose the largest one}

Example

The maximum packet size (MPS) is 1,024 bytes for a periodic USB packet and 512 bytes for a non-periodic USB packet. There are three OUT endpoints, three IN endpoints, and one control endpoints. One of the IN endpoints is isochronous. The Rx threshold size is 128 bytes and the Tx threshold size is also 128 bytes.

Device RxFIFO = (7 * 1 + 6) + 2* ((128 / 4) +4) + 1 = 86

Host RxFIFO = ((1,024 / 4) + 1) + 1 = 258

Host Non-Periodic TxFIFO = (512 / 4) = 128

Host Periodic TxFIFO = (1,024 / 4) = 256

Device IN Endpoint TxFIFO:

FIFO #0= (64/ 4) = 16 (Assuming this is used for EP0)
FIFO #1= (128/ 4) = 32
FIFO #2 = (128/4) = 32
FIFO #3= (128 / 4) = 32(Asuming this is used for Isochronous).

OTG Total RAM = (Device RxFIFO or Host RxFIFO; choose the largest one) + ((Host Non-Periodic TxFIFO + Host periodic TxFIFO) OR Device IN Endpoint TxFIFO #0 + #1 + #2 + #n; choose the largest one)

OTG_TOTAL_RAM = max(258,86) + max((128+256),(16+32+32+32)) = 258 + 384 = 642.

13.13.2 Dynamic FIFO Allocation

The application can change the RAM allocation for each FIFO during the operation of the core.

Host Mode

In Host mode, before changing FIFO data RAM allocation, the application must determine the following.

- All channels are disabled
- All FIFOs are empty

Once these conditions are met, the application can reallocate FIFO data RAM as explained in "Data FIFO RAM Allocation".

After reallocating the FIFO data RAM, the application must flush all FIFOs in the core using the GRSTCTL.TxFIFO Flush and GRSTCTL.RxFIFO Flush fields. Flushing is required to reset the pointers in the FIFOs for proper FIFO operation after reallocation.

Device Mode

In Device mode, before changing FIFO data RAM allocation, the application must determine the following.

- All IN and OUT endpoints are disabled
- NAK mode is enabled in the core on all IN endpoints
- Global OUT NAK mode is enabled in the core
- All FIFOs are empty

Once these conditions are met, the application can reallocate FIFO data RAM as explained in "Data FIFO RAM Allocation". When NAK mode is enabled in the core, the core responds with a NAK handshake on all tokens received on the USB, except for SETUP packets.

After the reallocating the FIFO data RAM, the application must flush all FIFOs in the core using the GRSTCTL.TxFIFO Flush and GRSTCTL.RxFIFO Flush fields. Flushing is required to reset the pointers in the FIFOs for proper FIFO operation after reallocation.

13.13.3 Core Interrupt Handler

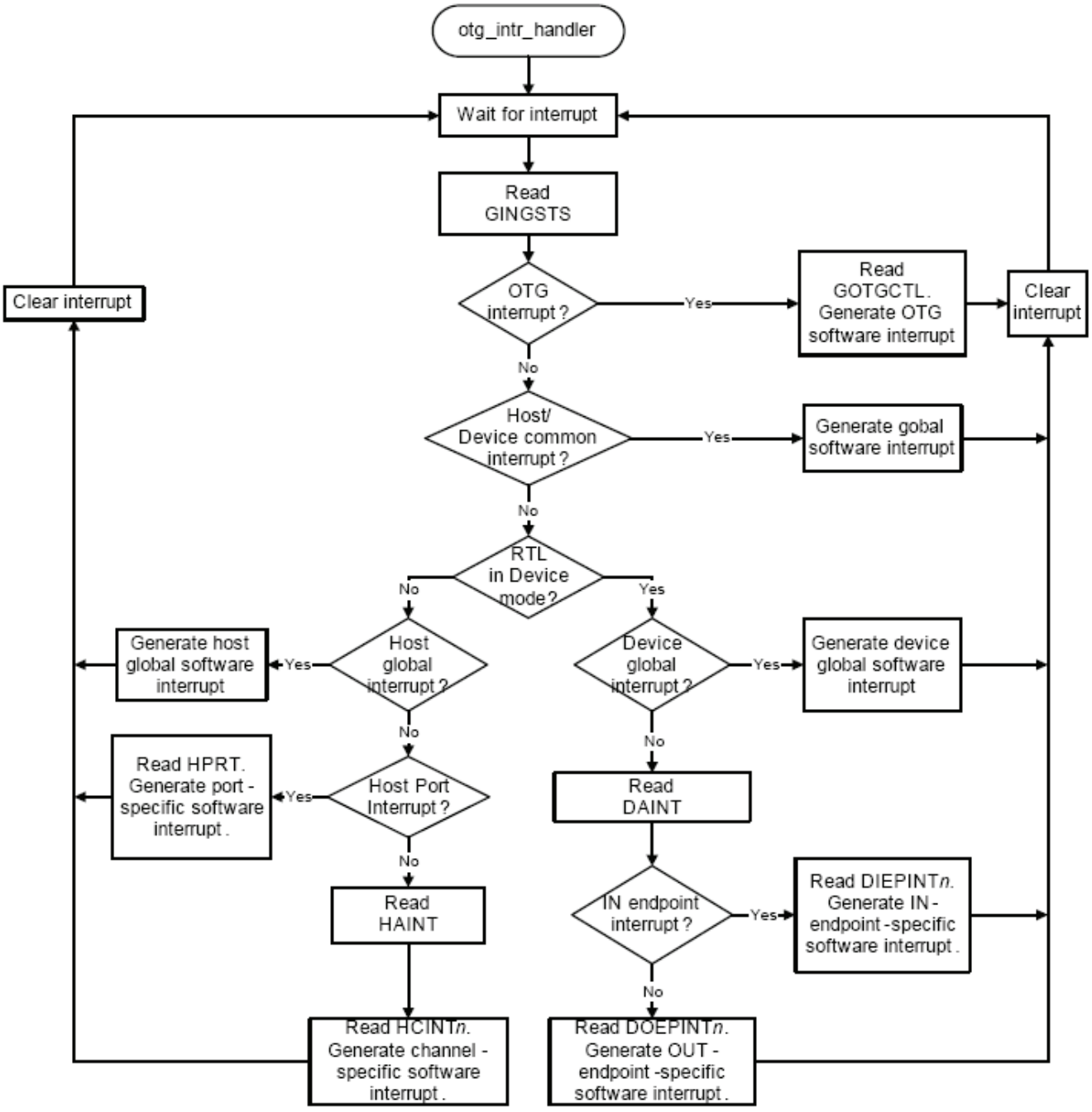


Figure 13.62 Core Interrupt Handler

14 I2C Controller

14.1 Functional Description

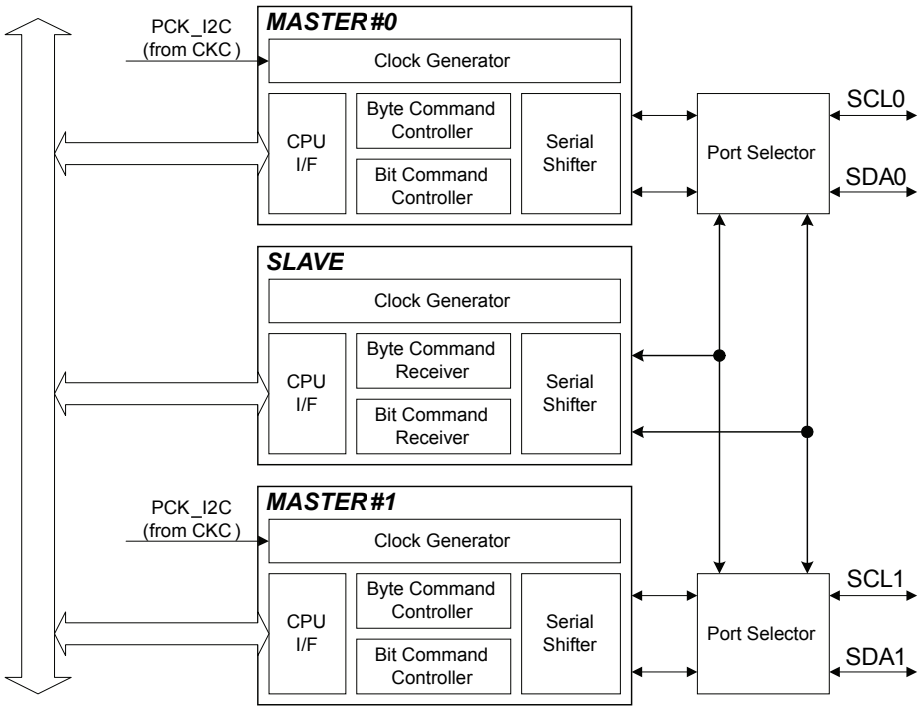


Figure 14.1 I2C Block Diagram

The I2C controller in the NVS2310 has two masters and one slave controller. The each master can control the dedicated I2C bus and the slave controller can select the one of two I2C buses. For each I2C bus can be selected by the register.

14.2 I2C Slave Function

- Supports Fast mode (400Kbps)
- Supports wait state by clock stretching
- 4byte TX FIFO, 4byte RX FIFO and 8byte Buffer

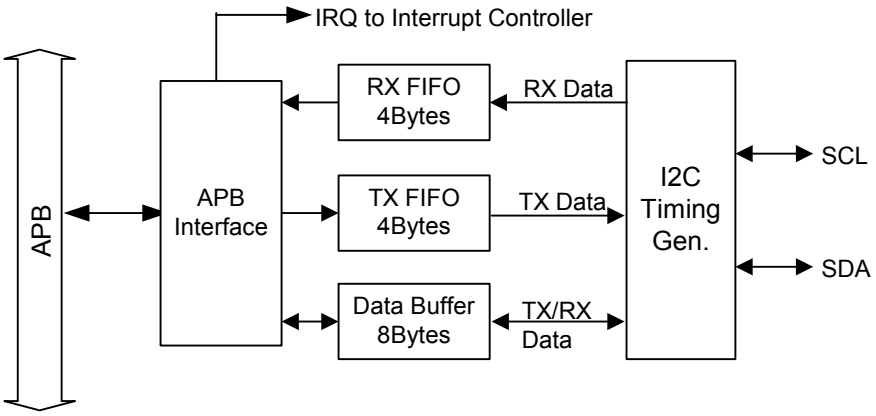
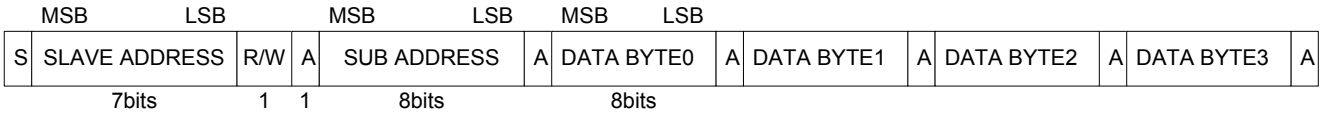


Figure 14.2 I2C Slave Block Diagram

The following figure and table show an example format for accessing Buffers and FIFOs in the I2C slave core.



S: Start Condition
R/W: Read (1) or Write(0)
A: Acknowledge from/to the core

SUB ADDRESS[7:0]	DATA[7:0] Source / Destination	Note
0x00 ~ 0x03	MB0 (Data Buffer 0) Byte 0 ~ 3	
0x04 ~ 0x07	MB1 (Data Buffer 1) Byte 0 ~ 3	
0x08 ~ 0x7F	Reserved	NACK is sent to a master
0x80 ~ 0xFF	RX/TX FIFO	All addresses are directed to the RX/TX FIFO

14.3 Related Blocks

After the signals are enabled, PCK_I2C (the main clock of I2C) must be enabled and configured to the proper frequency.

For internal synchronization, the APB clock frequency must be faster than the PCK_I2C frequency.

$f_{PCK\_I2C} \leq f_{HCLK} / 4.0$

14.4 Register Description

Table 14.1 I2C Register Map (Base Address = 0xB0109n00 n*=0, 1, 2)

Ch	Name	Addr. Offset	RW	Reset	Description
Master 0	PRES	0x00	R/W	0xFFFF	Clock Prescale register
	CTRL	0x04	R/W	0x0000	Control Register
	TXR	0x08	W	0x0000	Transmit Register
	CMD	0x0C	W	0x0000	Command Register
	RXR	0x10	R	0x0000	Receive Register
	SR	0x14	R	0x0000	Status Register
	TIME	0x18	R/W	0x0000	Timing Control Register
Master 1	PRES	0x40	R/W	0xFFFF	Clock Prescale register
	CTRL	0x44	R/W	0x0000	Control Register
	TXR	0x48	W	0x0000	Transmit Register
	CMD	0x4C	W	0x0000	Command Register
	RXR	0x50	R	0x0000	Receive Register
	SR	0x54	R	0x0000	Status Register
	TIME	0x58	R/W	0x0000	Timing Control Register
Slave	PORT	0x80	R/W	-	Data Access port (TX/RX FIFO)
	CTL	0x84	R/W	0x00000000	Control register
	ADDR	0x88	W	0x00000000	Address register
	INT	0x8C	W	0x00000000	Interrupt Enable Register
	STAT	0x90	R	0x00000000	Status Register
	MBF	0x9C	R/W	0x00000000	Buffer Valid Flag
	MB0	0xA0	R/W	0x00000000	Data Buffer 0 (Byte 3 ~ 0)
	MB1	0xA4	R/W	0x00000000	Data Buffer 1 (Byte 7 ~ 4)
Status	IRQSTR	0xC0	R	0x00000000	IRQ Status Register

Table 14.2 I2C Group IRQ, DMA register Map

Name	Address	RW	Reset	Description
I2C_IRQ	0xB0109300	R	3'b000	IRQ Status of I2C Channel2 Group
DREQ_SEL	0xB0109400	R/W	1'b0	DMA Request Selection of I2C Group

* n = Channel Number 0 ~ 2

Prescale Register (PRES) 0xB0109n00, 0xB0109n40

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Clock Prescale data															

Field	Name	RW	Reset	Description
15-0	CPD	RW	0x0000FFFF	Prescale the SCL clock

This register is used to prescale the SCL clock line. Due to the structure of the I2C interface, the core uses a 5*SCL clock internally. The prescale register must be programmed to this 5*SCL frequency (minus 1). Change the value of the prescale register only when 'EN' bit is cleared.

Example :

CLK Input frequency = 8MHz , Desired SCL frequency = 100KHz

Prescale = (8MHz / 100KHz) – 1 = 15

Control Register (CTR) 0xB0109n04, 0xB0109n44

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								EN	IEN	MOD	RESERVED				

Field	Name	RW	Reset	Description
31-8	Reserved	-	-	
7	EN	RW	1'b0	I2C Core enable bit 1'b0=Disable 1'b1=Enabled
6	IEN	RW	1'b0	I2C Core interrupt enable bit 0'b0=Disable 1'b1=Enable
5	MOD	RW	1'b0	I2C Data Width 0'b0=8bit Mode 1'b1=16bit Mode
4-0	Reserved	-	-	

Transmit Register (TXR) 0xB0109n08, 0xB0109n48

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Transmit Data															

Field	Name	RW	Reset	Description
16-0	TD	W	0x00000000	16bit mode Transmit data

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Transmit Data (8bit mode)							

Field	Name	RW	Reset	Description
7-0	TD	W	0x00000000	8bit mode Transmit data

When CTRL[5] is set, in case of 16Bit Mode is selected, Transmit Data bit width become 16 bit. Default mode is 8bit mode.

Command Register (CMD)

0xB0109n0C, 0xB0109n4C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								STA	STO	RD	WR	ACK	RESERVE		IACK

Field	Name	RW	Reset	Description
31-8	Reserved	-	-	-
7	STA	W	1'b0	Start Condition Generation 1'b0=Disabled 1'b1=Enabled
6	STO	W	1'b0	Stop Condition Generation 1'b0=Disabled 1'b1=Enabled.
5	RD	W	1'b0	Read From Slave 1'b0=Disabled 1'b1=Enabled
4	WR	W	1'b0	Write to Slave 1'b0=Disabled 1'b1=Enabled
3	ACK	W	1'b0	Sent ACK 1'b0=Disabled 1'b1=Enabled
2-1	Reserved	-		-
0	IACK	W	1'b0	Interrupt Acknowledge Clear a pending interrupt

Receive Register (RXR)

0xB0109n10, 0xB0109n50

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Receive Data															

Field	Name	RW	Reset	Description
15-0	RD	R	0x00000000	16bit mode Receive data

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								Receive Data (8bit mode)							

Field	Name	RW	Reset	Description
7-0	RD	R	0x00000000	8bit mode Receive data

When CTRL[5] is set, in case of 16Bit Mode is selected, Transmit Data bit width become 16 bit. Default mode is 8bit mode.

Status Register (SR) 0xB0109n14, 0xB0109n54

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RxACK	BUSY	AL				TIP	IF

Field	Name	RW	Reset	Description
31-8	Reserved	-	-	-
7	RxACK	R	1'b0	Received acknowledge from slave 1'b0 = Acknowledge received 1'b1= No Acknowledge received
6	BUSY	R	1'b0	I2C Bus Busy 1'b0 ='0' after STOP signal detected 1'b1='1' after START signal detected
5	AL	R	1'b0	Arbitration lost 1'b0 = The core does not lose arbitration 1'b1= The core loses arbitration Arbitration is lost when a STOP signal is detected, but non requested master drives SDA high, but SDA is low
4-2	Reserved	-	-	-
1	TIP	R	1'b0	Transfer in progress 1'b0 = Transfer Complete 1'b1= Transferring Data
0	IF	R	1'b0	Interrupt Flag 1'b0 =none 1'b1= Interrupt is pending

Timing Register (TR) 0xB0109n18, 0xB0109n58

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RC								-		CKSEL	FC				

Field	Name	RW	Reset	Description
31-16	Reserved	-	-	-
15-8	RC	R/W	8'b00000000	Recovery time counter load value "0" disables recovery time counter. The recovery time counter is enabled and loaded with RC[7:0] whenever a STOP condition is issued by the core. Execution of a new command written to CMD register is delayed until the counter is expired. Actual wait time = (I2CCLK period) * PRES[15:0] * 5 * RC[7:0]
7-6	Reserved	-	-	-
5	CKSEL	R/W	1'b0	Clock Source Select 1'b0= I2CCLK from Clock controller 1'b1= PCLK (HCLK) divided by 2 Recommended if PCLK is not variable during system operation.
4-0	FC	-	5'b0	Noise filter counter load value "0" disables noise filter. SCL and SDA inputs are checked for stability until the counter is expired.

Data Port (DPORT) 0xB0109n80

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-					TXVC			-					RXVC		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-								PORT							

Field	Name	RW	Reset	Description
31-27	Reserved	-	-	-
26-24	TXVC	R/W	Unknown	TX buffer valid entry count
23-19	Reserved	-	-	-
18-16	RXVC	R/W	Unknown	RX buffer valid entry count
15-8	Reserved	-	-	-
7-0	PORT	R/W	Unknown	TX / RX FIFO access port

Master Write Cycle to RX FIFO

	MSB LSB			MSB LSB		MSB LSB		MSB LSB		MSB LSB		MSB LSB	
S	WRITE SLAVE ADDRESS	W(0)	A	WRITE 1XXXXXXXb	A	WRITE DATA BYTE0	A	WRITE DATA BYTE1	A	WRITE DATA BYTE2	A	WRITE DATA BYTE3	A
	7bits	1	1	8bits		8bits							

Master Read Cycle to TX FIFO

	MSB LSB			MSB LSB		MSB LSB		MSB LSB		MSB LSB		MSB LSB	
S	WRITE SLAVE ADDRESS	R(1)	A	always read as 11111111b	A	READ DATA BYTE0	A	READ DATA BYTE1	A	READ DATA BYTE2	A	READ DATA BYTE3	A
	7bits	1	1	8bits		8bits							

The first byte is always read as 0xFF. The first byte must be discarded by the master

Control Register (CTL) 0xB0109n84

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLV		-					FC					DRQEN			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-		TXTH		-		RXTH		--		RCLR	WS	SDA	CLR	-	EN

Field	Name	RW	Reset	Description
31-30	SLV	R/W	2'b00	Pin Configuration for Slave Core 2'b00/2'b11 2master operation I2C Channel 0 : Master 0 I2C Channel 1 : Master 1 2'b01 1 master / 1 slave I2C Channel 0 : Slave I2C Channel 1 : Master 1 2'b10 1 master / 1 slave I2C Channel 0 : Master0 I2C Channel 1 : Slave
29-25	Reserved	-	-	-
24-20	FC	R/W	5b'00000	Filter Counter Load Value "0" disables noise filter. SCL and SDA inputs are checked for stability until the counter is expired.
19-16	DRQEN	R/W	3'b000	DMA Request Enable
15-14	Reserved	-	-	-
13-12	TXTH	R/W	2'b00	TX FIFO threshold for DMA Request TX FIFO threshold value
11-10	Reserved	-	-	-
9-8	RXTH	R/W	2'b00	RX FIFO threshold for DMA Request RX FIFO threshold value
7-6	Reserved	-	-	-
5	RCLR	R/W	1'b0	Clear Interrupt Status at read cycle 1'b1=Clear
4	WS	R/W	1'b0	Wait Status Control by SCL Stretching 1'b0=Enable 1'b1=Disable
3	SDA	R/W	1'b0	Reserved for Test
2	CLR	R/W	1'b0	Clear FIFO 1'b1= Clear FIFO
1	Reserved	-	-	-
0	EN	R/W	1'b0	Enable for this slave core 1'b0=Disable 1'b1=Enable

Address Register (ADDR) 0xB0109n88

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								ADDR							-

Field	Name	RW	Reset	Description
31-8	Reserved	-	-	-
7-1	ADDR	W	7'b0000000	Device Slave address
0	Reserved	-	-	-

Interrupt Register (INT) **0xB0109n8C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-				IRQSTAT											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-				IRQEN											

Field	Name	RW	Reset	Description
31-28	Reserved	-	-	-
27-16	IRQSTAT	W	3'h0	Interrupt Status 27th bit: All bytes of data buffer have been read by a master 26th bit: All bytes of data buffer have been written by a master 25th bit: Data buffer has been read by a master 24th bit: Data buffer has been written by a master 23th bit: TX FIFO under run 22th bit: RX FIFO over run 21th bit: TX bus cycle started with TX FIFO empty 20th bit: RX FIFO full 19th bit: TX FIFO empty 18th bit: RX FIFO not empty 17th bit: TX FIFO Level (TXVC <= TXTH) 16th bit: RX FIFO Level (RXVC >= RXTH)
15-12	Reserved	-	-	-
11-0	IRQEN	W	3'h0	Interrupt Enable 11th bit: All byte of data buffer has been read by a master 10th bit: All byte of data buffer has been written by a master 9th bit: Data buffer has been read by a master 8th bit: Data buffer has been written by a master 7th bit: TX FIFO under run 6th bit: RX FIFO over run 5th bit TX bus cycle started with TX FIFO empty 4th bit RX FIFO full 3rd bit: TX FIFO empty 2nd bit: RX FIFO not empty 1st bit: TX FIFO Level (TXVC <= TXTH) 0th bit: RX FIFO Level (RXVC >= RXTH)

Note: When RCLR bit of Control Register is "1", IRQSTAT bits are cleared at the end of read cycle.

When RCLR bit of Control Register is "0", IRQSTAT bits are cleared by writing "1".

Status Register (STAT) 0xB0109n90

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-								DDIR	-						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-								SADR							

Field	Name	RW	Reset	Description
31-24	Reserved	-	-	-
23	DDIR	R	1'b0	Data Direction 1'b0=RX 1'b1=TX
22-8	Reserved	-	-	-
7-0	SADR	R	2'h02	Sub address received Sub address received from address cycle.

Buffer Valid Flag Register (MBF) 0xB010909C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-								MBFT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-								MBFR							

Field	Name	RW	Reset	Description
31-24	Reserved	-	-	-
22-16	MBFT	R/W	2h'0	Buffer TX Flag Buffer TX flag for byte 7 ~ 0. When the data buffer 0/1 is read by an external master, the corresponding bit is set. MBFT[3:0] is cleared when the CPU writes to MB0 and MBFT[7:4] is cleared when MB1 is written by the CPU
15-8	Reserved	-	-	-
7-0	MBFR	R/W	2'h0	Buffer RX Flag When the data buffer 0/1 is read by an external master, the corresponding bit is set. MBFR[3:0] is cleared when the CPU writes to MB0 and MBFR[7:4] is cleared when MB1 is written by the CPU

Data buffer 0(MB0) 0xB0109nA0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Data byte 3								Data byte 2							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data byte 1								Data byte 0							

Field	Name	RW	Reset	Description
31-24	DB3	R/W	0x00000000	Data byte 3
23-16	DB2	R/W	0x00000000	Data byte 2
15-8	DB1	R/W	0x00000000	Data byte 1
7-0	DB0	R/W	0x00000000	Data byte 0

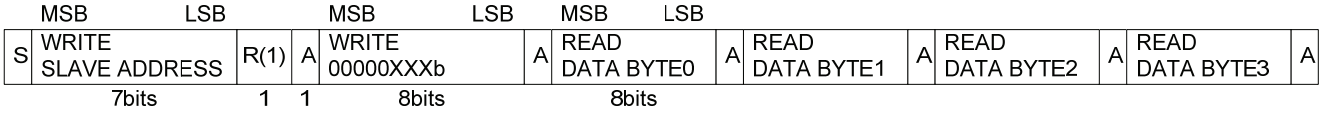
Data buffer 1(MB1) 0xB0109nA4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Data byte 7								Data byte 6							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data byte 5								Data byte 4							

Field	Name	RW	Reset	Description
31-24	DB7	R/W	0x00000000	Data byte 7
23-16	DB6	R/W	0x00000000	Data byte 6
15-8	DB5	R/W	0x00000000	Data byte 5
7-0	DB4	R/W	0x00000000	Data byte 4

NOTE: Deviation from the standard.

The MB0 and MB1 registers can not be read with the standard I2C read timing sequence. Always TX FIFO is accessed with the standard read timing. To read MB0 or MB1, the sequence below must be followed. If a master is not capable of this non-standard sequence, MB0 and MB1 can not be read.



IRQSTR 0xB0109nC0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													ST2	ST1	ST0

Field	Name	RW	Reset	Description
31-3	Reserved	-	-	-
2	ST2	R	1'b0	IRQ Status of I2C Slave Controller IRQ Status Flag for I2C Slave Controller This would be read as '1' when the IRQ status of I2C slave controller is activated.
1	ST1	R	1'b0	IRQ Status of I2C Master Controller Channel 1 IRQ Status Flag for I2C Master Controller of Channel 1. This would be read as '1' when the IRQ status of I2C master controller channel 1 is activated.
0	ST0	R	1'b0	IRQ Status of I2C Master Controller Channel 0 IRQ Status Flag for I2C Master Controller of Channel 0. This would be read as '1' when the IRQ status of I2C master controller channel 0 is activated.

IRQ_I2C **0xB0109300**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0													Ch2	Ch1	Ch0

Field	Name	RW	Reset	Description
31-3	Reserved	-	-	-
2	Ch2	R	1'b0	IRQ Status of I2C Channel2 Group IRQ Status Flag for each channe2 of I2C Group
1	Ch1	R	1'b0	IRQ Status of I2C Channel1 Group IRQ Status Flag for each channe1 of I2C Group
0	Ch0	R	1'b0	IRQ Status of I2C Channel0 Group IRQ Status Flag for each channe0 of I2C Group

DREQ_SEL **0xB0109400**

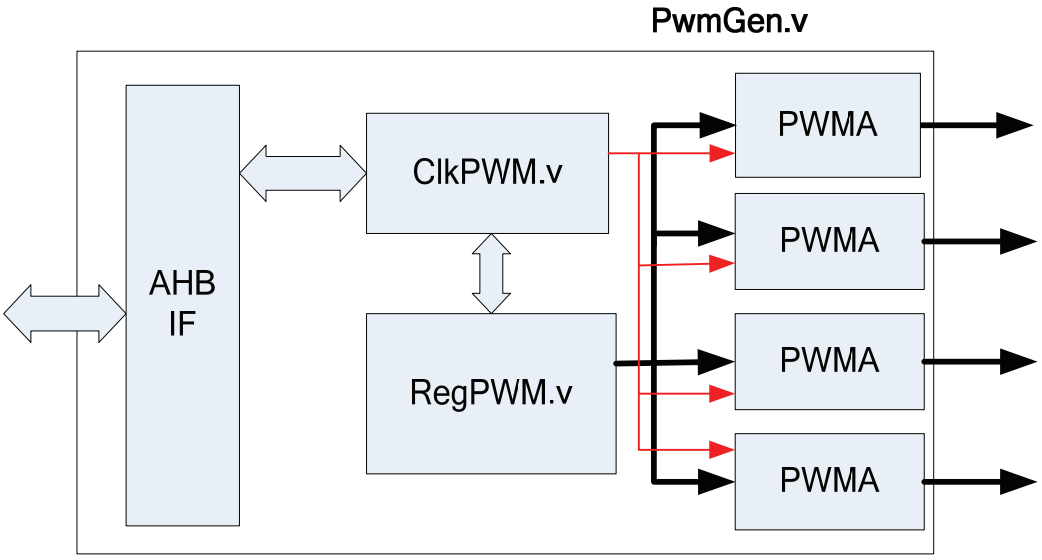
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0															Sel

Field	Name	RW	Reset	Description
31-1	Reserved	-	-	-
0	Sel	RW	1'b0	DMA Request Status of I2C Group This would be read as '1' when the DMA Request status of I2C group is activated.

15 PWM(Pulse Width Modulation)

15.1 Overview

This block designed for PWM(Pulse width Modulation) generation which is support external device를 control



15.2 Features

The PWM controller module is an AHB, providing a channel that can be programmed for actuator driver or other devices controlled by PWM pulse.

- General Feature
  - We have 4 separate PWM channel block
    (It can be programmed for actuator devices)
  - Each channel has enable control bit (We can turn on/off PWM block w/ this bit)

- Interrupt
  - One input interrupt pin required
    (When actuator goes to bottom side, photo interrupt will be occurred)

- Pulse control
  - 1 pulse mode (2 phase) : Width of phase is controllable
  - 2 pulse mode (4 phase) : Width of phase is controllable
  - Supports continuous mode(infinite loop) and Parallel mode.
  - Programmed repetition mode (1 ~ 15 steps) at Parallel mode
  - We can clock generate (To avoid noise issue at special frequency)

15.3 Register Description

Table 15.1 PWM Register Map (Base Address = 0xB0060000)

Name	Address	RW	Default	Description
PwmEn	0x0004	R/W	0x0000000F	PWM block enable & PWM Trigger
PwmMode	0x0008	R/W	0x000000AA	PWM output mode & PWMCLKDiv
PwmLoop	0x000C	R/W	0x00000000	PWM block operation Steps
PwmAPstn12	0x0010	R/W	0x00200010	PWM block A position 1 & 2
PwmAPstn34	0x0014	R/W	0x00400030	PWM block A position 3 & 4
PwmBPstn12	0x0018	R/W	0x00200010	PWM block B position 1 & 2
PwmBPstn34	0x001C	R/W	0x00400030	PWM block B position 3 & 4
PwmCPstn12	0x0020	R/W	0x00200010	PWM block C position 1 & 2
PwmCPstn34	0x0024	R/W	0x00400030	PWM block C position 3 & 4
PwmDPstn12	0x0028	R/W	0x00200010	PWM block D position 1 & 2
PwmDPstn34	0x002C	R/W	0x00400030	PWM block D position 3 & 4
PwmOutRegA1	0x0030	R/W	0xAA0000AA	PWM A1 block Reg mode Output value
PwmOutRegA2	0x0034	R/W	0x00AAAA00	PWM A2 block Reg mode Output value
PwmOutRegA3	0x0038	R/W	0x00AAAA00	PWM A3 block Reg mode Output value
PwmOutRegA4	0x003C	R/W	0xAA0000AA	PWM A4 block Reg mode Output value
PwmOutRegB1	0x0040	R/W	0x55000000	PWM B1 block Reg mode Output value
PwmOutRegB2	0x0044	R/W	0x00550000	PWM B2 block Reg mode Output value
PwmOutRegB3	0x0048	R/W	0x00005500	PWM B3 block Reg mode Output value
PwmOutRegB4	0x004C	R/W	0x00000055	PWM B4 block Reg mode Output value
PwmOutRegC1	0x0050	R/W	0xAA000000	PWM C1 block Reg mode Output value
PwmOutRegC2	0x0054	R/W	0x00AA0000	PWM C2 block Reg mode Output value
PwmOutRegC3	0x0058	R/W	0x0000AA00	PWM C3 block Reg mode Output value
PwmOutRegC4	0x005C	R/W	0x000000AA	PWM C4 block Reg mode Output value
PwmOutRegD1	0x0060	R/W	0x55000000	PWM D1 block Reg mode Output value
PwmOutRegD2	0x0064	R/W	0x00550000	PWM D2 block Reg mode Output value
PwmOutRegD3	0x0068	R/W	0x00005500	PWM D3 block Reg mode Output value
PwmOutRegD4	0x006C	R/W	0x00000055	PWM D4 block Reg mode Output value

PwmEn **0xB0060004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
												D_Trig	C_Trig	B_ Trig	A_ Trig
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
												D_En	C_En	B_En	A_En

Field	Name	RW	Reset	Description
31-20	-	-	-	Reserved
19	D_Trig	R/W	0x0	PWM D Output gen. Triggering 1: enable 0: disable
18	C_Trig	R/W	0x0	PWM C Output gen. Triggering 1: enable 0: disable
17	B_Trig	R/W	0x0	PWM B Output gen. Triggering 1: enable 0: disable
16	A_Trig	R/W	0x0	PWM A Output gen. Triggering 1: enable 0: disable
15-4	-	-	-	Reserved
3	D_En	R/W	0x1	PWM D block Enable 1: enable 0: disable
2	C_En	R/W	0x1	PWM C block Enable 1: enable 0: disable
1	B_En	R/W	0x1	PWM B block Enable 1: enable 0: disable
0	A_En	R/W	0x1	PWM A block Enable 1: enable 0: disable

PwmMode **0xB0060008**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Div_D		Div_C		Div_B		Div_A						Inv_D	Inv_C	Inv_B	Inv_A
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Mode_D				Mode_C				Mode_B				Mode_A			

Field	Name	RW	Reset	Description
31-30	Div_D	R/W	0x0	PDM D Block input Clock divide 3: Divide by 16 2: Divide by 8 1: Divide by 4 0: Divide by 2
29-28	Div_C	R/W	0x0	PDM C Block input Clock divide 3: Divide by 16 2: Divide by 8 1: Divide by 4 0: Divide by 2
27-26	Div_B	R/W	0x0	PDM B Block input Clock divide 3: Divide by 16 2: Divide by 8 1: Divide by 4 0: Divide by 2
25-24	Div_A	R/W	0x0	PDM A Block input Clock divide 3: Divide by 16 2: Divide by 8 1: Divide by 4 0: Divide by 2
23-20	-	-	-	Reserved
19	Inv_D	R/W	0x0	Pdm D Output Signal Inverse 1: enable 0: disable
18	Inv_C	R/W	0x0	Pdm C Output Signal Inverse 1: enable 0: disable
17	Inv_B	R/W	0x0	Pdm B Output Signal Inverse 1: enable 0: disable
16	Inv_A	R/W	0x0	Pdm A Output Signal Inverse 1: enable 0: disable
15-12	Mode_D	R/W	0x2	PWM D block mode Bit 2: Register Out mode2 Bit 1: Register Out mode Bit 0: Phase mode
11-8	Mode_C	R/W	0x2	PWM C block mode Bit 2: Register Out mode2 Bit 1: Register Out mode Bit 0: Phase mode
7-4	Mode_B	R/W	0x2	PWM B block mode Bit 2: Register Out mode2 Bit 1: Register Out mode Bit 0: Phase mode
3-0	Mode_A	R/W	0x2	PWM A block mode Bit 2: Register Out mode2 Bit 1: Register Out mode Bit 0: Phase mode

PwmLoop **0xB006000C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PwmLoopD				PwmLoopC				PwmLoopB				PwmLoopA			

Field	Name	RW	Reset	Description
31-16	-	-	-	Reserved
15-12	Loop_D	R/W	0x0	PWM block D step repetition value
11-8	Loop_C	R/W	0x0	PWM block C step repetition value
7-4	Loop_B	R/W	0x0	PWM block B step repetition value
3-0	Loop_A	R/W	0x0	PWM block A step repetition value

(* pdmLoop == 0 , Infinite value mode)

PwmPstn12_N (N=A,B,C,D) **0xB0060010/18/20/28**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PwmNPstn2															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PwmNPstn1															

Field	Name	RW	Reset	Description
31-16	PwmPstn2_N	R/W	0x20	Input Clock(PWMClk N) count value2 (N=A,B,C,D)
15-0	PwmPstn1_N	R/W	0x10	Input Clock(PWMClk N) count value1 (N=A,B,C,D)

PwmAPstn34_N (N=A,B,C,D) **0xB0060014/1C/24/2C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PwmAPstn4															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PwmAPstn3															

Field	Name	RW	Reset	Description
31-16	PwmPstn4_N	R/W	0x40	Input Clock(PWMClk N) count value4 (N=A,B,C,D)
15-0	PwmPstn3_N	R/W	0x30	Input Clock(PWMClk N) count value3 (N=A,B,C,D)

PwmOutReg_N1 (N=A,B,C,D) **0xB0060030/40/50/60**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PwmOutReg_N1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PwmOutReg_N1															

Field	Name	RW	Reset	Description
31-0	PwmOutReg_N1	R/W	A1: 0xAA0000AA B1: 0x55000000 C1: 0xAA000000 D1: 0x55000000	PWM output data pattern value 1

PwmOutReg_N2 (N=A,B,C,D) **0xB0060034/44/54/64**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PwmOutReg_N2															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PwmOutReg_N2															

Field	Name	RW	Reset	Description
31-0	PwmOutReg_N2	R/W	A2: 0x00AAAA00 B2: 0x00550000 C2: 0x00AA0000 D2: 0x00550000	PWM output data pattern value 2

PwmOutReg_N3 (N=A,B,C,D) **0xB0060038/48/58/68**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PwmOutReg_N3															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PwmOutReg_N3															

Field	Name	RW	Reset	Description
31-0	PwmOutReg_N3	R/W	A3: 0x00AAAA00 B3: 0x00005500 C3: 0x0000AA00 D3: 0x00005500	PWM output data pattern value 3

PwmOutReg_N4 (N=A,B,C,D) **0xB006003C/4C/5C/6C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PwmOutReg_N4															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PwmOutReg_N4															

Field	Name	RW	Reset	Description
31-0	PwmOutReg_N4	R/W	A4: 0xAA0000AA B4: 0x00000055 C4: 0x000000AA D4: 0x00000055	PWM output data pattern value 4

15.4 PWM clock generating

We can generate clock for PWM pulse generation
To Avoid noise issue , we can select clock source. (PWMClk)
(Input clock to PWM Gen module is 100 ~ 150Mhz)

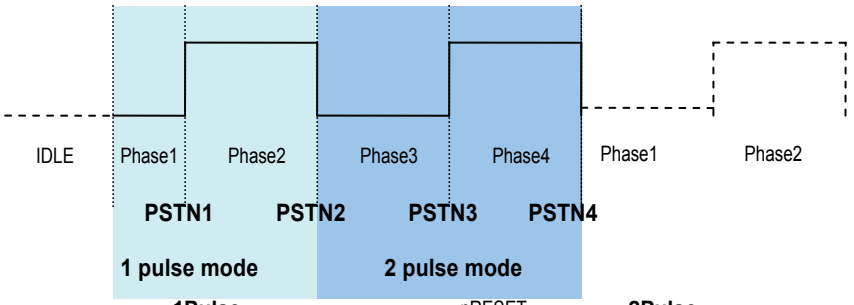
*** CLOCK divide method**

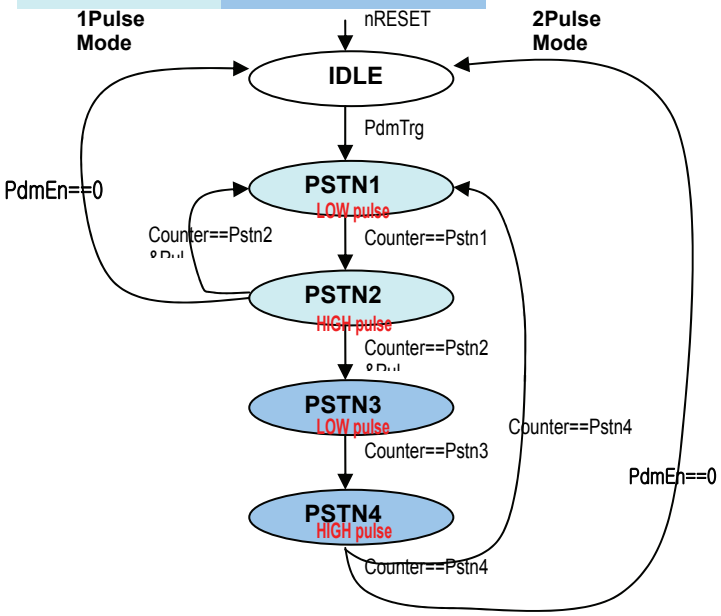
```
always@(PWMCLKDiv, DivCnt)
begin
        case(PWMCLKDiv)
        2'b11 : PWMCLK = DivCnt[3];
        2'b10 : PWMCLK = DivCnt[2];
        2'b01 : PWMCLK = DivCnt[1];
        2'b00 : PWMCLK = DivCnt[0];
        endcase
end
```

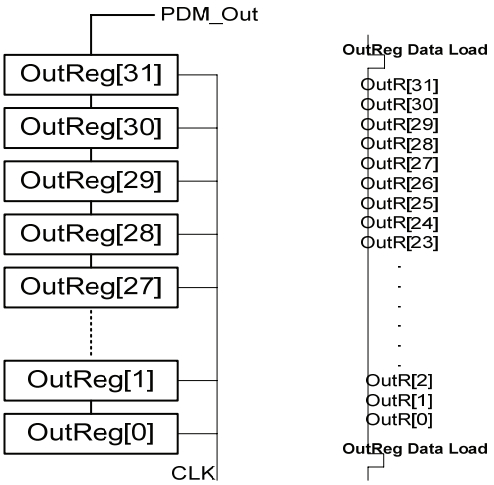
15.5 PWM block operation

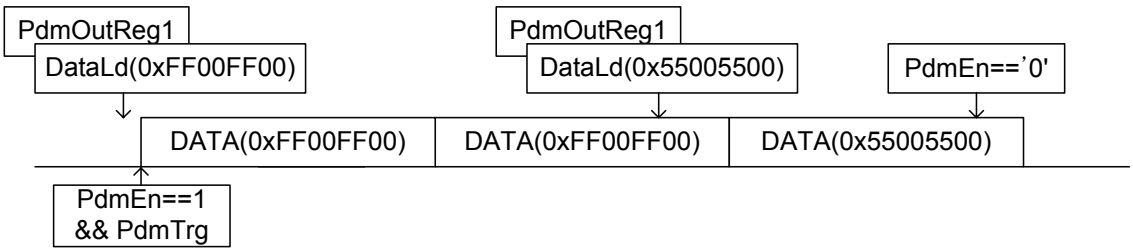
Operation
When Pwm loop==1
Channel Start (PWM trigger, (PwmTrg register set to '1'))
then goes to Phase1
(Counter value == Pstn1) then goes to Phase2
(Counter value == Pstn2) then goes to Phase3
Or goes to end(1pulse mode)
(Counter value == Pstn3) then goes to Phase4
(Counter value == Pstn4) then end(2pulse mode)

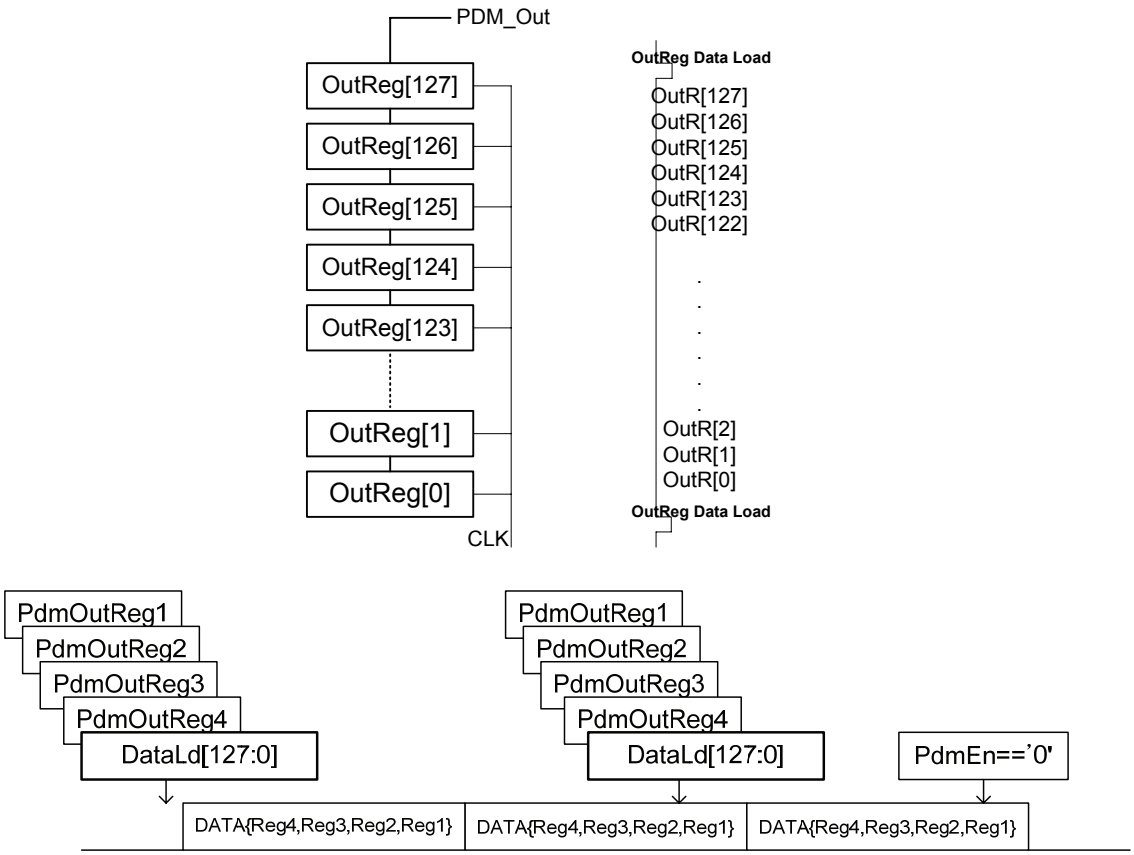
When Pwm loop==2 and 2-pulse mode
Channel Start (PWM trigger (PwmTrg register set to '1'))
then goes to Phase1
(Counter value == Pstn1) then goes to Phase2
(Counter value == Pstn2) then goes to Phase3
(Counter value == Pstn3) then goes to Phase4
(Counter value == Pstn4 && PwmEn==1) then goes to Phase1
(Counter value == Pstn1) then goes to Phase2
(Counter value == Pstn2) then goes to Phase3
(Counter value == Pstn3) then goes to Phase4
(Counter value == Pstn4) then end


IDLE
Phase1
Phase2
Phase3
Phase4
Phase1
Phase2
PSTN1
PSTN2
PSTN3
PSTN4
1 pulse mode
2 pulse mode


1Pulse Mode
nRESET
2Pulse Mode
IDLE
PdmTrg
PdmEn==0
PSTN1
LOW pulse
Counter==Pstn2
Counter==Pstn1
PSTN2
HIGH pulse
Counter==Pstn2
PSTN3
LOW pulse
Counter==Pstn4
Counter==Pstn3
PdmEn==0
PSTN4
HIGH pulse
Counter==Pstn4


PDM_Out
OutReg[31]
OutReg[30]
OutReg[29]
OutReg[28]
OutReg[27]
OutReg[1]
OutReg[0]
CLK
OutReg Data Load
OutR[31]
OutR[30]
OutR[29]
OutR[28]
OutR[27]
OutR[26]
OutR[25]
OutR[24]
OutR[23]
OutR[2]
OutR[1]
OutR[0]
OutReg Data Load


PdmOutReg1
DataLd(0xFF00FF00)
PdmOutReg1
DataLd(0x55005500)
PdmEn==’0'
DATA(0xFF00FF00)
DATA(0xFF00FF00)
DATA(0x55005500)
PdmEn==1
&& PdmTrg



15.6 AHB Transfer type

When HTRANS[1]==1 && HSEL==1 && HSReady ==1, AHB can be operated by Read/Write
(HTRAN==10, NONSEQ mode only)

16 Remote Control Interface

16.1 Functional Description

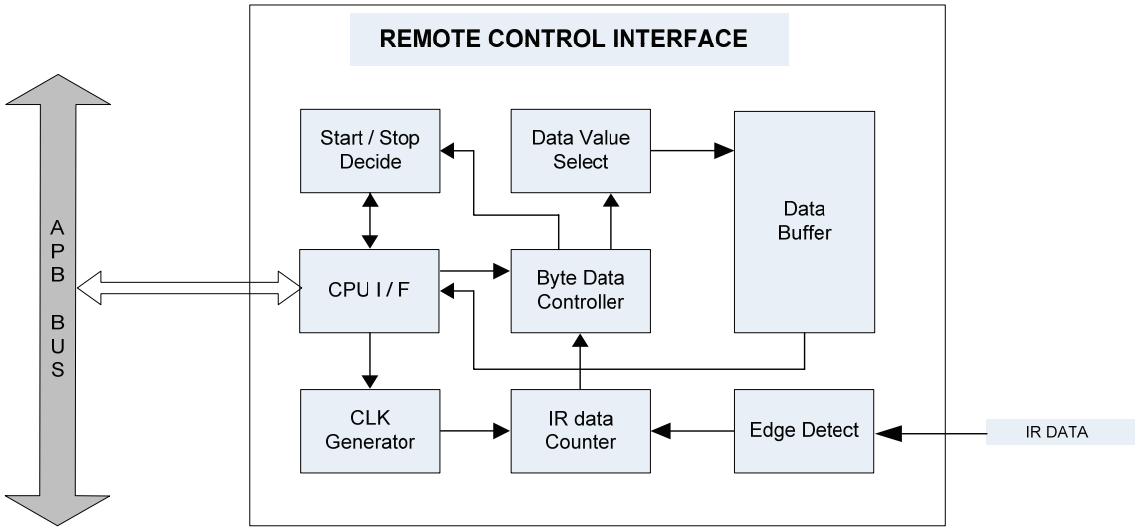


Figure 16.1 Remote Control Interface Block Diagram

The Remote Control Interface stores raw input samples of Infra-Red signal, generated from IR transmitter such as NOKIA, SONY, PHILLIPS, JVC, SHARP, etc. It does not support digital decoding like PWM, NRZ, NRZI but just counts runs of high and low level from IR input

Figure 16.2 shows the example of received IR data.

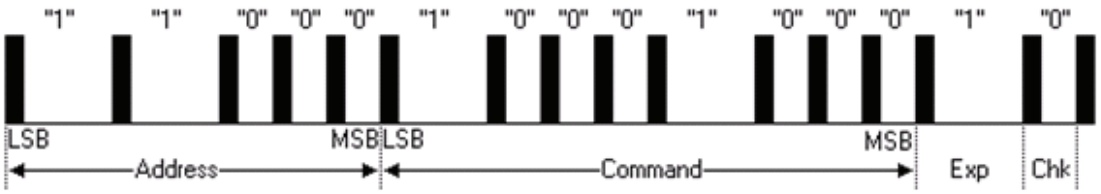


Figure 16.2 IR Data Input Example

When the rising edge of IR data is detected, high level run counting is started. If IR input level changes to low, it stops counting and stores the count number into the FIFO. And at the same time, low level counting is started.

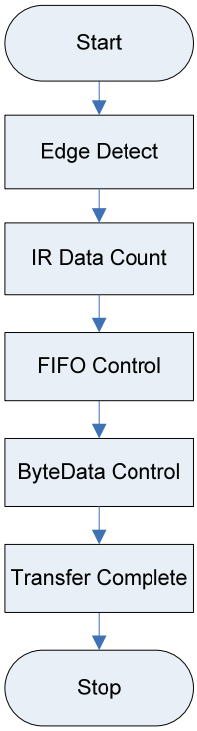


Figure 16.3 The Overall Flow of IR Data Capture

The high/low level run counts are written to internal FIFO in the order, shown in Figure 16.4.

Second IR data low dur_count value	Second IR data high dur_count value	First IR data low dur_cont value	First IR data high dur_count value
8bit	8bit	8bit	8bit

Figure 16.4 Data Write Format in FIFO

Note: The dur_count is 8-bit counter(ranges 0~255) and four count numbers are written at one time in the FIFO.

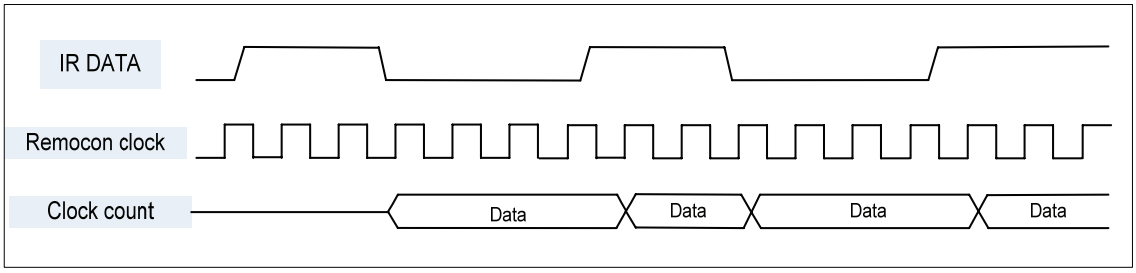


Figure 16.5 The Timing Diagram about Internal Counter Clock and dur_count Value

The sampling clock frequency can be changed according to clock divider register and be adjusted so that the count number does not exceed the counter range.

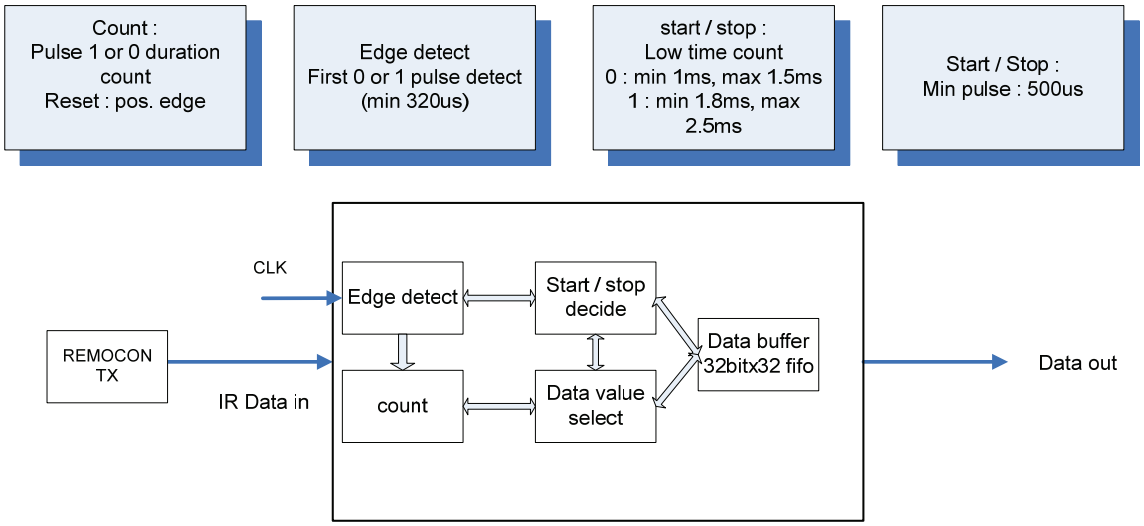


Figure 16.6 The Minimum or Maximum Pulse Width of IR Signal

Figure 16.6 shows the minimum or maximum pulse width of IR signal required.

16.2 Register Description

Table 16.1 Remocon Register Map (Base Address = 0xB0101000)

Name	Addr. Offset	RW	Reset	Description
RDATA	0x00	R	0x0000	IR read Data
CMD	0x04	R/W	0x0000	Command Register
INPOL	0x08	R/W	0x0000	Input Polarity Inversion Register
STA	0x0C	R/W	0x0000	Status Register
CLKDIV	0x40	R/W	0x0000	Clock Divide Register

IR Read Data Register (RDATA) 0xB0101000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDATA															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA															

Field	Name	RW	Reset	Description
31-0	RDATA	R	0x00000000	IR data count value read

When remocon interrupt arises, host processor can repeatedly reads this register only 18-times(the maximum number of IR signal bit is restricted to 36), and analyzes all IR input. Also the initialization of read FIFO would be taken like this.

Command Register (CMD) 0xB0101004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	FWEN	DEN	WS	TH									CLEAR	EN	FF

Field	Name	RW	Reset	Description
14	FWEN	R/W	0x0	FIFO Write Enable 0 = Disable 1 = Enable
13	DEN	R/W	0x0	Duration Count Enable 0 = Disable 1 = Enable
12	WS	R/W	0x0	IR CLK Wait Control 0 = Disable 1 = Enable
11-3	TH	R/W	0x00	FIFO Threshold value Data value : 0 ~ 31
2	CLEAR	R/W	0x0	Duration Count clear 0 = Disable 1 = Enable
1	EN	R/W	0x0	Remocon Core Enable 0 = Disable 1 = Enable
0	FF	R/W	0x0	Remocon FIFO clear 0 = Disable 1 = Enable

Input Polarity Inversion Register (INPOL) 0xB0101008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				FT		FIL	SCLK	0							INV

Field	Name	RW	Reset	Description
11-10	FT	R/W	0x0	IR signal Filter Tab Select 0 = 16Tab 1 = 4Tab 2 = 8Tab 3 = 16Tab
9	FIL	R/W	0x0	IR signal Filter enable 0 = Disable 1 = Enable
8	SCLK	R/W	0x0	Remocon Core Clock Select 0 = PBUS divide Clock 1 = XTIN Clock
0	INV	R/W	0x0	Input Polarity Inversion 0 = Disable 1 = Enable

Note: IR data input polarity inversion

Status Register (STA) 0xB010100C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			OF	ICF											

Field	Name	RW	Reset	Description
12	OF	R/W	0x0	FIFO Over Flow 0 = Disable 1 = Enable
11-0	CIF	R/W	0x0	Interrupt Configure 0 = Disable All 0xFFF : Enable

CLK DIVIDER Register (CLKDIV) 0xB0101040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLK_DIV															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								END_CNT							

Field	Name	RW	Reset	Description
31-16	CLK_DIV	R/W	0x0000	Clock Divider Data Value : PBUS CLK divide value
7-0	END_CNT	R/W	0x00	Transfer end Zero Count Data Value : IR data transfer end count

This value counts (low duration) the completion timing for one packet of IR data transmission. This value should be over low duration range of data. For example, if max low duration is 4.5ms, the value should be over 4.5ms. If Remocon duration count CLK is 64us, END_CNT value is 4500us/64us = 70.3. Therefore, the value should be set over 71.

Note: PBUS CLK is used for Remote Controller. However, when IR data is actually read, demultiplied PBUS CLK is used. Therefore,if a value to be demultiplied is written in this register, the clock generator occurs a clock and counts ID data duration.

Remocon top CLK = PBUS CLK
Remocon duration count CLK = PBUS/CLK_DIV

For example, If PBUS is 150Mhz and CLK_DIV is 500 * 20, Remocon duration count CLK is 150/(500*20) = 64us.

17 TSADC Interface

17.1 Overview

The NVS2310 has 16 channel general purpose low-power ADC for battery level detection, key detection, remote control interface, touch screen interface, etc. It is a CMOS type 10/12-bit A/D converter with 16-channel analog input multiplexer.It converts the analog input signal into 10/12-bit binary digital codes at a maximum conversion rate of 1 MSPS with 5MHz A/D converter clock(CKIN). A/D converter operates with on-chip sample-and-hold function. The power down mode is supported. Touch-screen interface controls input pads(XP,XM,YP,YM) to obtain X/Y-position on the 4-wire touch screen device.

- Resolution : 10-bit / 12-bit
- Maximum Conversion Rate : 1MSPS
- Main Clock : 5MHz (Max.)
- Input Range : 0.0V ~ ADC Power
- Normal Conversion mode
- Separate X/Y Conversion mode
- Auto(Sequential) X/Y Conversion mode
- Waiting for Interrupt mode

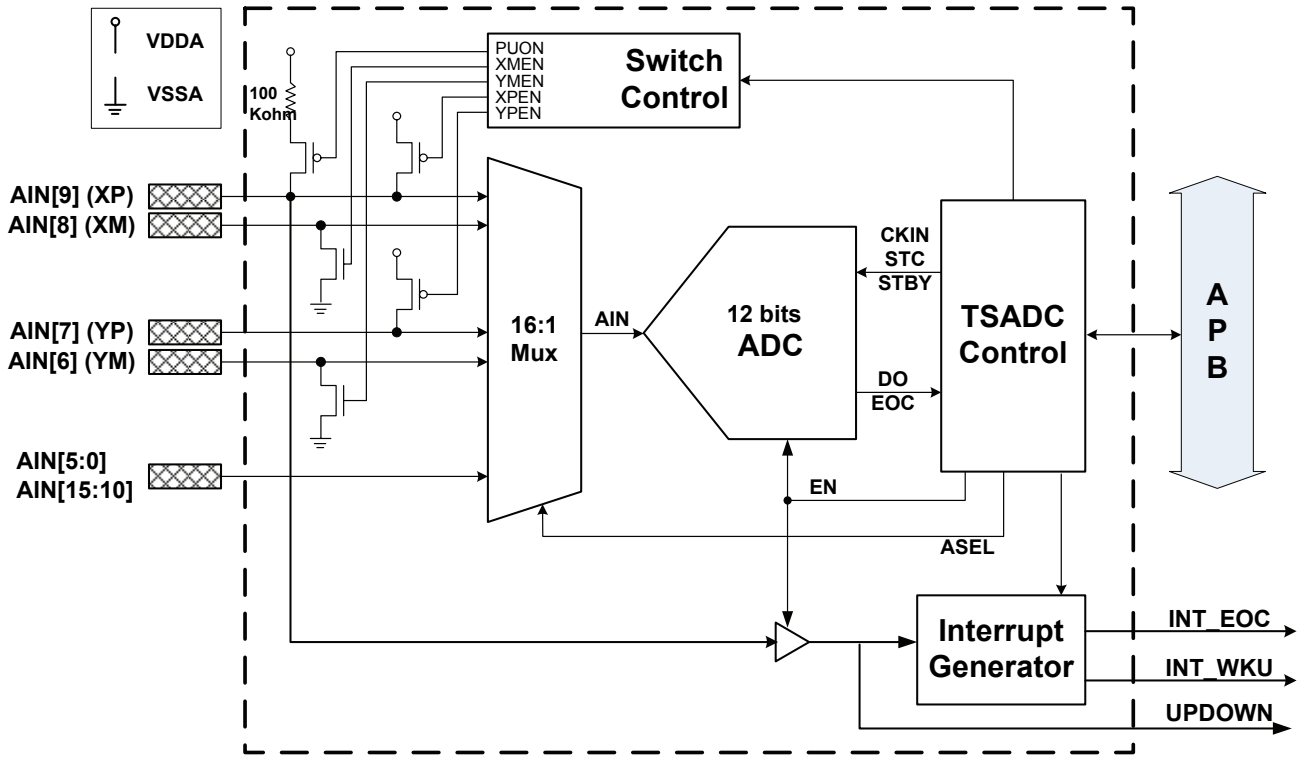


Figure 17.1 ADC Controller Block Diagram

TSADC interface have touch screen interface(4-wire) and 12 normal analog input. If touch screen is not used, touch screen interface is used as normal analog input. The maximum operating frequency of APB interface(PCLK_ADC) is 50 MHz. ADC clock(CKIN) is generated by PCLK_ADC. The maximum operating frequency of CKIN is 5 MHz(1 MSPS).

**CKIN should be set less than PCLK_ADC by 5 times

X-position conversion data is inserted to ADCDAT0, Y-position data is inserted to ADCDAT1.

17.1.1 A/D Conversion Time

When the PCLK_ADC freq. is 50 MHz and the prescaler value is 49, total 10-bit/12-bit conversion time is as follows.

ADC freq(CKIN) = 50 MHz / (49 + 1) = 1 MHz

Conversion time = 1 / (1 MHz / 5 cycles) = 1 / 200 KHz = 5 us

***** Maximum freq of PCLK_ADC is 50MHz.***
***** This A/D converter was designed to operate at maximum 5 MHz clock, so the conversion rate can go up to 1 MSPS.***

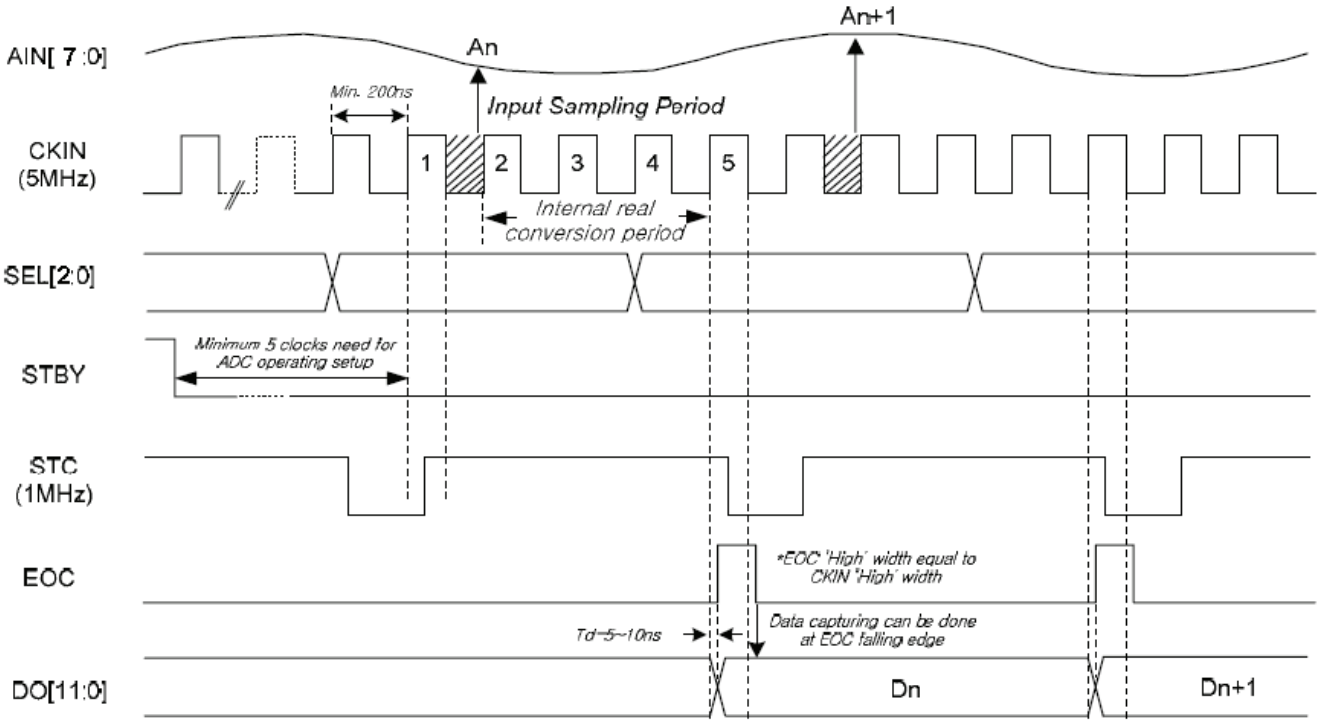


Figure 17.2 Main waveform

17.1.2 Touch Screen Interface Mode

17.1.2.1 Normal Conversion Mode (AUTO = 0, XY_PST = 0)

The operation of this mode is identical with AIN[0] ~ AIN[5], AIN[10] ~ AIN[15]. It can be initialized by setting the ADCCON and ADCTSC. All of the switches and pull-up resister should be turned off. The converted data can be read out from ADCDAT0 (A/D Conversion data 0 register).

17.1.2.2 Separate X/Y position Conversion Mode (AUTO = 0, XY_PST = control)

This mode consists of two states. One is X-position(X-pos.) measurement and the other is Y-position(Y-pos.) measurement state.

X-pos. conversion state is operated as the following way. Set XY_PST is '01' and read out the converted data from ADCDAT0. When XY_PST is '01', XP and XM switch is automatically on and ADC channel selection bits are automatically changed to '0111'(YP). The end of X-pos. conversion can be notified by interrupt(INT_EOC).

Y-pos. conversion state is operated as the following way. Set XY_PST is '10' and read out the converted data from ADCDAT1. When XY_PST is '10', YP and YM switch is automatically on and ADC channel selection bits are automatically changed to '1001'(XP). The end of Y-pos. conversion can be notified by interrupt(INT_EOC).

17.1.2.3 Auto(Sequential) X/Y position Conversion Mode (AUTO = 1, XY_PST = 0)

Auto(Sequential) X/Y Conversion mode is operated in the following method. Touch screen controller sequentially converts X-pos. and Y-pos. that is touched. After Touch Screen controller writes X-pos. data to ADCDAT0 and writes Y-pos. data to ADCDAT1, Touch screen interface generates Interrupt(INT_EOC). The conversion states are automatically changed. When X-pos is detected, XP and XM switch is automatically on and ADC channel selection bits are automatically changed to '0111'(YP).When Y-Pos is detected, YP and YM switch is automatically on and ADC channel selection bits are automatically changed to '1001'(XP). After auto X/Y position conversion, mode is changed for pull-up interrupt detection.

17.1.2.4 Waiting Interrupt Mode (ADCTSC = 0x00D3)

Touch screen controller generates an interrupt signal(INT_WKU) when the stylus pen is down or up. PUON is '0', XPEN is '1', XMEN is '0', YPEN is '1' and YMEN is '1'.

After touch screen controller generates interrupt signal(INT_WKU), waiting interrupt mode must be cleared. (XY_PST sets to '00' (No operation mode))

17.1.2.5 Stand-By Mode

Standby mode is activated when ADCCON[2] (STBY) is set to '1', In this mode, A/D Conversion operation is halted and ADCDAT0,ADCDAT1 register contains the previous converted data.

17.2 TSADC Controller Register Description

Table 17.1 TSADC Controller Register Map (Base Address = 0xB0100000)

Name	Address	RW	Reset	Description
ADCCON	0x00	R/W	0x00007F84	ADC Control Register
ADCTSC	0x04	R/W	0x00000058	ADC Touch Screen Control Register
ADCDLY	0x08	R/W	0x000000FF	ADC Start or Interval Delay Register
ADCDAT0	0x0C	R	-	ADC Conversion Data0 Register
ADCDAT1	0x10	R	-	ADC Conversion Data1 Register
ADCUPDN	0x14	R/W	0x00000000	Stylus Up or Down Interrupt Register
CLR_INT_EOC	0x18	W	-	Clear INT_EOC Interrrupt
Reserved	0x1C	-	-	Reserved
CLR_INT_WKU	0x20	W	-	Clear INT_WKU(Pen UP/DOWN) Interrupt

ADC Control Register (ADCCON) 0xB0100000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														RES	E_FLG
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PS_EN	PS_VAL								ASEL				STBY	RD_ST	EN_ST

Field	Name	RW	Reset	Description
17	RES	R/W	0x0	ADC Resolution Selection 0 = 10 bits A/D Conversion 1 = 12 bits A/D Conversion
16	E_FLG	R	0x0	End of Conversion(EOC) Flag 0 = A/D Conversion in Progress 1 = End of A/D Conversion
15	PS_EN	R/W	0x0	Enable Prescaler 0 = Disable 1 = Enable
14-7	PS_VAL	R/W	0xFF	Value of Prescaler Data value : 4 ~ 255 If N is even number, division factor is (N+2). If N is odd number, division factor is (N+1) ***** ADC frequency should be set less than PCLK_ADC by 5 times.*** (ex) PCLK_ADC=10MHz, ADC Freq<2MHz)
6-3	ASEL	R/W	0x0	Analog Input Channel Select 0000 = AIN0 0001 = AIN1 0010 = AIN2 0011 = AIN3 0100 = AIN4 0101 = AIN5 0110 = AIN6(YM) 0111 = AIN7(YP) 1000 = AIN8(XM) 1001 = AIN9(XP) 1010 = AIN10 1011 = AIN11 1100 = AIN12 1101 = AIN13 1110 = AIN14 1111 = AIN15
2	STBY	R/W	0x1	Stand-By mode selection 0 = Operation mode 1 = Stand-By mode
1	RD_ST	R/W	0x0	A/D Conversion Start by data read 0 = Disable 1 = Enable
0	EN_ST	R/W	0x0	A/D Conversion Start by Enable signal 0 = Disable 1 = Enable

ADC Touch Screen Register (ADCTSC) 0xB0100004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							UDEN	YMEN	YPEN	XMEN	XPEN	PUON	AUTO	XY_PST	

While waiting for stylus up/down interrupt, XPEN bit must be set to '1', and PUON bit must be set to '0', namely 'XP Pull-Up Enable'.

AUTO_PST bit should be set '1' only in Automatic and Sequential X/Y Position Conversion.

If you don't use touch screen, you should not tie AIN[9](XP) and AIN[7](YP) to VSSA.

Field	Name	RW	Reset	Description
8	UDEN	R/W	0x0	Detect Stylus UP / DOWN status 0 = Detect Stylus-DOWN status 1 = Detect Stylus-UP status
7	YMEN	R/W	0x0	YM to GND Switch Enable 0 = Disable (YM = AIN6, Hi-z) 1 = Enable (YM = VSSA)
6	YPEN	R/W	0x1	YP to VDD Switch Enable 0 = Enable (YP = VDDA) 1 = Disable (YP = AIN7, Hi-z)
5	XMEN	R/W	0x0	XM to GND Switch Enable 0 = Disable (XM = AIN8, Hi-z) 1 = Enable (XM = VSSA)
4	XPEN	R/W	0x1	XP to VDD Switch Enable 0 = Enable (XP = VDDA) 1 = Disable (XP = AIN9, Hi-z)
3	PUON	R/W	0x1	Pull-Up Switch Enable 0 = XP Pull-up Enable 1 = XP Pull-up Disable
2	AUTO	R/W	0x0	Automatic(sequencial) conversion of X-pos. and Y-pos 0 = Normal A/D conversion 1 = Automatic(sequencial) conversion of X-pos. and Y-pos
1-0	XY_PST	R/W	0x0	Select conversion of X-pos or Y-pos 00 = No operation 01 = X-pos conversion 10 = Y-pos conversion 11 = Waiting Interrupt mode

ADC Start Delay Register (ADCDLY) 0xB0100008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															CLKsrc
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DELAY															

Before A/D Conversion, touch screen uses X-tal clock

During A/D Conversion, PCLK_ADC is used.

Field	Name	RW	Reset	Description
16	CLKsrc	R/W	0	ADCDLY clock source In waiting for interrupt mode, CLKsrc is used as delay filter clock source. 0 = X-tal clock(External Input Clock) 1 = RTC clock
15-0	DELAY	R/W	0xFF	1. In case of A/D conversion mode (normal, separate, auto) : A/D conversion is delayed by counting clock is PCLK_ADC. -> A/D conversion delay value 2. In case of waiting interrupt mode : when stylus down occurs in waiting interrupt mode, it generates interrupt signal(INT_PENUPDN) at interval of several ms for AUTO X/Y-pos conversion.

ADC Data0 Register (ADCDAT0) 0xB010000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UPDN	AUTO	XYPST		XPDATA_12		XPDATA									

Field	Name	RW	Reset	Description
15	UPDN	R	-	UP/DOWN state of Stylus in Waiting Interrupt Mode. 0 = Stylus-DOWN 1 = Stylus-UP
14	AUTO	R	-	Automatic sequencing conversion of X-pos. and Y-pos. 0 = Normal A/D Conversion 1 = Automatic (sequencial) conversion of X-pos, Y-pos
13-12	XYPST	R	-	Select conversion of X-pos. or Y-pos. 00 = No operation. 01 = X-pos conversion. 10 = Y-pos conversion. 11 = Waiting Interrupt mode
11-10	XPDATA_12	R	-	When A/D resolution is 12bits, this is X-pos. conversion data[11:0] value.
9-0	XPDATA	R	-	X-pos. conversion data[9:0] value (Includes normal A/D Conversion data value) Data value : 0x00 ~ 0x3FF

ADC Data1 Register (ADCDAT1) 0xB0100010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UPDN	AUTO	XY_PST		XPDATA_12		XPDATA									

Field	Name	RW	Reset	Description
15	UPDN	R	-	Select UP/DOWN state of Stylus in Waiting Interrupt mode. 0 = Stylus-DOWN 1 = Stylus-UP
14	AUTO	R	-	Automatic(sequencial) conversion of X-pos. and Y-pos. 0 = Normal A/D conversion 1 = Sequencial conversion of X-pos, Y-pos
13-12	XY_PST	R	-	Select conversion of X-pos. or Y-pos. 00 = No operation 01 = X-pos conversion 10 = Y-pos conversion 11 = Waiting Interrupt mode
11-10	XPDATA_12	R	-	When A/D resolution is 12bits, this is X-pos conversion data[11:0] value.
9-0	XPDATA	R	-	X-pos. conversion data[9:0] value (Includes normal A/D conversion data value) Data value : 0x00 ~ 0x3FF

ADC Touch Screen UP/DOWN Register (ADCUPDN) 0xB0100014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														UP	DOWN

Field	Name	RW	Reset	Description
1	UP	R/W	0x0	Stylus-UP Interrupt history (***After check, this bit must be cleared manually***) 0 = No Stylus-UP Interrupt. 1 = Stylus-UP Interrupt has been occurred.
0	DOWN	R/W	0x0	Stylus-DOWN Interrupt history. (***After check, this bit should be cleared manually***) 0 = No Stylus-DOWN Interrupt 1 = Stylus-DOWN Interrupt has been occurred.

ADC INT_EOC Interrupt Clear Register (CLR_INT_EOC) 0xB0100018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															EOC_CLR

ISR(Interrupt Service Routine) is responsible for clearing interrupts after the interruptservice is completed. Writing any values on this register will clear up the relevant interrupts asserted.

Field	Name	RW	Reset	Description
0	EOC_CLR	W	-	Clear the INT_EOC Interrupt

ADC INT_WKU Interrupt Clear Register (CLR_INT_WKU) 0xB0100020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															WKU_CLR

ISR(Interrupt Service Routine) is responsible for clearing interrupts after the interrupt service is completed. Writing any values on this register will clear up the relevant interrupts asserted.

Field	Name	RW	Reset	Description
0	WKU_CLR	W	-	Clear the INT_WKU(Stylus UP/DOWN Interrupt)

17.3 Programming Note

The converted data can be accessed by means of interrupt or polling method. With interuupt method, the overall conversion time – from A/D converter start to convert data read – may be delayed because of the return time of interrupt service routine and data access time. With polling method, by checking the ADCCON[16] – end of conversion flag – bit, the read time from ADCDAT register can be determined.

A/D conversion can be activated in different way. After ADCCON[1] – A/D conversion start-by-read mode – is set to 1. A/D conversion starts simultaneously when converted data is read.

[Input range]

The analog input is single-ended type and the range is from VREF(ADC Power) to AGND(ADC Ground). This analog input voltage follows reference voltage range fundamentally. So, if you want to alter into another input range, you should change the voltage value of VREF(ADC Power).

Table 17.2 I/O Chart

Index	AIN input	Digital output	YPEN
0	~ 0.0008056	0000 0000 0000	1 LSB = 1.805 mV VREF = 3.3 V AGND = 0.0 V (at 3.3 V typical supply voltage)
1	0.0008056 ~ 0.0016112	0000 0000 0001	
~	~	~	
2047	1.6491944 ~ 1.65000	0111 1111 1111	
2048	1.6500000 ~ 1.6508056	1000 0000 0000	
2049	1.6508056 ~ 1.6516112	1000 0000 0001	
~		~	
	3.2983888 ~ 3.2991944	1111 1111 1110	
	3.2991944 ~ 3.3	1111 1111 1111	

[Analog Input Selection Table for 4-wire touch screen panels]

Five different functions are defined as shown in the following table. These functions require control of the switches in ADC as well as the interrupt function. After ADC generates the interrupt signal(UPDOWN), waiting for interrupt mode must be cleared by setting to the No operation mode.

Table 17.3 Analog Input Selection Table for 4-wire touch screen panels

XY_PST	XPEN	XMEN	YPEN	YMEN	PUON	ASEL[3:0]
X position measurement	0	1	1	0	1	0111(YP)
Y position measurement	1	0	0	1	1	1001(XP)
No operation (Normal ADC)	1	0	1	0	1	Select ASEL
Waiting Interrupt (for detecting pen up/down)	1	0	1	1	0	-
Auto measurement	auto	auto	auto	auto	auto	auto

For touch screen function, it is recommended to follow the below sequence.

1. Waiting for interrupt mode (for detecting pen up/down)
2. After receiving the interrupt signal, setting to no operation mode to clear waiting for interrupt mode
3. X position measurement (after several measurement, averaging over those results)
4. Y position measurement (the same method as X position measurement)
5. Waiting for interrupt mode (for detecting pen up/down)

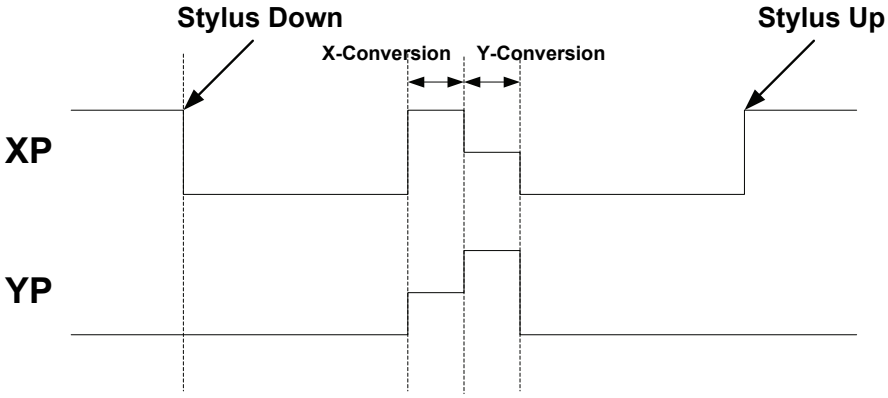


Figure 17.3 Operation Signal

[Analog Input Selection to Minimize Current Consumption]

When Power-Sleep mode, set the mode to Waiting Interrupt Mode.

Table 17.4 Analog Input Selection Table for Pressure Measurement

XY_PST	XPEN	XMEN	YPEN	YMEN	PUON	STBY
No Operation Mode	1	0	1	0	0	1
Waiting Interrupt Mode	1	0	1	1	0	1

If you don't use TSADC, PWR_EN(PMU) set to '0' to minimize current consumption.

[Analog Input Selection Table for Pressure Measurement]

Touch pressure can also be measured with this IP. Since the contact resistance between the X and Y plate de-creases as the pressure increases, the pressure of the touch screen can be determined by measuring it. To cal-culate the contact resistance, the method requires knowing the X-plate resistance, measurement of the X-position, and two additional cross panel measurement of the touch screen. The following table shows the con-figuration for pressure measurement using a 4-wire touch screen.

Table 17.5 Analog Input Selection Table for Pressure Measurement

XY_PST	XPEN	XMEN	YPEN	YMEN	PUON	ASEL[3:0]
X position measurement	0	1	1	0	1	0111(YP)
XP measurement	1	1	0	0	1	1001(XP)
YM measurement	1	1	0	0	1	0110(YM)

$$R_{contact} = R_{x-plate} \frac{X_{position}}{2^{12}} \left(\frac{YM_{measurement}}{XP_{measurement}} - 1 \right)$$

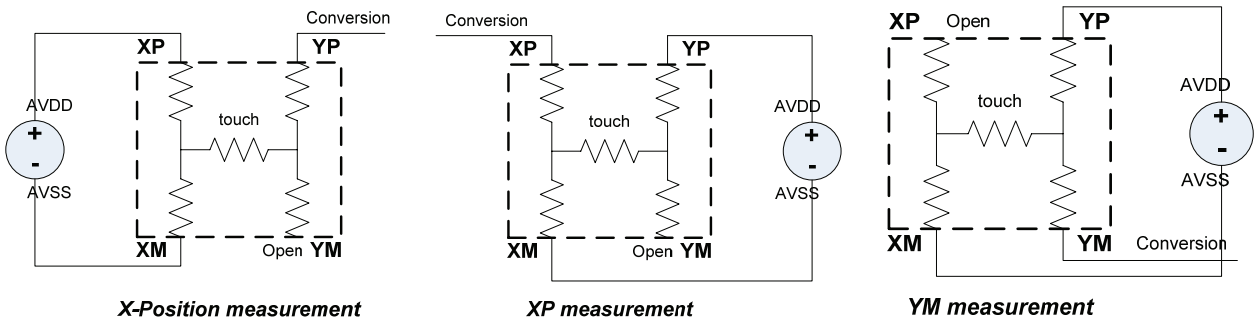


Figure 17.4 The principal of touch measurement

18 TSIF(The transport Stream Interface)

18.1 Overview

The NVS2310 includes two independent transport stream interface(TSIF) modules. The TSIF is able to receive 2 sets of parallel/serial input stream at once, supplied from overall 4 set of input ports.

The block diagram of TSIF is shown in Figure 18.1.

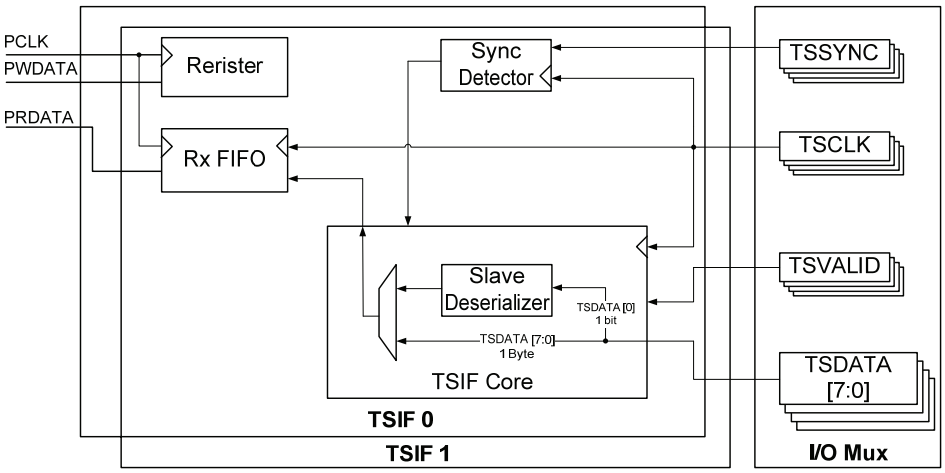


Figure 18.1 TSIF Block Diagram

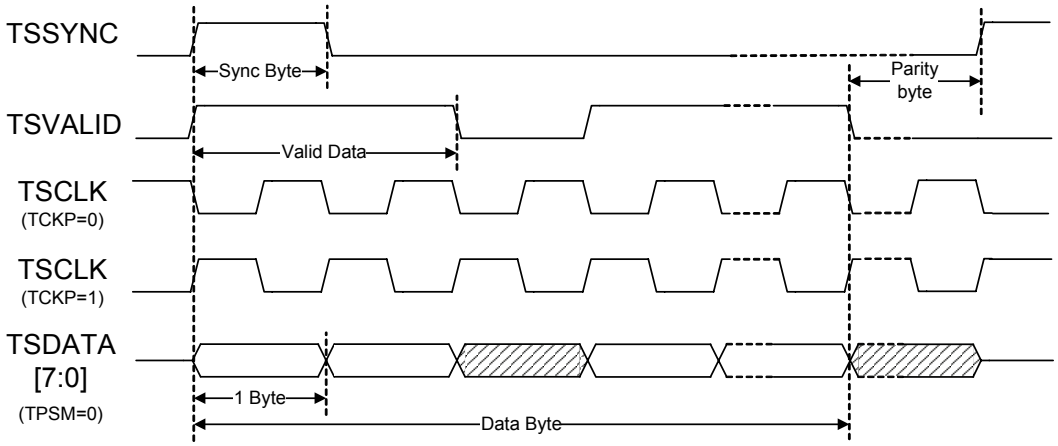


Figure 18.2 Timing of TS Data Input (Parallel Mode)

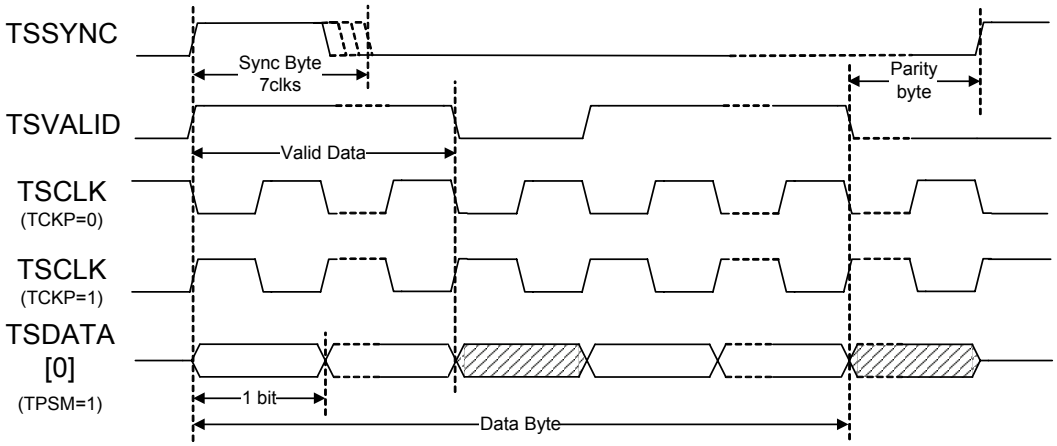


Figure 18.3 Timing of TS Data Input (Serial Mode)

Table 18.1 GPIO Port Map

GPIO_C	GPIO_D	GPIO_E	GPIO_F	I/O	TSIF
GPIO_C[0]	GPIO_D[9]	GPIO_E[19]	GPIO_F[7]	I	TSDATA [7]
GPIO_C[1]	GPIO_D[10]	GPIO_E[18]	GPIO_F[6]	I	TSDATA [6]
GPIO_C[2]	GPIO_D[13]	GPIO_E[17]	GPIO_F[5]	I	TSDATA [5]
GPIO_C[24]	GPIO_D[14]	GPIO_E[16]	GPIO_F[4]	I	TSDATA [4]
GPIO_C[25]	GPIO_D[17]	GPIO_E[15]	GPIO_F[3]	I	TSDATA [3]
GPIO_C[26]	GPIO_D[18]	GPIO_E[14]	GPIO_F[2]	I	TSDATA [2]
GPIO_C[27]	GPIO_D[19]	GPIO_E[13]	GPIO_F[1]	I	TSDATA [1]
GPIO_C[30]	GPIO_D[20]	GPIO_E[12]	GPIO_F[0]	I	TSDATA [0]
GPIO_C[28]	GPIO_D[15]	GPIO_E[22]	GPIO_F[8]	I	TSVALID
GPIO_C[29]	GPIO_D[16]	GPIO_E[20]	GPIO_F[9]	I	TSCLK
GPIO_C[31]	GPIO_D[12]	GPIO_E[21]	GPIO_F[10]	I	TSSYNC

18.2 Register Description

Table 18.2 TSIF Register Map (Base Address: Ch0 = 0xB0108000, Ch1 = Ch0 + 0x100)

Name	Address	RW	Reset	Description
TSDI	0x00	R	0x0000	TSIF Input Data Register
TSCR	0x04	R/W	0x0300	TSIF Control Register
TSPID	0x08~0x44	R/W	0x0000	TSIF PID Register
TSIC	0x48	R/W	0x0080	TSIF Interrupt Control Register
TSIS	0x4C	R	0x0000	TSIF Interrupt Status Register
TSISP	0x50	R	0x0000	TSIF interrupt Status(PID) Register
TSCHS	0x800	R/W	0x0000	TSIF Channel(Port) Select Register

* TSIF Ch_1 Address = TSIF Ch_0 + 0x100

TSIF Input Data Register (TSDI) **0xB0108000(100)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TSDI[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSDI[15:0]															

Field	Name	RW	Reset	Description
31-0	TSDI	R	32'h0	TSIF Input Data Register

TSIF Control Register (TSCR) **0xB0108004(104)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN	Reserved													SBE	M/L
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
END	TPSM	TCKP	TVEP	TSP	TrgPos	TBS		CRF	TVSC	TSDE	TSD				

Field	Name	RW	Reset	Description
31	EN	R/W	1'b0	TSIF Enable 1'b0 = Disable 1'b1 = Enable
30-18	R*	-	13'hx	Reserved
17	SBE	R/W	1'b0	Sync Byte Enable 1'b0 = Disable 1'b1 = Enable When the SBE is set to high, sync byte input value is compared with 0x47 and if it differs, the input data is not saved until next sync byte occurs.
16	M/L	R/W	1'b0	MSB/LSB 1'b0 = Send LSB firstly 1'b1 = Send MSB firstly
15	END	R/W	1'b0	Select Endian 1'b0 = Big Endian transfer mode 1'b1 = Little Endian transfer mode
14	TPSM	R/W	1'b0	TSIF Data Mode 1'b0 = TSIF Parallel Mode (TSDATA[7:0]) 1;b1 = TSIF Serial Model (TSDATA [0])
13	TCKP	R/W	1'b0	TSCLK Polarity 1'b0 = Starts transfer at the rising edge of TSCLK after TSSYNC is low 1'b1 = Starts transfer at the falling edge of TSCLK after TSSYNC is low
12	TVEP	R/W	1'b0	TSIF Valid Data Enable Polarity 1'b0 = TSVALID has low active pulse 1'b1 = TSVALID has high active pulse
11	TSP	R/W	1'b0	TSIF Sync Pulse Polarity 1'b0 = TSSYNC has low active pulse 1'b1 = TSSYNC has high active pulse
10	TrgPos	R/W	1'b0	Trigger Position 1'b0 = Regard the head of frame signal as a start point 1'b1 = Regard the tail of frame signal as a start point
9-8	TBS	R/W	2'b11	TSIF Byte Size 2'b00 = 1 Byte FIFO Save (Only Serial) 2'b01 = 2 Byte FIFO Save 2;b11 = 4 Byte FIFO Save (Default)
7	CRF	R/W	1'b0	Clear Rx FIFO 1'b0 = Don't empty Rx FIFO 1'b1 = Empties Rx FIFO
6	TVSC	R/W	1'b0	TSIF Valid Port Change 1'b0 = Disable 1'b1 = TSIF Valid port → TSIF Sync port(msm chip) If TSSYNC is not supplied, it can be replaced by TSVALID. In this case, only 3 signals(TSCLK, TSVALID, TSDATA) are used
5	TSDE	R/W	1'b0	TSIF Sync Delay Enable 1'b0 = Disable 1'b1 = Enable (msm chip)
4-0	TSD	R/W	5'h0	TSIF Sync Delay 5'hn = 1~32 Delay (Sync for msm chip)

TSIF PID Register (TSPID) 0xB0108008(108)~0xB0108044(144)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*		PIDEE	PIDEV(0,1,2, ~, 15)												

Field	Name	RW	Reset	Description
31-14	R*	-	2'hx	Reserved
13	PIDEE	R/W	1'b0	Paket ID (0,1,2, ~, 15) Enable 1'b0 = Disable 1'b1 = Enable
12-0	PIDEV	R/W	13'h0	Packet ID(0,1,2, ~, 15) Value 13'hn = Value(n = 0~1FFF)

The PID of new input data can be compared with PIDV. If it matches, it can set the pending bit of interrupt control register.

TSIF Interrupt Control Register (TSIC) 0xB0108048(148)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*(Reserved)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*(Reserved)						PID	R*	RxCnt			RCFul	REmp	RFul	ROR	RUR

These enable bits are used to monitor through Interrupt Status Register or make interrupt trigger or DMA request. It is recommended for you to set only what you want.

Field	Name	RW	Reset	Description
31-10	R*	-	22'hx	Reserved
9	PID	R/W	1'b0	Packet ID Matching 1'b0 = Disable 1'b1 = Enable
8	R*	-	1'bx	Reserved
7-5	RxCnt	R/W	3'h4	Rx FIFO Threshold Level 3'hn = Threshold Level (n=0~7) Sets threshold level of Rx FIFO to notify how many data it is filled with. It designates N depth data is available at Rx FIFO. Exceptionally, '0' stands for full state.
4	RCFul	R/W	1'b0	Rx FIFO Count Full Interrupt 1'b0 = Disable 1'b1 = Enable
3	REmp	R/W	1'b0	Rx FIFO Empty Interrupt 1'b0 = Disable 1'b1 = Enable
2	RFul	R/W	1'b0	Rx FIFO Full Interrupt 1'b0 = Disable 1'b1 = Enable
1	ROR	R/W	1'b0	Rx FIFO overrun error interrupt 1'b0 = Disable 1'b1 = Enable
0	RUR	R/W	1'b0	Rx FIFO underrun error interrupt 1'b0 = Disable 1'b1 = Enable

TSIF Interrupt Status Register (TSIS) 0xB010804C(14C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R*(Reserved)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R*(Reserved)											RCFul	REmp	RFul	ROR	RUR

* After you read this status register, it will be cleared

Field	Name	RW	Reset	Description
31-5	R*	-	27'hx	Reserved
4	RCFul	R	1'b0	Rx FIFO counter full status flag 1'b0 = Disable 1'b1 = Enable
3	REmp	R	1'b0	Rx FIFO empty status flag 1'b0 = Disable 1'b1 = Enable
2	RFul	R	1'b0	Rx FIFO full status flag 1'b0 = Disable 1'b1 = Enable
1	ROR	R	1'b0	Rx FIFO overrun error status flag 1'b0 = Disable 1'b1 = Enable
0	RUR	R	1'b0	Rx FIFO underrun error status flag 1'b0 = Disable 1'b1 = Enable

TSIF Interrupt Status(PID) Register (TSISP) 0xB0108050(150)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIDI15	PIDI14	PIDI13	PIDI12	PIDI11	PIDI10	PIDI9	PIDI8	PIDI7	PIDI6	PIDI5	PIDI4	PIDI3	PIDI2	PIDI1	PIDI0

Field	Name	RW	Reset	Description
31-16	-	-	-	Reserved
15-0	PIDI	R	16'h0	PID Interrupt Status (15~0) 1'b0 = Disable 1'b1 = Enable

TSIF Channel Select Register (TSCHS) 0xB0108200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												TSIF0		TSIF1	

Field	Name	RW	Reset	Description
31-16	R*	-	16'hx	Reserved
15-4	R*	-	12'hx	Reserved
3-2	TSIF0	R/W	2'b00	TSIF0 Port Select 2'b00 = GPIO_F Select 2'b01 = GPIO_E Select 2'b10 = GPIO_D Select 2'b11 = GPIO_C Select
1-0	TSIF1	R/W	2'b00	TSIF1 Port Select 2'b00 = GPIO_F Select 2'b01 = GPIO_E Select 2'b10 = GPIO_D Select 2'b11 = GPIO_C Select

*GPSB0 Base Address: 0xB010_7000 (TSIF0)
*GPSB1 Base Address: 0xB010_7100 (TSIF1)
*Register Setting: Rx Base(0x24), PACKET(0x28), DMA CTRL(0x2C) -> [2] TSIF Select, DMAICR(0x34)

19 UART

19.1 Overview

The NVS2310 has the 6-channel UART that can be used in programming the system software, IrDA interfacing or high speed serial communication. Additionally, these channels can be used as the smart card interface.

The block diagram of UART is shown in Figure 19.1.

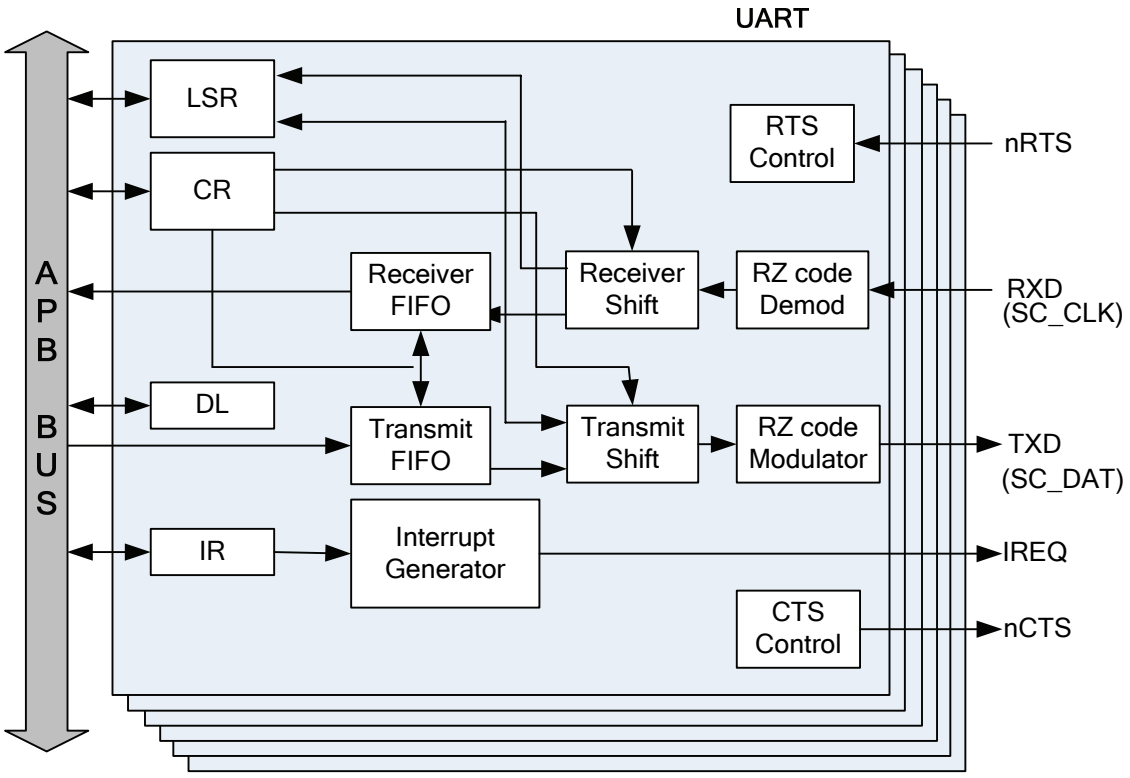
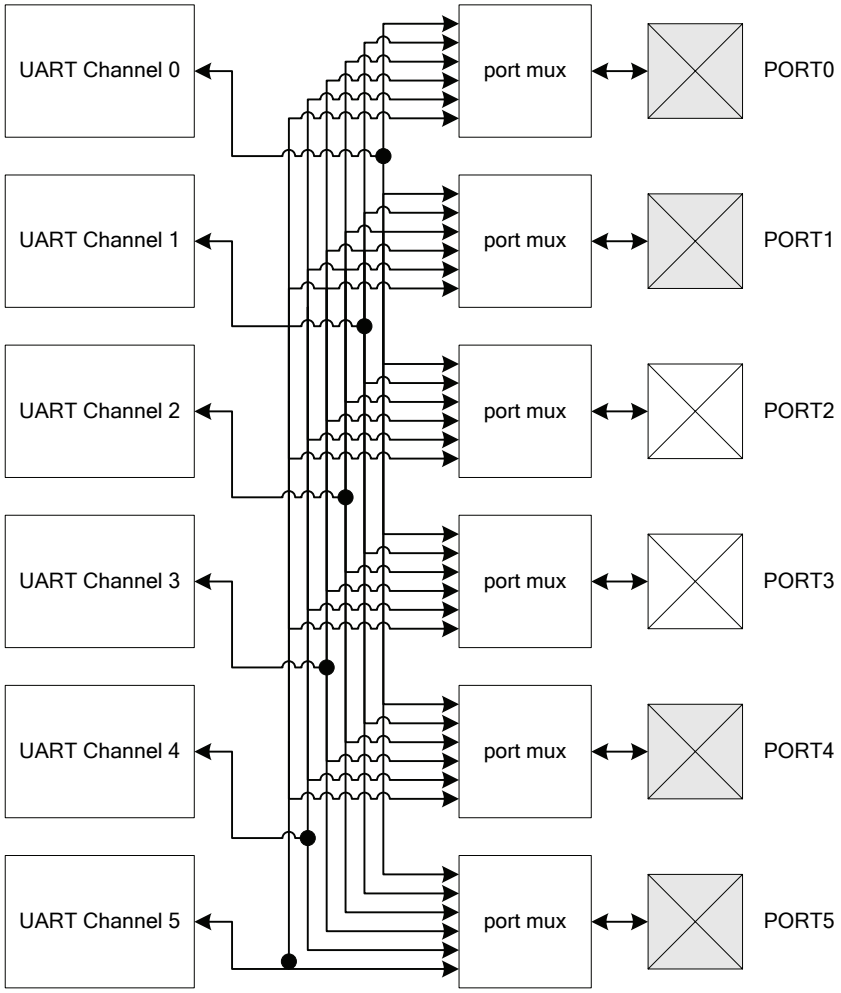


Figure 19.1 UART Block Diagram

Each channel can be connected to the one of 6 UART ports. UART port 0, port 1, port 4 and port 5 have flow-control signals, but UART port 2, and port 3 does not have them. Although all channels have the hardware flow control function, channels should be mapped to UART port 0, port 1, port 4, or port 5 to use the hardware flow control function.

For smart card interface, RXD signal is used as the smart card clock (SC_CLK) and TXD signal is used as the smart card data (SC_DAT). All UART ports and channels are available for smart card interface.



Port without hardware flow control signals

Port with hardware flow control signals

19.2 Operation Modes

19.2.1 AFC (Auto Flow Control) in RX Operation

Register Set Value
AFE (MCR[4]): Auto Flow Control Enable Bit = 1
DTL (AFT[3:0]) : nRTS Deassert Trigger Level = 15
ATL (AFT[7:4]) : nRTS Assert Trigger Level = 15
RXT (FCR[7:6]) : Rx Available FiFo Trigger Level = 0

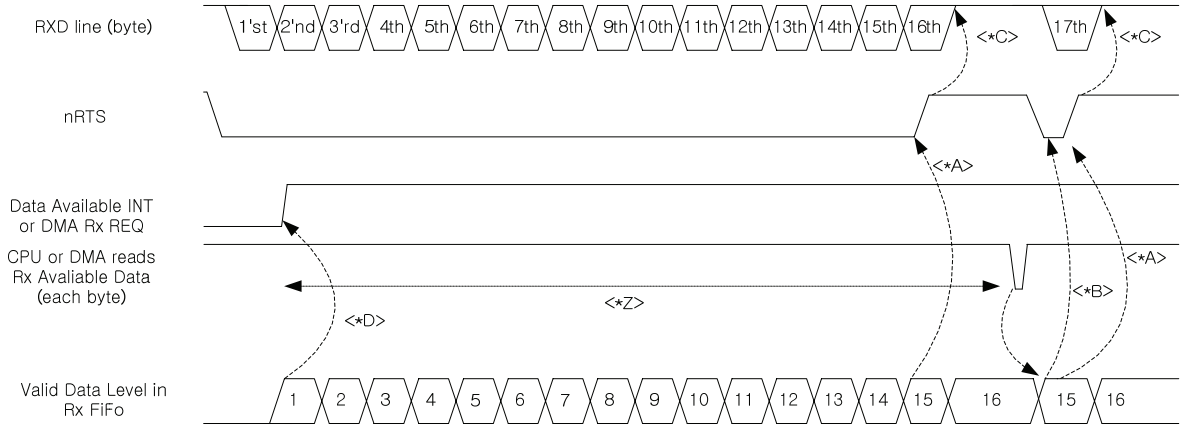


Figure 19.2 nRTS operation - Case: CPU/DMA is slow

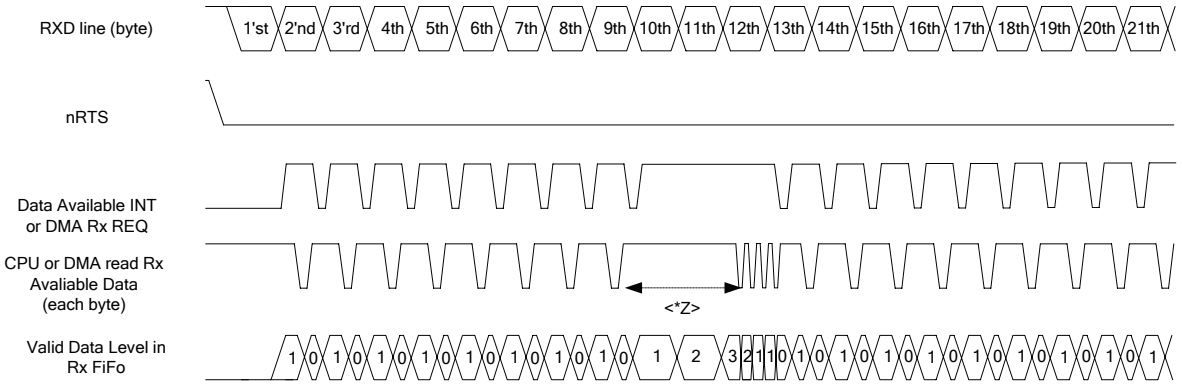


Figure 19.3 nRTS operation - Case: CPU/DMA is fast

<*Z>: When CPU can not read Rx available data due to other jobs or when DMA access can not be granted.
<*A>: Valida data is 15EA (AFT[3:0] = 15. According to RS(MCR[6]) value, nRTS is deasserted (nRTS = HIGH) whether under RXD Start Condition (RS=1) or under RXD Stop Condition (RS=0).
<*B>: When valid data decreases below 15(AFT [7:4]) and nRTS is deasserted, the nRTS wil be asserted.
<*C>: The uart counterpart communicating with NVS2310 does not send data when nRTS is deasserted.
<*D>: nRTS is asserted according to Rx Avaialble FiFo Trigger Level.
- AFT [7:0] should be set in such a way that latency to process is considered by detecting nRTS of the uart counterpart communicating with NVS2310.

19.2.2 AFC (Auto Flow Control) in TX Operation

Register Set Value :
AFE (MCR[4]): Auto Flow Control Enable Bit = 1 and RTS(MCR[1]) Bit = 1

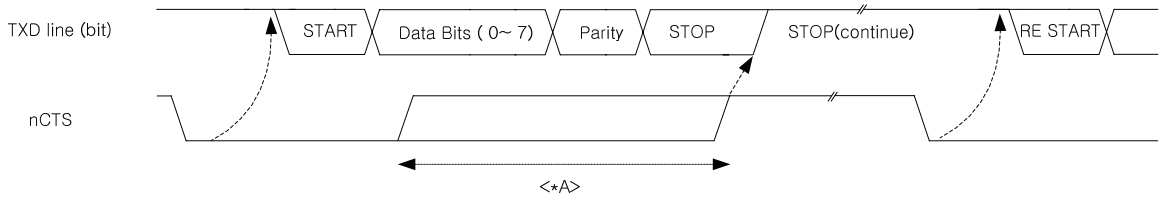


Figure 19.4 nCTS operation

<*A> : Even if nCTS is deasserted in the middle of TX transfer, current byte will be transferred.

19.2.3 Operation with General DMA

In UART channel 0~3, RX/TX data transmission through GDMA is available. Since in Channel 4 and 5, GDMA can not be used with hardware control style, in order to use it, make sure to use UART channel 0~3.

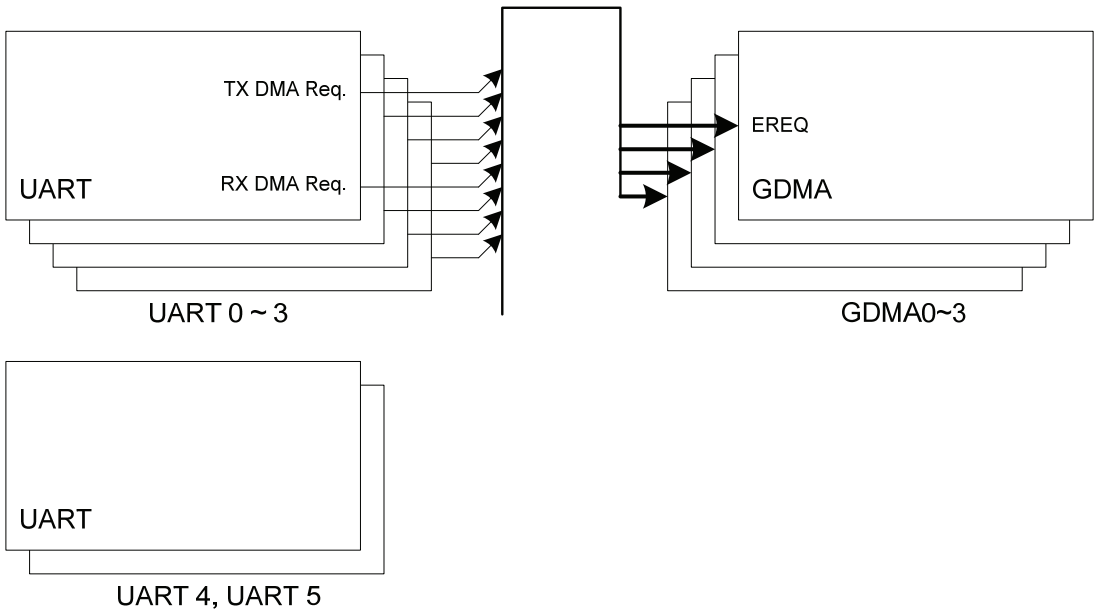


Figure 19.5 Operation with GDMA

19.2.4 RX Interrupt in DMA Transfer

Register Set Value :
FE(FCR[0]) : FiFo Enable Bit =1
RXT(FCR[7:6]) :
The Interrupt (or Rx Request) is asserted when
<*A> 2'b00 : 1 byte has been received. (CPU or DMA can read 1byte)
<*B> 2'b01 : 4 bytes has been received. (CPU or DMA can read 4bytes)
<*C> 2'b10 : 8 bytes has been received. (CPU or DMA can read 8bytes)
<*D> 3'b11 : 12 bytes has been received. (CPU or DMA can read 12bytes)

Cf. DMA Rx Request is asserted when RxDE(UCR[1]: Rx DMA Enable Bit)is Enabled.

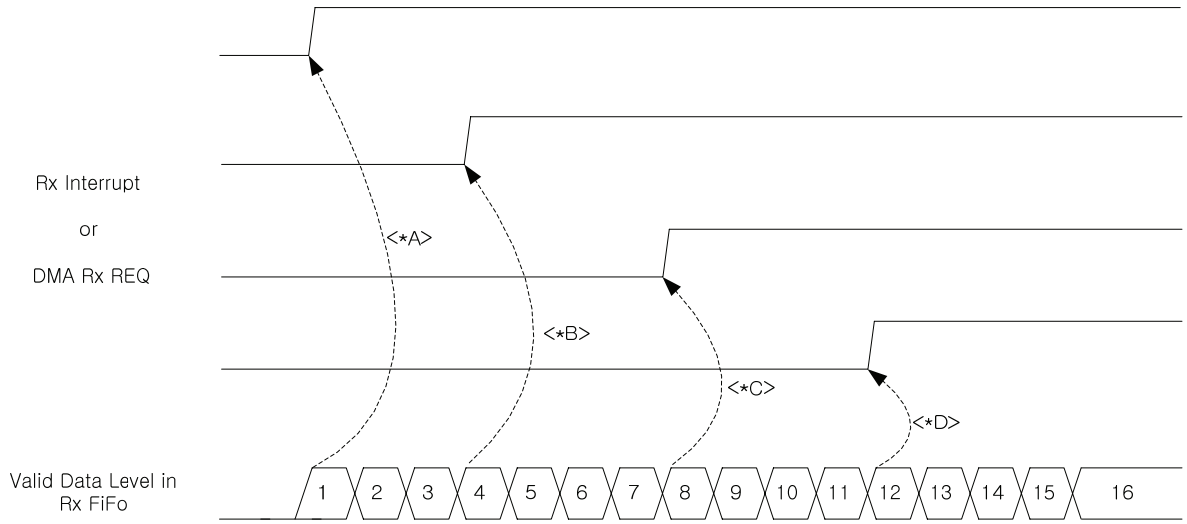


Figure 19.6 DMA operation – RX interrupt

19.2.4.1 TX Interrupt in DMA Transfer

Register Set Value :
FE(FCR[0]) : FiFo Enable Bit =1
TXT(FCR[5:4]) :
When Interrupt(or DMA Request) is asserted, The CPU (or DMA) is
<*A> 2'b00 : possible to write 16bytes at Tx FiFo (Empty)
<*B> 2'b01 : possible to write 8bytes at Tx FiFo
<*C> 2'b10 : possible to write 4bytes at Tx FiFo
<*D> 3'b11 : possible to write 1byte at Tx_FiFo

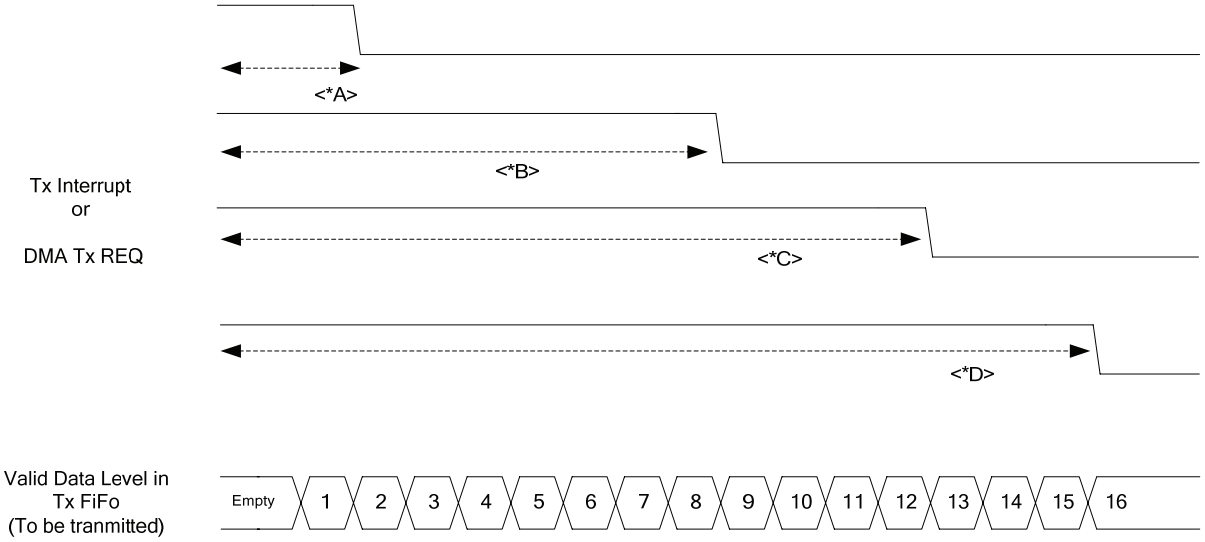
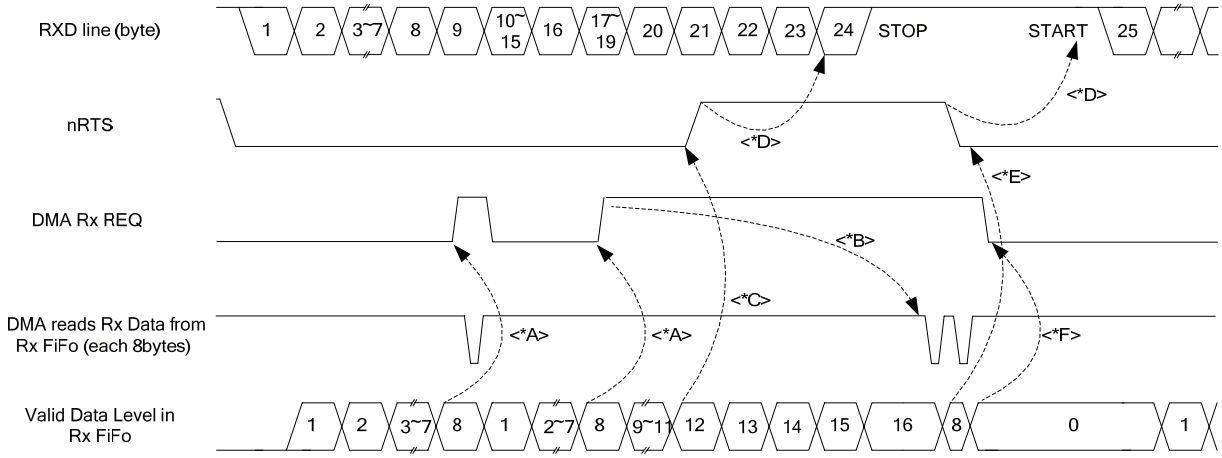


Figure 19.7 DMA operation – TX interrupt

19.2.5 RX Operation with DMA and AFC

UART Register Set Value
FE (FCR[0]) : FiFo Enable Bit = 1
AFE (MCR[4]): Auto Flow Control Enable Bit = 1
RxDE(UCR[1]: Rx DMA Enable bit = 1
DTL (AFT[3:0]) : nRTS Deassert Trigger Level = 12
ATL (AFT[7:4]) : nRTS Assert Trigger Level = 11
RXT (FCR[7:6]) : Rx Available FiFo Trigger Level = 8

DMA Register Set Value
TYPE(CHCTRL[9:8]) : H/W transfer with level sensitive = 2'b11
SYNC(CHCTRL[13]) : Syncronize External Request = 1
BSIZE(CHCTRL[7:6] : 8Read / 8Write = 2'b11 (depend on RXT)



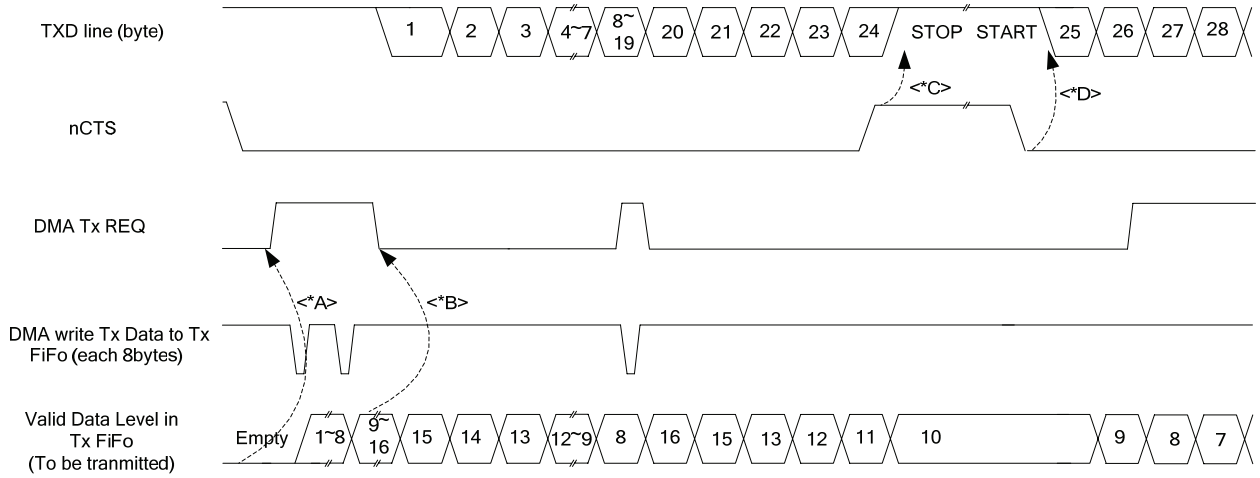
<*A> : DMA Rx Request is asserted by RXT
<*B> : DMA detects Rx Request but DMA can't transfer data (Other master higer than DMA occupies the bus)
<*C> : nRTS is reached by DTL
<*D> : The nRTS latency is between External UART communicated with TCCxx and TCCxxx.
The nRTS latency should be considered. if not, overrun error will occur
<*E> : nRTS is asserted by ATL
<*F> : DMA Rx Request is Deasserted By RxT

Figure 19.8 RX operation with DMA and AFC

19.2.6 TX Operation with DMA and AFC

UART Register Set Value
FE (FCR[0]) : FiFo Enable Bit = 1
TxDE(UCR[0] : Tx DMA Enable bit = 1
TXT (FCR[5:4]) : Tx FiFo Trigger Level = 8

DMA Register Set Value
TYPE(CHCTRL[9:8]) : H/W transfer with level sensitive = 2'b11
SYNC(CHCTRL[13]) : Syncronize External Request = 1
BSIZE(CHCTRL[7:6] : 8Read / 8Write = 2'b11 (depend on TXT)



<*A> : DMA Tx Request is asserted by TXT
<*B> : DMA Tx Request is deasserted by TXT

Figure 19.9 TX operation with DMA and AFC

19.3 Register Description

The base address of UART channel 0, 1, 2, 3, 4, and 5 are 0xB0102000, 0xB0102100, 0xB0102200, 0xB0102300, 0xB0102400, and 0xB0102500 respectively.

Table 19.1 UART Register Map

Name	Address	RW	Reset	Description
RBR	0x00	R	Unknown	Receiver Buffer Register(DLAB = 0)
THR	0x00	W	0x00	Transmitter Holding Register (DLAB=0)
DLL	0x00	R/W	0x00	Divisor Latch (LSB) (DLAB=1)
IER	0x04	R/W	0x00	Interrupt Enable Register (DLAB=0)
DLM	0x04	R/W	0x00	Divisor Latch (MSB) (DLAB=1)
IIR	0x08	R	Unknown	Interrupt Ident. Register (DLAB=0)
FCR	0x08	W	0xC0	FIFO Control Register (DLAB=1)
LCR	0x0C	R/W	0x03	Line Control Register
MCR	0x10	R/W	0x00	MODEM Control Register
LSR	0x14	R	Unknown	Line Status Register
MSR	0x18	R	Unknown	MODEM Status Register
SCR	0x1C	R/W	0x00	Scratch Register
AFT	0x20	R/W	0x00	AFC Trigger Level Register
UCR	0x24	R/W	0x00	UART Control Register
SRBR	0x40	R	Unknown	Rx Buffer Register
STHR	0x44	W	0x00	Transmitter Holding Register
SDLL	0x48	R/W	0x00	Divisor Latch (LSB)
SDLM	0x4C	R/W	0x00	Divisor Latch (MSB)
SIER	0x50	R/W	0x00	Interrupt Enable Register
SCCR	0x60	R/W	0x00	Smart Card Control Register
STC	0x64	R/W	0x00	Smart Card TX Count Register
IRCFG	0x80	R/W	0x00	IRDA Configuration Register

The base address of “Port Mux Registers” is 0xB0102600.

Table 19.2 UART Port Mux Register

Name	Address	RW	Reset	Description
CHSEL	0x00	R/W	0x3210	Channel Selection Register
CHST	0x00	R	0x0000	Channel Status Register

19.3.1 UART Controller Register

Receiver Buffer Register (RBR) **0xB0102n[30]00(DLAB=0)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Received Data (when reading)							

Field	Name	RW	Reset	Description
7-0	RxD	R	0x00	Received Data (when reading)

The RBR is actually the16byte FIFO, a received data from external device is stored in the RBR and CPU(or DMA) can read this register by Rx interrupt(or Rx DMA Request).

Transmitter Holding Register (THR) **0xB0102n00(DLAB=0)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Transmitting Data (when writing)							

Field	Name	RW	Reset	Description
7-0	TxD	W	-	Transmitting Data (when writing)

The THR is actually the 16byte FIFO. To transmit data to external device, CPU(or DMA) should write data to the THR.

Divisor Latch Register (DLL) **0xB0102n00(DLAB=1)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Divisor Latch LSB							

Field	Name	RW	Reset	Description
7-0	DLL	R/W	0x00	Divisor Latch LSB

Divisor Latch Register (DLM) **0xB0102n04(DLAB=1)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Divisor Latch MSB							

Field	Name	RW	Reset	Description
7-0	DLM	R/W	0x00	Divisor Latch MSB

This is for generation of the desired baud rate clock.

The value can be calculated as follows.

{DLM, DLL} = fUART / (16 * desired baud rate)

For example,

If UART clock frequency is 48MHz (from CKC) and the baud-rate you want is 115,200 bps, the divisor value should be 26(48M/ (115200 x16)) in decimal.

In UART interface with general DMA, the UART clock frequency (from CKC) and configuration should be set by the following description.

- f_{UART} frequency
- CHCTRL.BST of GDMA = 0 (DMA Arbitration Mode Enable).

[30] n = Channel Number 0 ~ 5.

f_{UART} x 4 >= f_{BUS}

- CHCTRL.BST of GDMA = 1 (DMA Arbitration Mode Disable).

f_{UART} x 3 >= f_{BUS}

The following setting value(DMA/UART setting value) is example of RX/TX operation with DMA for the NVS2310.

- GDMA Setting Value

CHCTRL.BSIZE = 0 : 1Read/1Write (every UART request).

CHCTRL.WSIZE = 0 : 8bit data (1 byte).

- UART Setting Value

FCR.TXT = 0 : TX Empty Condition in TX FIFO.

FCR.RXT = 0 : Rx Available data Condition (minimum 1 character available data is received in Rx FIFO).

Interrupt Register (IER) **0xB0102n[31]04(DLAB=0)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EMSI	ELSI	ETXI	ERXI

Field	Name	RW	Reset	Description
3	EMSI	R/W	0x0	0 : Disable MODEM status interrupt 1 : Enable MODEM status interrupt
2	ELSI	R/W	0x0	0 : Disable receiver line status interrupt 1 : Enable receiver line status interrupt
1	ETXI	R/W	0x0	0 : Disable transmitter holding register empty interrupt 1 : Enable transmitter holding register empty interrupt
0	ERXI	R/W	0x0	0 : Disable received data available interrupt 1 : Enable received data available interrupt

[31] n = Channel Number 0 ~5.

Interrupt Ident. Register (IIR)

0xB0102n[32]08

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				STF											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								0	0	0	0	IID[2:0]			IPF

Field	Name	RW	Reset	Description
27	STF	R	0x0	Smart Card TX done Flag 0: Normal 1: Tx done
3-1	IID	R	0x0	IID [3:1] / Type / Interrupt ID 011 / R / Receiver line status (Overrun/parity error/framing error/break error) To Clear : Reading the line status register 010 / R / Received data available or trigger level reached To Clear : Reading the receiver buffer register or the FIFO drops below the trigger level 110 / R / Character timeout indication (While the last 4 characters are received, no characters have been removed from or input to the RX FIFO and there is at least 1 character in the RX FIFO) To Clear : Reading the receiver buffer register 001 / R / Transmitter holding register empty To Clear : Reading the IIR register(if source of interrupt) or writing into the transmitter holding register 000 / R / MODEM status (Clear to send data set ready or ring indicator or data carrier detect) To Clear : Reading the MODEM status register
0	IPF	R	0x0	0 : Interrupt pending 1 : Interrupt has not generated

FIFO Control Register (FCR)

0xB0102n.[33]08

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								RXT[1:0]		TXT[1:0]		DRTE	TXFR	RXFR	FE

Field	Name	RW	Reset	Description
7-6	RXT	W	0x3	00 : 1 byte 01 : 4 bytes 10 : 8 bytes 11 : 14 bytes
5-4	TXT	W	0x0	00 : Possible to Write 16 byte (EMPTY) 01 : Possible to Write 8 byte 10 : Possible to Write 4 byte 11 : Possible to Write 1 byte
3	DRTE	W	0x0	0 : DMA transfer is depend on RxDE or TxDE bit. 1 : Enable both Rx & Tx DMA transfer (Regardless of RxDE or TxDE status)
2	TXFR	W	0x0	1 : Reset TX FIFO counter and FIFO data
1	RXFR	W	0x0	1 : Reset RX FIFO counter and FIFO data
0	FE	W	0x0	1 : Enable TX FIFO and RX FIFO.

[32] n = Channel Number 0 ~ 5.
[33] n = Channel Number 0 ~ 5.

Line Control Register (LCR) 0xB0102n[34]0C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								DLAB	SB	SP	EPS	PEN	STB	WLS[1:0]	

Field	Name	RW	Reset	Description
7	DLAB	R/W	0x0	0 : Access the receiver buff, the transmitter holding register, or the interrupt enable register. 1 : Access the divisor latches of the baud generator
6	SB	R/W	0x0	0 : Disable the break 1 : The serial output is forced to the spacing(logic 0) state
5	SP	R/W	0x0	0 : Disable stick parity 1 : When PEN[3], EPS[4], and SP[5] are set to 1, the parity bit is transmitted and checked as a logic 0. If PEN[3] and SP[5] are set to 1 and EPS[4] is set to 0, then the parity bit is transmitted and checked as a logic 1.
4	EPS	R/W	0x0	0 : Generate or check odd parity. 1 : Generate or check even parity.
3	Pen	R/W	0x0	0 : Parity generation/check disable 1 : A parity bit is generated(TX) or checked(RX).
2	STB	R/W	0x0	0 : One stop bit is generated in the transmitted data. 1 : When 5-bit word length is selected, one and a half stop bits are generated. When either 6, 7, or 8-bit word length is selected, two stop bits are generated.
1-0	WLS	R/W	0x3	00 : 5 bits 01 : 6 bits 10 : 7 bits 11 : 8 bits

Modem Control Register (MCR) 0xB0102n[35]10

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									RS	AFE	LOOP	-	-	RTS	-

Field	Name	RW	Reset	Description
6	RS	R/W	0x1	0 : nRTS is de-asserted at the Rx Stop Condition 1 : nRTS is de-asserted at the Rx Start Condition (Reset value)
5	AFE	R/W	0x0	0 : Disable Automatic Flow Control 1 : Enable Automatic Flow Control
4	LOOP	R/W	0x0	0 : Disable local loop back feature 1 : Enable local loop back feature
1	RTS	R/W	0x0	0 : Set the nRTS line to high state 1 : Reset the nRTS line to low state

[34] n = Channel Number 0 ~ 5.
[35] n = Channel Number 0 ~ 5.

Line Status Register (LSR)

0xB0102n[36]14

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								ERF	TEMT	THRE	BI	FE	PE	OE	DR

Field	Name	RW	Reset	Description
7	ERF	R	Unknown	0 : In the 16450 mode 1 : In the FIFO mode this bit is set when there is at least one parity error, framing error or break indication in the FIFO
6	TEMT	R	Unknown	0 : The transmitter shift register is not empty 1 : Transmitter holding register and the transmitter shift register are both empty
5	THRE	R	Unknown	0 : UART is not ready to accept a new char for transmission 1 : UART is ready to accept a new char for transmission
4	BI	R	Unknown	0 : The received data input is normal 1 : The received data input is held in the spacing(logic 0) state for longer than a full word transmission time
3	FE	R	Unknown	0 : The received data input is normal 1 : The received character did not have a valid stop bit
2	PE	R	Unknown	0 : The received data input is normal 1 : The received data character does not have the correct even or odd parity
1	OE	R	Unknown	0 : The received data input is normal 1 : The receiver buffer register was not read by the CPU before the next character was transferred into the receiver buffer register.
0	DR	R	Unknown	0 : The received data is not ready 1 : The received data is ready

Modem Status Register(MSR)

0xB0102n.[37]18

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								reserved			CTS	reserved			DCTS

Field	Name	RW	Reset	Description
4	CTS	R	Unknown	0 : CTS input is “0”. 1 : CTS input is “1”.
0	DCTS	R	Unknown	0 : CTS input is not changed. 1 : CTS input is changed.

Scratch Register (SCR)

0xB0102n1C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								Scratch Data[7:0]							

Field	Name	RW	Reset	Description
7-0	SCR	R/W	0x00	Scratch Data

This 8-bit Read/Write Register does not control the UART in anyway. It is intended as a scratchpad register to be used by the programmer to hold data temporarily.

[36] n = Channel Number 0 ~ 5.
[37] n = Channel Number 0 ~ 5.

AFC Trigger Level Register (AFT) 0xB0102n20

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ATL[3:0]				DTL[3:0]			

Field	Name	RW	Reset	Description
7-4	ATL	R/W	0x0	N : nRTS assert trigger level.
3-0	DTL	R/W	0x0	N : nRTS de-assert trigger level.

Control Register (UCR) 0xB0102n[38]24

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RWA	TWA	RxDE	TxDE

Field	Name	RW	Reset	Description
3	RWA	R/W	0x0	0 : Rx FIFO access to byte 1 : Rx FIFO access to word (4 bytes)
2	TWA	R/W	0x0	0 : Tx FIFO access to byte 1 : Tx FIFO access to word (4 bytes)
1	RxDE	R/W	0x0	0 : Rx DMA disable 1 : Rx DMA enable
0	TxDE	R/W	0x0	0 : Tx DMA disable 1 : Tx DMA enable

Receiver Buffer Register (SRBR) 0xB0102n[39]40

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Received Data (Read Only)							

Field	Name	RW	Reset	Description
7-0	RxD	R	0	Received Data (Read Only)

Transmitter Holding Register (STHR) 0xB0102n44

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Transmitting Data (Write Only)							

Field	Name	RW	Reset	Description
7-0	TxD	W	-	Transmitting Data (Write Only)

Divisor Latch Register (SDLL) 0xB0102n48

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Divisor Latch LSB							

Field	Name	RW	Reset	Description
7-0	DLL	R/W	0	Divisor Latch LSB

[38] n = Channel Number 0 ~ 5.
[39] n = Channel Number 0 ~ 5.

Divisor Latch Register (SDLM) 0xB0102n4C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Divisor Latch MSB							

Field	Name	RW	Reset	Description
7-0	DLM	R/W	0	Divisor Latch MSB

Interrupt Register (SIER) 0xB0102n50

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EMSI	ELSI	ETXI	ERXI

SRBR, STHR, SDLL, SDLM, SIER registers are copy of the RBR, THR, DLL, DLM, IER registers. These registers can be accessed without concern of DLAB state.

Field	Name	RW	Reset	Description
3	EMSI	R/W	0x0	0 : Disable MODEM status interrupt 1 : Enable MODEM status interrupt
2	ELSI	R/W	0x0	0 : Disable receiver line status interrupt 1 : Enable receiver line status interrupt
1	ETXI	R/W	0x0	0 : Disable transmitter holding register empty interrupt 1 : Enable transmitter holding register empty interrupt
0	ERXI	R/W	0x0	0 : Disable received data available interrupt 1 : Enable received data available interrupt

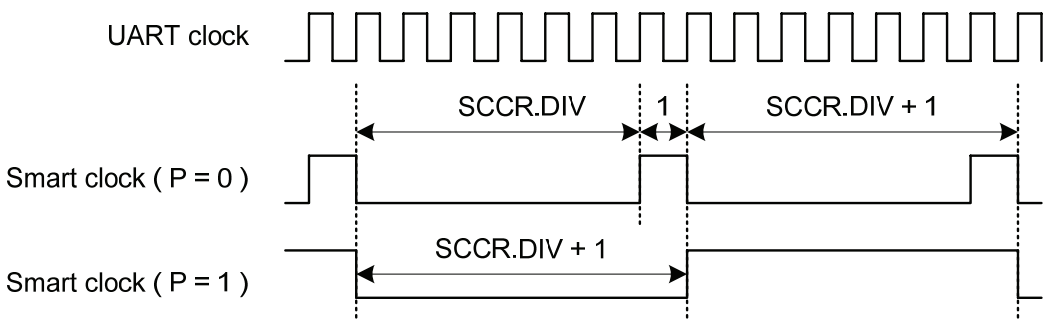
SmartCard Configuration Register (SCCR) **0xB0102n[40]60**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEN	STEN	DEN	P	STF	STE										
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIV [15:0]															

Field	Name	RW	Reset	Description
31	SEN	R/W	0x0	1 : The smart card interface is enabled
30	STEN	R/W	0x0	1 : Enable
29	DEN	R/W	0x0	1 : Enable
28	P	R/W	0x0	1 : Enable
27	STF	R/W	0x0	STF / Type / SmartCard Tx done Flag 1 / R / It represents that all characters specified by S_CNT are transmitted. It can be TX done interrupt source. 1 / W / When '1' is written to it, it is cleared.
26	STE	R/W	0x0	1 : Enable Tx done Interrupt
15-0	DIV	R/W	0x0000	1 : Fout = Fuart/((DIV+1)*2^P)

Fout and Fuart can not be the same. Therefore, In the case that DIV = 0 or DEN = 0, Smart clock operates only under the condition that P is set to 1 (P = 1).

DEN	DIV	P	Smart clock
0	x	0	Disabled
0	x	1	Fout = Fuart / 2
1	0	0	Disabled
1	0	1	Fout = Fuart / 2
1	N (!= 0)	P	Fout = Fuart / ((N + 1) * 2 ^ P)

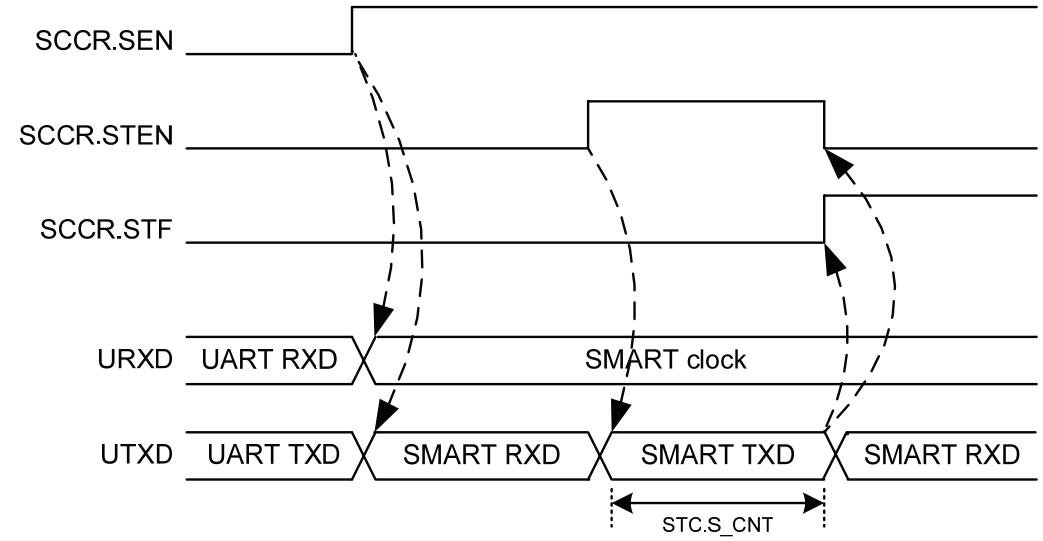


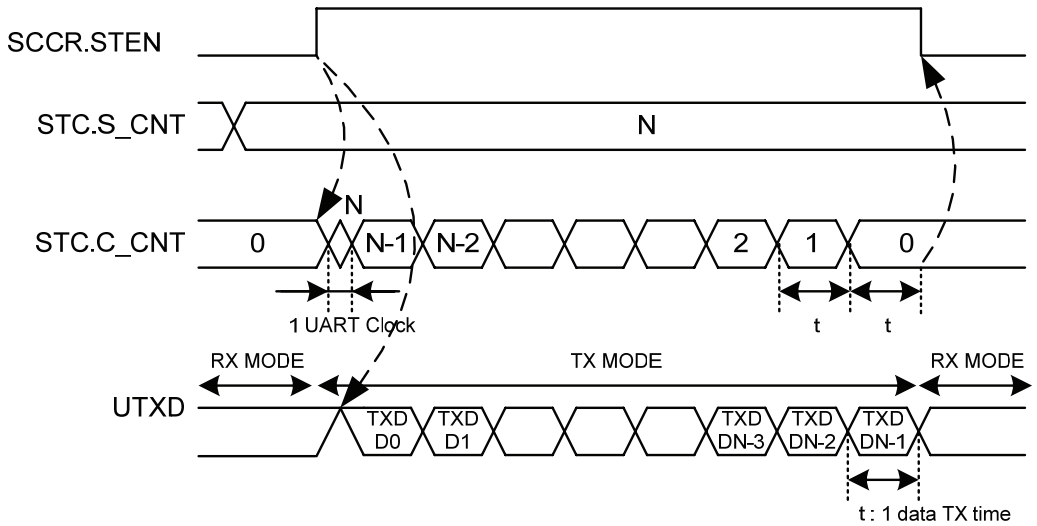
[40] n = Channel Number 0 ~ 5.

SmartCard Tx Count (STC) **0xB0102n64**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C_CNT[15:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S_CNT[15:0]															

Field	Name	RW	Reset	Description
31-16	C_CNT	R	0x0000	N : It represents current TX count when smart card interface is enabled
15-0	S_CNT	R/W	0x0000	N : When smart card interface is enabled, it specifies TX Count





IRDA Configuration Register (IRCFG) **0xB0102n[41]80**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												RXP	TXP	RXE	TXE

Field	Name	RW	Reset	Description
3	RXP	R/W	0x0	0 : Do not invert polarity of IrDA-RXD signal 1 : Invert polarity of IrDA-RXD signal
2	TXP	R/W	0x0	0 : Do not invert polarity of IrDA-TXD signal 1 : Invert polarity of IrDA-TXD signal
1	RXE	R/W	0x0	0 : Disable IrDA-Rx 1 : Enable IrDA-Rx
0	TXE	R/W	0x0	0 : Disable IrDA-Tx 1 : Enable IrDA-Tx

In IrDA Mode, each zero bit of TXD has a pulse width of 3/16 of a bit time.

19.3.2 Port Mux Register

NVS2310 has 6 UART ports, and each port can be allocated for one of UART channels. Base address is 0xF0055400

Channel Selection Register (CHSEL) **0xB0102600**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0									CH5			0	CH4		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	CH3			0	CH2			0	CH1			0	CH0		

Field	Name	RW	Reset	Description
22-20	CH5	R/W	0x5	N : UART Hw5 mapping to UART Port n
18-16	CH4	R/W	0x4	N : UART Hw4 mapping to UART Port n
14-12	CH3	R/W	0x3	N : UART Hw3 mapping to UART Port n
10-8	CH2	R/W	0x2	N : UART Hw2 mapping to UART Port n
6-4	CH1	R/W	0x1	N : UART Hw1 mapping to UART Port n
2-0	CH0	R/W	0x0	N : UART Hw0 mapping to UART Port n

The value of CH0, CH1, CH2, CH3, CH4, and CH 5 is the corresponding port number. Therefore, when CH0 is set to 3, UART channel0 is connected to UART port 3. Note that the value for each channel should be different.

Channel Status Register **0xB0102604**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										CH5	CH4	CH3	CH2	CH1	CH0

Field	Name	RW	Reset	Description
5	CH5	R	0x0	UART Hw5 Interrupt Flag
4	CH4	R	0x0	UART Hw4 Interrupt Flag
3	CH3	R	0x0	UART Hw3 Interrupt Flag
2	CH2	R	0x0	UART Hw2 Interrupt Flag
1	CH1	R	0x0	UART Hw1 Interrupt Flag
0	CH0	R	0x0	UART Hw0 Interrupt Flag

When one or more UART channels are issuing an interrupt request, the corresponding bit is set to 1.

[41] n = Channel Number 0 ~ 5.

20 VPIC(Vectored Priority Interrupt Controller)

20.1 Overview

The following figure represents the block diagram of interrupt controller. The interrupt controller can manage up to 64 interrupt sources. In the NVS2310, there are 12 external interrupt sources that can be detected various kind of method that is a rising edge/ falling edge / level high / level low. The external interrupt sources can be selected among the various GPIO ports and fed reliably into interrupt controller with dedicated noise filters.

There are two types of interrupt in ARM family processor; IRQ type, FIQ type. Interrupt controller can select these two types for each interrupt sources separately.

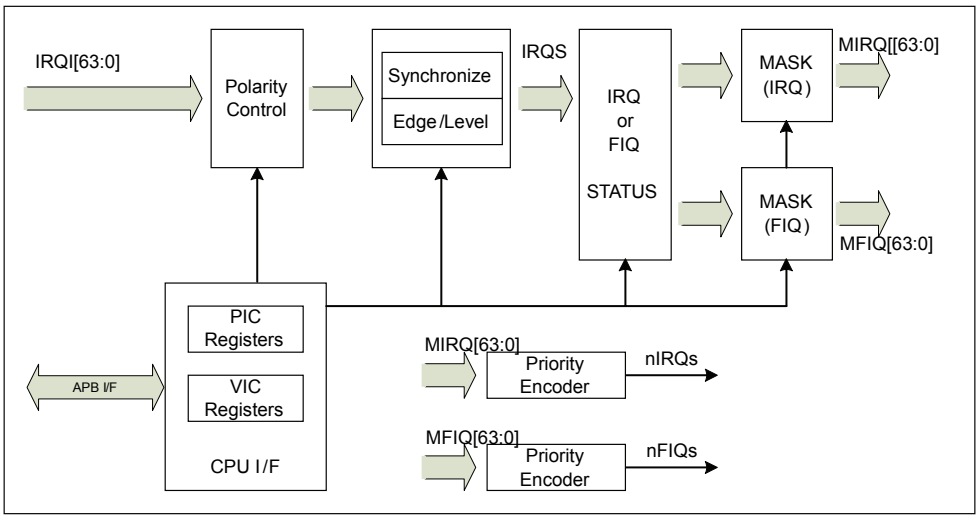


Figure 20.1 Block Diagram of Vectored Interrupt Controller

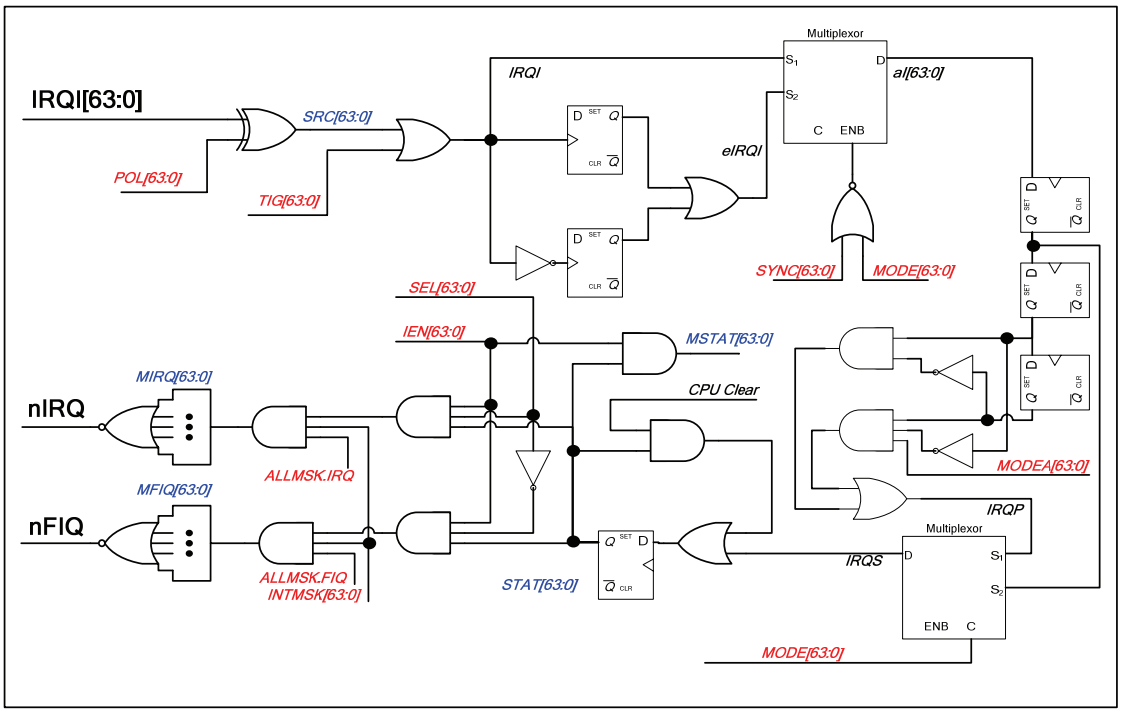


Figure 20.2 Detailed Interrupt Flow

The above figure shows the detailed interrupt flow from IRQI[63:0] to nIRQ or nFIQ. The red colored signals are configuration register fields and the blue colored ones are status registers describe in the register description. **The MIRQ and MFIQ are used to encode the vector number and vector address with pre-defined priorities in the vectored interrupt controller**.

20.2 Signal Descriptions

20.2.1 Global Signals

Table 20.1 Global Signals

Name	Direction	Descriptions	Related Blocks
PCLK	Input	Peripheral Bus Clock	CKC
PRESETn	Input	Reset Signal	CKC

20.2.2 Interrupt Source Signals

Table 20.2 Clock Source Signals

Name	Direction	Descriptions	Related Blocks
IRQI[63:0]	Input	Interrupt Sources from on-chip devices or external interrupts.	Each Block or GPIOs

20.2.3 Interrupt Output Signals (to ARM)

Table 20.3 Interrupt Output Signals (to ARM)

Name	Direction	Descriptions	Related Blocks
nFIQ	Output	This is FIQ signal to ARM processor and is active low and level sensitive. This goes to LOW just after interrupt occurred and goes to HIGH just after clearing the interrupt status.	ARM
nIRQ	Output	This is IRQ signal to ARM processor and is active low and level sensitive. This goes to LOW just after interrupt occurred and goes to HIGH just after clearing the interrupt status.	ARM

20.2.4 Timer Related Signals

Table 20.4 Interrupt Output Signals (to ARM)

Name	Direction	Descriptions	Related Blocks
IRQP[0]	Output	This is the interrupt pulse signal to counting in the timer. The corresponding source is IRQI[0]. The pulse can be generated at the rising edge or falling edge or both edges.	TIMER
IRQP[1]	Output	This is the interrupt pulse signal to counting in the timer. The corresponding source is IRQI[1]. The pulse can be generated at the rising edge or falling edge or both edges.	TIMER
IRQP[2]	Output	This is the interrupt pulse signal to counting in the timer. The corresponding source is IRQI[2]. The pulse can be generated at the rising edge or falling edge or both edges.	TIMER
IRQP[3]	Output	This is the interrupt pulse signal to counting in the timer. The corresponding source is IRQI[3]. The pulse can be generated at the rising edge or falling edge or both edges.	TIMER

20.3 Register Descriptions

Table 20.5 Priority Interrupt Controller Register Map (Base Address = 0xB0600000)

Name	Address	RW	Reset	Description
IEN0	0x000	R/W	0x00000000	Interrupt Enable0 Register
IEN1	0x004	R/W	0x00000000	Interrupt Enable1 Register
CLR0	0x008	R/W	0x00000000	Interrupt Clear0 Register
CLR1	0x00C	R/W	0x00000000	Interrupt Clear1 Register
STS0	0x010	R	Unknown	Interrupt Status0 Register
STS1	0x014	R	Unknown	Interrupt Status1 Register
SEL0	0x018	R/W	0x00000000	IRQ or FIQ Selection0 Register
SEL1	0x01C	R/W	0x00000000	IRQ or FIQ Selection1 Register
SRC0	0x020	R	Unknown	Source Interrupt Status0 Register
SRC1	0x024	R	Unknown	Source Interrupt Status1 Register
MSTS0	0x028	R	0x00000000	Masked Status0 Register
MSTS1	0x02C	R	0x00000000	Masked Status1 Register
TIG0	0x030	R/W	0x00000000	Test Interrupt Generation0 Register
TIG1	0x034	R/W	0x00000000	Test Interrupt Generation1 Register
POL0	0x038	R/W	0x00000000	Interrupt Polarity0 Register
POL1	0x03C	R/W	0x00000000	Interrupt Polarity1 Register
IRQ0	0x040	R	0x00000000	IRQ Raw Status0 Register
IRQ1	0x044	R	0x00000000	IRQ Raw Status1 Register
FIQ0	0x048	R	Unknown	FIQ Status0 Register
FIQ1	0x04C	R	Unknown	FIQ Status1 Register
MIRQ0	0x050	R	0x00000000	Masked IRQ Status0 Register
MIRQ1	0x054	R	0x00000000	Masked IRQ Status1 Register
MFIQ0	0x058	R	0x00000000	Masked FIQ Status0 Register
MFIQ1	0x05C	R	0x00000000	Masked FIQ Status1 Register
MODE0	0x060	R/W	0x00000000	Trigger Mode0 Register – Level or Edge
MODE1	0x064	R/W	0x00000000	Trigger Mode1 Register – Level or Edge
SYNC0	0x068	R/W	0xFFFFFFFF	Synchronization Enable0 Register
SYNC1	0x06C	R/W	0xFFFFFFFF	Synchronization Enable1 Register
WKEN0	0x070	R/W	0x00000000	Wakeup Event Enable0 Register
WKEN1	0x074	R/W	0x00000000	Wakeup Event Enable1 Register
MODEA0	0x078	R/W	0x00000000	Both Edge or Single Edge0 Register
MODEA1	0x07C	R/W	0x00000000	Both Edge or Single Edge1 Register
INTMSK0	0x100	R/W	0xFFFFFFFF	Interrupt Output Masking0 Register
INTMSK1	0x104	R/W	0xFFFFFFFF	Interrupt Output Masking1 Register
ALLMSK	0x108	R/W	0x00000003	All Mask Register

Table 20.6 Vectored Interrupt Controller Register Map (Base Address = 0xB0600200)

Name	Address	Type	Reset	Description
VAIRQ	0x200	R	0x800000XX	IRQ Vector Register
VAFIQ	0x204	R	0x800000XX	FIQ Vector Register
VNIRQ	0x208	R	0x800000XX	IRQ Vector Number Register
VNFIQ	0x20C	R	0x800000XX	FIQ Vector Number Register
VCTRL	0x210	R/W	0x00000000	Vector Control Register
PRIO0	0x220	R/W	0x03020100	Priorities for Interrupt 0 ~ 3
PRIO1	0x224	R/W	0x07060504	Priorities for Interrupt 4 ~ 7
PRIO2	0x228	R/W	0x0B0A0908	Priorities for Interrupt 8 ~ 11
PRIO3	0x22C	R/W	0x0F0E0D0C	Priorities for Interrupt 12 ~ 15
PRIO4	0x230	R/W	0x13121110	Priorities for Interrupt 16 ~ 19
PRIO5	0x234	R/W	0x17161514	Priorities for Interrupt 20 ~ 23
PRIO6	0x238	R/W	0x1B1A1918	Priorities for Interrupt 24 ~ 27
PRIO7	0x23C	R/W	0x1F1E1D1C	Priorities for Interrupt 28 ~ 31
PRIO8	0x240	R/W	0x23222120	Priorities for Interrupt 32 ~ 35
PRIO9	0x244	R/W	0x27262524	Priorities for Interrupt 36 ~ 39
PRIO10	0x248	R/W	0x2B2A2928	Priorities for Interrupt 40 ~ 43
PRIO11	0x24C	R/W	0x2F2E2D2C	Priorities for Interrupt 44 ~ 47
PRIO12	0x250	R/W	0x33323130	Priorities for Interrupt 48 ~ 51
PRIO13	0x254	R/W	0x37363534	Priorities for Interrupt 52 ~ 55
PRIO14	0x258	R/W	0x3B3A3938	Priorities for Interrupt 56 ~ 59
PRIO15	0x25C	R/W	0x3F3E3D3C	Priorities for Interrupt 60 ~ 63

20.3.1 Priority Interrupt Controller

Interrupt Enable Register (IEN0) **0xB0600000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host Interrupt 0 enable
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 interrupt enable
28	TSADC	R/W	0x0	TSADC interrupt enable
27	OM	R/W	0x0	Overlay mixer Interrupt Enable
26	3DMMU	R/W	0x0	3D MMU interrupt enable
25	3DGP	R/W	0x0	3D Geometry Processor interrupt enable
24	3DPP	R/W	0x0	3D Pixel Processor interrupt enable
23	VCDC	R/W	0x0	Video CODEC interrupt enable
22	ISP2	R/W	0x0	ISP Interrupt 2
21	JPGE	R/W	0x0	JPEG Encoder interrupt enable
20	VIPET	R/W	0x0	Video Enhancer interrupt enable
19	LCD1	R/W	0x0	LCD controller 1 interrupt enable
18	LCD0	R/W	0x0	LCD controller 0 interrupt enable
17	CIF	R/W	0x0	Camer Interface interrupt enable
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 interrupt enable
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 interrupt enable
14	EI11	R/W	0x0	External interrupt 11 enable
13	EI10	R/W	0x0	External interrupt 10 enable
12	EI9	R/W	0x0	External interrupt 9 enable
11	EI8	R/W	0x0	External interrupt 8 enable
10	EI7	R/W	0x0	External interrupt 7 enable
9	EI6	R/W	0x0	External interrupt 6 enable
8	EI5	R/W	0x0	External interrupt 5 enable
7	EI4	R/W	0x0	External interrupt 4 enable
6	EI3	R/W	0x0	External interrupt 3 enable
5	EI2	R/W	0x0	External interrupt 2 enable
4	EI1	R/W	0x0	External interrupt 1 enable
3	EI0	R/W	0x0	External interrupt 0 enable
2	RTC	R/W	0x0	RTC interrupt enable
1	TC32	R/W	0x0	Timer32 interrupt enable
0	TC0	R/W	0x0	Timer 0 interrupt enable

Interrupt Enable Register (IEN1) 0xB0600004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Note: For each bit, '1' means that corresponding interrupt is enabled and '0' for disabled

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1
29	SD2	R/W	0x0	SD/MMC 2 interrupt enable
28	APMU	R/W	0x0	ARM System Metrics interrupt enable Note: the interrupt request signal is active low.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) interrupt enable
26	ADMA0	R/W	0x0	Audio0(7.1ch) DMA interrupt enable
25	AUDIO1	R/W	0x0	Audio1(streo) interrupt enable
24	ISP3	R/W	0x0	ISP Interrupt 3
23	GMAC	R/W	0x0	GMAC interrupt enable
22	TSIF1	R/W	0x0	TS interface 1 interrupt enable
21	TSIF0	R/W	0x0	TS interface 0 interrupt enable
20	CIPHER	R/W	0x0	CIPHER interrupt enable
19	ADMA1	R/W	0x0	Audio1(stereo) DMA interrupt enable
18	GDMA1	R/W	0x0	GDMA1interrupt enable(HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 interrupt enable(HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 interrupt enable(PBUS)
15	UART	R/W	0x0	UART interrupt controller
14	SD3	R/W	0x0	SD/MMC 3 interrupt enable
13	SD1	R/W	0x0	SD/MMC 1 interrupt enable
12	SD0	R/W	0x0	SD/MMC 0 interrupt enable
11	SMUI2C	R/W	0x0	SMU I2C controller enable
10	RMT	R/W	0x0	Remote Control interrupt enable
9	NFC	R/W	0x0	Nand flash controller interrupt enable
8	MS	R/W	0x0	Memory stick controller interrupt enable
7	CKC	R/W	0x0	CKC interrupt enable
6	I2C	R/W	0x0	I2C interrupt enable
4	GPSB	R/W	0x0	GPSB Interrupt enable
2	HDMI	R/W	0x0	HDMI interrupt enable
0	EHI1	R/W	0x0	External Host interrupt 1 enable

The interrupt controller can receive up to 12 external interrupts simultaneously, which are EI0 ~ EI11. And each external interrupt can become one of the following 64 interrupt sources. EINTSEL0, EINTSEL1 and EINTSEL2 registers are used for selecting these interrupt sources.

Table 20.7 Sources of External Interrupt

NUM	External Interrupt Sources	NUM	External Interrupt Sources	NUM	External Interrupt Sources
0	GPIOA[0]	32	GPIOC[30]	64	GPIOG[11]
1	GPIOA[1]	33	GPIOC[31]	65	GPIOG[12]
2	GPIOA[2]	34	GPIOD[15]	66	GPIOG[17]
3	GPIOA[3]	35	GPIOD[16]	67	GPIOG[22]
4	GPIOA[4]	36	GPIOD[17]	68	GPIOG[23]
5	GPIOA[5]	37	GPIOD[18]	69	GPIOG[24]
6	GPIOA[6]	38	GPIOD[19]	70	GPIOG[25]
7	GPIOA[7]	39	GPIOD[20]	71	GPIOG[26]
8	GPIOA[8]	40	GPIOD[21]	72	GPIOG[27]
9	GPIOA[9]	41	GPIOD[22]	73	GPIOG[28]
10	GPIOA[10]	42	GPIOD[23]	74	GPIOG[29]
11	GPIOA[11]	43	GPIOD[24]	75	TSWKU
12	GPIOA[12]	44	GPIOD[25]	76	TSSTOP
13	GPIOA[13]	45	GPIOE[10]	77	TSUPDN
14	GPIOA[14]	46	GPIOE[11]	78	RMWKUP
15	GPIOA[15]	47	GPIOE[24]	79	PMKUP
16	GPIOA[16]	48	GPIOE[25]	80	USB1_VBON
17	GPIOA[17]	49	GPIOE[26]	81	USB1_VBOFF
18	GPIOB[17]	50	GPIOE[27]	82	USB0_VBON
19	GPIOB[19]	51	GPIOE[28]	83	USB0_VBOFF
20	GPIOB[26]	52	GPIOE[29]		
21	GPIOB[27]	53	GPIOE[30]		
22	GPIOB[30]	54	GPIOE[31]		
23	GPIOB[31]	55	GPIOF[16]		
24	GPIOC[18]	56	GPIOF[17]		
25	GPIOC[19]	57	GPIOF[26]		
26	GPIOC[20]	58	GPIOF[27]		
27	GPIOC[21]	59	GPIOF[28]		
28	GPIOC[22]	60	GPIOG[4]		
29	GPIOC[23]	61	GPIOG[5]		
30	GPIOC[28]	62	GPIOG[6]		
31	GPIOC[29]	63	GPIOG[7]		

EINTSELn (n= 0,…,11) **0xB010A204 + 4*n(n=0,...,11)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	EINT[4*n+3]SEL								EINT[4*n+3]SEL						
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	EINT[4*n+3]SEL								EINT[4*n+3]SEL						

Field	Name	RW	Reset	Description
29-24	EINT[4*n+3]SEL	R/W	0x3	External Interrupt 3 Selection Value
21-16	EINT[4*n+3]SEL	R/W	0x2	External Interrupt 2 Selection Value
13-8	EINT[4*n+3]SEL	R/W	0x1	External Interrupt 1 Selection Value
5-0	EINT[4*n+3]SEL	R/W	0x0	External Interrupt 0 Selection Value

EINTSELn (n=0,...,11) selects External Interrupt source in Table 20.7

Interrupt Clear Register (CLR0) 0xB0600008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0	ECC	DMA	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	JPGE	JPGD	VIPET	LCD1	LCD0	CAM	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	SMUI2C	TC1	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host Interrupt 0 clear
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 interrupt clear
28	TSADC	R/W	0x0	TSADC interrupt clear
27	OM	R/W	0x0	Overlay mixer Interrupt clear
26	3DMMU	R/W	0x0	3D MMU interrupt clear
25	3DGP	R/W	0x0	3D Geometry Processor interrupt clear
24	3DPP	R/W	0x0	3D Pixel Processor interrupt clear
23	VCDC	R/W	0x0	Video CODEC interrupt clear
22	ISP2	R/W	0x0	ISP Interrupt 2 clear
21	JPGE	R/W	0x0	JPEG Encoder interrupt clear
20	VIPET	R/W	0x0	Video Enhancer interrupt clear
19	LCD1	R/W	0x0	LCD controller 1 interrupt clear
18	LCD0	R/W	0x0	LCD controller 0 interrupt clear
17	CIF	R/W	0x0	Camer Interface interrupt clear
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 interrupt clear
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 interrupt clear
14	EI11	R/W	0x0	External interrupt 11 clear
13	EI10	R/W	0x0	External interrupt 10 clear
12	EI9	R/W	0x0	External interrupt 9 clear
11	EI8	R/W	0x0	External interrupt 8 clear
10	EI7	R/W	0x0	External interrupt 7 clear
9	EI6	R/W	0x0	External interrupt 6 clear
8	EI5	R/W	0x0	External interrupt 5 clear
7	EI4	R/W	0x0	External interrupt 4 clear
6	EI3	R/W	0x0	External interrupt 3 clear
5	EI2	R/W	0x0	External interrupt 2 clear
4	EI1	R/W	0x0	External interrupt 1 clear
3	EI0	R/W	0x0	External interrupt 0 clear
2	RTC	R/W	0x0	RTC interrupt clear
1	TC32	R/W	0x0	Timer32 interrupt clear
0	TC0	R/W	0x0	Timer 0 interrupt clear

Note: For each bit, the interrupt status is cleared by writing ‘1’ for corresponding bit.

Interrupt Clear Register (CLR1) 0xB060000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 clear
29	SD2	R/W	0x0	SD/MMC 2 interrupt clear
28	APMU	R/W	0x0	ARM System Metrics interrupt clear.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) interrupt clear
26	ADMA0	R/W	0x0	Audio0(7.1ch) DMA interrupt clear
25	AUDIO1	R/W	0x0	Audio1(streo) interrupt clear
24	ISP3	R/W	0x0	ISP Interrupt 3 clear
23	GMAC	R/W	0x0	GMAC interrupt clear
22	TSIF1	R/W	0x0	TS interface 1 interrupt clear
21	TSIF0	R/W	0x0	TS interface 0 interrupt clear
20	CIPHER	R/W	0x0	CIPHER interrupt clear
19	ADMA1	R/W	0x0	Audio1(stereo) DMA interrupt clear
18	GDMA1	R/W	0x0	GDMA1interrupt clear (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 interrupt clear (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 interrupt clear (PBUS)
15	UART	R/W	0x0	UART interrupt clear
14	SD3	R/W	0x0	SD/MMC 3 interrupt clear
13	SD1	R/W	0x0	SD/MMC 1 interrupt clear
12	SD0	R/W	0x0	SD/MMC 0 interrupt clear
11	SMUI2C	R/W	0x0	SMU I2C controller clear
10	RMT	R/W	0x0	Remote Control interrupt clear
9	NFC	R/W	0x0	Nand flash controller interrupt clear
8	MS	R/W	0x0	Memory stick controller interrupt clear
7	CKC	R/W	0x0	CKC interrupt clear
6	I2C	R/W	0x0	I2C interrupt clear
4	GPSB	R/W	0x0	GPSB Interrupt clear
2	HDMI	R/W	0x0	HDMI interrupt clear
0	EHI1	R/W	0x0	External Host interrupt 1 clear

Note: For each bit, the interrupt status is cleared by writing ‘1’ for corresponding bit.

Interrupt Status Register (STS0) 0xB0600010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R	-	External Host Interrupt 0 status
30	-	-	-	-
29	GDMA0	R	-	General DMA 0 interrupt status
28	TSADC	R	-	TSADC interrupt status
27	OM	R	-	Overlay mixer Interrupt status
26	3DMMU	R	-	3D MMU interrupt status
25	3DGP	R	-	3D Geometry Processor interrupt status
24	3DPP	R	-	3D Pixel Processor interrupt status
23	VCDC	R	-	Video CODEC interrupt status
22	ISP2	R	-	ISP Interrupt 2 status
21	JPGE	R	-	JPEG Encoder interrupt status
20	VIPET	R	-	Video Enhancer interrupt status
19	LCD1	R	-	LCD controller 1 interrupt status
18	LCD0	R	-	LCD controller 0 interrupt status
17	CIF	R	-	Camer Interface interrupt status
16	SC1	R	-	Mem-to-Mem scaler 1 interrupt status
15	SC0	R	-	Mem-to-Mem scaler 0 interrupt status
14	EI11	R	-	External interrupt 11 status
13	EI10	R	-	External interrupt 10 status
12	EI9	R	-	External interrupt 9 status
11	EI8	R	-	External interrupt 8 status
10	EI7	R	-	External interrupt 7 status
9	EI6	R	-	External interrupt 6 status
8	EI5	R	-	External interrupt 5 status
7	EI4	R	-	External interrupt 4 status
6	EI3	R	-	External interrupt 3 status
5	EI2	R	-	External interrupt 2 status
4	EI1	R	-	External interrupt 1 status
3	EI0	R	-	External interrupt 0 status
2	RTC	R	-	RTC interrupt status
1	TC32	R	-	Timer32 interrupt status
0	TC0	R	-	Timer 0 interrupt status

Note: For each bit, if it is ‘1’, the corresponding interrupt was activated.

Interrupt Status Register (STS1) 0xB0600014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R	-	ISP Interrupt 1 clear
29	SD2	R	-	SD/MMC 2 interrupt clear
28	APMU	R	-	ARM System Metrics interrupt clear.
27	AUDIO0	R	-	Audio0(7.1ch) interrupt clear
26	ADMA0	R	-	Audio0(7.1ch) DMA interrupt clear
25	AUDIO1	R	-	Audio1(streo) interrupt clear
24	ISP3	R	-	ISP Interrupt 3 clear
23	GMAC	R	-	GMAC interrupt clear
22	TSIF1	R	-	TS interface 1 interrupt clear
21	TSIF0	R	-	TS interface 0 interrupt clear
20	CIPHER	R	-	CIPHER interrupt clear
19	ADMA1	R	-	Audio1(stereo) DMA interrupt clear
18	GDMA1	R	-	GDMA1interrupt clear (HSBUS)
17	UOTG1	R	-	USB 2.0 OTG1 interrupt clear (HSBUS)
16	UOTG0	R	-	USB 2.0 OTG0 interrupt clear (PBUS)
15	UART	R	-	UART interrupt clear
14	SD3	R	-	SD/MMC 3 interrupt clear
13	SD1	R	-	SD/MMC 1 interrupt clear
12	SD0	R	-	SD/MMC 0 interrupt clear
11	SMUI2C	R	-	SMU I2C controller clear
10	RMT	R	-	Remote Control interrupt clear
9	NFC	R	-	Nand flash controller interrupt clear
8	MS	R	-	Memory stick controller interrupt clear
7	CKC	R	-	CKC interrupt clear
6	I2C	R	-	I2C interrupt clear
4	GPSB	R	-	GPSB Interrupt clear
2	HDMI	R	-	HDMI interrupt clear
0	EHI1	R	-	External Host interrupt 1 clear

Note: For each bit, if it is ‘1’, the corresponding interrupt was activated.

Interrupt Select Register (SEL0) 0xB0600018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host IRQ or FIQ Selection Register
30				
29	GDMA0	R/W	0x0	General DMA 0 IRQ or FIQ Selection Register
28	TSADC	R/W	0x0	TSADC IRQ or FIQ Selection Register
27	OM	R/W	0x0	Overlay mixer IRQ or FIQ Selection Register
26	3DMMU	R/W	0x0	3D MMU IRQ or FIQ Selection Register
25	3DGP	R/W	0x0	3D Geometry Processor IRQ or FIQ Selection Register
24	3DPP	R/W	0x0	3D Pixel Processor IRQ or FIQ Selection Register
23	VCDC	R/W	0x0	Video CODEC IRQ or FIQ Selection Register
22	ISP2	R//W	0x0	ISP Interrupt 2 IRQ or FIQ Selection Register
21	JPGE	R/W	0x0	JPEG Encoder IRQ or FIQ Selection Register
20	VIPET	R/W	0x0	Video Enhancer IRQ or FIQ Selection Register
19	LCD1	R/W	0x0	LCD controller 1 IRQ or FIQ Selection Register
18	LCD0	R/W	0x0	LCD controller 0 IRQ or FIQ Selection Register
17	CIF	R/W	0x0	Camer Interface IRQ or FIQ Selection Register
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 IRQ or FIQ Selection Register
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 IRQ or FIQ Selection Register
14	EI11	R/W	0x0	External interrupt 11 IRQ or FIQ Selection Register
13	EI10	R/W	0x0	External interrupt 10 IRQ or FIQ Selection Register
12	EI9	R/W	0x0	External interrupt 9 IRQ or FIQ Selection Register
11	EI8	R/W	0x0	External interrupt 8 IRQ or FIQ Selection Register
10	EI7	R/W	0x0	External interrupt 7 IRQ or FIQ Selection Register
9	EI6	R/W	0x0	External interrupt 6 IRQ or FIQ Selection Register
8	EI5	R/W	0x0	External interrupt 5 IRQ or FIQ Selection Register
7	EI4	R/W	0x0	External interrupt 4 IRQ or FIQ Selection Register
6	EI3	R/W	0x0	External interrupt 3 IRQ or FIQ Selection Register
5	EI2	R/W	0x0	External interrupt 2 IRQ or FIQ Selection Register
4	EI1	R/W	0x0	External interrupt 1 IRQ or FIQ Selection Register
3	EI0	R/W	0x0	External interrupt 0 IRQ or FIQ Selection Register
2	RTC	R/W	0x0	RTC IRQ or FIQ Selection Register
1	TC32	R/W	0x0	Timer32 IRQ or FIQ Selection Register
0	TC0	R/W	0x0	Timer 0 IRQ or FIQ Selection Register

Note: For each bit, if it is ‘1’, the corresponding interrupt is propagated to nIRQ otherwise to nFIQ.

Interrupt Select Register (SEL1) 0xB060001C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 interrupt IRQ or FIQ Selection
29	SD2	R/W	0x0	SD/MMC 2 interrupt interrupt IRQ or FIQ Selection
28	APMU	R/W	0x0	ARM System Metrics interrupt interrupt IRQ or FIQ Selection
27	AUDIO0	R/W	0x0	Audio0(7.1ch) interrupt interrupt IRQ or FIQ Selection
26	ADMA0	R/W	0x0	Audio0(7.1ch) DMA interrupt interrupt IRQ or FIQ Selection
25	AUDIO1	R/W	0x0	Audio1(streo) interrupt interrupt IRQ or FIQ Selection
24	ISP3	R/W	0x0	ISP Interrupt 3 interrupt IRQ or FIQ Selection
23	GMAC	R/W	0x0	GMAC interrupt interrupt IRQ or FIQ Selection
22	TSIF1	R/W	0x0	TS interface 1 interrupt interrupt IRQ or FIQ Selection
21	TSIF0	R/W	0x0	TS interface 0 interrupt interrupt IRQ or FIQ Selection
20	CIPHER	R/W	0x0	CIPHER interrupt interrupt IRQ or FIQ Selection
19	ADMA1	R/W	0x0	Audio1(stereo) DMA interrupt interrupt IRQ or FIQ Selection
18	GDMA1	R/W	0x0	GDMA1interrupt interrupt IRQ or FIQ Selection (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 interrupt interrupt IRQ or FIQ Selection (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 interrupt interrupt IRQ or FIQ Selection (PBUS)
15	UART	R/W	0x0	UART interrupt interrupt IRQ or FIQ Selection
14	SD3	R/W	0x0	SD/MMC 3 interrupt interrupt IRQ or FIQ Selection
13	SD1	R/W	0x0	SD/MMC 1 interrupt interrupt IRQ or FIQ Selection
12	SD0	R/W	0x0	SD/MMC 0 interrupt interrupt IRQ or FIQ Selection
11	SMUI2C	R/W	0x0	SMU I2C controller interrupt IRQ or FIQ Selection
10	RMT	R/W	0x0	Remote Control interrupt interrupt IRQ or FIQ Selection
9	NFC	R/W	0x0	Nand flash controller interrupt interrupt IRQ or FIQ Selection
8	MS	R/W	0x0	Memory stick controller interrupt IRQ or FIQ Selection
7	CKC	R/W	0x0	CKC interrupt IRQ or FIQ Selection
6	I2C	R/W	0x0	I2C interrupt IRQ or FIQ Selection
4	GPSB	R/W	0x0	GPSB Interrupt IRQ or FIQ Selection
2	HDMI	R/W	0x0	HDMI interrupt IRQ or FIQ Selection
0	EHI1	R/W	0x0	External Host interrupt IRQ or FIQ Selection

Note: For each bit, if it is '1', the corresponding interrupt is propagated to nIRQ otherwise to nFIQ.

Interrupt Source Status Register (SRC0) 0xB0600020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host Interrupt 0 source status
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 interrupt source status
28	TSADC	R/W	0x0	TSADC interrupt source status
27	OM	R/W	0x0	Overlay mixer Interrupt source status
26	3DMMU	R/W	0x0	3D MMU interrupt source status
25	3DGP	R/W	0x0	3D Geometry Processor interrupt source status
24	3DPP	R/W	0x0	3D Pixel Processor interrupt source status
23	VCDC	R/W	0x0	Video CODEC interrupt source status
22	ISP2	R/W	0x0	ISP Interrupt 2 source status
21	JPGE	R/W	0x0	JPEG Encoder interrupt source status
20	VIPET	R/W	0x0	Video Enhancer interrupt source status
19	LCD1	R/W	0x0	LCD controller 1 interrupt source status
18	LCD0	R/W	0x0	LCD controller 0 interrupt source status
17	CIF	R/W	0x0	Camer Interface interrupt source status
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 interrupt source status
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 interrupt source status
14	EI11	R/W	0x0	External interrupt 11 source status
13	EI10	R/W	0x0	External interrupt 10 source status
12	EI9	R/W	0x0	External interrupt 9 source status
11	EI8	R/W	0x0	External interrupt 8 source status
10	EI7	R/W	0x0	External interrupt 7 source status
9	EI6	R/W	0x0	External interrupt 6 source status
8	EI5	R/W	0x0	External interrupt 5 source status
7	EI4	R/W	0x0	External interrupt 4 source status
6	EI3	R/W	0x0	External interrupt 3 source status
5	EI2	R/W	0x0	External interrupt 2 source status
4	EI1	R/W	0x0	External interrupt 1 source status
3	EI0	R/W	0x0	External interrupt 0 source status
2	RTC	R/W	0x0	RTC interrupt source status
1	TC32	R/W	0x0	Timer32 interrupt source status
0	TC0	R/W	0x0	Timer 0 interrupt source status

Note: This represents the status for each interrupt source by XOR with interrupt input and polarity register for the corresponding interrupt source.

Interrupt Source Status Register (SRC1) 0xB0600024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1
29	SD2	R/W	0x0	SD/MMC 2 interrupt source status
28	APMU	R/W	0x0	ARM System Metrics interrupt source status
27	AUDIO0	R/W	0x0	Audio0(7.1ch) interrupt source status
26	ADMA0	R/W	0x0	Audio0(7.1ch) DMA interrupt source status
25	AUDIO1	R/W	0x0	Audio1(streo) interrupt source status
24	ISP3	R/W	0x0	ISP Interrupt 3 source status
23	GMAC	R/W	0x0	GMAC interrupt source status
22	TSIF1	R/W	0x0	TS interface 1 interrupt source status
21	TSIF0	R/W	0x0	TS interface 0 interrupt source status
20	CIPHER	R/W	0x0	CIPHER interrupt source status
19	ADMA1	R/W	0x0	Audio1(stereo) DMA interrupt source status
18	GDMA1	R/W	0x0	GDMA1interrupt source status (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 interrupt source status (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 interrupt source status (PBUS)
15	UART	R/W	0x0	UART interrupt source status
14	SD3	R/W	0x0	SD/MMC 3 interrupt source status
13	SD1	R/W	0x0	SD/MMC 1 interrupt source status
12	SD0	R/W	0x0	SD/MMC 0 interrupt source status
11	SMUI2C	R/W	0x0	SMU I2C controller source status
10	RMT	R/W	0x0	Remote Control interrupt source status
9	NFC	R/W	0x0	Nand flash controller interrupt source status
8	MS	R/W	0x0	Memory stick controller interrup source status
7	CKC	R/W	0x0	CKC interrupt source status
6	I2C	R/W	0x0	I2C interrupt source status
4	GPSB	R/W	0x0	GPSB Interrupt source status
2	HDMI	R/W	0x0	HDMI interrupt source status
0	EHI1	R/W	0x0	External Host interrupt 1 source status

Note: This represents the status for each interrupt source by XOR with interrupt input and polarity register for the corresponding interrupt source.

Masked Interrupt Status Register (MSTS0) 0xB0600028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host masked interrupt status
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 masked interrupt status
28	TSADC	R/W	0x0	TSADC masked interrupt status
27	OM	R/W	0x0	Overlay mixer masked interrupt status
26	3DMMU	R/W	0x0	3D MMU masked interrupt status
25	3DGP	R/W	0x0	3D Geometry Processor masked interrupt status
24	3DPP	R/W	0x0	3D Pixel Processor masked interrupt status
23	VCDC	R/W	0x0	Video CODEC masked interrupt status
22	ISP2	R/W	0x0	ISP Interrupt 2 masked interrupt status
21	JPGE	R/W	0x0	JPEG Encoder masked interrupt status
20	VIPET	R/W	0x0	Video Enhancer masked interrupt status
19	LCD1	R/W	0x0	LCD controller 1 masked interrupt status
18	LCD0	R/W	0x0	LCD controller 0 masked interrupt status
17	CIF	R/W	0x0	Camer Interface masked interrupt status
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 masked interrupt status
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 masked interrupt status
14	EI11	R/W	0x0	External interrupt 11 masked interrupt status
13	EI10	R/W	0x0	External interrupt 10 masked interrupt status
12	EI9	R/W	0x0	External interrupt 9 masked interrupt status
11	EI8	R/W	0x0	External interrupt 8 masked interrupt status
10	EI7	R/W	0x0	External interrupt 7 masked interrupt status
9	EI6	R/W	0x0	External interrupt 6 masked interrupt status
8	EI5	R/W	0x0	External interrupt 5 masked interrupt status
7	EI4	R/W	0x0	External interrupt 4 masked interrupt status
6	EI3	R/W	0x0	External interrupt 3 masked interrupt status
5	EI2	R/W	0x0	External interrupt 2 masked interrupt status
4	EI1	R/W	0x0	External interrupt 1 masked interrupt status
3	EI0	R/W	0x0	External interrupt 0 masked interrupt status
2	RTC	R/W	0x0	RTC masked interrupt status
1	TC32	R/W	0x0	Timer32 masked interrupt status
0	TC0	R/W	0x0	Timer 0 masked interrupt status

Note: This register represents the interrupt status for enabled sources. If a interrupt is not enabled, the corresponding bit is '0'.

Masked Interrupt Status Register (MSTS1) 0xB060002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP masked interrupt status 1
29	SD2	R/W	0x0	SD/MMC 2 masked interrupt status
28	APMU	R/W	0x0	ARM System Metrics masked interrupt status
27	AUDIO0	R/W	0x0	Audio0(7.1ch) masked interrupt status
26	ADMA0	R/W	0x0	Audio0(7.1ch) DMA masked interrupt status
25	AUDIO1	R/W	0x0	Audio1(streo) masked interrupt status
24	ISP3	R/W	0x0	ISP Interrupt 3 masked interrupt status
23	GMAC	R/W	0x0	GMAC masked interrupt status
22	TSIF1	R/W	0x0	TS interface 1 masked interrupt status
21	TSIF0	R/W	0x0	TS interface 0 masked interrupt status
20	CIPHER	R/W	0x0	CIPHER masked interrupt status
19	ADMA1	R/W	0x0	Audio1(stereo) DMA masked interrupt status
18	GDMA1	R/W	0x0	GDMA1 masked interrupt status (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 masked interrupt status (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 masked interrupt status (PBUS)
15	UART	R/W	0x0	UART masked interrupt status
14	SD3	R/W	0x0	SD/MMC 3 masked interrupt status
13	SD1	R/W	0x0	SD/MMC 1 masked interrupt status
12	SD0	R/W	0x0	SD/MMC 0 masked interrupt status
11	SMUI2C	R/W	0x0	SMU I2C controller masked interrupt status
10	RMT	R/W	0x0	Remote Control masked interrupt status
9	NFC	R/W	0x0	Nand flash controller masked interrupt status
8	MS	R/W	0x0	Memory stick controller masked interrupt status
7	CKC	R/W	0x0	CKC masked interrupt status
6	I2C	R/W	0x0	I2C masked interrupt status
4	GPSB	R/W	0x0	GPSB masked interrupt status
2	HDMI	R/W	0x0	HDMI masked interrupt status
0	EHI1	R/W	0x0	External Host 1 masked interrupt status

Note: This register represents the interrupt status for enabled sources. If a interrupt is not enabled, the corresponding bit is ‘0’.

Test Interrupt Source Register (TIG0) 0xB0600030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host test interrupt source
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 test interrupt source
28	TSADC	R/W	0x0	TSADC test interrupt source
27	OM	R/W	0x0	Overlay mixer test interrupt source
26	3DMMU	R/W	0x0	3D MMU test interrupt source
25	3DGP	R/W	0x0	3D Geometry Processor test interrupt source
24	3DPP	R/W	0x0	3D Pixel Processor test interrupt source
23	VCDC	R/W	0x0	Video CODEC test interrupt source
22	ISP2	R/W	0x0	ISP Interrupt 2 test interrupt source
21	JPGE	R/W	0x0	JPEG Encoder test interrupt source
20	VIPET	R/W	0x0	Video Enhancer test interrupt source
19	LCD1	R/W	0x0	LCD controller 1 test interrupt source
18	LCD0	R/W	0x0	LCD controller 0 test interrupt source
17	CIF	R/W	0x0	Camer Interface test interrupt source
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 test interrupt source
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 test interrupt source
14	EI11	R/W	0x0	External interrupt 11 test interrupt source
13	EI10	R/W	0x0	External interrupt 10 test interrupt source
12	EI9	R/W	0x0	External interrupt 9 test interrupt source
11	EI8	R/W	0x0	External interrupt 8 test interrupt source
10	EI7	R/W	0x0	External interrupt 7 test interrupt source
9	EI6	R/W	0x0	External interrupt 6 test interrupt source
8	EI5	R/W	0x0	External interrupt 5 test interrupt source
7	EI4	R/W	0x0	External interrupt 4 test interrupt source
6	EI3	R/W	0x0	External interrupt 3 test interrupt source
5	EI2	R/W	0x0	External interrupt 2 test interrupt source
4	EI1	R/W	0x0	External interrupt 1 test interrupt source
3	EI0	R/W	0x0	External interrupt 0 test interrupt source
2	RTC	R/W	0x0	RTC test interrupt source
1	TC32	R/W	0x0	Timer32 test interrupt source
0	TC0	R/W	0x0	Timer 0 test interrupt source

Note: In the case of the corresponding bit in SRC register is ‘1’, the interrupt is activated by writing ‘1’.

Test Interrupt Source Register (TIG1) 0xB0600034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 test interrupt source
29	SD2	R/W	0x0	SD/MMC 2 test interrupt source
28	APMU	R/W	0x0	ARM System Metrics test interrupt source.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) test interrupt source
26	ADMA0	R/W	0x0	Audio0(7.1ch) test interrupt source
25	AUDIO1	R/W	0x0	Audio1(streo) test interrupt source
24	ISP3	R//W	0x0	ISP Interrupt 3 test interrupt source
23	GMAC	R/W	0x0	GMAC test interrupt source
22	TSIF1	R/W	0x0	TS interface 1 test interrupt source
21	TSIF0	R/W	0x0	TS interface 0 test interrupt source
20	CIPHER	R/W	0x0	CIPHER test interrupt source
19	ADMA1	R/W	0x0	Audio1(stereo) DMA test interrupt source
18	GDMA1	R/W	0x0	GDMA1 test interrupt source (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 test interrupt source (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 test interrupt source (PBUS)
15	UART	R/W	0x0	UART test interrupt source
14	SD3	R/W	0x0	SD/MMC 3 test interrupt source
13	SD1	R/W	0x0	SD/MMC 1 test interrupt source
12	SD0	R/W	0x0	SD/MMC 0 test interrupt source
11	SMUI2C	R/W	0x0	SMU I2C controller test interrupt source
10	RMT	R/W	0x0	Remote Control test interrupt source
9	NFC	R/W	0x0	Nand flash controller test interrupt source
8	MS	R/W	0x0	Memory stick controller test interrupt source
7	CKC	R/W	0x0	CKC test interrupt source
6	I2C	R/W	0x0	I2C test interrupt source
4	GPSB	R/W	0x0	GPSB test interrupt source
2	HDMI	R/W	0x0	HDMI test interrupt source
0	EHI1	R/W	0x0	External Host test interrupt source

Note: In the case of the corresponding bit in SRC register is ‘1’, the interrupt is activated by writing ‘1’.

Interrupt Polarity Control Register (POL0) 0xB0600038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host interrupt polarity
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 interrupt polarity
28	TSADC	R/W	0x0	TSADC interrupt polarity
27	OM	R/W	0x0	Overlay mixer interrupt polarity
26	3DMMU	R/W	0x0	3D MMU interrupt polarity
25	3DGP	R/W	0x0	3D Geometry Processor interrupt polarity
24	3DPP	R/W	0x0	3D Pixel Processor interrupt polarity
23	VCDC	R/W	0x0	Video CODEC interrupt polarity
22	ISP2	R//W	0x0	ISP Interrupt 2 interrupt polarity
21	JPGE	R/W	0x0	JPEG Encoder interrupt polarity
20	VIPET	R/W	0x0	Video Enhancer interrupt polarity
19	LCD1	R/W	0x0	LCD controller 1 interrupt polarity
18	LCD0	R/W	0x0	LCD controller 0 interrupt polarity
17	CIF	R/W	0x0	Camer Interface interrupt polarity
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 interrupt polarity
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 interrupt polarity
14	EI11	R/W	0x0	External interrupt 11 interrupt polarity
13	EI10	R/W	0x0	External interrupt 10 interrupt polarity
12	EI9	R/W	0x0	External interrupt 9 interrupt polarity
11	EI8	R/W	0x0	External interrupt 8 interrupt polarity
10	EI7	R/W	0x0	External interrupt 7 interrupt polarity
9	EI6	R/W	0x0	External interrupt 6 interrupt polarity
8	EI5	R/W	0x0	External interrupt 5 interrupt polarity
7	EI4	R/W	0x0	External interrupt 4 interrupt polarity
6	EI3	R/W	0x0	External interrupt 3 interrupt polarity
5	EI2	R/W	0x0	External interrupt 2 interrupt polarity
4	EI1	R/W	0x0	External interrupt 1 interrupt polarity
3	EI0	R/W	0x0	External interrupt 0 interrupt polarity
2	RTC	R/W	0x0	RTC interrupt polarity
1	TC32	R/W	0x0	Timer32 interrupt polarity
0	TC0	R/W	0x0	Timer 0 interrupt polarity

Note: If the interrupt signal is active-high, the corresponding bit should be ‘0’, otherwise ‘1’ – active-low.

Interrupt Polarity Control Register (POL1) 0xB060003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 interrupt polarity
29	SD2	R/W	0x0	SD/MMC 2 interrupt polarity
28	APMU	R/W	0x0	ARM System Metrics interrupt polarity.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) interrupt polarity
26	ADMA0	R/W	0x0	Audio0(7.1ch) interrupt polarity
25	AUDIO1	R/W	0x0	Audio1(streo) interrupt polarity
24	ISP3	R/W	0x0	ISP Interrupt 3 interrupt polarity
23	GMAC	R/W	0x0	GMAC interrupt polarity
22	TSIF1	R/W	0x0	TS interface 1 interrupt polarity
21	TSIF0	R/W	0x0	TS interface 0 interrupt polarity
20	CIPHER	R/W	0x0	CIPHER interrupt polarity
19	ADMA1	R/W	0x0	Audio1(stereo) DMA interrupt polarity
18	GDMA1	R/W	0x0	GDMA1 interrupt polarity (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 interrupt polarity (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 interrupt polarity (PBUS)
15	UART	R/W	0x0	UART interrupt polarity
14	SD3	R/W	0x0	SD/MMC 3 interrupt polarity
13	SD1	R/W	0x0	SD/MMC 1 interrupt polarity
12	SD0	R/W	0x0	SD/MMC 0 interrupt polarity
11	SMUI2C	R/W	0x0	SMU I2C controller interrupt polarity
10	RMT	R/W	0x0	Remote Control interrupt polarity
9	NFC	R/W	0x0	Nand flash controller interrupt polarity
8	MS	R/W	0x0	Memory stick controller interrupt polarity
7	CKC	R/W	0x0	CKC interrupt polarity
6	I2C	R/W	0x0	I2C interrupt polarity
4	GPSB	R/W	0x0	GPSB interrupt polarity
2	HDMI	R/W	0x0	HDMI interrupt polarity
0	EHI1	R/W	0x0	External Host interrupt polarity

Note: If the interrupt signal is active-high, the corresponding bit should be ‘0’, otherwise ‘1’ – active-low.

IRQ Raw Status Register (IRQ0) 0xB0600040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host IRQ raw status
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 IRQ raw status
28	TSADC	R/W	0x0	TSADC IRQ raw status
27	OM	R/W	0x0	Overlay mixer IRQ raw status
26	3DMMU	R/W	0x0	3D MMU IRQ raw status
25	3DGP	R/W	0x0	3D Geometry Processor IRQ raw status
24	3DPP	R/W	0x0	3D Pixel Processor IRQ raw status
23	VCDC	R/W	0x0	Video CODEC IRQ raw status
22	ISP2	R/W	0x0	ISP Interrupt 2 IRQ raw status
21	JPGE	R/W	0x0	JPEG Encoder IRQ raw status
20	VIPET	R/W	0x0	Video Enhancer IRQ raw status
19	LCD1	R/W	0x0	LCD controller 1 IRQ raw status
18	LCD0	R/W	0x0	LCD controller 0 IRQ raw status
17	CIF	R/W	0x0	Camer Interface IRQ raw status
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 IRQ raw status
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 IRQ raw status
14	EI11	R/W	0x0	External interrupt 11 IRQ raw status
13	EI10	R/W	0x0	External interrupt 10 IRQ raw status
12	EI9	R/W	0x0	External interrupt 9 IRQ raw status
11	EI8	R/W	0x0	External interrupt 8 IRQ raw status
10	EI7	R/W	0x0	External interrupt 7 IRQ raw status
9	EI6	R/W	0x0	External interrupt 6 IRQ raw status
8	EI5	R/W	0x0	External interrupt 5 IRQ raw status
7	EI4	R/W	0x0	External interrupt 4 IRQ raw status
6	EI3	R/W	0x0	External interrupt 3 IRQ raw status
5	EI2	R/W	0x0	External interrupt 2 IRQ raw status
4	EI1	R/W	0x0	External interrupt 1 IRQ raw status
3	EI0	R/W	0x0	External interrupt 0 IRQ raw status
2	RTC	R/W	0x0	RTC IRQ raw status
1	TC32	R/W	0x0	Timer32 IRQ raw status
0	TC0	R/W	0x0	Timer 0 IRQ raw status

Note: This represents the raw status for IRQ register.

IRQ Raw Status Register (IRQ1) 0xB0600044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 IRQ raw status
29	SD2	R/W	0x0	SD/MMC 2 IRQ raw status
28	APMU	R/W	0x0	ARM System Metrics IRQ raw status.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) IRQ raw status
26	ADMA0	R/W	0x0	Audio0(7.1ch) IRQ raw status
25	AUDIO1	R/W	0x0	Audio1(streo) IRQ raw status
24	ISP3	R/W	0x0	ISP Interrupt 3 IRQ raw status
23	GMAC	R/W	0x0	GMAC IRQ raw status
22	TSIF1	R/W	0x0	TS interface 1 IRQ raw status
21	TSIF0	R/W	0x0	TS interface 0 IRQ raw status
20	CIPHER	R/W	0x0	CIPHER IRQ raw status
19	ADMA1	R/W	0x0	Audio1(stereo) DMA IRQ raw status
18	GDMA1	R/W	0x0	GDMA1 IRQ raw status (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 IRQ raw status (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 IRQ raw status (PBUS)
15	UART	R/W	0x0	UART IRQ raw status
14	SD3	R/W	0x0	SD/MMC 3 test IRQ raw status
13	SD1	R/W	0x0	SD/MMC 1 IRQ raw status
12	SD0	R/W	0x0	SD/MMC 0 IRQ raw status
11	SMUI2C	R/W	0x0	SMU I2C IRQ raw status
10	RMT	R/W	0x0	Remote Control IRQ raw status
9	NFC	R/W	0x0	Nand flash controller IRQ raw status
8	MS	R/W	0x0	Memory stick controller IRQ raw status
7	CKC	R/W	0x0	CKC IRQ raw status
6	I2C	R/W	0x0	I2C IRQ raw status
4	GPSB	R/W	0x0	GPSB IRQ raw status
2	HDMI	R/W	0x0	HDMI IRQ raw status
0	EHI1	R/W	0x0	External Host IRQ raw status

Note: This represents the raw status for IRQ register.

FIQ Raw Status Register (FIQ0) 0xB0600048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host FIQ raw status
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 FIQ raw status
28	TSADC	R/W	0x0	TSADC FIQ raw status
27	OM	R/W	0x0	Overlay mixer FIQ raw status
26	3DMMU	R/W	0x0	3D MMU FIQ raw status
25	3DGP	R/W	0x0	3D Geometry Processor FIQ raw status
24	3DPP	R/W	0x0	3D Pixel Processor FIQ raw status
23	VCDC	R/W	0x0	Video CODEC FIQ raw status
22	ISP2	R/W	0x0	ISP Interrupt 2 FIQ raw status
21	JPGE	R/W	0x0	JPEG Encoder FIQ raw status
20	VIPET	R/W	0x0	Video Enhancer FIQ raw status
19	LCD1	R/W	0x0	LCD controller 1 FIQ raw status
18	LCD0	R/W	0x0	LCD controller 0 FIQ raw status
17	CIF	R/W	0x0	Camer Interface FIQ raw status
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 FIQ raw status
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 FIQ raw status
14	EI11	R/W	0x0	External interrupt 11 FIQ raw status
13	EI10	R/W	0x0	External interrupt 10 FIQ raw status
12	EI9	R/W	0x0	External interrupt 9 FIQ raw status
11	EI8	R/W	0x0	External interrupt 8 FIQ raw status
10	EI7	R/W	0x0	External interrupt 7 FIQ raw status
9	EI6	R/W	0x0	External interrupt 6 FIQ raw status
8	EI5	R/W	0x0	External interrupt 5 FIQ raw status
7	EI4	R/W	0x0	External interrupt 4 FIQ raw status
6	EI3	R/W	0x0	External interrupt 3 FIQ raw status
5	EI2	R/W	0x0	External interrupt 2 FIQ raw status
4	EI1	R/W	0x0	External interrupt 1 FIQ raw status
3	EI0	R/W	0x0	External interrupt 0 FIQ raw status
2	RTC	R/W	0x0	RTC FIQ raw status
1	TC32	R/W	0x0	Timer32 FIQ raw status
0	TC0	R/W	0x0	Timer 0 FIQ raw status

Note: This represents the raw status for FIQ register.

FIQ Raw Status Register (FIQ1) 0xB060004C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 FIQ raw status
29	SD2	R/W	0x0	SD/MMC 2 FIQ raw status
28	APMU	R/W	0x0	ARM System Metrics FIQ raw status.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) FIQ raw status
26	ADMA0	R/W	0x0	Audio0(7.1ch) FIQ raw status
25	AUDIO1	R/W	0x0	Audio1(streo) FIQ raw status
24	ISP3	R/W	0x0	ISP Interrupt 3 FIQ raw status
23	GMAC	R/W	0x0	GMAC FIQ raw status
22	TSIF1	R/W	0x0	TS interface 1 FIQ raw status
21	TSIF0	R/W	0x0	TS interface 0 FIQ raw status
20	CIPHER	R/W	0x0	CIPHER FIQ raw status
19	ADMA1	R/W	0x0	Audio1(stereo) DMA FIQ raw status
18	GDMA1	R/W	0x0	GDMA1 FIQ raw status (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 FIQ raw status (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 FIQ raw status (PBUS)
15	UART	R/W	0x0	UART FIQ raw status
14	SD3	R/W	0x0	SD/MMC 3 test IRQ raw status
13	SD1	R/W	0x0	SD/MMC 1 FIQ raw status
12	SD0	R/W	0x0	SD/MMC 0 FIQ raw status
11	SMUI2C	R/W	0x0	SMU I2C FIQ raw status
10	RMT	R/W	0x0	Remote Control FIQ raw status
9	NFC	R/W	0x0	Nand flash controller FIQ raw status
8	MS	R/W	0x0	Memory stick controller FIQ raw status
7	CKC	R/W	0x0	CKC FIQ raw status
6	I2C	R/W	0x0	I2C FIQ raw status
4	GPSB	R/W	0x0	GPSB FIQ raw status
2	HDMI	R/W	0x0	HDMI FIQ raw status
0	EHI1	R/W	0x0	External Host FIQ raw status

Note: This represents the raw status for FIQ register.

Masked IRQ Status Register (MIRQ0) 0xB0600050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host masked IRQ status register
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 masked IRQ status register
28	TSADC	R/W	0x0	TSADC masked IRQ status register
27	OM	R/W	0x0	Overlay mixer masked IRQ status register
26	3DMMU	R/W	0x0	3D MMU masked IRQ status register
25	3DGP	R/W	0x0	3D Geometry Processor masked IRQ status register
24	3DPP	R/W	0x0	3D Pixel Processor masked IRQ status register
23	VCDC	R/W	0x0	Video CODEC masked IRQ status register
22	ISP2	R/W	0x0	ISP Interrupt 2 masked IRQ status register
21	JPGE	R/W	0x0	JPEG Encoder masked IRQ status register
20	VIPET	R/W	0x0	Video Enhancer masked IRQ status register
19	LCD1	R/W	0x0	LCD controller 1 masked IRQ status register
18	LCD0	R/W	0x0	LCD controller 0 masked IRQ status register
17	CIF	R/W	0x0	Camer Interface masked IRQ status register
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 masked IRQ status register
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 masked IRQ status register
14	EI11	R/W	0x0	External interrupt 11 masked IRQ status register
13	EI10	R/W	0x0	External interrupt 10 masked IRQ status register
12	EI9	R/W	0x0	External interrupt 9 masked IRQ status register
11	EI8	R/W	0x0	External interrupt 8 masked IRQ status register
10	EI7	R/W	0x0	External interrupt 7 masked IRQ status register
9	EI6	R/W	0x0	External interrupt 6 masked IRQ status register
8	EI5	R/W	0x0	External interrupt 5 masked IRQ status register
7	EI4	R/W	0x0	External interrupt 4 masked IRQ status register
6	EI3	R/W	0x0	External interrupt 3 masked IRQ status register
5	EI2	R/W	0x0	External interrupt 2 masked IRQ status register
4	EI1	R/W	0x0	External interrupt 1 masked IRQ status register
3	EI0	R/W	0x0	External interrupt 0 masked IRQ status register
2	RTC	R/W	0x0	RTC masked IRQ status register
1	TC32	R/W	0x0	Timer32 masked IRQ status register
0	TC0	R/W	0x0	Timer 0 masked IRQ status register

Note: This represents the masked status for IRQ register.

Masked IRQ Status Register (MIRQ1) 0xB0600054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Note: This represents the masked status for IRQ register.

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 masked IRQ status register
29	SD2	R/W	0x0	SD/MMC 2 masked IRQ status register
28	APMU	R/W	0x0	ARM System Metrics masked IRQ status register.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) masked IRQ status register
26	ADMA0	R/W	0x0	Audio0(7.1ch) masked IRQ status register
25	AUDIO1	R/W	0x0	Audio1(streo) masked IRQ status register
24	ISP3	R/W	0x0	ISP Interrupt 3 masked IRQ status register
23	GMAC	R/W	0x0	GMAC masked IRQ status register
22	TSIF1	R/W	0x0	TS interface 1 masked IRQ status register
21	TSIF0	R/W	0x0	TS interface 0 masked IRQ status register
20	CIPHER	R/W	0x0	CIPHER masked IRQ status register
19	ADMA1	R/W	0x0	Audio1(stereo) DMA masked IRQ status register
18	GDMA1	R/W	0x0	GDMA1 masked IRQ status register (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 masked IRQ status register (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 masked IRQ status register (PBUS)
15	UART	R/W	0x0	UART masked IRQ status register
14	SD3	R/W	0x0	SD/MMC 3 masked IRQ status register
13	SD1	R/W	0x0	SD/MMC 1 masked IRQ status register
12	SD0	R/W	0x0	SD/MMC 0 masked IRQ status register
11	SMUI2C	R/W	0x0	SMU I2C masked IRQ status register
10	RMT	R/W	0x0	Remote Control masked IRQ status register
9	NFC	R/W	0x0	Nand flash controller masked IRQ status register
8	MS	R/W	0x0	Memory stick controller masked IRQ status register
7	CKC	R/W	0x0	CKC masked IRQ status register
6	I2C	R/W	0x0	I2C masked IRQ status register
4	GPSB	R/W	0x0	GPSB masked IRQ status register
2	HDMI	R/W	0x0	HDMI masked IRQ status register
0	EHI1	R/W	0x0	External Host masked IRQ status register

Masked FIQ Status Register (MFIQ0) 0xB0600058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host masked IRQ status register
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 masked IRQ status register
28	TSADC	R/W	0x0	TSADC masked IRQ status register
27	OM	R/W	0x0	Overlay mixer masked IRQ status register
26	3DMMU	R/W	0x0	3D MMU masked IRQ status register
25	3DGP	R/W	0x0	3D Geometry Processor masked IRQ status register
24	3DPP	R/W	0x0	3D Pixel Processor masked IRQ status register
23	VCDC	R/W	0x0	Video CODEC masked FIQ status register
22	ISP2	R/W	0x0	ISP Interrupt 2 masked FIQ status register
21	JPGE	R/W	0x0	JPEG Encoder masked FIQ status register
20	VIPET	R/W	0x0	Video Enhancer masked FIQ status register
19	LCD1	R/W	0x0	LCD controller 1 masked FIQ status register
18	LCD0	R/W	0x0	LCD controller 0 masked FIQ status register
17	CIF	R/W	0x0	Camer Interface masked FIQ status register
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 masked FIQ status register
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 masked FIQ status register
14	EI11	R/W	0x0	External interrupt 11 masked FIQ status register
13	EI10	R/W	0x0	External interrupt 10 masked FIQ status register
12	EI9	R/W	0x0	External interrupt 9 masked FIQ status register
11	EI8	R/W	0x0	External interrupt 8 masked FIQ status register
10	EI7	R/W	0x0	External interrupt 7 masked FIQ status register
9	EI6	R/W	0x0	External interrupt 6 masked FIQ status register
8	EI5	R/W	0x0	External interrupt 5 masked FIQ status register
7	EI4	R/W	0x0	External interrupt 4 masked FIQ status register
6	EI3	R/W	0x0	External interrupt 3 masked FIQ status register
5	EI2	R/W	0x0	External interrupt 2 masked FIQ status register
4	EI1	R/W	0x0	External interrupt 1 masked FIQ status register
3	EI0	R/W	0x0	External interrupt 0 masked FIQ status register
2	RTC	R/W	0x0	RTC masked FIQ status register
1	TC32	R/W	0x0	Timer32 masked FIQ status register
0	TC0	R/W	0x0	Timer 0 masked FIQ status register

Note: This represents the masked status for FIQ register.

Masked FIQ Status Register (MFIQ1) 0xB060005C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 masked FIQ status register
29	SD2	R/W	0x0	SD/MMC 2 masked FIQ status register
28	APMU	R/W	0x0	ARM System Metrics masked FIQ status register.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) masked FIQ status register
26	ADMA0	R/W	0x0	Audio0(7.1ch) masked FIQ status register
25	AUDIO1	R/W	0x0	Audio1(streo) masked FIQ status register
24	ISP3	R/W	0x0	ISP Interrupt 3 masked FIQ status register
23	GMAC	R/W	0x0	GMAC masked FIQ status register
22	TSIF1	R/W	0x0	TS interface 1 masked FIQ status register
21	TSIF0	R/W	0x0	TS interface 0 masked FIQ status register
20	CIPHER	R/W	0x0	CIPHER masked FIQ status register
19	ADMA1	R/W	0x0	Audio1(stereo) DMA masked FIQ status register
18	GDMA1	R/W	0x0	GDMA1 masked FIQ status register (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 masked FIQ status register (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 masked FIQ status register (PBUS)
15	UART	R/W	0x0	UART masked FIQ status register
14	SD3	R/W	0x0	SD/MMC 3 masked FIQ status register
13	SD1	R/W	0x0	SD/MMC 1 masked FIQ status register
12	SD0	R/W	0x0	SD/MMC 0 masked FIQ status register
11	SMUI2C	R/W	0x0	SMU I2C masked FIQ status register
10	RMT	R/W	0x0	Remote Control masked FIQ status register
9	NFC	R/W	0x0	Nand flash controller masked FIQ status register
8	MS	R/W	0x0	Memory stick controller masked FIQ status register
7	CKC	R/W	0x0	CKC masked FIQ status register
6	I2C	R/W	0x0	I2C masked FIQ status register
4	GPSB	R/W	0x0	GPSB masked FIQ status register
2	HDMI	R/W	0x0	HDMI masked FIQ status register
0	EHI1	R/W	0x0	External Host masked FIQ status register

Note: This represents the masked status for FIQ register.

Mode Register (MODE0) 0xB0600060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host mode register
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 mode register
28	TSADC	R/W	0x0	TSADC mode register
27	OM	R/W	0x0	Overlay mixer mode register
26	3DMMU	R/W	0x0	3D MMU mode register
25	3DGP	R/W	0x0	3D Geometry Processor mode register
24	3DPP	R/W	0x0	3D Pixel Processor mode register
23	VCDC	R/W	0x0	Video CODEC mode register
22	ISP2	R/W	0x0	ISP Interrupt 2 mode register
21	JPGE	R/W	0x0	JPEG Encoder mode register
20	VIPET	R/W	0x0	Video Enhancer mode register
19	LCD1	R/W	0x0	LCD controller 1 mode register
18	LCD0	R/W	0x0	LCD controller 0 mode register
17	CIF	R/W	0x0	Camer Interface mode register
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 mode register
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 mode register
14	EI11	R/W	0x0	External interrupt 11 mode register
13	EI10	R/W	0x0	External interrupt 10 mode register
12	EI9	R/W	0x0	External interrupt 9 mode register
11	EI8	R/W	0x0	External interrupt 8 mode register
10	EI7	R/W	0x0	External interrupt 7 mode register
9	EI6	R/W	0x0	External interrupt 6 mode register
8	EI5	R/W	0x0	External interrupt 5 mode register
7	EI4	R/W	0x0	External interrupt 4 mode register
6	EI3	R/W	0x0	External interrupt 3 mode register
5	EI2	R/W	0x0	External interrupt 2 mode register
4	EI1	R/W	0x0	External interrupt 1 mode register
3	EI0	R/W	0x0	External interrupt 0 mode register
2	RTC	R/W	0x0	RTC mode register
1	TC32	R/W	0x0	Timer32 mode register
0	TC0	R/W	0x0	Timer 0 mode register

Note: If the corresponding bit is ‘1’, the interrupt acts as level-triggered mode, otherwise edge-triggered mode.

Mode Register (MODE1) 0xB0600064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 mode register
29	SD2	R/W	0x0	SD/MMC 2 mode register
28	APMU	R/W	0x0	ARM System Metrics mode register.
27	AUDIO0	R/W	0x0	Audio0(7.1ch mode register
26	ADMA0	R/W	0x0	Audio0(7.1ch) mode register
25	AUDIO1	R/W	0x0	Audio1(streo) mode register
24	ISP3	R//W	0x0	ISP Interrupt 3 mode register
23	GMAC	R/W	0x0	GMAC mode register
22	TSIF1	R/W	0x0	TS interface 1 mode register
21	TSIF0	R/W	0x0	TS interface mode register
20	CIPHER	R/W	0x0	CIPHER mode register
19	ADMA1	R/W	0x0	Audio1(stereo) DMA mode register
18	GDMA1	R/W	0x0	GDMA1 mode register (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 mode register (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 mode register (PBUS)
15	UART	R/W	0x0	UART mode register
14	SD3	R/W	0x0	SD/MMC 3 mode register
13	SD1	R/W	0x0	SD/MMC 1 mode register
12	SD0	R/W	0x0	SD/MMC 0 mode register
11	SMUI2C	R/W	0x0	SMU I2C mode register
10	RMT	R/W	0x0	Remote Control mode register
9	NFC	R/W	0x0	Nand flash controller mode register
8	MS	R/W	0x0	Memory stick controller mode register
7	CKC	R/W	0x0	CKC mode register
6	I2C	R/W	0x0	I2C mode register
4	GPSB	R/W	0x0	GPSB mode register
2	HDMI	R/W	0x0	HDMI mode register
0	EHI1	R/W	0x0	External Host mode register

Note: If the corresponding bit is ‘1’, the interrupt acts as level-triggered mode, otherwise edge-triggered mode.

Synchronization Enable Register (SYNC0) 0xB0600068

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host synchronization enable
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 synchronization enable
28	TSADC	R/W	0x0	TSADC synchronization enable
27	OM	R/W	0x0	Overlay mixer synchronization enable
26	3DMMU	R/W	0x0	3D MMU synchronization enable
25	3DGP	R/W	0x0	3D Geometry Processor synchronization enable
24	3DPP	R/W	0x0	3D Pixel Processor synchronization enable
23	VCDC	R/W	0x0	Video CODEC synchronization enable
22	ISP2	R/W	0x0	ISP Interrupt 2 synchronization enable
21	JPGE	R/W	0x0	JPEG Encoder synchronization enable
20	VIPET	R/W	0x0	Video Enhancer synchronization enable
19	LCD1	R/W	0x0	LCD controller 1 synchronization enable
18	LCD0	R/W	0x0	LCD controller 0 synchronization enable
17	CIF	R/W	0x0	Camer Interface synchronization enable
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 synchronization enable
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 synchronization enable
14	EI11	R/W	0x0	External interrupt 11 synchronization enable
13	EI10	R/W	0x0	External interrupt 10 synchronization enable
12	EI9	R/W	0x0	External interrupt 9 synchronization enable
11	EI8	R/W	0x0	External interrupt 8 synchronization enable
10	EI7	R/W	0x0	External interrupt 7 synchronization enable
9	EI6	R/W	0x0	External interrupt 6 synchronization enable
8	EI5	R/W	0x0	External interrupt 5 synchronization enable
7	EI4	R/W	0x0	External interrupt 4 synchronization enable
6	EI3	R/W	0x0	External interrupt 3 synchronization enable
5	EI2	R/W	0x0	External interrupt 2 synchronization enable
4	EI1	R/W	0x0	External interrupt 1 synchronization enable
3	EI0	R/W	0x0	External interrupt 0 synchronization enable
2	RTC	R/W	0x0	RTC synchronization enable
1	TC32	R/W	0x0	Timer32 synchronization enable
0	TC0	R/W	0x0	Timer 0 synchronization enable

Note: If the corresponding bit is ‘1’, the controller synchronizes the corresponding interrupt source with SMU clock.

Synchronization Enable Register (SYNC1) 0xB060006C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 synchronization enable
29	SD2	R/W	0x0	SD/MMC 2 synchronization enable
28	APMU	R/W	0x0	ARM System Metrics synchronization enable.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) synchronization enable
26	ADMA0	R/W	0x0	Audio0(7.1ch) DMA synchronization enable
25	AUDIO1	R/W	0x0	Audio1(streo) synchronization enable
24	ISP3	R/W	0x0	ISP Interrupt 3 synchronization enable
23	GMAC	R/W	0x0	GMAC synchronization enable
22	TSIF1	R/W	0x0	TS interface 1 synchronization enable
21	TSIF0	R/W	0x0	TS interface 0 synchronization enable
20	CIPHER	R/W	0x0	CIPHER synchronization enable
19	ADMA1	R/W	0x0	Audio1(stereo) DMA synchronization enable
18	GDMA1	R/W	0x0	GDMA1 synchronization enable (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 synchronization enable (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 synchronization enable (PBUS)
15	UART	R/W	0x0	UART synchronization enable
14	SD3	R/W	0x0	SD/MMC 3 synchronization enable
13	SD1	R/W	0x0	SD/MMC 1 synchronization enable
12	SD0	R/W	0x0	SD/MMC 0 synchronization enable
11	SMUI2C	R/W	0x0	SMU I2C synchronization enable
10	RMT	R/W	0x0	Remote Control synchronization enable
9	NFC	R/W	0x0	Nand flash controller synchronization enable
8	MS	R/W	0x0	Memory stick controller synchronization enable
7	CKC	R/W	0x0	CKC synchronization enable
6	I2C	R/W	0x0	I2C synchronization enable
4	GPSB	R/W	0x0	GPSB synchronization enable
2	HDMI	R/W	0x0	HDMI synchronization enable
0	EHI1	R/W	0x0	External Host synchronization enable

Note: If the corresponding bit is ‘1’, the controller synchronizes the corresponding interrupt source with SMU clock.

Wakeup Enable Register (WKEN0) 0xB0600070

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host wake up enable
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 wake up enable
28	TSADC	R/W	0x0	TSADC wake up enable
27	OM	R/W	0x0	Overlay mixer wake up enable
26	3DMMU	R/W	0x0	3D MMU wake up enable
25	3DGP	R/W	0x0	3D Geometry Processor wake up enable
24	3DPP	R/W	0x0	3D Pixel Processor wake up enable
23	VCDC	R/W	0x0	Video CODEC wake up enable
22	ISP2	R/W	0x0	ISP Interrupt 2 wake up enable
21	JPGE	R/W	0x0	JPEG Encoder wake up enable
20	VIPET	R/W	0x0	Video Enhancer wake up enable
19	LCD1	R/W	0x0	LCD controller 1 wake up enable
18	LCD0	R/W	0x0	LCD controller 0 wake up enable
17	CIF	R/W	0x0	Camer Interface wake up enable
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 wake up enable
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 wake up enable
14	EI11	R/W	0x0	External interrupt 11 wake up enable
13	EI10	R/W	0x0	External interrupt 10 wake up enable
12	EI9	R/W	0x0	External interrupt 9 wake up enable
11	EI8	R/W	0x0	External interrupt 8 wake up enable
10	EI7	R/W	0x0	External interrupt 7 wake up enable
9	EI6	R/W	0x0	External interrupt 6 wake up enable
8	EI5	R/W	0x0	External interrupt 5 wake up enable
7	EI4	R/W	0x0	External interrupt 4 wake up enable
6	EI3	R/W	0x0	External interrupt 3 wake up enable
5	EI2	R/W	0x0	External interrupt 2 wake up enable
4	EI1	R/W	0x0	External interrupt 1 wake up enable
3	EI0	R/W	0x0	External interrupt 0 wake up enable
2	RTC	R/W	0x0	RTC wake up enable
1	TC32	R/W	0x0	Timer32 wake up enable
0	TC0	R/W	0x0	Timer 0 wake up enable

Note: If the corresponding bit is ‘0’, the CPU clock or bus clock can be waked up by the corresponding interrupt source.

Wakeup Enable Register (WKEN1) 0xB0600074

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 wake up enable
29	SD2	R/W	0x0	SD/MMC 2 wake up enable
28	APMU	R/W	0x0	ARM System Metrics wake up enable.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) wake up enable
26	ADMA0	R/W	0x0	Audio0(7.1ch) DMA wake up enable
25	AUDIO1	R/W	0x0	Audio1(streo) wake up enable
24	ISP3	R/W	0x0	ISP Interrupt 3 wake up enable
23	GMAC	R/W	0x0	GMAC wake up enable
22	TSIF1	R/W	0x0	TS interface 1 wake up enable
21	TSIF0	R/W	0x0	TS interface 0 wake up enable
20	CIPHER	R/W	0x0	CIPHER wake up enable
19	ADMA1	R/W	0x0	Audio1(stereo) DMA wake up enable
18	GDMA1	R/W	0x0	GDMA1 wake up enable (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 wake up enable (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 wake up enable (PBUS)
15	UART	R/W	0x0	UART wake up enable
14	SD3	R/W	0x0	SD/MMC 3 wake up enable
13	SD1	R/W	0x0	SD/MMC 1 wake up enable
12	SD0	R/W	0x0	SD/MMC 0 wake up enable
11	SMUI2C	R/W	0x0	SMU I2C wake up enable
10	RMT	R/W	0x0	Remote Control wake up enable
9	NFC	R/W	0x0	Nand flash controller wake up enable
8	MS	R/W	0x0	Memory stick controller wake up enable
7	CKC	R/W	0x0	CKC wake up enable
6	I2C	R/W	0x0	I2C wake up enable
4	GPSB	R/W	0x0	GPSB wake up enable
2	HDMI	R/W	0x0	HDMI wake up enable
0	EHI1	R/W	0x0	External Host wake up enable

Note: If the corresponding bit is ‘0’, the CPU clock or bus clock can be waked up by the corresponding interrupt source.

Edge Trigger Mode Register (MODEA0) 0xB0600078

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC		JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host edge trigger mode
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 edge trigger mode
28	TSADC	R/W	0x0	TSADC edge trigger mode
27	OM	R/W	0x0	Overlay mixer edge trigger mode
26	3DMMU	R/W	0x0	3D MMU edge trigger mode
25	3DGP	R/W	0x0	3D Geometry Processor edge trigger mode
24	3DPP	R/W	0x0	3D Pixel Processor edge trigger mode
23	VCDC	R/W	0x0	Video CODEC edge trigger mode
22	ISP2	R/W	0x0	ISP Interrupt 2 edge trigger mode
21	JPGE	R/W	0x0	JPEG Encoder edge trigger mode
20	VIPET	R/W	0x0	Video Enhancer edge trigger mode
19	LCD1	R/W	0x0	LCD controller 1 edge trigger mode
18	LCD0	R/W	0x0	LCD controller 0 edge trigger mode
17	CIF	R/W	0x0	Camer Interface edge trigger mode
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 edge trigger mode
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 edge trigger mode
14	EI11	R/W	0x0	External interrupt 11 edge trigger mode
13	EI10	R/W	0x0	External interrupt 10 edge trigger mode
12	EI9	R/W	0x0	External interrupt 9 edge trigger mode
11	EI8	R/W	0x0	External interrupt 8 edge trigger mode
10	EI7	R/W	0x0	External interrupt 7 edge trigger mode
9	EI6	R/W	0x0	External interrupt 6 edge trigger mode
8	EI5	R/W	0x0	External interrupt 5 edge trigger mode
7	EI4	R/W	0x0	External interrupt 4 edge trigger mode
6	EI3	R/W	0x0	External interrupt 3 edge trigger mode
5	EI2	R/W	0x0	External interrupt 2 edge trigger mode
4	EI1	R/W	0x0	External interrupt 1 edge trigger mode
3	EI0	R/W	0x0	External interrupt 0 edge trigger mode
2	RTC	R/W	0x0	RTC edge trigger mode
1	TC32	R/W	0x0	Timer32 edge trigger mode
0	TC0	R/W	0x0	Timer 0 edge trigger mode

Note: If the corresponding bit is ‘0’ in case of the edge-triggered mode, the interrupt propagated to nFIQ or nIRQ in single edge, otherwise in both edge.

Edge Trigger Mode Register (MODEA1) 0xB060007C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 edge trigger mode
29	SD2	R/W	0x0	SD/MMC 2 edge trigger mode
28	APMU	R/W	0x0	ARM System Metrics edge trigger mode.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) edge trigger mode
26	ADMA0	R/W	0x0	Audio0(7.1ch) DMA edge trigger mode
25	AUDIO1	R/W	0x0	Audio1(streo) edge trigger mode
24	ISP3	R/W	0x0	ISP Interrupt 3 edge trigger mode
23	GMAC	R/W	0x0	GMAC edge trigger mode
22	TSIF1	R/W	0x0	TS interface 1 edge trigger mode
21	TSIF0	R/W	0x0	TS interface 0 edge trigger mode
20	CIPHER	R/W	0x0	CIPHER edge trigger mode
19	ADMA1	R/W	0x0	Audio1(stereo) DMA edge trigger mode
18	GDMA1	R/W	0x0	GDMA1 edge trigger mode (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 edge trigger mode (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 edge trigger mode (PBUS)
15	UART	R/W	0x0	UART edge trigger mode
14	SD3	R/W	0x0	SD/MMC 3 edge trigger mode
13	SD1	R/W	0x0	SD/MMC 1 edge trigger mode
12	SD0	R/W	0x0	SD/MMC 0 edge trigger mode
11	SMUI2C	R/W	0x0	SMU I2C edge trigger mode
10	RMT	R/W	0x0	Remote Control edge trigger mode
9	NFC	R/W	0x0	Nand flash controller edge trigger mode
8	MS	R/W	0x0	Memory stick controller edge trigger mode
7	CKC	R/W	0x0	CKC edge trigger mode
6	I2C	R/W	0x0	I2C edge trigger mode
4	GPSB	R/W	0x0	GPSB edge trigger mode
2	HDMI	R/W	0x0	HDMI edge trigger mode
0	EHI1	R/W	0x0	External Host edge trigger mode

IRQ Output Mask Register (INTMSK0) 0xB0600100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EHI0		GDMA0	TSADC	OM	3DMMU	3DGP	3DPP	VCDC	ISP2	JPGE	VIPET	LCD1	LCD0	CIF	SC1
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SC0	EI11	EI10	EI9	EI8	EI7	EI6	EI5	EI4	EI3	EI2	EI1	EI0	RTC	TC32	TC0

Field	Name	RW	Reset	Description
31	EHI0	R/W	0x0	External Host output mask
30	-	-	-	-
29	GDMA0	R/W	0x0	General DMA 0 output mask
28	TSADC	R/W	0x0	TSADC output mask
27	OM	R/W	0x0	Overlay mixer output mask
26	3DMMU	R/W	0x0	3D MMU output mask
25	3DGP	R/W	0x0	3D Geometry Processor output mask
24	3DPP	R/W	0x0	3D Pixel Processor output mask
23	VCDC	R/W	0x0	Video CODEC output mask
22	ISP2	R/W	0x0	ISP Interrupt 2 output mask
21	JPGE	R/W	0x0	JPEG Encoder output mask
20	VIPET	R/W	0x0	Video Enhancer output mask
19	LCD1	R/W	0x0	LCD controller 1 output mask
18	LCD0	R/W	0x0	LCD controller 0 output mask
17	CIF	R/W	0x0	Camer Interface output mask
16	SC1	R/W	0x0	Mem-to-Mem scaler 1 output mask
15	SC0	R/W	0x0	Mem-to-Mem scaler 0 output mask
14	EI11	R/W	0x0	External interrupt 11 output mask
13	EI10	R/W	0x0	External interrupt 10 output mask
12	EI9	R/W	0x0	External interrupt 9 output mask
11	EI8	R/W	0x0	External interrupt 8 output mask
10	EI7	R/W	0x0	External interrupt 7 output mask
9	EI6	R/W	0x0	External interrupt 6 output mask
8	EI5	R/W	0x0	External interrupt 5 output mask
7	EI4	R/W	0x0	External interrupt 4 output mask
6	EI3	R/W	0x0	External interrupt 3 output mask
5	EI2	R/W	0x0	External interrupt 2 output mask
4	EI1	R/W	0x0	External interrupt 1 output mask
3	EI0	R/W	0x0	External interrupt output mask
2	RTC	R/W	0x0	RTC output mask
1	TC32	R/W	0x0	Timer32 output mask
0	TC0	R/W	0x0	Timer 0 output mask

Note: If the corresponding bit is ‘1’, interrupt are passed to IRQ or FIQ.

IRQ Output Mask Register (INTMSK1) 0xB0600104

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ISP1	-	SD2	APMU	AUDIO0	ADMA0	AUDIO1	ISP3	GMAC	TSIF1	TSIF0	CIPHER	ADMA1	GDMA3	UOTG1	UOTG0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UART	SD3	SD1	SD0	SMUI2C	RMT	NFC	MS	CKC	I2C	-	GPSB	-	HDMI	-	EHI1

Field	Name	RW	Reset	Description
31	ISP1	R/W	0x0	ISP Interrupt 1 output mask
29	SD2	R/W	0x0	SD/MMC 2 output mask
28	APMU	R/W	0x0	ARM System Metrics output mask.
27	AUDIO0	R/W	0x0	Audio0(7.1ch) output mask
26	ADMA0	R/W	0x0	Audio0(7.1ch) DMA output mask
25	AUDIO1	R/W	0x0	Audio1(streo) output mask
24	ISP3	R/W	0x0	ISP Interrupt 3 output mask
23	GMAC	R/W	0x0	GMAC output mask
22	TSIF1	R/W	0x0	TS interface 1 output mask
21	TSIF0	R/W	0x0	TS interface 0 output mask
20	CIPHER	R/W	0x0	CIPHER output mask
19	ADMA1	R/W	0x0	Audio1(stereo) DMA output mask
18	GDMA1	R/W	0x0	GDMA1 output mask (HSBUS)
17	UOTG1	R/W	0x0	USB 2.0 OTG1 output mask (HSBUS)
16	UOTG0	R/W	0x0	USB 2.0 OTG0 output mask (PBUS)
15	UART	R/W	0x0	UART output mask
14	SD3	R/W	0x0	SD/MMC 3 output mask
13	SD1	R/W	0x0	SD/MMC 1 output mask
12	SD0	R/W	0x0	SD/MMC 0 output mask
11	SMUI2C	R/W	0x0	SMU I2C output mask
10	RMT	R/W	0x0	Remote Control output mask
9	NFC	R/W	0x0	Nand flash controller output mask
8	MS	R/W	0x0	Memory stick controller output mask
7	CKC	R/W	0x0	CKC output mask
6	I2C	R/W	0x0	I2C output mask
4	GPSB	R/W	0x0	GPSB output mask
2	HDMI	R/W	0x0	HDMI output mask
0	EHI1	R/W	0x0	External Host output mask

All Register (ALLMSK) 0xB0600108

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-														FIQ	IRQ

Field	Name	RW	Reset	Description
31-2	-	-	-	Undefined
1	FIQ	R/W	0x1	FIQ mask register When it is 0, FIQ interrupt is masked.
0	IRQ	R/W	0x1	IRQ mask register When it is 0, IRQ interrupt is masked.

20.3.2 Vectored Interrupt Controller

The vectored interrupt controller can make it possible to service the corresponding interrupt with any other unnecessary comparison operation. The vectored interrupt controller has the 4 registers for this purpose, VAIRQ, VAFIQ, VNIRQ, and VNIRQ.

The VAIRQ and VAFIQ represent the vectored address offset for current activated interrupt in word-based address. For example, if the 2nd priority interrupt has been activated to IRQ, the VAIRQ represents the '4' in VA field below with '0' INV field. The VAFIQ is same as VAIRQ.

The VNIRQ and VNFIQ represent the vectored number offset for current interrupt. For example, if the 2nd priority interrupt has been activated to IRQ, the VNIRQ represents the '1' in VA field below. You can use these features for fast handler service.

IRQ Vector Register (VAIRQ) **0xB0600200**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INV	0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VA															

Field	Name	RW	Reset	Description
31	INV	R	0x0	0 : Valid for Current Interrupt Source 1 : Invalid for Current Interrupt Source.
30-9	-	-	-	-
8-0	VA	R	0x0	N : Interrupt Vector Address Offset in Word-Address. This is one of '0', '4', '8',.

FIQ Vector Register (VAFIQ) **0xB0600204**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INV	0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VA															

Field	Name	RW	Reset	Description
31	INV	R	0x0	0 : Valid for Current Interrupt Source 1 : Invalid for Current Interrupt Source.
30-9	-	-	-	-
8-0	VA	R	0x0	N : Interrupt Vector Address Offset in Word-Address. This is one of '0', '4', '8',.

IRQ Number Register (VNIRQ) **0xB0600208**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INV	0														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VN															

Field	Name	RW	Reset	Description
31	INV	R	0x0	0 : Valid for Current Interrupt Source 1 : Invalid for Current Interrupt Source.
30-7	-	-	-	-
6-0	VA	R	0x0	N : Interrupt Vector Address Offset in Word-Address. This is one of '0', '1', '2',.

FIQ Number Register (VNFIQ) 0xB060020C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INV	0														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VN															

Field	Name	RW	Reset	Description
31	INV	R	0x0	0 : Valid for Current Interrupt Source 1 : Invalid for Current Interrupt Source.
30-7	-	-	-	-
6:0	VA	R	0x0	N : Interrupt Vector Address Offset in Word-Address. This is one of '0', '1', '2',.

Vector Control Register (VCTRL) 0xB0600210

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCL	FPOL	FFLG	IFLG	FHD	IHD	0									
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0															

Field	Name	RW	Reset	Description
31	RCL	R/W	0x0	Clear Interrupt Status 0 : The write operation is needed for clearing interrupt status. 1 :. The reading the VN or VA register clears the interrupt status
30	FPOL	R/W	0x0	Valid Flag Polarity 0 : The INV field means valid for '0', invalid for '1'. 1 : The INV field means valid for '1', invalid for '0'
29	FFLG	R/W	0x0	FIQ Valid Flag Enable 0 : Invalid flag enable for FIQ vector registers.. 1 : Valid flag enable for FIQ vector registers
28	IFLG	R/W	0x0	IRQ Valid Flag Enable 0 : Invalid flag enable for IRQ vector registers.. 1 : Valid flag enable for IRQ vector registers
27	FHD	R/W	0x0	FIQ Hold Enable 0 : Disable the holding vector for FIQ until cleared... 1 : Enable the holding vector for FIQ until cleared.
26	IHD	R/W	0x0	IRQ Hold Enable 0 : Disable the holding vector for IRQ until cleared... 1 : Enable the holding vector for IRQ until cleared.
25-0	-	-	-	-

Priority Configuration Register (PRIOn) 0xB0600220+4*n,n=0~15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI[4*n+3]								PRI[4*n+2]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRI[4*n+1]								PRI[4*n+0]							

After resetting the system, the priority of each interrupt controller is determined with default value. The default priority of interrupt source 0 is 0 – highest priority and the default priority of interrupt source 63 is 63 – lowest priority. If you want to change the priority of each interrupt sources, you have to write the priority value to the corresponding field.

Be advised that the controller can do the unpredicted operation in case the priorities of different interrupt sources are set with the same value.

Field	Name	RW	Reset	Description
31-24	PRI[4*n+3]	R/W	0x0	Priority Configuration Register Priority for Interrupt [4*n], n=0~15
23-16	PRI[4*n+2]	R/W	0x0	Priority Configuration Register Priority for Interrupt [4*n], n=0~15
15-8	PRI[4*n+2]	R/W	0x0	Priority Configuration Register Priority for Interrupt [4*n], n=0~15
7-0	PRI[4*n+2]	R/W	0x0	Priority Configuration Register

				Priority for Interrupt [4*n], n=0~15

20.4 Operation & Timing Diagram

20.4.1 How to Configure the Interrupt Source

The following figure shows the active condition and corresponding IRQI waveform. The active condition can be configured by the POL0 and POL1 registers. If the IRQI is low during the active period, the corresponding bit-field of the POL0 and POL1 registers should be '1'.

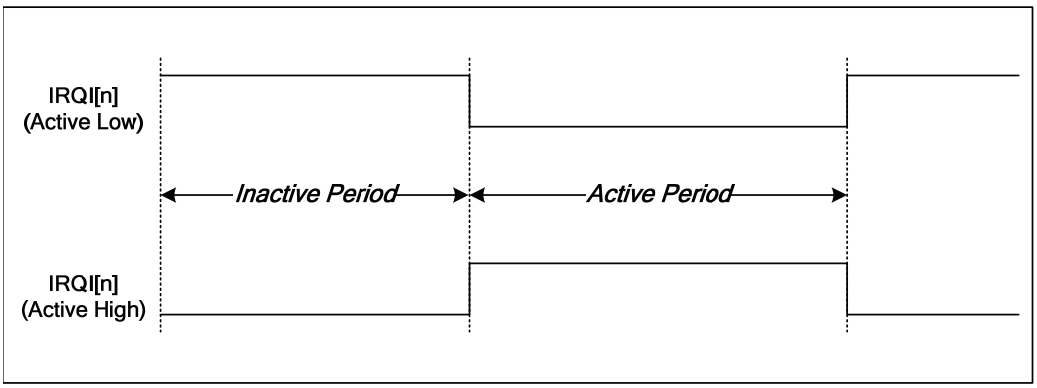


Figure 20.3 Active High vs. Active Low

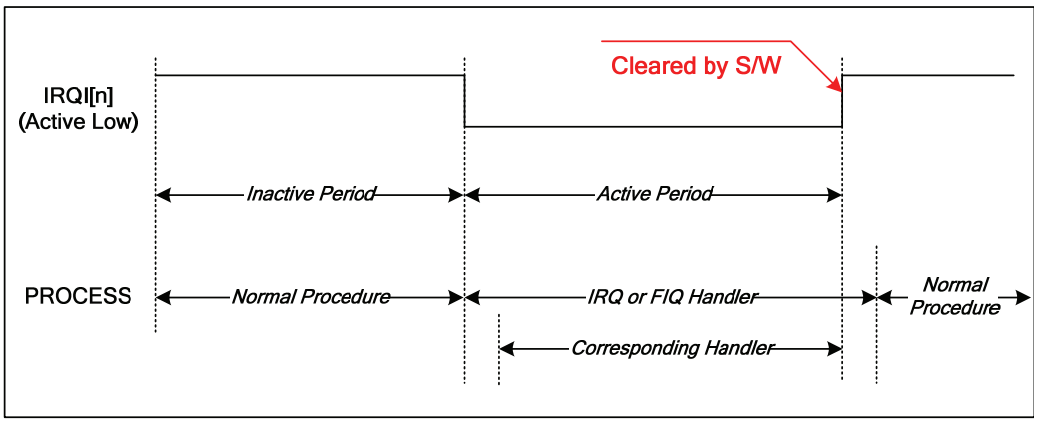


Figure 20.4 Level Triggered vs. Edge Triggered

The above figure shows the IRQI timing diagram and the corresponding S/W flow.
At the falling edge of the IRQI, the nIRQ or nFIQ will be "LOW" to process the interrupt handler. If the IRQI can be cleared during the IRQ or FIQ handler, the level-triggered type is preferred. But, if the clearing the interrupt can not be finished during the IRQ or FIQ handler, the corresponding IRQI should be configured by edge-triggered type because the IRQI can make unexpected IRQ or FIQ handler called continuously until the IRQI changed to inactive state.

But, in the edge-triggered type, the IRQI can be missed in the special case. The following shows the possibility to miss the edge-triggered IRQI signal.

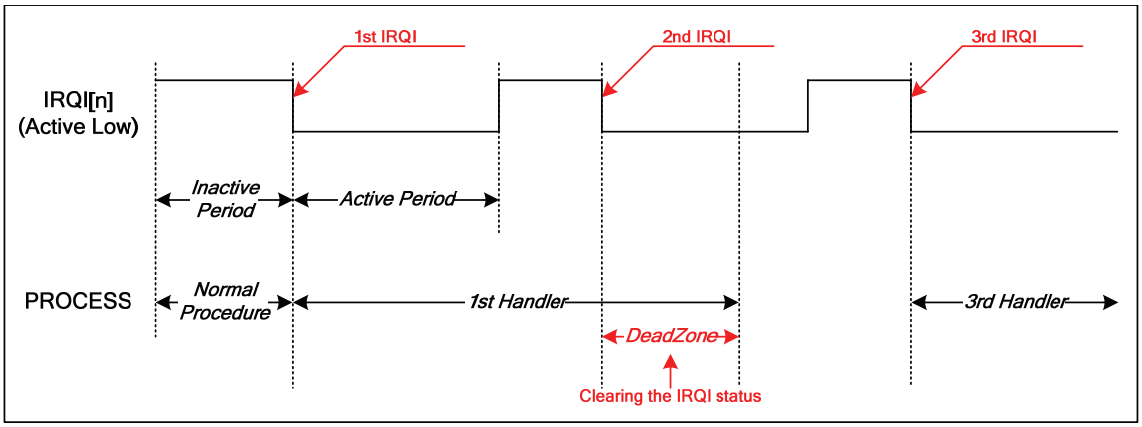


Figure 20.5 A Case of Missing the Edge-Triggered IRQI

If the edge-triggered interrupt occurred, the interrupt controller can not receive the next interrupt until the corresponding interrupt status being cleared. In the above figure, if the "1st IRQI' would be cleared in the "DeadZone", the 2nd IRQI interrupt can not be received by the interrupt controller. After than, the controller can accept the "3rd IRQI" event in the next falling edge. For the above case, the edge-triggered interrupt should be taken care of.

The following paragraphs and figure are about the difference between sync and async mode.

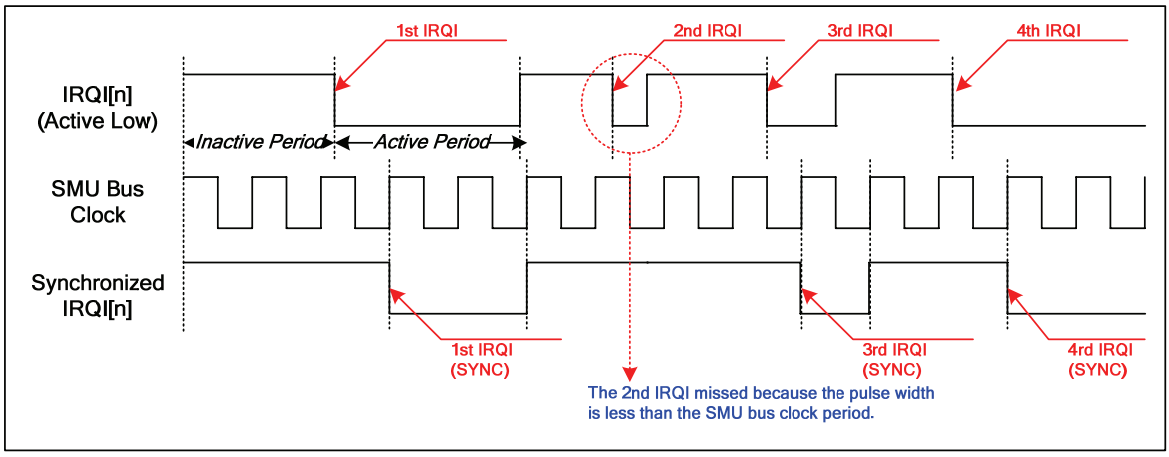


Figure 20.6 The timing Relations between Original IRQI and Synchronized IRQI

As shown in the above figure, if the active pulse width of the IRQI is greater than the SMU bus clock period, the interrupt controller can accept the corresponding IRQI with timing of the synchronized IRQI. But, if the pulse width is shorter than the clock period such as "2nd IRQI", the interrupt controller can't recognize the corresponding IRQI because of the corresponding interrupt can be registered by the SMU bus clock as shown in the "Synchronized IRQI[n]" in the above figure.

Finally, the INTMSK0, INTMSK1 and ALLMSK register can be used for masking the interrupt temporarily. The IEN0 and IEN1 register can also be used to mask. But, major difference is the possibility of accepting the interrupt source during the masked period. The INTMSK0, INTMSK1 and ALLMSK can recognize the interrupt during the masked period but IEN0 and IEN1 can not accept the interrupt during the masked period.

20.4.2 How to Enable Interrupt for IRQI

Before enabling the interrupt for the corresponding IRQI, the followings should be determined.

1. Is the IRQI level-triggered or edge-triggered ? (MODE0 and MODE1)
2. If edge-triggered case, which edge(s) is(are) used ? (MODEA0 and MODEA1)
3. Which type of the corresponding interrupt be used ? IRQ or FIQ (SEL0 and SEL1)
4. Whether the synchronization function is required or not? The recommended is to use this function. (SYNC0 and SYNC1)

After the above configurations decided, you have only to enable the interrupt with IEN0 and IEN1.

20.4.3 Recommended IRQI Configurations

The following table is the recommended configurations for the corresponding IRQI. ***But, the following configurations can be changed according to the system or software platform.*** The SYNC field in the following table is recommended to '1', but if the SMU bus clock is disabled, the interrupt controller can not receive the interrupt.

Table 20.8 Recommended IRQI Configurations

Field	Name	MODE	MODEA	POL	SYNC	Related Hardware(s)
0	TC0	1	-	0	1	TIMER Interrupt
1	TC32	1	-	0	1	TIMER32 Interrupt
2	RTC	1	-	0	1	RTC(Real-Time Clock)
3~14	EI0~EI11	-	-	-	1	External Interrupts * Refer to the next table.
15	SC0	1		1	1	Mem-to-Mem Scaler Controller 0
16	SC1	1		1	1	Mem-to-Mem Scaler Controller 1
17	CIF	1	-	0	1	Camera Interface
18	LCD0	1		0	1	LCD Controller 0
19	LCD1	1		0	1	LCD Controller 1
20	VIPET	1		0	1	VIQE (Video Enhancer)
21	JPGE	1	-	0	1	JPEG Encoder
22	ISP2	1	-	0	1	ISP
23	VCDC	1	-	0	1	Video Codec
24	3DPP	1	-	0	1	3D Pixel Processor Interrupt
25	3DGP	1	-	0	1	3D Geometry Processor Interrupt
26	3DMMU	1	-	0	1	3D MMU Interrupt
27	OM	1	-	0	1	Overlay Mixer (2D Graphic Engine)
28	TSADC	1	-	0	1	Touch Screen Controller
29	DMA0	1	-	0	1	DMA for Peripheral BUS
30	-					
31	EHI0	1	-	0	1	External Host Interface Controller 0
32	EHI1	1	-	0	1	External Host Interface Controller 1
33	-	0	0	0	1	-
34	HDMI	1	-	0	1	HDMI Link Controller
35	-	1	-	0	1	-
36	GPSB	1	-	0	1	GPSB Controller
37	-	1		0		-
38	I2C	1	-	0	1	I2C Controller
39	CKC	1	-	0	1	CKC
40	MS	1	-	1	1	Memory Stick Controller
41	NFC	1	-	0	1	NAND Flash Controller
42	RMT	1	-	0	1	Remocon Controller
43	SMUI2C	1	-	0	1	SMU I2C
44	SD0	1	-	0	1	SDMMC Controller 0
45	SD1	1	-	0	1	SDMMC Controller 1
46	SD3	1	-	0	1	SDMMC Controller 3
47	UART	1	-	0	1	UART Controller
48	UOTG0	1	-	0	1	USB 2.0 OTG for Peripheral BUS
49	UOTG1	1	-	0	1	USB 2.0 OTG for HSIO BUS
50	DMA1	1	-	0	1	DMA for HSIO BUS
51	ADMA1	0	0	0	1	Audio1(7.1ch) DMA
52	CIPHER	1	-	0	1	CIPHER
53	TSIF0	0	0	0	1	TS Interface Controller 0
54	TSIF1	0	0	0	1	TS Interface Controller 1

55	GMAC	0	0	0	1	GMAC
56	ISP3	1	-	0	1	ISP JPEG ERR or JPEG_STAT
57	AUDIO1	0	0	0	1	Audio1(stereo)
58	ADMA0	0	0	0	1	Audio0(7.1ch) DMA
58	ADMA1	0	0	0	1	Audio1(stereo) DMA Controller
59	AUDIO0	0	0	0	1	Audio0(7.1ch)
60	APMU	1	-	1	1	ARM System Metrics interrupt enable
61	SD2	1	-	1	1	SDMMC Controller 2
62	-	1	-	0	1	-
63	ISP1	1		0	1	ISP MI

The following sources can be mapped to external interrupt source. And the recommended configurations are shown in the following table.

Table 20.9 Recommended External Interrupt Configurations

Field	Name	MODE	MODEA	POL	SYNC	Related Hardware(s)
1	GPIOs	-	-	-	1	The configurations can be changed to the external components.
2	TSWKU	1	-	0	1	Wake-up interrupt from the Touch Screen Controller.
3	TSSTOP	0	0	0	0	Stop interrupt from Touch Screen Controller
4	TSUPDN	0	1	1	0	Up/Down interrupt from Touch Screen Controller
5	PMWKUP	0	0	0	0	RTC Wakeup Interrupt * The polarity can be changed to active low.
6	USB0_VBON	0	0	0	0	
7	USB0_VBOFF	0	0	0	0	
8	USB1_VBON	0	0	0	0	
9	USB1_VBOFF	0	0	0	0	

For the GPIO case, the values of MODE, MODEA, POL should be determined appropriatly according the external component or any type of interrupt source,

20.4.4 How to Use Vectored Interrupts

In the real-time system, the interrupt latency and interrupt processing time is very important. The vectored interrupt controller can make them faster than that of non-vectored system. To use of the vectored interrupt controller, the interrupt priorities according to the IRQI should be pre-defined. For the safe system management, the priorities is preferred not to change in run-time operation.

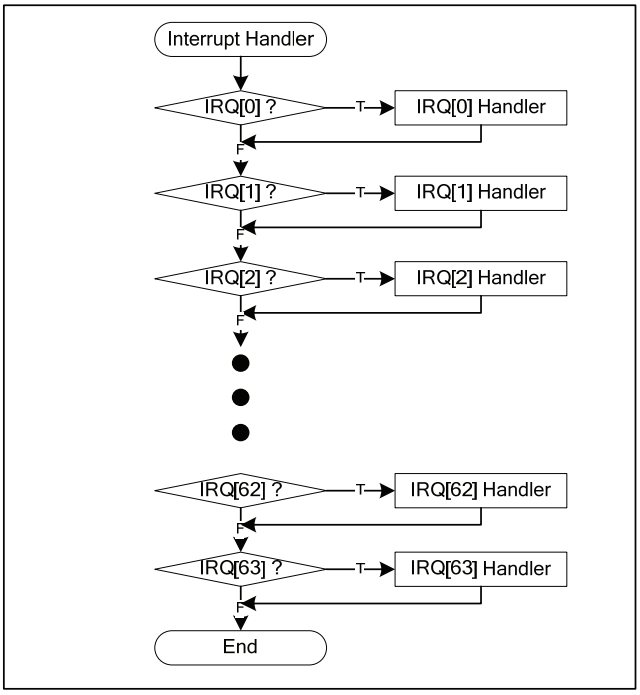


Figure 20.7 Structure of the Non-Vectored Interrupt Handler

The above figure shows the software structure of the non-vectored interrupt. In the above figure, the IRQ[0] has the highest priority and IRQ[63] has the lowest priority which can be determined and changed by software. The software searches from IRQ[0] to IRQ[63] consequently to find which interrupt occurred. The IRQ[0], which has highest priority, can be processed lowest latency. But for the IRQ[63], lowest priority, even if there is no interrupt in the IRQ[0] ~ IRQ[62], the large latency for IRQ[63] is required because of searching routine.

The following figure shows that the structure of the vectored interrupt.

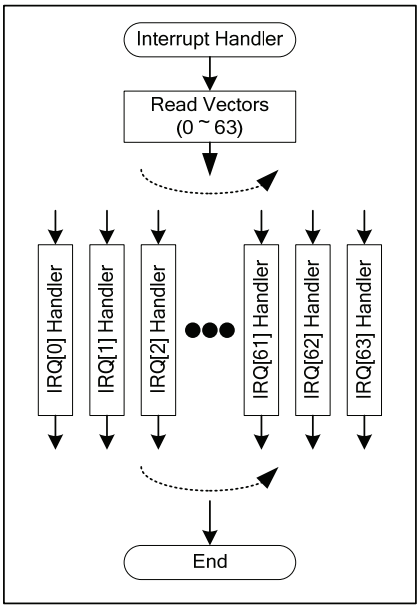


Figure 20.8 Structure of the Vectored Interrupt Handler

All the interrupt handler has the same latency as you can see in the above figure because the vector indicating which interrupt occurred can be read. The software can branch to the corresponding handler directly with function pointer array.

```
typedef void Handler(void);                    // Handler Function Prototype

unsigned int gnIntHandler[64];                 // Vector Table
                                               // should be initialized at the system boot-up

__irq IRQHandler (void)
{
  Handler *fUserHandler;
  unsigned int nVectorID;

  while (1) {                                  // Looping Until All the interrupts processed
    nVectorID = VNIRQ;                         // Reading Vector
    if ( nVectorID & 0x80000000) break;        // Checking Invalid Interrupt or Not
    fUserHandler = (Handler *) (gnIntHandler[nVectorID]);
    fUserHandler ();                           // Branching to the handler
    VNIRQ = nVectorID;                         // Clearing the corresponding interrupt
  }
}
```

Figure 20.9 Example Code of the Vectored Interrupt Handler

The above figure shows the example code for the IRQ handler. In the FIQ case, the main structure is same as IRQ. The bold, italic and underlined indicates the register value.
The “gnIntHandler” arrays should have the base address for all interrupts and the handler function should have same prototype.

21 IOBUS Configuration Registers

The IOBUS Configuration block has several registers named as USBOTG, USB11H, IOBAPB, STORAGE, HCLKEN0/1, HCLKMEN, HRSTEN0/1, USBOTG0/1/2/3 and IO_A2X.

Table 21.1 IOBUS Configuration Register Map (Base Address = 0xB0080000)

Name	Address	Description
HCLKMASK0	0x00	Module Clock Mask Register 0
SWRESET0	0x04	Module Software Reset Register 0
HCLKMASK1	0x08	Module Clock Mask Register 1
SWRESET1	0x0C	Module Software Reset Register 1
WAITCTRL	0x18	IOBUS APB wait counter Register
NFCACK_SEL	0x1C	
USBOTG	0x20	Refer to **USB OTG Configuration Register (OTGCR)** in "**13.2 Register Description for USB 2.0 OTG** Controller"**USB OTG Configuration Register (OTGCR)USB OTG Configuration Register (OTGCR)USB OTG Configuration Register (OTGCR)USB OTG Configuration Register (OTGCR)USB OTG Configuration Register (OTGCR)**.
USBOTG0	0x24	Refer to **USB PHY Configuration Register0 (UPCR0)** in "**13.2 Register Description for USB 2.0 OTG** Controller"
USBOTG1	0x28	Refer to **USB PHY Configuration Register1 (UPCR1)** in "**13.2 Register Description for USB 2.0 OTG** Controller"
USBOTG2	0x2C	Refer to **USB PHY Configuration Register3 (UPCR3)** in "**13.2 Register Description for USB 2.0 OTG** Controller"
USBOTG3	0x30	Refer to **USB PHY Configuration Register3 (UPCR3)** in "**13.2 Register Description for USB 2.0 OTG** Controller
IO_A2X	0x40	IOBUS AHB2AXI Control Register

IOBUS HCLK MASK Register 0(HCLKMASK0) 0xB0080000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HCLKMASK0[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HCLKMASK0[15:0]															

Field	Name	RW	Reset	Description
31-0	HCLKMASK0	R/W	0x0	Enable signals of HCLK clock gating logic AHB HCLKs, supplied to each peripheral block, are controlled by clock enable signal. 0 : enable clock 1: disable clock The bit position indicates each sub-block which is controlled by. BIT 0:Prefetch/Write Buffer BIT 1: EHI0 BIT 2: EHI1 BIT 3: USB OTG BIT 4: NFC controller BIT 5: SD/MMC Channel Control Register BIT 6: SD/MMC0 Controller BIT 7: SD/MMC1 Controller BIT 8: SD/MMC2 Controller BIT 9: GDMA0 Controller BIT 10: GDMA1 Controller BIT 11: GDMA2 Controller BIT 12: PWM Controller 1 BIT 13: Overlay mixer Controller BIT 14: SD/MMC3 Controller BIT 15: Remote Controller BIT 16: TSIF0 Controller 5 BIT 17: TSIF1 Controller 0 BIT 18: ADMA0 Controller 1 BIT 19: DAI0 Controller 2 BIT 20: ADMA1 Controller 3 BIT 21: DAI1 Controller 4 BIT 22: UART Controller 0 BIT 23: UART Controller 1 BIT 24: UART Controller 2 BIT 25: UART Controller 3 BIT 26: UART Controller 4 BIT 27: UART Controller 5 BIT 28 I2C Controller0 BIT 29: I2C Controller1 BIT 30: I2C Controller2 BIT 31: GPSB0 Controller BIT 32: GPSB1 Controller BIT 33: GPSB2 Controller BIT 34: GPSB3 Controller BIT 35 GPSB4 Controller BIT 36: GPSB5 Controller BIT 37: TSADC Controller BIT 38: GPIO Controller BIT 39: SPDIF0 Controller BIT 40: PROTECT Controller BIT 41: SPDIF1 Controller

IOBUS SoftReset Register 0(SWRESET0) 0xB0080004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SWRESET0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SWRESET0[15:0]															

Field	Name	RW	Reset	Description
31-0	SWRESET0	R/W	0x0	Enable signals of SWRESET logic AHB SWRESETs, supplied to each peripheral block, are controlled by SWRESET signal. 0 : disable SWRESET 1: enable SWRESET The bit position indicates each sub-block which is controlled by. BIT 0:Prefetch/Write Buffer BIT 1: EHI0 BIT 2: EHI1 BIT 3: USB OTG BIT 4: NFC controller BIT 5: SD/MMC Channel Control Register BIT 6: SD/MMC0 Controller BIT 7: SD/MMC1 Controller BIT 8: SD/MMC2 Controller BIT 9: GDMA0 Controller BIT 10: GDMA1 Controller BIT 11: GDMA2 Controller BIT 12: PWM Controller 1 BIT 13: Overlay mixer Controller BIT 14: SD/MMC3 Controller BIT 15: Remote Controller BIT 16: TSIF0 Controller 5 BIT 17: TSIF1 Controller 0 BIT 18: ADMA0 Controller 1 BIT 19: DAI0 Controller 2 BIT 20: ADMA1 Controller 3 BIT 21: DAI1 Controller 4 BIT 22: UART Controller 0 BIT 23: UART Controller 1 BIT 24: UART Controller 2 BIT 25: UART Controller 3 BIT 26: UART Controller 4 BIT 27: UART Controller 5 BIT 28 I2C Controller0 BIT 29: I2C Controller1 BIT 30: I2C Controller2 BIT 31: GPSB0 Controller BIT 32: GPSB1 Controller BIT 33: GPSB2 Controller BIT 34: GPSB3 Controller BIT 35 GPSB4 Controller BIT 36: GPSB5 Controller BIT 37: TSADC Controller BIT 38: GPIO Controller BIT 39: SPDIF0 Controller BIT 40: PROTECT Controller BIT 41: SPDIF1 Controller

IOBUS HCLK MASK Register 1(HCLKMASK1) 0xB0080008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						HCLKMASK1[9:0]									

Field	Name	RW	Reset	Description
31-0	HCLKMASK1	R/W	0x0	Enable signals of HCLK clock gating logic AHB HCLKs, supplied to each peripheral block, are controlled by clock enable signal. 0 : enable clock 1: disable clock The bit position indicates each sub-block which is controlled by. BIT 0: GPSB1 Controller BIT 1: GPSB2 Controller BIT 2: GPSB3 Controller BIT 3: GPSB4 Controller BIT 4: GPSB5 Controller BIT 5: TSADC Controller BIT 6: GPIO Controller BIT 7: SPDIF0 Controller BIT 8: PROTECT Controller BIT 9: SPDIF1 Controller

IOBUS SoftReset Register 1(SWRESET1) 0xB008000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						SWRESET1[9:0]									

Field	Name	RW	Reset	Description
9-0	SWRESET1	R/W	0x0	Enable signals of SWRESET logic AHB SWRESETs, supplied to each peripheral block, are controlled by SWRESET signal. 0 : disable SWRESET 1: enable SWRESET The bit position indicates each sub-block which is controlled by. BIT 0: GPSB1 Controller BIT 1: GPSB2 Controller BIT 2: GPSB3 Controller BIT 3: GPSB4 Controller BIT 4: GPSB5 Controller BIT 5: TSADC Controller BIT 6: GPIO Controller BIT 7: SPDIF0 Controller BIT 8: PROTECT Controller BIT 9: SPDIF1 Controller

IOBUS APB Wait Counter Register (IOBAPB) 0xB0080010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-										TSWAIT			RTCWAIT		

Field	Name	RW	Reset	Description
31-6	-	-	0	Reserved
5-3	TSWAIT	R/W	0x0	Delay latency added when access to TSADC block
2-0	RTCWAIT	R/W	0x0	Delay latency added when access to RTC block

IOBUS Miscellaneous Configruation Register(IO_MCFG) 0xB0080014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BWRAP															RDYCFG

Field	Name	RW	Reset	Description
31-16	-	-	0	Reserved
15	BWRAP	R/W	0x0	0 : Overlay mixer BWRAP off 1 : Overlay mixer BWRAP on
0	RYCFG	R/W	0x0	NFC Ready Configuration 0 : The four ready input(ND_RDY0~3) of NFC is used for generating internal ready end flag. Each ready input(ND_RDY0~3) can be masked when the corresponding ND_nCS is high. If anyone of non-masked ready inputs turn into logic '1' from logic '0', ready end flag occure. 1 : The ready 0(ND_RDY0) input of NFC is used for generating internal ready end flag. If ND_RDY0 turn into logic '1' from logic '0', the ready end flag occure.

IOBUS AHB2AXI Control Register (IO_A2X) 0xB0080040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									A2XMOD1				A2XMOD0		

Field	Name	RW	Reset	Description
31-6	-	-	0	Reserved
5-3	A2XMOD1	R/W	0x7	IOBUS to Memory Controller interface1 control register It affects READ operation from following bus master to memory A2XMOD1[0] : flushs prefetch buffer when bus state is IDLE or WRITE. A2XMOD1[1] : stop reading prefetch buffer during WRITE operation is not completed. A2XMOD1[2] : not used
2-0	A2XMOD0	R/W	0x7	IOBUS to Memory Controller interface0 control register It affects READ operation from following bus master to memory A2XMOD0[0] : flushs prefetch buffer when bus state is IDLE or WRITE. A2XMOD0[1] : stop reading prefetch buffer during WRITE operation is not completed. A2XMOD0[2] : not used

PART6 – HSIO BUS
NVS2310

Rev. 1.01

Apr 22, 2011

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial Release
2011-04-22	1.01	∗ Correct the register table and document format. ∗ Fill Version Register (VERSION) register table.

TABLE OF CONTENTS

Contents

Figures

Tables

1 Introduction

NVS2310 HSIOBUS(High Speed I/O BUS) provides connections between the internal bus and external device controller such as USBOTG, GMAC, etc. The sort of interfaces supported is as follows : high speed interface, storage interface, miscellaneous interface.

[FEATURES]

- STORAGE INTERFACES
  - Memory Stick controller
    - MS/MS PRO/MS PRO-HG
- HIGH SPEED INTERFACES
  - High speed USB2.0 FS/HS OTG controller
  - GMAC
- MISCELLANEOUS INTERFACE
  - CIPHER

2 Bus Architecture

Figure 2.1 shows the HSIO bus overall architecture.

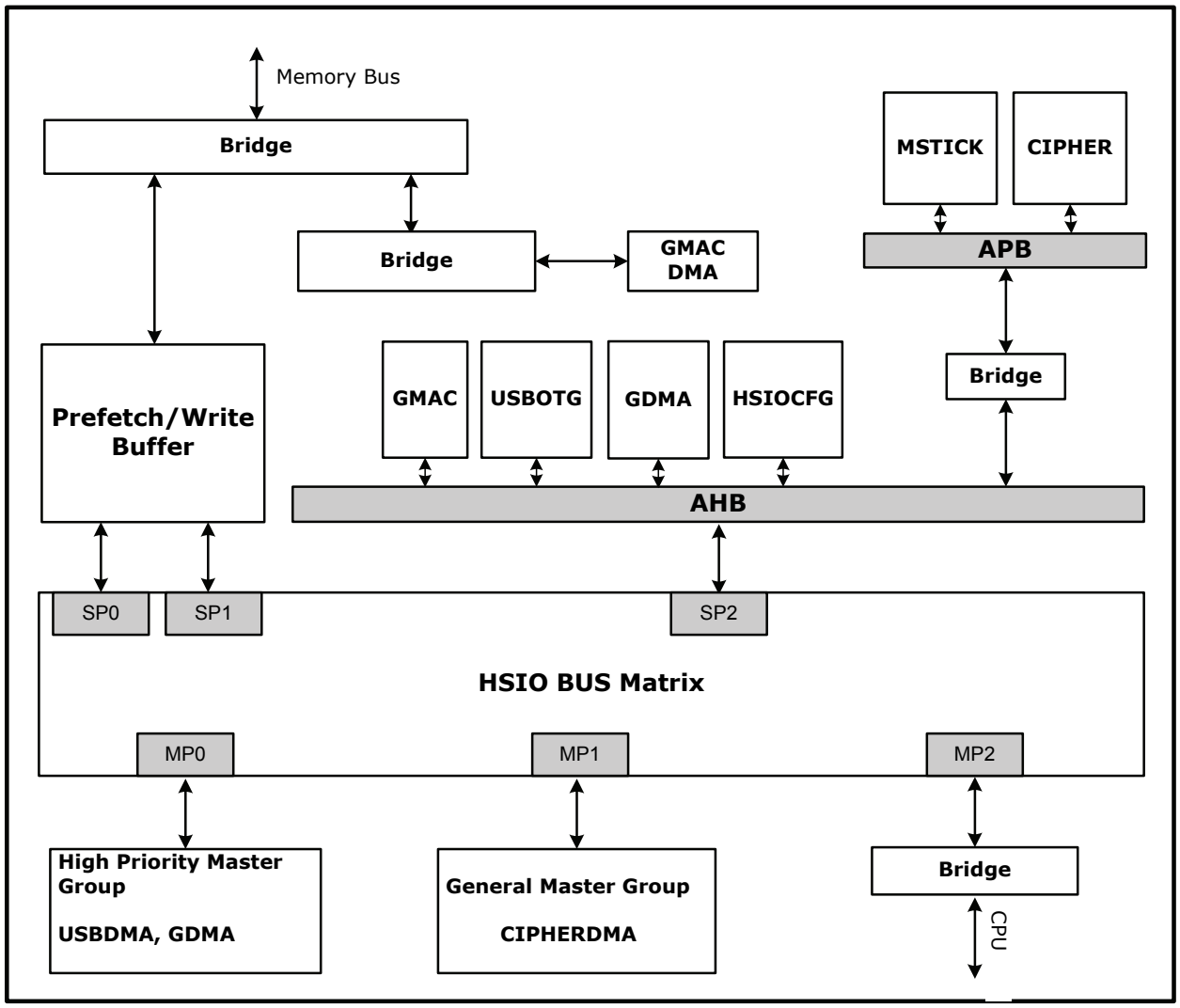


Figure 2.1 The HS I/O Hardware Bus Architecture

3 Address and Register Map

The NVS2310 has various peripherals for specific interface controllers or on-chip hardware components. These peripherals can be configured appropriately by its own registers that can be accessed through specially allocated address. These address maps are represented in the following table.

Refer to corresponding sections for detail information of each peripheral.

Base Address		Peripherals
0xB0800000		USB2.0 OTG1 Controller
0xB0820000		GMAC Contoller
0xB0830000		GDMA3 Controller
0xB0840000		Memory Stick Host Controller
0xB0870000		CIPHER
0xB0880000		HSIO BUS Configuration

4 USB 2.0 OTG Controller

4.1 Overview

The NVS2310 supports Dual-Role Device (DRD) controller, which supports both device and host functions and is fully compliant with the On-The-Go Supplement to the USB 2.0 Specification, Revision 1.0a.

It can also be configured as a Host-only or Device-only controller, fully compliant with the USB 2.0 Specification. The USB 2.0 configurations support high-speed (HS, 480-Mbps), full-speed (FS, 12-Mbps), and low-speed (LS, 1.5-Mbps) transfers. Additionally, it can be configured as a USB 1.1 full-speed/low-speed DRD.

The main features of NVS2310 USB 2.0 OTG controller are as follows.

[GENERAL FEATURES]

- Supports Slave, External DMA Controller Interface
- Includes USB power management features
- Includes power-saving features (clock gating, two power rails for advanced power management)
- Supports packet-based, Dynamic FIFO memory allocation for endpoints for small FIFOs and flexible, efficient use of RAM
- Uses single-port RAM
- Provides support to change an endpoint's FIFO memory size during transfers
- Supports endpoint FIFO sizes that are not powers of 2, to allow the use of contiguous memory locations
- Supports the Keep-Alive in Low-Speed mode and SOFs in High/Full-Speed modes
- Optional support for Transmit and Receive thresholding in DMA mode when dedicated Tx FIFO is selected in Device mode. Thresholding and threshold length selectable through global registers. For supporting thresholding, the AHB must be run at 60 MHz or higher.

[SOFTWARE FEATURES]

- Software handles USB commands (SETUP transactions are detected and their command payloads are forwarded to the application for decoding).
- Software handles USB errors.

[APPLICATION FEATURES]

- Interfaces for the application via the AHB:
- AHB Slave interface for accessing Control and Status Registers (CSRs), the Data FIFO, and queues
- Optional AHB Master interface for Data FIFO access when Internal DMA is enabled
- Supports all AHB burst types in AHB Slave interface
- Software-selectable AHB burst type on AHB Master interface
- Takes care of the 1KB boundary breakup.
- Optional support for a dedicated transmit FIFO for each of the device IN endpoints in Slave and DMA modes. Each FIFO can hold multiple packets.

[USB 2.0 SUPPORTED FEATURES]

- Complies with the On-The-Go Supplement to the USB 2.0 Specification (Revision 1.0a)
- Operates in High-Speed (HS, 480-Mbps), Full-Speed (FS, 12-Mbps) and Low-Speed (LS, 1.5-Mbps) modes
  Caution : In host role controller, Low-Speed device connection through Full-Speed Hub is not supported.
- Supports the UTMI+ Level 3 interface (Revision 1.0, February 25th, 2004). 8-, 16-, and 8/16-bit data buses are supported.
- Supports Session Request Protocol (SRP)

- Supports Host Negotiation Protocol (HNP)
- Supports up to 16 bidirectional endpoints, including control endpoint 0
- Supports up to 16 host channels. In Host mode, when the number of device endpoints to be supported is more than the number of host channels, software can reprogram the channels to support up to 127 devices, each having 32 endpoints (IN + OUT), for a maximum of 4, 064 endpoints.
- Supports a generic root hub
- Includes automatic ping capabilities

[POWER FEATURES]

- PHY clock gating support during USB Suspend mode and Session-Off mode
- AHB clock gating support during USB Suspend mode and Session-Off mode

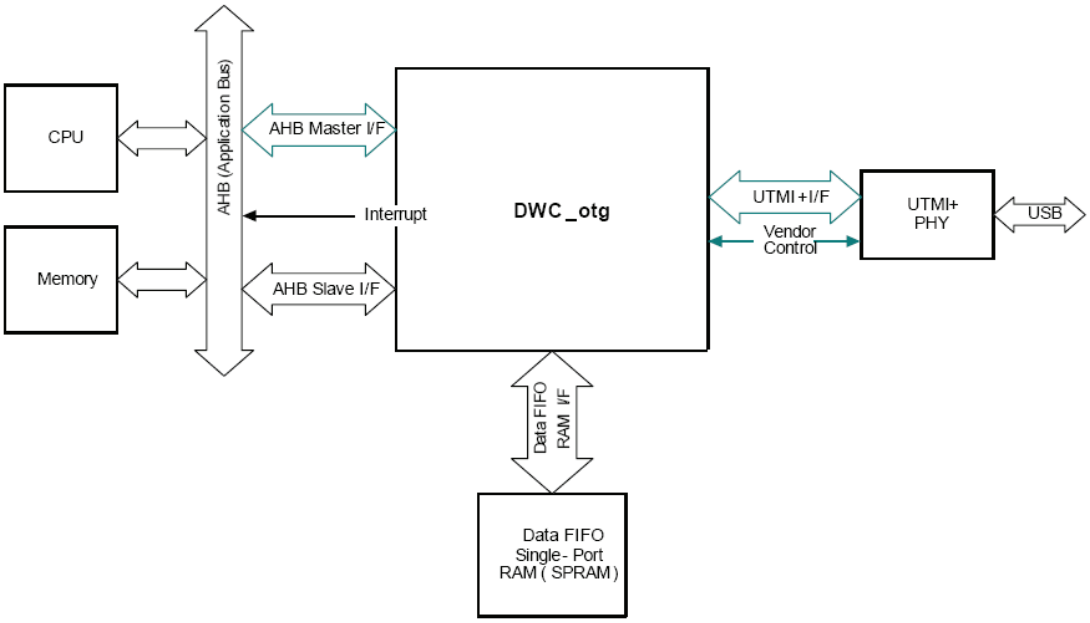


Figure 4.1 USB Controller Block Diagram

4.2 Register Description for USB 2.0 OTG Controller

By reading from and writing to the Control and Status Registers (CSRs) through the AHB Slave interface. CSRs are classified as follows:

- Core Global Registers
- Host Mode Registers
- -Host Global Registers
- -Host Port CSRs
- -Host Channel-Specific Registers
- Device Mode Registers
- -Device Global Registers
- -Device Endpoint-Specific Registers
- Power and Clock-Gating Registers
- Data FIFO (DFIFO) Access Registers

Only the Core Global, Power and Clock Gating, Data FIFO Access, and Host Port registers can be accessed in both Host and Device modes. When operating in one mode, either Device or Host, the application must not access registers from the other mode. If an illegal access occurs, a Mode Mismatch interrupt is generated and reflected in the Core Interrupt register (GINTSTS.ModeMis).

When the core switches from one mode to another, the registers in the new mode of operation must be reprogrammed as they would be after a power-on reset.

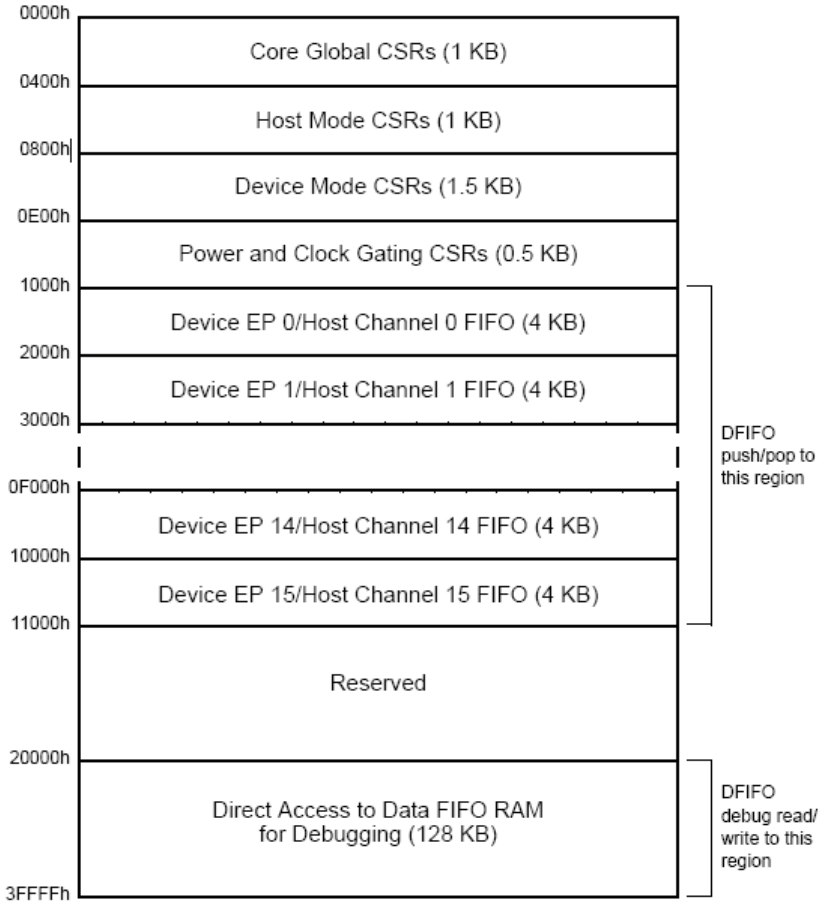


Figure 4.2 USB OTG CSR Memory Map

Table 4.1 USB Register Map (Base Address = 0xB0800000)

Core Global CSR Map			
Name	**Address**	**Reset**	**Description**
GOTGCTL	0x000	0x00010000	OTG Control and Status Register
GOTGINT	0x004	0x00000000	OTG Interrupt Register
GAHBCFG	0x008	0x00000000	Core AHB Configuration Register
GUSBCFG	0x00C	0x00001400	Core USB Configuration Register
GRSTCTL	0x010	0x80000000	Core Reset Register
GINTSTS	0x014	0x04000000	Core Interrupt Register
GINTMSK	0x018	0x00000000	Core Interrupt Mask Register
GRXSTSR	0x01C	0x00000000	Receive Status Debug Read Register (Read Only)
GRXSTSP	0x020	0x00000000	Receive Status Read /Pop Register (Read Only)
GRXFSIZ	0x024	0x00000000	Receive FIFO Size Register
GNPTXFSIZ	0x028	0x00000000	Non-periodic Transmit FIFO Size Register
GNPTXSTS	0x02C	0x00000000	Non-periodic Transmit FIFO/Queue Status Register
	0x030-0x038		Reserved
GUID	0x03C	0x01234567	User ID Register
	0x040		Reserved
GHWCFG1	0x044	0x00000000	User HW Config1 Register(Read Only)
GHWCFG2	0x048	0x228FFCF0	User HW Config2 Register(Read Only)
GHWCFG3	0x04C	0x800000E9	User HW Config3 Register(Read Only)
GHWCFG4	0x050	0x1FF08030	User HW Config4 Register(Read Only)
	0x054-0x0FF		Reserved
HPTXFSIZ	0x100	0x00000000	Host Periodic Transmit FIFO Size Register
DIEPTXFn	0x104-0x13C	0x00000000	Device IN Endpoint Transmit FIFO Size Register
	0x140-0x3FF		Reserved
Host Mode CSR Map			
Name	**Address**	**Reset**	**Description**
HCFG	0x400	0x00000000	Host Configuration Register
HFIR	0x404	0x0000EA60	Host Frame Interval Register
HFNUM	0x408	0x00003FFF	Host Frame Number/Frame Time Remaining Register
	0x40C		Reserved
HPTXSTS	0x410	0x00000000	Host Periodic Transmit FIFO/Queue Status Register
HAINT	0x414	0x00000000	Host All Channels Interrupt Register
HAINTMSK	0x418	0x00000000	Host All Channels Interrupt Mask Register
HPRT	0x440	0x00000000	Host Port Control and Status Register
	0x444-0x4FC		Reserved
HCCHARn	0x500 + n * 0x20	0x00000000	Host Channel n Characteristics Register (n=0~15)
HCSPLTn	0x504 + n * 0x20	0x00000000	Host Channel n Split Control Register (n=0~15)
HCINTn	0x508 + n * 0x20	0x00000000	Host Channel n Interrupt Register (n=0~15)
HCINTMSKn	0x50C + n * 0x20	0x00000000	Host Channel n Interrupt Mask Register (n=0~15)
HCTSIZn	0x510 + n * 0x20	0x00000000	Host Channel n Transfer Size Register (n=0~15)
HCDMAn	0x514 + n * 0x20	0x00000000	Host Channel n DMA Address Register (n=0~15)
	0x518 + n * 0x20 - 0x51C + n * 0x20		Reserved
	0x6FD-0x7FF		Reserved
Device Mode CSR Map			
Name	**Address**	**Reset**	**Description**
DCFG	0x800	0x00000000	Device Configuration Register
DCTL	0x804	0x00000000	Device Control Register
DSTS	0x808	0x00000002	Device Status Register (Read Only)
	0x80C		Reserved
DIEPMSK	0x810	0x00000000	Device IN Endpoint Common Interrupt Mask Register
DOEPMSK	0x814	0x00000000	Device OUT Endpoint Common Interrupt Mask Register
DAINT	0x818	0x00000000	Device All Endpoints Interrupt Register
DAINTMSK	0x81C	0x00000000	Device All Endpoints Interrupt Mask Register
	0x820-0x824		Reserved
	0x830-0x834		Reserved
DVBUSDIS	0x828	0x00000B8F	Device VBUS Discharge Time Register
DVBUSPULSE	0x82C	0x000002C6	Device VBUS Pulsing Time Register
DTHRCTL	0x830	0x01000000	Device Threshold Control Register
DIEPEMPMSK	0x834	0x00000000	Device IN Endpoint FIFO Empty Interrupt Mask Register
	0x838-0x8FF		Reserved

DIEPCTL0	0x900	0x00001000	Device Control IN Endpoint 0 Control Register
DIEPCTLn	0x900 + (n * 0x20)	0x00000000	Device Control IN Endpoint n Control Register (n=1~15)
	0x904 + (n * 0x20)		Reserved (n=0~15)
DIEPINTn	0x908 + (n * 0x20)	0x00000080	Device IN Endpoint n Interrupt Register (n=0~15)
	0x90C + (n * 0x20)		Reserved (n=0~15)
DIEPTSIZ0	0x910	0x00000000	Device IN Endpoint 0 Transfer Size Register
DIEPTSIZn	0x910 + (n * 0x20)	0x00000000	Device IN Endpoint n Transfer Size Register (n=1~15)
DIEPDMAn	0x914 + (n * 0x20)	0x00000000	Device IN Endpoint n DMA Address Register (n=0~15)
DTXFSTSn	0x918 + (n * 0x20)	0x00000000	Device IN Endpoint Transmit FIFO Status Register (n=0~15)
	0x918 + (n * 0x20) - 0x91C + (n * 0x20)		Reserved (n=0~15)
DOEPCTL0	0xB00	0x00001000	Device Control OUT Endpoint 0 Control Register
DOEPCTLn	0xB00 + (n * 0x20)	0x00000000	Device Control OUT Endpoint n Control Register (n=1~15)
	0xB04 + (n * 0x20)		Reserved (n=0~15)
DOEPINTn	0xB08 + (n * 0x20)	0x00000080	Device OUT Endpoint n Interrupt Register (n=0~15)
	0xB0C + (n * 0x20)		Reserved (n=0~15)
DOEPTSIZ0	0xB10	0x00000000	Device OUT Endpoint 0 Transfer Size Register
DOEPTSIZn	0xB10 + (n * 0x20)	0x00000000	Device OUT Endpoint 0 Transfer Size Register (n=1~15)
DOEPDMAn	0xB14 + (n * 0x20)	0x00000000	Device OUT Endpoint 0 DMA Address Register (n=0~15)
	0xB18 + (n * 0x20) - 0xB1C + (n * 0x20)		Reserved (n=0~15)
	0xCFD-0xDFF		Reserved

Power and Clock Gating CSR Map

Name	Address	Reset	Description
PCGCR	0xE00	0x00000000	Power and Clock Gating Control Register
	0xE04-0xFFF		Reserved

Data FIFO(DFIFO) Access Register Map

Name	Address	Type	Reset	Description
DFIFO0W	0x1000-0x1FFC	W	-	Device IN Endpoint 0/Host OUT Channel 0 : DFIFO Write Access
DFIFO0R		R	-	Device OUT Endpoint 0/Host IN Channel 0 : DFIFO Read Access
DFIFO1W	0x2000-0x1FFC	W	-	Device IN Endpoint 1/Host OUT Channel 1 : DFIFO Write Access
DFIFO1R		R	-	Device OUT Endpoint 1/Host IN Channel 1 : DFIFO Read Access
~	~			~
DFIFO14W	0xF000-0xFFFC	W	-	Device IN Endpoint 14/Host OUT Channel 14 : DFIFO Write Access
DFIFO14R		R	-	Device OUT Endpoint 14/Host IN Channel 14 : DFIFO Read Access
DFIFO15W	0x10000-0x10FFC	W	-	Device IN Endpoint 15/Host OUT Channel 15 : DFIFO Write Access
DFIFO15R		R	-	Device OUT Endpoint 15/Host IN Channel 15 : DFIFO Read Access

Table 4.2 USB OTG Register (Base Address = 0xB0880020)

Name	Addr.	Description
OTGCR	0x0	**USBOTG Configuration Register**

OTG Control and Status Register (GOTGCTL) 0xB0800000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
												BSesVld	ASesVld	DbncTime	ConIDSts
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				DevHNPEn	HstSetHNPEn	HNPReq	HstNegScs							SesReq	SesReqScs

The OTG Control and Status register controls the behavior and reflects the status of the OTG function of the core.

Field	Name	Mode	RW	Reset	Description
31-20	-	-	-	-	Reserved
19	BSesVld	Device	R	0x0	B-Session Valid Indicates the Device mode transceiver status. • 0: B-session is not valid. • 1: B-session is valid. In OTG mode, you can use this bit to determine if the device is connected or disconnected.
18	ASesVld	Host	R	0x0	A-Session Valid Indicates the Host mode transceiver status. • 0: A-session is not valid • 1: A-session is valid
17	DbncTime	Host	R	0x0	Long/Short Debounce Time Indicates the debounce time of a detected connection. • 0: Long debounce time, used for physical connections (100 ms + 2.5 μs) • 1: Short debounce time, used for soft connections (2.5 μs)
16	ConIDSts	Host Device	R	0x1	Connector ID Status Indicates the connector ID status on a connect event. • 0: The OTG core is in A-Device mode • 1: The OTG core is in B-Device mode
15-12	-	-	-	0x0	Reserved
11	DevHNPEn	Device	R/W	0x0	Device HNP Enabled The application sets this bit when it successfully receives a SetFeature.SetHNPEnable command from the connected USB host. • 0: HNP is not enabled in the application • 1: HNP is enabled in the application
10	HstSetHNPEn	Host	R/W	0x0	Host Set HNP Enable The application sets this bit when it has successfully enabled HNP (using the SetFeature. SetHNPEnable command) on the connected device. • 0: Host Set HNP is not enabled • 1: Host Set HNP is enabled
9	HNPReq	Device	R/W	0x0	HNP Request The application sets this bit to initiate an HNP request to the connected USB host. The application can clear this bit by writing a 0 when the Host Negotiation Success Status Change bit in the OTG Interrupt register (GOTGINT.HstNegSucStsChng) is set. The core clears this bit when the HstNegSucStsChng bit is cleared. • 0: No HNP request • 1: HNP request
8	HstNegScs	Device	R	0x0	Host Negotiation Success The core sets this bit when host negotiation is successful. The core clears this bit when the HNP Request (HNPReq) bit in this register is set. • 0: Host negotiation failure • 1: Host negotiation success
7-2	-	-	-	0x0	Reserved
1	SesReq	Device	R/W	0x0	Session Request The application sets this bit to initiate a session request on the USB. The application can clear this bit by writing a 0 when the Host Negotiation Success Status Change bit in the OTG Interrupt register (GOTGINT.HstNegSucStsChng) is set. The core clears this

					bit when the HstNegSucStsChng bit is cleared. If you use the USB 1.1 Full-Speed Serial Transceiver interface to initiate the session request, the application must wait until the VBUS discharges to 0.2 V, after the B-Session Valid bit in this register (GOTGCTL.BSesVld) is cleared. This discharge time varies between different PHYs and can be obtained from the PHY vendor. • 0: No session request • 1: Session request
0	SesReqScs	Device	R	0x0	Session Request Success The core sets this bit when a session request initiation is successful. • 0: Session request failure • 1: Session request success

OTG Interrupt Register (GOTGINT) 0xB0800004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
												DbnceDone	ADevTOUTChg	HstNegDet	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						HstNegSucStsChng	SesReqSucStsChng						SesEndDet		

The application reads this register whenever there is an OTG interrupt and clears the bits in this register to clear the OTG interrupt.

Field	Name	Mode	RW	Reset	Description
31-20	-	-	-	-	Reserved
19	DbnceDone	Host	R/W	0x0	Debounce Done The core sets this bit when the debounce is completed after the device connect. The application can start driving USB reset after seeing this interrupt. This bit is only valid when the HNP Capable or SRP Capable bit is set in the Core USB Configuration register (GUSBCFG.HNPCap or GUSBCFG.SRPCap, respectively).
18	ADevTOUTChg	Host Device	R/W	0x0	A-Device Timeout Change () The core sets this bit to indicate that the A-device has timed out while waiting for the B-device to connect.
17	HstNegDet	Host Device	R/W	0x0	Host Negotiation Detected The core sets this bit when it detects a host negotiation request on the USB.
16-10	-	-	-	0x0	Reserved
9	HstNegSucStsChng	Host Device	R/W	0x0	Host Negotiation Success Status Change The core sets this bit on the success or failure of a USB host negotiation request. The application must read the Host Negotiation Success bit of the OTG Control and Status register (GOTGCTL.HstNegScs) to check for success or failure.
8	SesReqSucStsChng	Host Device	R/W	0x0	Session Request Success Status Change The core sets this bit on the success or failure of a session request. The application must read the Session Request Success bit in the OTG Control and Status register (GOTGCTL.SesReqScs) to check for success or failure.
7-3	-	-	-	0x0	Reserved
2	SesEndDet	Host Device	R/W	0x0	Session End Detected The core sets this bit when the utmiotg_bvalid signal is deasserted.
1-0	-	-	-	0x0	Reserved

Core AHB Configuration Register (GAHBCFG) 0xB0800008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							PTxFEmpLvl	NPTxFEmpLvl		DMAEn	HBstLen				GlblIntrMsk

This register can be used to configure the core after power-on or a change in mode of operation. This register mainly contains AHB system-related configuration parameters. Do not change this register after the initial programming. The application must program this register before starting any transactions on either the AHB or the USB..

Field	Name	Mode	RW	Reset	Description
31-9	-	-	-	0	Reserved
8	PTxFEmpLvl	Host	R/W	0x0	Periodic TxFIFO Empty Level Indicates when the Periodic TxFIFO Empty Interrupt bit in the Core Interrupt register (GINTSTS.PTxFEmp) is triggered. This bit is used only in Slave mode. • 1’b0: GINTSTS.PTxFEmp interrupt indicates that the Periodic TxFIFO is half empty • 1’b1: GINTSTS.PTxFEmp interrupt indicates that the Periodic TxFIFO is completely empty
7	NPTxFEmpLvl	Host Device	R/W	0x0	Non-Periodic TxFIFO Empty Level This bit is used only in Slave mode. In host mode and with Shared FIFO with device mode, this bit indicates when the Non-Periodic TxFIFO Empty Interrupt bit in the Core Interrupt register (GINTSTS.NPTxFEmp) is triggered. With dedicated FIFO in device mode, this bit indicates when IN endpoint Transmit FIFO empty interrupt (DIEPINTn.TxFEmp) is triggered. • 1’b0: DIEPINTn.TxFEmp interrupt indicates that the IN Endpoint TxFIFO is half empty • 1’b1: DIEPINTn.TxFEmp interrupt indicates that the IN Endpoint TxFIFO is completely empty
6	-	-	-	0x0	Reserved
5	DMAEn	Host Device	R/W	0x0	DMA Enable • 1’b0: Core operates in Slave mode • 1’b1: Core operates in a DMA mode
4-1	HBstLen	Host Device	R/W	0x0	Burst Length/Type () AHB Master burst type: • 4’b0000 Single • 4’b0001 INCR • 4’b0011 INCR4 • 4’b0101 INCR8 • 4’b0111 INCR16 • Others: Reserved
0	GlblIntrMsk	Host Device	R/W	0x0	Global Interrupt Mask The application uses this bit to mask or unmask the interrupt line assertion to itself. Irrespective of this bit’s setting, the interrupt status registers are updated by the core. • 1’b0: Mask the interrupt assertion to the application. • 1’b1: Unmask the interrupt assertion to the application.

Core USB Configuration Register (GUSBCFG) 0xB080000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CorTxP	ForceDevMode	ForceHstMode							TermSelDLPulse						OtgI2CSel
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PhyLPwrClkSel		USBTrdTim				HNPCap	SRPCap		PHYSel	FSIntf	ULPI_UTMI_Sel	PHYIf	TOutCal		

This register can be used to configure the core after power-on or a changing to Host mode or Device mode. It contains USB and USB-PHY related configuration parameters. The application must program this register before starting any transactions on either the AHB or the USB. Do not make changes to this register after the initial programming.

Field	Name	Mode	RW	Reset	Description
31	CorTXP	Host Device	R	0x0	Corrupt Tx packet This bit is for debug purposes only. Never set this bit to 1.
30	ForceDevMode	Host Device	R/W	0x0	Force Device Mode Writing a 1 to this bit will force the core to device mode irrespective of utmiotg_iddig input pin. • 1’b0 : Normal Mode. • 1’b1 : Force Device Mode. After setting the force bit, the application must wait at least 25ms before the change to take effect.
29	ForceHstMode	Host Device	R/W	0x0	Force Host Mode Writing a 1 to this bit will force the core to host mode irrespective of utmiotg_iddig input pin. • 1’b0 : Normal Mode. • 1’b1 : Force Host Mode. After setting the force bit, the application must wait at least 25ms before the change to take effect.
28-23	-	-	-	0x0	Reserved
22	TermSelDLPulse	Device	R/W	0x0	TermSel DLine Pulsing Selection This bit selects utmi_termselect to drive data line pulse during SRP. • 1’b0: Data line pulsing using utmi_txvalid (default). • 1’b1: Data line pulsing using utmi_termsel.
21-17	-	-	-	0x0	Reserved
16	OtgI2CSel	Host Device	R	0x0	UTMIFS or I2C Interface Select The application uses this bit to select the I2C interface. • 1’b0: UTMI USB 1.1 Full-Speed interface for OTG signals • 1’b1: I2C interface for OTG signals(not supported)
15	PhyLPwrClkSel	Host Device	R/W	0x0	PHY Low-Power Clock Select Selects either 480-MHz or 48-MHz (low-power) PHY mode. In FS and LS modes, the PHY can usually operate on a 48-MHz clock to save power. • 1’b0: 480-MHz Internal PLL clock • 1’b1: 48-MHz External Clock In 480 MHz mode, the UTMI interface operates at either 60 or 30-MHz, depending upon whether 8- or 16-bit data width is selected. In 48-MHz mode, the UTMI interface operates at 48MHz in FS mode and at either 48 or 6 MHz in LS mode (depending on the PHY vendor). This bit drives the utmi_fsls_low_power core output signal, and is valid only for UTMI+ PHYs.
14	-	-	-	0x0	Reserved
13-10	USBTrdTim	Device	R/W	0x5	USB Turnaround Time Sets the turnaround time in PHY clocks. Specifies the response time for a MAC request to the Packet FIFO Controller (PFC) to fetch data from the DFIFO (SPRAM). This must be programmed to • 4’h5: When the MAC interface is 16-bit UTMI+ . • 4’h9: When the MAC interface is 8-bit UTMI+ . Note: The values above are calculated for the minimum AHB

					frequency of 30MHz. USB turnaround time is critical for certification where long cables and 5-Hubs are used, so if you need the AHB to run at less than 30 MHz, and if USB turnaround time is not critical, these bits can be programmed to a larger value.
9	HNPCap	Host Device	R/W, R	0x0	HNP-Capable The application uses this bit to control the OTG core's HNP capabilities. • 1'b0: HNP capability is not enabled. • 1'b1: HNP capability is enabled..
8	SRPCap	Host Device	R/W, R	0x0	SRP-Capable The application uses this bit to control SRP capabilities. If the core operates as a non-SRP-capable B-device, it cannot request the connected A-device (host) to activate VBUS and start a session. • 1'b0: SRP capability is not enabled. • 1'b1: SRP capability is enabled.
7	-	-	-	0x0	Reserved
6	PHYSel	Host Device	R/W, W	0x0	USB 2.0 High-Speed PHY or USB 1.1 Full-Speed Serial Transceiver Select The application uses this bit to select either a high-speed UTMI+, or a full-speed transceiver. • 1'b0: USB 2.0 high-speed UTMI+ • 1'b1: USB 1.1 full-speed serial transceiver
5	FSIntf	Host Device	R/W, W	0x0	Full-Speed Serial Interface Select The application uses this bit to select either a unidirectional or bidirectional USB 1.1 full-speed serial transceiver interface. • 1'b0: 6-pin unidirectional full-speed serial interface • 1'b1: 3-pin bidirectional full-speed serial interface
4	ULPI_UTMI_Sel	Host Device	R/W, R	0x0	ULPI or UTMI+ Select The application uses this bit to select either a UTMI+ interface or ULPI Interface. • 1'b0: UTMI+ Interface • 1'b1: ULPI Interface(not supported)
3	PHYIf	Host Device	R/W, R	0x0	PHY Interface The application uses this bit to configure the core to support a UTMI+ PHY with an 8- or 16-bit interface. • 1'b0: 8 bits (not supported) • 1'b1: 16 bits
2-0	TOutCal	Host Device	R/W	0x0	HS/FS Timeout Calibration The number of PHY clocks that the application programs in this field is added to the high-speed/full-speed interpacket timeout duration in the core to account for any additional delays introduced by the PHY. This can be required, because the delay introduced by the PHY in generating the linestate condition can vary from one PHY to another. The USB standard timeout value for high-speed operation is 736 to 816 (inclusive) bit times. The USB standard timeout value for full-speed operation is 16 to 18 (inclusive) bit times. The application must program this field based on the speed of enumeration. The number of bit times added per PHY clock are: High-speed operation: • One 30-MHz PHY clock = 16 bit times • One 60-MHz PHY clock = 8 bit times Full-speed operation: • One 30-MHz PHY clock = 0.4 bit times • One 60-MHz PHY clock = 0.2 bit times • One 48-MHz PHY clock = 0.25 bit times

Core Reset Register (GRSTCTL) 0xB0800010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AHBIdle	DMAReq														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					TxFNum					TxFFlsh	RxFFlsh	INTknQFlsh	FrmCntrRst	HSftRst	CSftRst

The application uses this register to reset various hardware features inside the core..

Field	Name	Mode	RW	Reset	Description
31	AHBIdle	Host Device	R	0x1	AHB Master Idle Indicates that the AHB Master State Machine is in the IDLE condition.
30	DMAReq	Host Device	R	0x0	DMA Request Signal Indicates that the DMA request is in progress. Used for debug.
29-11	-	-	-	0x0	Reserved
10-6	TxFNum	Host Device	R/W	0x0	TxFIFO Number This is the FIFO number that must be flushed using the TxFIFO Flush bit. This field must not be changed until the core clears the TxFIFO Flush bit. • 5’h0: - Non-periodic TxFIFO flush in Host mode - Tx FIFO 0 flush in device mode • 5’h1: - Periodic TxFIFO flush in Host mode - TXFIFO 1 flush in device mode • 5’h2: - TXFIFO 2 flush in device mode ... • 5’hF: - TXFIFO 15 flush in device mode • 5’h10: Flush all the transmit FIFOs in device or host mode.
5	TxFFlsh	Host Device	R/W	0x0	TxFIFO Flush This bit selectively flushes a single or all transmit FIFOs, but cannot do so if the core is in the midst of a transaction. The application must write this bit only after checking that the core is neither writing to the TxFIFO nor reading from the TxFIFO. Verify using these registers: • Read–NAK Effective Interrupt ensures the core is not reading from the FIFO • Write–GRSTCTL.AHBIdle ensures the core is not writing anything to the FIFO. Flushing is normally recommended when FIFOs are reconfigured. FIFO flushing is also recommended during device endpoint disable. The application must wait until the core clears this bit before performing any operations. This bit takes eight clocks to clear, using the slower clock of phy_clk or hclk.
4	RxFFlsh	Host Device	R/W	0x0	RxFIFO Flush The application can flush the entire RxFIFO using this bit, but must first ensure that the core is not in the middle of a transaction. The application must only write to this bit after checking that the core is neither reading from the RxFIFO nor writing to the RxFIFO. The application must wait until the bit is cleared before performing any other operations. This bit requires 8 clocks (slowest of PHY or AHB clock) to clear.
3	INTknQFlsh	Device	R/W	0x0	IN Token Sequence Learning Queue Flush The application writes this bit to flush the IN Token Sequence Learning Queue.
2	FrmCntrRst	Host	R/W	0x0	Host Frame Counter Reset The application writes this bit to reset the (micro)frame number counter inside the core. When the (micro)frame counter is reset, the subsequent SOF sent out by the core has a (micro)frame number of 0.
1	HSftRst	Host	R/W	0x0	HClk Soft Reset

		Device			The application uses this bit to flush the control logic in the AHB Clock domain. Only AHB Clock Domain pipelines are reset. • FIFOs are not flushed with this bit. • All state machines in the AHB clock domain are reset to the Idle state after terminating the transactions on the AHB, following the protocol. • CSR control bits used by the AHB clock domain state machines are cleared. • To clear this interrupt, status mask bits that control the interrupt status and are generated by the AHB clock domain state machine are cleared. • Because interrupt status bits are not cleared, the application can get the status of any core events that occurred after it set this bit. This is a self-clearing bit that the core clears after all necessary logic is reset in the core. This can take several clocks, depending on the core’ s current state.
0	CSftRst	Host Device	R/W	0x0	Core Soft Reset Resets the hclk and phy_clock domains as follows: • Clears the interrupts and all the CSR registers except the following register bits: - PCGCCTL.RstPdwnModule - PCGCCTL.GateHclk - PCGCCTL.PwrClmp - PCGCCTL.StopPPhyLPwrClkSelclk - GUSBCFG.PhyLPwrClkSel - GUSBCFG.DDRSel - GUSBCFG.PHYSel - GUSBCFG.FSIntf - GUSBCFG.ULPI_UTMI_Sel - GUSBCFG.PHYIf - HCFG.FSLSPclkSel - DCFG.DevSpd - GGPIO • All module state machines (except the AHB Slave Unit) are reset to the IDLE state, and all the transmit FIFOs and the receive FIFO are flushed. • Any transactions on the AHB Master are terminated as soon as possible, after gracefully completing the last data phase of an AHB transfer. Any transactions on the USB are terminated immediately. The application can write to this bit any time it wants to reset the core. This is a self-clearing bit and the core clears this bit after all the necessary logic is reset in the core, which can take several clocks, depending on the current state of the core. Once this bit is cleared software must wait at least 3 PHY clocks before doing any access to the PHY domain (synchronization delay). Software must also must check that bit 31 of this register is 1 (AHB Master is IDLE) before starting any operation. Typically software reset is used during software development and also when you dynamically change the PHY selection bits in the USB configuration registers listed above. When you change the PHY, the corresponding clock for the PHY is selected and used in the PHY domain. Once a new clock is selected, the PHY domain has to be reset for proper operation.

Core Interrupt Register (GINTSTS) 0xB0800014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WkUpInt	SessReqInt	DisconnInt	ConIDStsChng		PTxFEmp	HChInt	PrtInt		FetSusp	incompIP/ incompISOOUT	incompISOIN	OEPInt	IEPInt	EPMis	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EOPF	ISOOutDrop	EnumDone	USBRst	USBSusp	ErlySusp			GOUTNakEff	GINNakEff		RxFLvl	Sof	OTGInt	ModeMis	CurMod

This register interrupts the application for system-level events in the current mode of operation (Device mode or Host mode). Some of the bits in this register are valid only in Host mode, while others are valid in Device mode only. This register also indicates the current mode of operation. In order to clear the interrupt status bits of type R/W, the application must write 1'b1 into the bit.

The FIFO status interrupts are read only; once software reads from or writes to the FIFO while servicing these interrupts, FIFO interrupt conditions are cleared automatically.

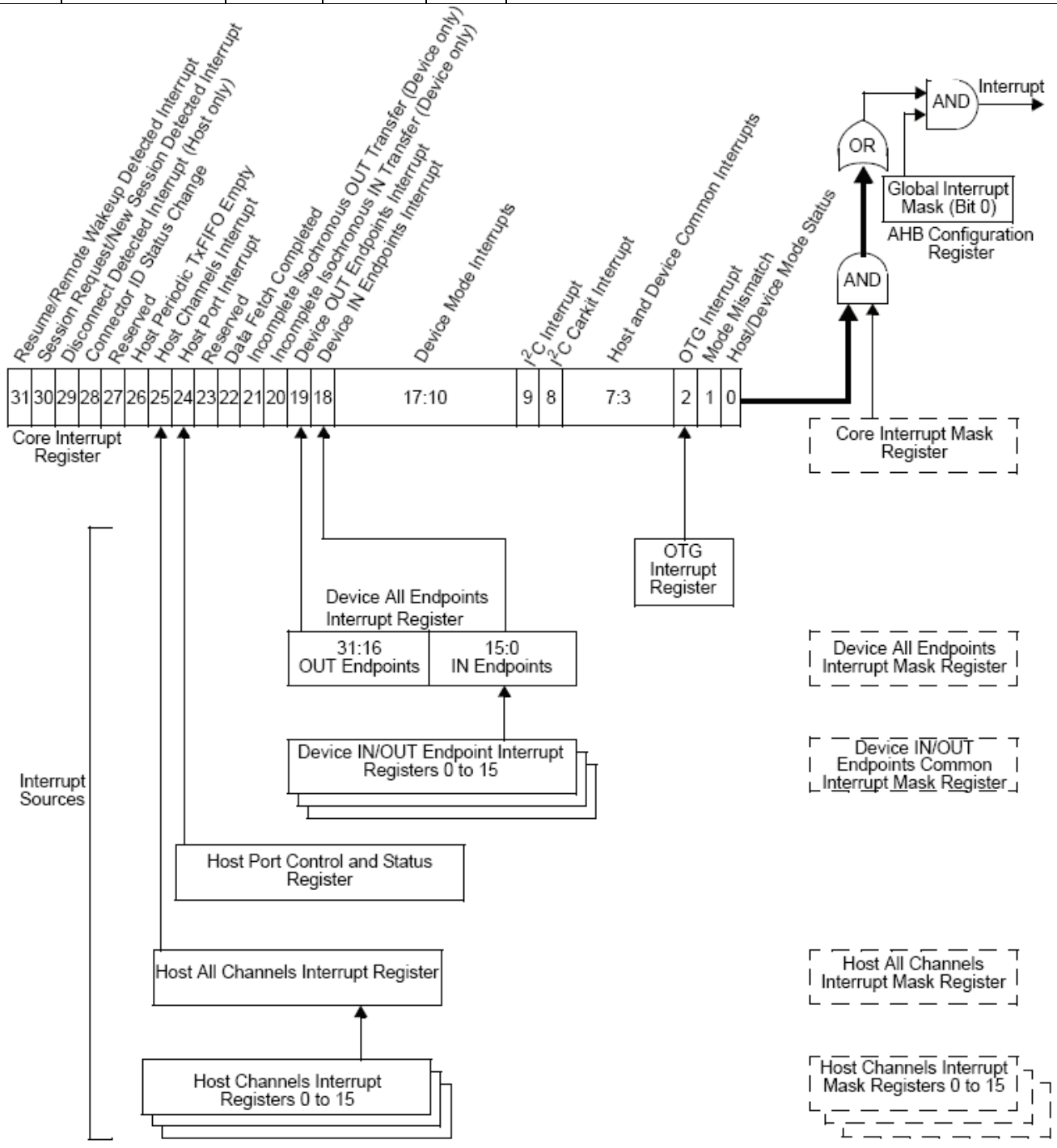
The Application must clear the GINTSTS register at initialization before unmasking the interrupt bit to avoid any interrupts generated prior to initialization.

Field	Name	Mode	RW	Reset	Description
31	WkUpInt	Host Device	R/W	0x0	Resume/Remote Wakeup Detected Interrupt In Device mode, this interrupt is asserted when a resume is detected on the USB. In Host mode, this interrupt is asserted when a remote wakeup is detected on the USB.
30	SessReqInt	Host Device	R/W	0x0	Session Request/New Session Detected Interrupt In Host mode, this interrupt is asserted when a session request is detected from the device. In Device mode, this interrupt is asserted when the utmiotg_bvalid signal goes high.
29	DisconnInt	Host	R/W	0x0	Disconnect Detected Interrupt Asserted when a device disconnect is detected.
28	ConIDStsChng	Host Device	R/W	0x0	Connector ID Status Change The core sets this bit when there is a change in connector ID status.
27	-	-	-	0x0	Reserved
26	PTxFEmp	Host	R	0x1	Periodic TxFIFO Empty Asserted when the Periodic Transmit FIFO is either half or completely empty and there is space for at least one entry to be written in the Periodic Request Queue. The half or completely empty status is determined by the Periodic TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.PTxFEmpLvl).
25	HChInt	Host	R	0x0	Host Channels Interrupt The core sets this bit to indicate that an interrupt is pending on one of the channels of the core (in Host mode). The application must read the Host All Channels Interrupt (HAINT) register to determine the exact number of the channel on which the interrupt occurred, and then read the corresponding Host Channel-n Interrupt (HCINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the HCINTn register to clear this bit.
24	PrtInt	Host	R	0x0	Host Port Interrupt The core sets this bit to indicate a change in port status of one of ports in Host mode. The application must read the Host Port Control and Status (HPRT) register to determine the exact event that caused this interrupt. The application must clear the appropriate status bit in the Host Port Control and Status register to clear this bit.
23	-	-	-	0x0	Reserved
22	FetSusp	Device	R/W	0x0	Data Fetch Suspended This interrupt is valid only in DMA mode. This interrupt indicates that the core has stopped fetching data for IN endpoints due to the unavailability of TxFIFO space or Request Queue space.

					This interrupt is used by the application for an endpoint mismatch algorithm. For example, after detecting an endpoint mismatch, the application: • Sets a global non-periodic IN NAK handshake • Disables In endpoints • Flushes the FIFO • Determines the token sequence from the IN Token Sequence Learning Queue • Re-enables the endpoints • Clears the global non-periodic IN NAK handshake If the global non-periodic IN NAK is cleared, the core has not yet fetched data for the IN endpoint, and the IN token is received: the core generates an “IN token received when FIFO empty” interrupt. The OTG then sends the host a NAK response. To avoid this scenario, the application can check the GINTSTS.FetSusp interrupt, which ensures that the FIFO is full before clearing a global NAK handshake. Alternatively, the application can mask the “IN token received when FIFO empty” interrupt when clearing a global IN NAK handshake.
21	incomplP	Host	R/W	0x0	Incomplete Periodic Transfer In Host mode, the core sets this interrupt bit when there are incomplete periodic transactions still pending which are scheduled for the current microframe.
	incompISOOUT	Device			Incomplete Isochronous OUT Transfer The Device mode, the core sets this interrupt to indicate that there is at least one isochronous OUT endpoint on which the transfer is not completed in the current microframe. This interrupt is asserted along with the End of Periodic Frame Interrupt (EOPF) bit in this register.
20	incompISOIN	Device	R/W	0x0	Incomplete Isochronous IN Transfer The core sets this interrupt to indicate that there is at least one isochronous IN endpoint on which the transfer is not completed in the current microframe. This interrupt is asserted along with the End of Periodic Frame Interrupt (EOPF) bit in this register.
19	OEPInt	Device	R	0x0	OUT Endpoints Interrupt The core sets this bit to indicate that an interrupt is pending on one of the OUT endpoints of the core (in Device mode). The application must read the Device All Endpoints Interrupt (DAINT) register to determine the exact number of the OUT endpoint on which the interrupt occurred, and then read the corresponding Device OUT Endpoint-n Interrupt (DOEPINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the corresponding DOEPINTn register to clear this bit.
18	IEPInt	Device	R	0x0	IN Endpoints Interrupt The core sets this bit to indicate that an interrupt is pending on one of the IN endpoints of the core (in Device mode). The application must read the Device All Endpoints Interrupt (DAINT) register to determine the exact number of the IN endpoint on which the interrupt occurred, and then read the corresponding Device IN Endpoint-n Interrupt (DIEPINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the corresponding DIEPINTn register to clear this bit.
17	EPMis	Device	R	0x0	Endpoint Mismatch Interrupt
16	-	-	-	0x0	Reserved
15	EOPF	Device	R/W	0x0	End of Periodic Frame Interrupt Indicates that the period specified in the Periodic Frame Interval field of the Device Configuration register (DCFG.PerFrInt) has been reached in the current microframe.
14	ISOOutDrop	Device	R/W	0x0	Isochronous OUT Packet Dropped Interrupt The core sets this bit when it fails to write an isochronous OUT

					packet into the RxFIFO because the RxFIFO doesn't have enough space to accommodate a maximum packet size packet for the isochronous OUT endpoint.
13	EnumDone	Device	R/W	0x0	Enumeration Done The core sets this bit to indicate that speed enumeration is complete. The application must read the Device Status (DSTS) register to obtain the enumerated speed.
12	USBRst	Device	R/W	0x0	USB Reset The core sets this bit to indicate that a reset is detected on the USB.
11	USBSusp	Device	R/W	0x0	USB Suspend The core sets this bit to indicate that a suspend was detected on the USB. The core enters the Suspended state when there is no activity on the phy_line_state_i signal for an extended period of time.
10	ErlySusp	Device	R/W	0x0	Early Suspend The core sets this bit to indicate that an Idle state has been detected on the USB for 3 ms.
9-8	-	-	-	0x0	Reserved
7	GOUTNakEff	Device	R/W	0x0	Global OUT NAK Effective Indicates that the Set Global OUT NAK bit in the Device Control register (DCTL.SGOUTNak), set by the application, has taken effect in the core. This bit can be cleared by writing the Clear Global OUT NAK bit in the Device Control register (DCTL.CGOUTNak).
6	GINNakEff	Device	R/W	0x0	Global IN Non-periodic NAK Effective Indicates that the Set Global Non-periodic IN NAK bit in the Device Control register (DCTL.SGNPInNak), set by the application, has been taken effect in the core. That is, the core has sampled the Global IN NAK bit set by the application. This bit can be cleared by clearing the Clear Global Non-periodic IN NAK bit in the Device Control register (DCTL.CGNPInNak). This interrupt does not necessarily mean that a NAK handshake is sent out on the USB. The STALL bit takes precedence over the NAK bit.
5	-	-	-	0x0	Reserved
4	RxFLvl	Host Device	R	0x0	RxFIFO Non-Empty Indicates that there is at least one packet pending to be read from the RxFIFO.
3	Sof	Host Device	R/W	0x0	Start of (micro)Frame In Host mode, the core sets this bit to indicate that an SOF (FS), micro-SOF (HS), or Keep-Alive (LS) is transmitted on the USB. The application must write a 1 to this bit to clear the interrupt. In Device mode, in the core sets this bit to indicate that an SOF token has been received on the USB. The application can read the Device Status register to get the current (micro)frame number. This interrupt is seen only when the core is operating at either HS or FS.
2	OTGInt	Host Device	R	0x0	OTG Interrupt The core sets this bit to indicate an OTG protocol event. The application must read the OTG Interrupt Status (GOTGINT) register to determine the exact event that caused this interrupt. The application must clear the appropriate status bit in the GOTGINT register to clear this bit.
1	ModeMis	Host Device	R/W	0x0	Mode Mismatch Interrupt The core sets this bit when the application is trying to access: • A Host mode register, when the core is operating in Device mode • A Device mode register, when the core is operating in Host mode The register access is completed on the AHB with an OKAY response, but is ignored by the core internally and doesn't affect the operation of the core.
0	CurMod	Host Device	R	0x0	Current Mode of Operation Indicates the current mode of operation.

					• 1’b0: Device mode • 1’b1: Host mode



Note: Because an interrupt mask only masks an interrupt, software must clear an interrupt before unmasking it, to avoid servicing an old interrupt.

Figure 4.3 USB OTG Controller Interrupt Hierarachy

Core Interrupt Mask Register (GINTMSK) 0xB0800018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WkUpIntMsk	SessReqIntMsk	DisconnIntMsk	ConIDStsChngMsk		PTxFEmpMsk	HChIntMsk	PrtIntMsk		FetSuspMsk	incomplPMsk/ incompISOOUTMsk	incompISOINMsk	OEPIntMsk	IEPIntMsk	EPMisMsk	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EOPFMsk	ISOOutDropMsk	EnumDoneMsk	USBRstMsk	USBSuspMsk	ErlySuspMsk			GOUTNakEffMsk	GINNakEffMsk		RxFLvlMsk	SofMsk	OTGIntMsk	ModeMisMsk	

This register works with the Core Interrupt register to interrupt the application. When an interrupt bit is masked, the interrupt associated with that bit is not generated. However, the Core Interrupt (GINTSTS) register bit corresponding to that interrupt is still set.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	Mode	RW	Reset	Description
31	WkUpIntMsk	Host Device	R/W	0x0	Resume/Remote Wakeup Detected Interrupt Mask
30	SessReqIntMsk	Host Device	R/W	0x0	Session Request/New Session Detected Interrupt Mask()
29	DisconnIntMsk	Host Device	R/W	0x0	Disconnect Detected Interrupt Mask
28	ConIDStsChngMsk	Host Device	R/W	0x0	Connector ID Status Change Mask
27	-	-	-	0x0	Reserved
26	PTxFEmpMsk	Host	R/W	0x0	Periodic TxFIFO Empty Mask
25	HChIntMsk	Host	R/W	0x0	Host Channels Interrupt Mask
24	PrtIntMsk	Host	R/W	0x0	Host Port Interrupt Mask
23	-	-	-	0x0	Reserved
22	FetSuspMsk	Device	R/W	0x0	Data Fetch Suspended Mask
21	incomplPMsk	Host	R/W	0x0	Incomplete Periodic Transfer Mask
	incomplSOOUTMsk	Device			Incomplete Isochronous OUT Transfer Mask
20	incomplSOINMsk	Device	R/W	0x0	Incomplete Isochronous IN Transfer Mask
19	OEPIntMsk	Device	R/W	0x0	OUT Endpoints Interrupt Mask
18	IEPIntMsk	Device	R/W	0x0	IN Endpoints Interrupt Mask
17	EPMisMsk	Device	R/W	0x0	Endpoint Mismatch Interrupt Mask
16	-	-	-	-	Reserved
15	EOPFMsk	Device	R/W	0x0	End of Periodic Frame Interrupt Mask
14	ISOOutDropMsk	Device	R/W	0x0	Isochronous OUT Packet Dropped Interrupt Mask
13	EnumDoneMsk	Device	R/W	0x0	Enumeration Done Mask
12	USBRstMsk	Device	R/W	0x0	USB Reset Mask
11	USBSuspMsk	Device	R/W	0x0	USB Suspend Mask
10	ErlySuspMsk	Device	R/W	0x0	Early Suspend Mask
9-8	-	-	-	0x0	Reserved
7	GOUTNakEffMsk	Device	R/W	0x0	Global OUT NAK Effective Mask
6	GINNakEffMsk	Device	R/W	0x0	Global Non-periodic IN NAK Effective Mask
5	-	-	-	0x0	Reserved
4	RxFLvlMsk	Host Device	R/W	0x0	Receive FIFO Non-Empty Mask
3	SofMsk	Host Device	R/W	0x0	Start of (micro)Frame Mask ()
2	OTGIntMsk	Host Device	R/W	0x0	OTG Interrupt Mask
1	ModeMisMsk	Host Device	R/W	0x0	Mode Mismatch Interrupt Mask
0	-	-	-	0x0	Reserved

Receive Status Debug Read/Status Read and Pop Registers (GRXSTSR/GRXSTSP)

0xB080001C(Read)/0xB0800020(Pop)

A read to the Receive Status Debug Read register returns the contents of the top of theReceive FIFO. A read to the Receive Status Read and Pop register additionally pops the top data entry out of the RxFIFO.

The receive status contents must be interpreted differently in Host and Device modes. The core ignores the receive status pop/read when the receive FIFO is empty and returns a value of 32'h0000_0000. The application must only pop the Receive Status FIFO when the Receive FIFO Non-Empty bit of the Core Interrupt register (GINTSTS.RxFLvl) is asserted.

The following table shows the use of these registers in Host mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
											PktSts				DPID[1]
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DPID[0]	BCnt											ChNum			

Field	Name	Mode	RW	Reset	Description
31-21	-	-	-	0x0	Reserved
20-17	PktSts	Host	R	0x0	Packet Status Indicates the status of the received packet • 4'b0010: IN data packet received • 4'b0011: IN transfer completed (triggers an interrupt) • 4'b0101: Data toggle error (triggers an interrupt) • 4'b0111: Channel halted (triggers an interrupt) • Others: Reserved
16-15	DPID	Host	R	0x0	Data PID Indicates the Data PID of the received packet • 2'b00: DATA0 • 2'b10: DATA1 • 2'b01: DATA2 • 2'b11: MDATA
14-4	BCnt	Host	R	0x0	Byte Count Indicates the byte count of the received IN data packet.
3-0	ChNum	Host	R	0x0	Channel Number () Indicates the channel number to which the current received packet belongs.

The following table shows the use of these registers in Device mode.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							FN				PktSts				DPID[1]
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DPID[0]	BCnt											EPNum			

Field	Name	Mode	RW	Reset	Description
31-25	-	-	-	0x0	Reserved
24-21	FN	Device	R	0x0	Frame Number This is the least significant 4 bits of the (micro)frame number in which the packet is received on the USB. This field is supported only when isochronous OUT endpoints are supported.
20-17	PktSts	Device	R	0x0	Packet Status () Indicates the status of the received packet • 4’b0001: Global OUT NAK (triggers an interrupt) • 4’b0010: OUT data packet received • 4’b0011: OUT transfer completed (triggers an interrupt) • 4’b0100: SETUP transaction completed (triggers an interrupt) • 4’b0110: SETUP data packet received • Others: Reserved
16-15	DPID	Device	R	0x0	Data PID Indicates the Data PID of the received OUT data packet • 2’b00: DATA0 • 2’b10: DATA1 • 2’b01: DATA2 • 2’b11: MDATA
14-4	BCnt	Device	R	0x0	Byte Count Indicates the byte count of the received IN data packet.

Receive FIFO Size Register (GRXFSIZ) **0xB0800024**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RxFDep															

The application can program the RAM size and the memory start address for the Nonperiodic TxFIFO.

Field	Name	Mode	RW	Reset	Description
31-16	-	-	-	0x0	Reserved
15-0	RxFDep	Host Device	R/W, R	0x0	RxFIFO Depth This value is in terms of 32-bit words. • Minimum value is 16 • Maximum value is 4160 The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth (parameter OTG_RX_DFIFO_DEPTH) during coreConsultant configuration. If Enable Dynamic FIFO Sizing? was deselected in coreConsultant (parameter OTG_DFIFO_DYNAMIC = 0), these flops are optimized, and reads return the power-on value. If Enable Dynamic FIFO Sizing? was selected in coreConsultant (parameter OTG_DFIFO_DYNAMIC = 1), you can write a new value in this field. You can write a new value in this field. Programmed values must not exceed the power-on value set in coreConsultant.

Non-Periodic Transmit FIFO Size Register (GNPTXFSIZ) 0xB0800028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NPTxFDep/ INEPTxF0Dep															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NPTxFStAddr/ INEPTxF0StAddr															

The application can program the RAM size that must be allocated to the RxFIFO.

Field	Name	Mode	RW	Reset	Description
31-16	NPTxFDep	Host	R/W, R	0x0	Non-periodic TxFIFO Depth This value is in terms of 32-bit words. Minimum value is 16. Maximum value is 4160.
	INEPTxF0Dep	Device			IN Endpoint TxFIFO 0 Depth This value is in terms of 32-bit words. Minimum value is 16. Maximum value is 4160.
15-0	NPTxFStAddr	Host	R/W, R	0x0	Non-periodic Transmit RAM Start Address This field contains the memory start address for Non-periodic Transmit FIFO RAM.
	INEPTxF0StAddr	Device			IN Endpoint FIFO0 Transmit RAM Start Address This field contains the memory start address for IN Endpoint Transmit FIFO# 0.

Non-Periodic Transmit FIFO/Queue Status Register (GNPTXSTS) 0xB080002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	NPTxQTop							NPTxQSpcAvail							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NPTxFSpcAvail															

In Device mode, this register is invalid.

This read-only register contains the free space information for the Non-periodic TxFIFO and the Non-periodic Transmit Request Queue.

Field	Name	Mode	RW	Reset	Description
31	-	-	-	0x0	Reserved
30-24	NPTxQTop	Host	R	0x0	Top of the Non-periodic Transmit Request Queue Entry in the Non-periodic Tx Request Queue that is currently being processed by the MAC. • Bits [30:27]: Channel/endpoint number • Bits [26:25]: - 2’b00: IN/OUT token - 2’b01: Zero-length transmit packet (device IN/host OUT) - 2’b10: PING/CSPLIT token - 2’b11: Channel halt command • Bit [24]: Terminate (last entry for selected channel/endpoint)
23-16	NPTxQSpcAvail	Host	R	0x0	Non-periodic Transmit Request Queue Space Available Indicates the amount of free space available in the Non-periodic Transmit Request Queue. This queue holds both IN and OUT requests in Host mode. • 8’h0: Non-periodic Transmit Request Queue is full • 8’h1: 1 location available • 8’h2: 2 locations available • n: n locations available (0≤n≤8) • Others: Reserved
15-0	NPTxFSpcAvail	Host	R	0x0	Non-periodic TxFIFO Space Avail Indicates the amount of free space available in the Non-periodic TxFIFO. Values are in terms of 32-bit words. • 16’h0: Non-periodic TxFIFO is full • 16’h1: 1 word available • 16’h2: 2 words available • 16’hn: n words available (where 0≤n≤ 32,768) • 16’h8000: 32,768 words available • Others: Reserved

User ID Register (GUID) 0xB080003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UserID[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UserID[15:0]															

This is a read/write register containing the User ID.

This register can be used in the following ways:

- To store the version or revision of your system
- To store hardware configurations
- As a scratch register

Field	Name	Mode	RW	Reset	Description
31-0	UserID	Host Device	R	0x1234567	User ID Application-programmable ID field

User HW Config1 Register (GHWCFG1) 0xB0800044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
epdir [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
epdir [15:0]															

Field	Name	Mode	RW	Reset	Description
31-0	epdir	Host Device	R	0x0	Endpoint Direction Two bits per endpoint represent the direction. • 2’b00: BIDIR (IN and OUT) endpoint • 2’b01: IN endpoint • 2’b10: OUT endpoint • 2’b11: Reserved Bits [31:30]: Endpoint 15 direction Bits [29:28]: Endpoint 14 direction ... Bits [3:2]: Endpoint 1 direction Bits[1:0]: Endpoint 0 direction (always BIDIR)

User HW Config2 Register (GHWCFG2) 0xB0800048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TknQDepth					PTxQDepth		NPTxQDepth				DynFifo Sizing	PerioS upport	NumHstChnl[3:2]	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NumHstChnl[1:0]		NumDevEps				FSPhyType		HSPhyType		SingPnt	OtgArch				

Field	Name	RW	Reset	Description
31	-	-	0x0	Reserved
30-26	TknQDepth	R	0x8	Device Mode IN Token Sequence Learning Queue Depth: Range: 0‑30
25-24	PTxQDepth	R	0x2	Host Mode Periodic Request Queue Depth • 2’b00: 2 • 2’b01: 4 • 2’b10: 8 • Others: Reserved
23-22	NPTxQDepth	R	0x2	Non-periodic Request Queue Depth • 2’b00: 2 • 2’b01: 4 • 2’b10: 8 • Others: Reserved
21-20	-	-	0x0	Reserved
19	DynFifoSizing	R	0x1	Dynamic FIFO Sizing Enabled • 1’b0: No

				• 1’b1: Yes
18	PerioSupport	R	0x1	Periodic OUT Channels Supported in Host Mode • 1’b0: No • 1’b1: Yes
17-14	NumHstChnl	R	0xF	Number of Host Channels Indicates the number of host channels supported by the core in Host mode. The range of this field is 0-15: 0 specifies 1 channel, 15 specifies 16 channels.
13-10	NumDevEps	R	0xF	Number of Device Endpoints Indicates the number of device endpoints supported by the core in Device mode in addition to control endpoint 0. The range of this field is 1-15.
9-8	FSPhyType	R	0x0	Full-Speed PHY Interface Type • 2’b00: Full-speed interface not supported • 2’b01: Dedicated full-speed interface • 2’b10: FS pins shared with UTMI+ pins • 2’b11: FS pins shared with ULPI pins
7-6	HSPhyType	R	0x3	High-Speed PHY Interface Type • 2’b00: High-Speed interface not supported • 2’b01: UTMI+ • 2’b10: ULPI • 2’b11: UTMI+ and ULPI
5	SingPnt	R	0x1	Point-to-Point • 1’b0: Multi-point application • 1’b1: Single-point application
4-3	OtgArch	R	0x2	Architecture • 2’b00: Slave-Only • 2’b01: External DMA • 2’b10: Internal DMA
2-0	OtgMode	R	0x0	Mode of Operation • 3’b000: HNP- and SRP-Capable OTG (Host & Device) • 3’b001: SRP-Capable OTG (Host & Device) • 3’b010: Non-HNP and Non-SRP Capable OTG (Host & Device) • 3’b011: SRP-Capable Device • 3’b100: Non-OTG Device • 3’b101: SRP-Capable Host • 3’b110: Non-OTG Host • Others: Reserved

User HW Config3 Register (GHWCFG3) 0xB080004C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DfifoDepth															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				RstType	OptFeature			OtgEn	PktSizeWidth			XferSizeWidth			

Field	Name	RW	Reset	Description
31-16	DfifoDepth	R	0x8000	DFIFO Depth This value is in terms of 32-bit words. • Minimum value is 32 • Maximum value is 4160
15-12	-	-	0x0	Reserved
11	RstType	R	0x0	Reset Style for Clocked always Blocks in RTL • 1’b0: Asynchronous reset is used in the core • 1’b1: Synchronous reset is used in the core
10	OptFeature	R	0x0	Optional Features Removed Indicates whether the User ID register, GPIO interface ports, and SOF toggle and counter ports were removed for gate count optimization by enabling Remove Optional Features during coreConsultant configuration. • 1’b0: No • 1’b1: Yes
9-8	-	-	0x0	Reserved
7	OtgEn	R	0x1	OTG Function Enabled The application uses this bit to indicate core’s OTG capabilities. • 1’b0: Not OTG capable • 1’b1: OTG Capable
6-4	PktSizeWidth	R	0x6	Width of Packet Size Counters • 3’b000: 4 bits • 3’b001: 5 bits • 3’b010: 6 bits • 3’b011: 7 bits • 3’b100: 8 bits • 3’b101: 9 bits • 3’b110: 10 bits • Others: Reserved
3-0	XferSizeWidth	R	0x9	Width of Transfer Size Counters • 4’b0000: 11 bits • 4’b0001: 12 bits ... • 4’b1000: 19 bits • Others: Reserved

User HW Config4 Register (GHWCFG4) 0xB0800050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		INEps				DedFifoMode	SessEndFltr	BValidFltr	AValidFltr	VBusValidFltr	IddgFltr	NumCtlEps			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PhyDataWidth										AhbFreq	EnablePwrOpt	NumDevPerioEps			

Field	Name	RW	Reset	Description
31-30	-	-	0x0	Reserved
29-26	INEps	R	0x7	Endpoints Range 0 -15 • 0 : 1 IN Endpoint • 1 : 2 IN Endpoints • .... 15 : 16 IN Endpoints
25	DedFifoMode	R	0x1	Enable Dedicated Transmit FIFO for device IN Endpoints • 1’b0 : Dedicated Transmit FIFO Operation not enabled. • 1’b1 : Dedicated Transmit FIFO Operation enabled.
24	SessEndFltr	R	0x1	“session_end” Filter Enabled () • 1’b0: No filter • 1’b1: Filter
23	BValidFltr	R	0x1	“b_valid” Filter Enabled • 1’b0: No filter • 1’b1: Filter
22	AValidFltr	R	0x1	“a_valid” Filter Enabled • 1’b0: No filter • 1’b1: Filter
21	VBusValidFltr	R	0x1	“vbus_valid” Filter Enabled • 1’b0: No filter • 1’b1: Filter
20	IddgFltr	R	0x1	“iddig” Filter Enable • 1’b0: No filter • 1’b1: Filter
19-16	NumCtlEps	R	0x0	Number of Device Mode Control Endpoints in Addition to Endpoint 0 Range: 0-15
15-14	PhyDataWidth	R	0x2	UTMI+ PHY/ULPI-to-Internal UTMI+ Wrapper Data Width When a ULPI PHY is used, an internal wrapper converts ULPI to UTMI+ . • 2’b00: 8 bits • 2’b01: 16 bits • 2’b10: 8/16 bits, software selectable • Others: Reserved
13-6	-	-	0x0	Reserved
5	AhbFreq	R	0x1	Minimum AHB Frequency Less Than 60 MHz • 1’b0: No • 1’b1: Yes
4	EnablePwrOpt	R	0x1	Enable Power Optimization • 1’b0: No • 1’b1: Yes
3-0	NumDevPerioEps	R	0x0	Number of Device Mode Periodic IN Endpoints Range: 0‑15

Host Periodic Transmit FIFO Size Register (HPTXFSIZ) 0xB0800100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTxFSize															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTxFStAddr															

This register holds the size and the memory start address of the Periodic TxFIFO.

Field	Name	RW	Reset	Description
31-0	PTxFSize	R/W, R	0	Host Periodic TxFIFO Depth This value is in terms of 32-bit words. • Minimum value is 16 • Maximum value is 4160
15-0	PTxFStAddr	R/W, R	0	Host Periodic TxFIFO Start Address In shared FIFO operation : • OTG_RX_DFIFO_DEPTH + OTG_TX_HNPERIO_DFIFO_DEPTH In dedicated FIFO mode : • OTG_RX_DFIFO_DEPTH + OTG_TX_HNPERIO_DFIFO_DEPTH

Device IN Endpoint Transmit Fifo Size Register: (DIEPTXFn) 0xB0800104+(FIFO_number-1)*0x4

FIFO_number:1≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INEPnTxFDep															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INEPnTxFStAddr															

This register holds the size and the memory start address of IN endpoint TxFIFOs implemented in Device mode. Each FIFO holds the data for one IN endpoint. This register is repeated for instantiated IN endpoint FIFOs 1 to 15. For IN endpoint FIFO 0 use GNPTXFSIZ register for programming the size and memory start address.

Field	Name	RW	Reset	Description
31-0	INEPnTxFDep	R/W, R	0	IN Endpoint TxFIFO Depth This value is in terms of 32-bit words. Minimum value is 16. Maximum value is 4160.
15-0	INEPnTxFStAddr	R/W, R	0	IN Endpoint FIFOn Transmit RAM Start Address This field contains the memory start address for IN endpoint Transmit FIFOn (0<n< = 15). OTG_RX_DFIFO_DEPTH + SUM 0 to n – 1 (OTG_DINEP_TXFIFO_DEPTH_n) For example start address of IN endpoint FIFO 1 is OTG_RX_DFIFO_DEPTH + OTG_DINEP_TXFIFO_DEPTH_0 The start address of IN endpoint FIFO 2 is OTG_RX_DFIFO_DEPTH + OTG_DINEP_TXFIFO_DEPTH_0 + OTG_DINEP_TXFIFO_DEPTH_1

Host Configuration Register (HCFG) 0xB0800400

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													FSLSSupp	FSLSPclkSel	

This register configures the core after power-on. Do not make changes to this register after initializing the host.

Field	Name	RW	Reset	Description
31-3	-	-	0	Reserved
2	FSLSSupp	R/W	0	FS- and LS-Only Support The application uses this bit to control the core's enumeration speed. Using this bit, the application can make the core enumerate as a FS host, even if the connected device supports HS traffic. Do not make changes to this field after initial programming. • 1'b0: HS/FS/LS, based on the maximum speed supported by the connected device • 1'b1: FS/LS-only, even if the connected device can support HS
1-0	FSLSPclkSel	R/W	0	FS/LS PHY Clock Select When the core is in FS Host mode • 2'b00: PHY clock is running at 30/60 MHz • 2'b01: PHY clock is running at 48 MHz • Others: Reserved When the core is in LS Host mode • 2'b00: PHY clock is running at 30/60 MHz. When the UTMI+/ULPI PHY Low Power mode is not selected, use 30/60 MHz. • 2'b01: PHY clock is running at 48 MHz. When the UTMI+ PHY Low Power mode is selected, use 48MHz if the PHY supplies a 48 MHz clock during LS mode. • 2'b10: PHY clock is running at 6 MHz. In USB 1.1 FS mode, use 6 MHz when the UTMI+ PHY Low Power mode is selected and the PHY supplies a 6 MHz clock during LS mode. If you select a 6 MHz clock during LS mode, you must do a soft reset. • 2'b11: Reserved

Host Frame Interval Register (HFIR) 0xB0800404

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FrInt															

This register stores the frame interval information for the current speed to which the otg core has enumerated.

Field	Name	RW	Reset	Description
31-16	-	-	0	Reserved
15-0	FrInt	R/W	60000 (decimal)	Frame Interval The value that the application programs to this field specifies the interval between two consecutive SOFs (FS) or micro-SOFs (HS) or Keep-Alive tokens (HS). This field contains the number of PHY clocks that constitute the required frame interval. The default value set in this field for a FS operation when the PHY clock frequency is 60 MHz. The application can write a value to this register only after the Port Enable bit of the Host Port Control and Status register (HPRT.PrtEnaPort) has been set. If no value is programmed, the core calculates the value based on the PHY clock specified in the FS/LS PHY Clock Select field of the Host Configuration register (HCFG.FSLSPclkSel). Do not change the value of this field after the initial configuration. • 125 μs * (PHY clock frequency for HS) • 1 ms * (PHY clock frequency for FS/LS)

Host Frame Number/Frame Time Remaining Register (HFNUM) 0xB0800408

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FrRem															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FrNum															

This register indicates the current frame number. It also indicates the time remaining (interms of the number of PHY clocks) in the current (micro)frame.

Field	Name	RW	Reset	Description
31-16	FrRem	R	0x0	Frame Time Remaining Indicates the amount of time remaining in the current microframe (HS) or frame (FS/LS), in terms of PHY clocks. This field decrements on each PHY clock. When it reaches zero, this field is reloaded with the value in the Frame Interval register and a new SOF is transmitted on the USB.
15-0	FrNum	R/W	0x3FFF	Frame Number This field increments when a new SOF is transmitted on the USB, and is reset to 0 when it reaches 16’h3FFF.

Host Periodic Transmit FIFO/Queue Status Register (HPTXSTS) 0xB0800410

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PTxQTop								PTxQSpcAvail							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTxFSpcAvail															

This read-only register contains the free space information for the Periodic TxFIFO and the Periodic Transmit Request Queue.

Field	Name	RW	Reset	Description
31-24	PTxQTop	R	0x0	Top of the Periodic Transmit Request Queue This indicates the entry in the Periodic Tx Request Queue that is currently being processes by the MAC. This register is used for debugging. • Bit [31]: Odd/Even (micro)frame - 1’b0: send in even (micro)frame - 1’b1: send in odd (micro)frame • Bits [30:27]: Channel/endpoint number • Bits [26:25]: Type - 2’b00: IN/OUT - 2’b01: Zero-length packet - 2’b10: CSPLIT - 2’b11: Disable channel command • Bit [24]: Terminate (last entry for the selected channel/endpoint)
23-16	PTxQSpcAvail	R	0x0	Periodic Transmit Request Queue Space Available Indicates the number of free locations available to be written in the Periodic Transmit Request Queue. This queue holds both IN and OUT requests. • 8’h0: Periodic Transmit Request Queue is full • 8’h1: 1 location available • 8’h2: 2 locations available • n: n locations available (0≤n≤8) • Others: Reserved
15-0	PTxFSpcAvail	R/W	0x0	Periodic Transmit Data FIFO Space Available Indicates the number of free locations available to be written to in the Periodic TxFIFO. Values are in terms of 32-bit words • 16’h0: Periodic TxFIFO is full • 16’h1: 1 word available • 16’h2: 2 words available • 16’hn: n words available (where 0≤n≤ 32,768) • 16’h8000: 32,768 words available • Others: Reserved

Host All Channels Interrupt Register (HAINT) 0xB0800414

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HAINT															

When a significant event occurs on a channel, the Host All Channels Interrupt register interrupts the application using the Host Channels Interrupt bit of the Core Interrupt register (GINTSTS.HChInt). This is shown in Figure 4.3. There is one interrupt bit per channel, up to a maximum of 16 bits. Bits in this register are set and cleared when the application sets and clears bits in the corresponding Host Channel-n Interrupt register.

Field	Name	RW	Reset	Description
31-16	-	-	0x0	Reserved
15-0	HAINT	R	0x0	Channel Interrupts One bit per channel: Bit 0 for Channel 0, bit 15 for Channel 15

Host All Channels Interrupt Mask Register (HAINTMSK) 0xB0800418

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HAINTMsk															

The Host All Channel Interrupt Mask register works with the Host All Channel Interrupt register to interrupt the application when an event occurs on a channel. There is one interrupt mask bit per channel, up to a maximum of 16 bits.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-16	-	-	0x0	Reserved
15-0	HAINTMsk	R/W	0x0	Channel Interrupts Mask One bit per channel: Bit 0 for Channel 0, bit 15 for Channel 15

Host Port Control and Status Register (HPRT) 0xB0800440

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
													PrtSpd		PrtTstCtl[3]
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PrtTstCtl[2:0]			PrtPwr	PrtLnSts			PrtRst	PrtSusp	PrtRes	PrtOvrCurrChng	PrtOvrCurrAct	PrtEnChng	PrtEna	PrtConnDet	PrtConnSts

This register is available only in Host mode. Currently, the OTG Host supports only one port.

A single register holds USB port-related information such as USB reset, enable, suspend, resume, connect status, and test mode for each port. It is shown in

Figure 4.3. The PrtOvrCurrChng and PrtEnChng in this register can trigger an interrupt to the application through the Host Port Interrupt bit of Core Interrupt register (GINTSTS.PrtInt). On a Port Interrupt, the application must read this register and clear the bit that caused the interrupt.

Field	Name	RW	Reset	Description
31-19	-	-	0x0	Reserved
18-17	PrtSpd	R	0x0	Port Speed () Indicates the speed of the device attached to this port. • 2’b00: High speed • 2’b01: Full speed • 2’b10: Low speed • 2’b11: Reserved
16-13	PrtTstCtl	R/W	0x0	Port Test Control The application writes a nonzero value to this field to put the port into a Test mode, and the corresponding pattern is signaled on the port. • 4’b0000: Test mode disabled • 4’b0001: Test_J mode • 4’b0010: Test_K mode • 4’b0011: Test_SE0_NAK mode • 4’b0100: Test_Packet mode • 4’b0101: Test_Force_Enable • Others: Reserved
12	PrtPwr	R/W	0x0	Port Power The application uses this field to control power to this port, and the core clears this bit on an overcurrent condition. • 1’b0: Power off • 1’b1: Power on
11-10	PrtLnSts	R	0x0	Port Line Status Indicates the current logic level USB data lines • Bit [10]: Logic level of D- • Bit [11]: Logic level of D +
9	-	-	0x0	Reserved
8	PrtRst	R/W	0x0	Port Reset When the application sets this bit, a reset sequence is started on this port. The application must time the reset period and clear this bit after the reset sequence is complete. • 1’b0: Port not in reset • 1’b1: Port in reset The application must leave this bit set for at least a minimum duration mentioned below to start a reset on the port. The application can leave it set for another 10 ms in addition to the required minimum duration, before clearing the bit, even though there is no maximum limit set by the USB standard. • High speed: 50 ms • Full speed/Low speed: 10 ms
7	PrtSusp	R/W	0x0	Port Suspend The application sets this bit to put this port in Suspend mode. The core only stops sending SOFs when this is set. To stop the PHY clock, the application must set the Port Clock Stop bit, which asserts the suspend input pin of the PHY.

				The read value of this bit reflects the current suspend status of the port. This bit is cleared by the core after a remote wakeup signal is detected or the application sets the Port Reset bit or Port Resume bit in this register or the Resume/Remote Wakeup Detected Interrupt bit or Disconnect Detected Interrupt bit in the Core Interrupt register (GINTSTS.WkUpInt or GINTSTS.DisconnInt, respectively). • 1’b0: Port not in Suspend mode • 1’b1: Port in Suspend mode
6	PrtRes	R/W	0x0	Port Resume The application sets this bit to drive resume signaling on the port. The core continues to drive the resume signal until the application clears this bit. If the core detects a USB remote wakeup sequence, as indicated by the Port Resume/Remote Wakeup Detected Interrupt bit of the Core Interrupt register (GINTSTS.WkUpInt), the core starts driving resume signaling without application intervention and clears this bit when it detects a disconnect condition. The read value of this bit indicates whether the core is currently driving resume signaling. • 1’b0: No resume driven • 1’b1: Resume driven
5	PrtOvrCurrChng	R/W	0x0	Port Overcurrent Change The core sets this bit when the status of the Port Overcurrent Active bit (bit 4) in this register changes. The application must write a 1 to the bit to clear the interrupt
4	PrtOvrCurrAct	R	0x0	Port Overcurrent Active Indicates the overcurrent condition of the port. • 1’b0: No overcurrent condition • 1’b1: Overcurrent condition
3	PrtEnChng	R/W	0x0	Port Enable/Disable Change The core sets this bit when the status of the Port Enable bit [2] of this register changes. The application must write a 1 to the bit to clear the interrupt
2	PrtEna	R/W	0x0	Port Enable A port is enabled only by the core after a reset sequence, and is disabled by an overcurrent condition, a disconnect condition, or by the application clearing this bit. The application cannot set this bit by a register write. It can only clear it to disable the port. This bit does not trigger any interrupt to the application. • 1’b0: Port disabled • 1’b1: Port enabled
1	PrtConnDet	R/W	0x0	Port Connect Detected The core sets this bit when a device connection is detected to trigger an interrupt to the application using the Host Port Interrupt bit of the Core Interrupt register (GINTSTS.PrtInt). The application must write a 1 to this bit to clear the interrupt.
0	PrtConnSts	R	0x0	Port Connect Status • 0: No device is attached to the port. • 1: A device is attached to the port.

Host Channel-n Characteristics Register (HCCHARn) **0xB0800500+(Channel_number*0x20)**
Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ChEna	ChDis	OddFrm	DevAddr							MC/ EC		EPType		LSpdDev	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPDir	EPNum				MPS										

Field	Name	RW	Reset	Description
31	ChEna	R/W	0x0	Channel Enable This field is set by the application and cleared by the OTG host. • 1’b0: Channel disabled • 1’b1: Channel enabled
30	ChDis	R/W	0x0	Channel Disable The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer for that channel is complete. The application must wait for the Channel Disabled interrupt before treating the channel as disabled.
29	OddFrm	R/W	0x0	Odd Frame This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. • 1’b0: Even (micro)frame • 1’b1: Odd (micro)frame
28-22	DevAddr	R/W	0x0	Device Address This field selects the specific device serving as the data source or sink.
21-20	MC/ EC	R/W	0x0	Multi Count / Error Count When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltEna) is reset (1’b0), this field indicates to the host the number of transactions that must be executed per microframe for this endpoint. • 2’b00: Reserved This field yields undefined results. • 2’b01: 1 transaction • 2’b10: 2 transactions to be issued for this endpoint per microframe • 2’b11: 3 transactions to be issued for this endpoint per microframe When HCSPLTn.SpltEna is set (1’b1), this field indicates the number of immediate retries to be performed for a periodic split transactions on transaction errors. This field must be set to at least 2’b01.
19-18	EPType	R/W	0x0	Endpoint Type Indicates the transfer type selected. • 2’b00: Control • 2’b01: Isochronous • 2’b10: Bulk • 2’b11: Interrupt
17	LSpdDev	R/W	0x0	Low-Speed Device This field is set by the application to indicate that this channel is communicating to a low-speed device.
16	-	-	0x0	Reserved
15	EPDir	R/W	0x0	Endpoint Direction Indicates whether the transaction is IN or OUT. • 1’b0: OUT • 1’b1: IN
14-11	EPNum	R/W	0x0	Endpoint Number Indicates the endpoint number on the device serving as the data source or sink.
10-0	MPS	R/W	0x0	Maximum Packet Size Indicates the maximum packet size of the associated endpoint.

Host Channel-n Split Control Register (HCSPLTn)

0xB0800504+(Channel_number*0x20)
Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SpltEna															CompSplt
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
XactPos		HubAddr							PrtAddr						

Field	Name	RW	Reset	Description
31	SpltEna	R/W	0x0	Split Enable The application sets this field to indicate that this channel is enabled to perform split transactions.
30-17	-	-	0x0	Reserved
16	CompSplt	R/W	0x0	Do Complete Split The application sets this field to request the OTG host to perform a complete split transaction.
15-14	XactPos	R/W	0x0	Transaction Position This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction. • 2’b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes). • 2’b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes). • 2’b00: Mid. This is the middle payload of this transaction (which is larger than 188 bytes). • 2’b01: End. This is the last payload of this transaction (which is larger than 188 bytes).
13-7	HubAddr	R/W	0x0	Hub Address This field holds the device address of the transaction translator’s hub.
6-0	PrtAddr	R/W	0x0	Port Address This field is the port number of the recipient transaction translator.

Host Channel-n Interrupt Register (HCINTn) **0xB0800508+(Channel_number*0x20)**
Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					DataTglErr	FrmOvrun	BblErr	XactErr	NYET	ACK	NAK	STALL	AHBErr	ChHltd	XferCompl

This register indicates the status of a channel with respect to USB- and AHB-related events. It is shown in

Figure 4.3. The application must read this register when the Host Channels Interrupt bit of the Core Interrupt register (GINTSTS.HChInt) is set. Before the application can read this register, it must first read the Host All Channels Interrupt (HAINT) register to get the exact channel number for the Host Channel-n Interrupt register. The application must clear the appropriate bit in this register to clear the corresponding bits in the HAINT and GINTSTS registers.

Field	Name	RW	Reset	Description
31-11	-	-	0x0	Reserved
10	DataTglErr	R/W	0x0	Data Toggle Error
9	FrmOvrun	R/W	0x0	Frame Overrun
8	BblErr	R/W	0x0	Babble Error
7	XactErr	R/W	0x0	Transaction Error Indicates one of the following errors occurred on the USB. • CRC check failure • Timeout • Bit stuff error • False EOP
6	NYET	R/W	0x0	NYET Response Received Interrupt
5	ACK	R/W	0x0	ACK Response Received/Transmitted Interrupt
4	NAK	R/W	0x0	NAK Response Received Interrupt
3	STALL	R/W	0x0	STALL Response Received Interrupt
2	AHBErr	R/W	0x0	AHB Error This is generated only in Internal DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel’s DMA address register to get the error address.
1	ChHltd	R/W	0x0	Channel Halted Indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application.
0	XferCompl	R/W	0x0	Transfer Completed Transfer completed normally without any errors.

Host Channel-n Interrupt Mask Register (HCINTMSKn) 0xB080050C+(Channel_number*0x20)

Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					DataTglErrMsk	FrmOvrunMsk	BblErrMsk	XactErrMsk	NyetMsk	AckMsk	NakMsk	StallMsk	AHBErrMsk	ChHltdMsk	XferComplMsk

This register reflects the mask for each channel status described in the previous section.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-11	-	-	0x0	Reserved
10	DataTglErrMsk	R/W	0x0	Data Toggle Error Mask
9	FrmOvrunMsk	R/W	0x0	Frame Overrun Mask
8	BblErrMsk	R/W	0x0	Babble Error Mask
7	XactErrMsk	R/W	0x0	Transaction Error Mask
6	NyetMsk	R/W	0x0	NYET Response Received Interrupt Mask
5	AckMsk	R/W	0x0	ACK Response Received/Transmitted Interrupt Mask
4	NakMsk	R/W	0x0	NAK Response Received Interrupt Mask
3	StallMsk	R/W	0x0	STALL Response Received Interrupt Mask
2	AHBErrMsk	R/W	0x0	AHB Error Mask
1	ChHltdMsk	R/W	0x0	Channel Halted Mask
0	XferComplMsk	R/W	0x0	Transfer Completed Mask

Host Channel-n Transfer Size Register (HCTSIZn) 0xB0800510+(Channel_number*0x20)

Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DoPng	Pid		PktCnt										XferSize[18:16]		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
XferSize[15:0]															

Field	Name	RW	Reset	Description
31	DoPng	R/W	0x0	Do Ping () Setting this field to 1 directs the host to do PING protocol.
30-29	Pid	R/W	0x0	PID The application programs this field with the type of PID to use for the initial transaction. The host maintains this field for the rest of the transfer. • 2'b00: DATA0 • 2'b01: DATA2 • 2'b10: DATA1 • 2'b11: MDATA (non-control)/SETUP (control)
28-19	PktCnt	R/W	0x0	Packet Count This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decreme ts this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion.
18-0	XferSize	R/W	0x0	Transfer Size For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved for the transfer. The application is expected to program this field as an integer multiple of the maximum packet size for IN transactions (periodic and non-periodic).

Host Channel-n DMA Address Register (HCDMAn) 0xB0800514+(Channel_number*0x20)

Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAAddr [31:0]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAAddr [15:0]															

This register is used by the OTG host in the internal DMA mode to maintain the current buffer pointer for IN/OUT transactions. The starting DMA address must be DWORD-aligned.

Field	Name	RW	Reset	Description
31-0	DMAAddr	R/W	0x0	DMA Address This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transaction.

Device Configuration Register (DCFG) 0xB0800800

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
									EPMisCnt						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			PerFrInt		DevAddr								NZSts OUTH Shk	DevSpd	

This register configures the core in Device mode after power-on or after certain control commands or enumeration. Do not make changes to this register after initial programming.

Field	Name	RW	Reset	Description
31-23	-	-	0x0	Reserved
22-18	EPMisCnt	R/W	0x0	IN Endpoint Mismatch Count This field is valid only in shared FIFO operation. The application programs this filed with a count that determines when the core generates an Endpoint Mismatch interrupt (GINTSTS.EPMis). The core loads this value into an internal counter and decrements it. The counter is reloaded whenever there is a match or when the counter expires. The width of this counter depends on the depth of the Token Queue.
17-13	-	-	0x0	Reserved
12-11	PerFrInt	R/W	0x0	Periodic Frame Interval Indicates the time within a (micro)frame at which the application must be notified using the End Of Periodic Frame Interrupt. This can be used to determine if all the isochronous traffic for that (micro)frame is complete. • 2’b00: 80% of the (micro)frame interval • 2’b01: 85% • 2’b10: 90% • 2’b11: 95%
10-4	DevAddr	R/W	0x0	Device Address The application must program this field after every SetAddress control command.
3	-	-	0x0	Reserved
2	NZStsOUTHShk	R/W	0x0	Non-Zero-Length Status OUT Handshake The application can use this field to select the handshake the core sends on receiving a nonzero-length data packet during the OUT transaction of a control transfer’s Status stage. • 1’b1: Send a STALL handshake on a nonzero-length status OUT transaction and do not send the received OUT packet to the application. • 1’b0: Send the received OUT packet to the application (zerolength or nonzero-length) and send a handshake based on the NAK and STALL bits for the endpoint in the Device Endpoint Control register.
1-0	DevSpd	R/W	0x0	Device Speed Indicates the speed at which the application requires the core to enumerate, or the maximum speed the application can support. However, the actual bus speed is determined only after the chirp sequence is completed, and is based on the speed of the USB host to which the core is connected. See “Device Initialization” on page 210 for details. • 2’b00: High speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) • 2’b01: Full speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) • 2’b10: Low speed (USB 1.1 transceiver clock is 6 MHz). If you select 6 MHz LS mode, you must do a soft reset. • 2’b11: Full speed (USB 1.1 transceiver clock is 48 MHz)

Device Control Register (DCTL) 0xB0800804

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				PWROnPrgDone	CGOUTNak	SGOUTNak	CGNPInNak	SGNPInNak	TstCtl			GOUTNakSts	GNPINNakSts	SftDiscon	RmtWkUpSig

Field	Name	RW	Reset	Description
31-12	-	-	0x0	Reserved
11	PWROnPrgDone	R/W	0x0	Power-On Programming Done The application uses this bit to indicate that register programming is completed after a wake-up from Power Down Mode.
10	CGOUTNak	W	0x0	Clear Global OUT NAK A write to this field clears the Global OUT NAK.
9	SGOUTNak	W	0x0	Set Global OUT NAK A write to this field sets the Global OUT NAK. The application uses this bit to send a NAK handshake on all OUT endpoints. The application must set the this bit only after making sure that the Global OUT NAK Effective bit in the Core Interrupt Register (GINTSTS.GOUTNakEff) is cleared.
8	CGNPInNak	W	0x0	Clear Global Non-periodic IN NAK A write to this field clears the Global Non-periodic IN NAK.
7	SGNPInNak	W	0x0	Set Global Non-periodic IN NAK A write to this field sets the Global Non-periodic IN NAK.The application uses this bit to send a NAK handshake on all nonperiodic IN endpoints. The application must set this bit only after making sure that the Global IN NAK Effective bit in the Core Interrupt Register (GINTSTS.GINNakEff) is cleared.
6-4	TstCtl	R/W	0x0	Test Control • 3’b000: Test mode disabled • 3’b001: Test_J mode • 3’b010: Test_K mode • 3’b011: Test_SE0_NAK mode • 3’b100: Test_Packet mode • 3’b101: Test_Force_Enable • Others: Reserved
3	GOUTNakSts	R	0x0	Global OUT NAK Status • 1’b0: A handshake is sent based on the FIFO Status and the NAK and STALL bit settings. • 1’b1: No data is written to the RxFIFO, irrespective of space availability. Sends a NAK handshake on all packets, except on SETUP transactions. All isochronous OUT packets are dropped.
2	GNPINNakSts	R	0x0	Global Non-periodic IN NAK Status • 1'b0: A handshake is sent out based on the data availability in the transmit FIFO. • 1'b1: A NAK handshake is sent out on all non-periodic IN endpoints, irrespective of the data availability in the transmit FIFO.
1	SftDiscon	R/W	0x0	Soft Disconnect The application uses this bit to signal the otg core to do a soft disconnect. As long as this bit is set, the host does not see that the device is connected, and the device does not receive signals on the USB. The core stays in the disconnected state until the application clears this bit. The minimum duration for which the core must keep this bit set is specified in table 15.3 • 1’b0: Normal operation. When this bit is cleared after a soft disconnect, the core drives the phy_opmode_o signal on the UTMI+ to 2’b00, which generates a device connect event to the USB host. When the device is reconnected, the USB host restarts device enumeration. • 1’b1: The core drives the phy_opmode_o signal on the UTMI+

				to 2’b01, which generates a device disconnect event to the USB host.
0	RmtWkUpSig	R/W	0x0	Remote Wakeup Signaling When the application sets this bit, the core initiates remote signaling to wake up the USB host. The application must set this bit to instruct the core to exit the Suspend state. As specified in the USB 2.0 specification, the application must clear this bit 1‑15 ms after setting it.

Table 4.3 lists the minimum duration under various conditions for which the SoftDisconnect (SftDiscon) bit must be set for the USB host to detect a device disconnect. To accommodate clock jitter, it is recommended that the application add some extra delay to the specified minimum duration.

Table 4.3 Minimum Duration for Soft Disconnect

Operating Speed	Device Status	Minimum Duration
High Speed	Suspended	1 ms + 2.5 us
High Speed	Idle	3 ms + 2.5 us
High Speed	Not Idle or Suspended (Performing transactions)	125 us
Full speed/Low speed	Suspended	1 ms + 2.5 us
Full speed/Low speed	Idle	2.5 us
Full speed/Low speed	Not Idle or Suspended (Performing transactions)	2.5 us

Device Status Register (DSTS) 0xB0800808

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
										SOFFN[13:8]					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SOFFN[7:0]												ErrticErr	EnumSpd		SuspSts

This register indicates the status of the core with respect to USB-related events. It must be read on interrupts from Device All Interrupts (DAINT) register.

Field	Name	RW	Reset	Description
31-22	-	-	0x0	Reserved
21-8	SOFFN	R	0x0	Frame or Microframe Number of the Received SOF When the core is operating at high speed, this field contains a microframe number. When the core is operating at full or low speed, this field contains a frame number.
7-4	-	-	0x0	Reserved
3	ErrticErr	R	0x0	Erratic Error The core sets this bit to report any erratic errors (phy_rxvalid_i/phy_rxvldh_i or phy_rxactive_i is asserted for at least 2 ms, due to PHY error) seen on the UTMI+ . Due to erratic errors, the OTG core goes into Suspended state and an interrupt is generated to the application with Early Suspend bit of the Core Interrupt register (GINTSTS.ErlySusp). If the early suspend is asserted due to an erratic error, the application can only perform a soft disconnect recover.
2-1	EnumSpd	R	0x1	Enumerated Speed Indicates the speed at which the OTG core has come up after speed detection through a chirp sequence. • 2’b00: High speed (PHY clock is running at 30 or 60 MHz) • 2’b01: Full speed (PHY clock is running at 30 or 60 MHz) • 2’b10: Low speed (PHY clock is running at 6 MHz) • 2’b11: Full speed (PHY clock is running at 48 MHz) Low speed is not supported for devices using a UTMI+ PHY.
0	SuspSts	R	0x0	Suspend Status In Device mode, this bit is set as long as a Suspend condition is detected on the USB. The core enters the Suspended state when there is no activity on the phy_line_state_i signal for an extended period of time. The core comes out of the suspend: • When there is any activity on the phy_line_state_i signal

				• When the application writes to the Remote Wakeup Signaling bit in the Device Control register (DCTL.RmtWkUpSig).

Device IN Endpoint Common Interrupt Mask Register (DIEPMSK) 0xB0800810

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							TxfifoUndrnMsk		INEPNakEffMsk	INTknEPMisMsk	INTknTXFEmpMsk	TimeOUTMsk	AHBErrMsk	EPDisbldMsk	XferComplMsk

This register works with each of the Device IN Endpoint Interrupt (DIEPINTn) registers for all endpoints to generate an interrupt per IN endpoint. The IN endpoint interrupt for a specific status in the DIEPINTn register can be masked by writing to the corresponding bit in this register. Status bits are masked by default.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-9	-	-	0x0	Reserved
8	TxfifoUndrnMsk	R/W	0x0	Fifo Underrun Mask
7	-	-	0x0	Reserved
6	INEPNakEffMsk	R/W	0x0	IN Endpoint NAK Effective Mask
5	INTknEPMisMsk	R/W	0x0	IN Token received with EP Mismatch Mask
4	INTknTXFEmpMsk	R/W	0x0	IN Token Received When TxFIFO Empty Mask
3	TimeOUTMsk	R/W	0x0	Timeout Condition Mask (Non-isochronous endpoints)
2	AHBErrMsk	R/W	0x0	AHB Error Mask
1	EPDisbldMsk	R/W	0x0	Endpoint Disabled Interrupt Mask
0	XferComplMsk	R/W	0x0	Transfer Completed Mask

Device OUT Endpoint Common Interrupt Mask Register (DOEPMSK) 0xB0800814

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							OutPktErrMsk		Back2BackSETup		OUTTknEPdisMsk	SetUPMsk	AHBErrMsk	EPDisbldMsk	XferComplMsk

This register works with each of the Device OUT Endpoint Interrupt (DOEPINTn) registers for all endpoints to generate an interrupt per OUT endpoint. The OUT endpoint interrupt for a specific status in the DOEPINTn register can be masked by writing into the corresponding bit in this register. Status bits are masked by default.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-9	-	-	0x0	Reserved
8	OutPktErrMsk	R/W	0x0	OUT Packet Error Mask
7	-	-	0x0	Reserved
6	Back2BackSETup	R/W	0x0	Back-to-Back SETUP Packets Received Mask Applies to control OUT endpoints only.
5	-	-	0x0	Reserved
4	OUTTknEPdisMsk	R/W	0x0	OUT Token Received when Endpoint Disabled Mask Applies to control OUT endpoints only.
3	SetUPMsk	R/W	0x0	SETUP Phase Done Mask Applies to control endpoints only.
2	AHBErrMsk	R/W	0x0	AHB Error Mask
1	EPDisbldMsk	R/W	0x0	Endpoint Disabled Interrupt Mask
0	XferComplMsk	R/W	0x0	Transfer Completed Interrupt Mask

Device All Endpoints Interrupt Register (DAINT) 0xB0800818

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OutEPInt															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
InEpInt															

When a significant event occurs on an endpoint, a Device All Endpoints Interrupt register interrupts the application using the Device OUT Endpoints Interrupt bit or Device IN Endpoints Interrupt bit of the Core Interrupt register (GINTSTS.OEPInt or GINTSTS.IEPInt, respectively). This is shown in Figure 4-2. There is one interrupt bit per endpoint, up to a maximum of 16 bits for OUT endpoints and 16 bits for IN endpoints. For a bidirectional endpoint, the corresponding IN and OUT interrupt bits are used. Bits in this register are set and cleared when the application sets and clears bits in the corresponding Device Endpointn Interrupt register (DIEPINTn/DOEPINTn).

Field	Name	RW	Reset	Description
31-16	OutEPInt	R	0x0	OUT Endpoint Interrupt Bits One bit per OUT endpoint: Bit 16 for OUT endpoint 0, bit 31 for OUT endpoint 15
15-0	InEpInt	R	0x0	IN Endpoint Interrupt Bits One bit per IN Endpoint: Bit 0 for IN endpoint 0, bit 15 for endpoint 15

Device All Endpoints Interrupt Mask Register (DAINTMSK) 0xB080081C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OutEPInt															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
InEpInt															

The Device Endpoint Interrupt Mask register works with the Device Endpoint Interrupt register to interrupt the application when an event occurs on a device endpoint. However, the Device All Endpoints Interrupt (DAINT) register bit corresponding to that interrupt is still set.

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-16	OutEPMsk	R	0x0	OUT Endpoint Interrupt Mask Bits One bit per OUT endpoint: Bit 16 for OUT endpoint 0, bit 31 for OUT endpoint 15
15-0	InEpMsk	R	0x0	IN Endpoint Interrupt Mask Bits One bit per IN Endpoint: Bit 0 for IN endpoint 0, bit 15 for IN endpoint 15

Device VBUS Discharge Time Register (DVBUSDIS) 0xB0800828

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DVBUSDis															

This register specifies the VBUS discharge time after VBUS pulsing during SRP.

Field	Name	RW	Reset	Description
31-16	-	-	0x0	Reserved
15-0	DVBUSDis	R/W	30 MHz: 0x0B8F 60 MHz: 0x17D7	Device VBUS Discharge Time Specifies the VBUS discharge time after VBUS pulsing during SRP. This value equals: VBUS discharge time in PHY clocks / 1, 024 The value you use depends whether the PHY is operating at 30 MHz (16-bit data width) or 60 MHz (8-bit data width). Depending on your VBUS load, this value can need adjustment.

Device VBUS Pulsing Time Register (DVBUSPULSE) 0xB080082C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				DVBUSPulse											

This register specifies the VBUS pulsing time during SRP.

Field	Name	RW	Reset	Description
31-12	-	-	0x0	Reserved
11-0	DVBUSPulse	R/W	30 MHz: 12'h2C6 60 MHz: 12'h5B8	Device VBUS Pulsing Time Specifies the VBUS pulsing time during SRP. This value equals: VBUS pulsing time in PHY clocks / 1, 024 The value you use depends whether the PHY is operating at 30 MHz (16-bit data width) or 60 MHz (8-bit data width).

Device Threshold Control Register (DTHRCTL) 0xB0800830

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				ArbPrkEn		RxThrLen									RxThrEn
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					TxThrLen									ISOThrEn	NonISOThrEn

Thresholding is not supported in Slave mode and so this register should not be programmed in Slave mode. For threshold support, the AHB needs to be run at 60MHz or higher.

Field	Name	RW	Reset	Description
31-28	-	-	0x0	Reserved
27	ArbPrkEn	R/W	0x1	Arbiter Parking Enable This bit controls internal DMA arbiter parking for IN endpoints. When thresholding is enabled and this bit is set to one, then the arbiter parks on the IN endpoint for which there is a token received on the USB. This is done to avoid getting into underrun conditions. By default the parking is enabled.
26	-	-	0x0	Reserved
25-17	RxThrLen	R/W	0x0	Receive Threshold Length This field specifies Receive thresholding size in DWORDS. This field also specifies the amount of data received on the USB before the core can start transmitting on the AHB. The threshold length has to be at least eight DWORDS. The recommended value for ThrLen is to be the same as the programmed AHB Burst Length (GAHBCFG.HBstLen).
16	RxThrEn	R/W	0x0	Receive Threshold Enable When this bit is set, the core enables thresholding in the receive direction.
15-11	-	-	0x0	Reserved
10-2	TxThrLen	R/W	0x0	Transmit Threshold Length This field specifies Transmit thresholding size in DWORDS. This field specifies the amount of data in bytes to be in the corresponding endpoint transmit FIFO, before the core can start transmit on the USB. The threshold length has to be at least eight DWORDS. This field controls both isochronous and non-isochronous IN endpoint thresholds. The recommended value for ThrLen is to be the same as the programmed AHB Burst Length (GAHBCFG.HBstLen).
1	ISOThrEn	R/W	0x0	ISO IN Endpoints Threshold Enable. When this bit is set, the core enables thresholding for isochronous IN endpoints.
0	NonISOThrEn	R/W	0x0	Non-ISO IN Endpoints Threshold Enable When this bit is set, the core enables thresholding for Non Isochronous IN endpoints.

Device IN Endpoint FIFO Empty Interrupt Mask Register: (DIEPEMPMSK) 0xB0800834

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
InEpTxfEmpMsk															

This register is used to control the IN endpoint FIFO empty interrupt generation (DIEPINTn.TxfEmp).

- Mask interrupt: 1'b0
- Unmask interrupt: 1'b1

Field	Name	RW	Reset	Description
31-16	-	-	0x0	Reserved
15-0	InEpTxfEmpMsk	R/W	0x0	IN EP Tx FIFO Empty Interrupt Mask Bits These bits acts as mask bits for DIEPINTn. TxFEmp interrupt One bit per IN Endpoint: Bit 0 for IN EP 0, bit 15 for IN EP 15

Device Control IN Endpoint 0 Control Register (DIEPCTL0) 0xB0800900

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPEna	EPDis			SNAK	CNAK	TxFNum				Stall		EPType		NAKSts	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBA ctEP														MPS	

This section describes the Control IN Endpoint 0 Control register. Nonzero control endpoints use registers for endpoints 1–15.

Field	Name	RW	Reset	Description
31	EPEna	R/W	0x0	Endpoint Enable Indicates that data is ready to be transmitted on the endpoint. The core clears this bit before setting any of the following interrupts on this endpoint: • Endpoint Disabled • Transfer Completed
30	EPDis	R/W	0x0	Endpoint Disable The application sets this bit to stop transmitting data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled Interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint.
29-28	-	-	0x0	Reserved
27	SNAK	W	0x0	Set NAK A write to this bit sets the NAK bit for the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also set this bit for an endpoint after a SETUP packet is received on that endpoint.
26	CNAK	R/W	0x0	Clear NAK A write to this bit clears the NAK bit for the endpoint.
25-22	TxFNum	R/W	0x0	TxFIFO Number This value is set to the FIFO number that is assigned to IN Endpoint 0. ********** Warning TxFIFO Number should be same as the Device IN Endpoint Number
21	Stall	R/W	0x0	STALL Handshake The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Nonperiodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority.
20	-	-	0x0	Reserved
19-18	EPType	R	0x0	Endpoint Type Hardcoded to 00 for control.
17	NAKSts	R	0x0	NAK Status Indicates the following: • 1’b0: The core is transmitting non-NAK handshakes based on the FIFO status • 1’b1: The core is transmitting NAK handshakes on this endpoint. When this bit is set, either by the application or core, the core stops transmitting data, even if there is data available in the TxFIFO. Irrespective of this bit’s setting, the core always responds to SETUP data packets with an ACK handshake.
16	-	-	0x0	Reserved
15	USBActEP	R	0x1	USB Active Endpoint This bit is always set to 1, indicating that control endpoint 0 is always active in all configurations and interfaces.
14-2	-	-	0x0	Reserved
1-0	MPS	R/W	0x0	Maximum Packet Size Applies to IN and OUT endpoints. The application must program this field with the maximum packet

				size for the current logical endpoint. • 2’b00: 64 bytes • 2’b01: 32 bytes • 2’b10: 16 bytes • 2’b11: 8 bytes

Device Control OUT Endpoint 0 Control Register (DOEPCTL0) **0xB0800B00**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPEna	EPDis			SNAK	CNAK					Stall	Snp	EPType		NAKSts	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBA ctEP														MPS	

This section describes the Control OUT Endpoint 0 Control register. Nonzero control endpoints use registers for endpoints 1–15.

Field	Name	RW	Reset	Description
31	EPEna	R/W	0x0	Endpoint Enable Indicates that the application has allocated the memory to start receiving data from the USB. The core clears this bit before setting any of the following interrupts on this endpoint: • SETUP Phase Done • Endpoint Disabled • Transfer Completed Note: In DMA mode, this bit must be set for the core to transfer SETUP data packets into memory.
30	EPDis	R	0x0	Endpoint Disable The application cannot disable control OUT endpoint 0.
29-28	-	-	0x0	Reserved
27	SNAK	W	0x0	Set NAK A write to this bit sets the NAK bit for the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also set this bit on a Transfer Completed interrupt, or after a SETUP packet is received on the endpoint.
26	CNAK	R/W	0x0	Clear NAK A write to this bit clears the NAK bit for the endpoint.
25-22	-	-	0x0	Reserved
21	Stall	R/W	0x0	STALL Handshake The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit’s setting, the core always responds to SETUP data packets with an ACK handshake.
20	Snp	R/W	0x0	Snoop Mode This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory.
19-18	EPType	R	0x0	Endpoint Type Hardcoded to 00 for control.
17	NAKSts	R	0x0	NAK Status Indicates the following: • 1’b0: The core is transmitting non-NAK handshakes based on the FIFO status. • 1’b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit, the core stops receiving data, even if there is space in the RxFIFO to accommodate the incoming packet. Irrespective of this bit’s setting, the core always responds to SETUP data packets with an ACK handshake.
16	-	-	0x0	Reserved
15	USBActEP	R	0x1	USB Active Endpoint

				This bit is always set to 1, indicating that control endpoint 0 is always active in all configurations and interfaces.
14-2	-	-	0x0	Reserved
1-0	MPS	R	0x0	Maximum Packet Size The maximum packet size for control OUT endpoint 0 is the same as what is programmed in control IN Endpoint 0. • 2'b00: 64 bytes • 2'b01: 32 bytes • 2'b10: 16 bytes • 2'b11: 8 bytes

Device Endpoint-n Control Register (DIEPCTLn/DOEPCTLn)

0xB0800900+(Endpoint_number*0x20) for IN endpoints
0xB0800B00+(Endpoint_number*0x20) for OUT endpoints
Endpoint_number:1≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EPEna	EPDis	SetD1PID/SetOddFr	SetD0PID/SetEvenFr	SNAK	CNAK	TxFNum				Stall	Snp	EPType		NAKSts	DPID/EO_FrNum
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
USBActEP					MPS										

The application uses this register to control the behavior of each logical endpoint other than endpoint 0.

Field	Name	RW	Reset	Description
31	EPEna	R/W	0x0	Endpoint Enable Applies to IN and OUT endpoints. For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint. For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. The core clears this bit before setting any of the following interrupts on this endpoint: • SETUP Phase Done (OUT only) • Endpoint Disabled • Transfer Completed Note: For control OUT endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory.
30	EPDis	R/W	0x0	Endpoint Disable Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint.
29	SetD1PID	W	0x0	Set DATA1 PID Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1.
	SetOddFr			Set Odd (micro)frame Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame.
28	SetD0PID	W	0x0	Set DATA0 PID Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0.
	SetEvenFr			Set Even (micro)frame Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum)

				field to even (micro)frame.
27	SNAK	W	0x0	Set NAK Applies to IN and OUT endpoints. A write to this bit sets the NAK bit for the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also set this bit for OUT endpoints on a Transfer Completed interrupt, or after a SETUP is received on the endpoint.
26	CNAK	W	0x0	Clear NAK A write to this bit clears the NAK bit for the endpoint.
25-22	TxFNum	R/W	0x0	TxFIFO Number These bits specify the FIFO number associated with this endpoint. Each active IN endpoint should be programmed to a separate FIFO number. This field is valid only for IN endpoints. ********** Warning TxFIFO Number should be same as the Device IN Endpoint Number
21	Stall	R/W	0x0	STALL Handshake Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core. Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit’ s setting, the core always responds to SETUP data packets with an ACK handshake.
20	Snp	R/W	0x0	Snoop Mode Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory.
19-18	EPType	R/W	0x0	Endpoint Type Applies to IN and OUT endpoints. This is the transfer type supported by this logical endpoint. • 2’ b00: Control • 2’ b01: Isochronous • 2’ b10: Bulk • 2’ b11: Interrupt
17	NAKSts	R	0x0	NAK Status Applies to IN and OUT endpoints. Indicates the following: • 1’ b0: The core is transmitting non-NAK handshakes based on the FIFO status. • 1’ b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: • The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. • For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. • For isochronous IN endpoints: The core sends out a zerolength data packet, even if there data is available in the TxFIFO. Irrespective of this bit’ s setting, the core always responds to SETUP data packets with an ACK handshake.
16	DPID	R	0x0	Endpoint Data PID ~~Applies to interrupt/bulk IN and OUT endpoints only.~~ Contains the PID of the packet to be received or transmitted on

				this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. Applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. • 1’b0: DATA0 • 1’b1: DATA1
	EO_FrNum			Even/Odd (Micro)Frame Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. • 1’b0: Even (micro)frame • 1’b1: Odd (micro)frame
15	USBActEP	R/W	0x0	USB Active Endpoint Applies to IN and OUT endpoints. Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program endpoint registers accordingly and set this bit.
14-11	-	-	0x0	Reserved
10-0	MPS	R/W	0x0	Maximum Packet Size Applies to IN and OUT endpoints. The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.

Device Endpoint-n Interrupt Register (DIEPINTn/DOEPINTn)

0xB0800908+(Channel_number*0x20) for IN Endpoints
0xB0800B08+(Channel_number*0x20) for OUT Endpoints
Channel_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							TxfifoUndrn/OutPktErr	TxFEmp	INEPNakEff/Back2BackSETup	INTknEPMis	INTknTXFEmp/OUTTknEPdis	TimeOUT/SetUp	AHBErr	EPDisbld	XferCompl

This register indicates the status of an endpoint with respect to USB- and AHB-related events. It is shown in

Figure 4.3. The application must read this register when the OUT Endpoints Interrupt bit or IN Endpoints Interrupt bit of the Core Interrupt register (GINTSTS.OEPInt or GINTSTS.IEPInt, respectively) is set. Before the application can read this register, it must first read the Device All Endpoints Interrupt (DAINT) register to get the exact endpoint number for the Device Endpoint-n Interrupt register. The application must clear the appropriate bit in this register to clear the corresponding bits in the DAINT and GINTSTS registers.

Field	Name	RW	Reset	Description
31-9	-	-	0x0	Reserved
8	TxfifoUndrn	R/W	0x0	Fifo Underrun Applies to IN endpoints Only This bit is valid only if thresholding is enabled. Core generates this interrupt when it see a transmit FIFO underrun condition for this endpoint.
	OutPktErr			OUT Packet Error Applies to OUT endpoints Only This interrupt is valid only when thresholding is enabled. This interrupt is asserted when the core sees an overflow or a CRC error for non-ISOC OUT packet.
7	TxFEmp	R	0x1	Transmit FIFO Empty This bit is valid only for IN Endpoints This interrupt is asserted when the TxFIFO for this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl)).
6	INEPNakEff	R/W	0x0	IN Endpoint NAK Effective Applies to periodic IN endpoints only. Indicates that the IN endpoint NAK bit set by the application has taken effect in the core. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit set (either by the application or by the core). This interrupt does not necessarily mean that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit.
	Back2BackSETup			Back-to-Back SETUP Packets Received Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets for this particular endpoint. For information about handling this interrupt, see page 4-119.
5	INTknEPMis	R/W	0x0	IN Token Received with EP Mismatch Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one for which the IN token was received. This interrupt is asserted on the endpoint for which the IN token was received. For OUT endpoints, this bit is Reserved
4	INTknTXFEmp	R/W	0x0	IN Token Received When TxFIFO is Empty Applies to non-periodic IN endpoints only. Indicates that an IN token was received when the associated TxFIFO (periodic/non-periodic) was empty. This interrupt is

				asserted on the endpoint for which the IN token was received.
	OUTTknEPdis			OUT Token Received When Endpoint Disabled Applies only to control OUT endpoints. Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint for which the OUT token was received.
3	TimeOUT	R/W	0x0	Timeout Condition Applies only to Control IN endpoints. Indicates that the core has detected a timeout condition on the USB for the last IN token on this endpoint.
	SetUp			SETUP Phase Done Applies to control OUT endpoints only. Indicates that the SETUP phase for the control endpoint is complete and no more back-to-back SETUP packets were received for the current control transfer. On this interrupt, the application can decode the received SETUP data packet.
2	AHBErr	R/W	0x0	AHB Error Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address.
1	EPDisbld	R/W	0x0	Endpoint Disabled Interrupt Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application’s request.
0	XferCompl	R/W	0x0	Transfer Completed Interrupt Applies to IN and OUT endpoints. Indicates that the programmed transfer is complete on the AHB as well as on the USB, for this endpoint.

Device Endpoint 0 Transfer Size Register (DIEPTSIZ0/DOEPTSIZ0)

0xB0800910 for IN Endpoints
0xB0800B10 for OUT Endpoints

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SUPCnt										PktCnt				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
									XferSize						

The application must modify this register before enabling endpoint 0. Once endpoint 0 is enabled using Endpoint Enable bit of the Device Control Endpoint 0 Control registers (DIEPCTL0.EPEna/DOEPCTL0.EPEna), the core modifies this register. The application can only read this register once the core has cleared the Endpoint Enable bit. SUPCnt field is available only in DOEPTSIZ0

Nonzero endpoints use the registers for endpoints 1–15.

[Device IN Endpoint 0 Transfer Size Register: DIEPTSIZ0]

Field	Name	RW	Reset	Description
31-21	-	-	0x0	Reserved
20-19	PktCnt	R/W	0x0	Packet Count Indicates the total number of USB packets that constitute the Transfer Size amount of data for endpoint 0. This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.
18-7	-	-	0x0	Reserved
6-0	XferSize	R/W	0x0	Transfer Size Indicates the transfer size in bytes for endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet from the external memory is written to the TxFIFO.

[Device OUT Endpoint 0 Transfer Size Register: DOEPTSIZ0]

Field	Name	RW	Reset	Description
31	-	-	0x0	Reserved
30-29	SUPCnt	R/W	0x0	SETUP Packet Count This field specifies the number of back-to-back SETUP data packets the endpoint can receive. • 2’b01: 1 packet • 2’b10: 2 packets • 2’b11: 3 packets
28-20	-	-	0x0	Reserved
20-19	PktCnt	R/W	0x0	Packet Count This field is decremented to zero after a packet is written into the RxFIFO.
18-7	-	-	0x0	Reserved
6-0	XferSize	R/W	0x0	Transfer Size Indicates the transfer size in bytes for endpoint 0. The core interrupts the application only after it has exhausted the transfer size amount of data. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.

Device Endpoint-n Transfer Size Register (DIEPTSIZn/DOEPTSIZn)

0xB0800910+(Endpoint_number*0x20) for IN Endpoints
0xB0800B10+(Endpoint_number*0x20) for OUT Endpoints
Endpoint_number:1≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	MC/ RxDPID/ SUPCnt		PktCnt										XferSize[18:16]		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
XferSize[15:0]															

The application must modify this register before enabling the endpoint. Once the endpoint is enabled using Endpoint Enable bit of the Device Endpoint-n Control registers (DIEPCTLn.EPEna/DOEPCTLn.EPEna), the core modifies this register. The application can only read this register once the core has cleared the Endpoint Enable bit.

This register is used only for endpoints other than Endpoint 0.

Field	Name	RW	Reset	Description
31	-	-	0x0	Reserved
30-29	MC	R/W	0x0	Multi Count Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints. • 2’b01: 1 packet • 2’b10: 2 packets • 2’b11: 3 packets
	-	R		For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core should fetch for an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp).
	RxDPID	R		Received Data PID Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint. • 2’b00: DATA0 • 2’b01: DATA2 • 2’b10: DATA1 • 2’b11: MDATA
	SUPCnt	R/W		SETUP Packet Count Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive. • 2’b01: 1 packet • 2’b10: 2 packets • 2’b11: 3 packets
28-19	PktCnt	R/W	0	Packet Count Indicates the total number of USB packets that constitute the Transfer Size amount of data for this endpoint. • IN Endpoints: This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO. • OUT Endpoints: This field is decremented every time a packet (maximum size or short packet) is written to the RxFIFO.
18-0	XferSize	R/W	0	Transfer Size This field contains the transfer size in bytes for the current endpoint. The core only interrupts the application after it has exhausted the transfer size amount of data. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. • IN Endpoints: The core decrements this field every time a packet from the external memory is written to the TxFIFO. • OUT Endpoints: The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.

Device Endpoint-n DMA Address Register (DIEPDMAn/DOEPDMAn)

0xB0800914+(Endpoint_number*0x20) for IN Endpoints
0xB0800B14+(Endpoint_number*0x20) for OUT Endpoints
Endpoint_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAAddr[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMAAddr[15:0]															

Field	Name	RW	Reset	Description
31-0	DMAAddr	R/W	0x0	DMA Address Holds the start address of the external memory for storing or fetching endpoint data. This register is incremented on every AHB transaction. Application can give only a DWORD-aligned address. Note: For control endpoints, this address stores control OUT data packets as well as SETUP transaction data packets. If more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten.

Device IN Endpoint Transmit FIFO Status Register (DTXFSTSn)

0xB0800918+(Endpoint_number*0x20) for IN Endpoints
Endpoint_number:0≤n≤15

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INEPTxFSpcAvail															

This read-only register contains the free space information for the Device IN endpoint TxFIFO.

Field	Name	RW	Reset	Description
31-16	-	-	0x0	Reserved
15-0	INEPTxFSpcAvail	R	0x0	IN Endpoint TxFIFO Space Avail Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words. • 16’h0: Endpoint TxFIFO is full • 16’h1: 1 word available • 16’h2: 2 words available • 16’hn: n words available (where 0≤n≤ 32,768) • 16’h8000: 32,768 words available • Others: Reserved

Power and Clock Gating Control Register (PCGCCTL) 0xB0800E00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INEPTxFSpcAvail															

This register is available in Host and Device modes. The application can use this register to control the core's power-down and clock gating features. Because the CSR module is turned off during power-down, this registers is implemented inthe AHB Slave BIU module.

Field	Name	RW	Reset	Description
31-5	-	-	0x0	Reserved
4	PhySuspended	R	0x0	PHY Suspended. Indicates that the PHY has been suspended. After the application sets the Stop Pclk bit (bit 0), this bit is updated once the PHY is suspended. Because the UTMI+ PHY suspend is controlled through a port, the UTMI+ PHY is suspended immediately after Stop Pclk is set.
3	RstPdwnModule	R/W	0x0	Reset Power-Down Modules This bit is valid only in Partial Power-Down mode. The application sets this bit when the power is turned off. The application clears this bit after the power is turned on and the PHY clock is up.
2	PwrClmp	R/W	0x0	Power Clamp The application sets this bit before the power is turned off to clamp the signals between the poweron modules and the power-off modules. The application clears the bit to disable the clamping before the power is turned on.
1	GateHclk	R/W	0x0	Gate Hclk The application sets this bit to gate hclk to modules other than the AHB Slave and Master and wakeup logic when the USB is suspended or the session is not valid. The application clears this bit when the USB is resumed or a new session starts.
0	StopPclk	R/W	0x0	Stop Pclk The application sets this bit to stop the PHY clock (phy_clk) when the USB is suspended, the session is not valid, or the device is disconnected. The application clears this bit when the USB is resumed or a new session starts.

USB OTG Configuration Register (OTGCR) 0xB0880020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												OTGAMAP		ENDS	ENDM

BIT	Name	RW	Reset	Description
3-2	OTGAMAP	R/W	2'b00	Fixed value of address[17:16] into USB OTG link slave interface
1	ENDS	R/W	0	Endian of slave interface of USB OTG link 0: little, 1: big
0	ENDM	R/W	0	Endian of master interface of USB OTG link 0: little, 1: big

4.3 Register Description for UTMI (USB PHY)

USB PHY Configuration Register0 (UPCR0) 0xB0880024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	PR	CM	RCS		RCD		SDI	FO	VBDS	DMPD	DPPD	TBSH	TBS	VBD	LBE

Field	Name	RW	Reset	Description
14	PR	R/W	0	When asserted, this customer-specific signal resets the corresponding port's transmit and receive logic without disabling the clocks within the USB 2.0 nanoPHY. 0: The transmit and receive FSMs are operational, and the line_state logic becomes sequential after 11 PHYCLOCK cycles. 1: The transmit and receive finite state machines are reset, and the line_state logic combinationally reflects the state of the single-ended receivers.
13	CM	R/W	1	This signal controls the power-down signals in the XO, Bias, and PLL blocks when the USB 2.0 nanoPHY is suspended. 0: The XO, Bias, and PLL Blocks remains powered in suspend mode. 1: The XO, Bias, and PLL Blocks are powered down in suspend mode.
12-11	RCS	R/W	1	This signal selects the reference clock source for the PLL block. **RCS** / **Common Block Power Down Control** 11 — The PLL uses CLKCORE as reference 10 — The PLL uses CLKCORE as reference 01 — The XO block uses an external clock supplied on the XO pin(default) 00 — The XO block uses the clock from a crystal
10-9	RCD	R/W	0	This signal selects the reference clock source for the PLL block. **RCD** / **Reference Clock Frequency Select** 11 — Reserved 10 — 48 MHz 01 — 24 MHz 00 — 12 MHz
8	SDI	R/W	1	This test signal enables you to perform IDDQ testing by powering down all analog blocks. **SID** / **IDDQ Test Enable** 1 — The analog blocks are powered down 0 — The analog blocks are not powered down.
7	FO	R/W	0	This controller signal enables the UTMI or serial interface. ** It is effective only when UPCR0.MODE is set and you should not clear this bit when UPCR0.MODE is set **FO** / **UTMI/Serial Interface Select** 1 — The TXENABLEN, FSDATAEXT, and FSSE0EXT inputs drive USB 2.0 nanoPHY data output onto the D+ and D- lines 0 — Data on the D+ and D- lines is transmitted and received through the UTMI.
6	VBDS	R/W	0	This controller signal enables the UTMI or serial interface. This signal selects the VBUSVLDEXT input or the internal Session Valid

<table>
<tr><td></td><td></td><td></td><td></td><td>comparator to indicate when the VBUS signal on the USB cable is valid

<table><tr><th>VBDS</th><th>External VBUS Valid Select</th></tr><tr><td>1</td><td>The VBUSVLDEXT input is used</td></tr><tr><td>0</td><td>The internal Session Valid comparator is used</td></tr></table></td></tr>
<tr><td>5</td><td>DMPD</td><td>R/W</td><td>1</td><td>This controller signal enables or disables the pull-down resistance on the D-line.
When an A/B device is acting as a host (Downstream-facing port), DPPULLDOWN and DMPULLDOWN are enabled. UTMI+/OTG-compliant controllers are not allowed to toggle DPPULLDOWN and DMPULLDOWN during normal operation. USB PHY Configuration Register2 (UPCR2).OPMODE = 2’b01(non-driving) has precedence over the DPPULLDOWN and DMPULLDOWN controls.
** It is effective only when UPCR0.MODE is set.

<table><tr><th>DMPD</th><th>D- Pull-Down Resistor Enable</th></tr><tr><td>1</td><td>The pull-down resistance on D- is enabled.</td></tr><tr><td>0</td><td>The pull-down resistance on D- is disabled.</td></tr></table></td></tr>
<tr><td>4</td><td>DPPD</td><td>R/W</td><td>1</td><td>This controller signal enables or disables the pull-down resistance on the D+line.
** It is effective only when UPCR0.MODE is set.

<table><tr><th>DPPD</th><th>D+ Pull-Down Resistor Enable</th></tr><tr><td>1</td><td>The pull-down resistance on D+ is enabled</td></tr><tr><td>0</td><td>The pull-down resistance on D+ is disabled</td></tr></table></td></tr>
<tr><td>3</td><td>TBSH</td><td>R/W</td><td>0</td><td>This controller signal enables or disables bits stuffing on DATAIN[7:0] when OPMODE[1:0] =2’b11

<table><tr><th>TBSH</th><th>Low-Byte Transmit Bit-Stuffing Enable</th></tr><tr><td>1</td><td>Bit stuffing is enabled</td></tr><tr><td>0</td><td>Bit stuffing is disabled.</td></tr></table></td></tr>
<tr><td>2</td><td>TBS</td><td>R/W</td><td>0</td><td>This controller signal enables or disables bits stuffing on DATAIN[7:0] when OPMODE[1:0] =2’b11

<table><tr><th>TBS</th><th>Low-Byte Transmit Bit-Stuffing Enable</th></tr><tr><td>1</td><td>Bit stuffing is enabled</td></tr><tr><td>0</td><td>Bit stuffing is disabled</td></tr></table></td></tr>
<tr><td>1</td><td>VBD</td><td>R/W</td><td>0</td><td>This signal is valid in device mode only when the VBUSVLDEXTSEL signal is high. VBUSVLDEXTSEL indicates whether the VBUS signal on the USB cable is valid. In addition, VBUSVLDEXTSEL enables the pull-up resistor on the D+ line.

<table><tr><th>VBD</th><th>External VBUS Valid Indicator</th></tr><tr><td>1</td><td>The VBUS signal is valid, and the pull-up resistor on D+ is enabled</td></tr><tr><td>0</td><td>The VBUS signal is not valid, and the pull-up resistor on D+ is disabled.</td></tr></table></td></tr>
<tr><td>0</td><td>LBE</td><td>R/W</td><td>0</td><td>This controller signal places the USB 2.0 nanoPHY in Loopback mode, which enables the receive and transmit logic concurrently.

<table><tr><th>LBE</th><th>Loopback Test Enable</th></tr><tr><td>1</td><td>During data transmission, the receive logic is enabled.</td></tr><tr><td>0</td><td>During data transmission, the receive logic is disabled.</td></tr></table></td></tr>
</table>

[VBUS Valid and End Interrupt]

VBUS is the USB power supply pin. When in device mode, an off-chip charge pump must provide power to VBUS pin. Alternatively VBUS can be set internally by VBUSVLDEXT and VBUSVLDEXTSEL in USB PHY Configuration Register 0(UPCR0). If VBUSVLDEXTSEL is set to 1, VBUSVLDEXT value can be supplied as VBUS data of USB PHY.

There are two interrupts related to VBUS and these interrupts, which indicate the state of VBUS, are generated from B_VALID and SESSEND signals. B_VALID signal is set to 1 when VBUS is 1 and SESSEND signal is set to 1 when VBUS is 0. Therefore, when VBUS value changes, one of two interrupts definitely occurs

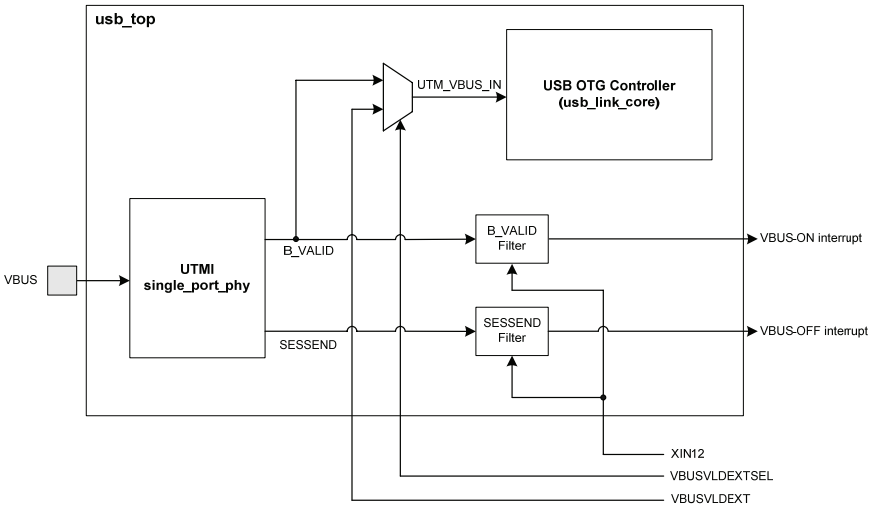


Figure 4.4 VBUS Control and Interrupts

USB PHY Configuration Register1 (UPCR1) 0xB0880028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXFSLST					SQRXT				OTGT				CDT		

Field	Name	RW	Reset	Description
15-12	TXFSLST	R/W	0x7	FS/LS Pull-Up Resistance Adjustment This bus adjusts the low- and full-speed pull-up resistance, based on nominal power, voltage, and temperature. 1111 -2.5% 0111 Design default (default) 0011 +2.5% 0001 +5% 0000 +7.5% Others Reserved
10-8	SQRXT	R/W	0x3	Squelch Threshold Tune This bus adjusts the voltage level for the threshold used to detect valid high-speed data 111 -20% 110 -15% 101 -10% 100 -5% 011 Design default (default) 010 +5% 001 +10% 000 +15% Others Reserved
6-4	OTGT	R/W	0x4	VBUS Valid Threshold Adjustment This bus adjusts the voltage level for the threshold used to detect valid high-speed data. 111 +9% 110 +6% 101 +3% 100 Design default(default) 011 -3% 010 -6% 001 -7% 000 -12% Others Reserved
2-0	CDT	R/W	0x3	Disconnect Threshold Adjustment This bus adjusts the voltage level for the threshold used to detect a disconnect event at the host. 111 +6% 110 +4.5% 101 +3% 100 +1.5% 011 Design default (default) 010 -1.5% 001 -3% 000 -4.5%

USB PHY Configuration Register2 (UPCR2) 0xB088002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM	XCVRSEL			OPMODE		TXVRT					TXRT			TP

Field	Name	RW	Reset	Description
14	TM	R/W	0	USB Termination Select 1 Full Speed termination is enabled (default). 0 High Speed termination is enabled. This controller signal selects FS or HS termination. ** It is effective only when UPCR0.MODE is set
13-12	XCVRSEL	R/W	0	FS/LS Pull-Up Resistance Adjustment 11 Sends an LS packet on an FS bus or receives an Ls packet 10 LS transceiver 01 FS transceiver (default) 00 HS transceiver This controller bus selects the HS, FS, or LS transceiver. ** It is effective only when UPCR0.MODE is set
10-9	OPMODE	R/W	0x3	UTMI Operational Mode 11 Normal operation SYNC or EOP generation 10 Disable bit stuffing and NRZI encoding 01 Non driving 00 Normal(default) This controller bus selects the UTMI operational mode. ** It is effective only when UPCR0.MODE is set.
8-5	TXVRT	R/W	0x8	HS DC Voltage Level Adjustment 1111 +8.75% 1110 +7.5% 1101 +6.25% 1100 +5% 1011 +3.75% 1010 +2.5% 1001 +1.25% 1000 Design default (default) 0111 -1.25% 0110 -2.5% 0101 -3.75% 0100 -5% 0011 -6.25% 0010 -7.5% 0001 -8.75% 0000 -10% This bus adjusts the voltage to which the high speed DC level is tuned.
3-2	TXRT	R/W	0	HS Transmitter Rise/Fall Time Adjustment 01 -8% 00 Design default (default) Others Reserved This bus adjusts the rise/fall times the high speed waveform
0	TP	R/W	0	HS transmitter Pre-Emphasis Enable 1 The HS Transmitter pre-emphasis is enabled 0 The HS transmitter pre-emphasis is disabled. (default) This signal enables or disables the pre-emphasis for a J-K or K-J state transition in HS mode

USB PHY Configuration Register3 (UPCR3) **0xB0880030**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SCALE		OTG									TXHSXVT		TXF	ABE

Field	Name	RW	Reset	Description
14-13	SCALE	R/W	0	Scale down mode 0 – This is enabled during simulation only. This controller signal powers down the OTG block, including the VBUS Valid comparator. If the application does not use OTG functionality, you can set this input high to save power.
12	OTG	R/W	0	OTG Disable 1 – The OTG block is powered down. 0 – The OTG block is not powered down(default) This controller signal powers down the OTG block, including the VBUS Valid comparator. If the application does not use OTG functionality, you can set this input high to save power.
3-2	TXHSXVT	R/W	1	Transmitter High Speed Crossover Adjustment 3 – The crossover voltage is increased by 15mV 2 – The crossover voltage is increased by 30mV 1 – Default setting 0 – Reserved TXHSXVT adjusts the voltage at which the DP and DM signals cross while transmitting in HS mode
1	TXF	R/W	0	TXF Mode 1 – Synchronize txf empty signal 0 – Do not synchronize txf empty signal
0	ABE	R/W	0	TX Jitter Adjustment 1 – TX Jitter Fix enable (default) 0 – TX Jitter Fix disable

4.4 Programming Model

4.4.1 Overview

This chapter describes the programming requirements for the otg core in Host and Device modes. Each significant programming feature of the otg core is discussed in a separate section.

4.4.2 Core Initialization

The application must perform the core initialization sequence following the configuration parameters the otg core. If the cable is connected during powerup, the Current Mode of Operation bit in the Core Interrupt register (GINTSTS.CurMod) reflects the mode. The otg core enters Host mode when an “A” plug is connected, or Device mode when a “B” plug is connected.

This section explains the initialization of the otg core after power-on. The application must follow the initialization sequence irrespective of Host or Device mode operation. All core global registers are initialized according to the core’s configuration.

1. Read the User Hardware Configuration registers (GHWCFG1, 2, 3, and 4) to find the configuration parameters selected for otg core.

2. Program the following fields in the Global AHB Configuration (GAHBCFG) register.

- DMA Mode bit
- AHB Burst Length field
- Global Interrupt Mask bit = 1

- RxFIFO Non-Empty (GINTSTS.RxFLvl) (applicable only when the core is operating in Slave mode)
- non-periodic TxFIFO Empty Level (can be enabled only when the core is operating in Slave mode as a host)
- Periodic TxFIFO Empty Level (can be enabled only when the core is operating in Slave mode)

3. Program the following fields in GUSBCFG register.

- HNP Capable bit
- SRP Capable bit
- External HS PHY
- UTMI+ Selection bit
- PHY Interface bit
- HS/FS TimeOUT Time-Out Calibration field
- USB Turnaround Time field

4. The software must unmask the following bits in the GINTMSK register.

- OTG Interrupt Mask
- Mode Mismatch Interrupt Mask

5. If the GUID register is selected for implementation, the software has the option of programming this register.

6. The software can read the GINTSTS.CurMod bit to determine whether the otg core is operating in Host or Device mode. The software the follows either the “Host Initialization” or “Device Initialization” sequence.

4.4.3 Host Initialization

To initialize the core as host, the application must perform the following steps.

1. Program GINTMSK.PrtInt to unmask.

2. Program the HCFG register to select full-speed host or high-speed host.

3. Program the HPRT.PrtPwr bit to 1’b1. This drives VBUS on the USB.

4. Wait for the HPRT0.PrtConnDet interrupt. This indicates that a device is connect to the port.

5. Program the HPRT.PrtRst bit to 1’b1. This starts the reset process.

6. Wait at least 10 ms for the reset process to complete.

7. Program the HPRT.PrtRst bit to 1’b0.

8. Wait for the HPRT.PrtEnChng interrupt.

9. Read the HPRT.PrtSpd field to get the enumerated speed.

10. Program the HFIR register with a value corresponding to the selected PHY clock[1].

11. Program the RXFSIZE register to select the size of the receive.

12. Program the NPTXFSIZE register to select the size and the start address of the Nonperiodic Transmit FIFO for non-periodic.

13. Program the HPTXFSIZ register to select the size and start address of the Periodic.

Transmit FIFO for periodic transactions.

To communicate with devices, the system software must initialize and enable at least one channel.

[1] At this point, the host is up and running and the port register begins to report device disconnects, etc. The port is active with SOFs occurring down the enabled port.

4.4.4 Device Initialization

The application must perform the following steps to initialize the core as a device on powerup or after a mode change from Host to Device.

1. Program the following fields in the DCFG register.

- Device Speed
- Non-Zero-Length Status OUT Handshake
- Periodic Frame Interval (when periodic endpoints are supported)

2. Program the Device threshold control register. This is required only if you are using DMA mode and you are planning to enable thresholding.

3. Program the GINTMSK register to unmask the following interrupts.

- USB Reset
- Enumeration Done
- Early Suspend
- USB Suspend
- SOF

4. Wait for the GINTSTS.USBReset interrupt, which indicates a reset has been detected on the USB and lasts for about 10 ms. On receiving this interrupt, the application must perform the steps listed in “Initialization on USB Reset”.

5. Wait for the GINTSTS.EnumerationDone interrupt. This interrupt indicates the end of reset on the USB. On receiving this interrupt, the application must read the DSTS register to determine the enumeration speed and perform the steps listed in “Initialization on Enumeration Completion”. At this point, the device is ready to accept SOF packets and perform control transfers on control endpoint 0.

4.5 Modes of Operation

The application can operate the core either in DMA mode, where the core fetches the data to be transmitted or updates the received data on the AHB, or in Slave mode, where the application initiates transfers for data fetch and store. The application cannot operate the core using DMA and Slave modes simultaneously.

4.5.1 DMA Mode

With an internal DMA option, the otg host uses the AHB Master interface for transmit packet data fetch (AHB to USB) and receive data update (USB to AHB). The AHB Master uses the programmed DMA address (HCDMAn register in Host mode and DIEPWMAn/DOEPWMAn register in Device mode) to access the data buffers.
When Scatter/Gather DMA is enabled in device mode DIEPDMAn/DOEPDMAn registers are used to access the base descriptor.

4.5.1.1 Transfer-Level Operation

In DMA mode, the application is interrupted only after the programmed transfer size istransmitted or received (provided the otg core detects no NAK/NYET/Timeout/Error response in Host mode, or Timeout/CRC Error in Device mode). The application must handle all transaction errors. In Device mode, all the USB errors are handled by the core itself

4.5.1.2 Transaction-Level Operation

This mode is similar to transfer-level operation, with the programmed transfer size equal to one packet size (maximum size or short packet). When Scatter/Gather DMA is enabled, the transfer size is extracted from the descriptors.

4.5.2 Slave Mode

In Slave mode, the application can operate the otg core either in transaction-level (packet-level) operation or in pipelined transaction-level operation.

4.5.2.1 Transaction-Level Operation

The application handles one data packet at a time per channel/endpoint in transaction-level operations. Based on the handshake response received on the USB, the application determines whether to retry the transaction or proceed with the next, until the end of the transfer. The application is interrupted on completion of every packet. The application performs transaction-level operations for a channel/endpoint for a transmission (host: OUT/device: IN) or reception (host: IN/device: OUT).

[Host Mode]

For an OUT transaction, the application enables the channel and writes the data packet into the corresponding (Periodic or Non-periodic) transmit FIFO. The otg core automatically writes the channel number into the corresponding (Periodic or Non-periodic) Request Queue, along with the last DWORD write of the packet.

For an IN transaction, the application enables the channel and the otg core automatically writes the channel number into the corresponding Request queue. The application must wait for the packet received interrupt, then empty the packet from the receive FIFO.

[Device Mode]

For an IN transaction, the application enables the endpoint, writes the data packet into the corresponding transmit FIFO, and waits for the packet completion interrupt from the core.

For an OUT transaction, the application enables the endpoint, waits for the packet received interrupt from the core, then empties the packet from the receive FIFO.

Note

The application has to finish writing one complete packet before switching to a different channel/endpoint FIFO. Violating this rule results in an error.

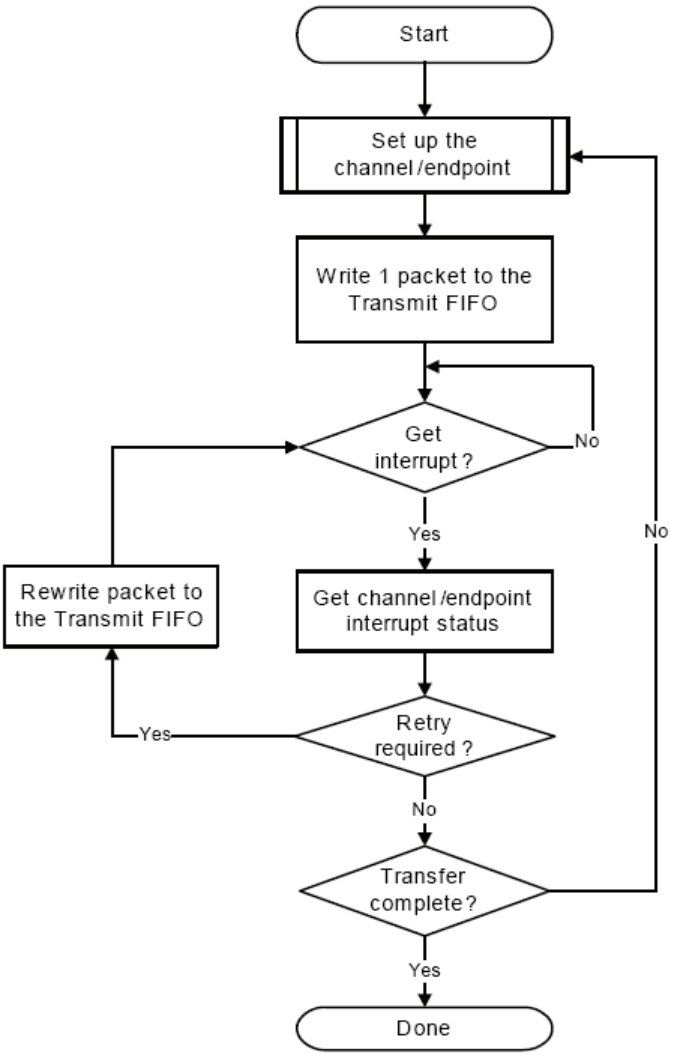


Figure 4.5 Transmit Transaction-Level Operation in Slave Mode

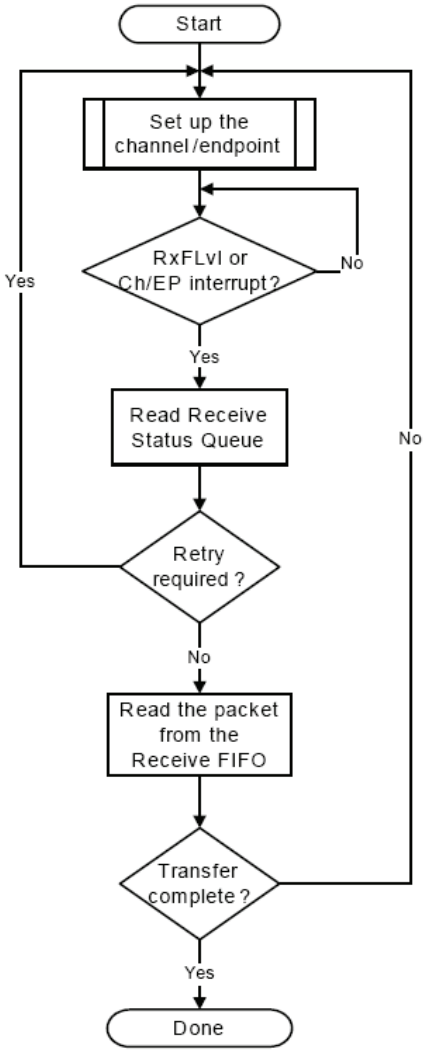


Figure 4.6 Receive Transaction-Level Operation in Slave Mode

4.5.2.2 Pipelined Transaction-Level Operation

The application can pipeline more than one transaction (IN or OUT) with pipelined transaction-level operation, which is analogous to Transfer mode in DMA mode. In pipelined transaction-level operation, the application can program the core to perform multiple transactions. The advantage of this mode of operation compared to transactionlevel operation is that the application is not interrupted on packet basis.

[Host Mode]

For an OUT transaction, the application sets up a transfer and enables the channel. The application can write multiple packets back-to-back for the same channel into the transmit FIFO, based on the space availability. It can also pipeline OUT transactions for multiple channels by writing into the HCHARn register, followed by a packet write to that channel. The core writes the channel number, along with the last DWORD write for the packet, into the Request queue and schedules transactions on the USB in the same order.
For an IN transaction, the application sets up a transfer and enables the channel, and the otg core writes the channel number into the Request queue. The application can schedule IN transactions on multiple channels, provided space is available in the Request queue. The core initiates an IN token on the USB only when there is enough space to receive at least of one maximum-packet-size packet of the channel in the top of the Request queue.

[Device Mode]

For an IN transaction, the application sets up a transfer and enables the endpoint. The application can write multiple packets back-to-back for the same endpoint into the transmit FIFO, based on available space. It can also pipeline IN transactions for multiple channels by writing into the DIEPCTLn register followed by a packet write to that endpoint. The core writes the endpoint number, along with the last DWORD write for the packet into the Request queue. The core transmits the data in the

transmit FIFO when an IN token is received on the USB.

For an OUT transaction, the application sets up a transfer and enables the endpoint. The core receives the OUT data into the receive FIFO, when it has available space. As the packets are received into the FIFO, the application must empty data from it.

From this point on in this chapter, the terms “Pipelined Transaction mode” and “Transfe mode” are used interchangeably.

4.5.3 Thresholding in DMA Mode

The application can program the core to do FIFO thresholding when operating as a device in DMA mode. With threshold support, the core can be configured to operate with less than maximum packet size FIFOs for a particular endpoint. This results in a smaller FIFO requirement when compared to non-thresholding mode.
When Scatter/Gather DMA is enabled and the application is setting up multiple transfers with one packet data scattered in more than one buffer, the application disables thresholding, to avoid reverting updated pointers in more than one descriptor on an underrun.
FIFO thresholding is supported only in DMA mode. FIFO thresholding is not supported when the core is operating as a host, even in DMA mode.

- The core allows both receive and transmit FIFO thresholding.
- Device Threshold Control Register bit DTHRCTL.RxThrRn must be set to enable receive thresholding. DTHRCTL.RxThrLen specifies the receive threshold size.
- Transmit uses separate Threshold Enable controls for isochronous and non-isochronous endpoints . Bits DTHRCTL.NonISOThrEn and DTHRCTL.ISOThrEn specify these Threshold Enable controls.
- The register filed DTHRCTL.TxThrLen specifies the transmit threshold length and is common for isochronous and non-isochronous endpoints. The minimum threshold length supported by the core is four DWORDs.
- Threshold enable controls cannot be changed randomly. The application can set or reset the threshold enable bits only after ensuring that the core is not programmed to do any transfers (FIFOs are flushed, NAK bits are set, all endpoints are disabled).
- One of the limitation of thresholding mode is in ping protocol, and could violate PING protocol. A PING token will be responded with an ACK handshake and the following OUT token could result in a NAK handshake. This behavior is a result of receive fifo overflow, and cannot be avoided in thresholding mode. This scenario will not occur ifthere are no overflows.

When transmit thresholding is enabled, the core starts transmitting data on the USB for a particular endpoint when there is threshold amount of data available in the corresponding transmit FIFO.

When receive thresholding is enabled, the core starts transferring data from the receive FIFO to the system memory as soon as there is threshold amount of data available in the receive FIFO. Any underrun or overflow conditions are handled by the core internally.

4.6 Host Programming Model

4.6.1 Channel Initialization

The application must initialize one or more channels before it can communicate with connected devices. To initialize and enable a channel, the application must perform the following steps.

1. Program the GINTMSK register to unmask the following:

- Channel Interrupt
- Non-periodic Transmit FIFO Empty for OUT transactions (applicable for Slave mode that operates in pipelined transaction-level with the Packet Count field programmed with more than one).
- Non-periodic Transmit FIFO Half-Empty for OUT transactions (applicable for Slave mode that operates in pipelined transaction-level with the Packet Count field programmed with more than one).

2. Program the HAINTMSK register to unmask the selected channels' interrupts.

3. Program the HCINTMSK register to unmask the transaction-related interrupts of interest given in the Host Channel Interrupt register.

4. Program the selected channel's HCTSIZn register with the total transfer size, in bytes, and the expected number of packets, including short packets. The application must program the PID field with the initial data PID (to be used on the first OUT transaction or to be expected from the first IN transaction).

5. Program the selected channels' HCSPLTn register(s) with the hub and port addresses (split transactions only).

6. Program the selected channels' HCDMAn register(s) with the buffer start address.

7. Program the HCCHARn register of the selected channel with the device's endpoint characteristics, such as type, speed, direction, and so forth. (The channel can be enabled by setting the Channel Enable bit to 1'b1 only when the application is ready to transmit or receive any packet).

Repeat steps 1–7 for other channels.

4.6.2 Halting a Channel

The application can disable any channel by programming the HCCHARn register with the HCCHARn.ChDis and HCCHARn.ChEna bits set to 1'b1. This enables the OTG host to flush the posted requests (if any) and generates a Channel Halted interrupt. The application must wait for the HCINTn.ChHltd interrupt before reallocating the channel for other transactions. The OTG host does not interrupt the transaction that has been already started on USB.
In Slave mode operation, before disabling a channel, the application must ensure that there is at least one free space available in the Non-periodic Request Queue (when disabling a nonperiodic channel) or the Periodic Request Queue (when disabling a periodic channel). The application can simply flush the posted requests when the Request queue is full (before disabling the channel), by programming the HCCHARn register with the HCCHARn.ChDis bit set to 1'b1, and the HCCHARn.ChEna bit reset to 1'b0.

To disable a channel in DMA mode operation, the application need not check for space in the Request queue. The OTG host checks for space in which to write the Disable request on the disabled channel's turn during arbitration. Meanwhile, all posted requests are dropped from the Request queue when the HCCHARn.ChDis bit is set to 1'b1.

The application is expected to disable a channel on any of the following conditions:

1. When a HCINTn.XferCompl interrupt is received during a non-periodic IN transfer or
high-bandwidth interrupt IN transfer (Slave mode only)

2. When a HCINTn.STALL, HCINTn.XactErr, HCINTn.BblErr, or HCINTn.DataTglErr interrupt is received for an IN or OUT channel (Slave mode only). For high-bandwidth interrupt INs in Slave mode, once the application has received a DataTglErr interrupt it must disable the channel and wait for a Channel Halted interrupt. The application must be able to receive other interrupts (DataTglErr, Nak, Data, XactErr, BabbleErr) for the same channel before receiving the halt.

3. When a GINTSTS.DisconnInt (Disconnect Device) interrupt is received. (The application is expected to disable all enabled channels in Slave and DMA modes)

4. When the application aborts a transfer before normal completion (Slave and DMA modes).

4.6.3 Ping Protocol

When the OTG host operates in high speed, the application must initiate the ping protocol when communicating with high-speed bulk or control (Data and Status stage) OUT endpoints. The application must initiate the ping protocol when it receives a NAK/NYET/XactErr interrupt. When the otg host receives one of the above responses, it does not continue any transaction for a specific endpoint, drops all posted or fetched OUT requests (from the Request queue), and flushes the corresponding data (from the transmit FIFO).

In Slave mode, the application can send a ping token either by setting the HCTSIZn.DoPng bit before enabling the channel or by just writing the HCTSIZn register with DoPng bit set when the channel is already enabled. This enables the otg host to write a ping request entry to the Request queue. The application must wait for the response to the ping token (a NAK, ACK, or XactErr interrupt) before continuing the transaction or sending another ping token. The application can continue the data transaction only after receiving an ACK from the OUT endpoint for the requested ping. The channel-specific interrupt service routine for the ping protocol in Slave mode is shown in table below.

In DMA mode operation, the application can start a ping protocol transfer by setting the HCTSIZn.DoPng bit before enabling the channel. The otg host continues sending ping tokens until receiving an ACK, then switches automatically to the data transaction.

Table 4.4 Interrupt Service Routine for Ping Protocol in Slave Mode

Ping Protocol for High Speed Bulk/Control OUT Endpoints

Unmask (ACK/NAK/XactErr/ChHltd/STALL)
if (ACK)
{
Reset Error Count
Re-initialize Channel (Do data transactions)
{
else if (NAK)
{
Reset Error Count
Send Ping
}
else if (STALL)
{
Disable Channel
}
else if (XactErr)
{
Increment Error Count
if (Error_count < 3)
{
Send Ping
}
else
{
Disable Channel
}
}
else if (ChHltd)
{
De-allocate Channel
}

4.6.4 Sending a Zero-Length Packet

To send a zero-length data packet, the application must initialize an OUT channel as follows.

1. Program the HCTSIZn register of the selected channel with a correct PID, XferSize = 0, and PktCnt = 1.

2. Program the HCCHARn register of the selected channel with ChEna = 1 and the device's endpoint characteristics, such as type, speed, and direction.

The application must treat a zero-length data packet as a separate transfer, and cannot combine it with a non-zero-length transfer.

4.6.5 Operational Model

The application must initialize a channel as described in “Channel Initialization” before communicating to the connected device. This section explains the sequence of operation to be performed for different types of USB transactions.

4.6.5.1 Writing the Transmit FIFO in Slave Mode

The figure below shows the flow diagram for writing to the transmit FIFO in Slave mode. The otg host automatically writes an entry (OUT request) to the Periodic/Non-periodic Request Queue, along with the last DWORD write of a packet. The application must ensure that at least one free space is available in the Periodic/Non-periodic Request Queue before starting to write to the transmit FIFO. The application must always write to the transmit FIFO in DWORDs. If the packet size is non-DWORD aligned, the application must use padding. The otg host determines the actual packet size based on the programmed maximum packet size and transfer size.

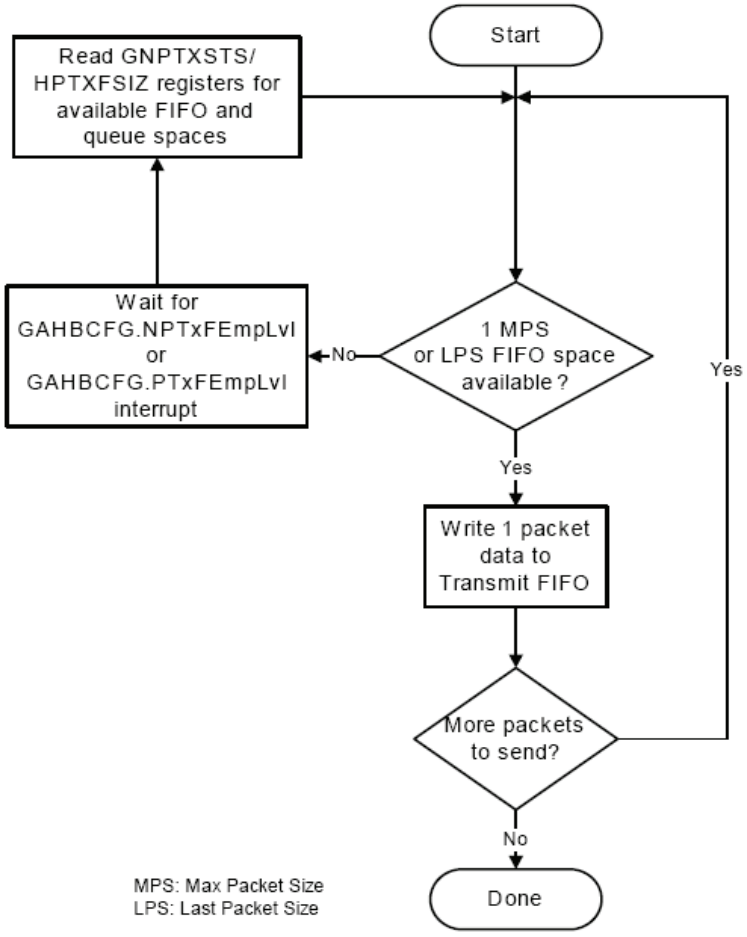


Figure 4.7 Transmit FIFO Write Task in Slave Mode

4.6.5.2 Reading the Receive FIFO in Slave Mode

The figure below shows the flow diagram for reading the receive FIFO in Slave mode. The application must ignore all packet statuses other than IN Data Packet (4’b0010).

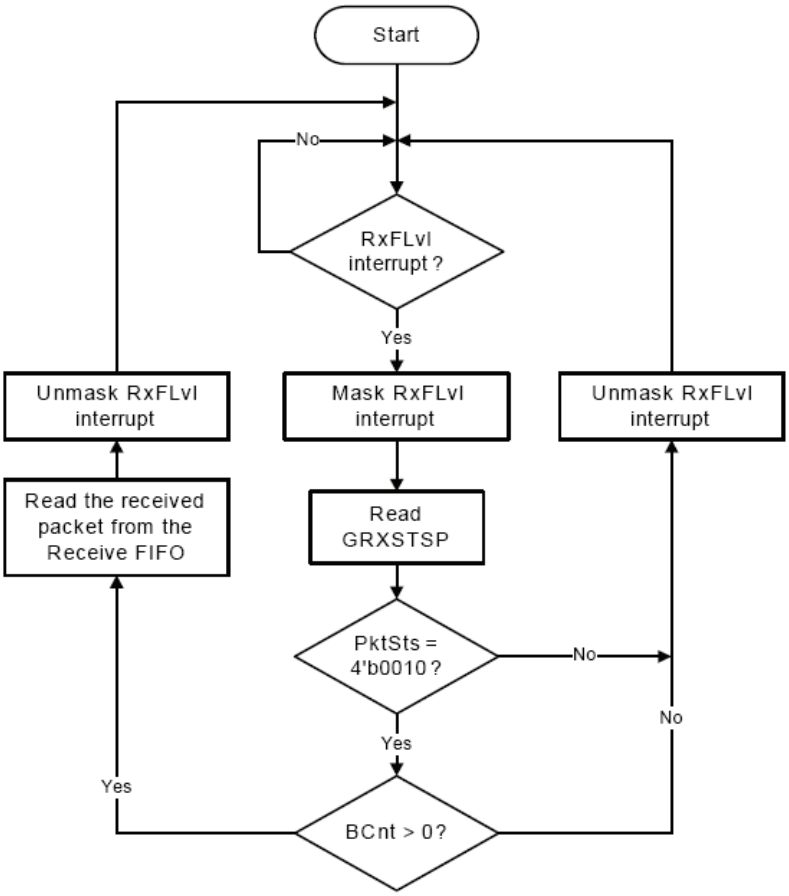


Figure 4.8 Receive FIFO Read Task in Slave Mode

4.6.5.3 Bulk and Control OUT/SETUP Transactions in Slave Mode

A typical bulk or control OUT/SETUP pipelined transaction-level operation in Slave mode is shown in figure below. See channel 1 (ch_1). Two bulk OUT packets are transmitted. A control SETUP transaction operates the same way but has only one packet. The assumptions are:

- The application is attempting to send two maximum-packet-size packets (transfer size = 1, 024 bytes).
- The Non-periodic Transmit FIFO can hold two packets (1 KB for HS or 128 KB for FS).
- The Non-periodic Request Queue depth = 4.

[Normal Bulk and Control OUT/SETUP Operations]

The sequence of operations in figure below(channel 1) is as follows:

1. Initialize channel 1 as explained in “Channel Initialization”.

2. Write the first packet for channel 1.

3. Along with the last DWORD write, the core writes an entry to the Non-periodic Request Queue.

4. As soon as the non-periodic queue becomes non-empty, the core attempts to send an OUT token in the current frame/microframe.

5. Write the second (last) packet for channel 1.

6. The core generates the XferCompl interrupt as soon as the last transaction is completed successfully.

7. In response to the XferCompl interrupt, de-allocate the channel for other transfers.

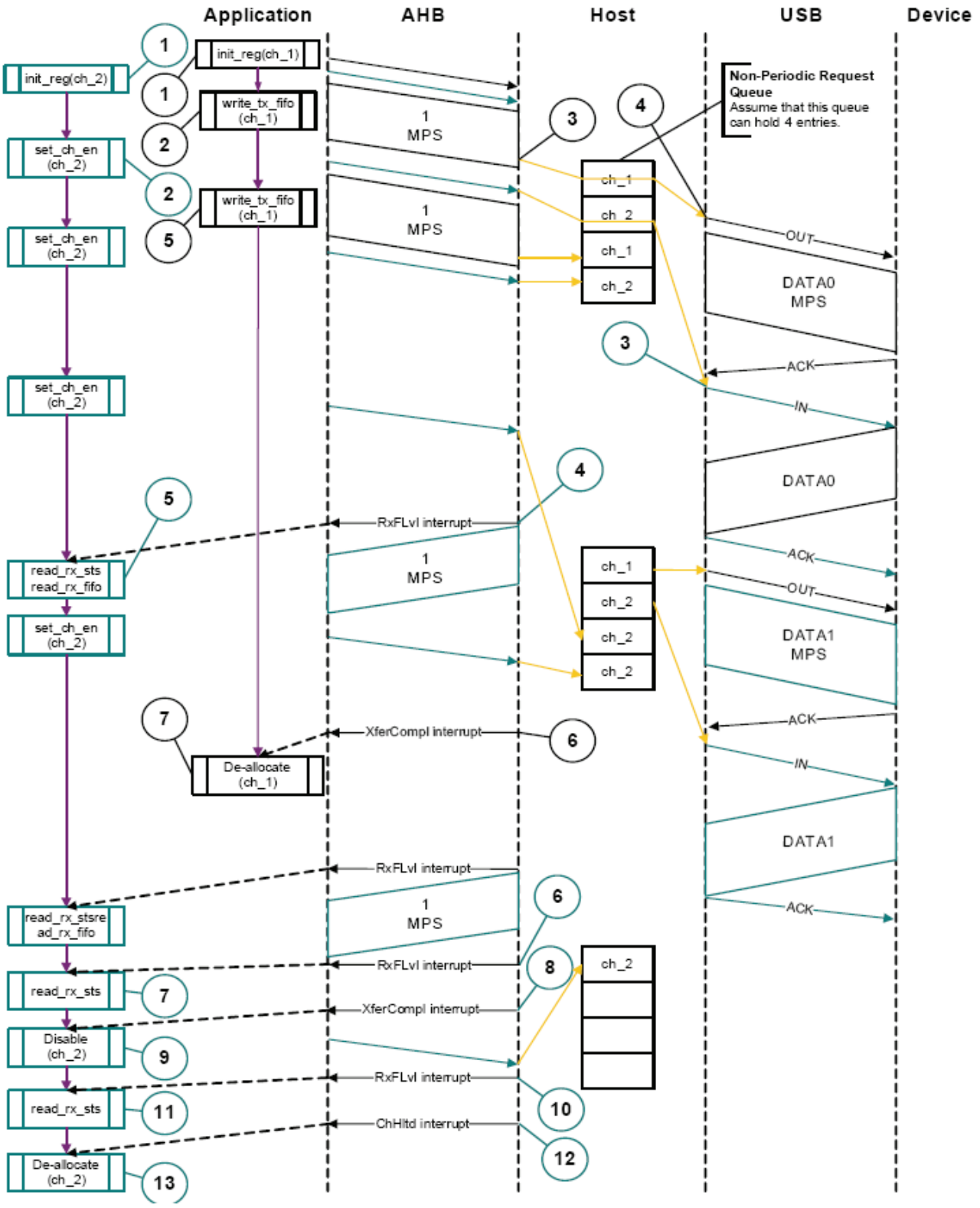


Figure 4.9 Normal Bulk/Control OUT/SETUP and Bulk/Control IN Transactions in Slave Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control OUT/SETUP transactions in Slave mode is shown in table below.

Table 4.5 Interrupt Service Routine for Bulk/Control OUT/SETUP and Bulk/Control IN Transactions in Slave Mode

Bulk/Control OUT/SETUP (Non-Split)	Bulk/Control IN (Non-Split)

Unmask (NAK/XactErr/NYET/STALL/XferCompl)
if (XferCompl)
{
Reset Error Count
Mask ACK
De-allocate Channel
}
else if (STALL)
{
Transfer Done = 1
Unmask ChHltd
Disable Channel
}
else if (NAK or XactErr or NYET)
{
Rewind Buffer Pointers
Unmask ChHltd
Disable Channel
if (XactErr)
{
Increment Error Count
Unmask ACK
}
else
{
Reset Error Count
}
}
else if (ChHltd)
{
Mask ChHltd
if (Transfer Done or (Error_count == 3))
{
De-allocate Channel
}
else
{
Re-initialize Channel (Do ping protocol for HS)
}
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
	Unmask
	(XactErr/XferCompl/BblErr/STALL/DataTglErr)
	if (XferCompl)
	{
	Reset Error Count
	Unmask ChHltd
	Disable Channel
	Reset Error Count
	Mask ACK
	}
	else if (XactErr or BblErr or STALL)
	{
	Unmask ChHltd
	Disable Channel
	if (XactErr)
	{
	Increment Error Count
	Unmask ACK
	}
	}
	else if (ChHltd)
	{
	Mask ChHltd
	if (Transfer Done or (Error_count == 3))
	{
	De-allocate Channel
	}
	else
	{
	Re-initialize Channel
	}
	}
	else if (ACK)
	{
	Reset Error Count
	Mask ACK
	}
	else if (DataTglErr)
	{
	Reset Error Count
	}

1. The application is expected to write the data packets into the transmit FIFO as and when the space is available in the transmit FIFO and the Request queue. The application can make use of GINTSTS.NPTxFEmp interrupt to find the transmit FIFO space.	1. The application is expected to write the requests as and when the Request queue space is available and until the XferCompl interrupt is received. 2. The application must clear and never modify the DoPing bit after enabling the channel and until the XferCompl or ChHltd interrupt is received. The core uses the DoPing bit to flush the excessive IN requests after receiving the last or short packet.

4.6.5.4 Bulk and Control IN Transactions in Slave Mode

A typical bulk or control IN pipelined transaction-level operation in Slave mode is shown in figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive two maximum-packet-size packets (transfer size = 1, 024 bytes).
- The receive FIFO can contain at least one maximum-packet-size packet and two status DWORDs per packet (520 bytes for HS or 72 bytes for FS).
- The Non-periodic Request Queue depth = 4.

[Normal Bulk and Control IN Operations]

The sequence of operations in figure above (channel 2) is as follows:

1. Initialize channel 2 as explained in “Channel Initialization”.

2. Set the HCCHAR2.ChEna bit to write an IN request to the Non-periodic Request Queue.

3. The core attempts to send an IN token after completing the current OUT transaction.

4. The core generates an RxFLvl interrupt as soon as the received packet is written to the receive FIFO.

5. In response to the RxFLvl interrupt, mask the RxFLvl interrupt and read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. Following this, unmask the RxFLvl interrupt.

6. The core generates the RxFLvl interrupt for the transfer completion status entry in the receive FIFO.

7. The application must read and ignore the receive packet status when the receive packet status is not an IN data packet (GRXSTSR.PktSts != 4’b0010).

8. The core generates the XferCompl interrupt as soon as the receive packet status is read.

9. In response to the XferCompl interrupt, disable the channel (as explained in “Halting a Channel”) and stop writing the HCCHAR2 register for further requests. The core writes a channel disable request to the Non-periodic Request Queue as soon as the HCCHAR2 register is written.

10. The core generates the RxFLvl interrupt as soon as the halt status is written to the receive FIFO.

11. Read and ignore the receive packet status.

12. The core generates a ChHltd interrupt as soon as the halt status is popped from the receive FIFO.

13. In response to the ChHltd interrupt, de-allocate the channel for other transfers.

Note

For Bulk/Control IN transfer, the application is expected to write the requests when the Request queue space is available, and until the XferCompl interrupt is received.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control IN transactions in Slave mode is shown in the table above.

4.6.5.5 Bulk and Control OUT/SETUP Transactions IN DMA Mode

A typical bulk or control OUT/SETUP operation in DMA mode is shown in figure below. See channel 1 (ch_1). Two bulk OUT packets are transmitted. A control SETUP transaction operates the same way but has only one packet. The assumptions are:

- The application is attempting to send two maximum-packet-size packets (transfer size = 1, 024 bytes).
- The Non-periodic Transmit FIFO can hold two packets (1 KB for HS or 128 bytes for FS).
- The Non-periodic Request Queue depth = 4.

[Normal Bulk and Control OUT/SETUP Operations]

The sequence of operations in
Figure 4.9 (channel 1) is as follows:

1. Initialize and enable channel 1 as explained in “Channel Initialization”.

2. The otg host starts fetching the first packet as soon as the channel is enabled. For internal DMA mode, the otg host uses the programmed DMA address to fetch the packet.

3. After fetching the last DWORD of the second (last) packet, the otg host masks channel 1 internally for further arbitration.

4. The otg host generates a ChHltd interrupt as soon as the last packet is received.

5. In response to the ChHltd interrupt, de-allocate the channel for other transfers.

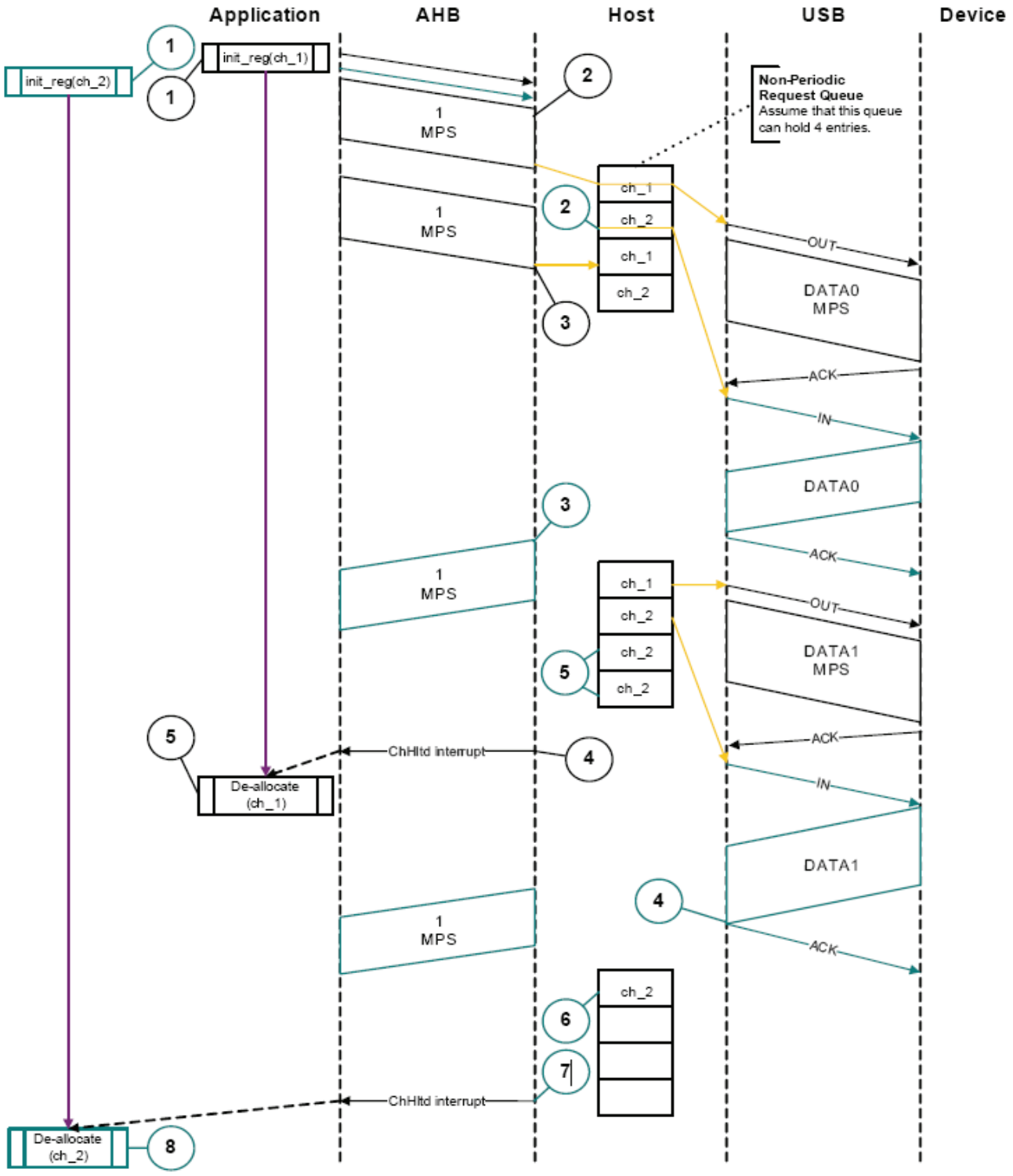


Figure 4.10 Normal Bulk/Control OUT/SETUP and Bulk/Control IN Transactions in DMA Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control OUT/SETUP transactions in DMA mode is shown in the table below.

Table 4.6 Interrupt Service Routines for Bulk/Control OUT/SETUP and Bulk/Control IN Transactions in DMA Mode

Bulk/Control OUT/SETUP (Non-Split)	Bulk/Control IN (Non-Split)

Unmask (ChHltd)
if (ChHltd)
{
if (XferCompl or STALL)
{
Reset Error Count
Mask ACK
De-allocate Channel
}
else if (NAK or XactErr or NYET)
{
Rewind Buffer Pointers
if (XactErr)
{
if (Error_count == 2)
{
De-allocate Channel
}
else
{
Increment Error Count
Re-initialize Channel
}
}
else
{
Reset Error Count
Mask ACK
}
Re-initialize Channel
}
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
	Unmask (ChHltd)
	if (ChHltd)
	{
	if (XferCompl or STALL or BblErr)
	{
	Reset Error Count
	Mask ACK
	De-allocate Channel
	}
	else if (XactErr)
	{
	if (Error_count == 2)
	{
	De-allocate Channel
	}
	else
	{
	Unmask ACK
	Unmask NAK
	Unmask DataTglErr
	Increment Error Count
	Re-initialize Channel
	}
	}
	}
	else if (ACK or NAK or DataTglErr)
	{
	Reset Error Count
	Mask ACK
	Mask NAK
	Mask DataTglErr
	}

1. As soon as the channel is enabled, the core attempts to fetch and write data packets, in multiples of the maximum packet size, to the transmit FIFO when space is available in the transmit FIFO and the Request queue. The core stops fetching as soon as the last packet is fetched. 2. While continuing the transfer to a high-speed device, the application must set the DoPing bit before enabling the channel if the previous transaction ended with NAK, NYET, or XacrErr	1. The application must clear and never modify the DoPing bit after enabling the channel and until the ChHltd interrupt is received. The core uses the DoPing bit to flush the excessive IN requests after receiving the last or short packet.

response. In this case, the core starts with the ping protocol, then automatically switches to Data Transfer mode.	

4.6.5.6 Bulk and Control IN Transactions in DMA Mode

A typical bulk or control IN operation in DMA mode is shown in Figure 5-6. See channel 2 (ch_2). The assumptions are:

- The application is attempting to send two maximum-packet-size packets (transfer size = 1, 024 bytes).
- The application is attempting to receive two maximum-packet-size packets (transfer size = 1, 024 bytes).
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet (520 bytes for HS or 72 bytes for FS).
- The Non-periodic Request Queue depth = 4.

[Normal Bulk and Control IN Operations]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”.

2. The otg host writes an IN request to the Request queue as soon as channel 2 receives the grant from the arbiter. (Arbitration is performed in a round-robin fashion, with fairness.).

3. The otg host starts writing the received data to the system memory as soon as the last byte is received with no errors.

4. When the last packet is received, the otg host sets an internal flag to remove any extra IN requests from the Request queue.

5. The otg host flushes the extra requests.

6. The final request to disable channel 2 is written to the Request queue. At this point, channel 2 is internally masked for further arbitration.

7. The otg host generates the ChHltd interrupt as soon as the disable request comes to the top of the queue.

8. In response to the ChHltd interrupt, de-allocate the channel for other transfers.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control IN transactions in DMA mode is shown in the table above.

4.6.5.7 Control Transactions in Slave Mode

Setup, Data, and Status stages of a control transfer must be performed as three separate transfers. Setup- Data- or Status-stage OUT transactions are performed similarly to the bulk OUT transactions explained in “Bulk and Control OUT/SETUP Transactions in Slave Mode”. Data- or Status-stage IN transactions are performed similarly to the bulk IN transactions explained in “Bulk and Control IN Transactions in Slave Mode”.

For all three stages, the application is expected to set the HCCHAR1.EPType field to Control. During the Setup stage, the application is expected to set the HCTSIZ1.PID field to SETUP.

4.6.5.8 Control Transactions in DMA Mode

Setup, Data, and Status stages of a control transfer must be performed as three separate transfers. Setup- and Data- or Status-stage OUT transactions are performed similarly to the bulk OUT transactions explained in “Bulk and Control OUT/SETUP Transactions in DMA Mode”. Data- or Status-stage IN transactions are performed similarly to the bulk IN transactions explained in “Bulk and Control IN Transactions in DMA Mode”.

For all three stages, the application is expected to set the HCCHAR1.EPType field to Control. During the Setup stage, the application is expected to set the HCTSIZ1.PID field to SETUP.

4.6.5.9 Interrupt OUT Transactions in Slave Mode

A typical interrupt OUT operation in Slave mode is shown in the figure below. See channel 1 (ch_1). The assumptions are:

- The application is attempting to send one packet in every frame/microframe (up to 1 maximum packet size), starting with the odd frame/microframe (transfer size = 1, 024 bytes).
- The Periodic Transmit FIFO can hold one packet (1 KB for HS or FS).
- Periodic Request Queue depth = 4.

[Normal Interrupt OUT Operation]

The sequence of operations in the figure below (channel 1) is as follows:

1. Initialize and enable channel 1 as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit.

2. Write the first packet for channel 1. For a high-bandwidth interrupt transfer, the application must write the subsequent packets up to MC (maximum number of packets to be transmitted in the next frame/microframe times before switching to another channel.

3. Along with the last DWORD write of each packet, the otg host writes an entry to the Periodic Request Queue.

4. The otg host attempts to send an OUT token in the next (odd) frame/microframe.

5. The otg host generates an XferCompl interrupt as soon as the last packet is transmitted successfully.

6. In response to the XferCompl interrupt, reinitialize the channel for the next transfer.

[Handling Non-ACK Responses]

The table below shows the channel-specific ISR for an interrupt OUT transaction in Slave mode.

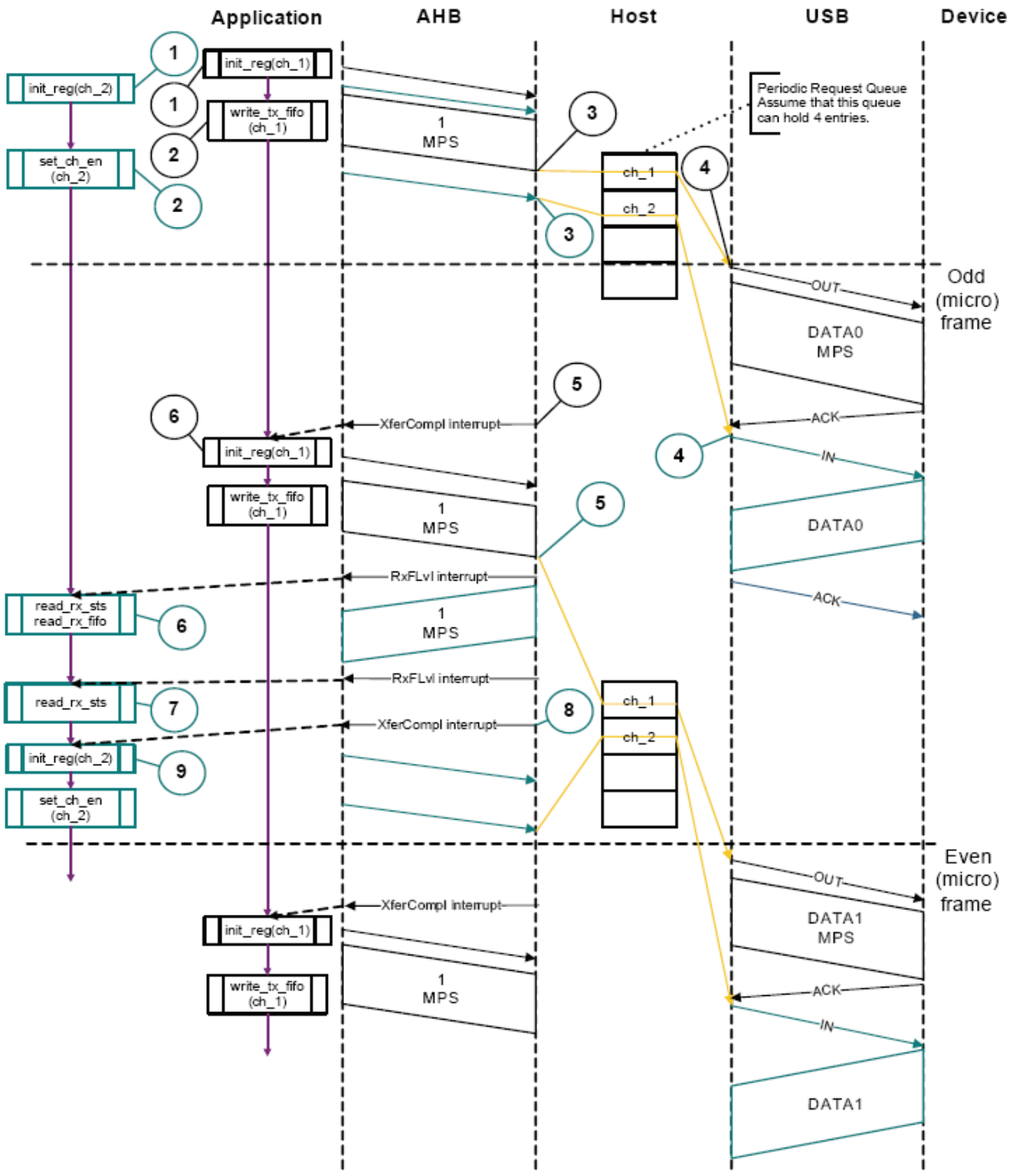


Figure 4.11 Normal Interrupt OUT/IN Transactions in Slave Mode

Table 4.7 Interrupt Service Routine for Interrupt OUT/IN Transactions in Slave Mode

Interrupt OUT (Non-Split)	Interrupt IN (Non-Split)

Unmask
(NAK/XactErr/STALL/XferCompl/FrmOvrun)
if (XferCompl)
{
Reset Error Count
Mask ACK
De-allocate Channel
}
else if (STALL or FrmOvrun)
{
Mask ACK
Unmask ChHltd
Disable Channel
if (STALL)
{
Transfer Done = 1
}
}
else if (NAK or XactErr)
{
Rewind Buffer Pointers
Reset Error Count
Mask ACK
Unmask ChHltd
Disable Channel
}
else if (ChHltd)
{
Mask ChHltd
if (Transfer Done or (Error_count == 3))
{
De-allocate Channel
}
else
{
Re-initialize Channel (in next b_interval - 1 uF/F)
}
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
	Unmask
	(NAK/XactErr/XferCompl/BblErr/STALL/FrmOvrun/DataTglErr)
	if (XferCompl)
	{
	Reset Error Count
	Mask ACK
	if (HCTSIZn.PktCnt == 0)
	{
	De-allocate Channel
	}
	else
	{
	Transfer Done = 1
	Unmask ChHltd
	Disable Channel
	}
	}
	else if (STALL or FrmOvrun or NAK or DataTglErr or BblErr)
	{
	Mask ACK
	Unmask ChHltd
	Disable Channel
	if (STALL or BblErr)
	{
	Reset Error Count
	Transfer Done = 1
	}
	else if (!FrmOvrun)
	{
	Reset Error Count
	}
	}
	else if (XactErr)
	{
	Increment Error Count
	Unmask ACK
	Unmask ChHltd
	Disable Channel
	}
	else if (ChHltd)
	{
	Mask ChHltd
	if (Transfer Done or (Error_count == 3))
	{
	De-allocate Channel
	}
	else
	{
	Re-initialize Channel (in next b_interval - 1 uF/F)
	}
	}
	else if (ACK)
	{
	Reset Error Count
	Mask ACK
	}

1. The application is expected to write the data packets into the transmit FIFO when the space is available in the transmit FIFO and the Request queue up to the count specified in the MC field before switching to another channel. The application uses the GINTSTS.NPTxFEmp interrupt to find the transmit FIFO space.	1. The application is expected to write the requests for the same channel when the Request queue space is available up to the count specified in the MC field before switching to another channel (if any).

4.6.5.10 Interrupt IN Transactions in Slave Mode

A typical interrupt IN operation in Slave mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one packet (up to 1 maximum packet size) in every frame/microframe, starting with odd. (transfer size = 1, 024 bytes).
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet (1, 032 bytes for HS or 1, 031 bytes for FS).
- Periodic Request Queue depth = 4.

[Normal Interrupt IN Operation]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize channel 2 as explained in “Channel Initialization”. The application must set the HCCHAR2.OddFrm bit.

2. Set the HCCHAR2.ChEna bit to write an IN request to the Periodic Request Queue. For a high-bandwidth interrupt transfer, the application must write the HCCHAR2 register MC (maximum number of expected packets in the next frame/microframe) times before switching to another channel.

3. The otg host writes an IN request to the Periodic Request Queue for each HCCHAR2 register write with a ChEna bit set.

4. The otg host attempts to send an IN token in the next (odd) frame/microframe.

5. As soon as the IN packet is received and written to the receive FIFO, the otg host generates an RxFLvl interrupt.

6. In response to the RxFLvl interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. The application must mask the RxFLvl interrupt before reading the receive FIFO, and unmask after reading the entire packet.

7. The core generates the RxFLvl interrupt for the transfer completion status entry in the receive FIFO. The application must read and ignore the receive packet status when the receive packet status is not an IN data packet (GRXSTSR.PktSts != 4’b0010).

8. The core generates an XferCompl interrupt as soon as the receive packet status is read.

9. In response to the XferCompl interrupt, read the HCTSIZ2.PktCnt field. If HCTSIZ2.PktCnt != 0, disable the channel (as explained in “Halting a Channel”) before re-initializing the channel for the next transfer, if any). If HCTSIZ2.PktCnt == 0, reinitialize the channel for the next transfer. This time, the application must reset the HCCHAR2.OddFrm bit.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for an interrupt IN transaction in Slave mode is shown in the Table above.

4.6.5.11 Interrupt OUT Transactions in DMA Mode

A typical interrupt OUT operation in DMA mode is shown in figure below. See channel 1 (ch_1). The assumptions are:

- The application is attempting to transmit one packet in every frame/microframe (up to 1 maximum packet size of 1, 024 bytes).
- The Periodic Transmit FIFO can hold one packet (1 KB for HS/FS).
- Periodic Request Queue depth = 4.

[Normal Interrupt OUT Operation]

1. Initialize and enable channel 1 as explained in “Channel Initialization”.

2. The otg host starts fetching the first packet as soon the channel is enabled and writes the OUT request along with the last DWORD fetch. In high-bandwidth transfers, the otg host continues fetching the next packet (up to the value specified in the MC field) before switching to the next channel.

3. The otg host attempts to send the OUT token in the beginning of the next odd frame/microframe.

4. After successfully transmitting the packet, the otg host generates a ChHltd interrupt.

5. In response to the ChHltd interrupt, reinitialize the channel for the next transfer.

[Handling Non-ACK Responses]

Table below shows the channel-specific ISR for an interrupt OUT transaction in DMA mode.

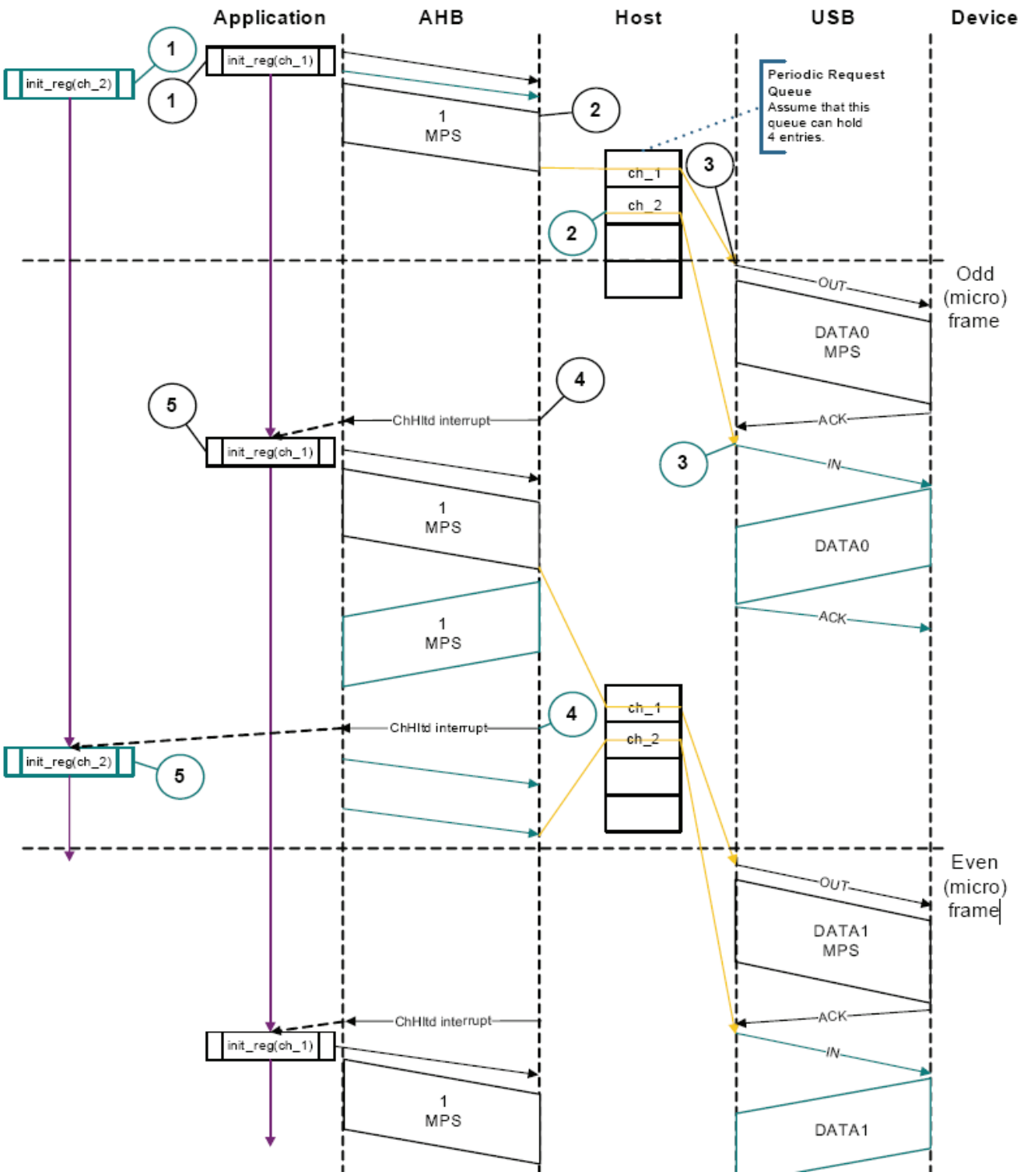


Figure 4.12 Normal Interrupt OUT/IN Transactions in DMA Mode

Table 4.8 Interrupt Service Routine for Interrupt OUT/IN Transactions in DMA Mode

Interrupt OUT (Non-Split)	Interrupt IN (Non-Split)

Unmask (ChHltd)
if (ChHltd)
{
if (XferCompl)
{
Reset Error Count
Mask ACK
if (Transfer Done)
{
De-allocate Channel
}
else
{
Re-initialize Channel (in next b_interval - 1 uF/F)
}
}
else if (STALL)
{
Transfer Done = 1
Reset Error Count
Mask ACK
De-allocate Channel
}
else if (NAK or FrmOvrun)
{
Mask ACK
Rewind Buffer Pointers
Re-initialize Channel (in next b_interval - 1 uF/F)
if (NAK)
{
Reset Error Count
}
}
else if (XactErr)
{
if (Error_count == 2)
{
De-allocate Channel
}
else
{
Increment Error Count
Rewind Buffer Pointers
Unmask ACK
Re-initialize Channel (in next b_interval - 1 uF/F)
}
}
}
else if (ACK)
{
Reset Error Count
Mask ACK
}
	Unmask (ChHltd)
	if (ChHltd)
	{
	if (XferCompl)
	{
	Reset Error Count
	Mask ACK
	if (Transfer Done)
	{
	De-allocate Channel
	}
	else
	{
	Re-initialize Channel (in next b_interval - 1 uF/F)
	}
	}
	else if (STALL or BblErr)
	{
	Reset Error Count
	Mask ACK
	De-allocate Channel
	}
	else if (NAK or DataTglErr or FrmOvrun)
	{
	Mask ACK
	Re-initialize Channel (in next b_interval - 1 uF/F)
	if (DataTglErr or NAK)
	{
	Reset Error Count
	}
	}
	else if (XactErr)
	{
	if (Error_count == 2)
	{
	De-allocate Channel
	}
	else
	{
	Increment Error Count
	Unmask ACK
	Re-initialize Channel (in next b_interval - 1 uF/F)
	}
	}
	}
	else if (ACK)
	{
	Reset Error Count
	Mask ACK
	}

1. As soon as the channel is enabled, the core attempts to fetch and write data packets, in maximum packet size multiples, to the transmit FIFO when the space is available in the transmit FIFO and the Request queue. The core stops fetching as soon as the last packet is fetched (the number of packets is determined by the MC field of the HCCHARn register).	1. As soon as the channel is enabled, the core attempts to write the requests into the Request queue when the space is available up to the count specified in the MC field.

4.6.5.12 Interrupt IN Transactions in DMA Mode

A typical isochronous IN operation in DMA mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one packet in every frame/microframe (up to 1 maximum packet size of 1, 024 bytes).
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet (1, 032 bytes for HS/FS).
- Periodic Request Queue depth = 4.

[Normal Interrupt IN Operation]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”.

2. The otg host writes an IN request to the Request queue as soon as the channel 2 gets the grant from the arbiter (round-robin with fairness). In high-bandwidth transfers, the otg host writes consecutive writes up to MC times.

3. The otg host attempts to send an IN token at the beginning of the next (odd) frame/microframe.

4. As soon the packet is received and written to the receive FIFO, the otg host generates a ChHltd interrupt.

5. In response to the ChHltd interrupt, reinitialize the channel for the next transfer.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for Interrupt IN transaction in DMA mode is shown in the table above.

4.6.5.13 Isochronous OUT Transactions in Slave Mode

A typical isochronous OUT operation in Slave mode is shown in figure below. See channel 1 (ch_1). The assumptions are: The application is attempting to send one packet every frame/microframe (up to 1 maximum packet size), starting with an odd frame/microframe. (transfer size = 1, 024 bytes).

- The Periodic Transmit FIFO can hold one packet (1 KB for HS/FS).
- Periodic Request Queue depth = 4.

[Normal Isochronous OUT Operation]

The sequence of operations in the figure below (channel 1) is as follows:

1. Initialize and enable channel 1 as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit.

2. Write the first packet for channel 1. For a high-bandwidth isochronous transfer, the application must write the subsequent packets up to MC (maximum number of packets to be transmitted in the next frame/microframe) times before switching to another channel.

3. Along with the last DWORD write of each packet, the otg host writes an entry to the Periodic Request Queue.

4. The otg host attempts to send the OUT token in the next frame/microframe (odd).

5. The otg host generates the XferCompl interrupt as soon as the last packet is transmitted successfully.

6. In response to the XferCompl interrupt, reinitialize the channel for the next transfer.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for an isochronous OUT transaction in Slave mode is shown in table below.

Table 4.9 Interrupt Service Routine for Isochronous OUT/IN Transactions in Slave Mode

Isochronous OUT (Non-Split)	Isochronous IN (Non-Split)

Unmask (FrmOvrun/XferCompl)
if (XferCompl)
{
De-allocate Channel
}
else if (FrmOvrun)
{
Unmask ChHltd
Disable Channel
}
else if (ChHltd)
{
Mask ChHltd
De-allocate Channel
}
	Unmask (XactErr/XferCompl/FrmOvrun/BblErr)
	if (XferCompl or FrmOvrun)
	{
	if (XferCompl and (HCTSIZn.PktCnt == 0))
	{
	Reset Error Count
	De-allocate Channel
	}
	else
	{
	Unmask ChHltd
	Disable Channel
	}
	}
	else if (XactErr or BblErr)
	{
	Increment Error Count
	Unmask ChHltd
	Disable Channel
	}
	else if (ChHltd)
	{
	Mask ChHltd
	if (Transfer Done or (Error_count == 3))
	{
	De-allocate Channel
	}
	else
	{
	Re-initialize Channel
	}
	}

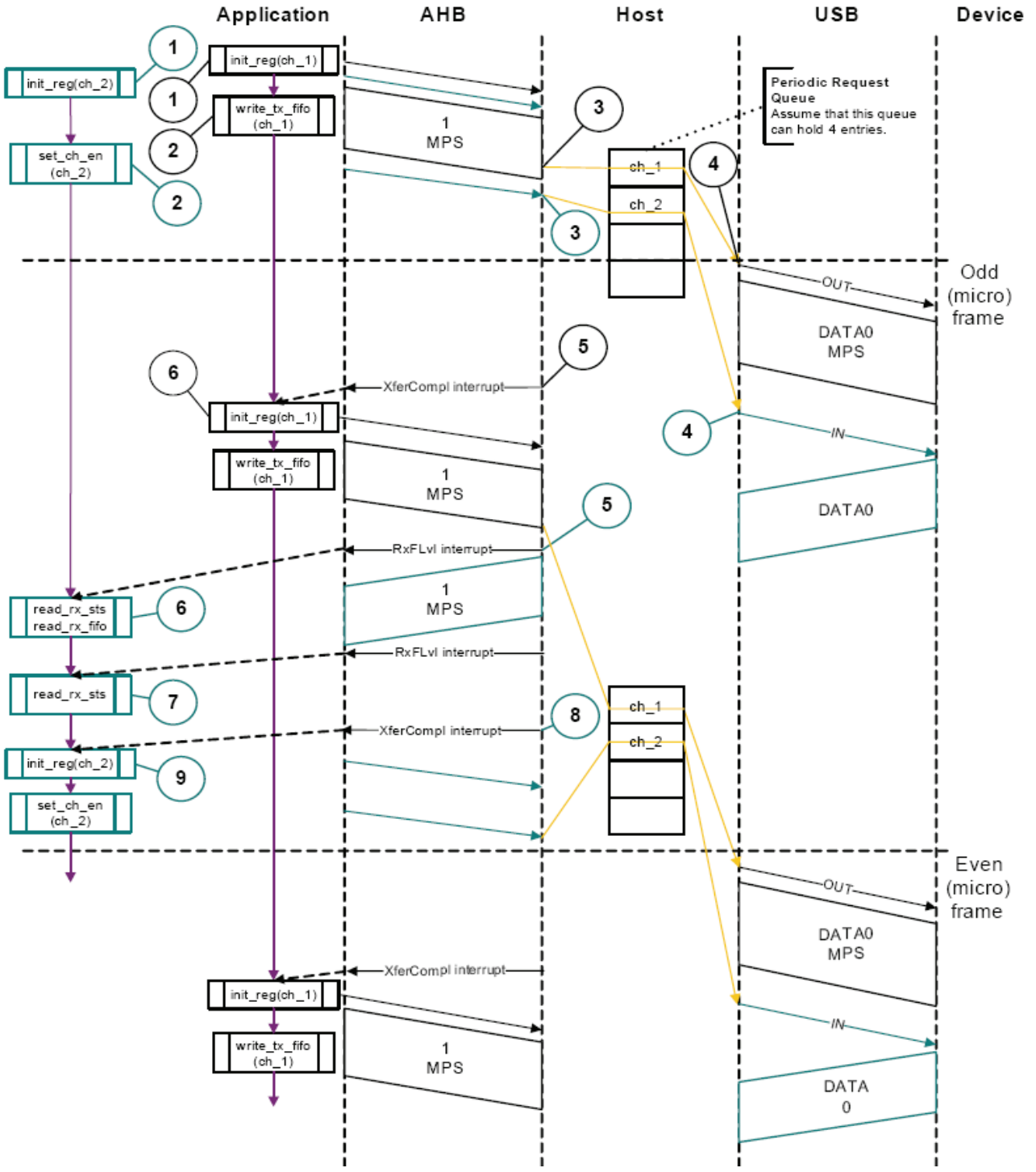


Figure 4.13 Normal Isochronous OUT/IN Transactions in Slave Mode

4.6.5.14 Isochronous IN Transactions in DMA Mode

A typical isochronous IN operation in DMA mode is shown in figure below. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one packet in every frame/microframe (up to 1 maximum packet size of 1, 024 bytes).
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDS per packet (1, 032 bytes for HS/FS).
- Periodic Request Queue depth = 4.

[Normal Isochronous IN Operation]

The sequence of operations in the figure above(channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”.

2. The otg host writes an IN request to the Request queue as soon as the channel 2 gets the grant from the arbiter (round-robin with fairness). In high-bandwidth transfers, the otg host performs consecutive writes up to MC times.

3. The otg host attempts to send an IN token at the beginning of the next (odd) frame/microframe.

4. As soon the packet is received and written to the receive FIFO, the OTG host generates a ChHltd interrupt.

5. In response to the ChHltd interrupt, reinitialize the channel for the next transfer.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for an isochronous IN transaction in DMA

mode is shown in the table above.

4.6.5.15 Bulk and Control OUT/SETUP Split Transactions in Slave Mode

A typical bulk or control SETUP/OUT operation in Slave mode is shown in figure below. See channel 1 (ch_1). Two bulk OUT packets are transmitted. A control SETUP transaction operates the same way but has only one packet. The assumptions are:

- The application is attempting to transmit one packet.

[Normal Bulk and Control OUT/SETUP Split Operations]

The sequence of operations in figure below (channel 1) is as follows:

1. Initialize and enable channel 1 as explained in “Channel Initialization”.

2. Write the packet for channel 1. Along with the last DWORD write of the packet, the otg host writes an entry to the Periodic Request Queue.

3. The otg host sends an OUT token.

4. The otg host generates an ACK interrupt as soon as the start split transaction completes successfully.

5. In response to the ACK interrupt, set the HCSPLT1.ComplSplt to send the complete split.

6. The otg host sends out the complete split transaction.

7. The otg host generates the XferCompl interrupt after successfully completing the complete split transaction.

8. In response to XferCompl interrupt, de-allocate the channel.

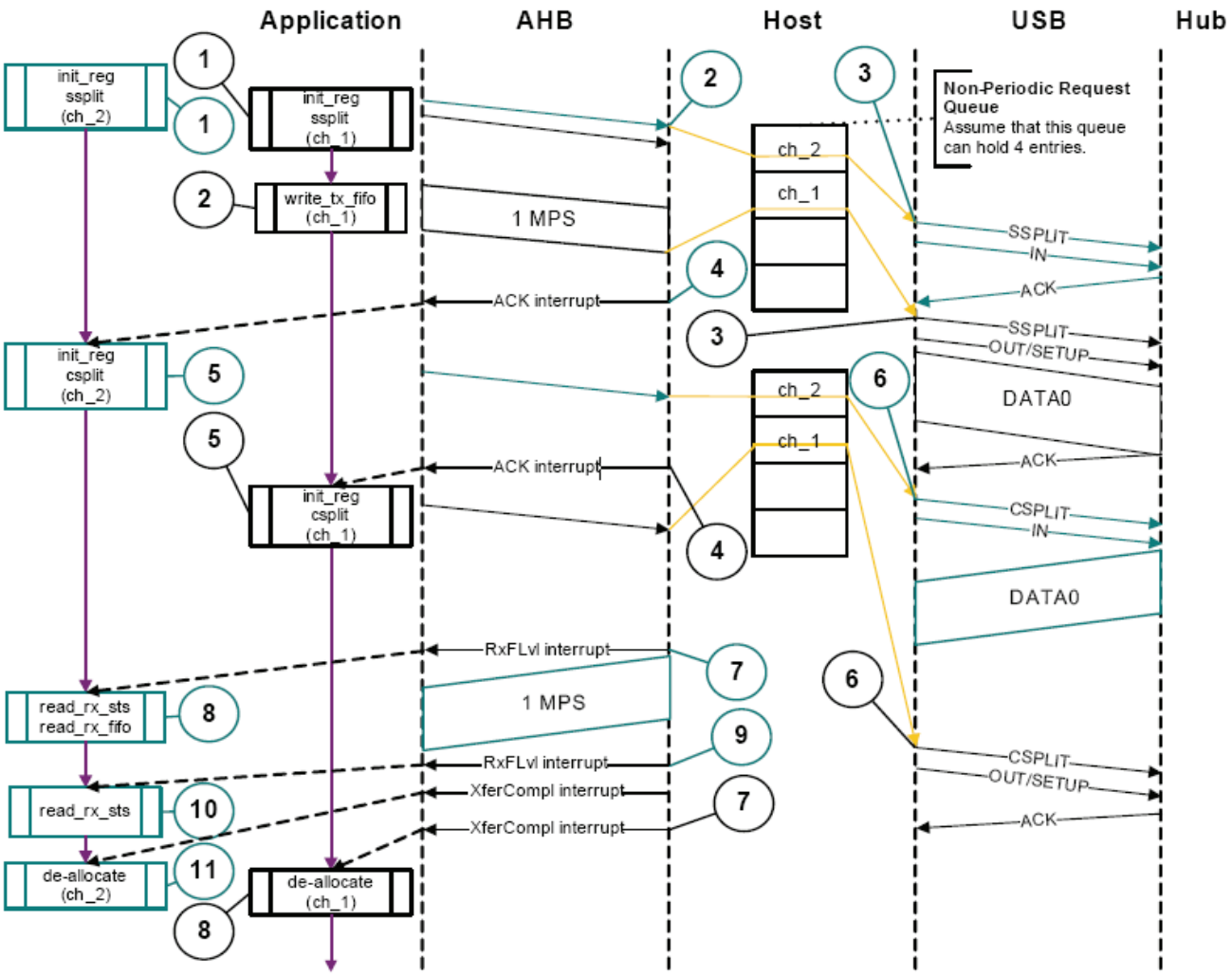


Figure 4.14 Normal Bulk/Control OUT/SETUP and Bulk/Control IN Split Transactions in Slave Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control OUT/SETUP split transactions in Slave mode is shown in table below.

Table 4.10 Interrupt Service Routine for Bulk/Control OUT/SETUP and Bulk/Control IN Split Transactions in Slave Mode

	Bulk/Control OUT/SETUP (Split)	Bulk/Control IN (Split)

Start Split	if (ACK)
	{
	Reset Error Count
	Do Complete Split
	}
	else if (NAK)
	{
	Rewind Buffer Pointers
	Retry Start Split
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	Increment Error Count
	if (Error_count < 3)
	{
	Retry Start Split
	}
	else
	{
	Unmask ChHltd
	Disable Channel
	}
	}
	else if (ChHltd)
	{
	Mask ChHltd
	De-allocate Channel
	}
		Unmask (ACK/NAK/XactErr/DataTglErr)
		if (ACK)
		{
		Reset Error Count
		Do Complete Split
		}
		else if (NAK)
		{
		Retry Start Split
		}
		else if (XactErr/DataTglErr)
		{
		Increment Error Count
		if (Error_count < 3)
		{
		Retry Start Split
		}
		else
		{
		Unmask ChHltd
		Disable Channel
		}
		}
		else if (ChHltd)
		{
		Mask ChHltd
		De-allocate Channel
		}

Complete Split	Unmask
	(NAK/XactErr/NYET/STALL/XferCompl)
	if (XferCompl)
	{
	De-allocate Channel
	}
	else if (NAK)
	{
	Rewind Buffer Pointers
	Retry Start Split
	}
	else if (NYET)
	{
	Retry Complete Split
	}
	else if (STALL)
	{
	Unmask ChHltd
	Disable Channel
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	Increment Error Count
	if (Error_count < 3)
	{
	Retry Start Split
	}
	else
	{
	Unmask ChHltd
	Disable Channel
	}
	}
		Unmask
		(NAK/XactErr/NYET/STALL/XferCompl)
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NAK)
		{
		Retry Start Split
		}
		else if (NYET)
		{
		Retry Complete Split
		}
		else if (STALL or BblErr)
		{
		Unmask ChHltd
		Disable Channel
		}
		else if (XactErr)
		{
		Increment Error Count
		if (Error_count < 3)
		{
		Retry Start Split
		}
		else
		{
		Unmask ChHltd
		Disable Channel
		}
		}
		else if (ChHltd)
		{

	else if (ChHltd)
	{
	Mask ChHltd
	De-allocate Channel
	}
		Mask ChHltd
		De-allocate Channel
		}

4.6.5.16 Bulk and Control IN Split Transactions in Slave Mode

A typical bulk or control IN operation in Slave mode is shown in the figure above (see channel 2 [ch_2). The assumptions are:

- The application is attempting to receive one packet.

[Normal Bulk and Control IN Split Operations]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”.

2. The otg host writes the Start Split request to the Periodic Request Queue as soon as the HCCHAR2 register is written.

3. The otg host sends the Start Split IN token.

4. As soon as the IN token is transmitted, the otg host generates an ACK interrupt.

5. In response to the ACK interrupt, set the HCSPLT2.ComplSplt bit to send the complete split token.

6. The otg host sends the complete split token.

7. As soon as the received packet is written to the receive FIFO, the otg host generates the RxFLvl interrupt.

8. In response to the RxFLvl interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. The application must mask the RxFLvl interrupt before reading the receive FIFO and unmask it after reading the entire packet.

9. The core generates the RxFLvl interrupt for the transfer completion status entry in the receive FIFO.

10. The application must read the receive packet status and, when the receive packet status is not an IN data packet (GRXSTSR.PktSts != 4’b0010), ignore it.

11. The core generates the XferCompl interrupt as soon as the receive packet status is read. In response to the XferCompl interrupt, read the HCTSIZ2.PktCnt field. If HCTSIZ2.PktCnt != 0, disable the channel (as explained in “Halting a Channel”) before re-initializing the channel for the next transfer, if any.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control IN split transactions inSlave mode is shown in the table above.

4.6.5.17 Bulk and Control OUT/SETUP Split Transactions in DMA Mode

A typical bulk or control OUT/SETUP operation in DMA mode is shown in figure below. See channel 1 (ch_1). Two bulk OUT packets are transmitted. A control SETUP transaction operates the same way but has only one packet. It is assumed that the application is attempting to transmit one packet.

[Normal Bulk and Control OUT/SETUP Split Operations]

The sequence of operations in figure below (channel 1) is as follows:

Initialize and enable channel 1 for start split as explained in “Channel Initialization”.
The host starts fetching the first packet as soon the channel is enabled and writes the OUT request along with the last DWORD fetch.

After successfully transmitting start split, the OTG host generates the ChHltd interrupt.

In response to the ChHltd interrupt, set the HCSPLT1.ComplSplt bit to send the complete split.

After successfully transmitting complete split, the OTG host generates the ChHltd interrupt.

In response to the ChHltd interrupt, de-allocate the channel.

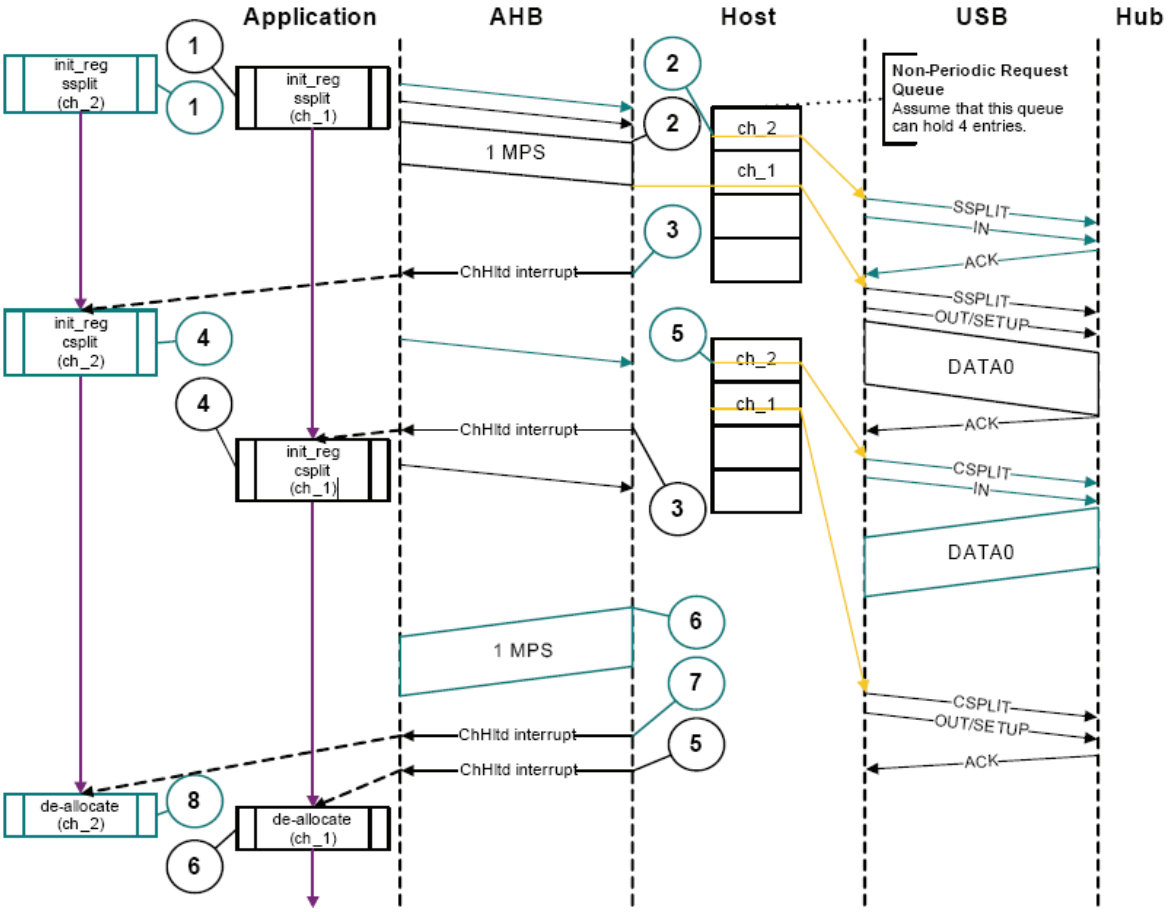


Figure 4.15 Normal Bulk/Control OUT/SETUP and Bulk/Control IN Split Transactions in DMA Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control OUT/SETUP split transactions in DMA mode is shown in table below.

Table 4.11 Interrupt Service Routine for Bulk/Control OUT/SETUP and Bulk/Control IN Split Transactions in DMA Mode

	Bulk/Control OUT/SETUP (Split)	Bulk/Control IN (Split)

Start Split	if (ChHltd)
	{
	if (ACK)
	{
	Reset Error Count
	Do Complete Split
	}
	else if (NAK)
	{
	Rewind Buffer Pointers
	Retry Start Split
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	Increment Error Count
	if (error_count < 3)
	{
	Retry Start Split
	}
	else
	{
	De-allocate Channel
	}
	}
	}
		Unmask (ChHltd)
		if (ChHltd)
		{
		if (ACK)
		{
		Reset Error Count
		Do Complete Split
		}
		else if (NAK)
		{
		Retry Start Split
		}
		else if (XactErr)
		{
		Increment Error Count
		if (error_count < 3)
		{
		Retry Start Split
		}
		else
		{
		De-allocate Channel
		}
		}
		}

Complete Split	Unmask (ChHltd)
	if (ChHltd)
	{
	if (XferCompl)
	{
	De-allocate Channel
	}
	else if (NAK)
	{
	Rewind Buffer Pointers
	Retry Start Split
	}
	else if (NYET)
	{
	Retry Complete Split
	}
	else if (STALL)
	{
	De-allocate Channel
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	Increment Error Count
	if (error_count < 3)
	{
	Retry Start Split
	}
	else
	{
	De-allocate Channel
	}
	}
	}
		Unmask (ChHltd)
		if (ChHltd)
		{
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NAK)
		{
		Retry Start Split
		}
		else if (NYET)
		{
		Retry Complete Split
		}
		else if (STALL/BblErr)
		{
		De-allocate Channel
		}
		else if (XactErr)
		{
		Increment Error Count
		if (error_count < 3)
		{
		Retry Start Split
		}
		else
		{
		De-allocate Channel
		}
		}
		}

4.6.5.18 Bulk/Control IN Split Transactions in DMA Mode

A typical bulk or control IN operation in DMA mode is shown in the figure above. See channel 1 (ch_1). The assumptions are:

- The application is attempting to receive one packet.

[Normal Bulk and Control IN Split Operations]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”.

2. The otg host writes the start split request to the non-periodic request after getting the grant from the arbiter. The otg host masks the channel 2 internally for the arbitration after writing the request.

3. As soon as the IN token is transmitted, the otg host generates the ChHltd interrupt.

4. In response to the ChHltd interrupt, set the HCSPLT2.ComplSplt bit and re-enable the channel to send the complete split token. This unmasks channel 2 for arbitration.

5. The otg host writes the complete split request to the non-periodic request after receiving the grant from the arbiter.

6. The otg host starts writing the packet to the system memory after receiving the packet successfully.

7. As soon as the received packet is written to the system memory, the otg host generates a ChHltd interrupt.

8. In response to the ChHltd interrupt, de-allocate the channel.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for bulk and control IN split transactions in DMA mode is shown in the table above.

4.6.5.19 Interrupt OUT Split Transactions in Slave Mode

A typical interrupt OUT split operation in Slave mode is shown in figure below. See channel 1 (ch_1). The assumptions are:

- The application is attempting to transmit one maximum-packet-size packet in an odd microframe.

[Normal Interrupt OUT Split Operation]

The sequence of operations in figure below (channel 1) is as follows:

1. Initialize and enable channel 1 as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit.

2. Write the packet for channel 1.

3. Along with the last DWORD write of the packet, the otg host writes an entry to the Periodic Request Queue.

4. The otg host attempts to send an OUT token in the next odd microframe.

5. The otg host generates an ACK interrupt as soon as the packet is transmitted successfully.

6. In response to the ACK interrupt, set the HCSPLT1.ComplSplt to send the complete split.

7. The otg host generates the XferCompl interrupt after successfully completing the complete split transaction.

8. In response to the XferCompl interrupt, de-allocate the channel.

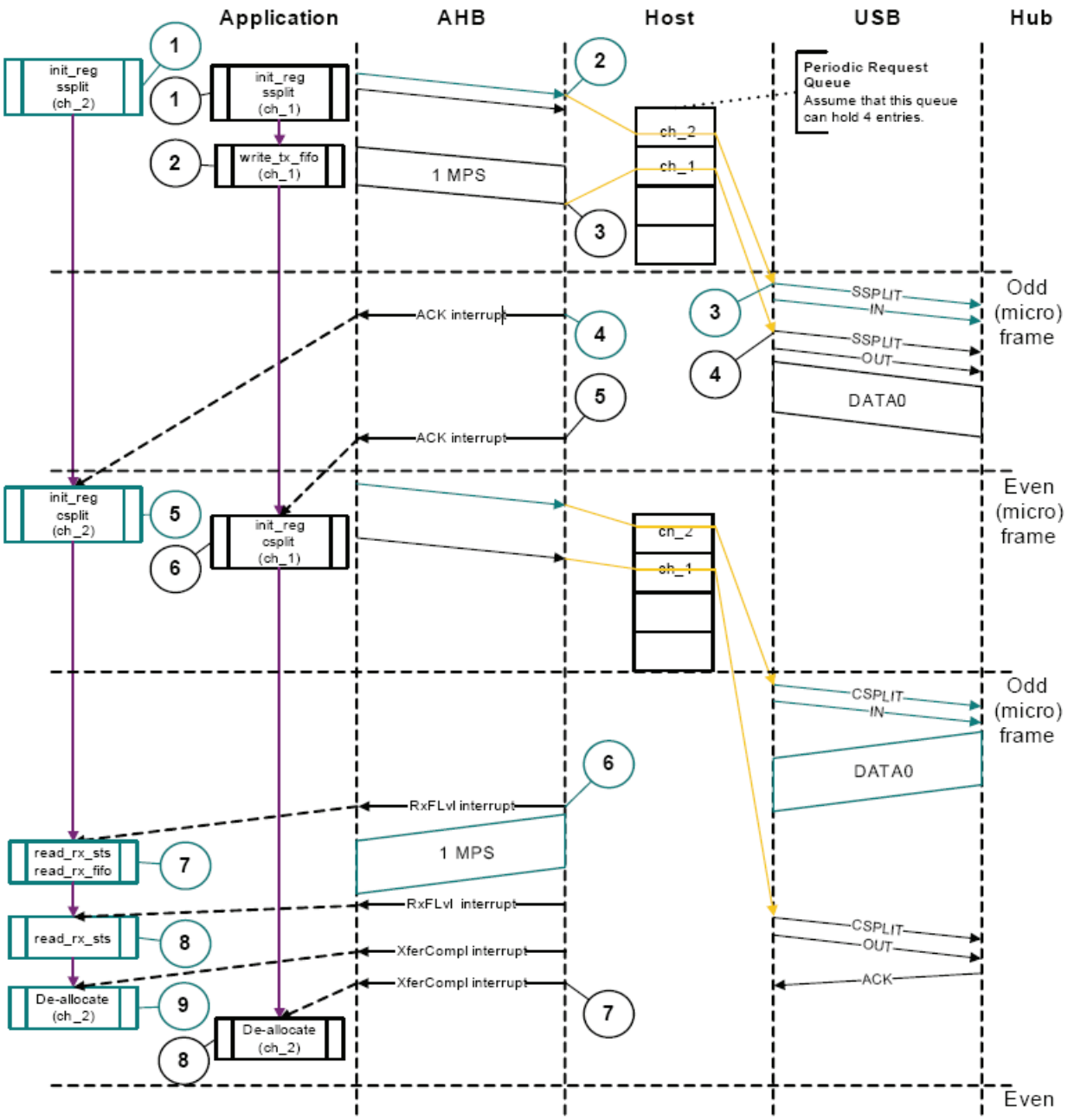


Figure 4.16 Normal Interrupt OUT/IN Split Transactions in Slave Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for interrupt OUT and IN split transactions in Slave mode is shown in table below.

	Interrupt OUT (Split)	Interrupt IN (Split)

Start Split	Unmask (ACK/FrmOvrun)
	if (ACK)
	{
	Do Complete Split
	}
	else if (FrmOvrun)
	{
	Rewind Buffer Pointers
	Unmask ChHltd
	Disable Channel
	(Retry Start Split (in next b_interval - 1 uF))
	}
	else if (ChHltd)
	{
	Mask ChHltd
	De-allocate Channel
	}
		Unmask (ACK/FrmOvrun/
		DataTglErr)
		if (ACK)
		{
		Do Complete Split
		}
		else if (FrmOvrun/DataTglErr)
		{
		Rewind Buffer Pointers
		Unmask ChHltd
		Disable Channel
		(Retry Start Split (in next b_interval - 1 uF))
		}
		else if (ChHltd)
		{
		Mask ChHltd
		De-allocate Channel
		}

Complete Split	Unmask
	(NAK/XactErr/NYET/STALL/XferCompl/FrmOvrun)
	if (XferCompl)
	{
	De-allocate Channel
	}
	else if (NAK)
	{
	Unmask ChHltd
	Disable Channel
	(Retry Start Split (in next b_interval - 1 uF))
	}
	else if (NYET)
	{
	Retry Complete Split
	}
	else if (STALL or FrmOvrun)
	{
	Unmask ChHltd
	Disable Channel
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	Unmask ChHltd
	Disable Channel
	(If (HCCHARn.EC == 3), Retry Start Split (in next b_interval - 1 uF))
	}
	else if (ChHltd)
	{
	Mask ChHltd
	De-allocate Channel
	}
		Unmask
		(NAK/XactErr/NYET/STALL/XferCompl/FrmOvrun/BblErr)
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NAK)
		{
		Unmask ChHltd
		Disable Channel
		(Retry Start Split (in next b_interval - 1 uF))
		}
		else if (NYET)
		{
		Retry Complete Split
		}
		else if (STALL or FrmOvrun or BblErr)
		{
		Unmask ChHltd
		Disable Channel
		}
		else if (XactErr)
		{
		Rewind Buffer Pointers
		Unmask ChHltd
		Disable Channel
		(If (HCCHARn.EC == 3), Retry Start Split (in next b_interval - 1 uF))
		}
		else if (ChHltd)
		{
		Mask ChHltd
		De-allocate Channel
		}

Note

The otg host tracks the error count in EC field for periodic split transactions. If the EC field matches the original programmed error count after XactErr interrupt, the application must treat the XactErr as “ERR response received.” The EC field indicates the number of immediate retries that the OTG host has performed before generating the XactErr interrupt.

4.6.5.20 Interrupt IN Split Transactions in Slave Mode

A typical interrupt IN split operation in Slave mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one maximum-packet-size packet in an odd microframe.
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet.

[Normal Interrupt IN Split Operation]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 as explained in “Channel Initialization”. The application must set the HCCHAR2.OddFrm bit.

2. The otg host writes the start split request to the Periodic Request Queue as soon as the HCCHAR2 register is written.

3. The otg host attempts to send a start split IN token in the next odd microframe.

4. As soon as the IN packet is transmitted, the otg host generates an ACK interrupt.

5. In response to the ACK interrupt, set the HCSPLT2.CompSplt bit in the next microframe to send the complete split token in the next odd microframe.

6. As soon as the received packet is written to the receive FIFO, the otg host generates the RxFLvl interrupt.

7. In response to the RxFLvl interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. The application must mask the RxFLvl interrupt before reading the receive FIFO and unmask after reading the entire packet.

8. The core generates the RxFLvl interrupt for the transfer completion status entry in the receive FIFO. The application must read the receive packet status and, if the receive packet status is not an IN data packet (GRXSTSR.PktSts != 4’b0010), ignore it.

9. The core generates the XferCompl interrupt as soon as the receive packet status is read. In response to the XferCompl interrupt, read the HCTSIZ2.PktCnt field. If HCTSIZ2.PktCnt != 0, disable the channel (as explained in “Halting a Channel” on page 216) before re-initializing the channel for the next transfer, if any. If HCTSIZ2.PktCnt == 0, reinitialize the channel for the next transfer. This time, the application must reset the HCCHAR2.OddFrm bit.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for an interrupt IN split transaction in Slave mode is shown in the table above.

4.6.5.21 Interrupt OUT Split Transactions in DMA Mode

A typical interrupt OUT split operation in DMA mode is shown in figure below (see channel 1 [ch_1]). It is assumed that the application is attempting to transmit one packet (1 maximum packet size) in an odd microframe.

[Normal Interrupt OUT Split Operation]

The sequence of operations in figure below (channel 1) is as follows:

1. Initialize and enable channel 1 for start split as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit.

2. The otg host starts reading the packet.

3. The otg host attempts to send the start split transaction.

4. After successfully transmitting the start split, the otg host generates the ChHltd interrupt.

5. In response to the ChHltd interrupt, set the HCSPLT1.ComplSplt bit to send the complete split.

6. After successfully completing the complete split transaction, the otg host generates the ChHltd interrupt.

7. In response to ChHltd interrupt, de-allocate the channel.

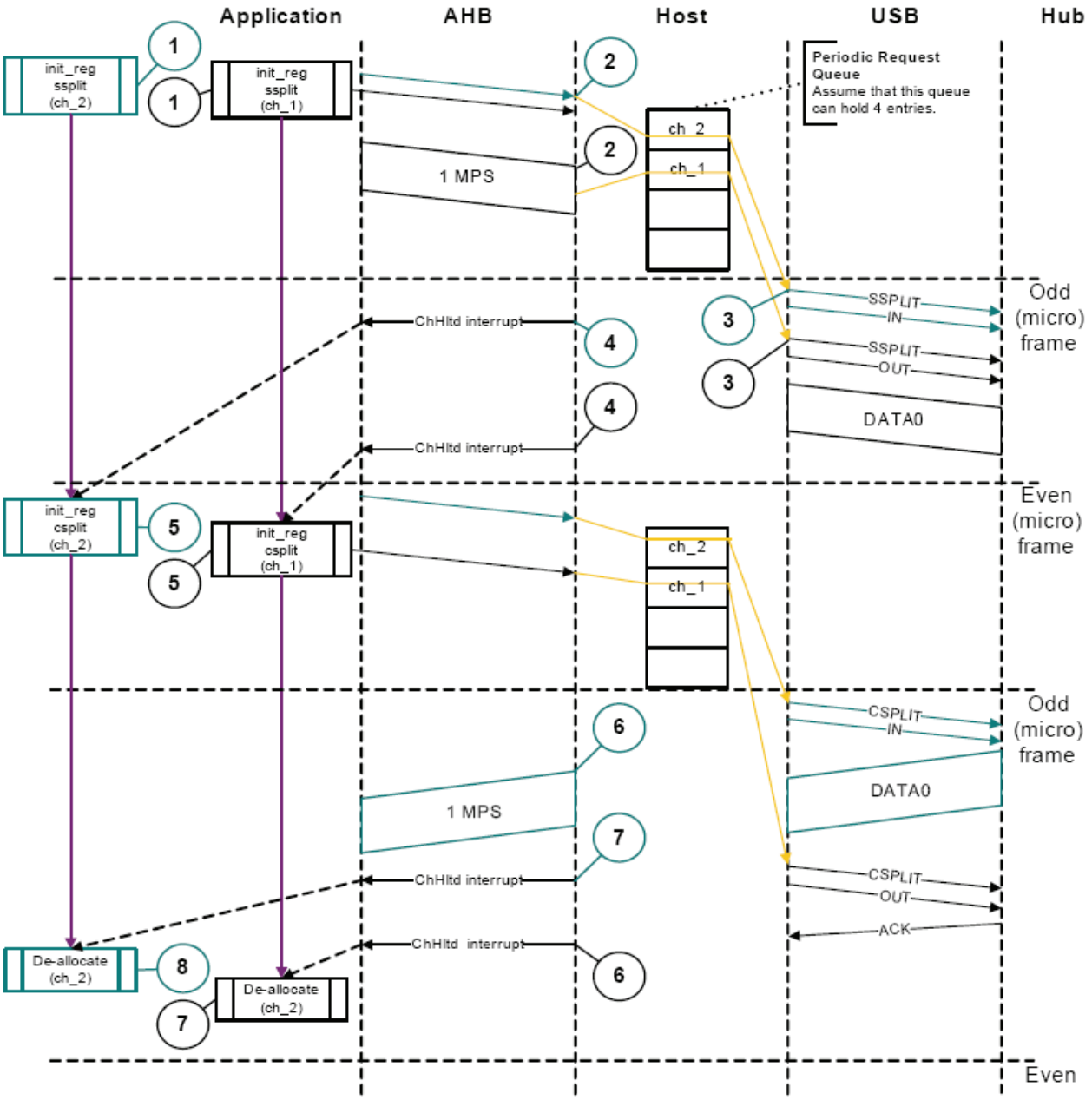


Figure 4.17 Normal Interrupt OUT/IN Split Transactions in DMA Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for interrupt OUT split transaction in DMA mode is shown in table below.

Table 4.12 Interrupt Service Routine for Interrupt OUT/IN Split Transactions in DMA Mode

	Interrupt OUT (Split)	Interrupt IN (Split)

Start Split	Unmask (ChHltd)
	if (ChHltd)
	{
	if (ACK)
	{
	Do Complete Split
	}
	else if (FrmOvrun)
	{
	Rewind Buffer Pointers
	Retry Start Split (in next b_interval - 1 uF)
	}
	}
		Unmask (ChHltd)
		if (ChHltd)
		{
		if (ACK)
		{
		Do Complete Split
		}
		else if (FrmOvrun)
		{
		Rewind Buffer Pointers
		Retry Start Split (in next b_interval - 1 uF)
		}
		}

Complete Split	Unmask (ChHltd)
	if (ChHltd)
	{
	if (XferCompl)
	{
	De-allocate Channel
	}
	else if (NAK)
	{
	Retry Start Split (in next b_interval - 1 uF)
	}
	else if (NYET)
	{
	Retry Complete Split
	}
	else if (STALL or FrmOvrun)
	{
	De-allocate Channel
	}
	else if (XactErr)
	{
	Rewind Buffer Pointers
	if (HCCHARn.EC == 3) // ERR response received
	{
	Retry Start Split (in next b_interval - 1 uF)
	}
	else
	{
	De-allocate Channel
	}
	}
	}
		Unmask (ChHltd)
		if (ChHltd)
		{
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NAK)
		{
		Retry Start Split (in next b_interval - 1 uF)
		}
		else if (NYET)
		{
		Retry Complete Split
		}
		else if (STALL or FrmOvrun or BblErr)
		{
		De-allocate Channel
		}
		else if (XactErr)
		{
		Rewind Buffer Pointers
		if (HCCHARn.EC == 3) // ERR response received
		{
		Retry Start Split (in next b_interval - 1 uF)
		}
		else
		{
		De-allocate Channel
		}
		}
		}

Note

The otg host tracks the error count in the EC field for periodic split transactions. If the EC field matches the original programmed error count after XactErr interrupt, the application must treat the XactErr as an ERR Response Received. The EC field indicates the number of immediate retries the OTG host has performed before generating the XactErr interrupt.

4.6.5.22 Interrupt IN Split Transactions in DMA Mode

A typical interrupt IN split operation in DMA mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one maximum-packet-size packet in an odd microframe.
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet.

[Normal Interrupt IN Split Operation]

The sequence of operations in the figure above (channel 2 {ch_2]) is as follows:

1. Initialize and enable channel 2 for start split as explained in “Channel Initialization”.

2. The otg host writes an IN request to the Request queue as soon as channel 2 receives the grant from the arbiter.

3. The otg host attempts to send the start split IN token in the beginning of the next odd microframe.

4. The otg host generates the ChHltd interrupt after successfully transmitting the start split IN token.

5. In response to the ChHltd interrupt, set the HCSPLT2.ComplSplt bit to send the complete split.

6. As soon the packet is received successfully, the otg host starts writing the data to the system memory.

7. The otg host generates the ChHltd interrupt after transferring the received data to the system memory.

8. In response to the ChHltd interrupt, de-allocate or reinitialize the channel for the next start split.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for interrupt IN split transaction in DMA mode is shown in the table above .

4.6.5.23 Isochronous OUT Split Transactions in Slave Mode

A typical isochronous OUT split operation in Slave mode is shown in the figure below. See channel 1 (ch_1). The assumptions are:

- The application is attempting to transmit a 376-byte packet in an odd microframe.

[Normal Isochronous OUT Split Operation]

The sequence of operations in the figure below (channel 1 [ch_1]) is as follows:

1. Initialize and enable channel 1 for start split (begin) as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit. Program the MPS field with 188 bytes.

2. Write the packet for channel 1.

3. Along with the last DWORD write of the packet, the otg host writes an entry to the Periodic Request Queue.

4. The otg host attempts to send an OUT token in the next odd microframe.

5. The otg host generates an ACK interrupt as soon as the packet is transmitted successfully.

6. In response to the ACK interrupt, reinitialize the registers to send the start split (end).

7. The otg host generates an ACK interrupt after successfully completing the start split (end) transaction.

8. In response to the ACK interrupt, de-allocate the channel.

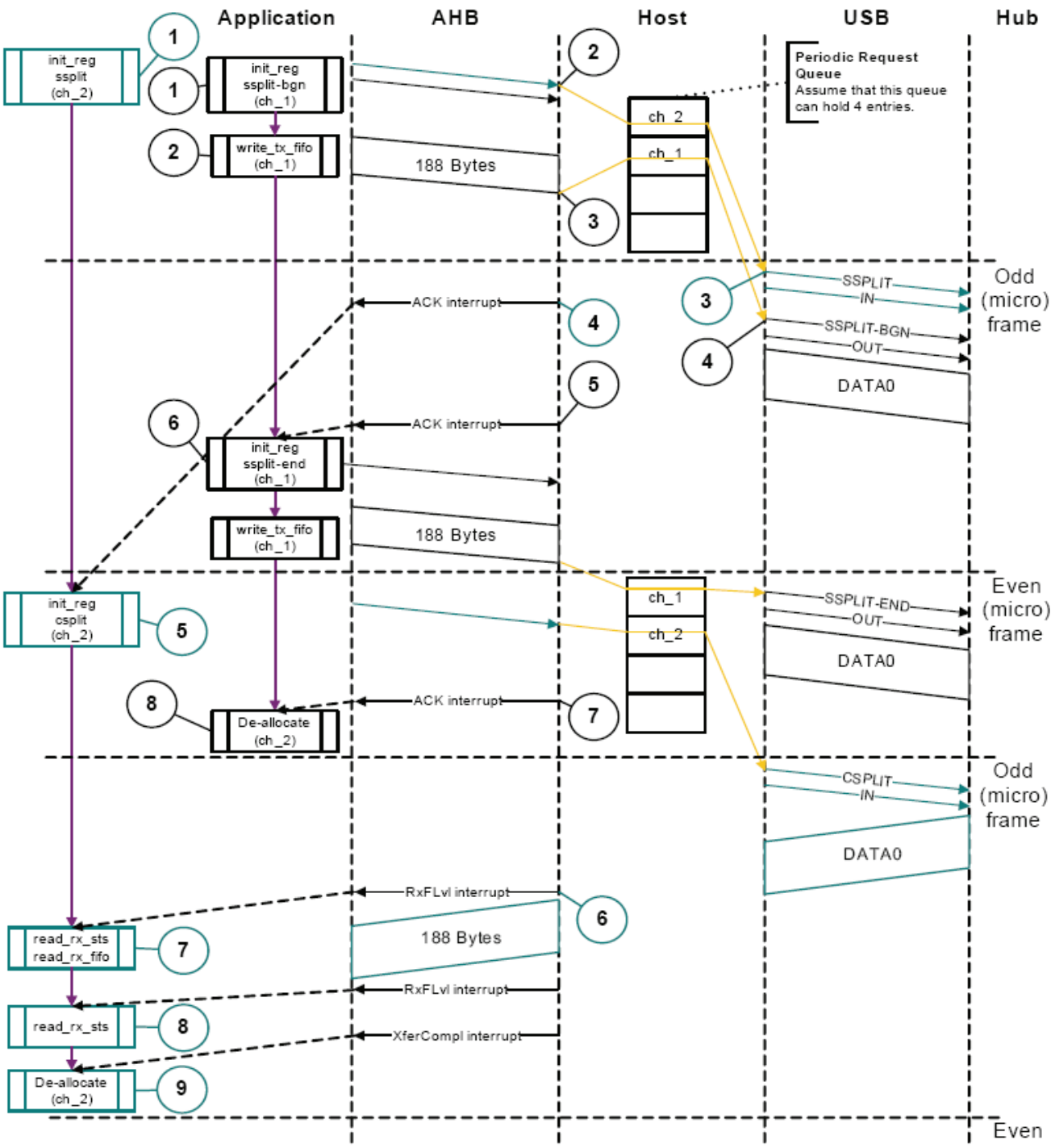


Figure 4.18 Normal Isochronous OUT/IN Split Transactions in Slave Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for isochronous OUT split transaction in Slave mode is shown in table below.

Table 4.13 Interrupt Service Routine for Isochronous OUT/IN Split Transactions in Slave Mode

	Isochronous OUT (Split)	Isochronous IN (Split)

Start Split	Unmask (XferCompl)
	if (XferCompl)
	{
	Do Next Start Split (in next b_interval - 1 uF)
	}
	else if (FrmOvrun)
	{
	Do Next Transaction in next frame.
	}
		Unmask (ACK)
		if (ACK)
		{
		Do Complete Split
		}
		else if (FrmOvrun)
		{
		Do Next Transaction in next frame.
		}

Complete Split	Not Applicable	Unmask (XactErr/NYET/STALL/XferCompl/FrmOvrun/BblErr)
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NYET)
		{
		Do Next Complete Split
		}
		else if (STALL or FrmOvrun or BblErr)
		{
		Unmask ChHltd
		Disable Channel
		}
		else if (XactErr)
		{
		Rewind Buffer Pointers
		if (HCCHARn.EC == 3) // ERR response received
		{
		Record ERR error
		Do Next Start Split (in next frame)
		}
		else
		{
		Unmask ChHltd
		Disable Channel
		}
		}
		else if (ChHltd)
		{
		Mask ChHltd
		De-allocate Channel
		}

Note

The otg host keeps track of error count in EC field for periodic split transactions. If the EC field matches the original programmed error count after XactErr interrupt, the application must treat the XactErr as “ERR response received”. The EC field indicates the number of immediate retries that the otg host has performed before generating the XactErr interrupt.

4.6.5.24 Isochronous IN Split Transactions in Slave Mode

A typical isochronous IN split operation in Slave mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one maximum-packet-size packet in an odd microframe.
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet.

[Normal Isochronous IN Split Operation]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 for start split as explained in “Channel Initialization”. The application must set the HCCHAR2.OddFrm bit.

2. The otg host writes the start split request to the Periodic Request Queue as soon as the HCCHAR2 register is written.

3. The otg host attempts to send the start split IN token in the next odd microframe.

4. As soon as the IN packet is transmitted, the otg host generates an ACK interrupt.

5. In response to the ACK interrupt, set the Do Complete Split bit (HCSPLT2.CompSplt) in the next microframe to send the complete split token in the next odd microframe.

6. As soon as the received packet is written to the receive FIFO, the otg host generates the RxFLvl interrupt.

7. In response to the RxFLvl interrupt, read the received packet status to determine the number of bytes received, then read the receive FIFO accordingly. The application must mask the RxFLvl interrupt before reading the receive FIFO and unmask it after reading the entire packet.

8. The core generates the RxFLvl interrupt for the transfer completion status entry in the receive FIFO. The application must read the receive packet status and if the receive packet status is not an IN data packet (GRXSTSR.PktSts != 4’b0010), ignore it.

9. The core generates the XferCompl interrupt as soon as the receive packet status is read. In response to the XferCompl interrupt, read the HCTSIZ2.PktCnt field. If HCTSIZ2.PktCnt != 0, disable the channel (as explained in “Halting a Channel” on page 216) before re-initializing the channel for the next transfer, if any. If HCTSIZ2.PktCnt == 0, reinitialize the channel for the next transfer. This time, the application must reset the HCCHAR2.OddFrm bit.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for isochronous IN split transaction in Slave mode is shown in the table above.

4.6.5.25 Isochronous OUT Split Transactions in DMA Mode

A typical isochronous OUT split operation in DMA mode is shown in figure below. See channel 1 (ch_1). The assumption is that the application is attempting to transmit a 376-byte packet in an odd microframe.

[Normal Isochronous OUT Split Operation]

The sequence of operations in figure below (channel 1) is as follows:

1. Initialize and enable channel 1 for start split (begin) as explained in “Channel Initialization”. The application must set the HCCHAR1.OddFrm bit. Program the MPS field with 188 bytes.

2. The otg host starts reading the packet.

3. After successfully transmitting the start split (begin), the otg host generates the ChHltd interrupt.

4. In response to the ChHltd interrupt, reinitialize the registers to send the start split (end).

5. After successfully transmitting the start split (end), the _otg host generates a ChHltd interrupt.

6. In response to the ChHltd interrupt, de-allocate the channel.

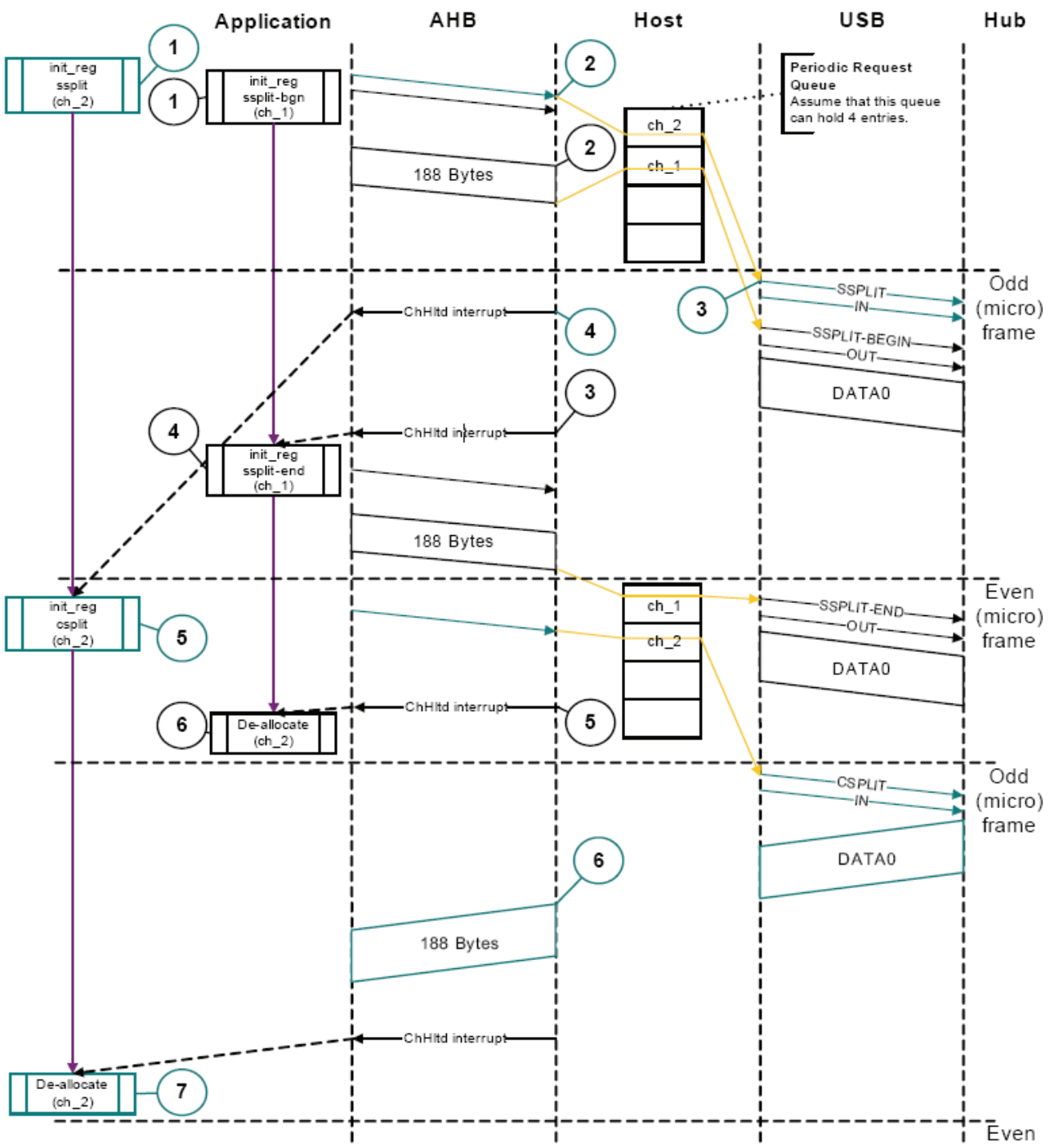


Figure 4.19 Normal Isochronous OUT/IN Split Transactions in DMA Mode

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for an isochronous OUT split transaction in DMA mode is shown in table below.

	Isochronous OUT (Split)	Isochronous IN (Split)

Start Split	Unmask (ChHltd)
	if (ChHltd)
	{
	if (ACK)
	{
	Do Next Start Split (in next b_interval – 1 uF)
	}
	else if (FrmOvrun)
	{
	Do Next Transaction in next frame.
	}
	}
		Unmask (ChHltd)
		if (ChHltd)
		{
		if (ACK)
		{
		Do Complete Split
		}
		else if (FrmOvrun)
		{
		Rewind Buffer Pointers
		Retry Start Split (in next b_interval – 1 uF)
		}
		}

Complete Split	Not Applicable	Unmask (ChHltd)
		if (ChHltd)
		{
		if (XferCompl)
		{
		De-allocate Channel
		}
		else if (NAK)
		{
		Retry Start Split (in next b_interval – 1 uF)
		}
		else if (NYET)
		{
		Do Next Complete Split
		}
		else if (STALL or FrmOvrun or BblErr)
		{
		De-allocate Channel
		}
		else if (XactErr)
		{
		Rewind Buffer Pointers
		if (HCCHARn.EC = = 3) // ERR response received
		{
		Record ERR error
		Do Next Start Split (in next frame)
		}
		else
		{
		De-allocate Channel
		}
		}
		}

Note

The otg host keeps track of error count in EC field for periodic split transactions. If the EC field matches the original programmed error count after XactErr interrupt, the application must treat the XactErr as “ERR response received”. The EC field indicates the number of immediate retries that the OTG host has performed before generating the XactErr interrupt.

4.6.5.26 Isochronous IN Split Transactions in DMA Mode

A typical isochronous IN split operation in DMA mode is shown in the figure above. See channel 2 (ch_2). The assumptions are:

- The application is attempting to receive one maximum-packet-size packet in an odd microframe.
- The receive FIFO can hold at least one maximum-packet-size packet and two status DWORDs per packet.

[Normal Isochronous IN Split Operation]

The sequence of operations in the figure above (channel 2) is as follows:

1. Initialize and enable channel 2 for start split as explained in “Channel Initialization”.

2. The otg host writes an IN request to the Request queue as soon as channel 2 receives the grant from the arbiter.

3. The otg host attempts to send the start split IN token in the beginning of the next odd microframe.

4. The otg host generates the ChHltd interrupt after successfully transmitting the start split IN token.

5. In response to the ChHltd interrupt, set the HCSPLT2.ComplSplt bit to send the complete split.

6. As soon the packet is received successfully, the otg host starts writing the data to the system memory.

7. The otg host generates the ChHltd interrupt after transferring the received data to the system memory. In response to the ChHltd interrupt, de-allocate the channel or reinitialize the channel for the next start split.

[Handling Non-ACK Responses]

The channel-specific interrupt service routine for isochronous IN split transaction in DMA mode is shown in the table above.

4.6.6 Selecting the Queue Depth

Choose the Periodic and Non-periodic Request Queue depths carefully to match the number of periodic/non-periodic endpoints accessed.

The Non-periodic Request Queue depth affects the performance of non-periodic transfers. The deeper the queue (along with sufficient FIFO size), the more often the core is able to pipeline non-periodic transfers. If the queue size is small, the core is able to put in new requests only when the queue space is freed up.

The core's Periodic Request Queue depth is critical to performing periodic transfers as scheduled. Select the periodic queue depth, based on the number of periodic transfers scheduled in a microframe. In Slave mode, however, the application must also take into account the disable entry that must be put into the queue. So, if there are two non-high bandwidth periodic endpoints, the Periodic Request Queue depth must be at least 4. If at least one high-bandwidth endpoint supported, the queue depth must be 8. If the Periodic Request Queue depth is smaller than the periodic transfers scheduled in a microframe, a frame overrun condition results.

4.6.7 Handling Babble Conditions

The otg handles two cases of babble: packet babble and port babble. Packet babble occurs if the device sends more data than the maximum packet size for the channel. Port babble occurs if the core continues to receive data from the device at EOF2 (the end of frame 2, which is very close to SOF).

When OTG detects a packet babble, it stops writing data into the Rx buffer and waits for the end of packet (EOP). When it detects an EOP, it flushes already-written data in the Rx buffer and generates a Babble interrupt to the application.

When OTG detects a port babble, it flushes the RxFIFO and disables the port. The core then generates a Port Disabled Interrupt (GINTSTS.PrtInt, HPRT.PrtEnChng). On receiving this interrupt, the application must determine that this is not due to an overcurrent condition (another cause of the Port Disabled interrupt) by checking HPRT.PrtOvrCurrAct, then perform a soft reset. The core does not send any more tokens after it has detected a port babble condition.

4.7 Device Programming Model

4.7.1 Endpoint Initialization

4.7.1.1 Initialization on USB Reset

1. Set the NAK bit for all OUT endpoints

- DOEPCTLn.SNAK = 1 (for all OUT endpoints)

2. Unmask the following interrupt bits

- DAINTMSK.INEP0 = 1 (control 0 IN endpoint)
- DAINTMSK.OUTEP0 = 1 (control 0 OUT endpoint)
- DOEPMSK.SETUP = 1
- DOEPMSK.XferCompl = 1
- DIEPMSK.XferCompl = 1
- DIEPMSK.TimeOut = 1

3. To transmit or receive data, the device must initialize more registers as specified in "Device DMA/Slave Mode Initialization".

4. Set up the Data FIFO RAM for each of the FIFOs

- Program the GRXFSIZ Register, to be able to receive control OUT data and setup data. If thresholding is not enabled, at a minimum, this must be equal to 1 max packet size of control endpoint 0 + 2 DWORDs (for the status of the control OUT data packet) + 10 DWORDs (for setup packets). If thresholding is enabled, at a minimum, this must be equal to 2* (Rx_threshold_length/4 +1)+ 2 DWORDs (for the status of the control OUT data packet) + 10 DWORDs (for setup packets)
- Program the GNPTXFSIZ Register in Shared FIFO operation or dedicated FIFO size register (depending on the FIFO number chosen) in Dedicated FIFO operation, to be able to transmit control IN data. At a minimum, this must be equal to 1 max packet size of control endpoint 0.If thresholding is enabled, this can be programmed to less than one max packet size.

5. (This step is not required if you are using Scatter/Gather DMA mode.) Program the following fields in the endpoint-specific registers for control OUT endpoint 0 to receive a SETUP packet

- DOEPTSIZ0.SetUP Count = 3 (to receive up to 3 back-to-back SETUP packets)
- In DMA mode, DOEPWMA0 register with a memory address to store any SETUP packets received

At this point, all initialization required to receive SETUP packets is done, except for enabling control OUT endpoint 0 in DMA mode.

4.7.1.2 Initialization on Enumeration Completion

1. On the Enumeration Done interrupt (GINTSTS.EnumDone, read the DSTS register to determine the enumeration speed.

2. Program the DIEPCTL0.MPS field to set the maximum packet size. This step configures control endpoint 0. The maximum packet size for a control endpoint depends on the enumeration speed.

3. In DMA mode, program the DOEPCTL0 register to enable control OUT endpoint 0, in order to receive a SETUP packet. In Scatter/Gather DMA mode, the descriptors must be set up in memory before enabling the endpoint.

- DOEPCTL0.EPEna = 1

At this point, the device is ready to receive SOF packets and is configured to perform control transfers on control endpoint 0.

4.7.1.3 Initialization on SetAddress Command

This section describes what the application must do when it receives a SetAddress command in a SETUP packet.

1. Program the DCFG register with the device address received in the SetAddress command

2. Program the core to send out a status IN packet.

4.7.1.4 Initialization on SetConfiguration/SetInterface Command

This section describes what the application must do when it receives a SetConfiguration or SetInterface command in a SETUP packet.

1. When a SetConfiguration command is received, the application must program the endpoint registers to configure them with the characteristics of the valid endpoints in the new configuration.

2. When a SetInterface command is received, the application must program the endpoint registers of the endpoints affected by this command.

3. Some endpoints that were active in the prior configuration or alternate setting are not valid in the new configuration or alternate setting. These invalid endpoints must be deactivated.

4. For details on a particular endpoint's activation or deactivation, see "Endpoint Activation" and "Endpoint Deactivation".

5. Unmask the interrupt for each active endpoint and mask the interrupts for all inactive endpoints in the DAINTMSK register.

6. Set up the Data FIFO RAM for each FIFO (only if Dynamic FIFO Sizing is enabled). See "Data FIFO RAM Allocation" for more detail.

7. After all required endpoints are configured, the application must program the core to send a status IN packet.
At this point, the device core is configured to receive and transmit any type of data packet.

4.7.1.5 Endpoint Activation

This section describes the steps required to activate a device endpoint or to configure an existing device endpoint to a new type.

1. Program the characteristics of the required endpoint into the following fields of theDIEPCTLn register (for IN or bidirectional endpoints) or the DOEPCTLn register (for OUT or bidirectional endpoints).

- Maximum Packet Size
- USB Active Endpoint = 1
- Endpoint Start Data Toggle (for interrupt and bulk endpoints)
- Endpoint Type
- TxFIFO Number1

2. Once the endpoint is activated, the core starts decoding the tokens addressed to that endpoint and sends out a valid handshake for each valid token received for the endpoint.

4.7.1.6 Endpoint Deactivation

This section describes the steps required to deactivate an existing endpoint.

1. In the endpoint to be deactivated, clear the USB Active Endpoint bit in the DIEPCTLn register (for IN or bidirectional endpoints) or the DOEPCTLn register (for OUT or bidirectional endpoints).

2. Once the endpoint is deactivated, the core ignores tokens addressed to that endpoint, resulting in a timeout on the USB.

4.7.1.7 Device DMA/Slave Mode Initialization

The application must meed the following conditions to set up the device core to handle traffic.

- In Slave mode, GINTMSK.NPTxFEmpMsk, and GINTMSK.RxFLvlMsk must be unset.
- In DMA mode, the aforementioned interrupts must be masked.

4.7.2 Operational Model

4.7.2.1 SETUP and OUT Data Transfers

This section describes the internal data flow and application-level operations during data OUT transfers and SETUP transactions.

[Packet Read in Slave Mode]

This section describes how to read packets (OUT data and SETUP packets) from the receive FIFO in Slave mode.

1. On catching a GINTSTS.RxFLvl interrupt, the application must read the Receive Status Pop register (GRXSTSP).

2. The application can mask the GINTSTS.RxFLvl interrupt by writing to GINTMSK.RxFLvl = 1'b0, until it has read the packet from the receive FIFO.

3. If the received packet's byte count is not 0, the byte count amount of data is popped from the receive Data FIFO and stored in memory. If the received packet byte count is 0, no data is popped from the Receive Data FIFO.

4. The receive FIFO's packet status readout indicates one of the following.

- Global OUT NAK Pattern: PktSts = Global OUT NAK, BCnt = 11'h000, EPNum = Dont Care (4'h0), DPID = Dont Care (2'b00). This data indicates that the global OUT NAK bit has taken effect.
- SETUP Packet Pattern: PktSts = SETUP, BCnt = 11'h008, EPNum = Control EPNum, DPID = D0. This data indicates that a SETUP packet for the specified endpoint is now available for reading from the receive FIFO.
- Setup Stage Done Pattern: PktSts = Setup Stage Done, BCnt = 11'h0, EPNum = Control EP Num, DPID = Don't Care (2'b00). This data indicates that the Setup stage for the specified endpoint has completed and the Data stage has started. After this entry is popped from the receive FIFO, the core asserts a Setup interrupt on the specified control OUT endpoint.
- Data OUT Packet Pattern: PktSts = DataOUT, BCnt = size of the Received data OUT packet ($0 \leq$ BCnt $\leq 1,024$), EPNum = EPNum on which the packet was received, DPID = Actual Data PID.
- Data Transfer Completed Pattern: PktSts = Data OUT Transfer Done, BCnt = 11'h0, EPNum = OUT EP Num on which the data transfer is complete, DPID = Dont Care (2'b00). This data indicates that a OUT data transfer for the specified OUT endpoint has completed. After this entry is popped from the receive FIFO, the core asserts a Transfer Completed interrupt on the specified OUT endpoint.

The encoding for the PktSts is listed in "Receive Status Debug Read/Status Read and Pop Registers (GRXSTSR/GRXSTSP)".

5. After the data payload is popped from the receive FIFO, the GINTSTS.RxFLvl interrupt must be unmasked.

6. Steps 1–5 are repeated every time the application detects assertion of the interrupt line due to GINTSTS.RxFLvl. Reading an empty receive FIFO can result in undefined core behavior.

Below figure provides a flow chart of the above procedure.

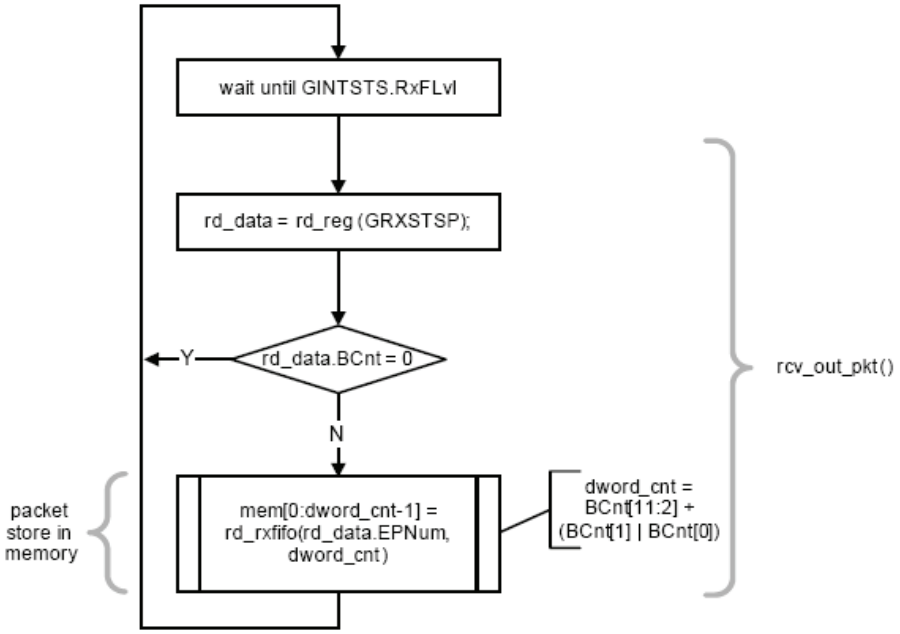


Figure 4.20 Receive FIFO Packet Read in Slave Mode

[SETUP Transactions]

This section describes how the core handles SETUP packets and the application's sequence for handling SETUP transactions.

Application Requirements

1. To receive a SETUP packet, the DOEPTSIZn.SUPCnt field in a control OUT endpoint must be programmed to a non-zero value. When the application programs the SUPCnt field to a non-zero value, the core receives SETUP packets and writes them to the receive FIFO, irrespective of the DOEPCTLn.NAK status and DOEPCTLn.EPEna bit setting. The SUPCnt field is decremented every time the control endpoint receives a SETUP packet. If the SUPCnt field is not programmed to a proper value before receiving a SETUP packet, the core still receives the SETUP packet and decrements the SUPCnt field, but the application possibly is not be able to determine the correct number of SETUP packets received in the Setup stage of a control transfer.

- DOEPTSIZn.SUPCnt = 3

2. In DMA mode, the OUT endpoint must also be enabled, to transfer the received SETUP packet data from the internal receive FIFO to the external memory.

- DOEPCTLn.EPEna = 1'b1

3. The application must always allocate some extra space in the Receive Data FIFO, to be able to receive up to three SETUP packets on a control endpoint.

- The space to be Reserved is (4 * n) + 6 DWORDs, where n is the number of control endpoints supported by the device. Three DWORDs are required for the first SETUP packet, 1 DWORD is required for the Setup Stage Done DWORD, and 6 DWORDs are required to store two extra SETUP packets among all control endpoints.
- 3 DWORDs per SETUP packet are required to store 8 bytes of SETUP data and 4 bytes of SETUP status (Setup Packet Pattern). The core reserves this space in the receive data.
- FIFO to write SETUP data only, and never uses this space for data packets.

4. In Slave mode, the application must read the 2 DWORDs of the SETUP packet from the receive FIFO. In DMA mode, the core writes the 2 DWORDs of SETUP data to the memory.

5. The application must read and discard the Setup Stage Done DWORD from the receive FIFO.

Internal Data Flow

1. When a SETUP packet is received, the core writes the received data to the receive FIFO, without checking for available space in the receive FIFO and irrespective of the endpoint's NAK and Stall bit settings.

- The core internally sets the IN NAK and OUT NAK bits for the control IN/OUT endpoints on which the SETUP packet was received.

2. For every SETUP packet received on the USB, 3 DWORDs of data is written to the receive FIFO, and the SUPCnt field is decremented by 1.

- The first DWORD contains control information used internally by the core
- The second DWORD contains the first 4 bytes of the SETUP command
- The third DWORD contains the last 4 bytes of the SETUP command

3. When the Setup stage changes to a Data IN/OUT stage, the core writes an entry (Setup Stage Done DWORD) to the receive FIFO, indicating the completion of the Setup stage.

4. On the AHB side, SETUP packets are emptied either by the DMA or the application. In DMA mode, the SETUP packets (2 DWORDs) are written to the memory location programmed in the DOEPWMAn register, only if the endpoint is enabled. If the endpoint is not enabled, the data remains in the receive FIFO until the enable bit is set.

5. When either the DMA or the application pops the Setup Stage Done DWORD from the receive FIFO, the core interrupts the application with a DOEPINTn.SETUP interrupt, indicating it can process the received SETUP packet.

- The core clears the endpoint enable bit for control OUT endpoints.

Application Programming Sequence

1. Program the DOEPTSIZn register.

- DOEPTSIZn.SUPCnt = 3

2. In DMA mode, program the DOEPWMAn register and DOEPCTLn register with the endpoint characteristics and set the Endpoint Enable bit (DOEPCTLn.EPEna).

- Endpoint Enable = 1

3. In Slave mode, wait for the GINTSTS.RxFLvl interrupt and empty the data packets from the receive FIFO, as explained in "Packet Read in Slave Mode". This step can be repeated many times.

4. Assertion of the DOEPINTn.SETUP interrupt marks a successful completion of the SETUP Data Transfer.

- On this interrupt, the application must read the DOEPTSIZn register to determine the number of SETUP packets received and process the last received SETUP packet.
- In DMA mode, the application must also determine if the interrupt bit DOEPINTn.Back2BackSETup is set. This bit is set if the core has received more than three back-to-back SETUP packets. If this is the case, the application must ignore the DOEPTSIZn.SUPCnt value and use the DOEPWMAn directly to read out the last SETUP packet received. DOEPWMAn-8 provides the pointer to the last valid SETUP data.

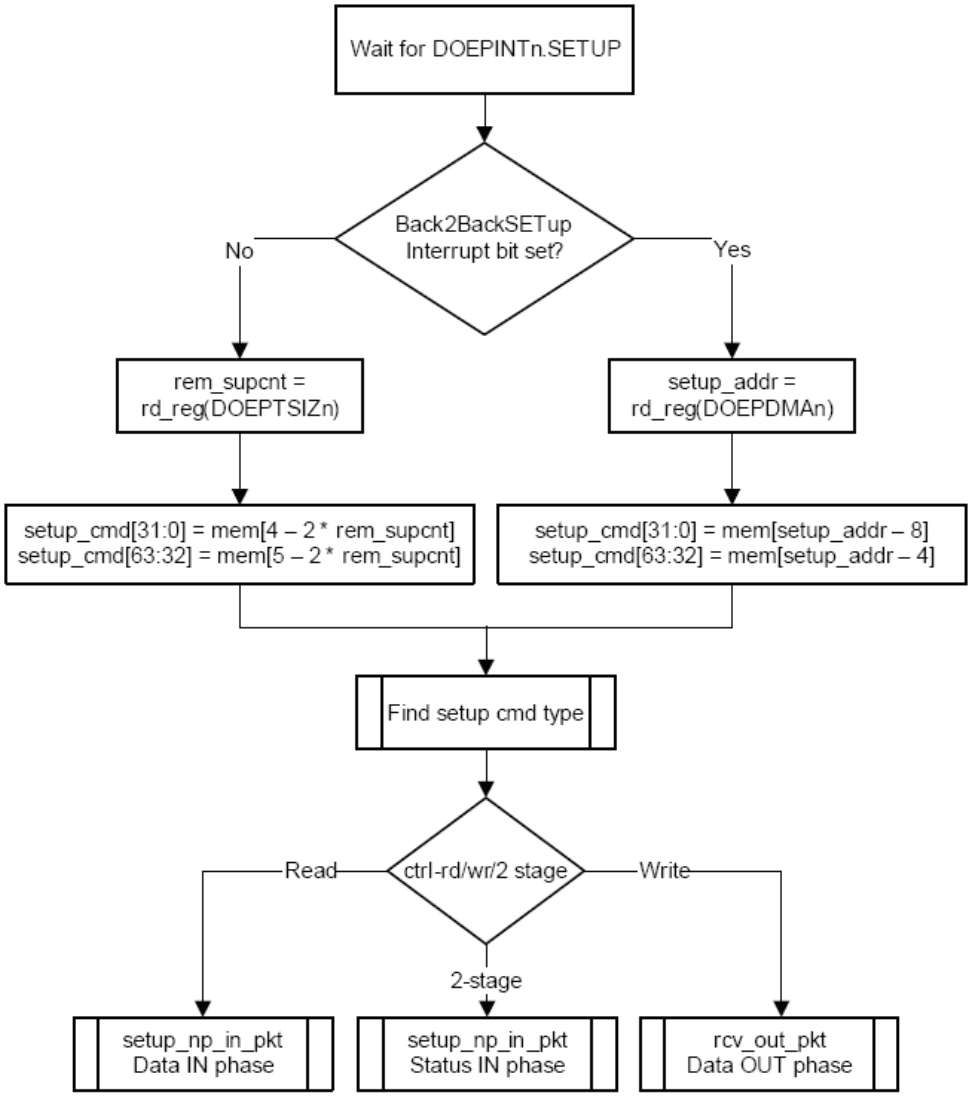


Figure 4.21 Processing a SETUP Packet

[Handling More Than Three Back-to-Back SETUP Packets]

Per the USB 2.0 specification, normally, during a SETUP packet error, a host does not send more than three back-to-back SETUP packets to the same endpoint. However, the USB 2.0 specification does not limit the number of back-to-back SETUP packets a host can send to the same endpoint. When this condition occurs, the OTG core generates an interrupt (DOEPINTn.Back2BackSETup). In DMA mode, the core also rewinds the DMA address for that endpoint (DOEPWMAn) and overwrites the first SETUP packet in system memory with the fourth, second with the fifth, and so on. If the Back2BackSETup interrupt is asserted, the application must read the OUT endpoint DMA register (DOEPWMAn) to determine the final SETUP data in system memory.

In DMA mode, the application can mask the Back2BackSETup interrupt, but after receiving the DOEPINT.SETUP interrupt, the application can read the DOEPINT.Back2BackSETup interrupt bit. In Slave mode, the application can use the GINTSTS.RxFLvl interrupt to read out the SETUP packets from the FIFO whenever the core receives the SETUP packet.

[Setting the Global OUT NAK]

Internal Data Flow

1. When the application sets the Global OUT NAK (DCTL.SGOUTNak), the core stops writing data, except SETUP packets, to the receive FIFO. Irrespective of the space availability in the receive FIFO, non-isochronous OUT tokens receive a NAK handshake response, and the core ignores isochronous OUT data packets

2. The core writes the Global OUT NAK pattern to the receive FIFO. The application must reserve enough receive FIFO space to write this data pattern. See “Data FIFO RAM Allocation”.

3. When either the core (in DMA mode) or the application (in Slave mode) pops the Global OUT NAK pattern DWORD from the receive FIFO, the core sets the GINTSTS.GOUTNakEff interrupt.

4. Once the application detects this interrupt, it can assume that the core is in Global OUT NAK mode. The application can clear this interrupt by clearing the DCTL.SGOUTNak bit.

Application Programming Sequence

1. To stop receiving any kind of data in the receive FIFO, the application must set the Global OUT NAK bit by programming the following field.

- DCTL.SGOUTNak = 1’b1

2. Wait for the assertion of the interrupt GINTSTS.GOUTNakEff. When asserted, this interrupt indicates that the core has stopped receiving any type of data except SETUP packets.

3. The application can receive valid OUT packets after it has set DCTL.SGOUTNak and before the core asserts the GINTSTS.GOUTNakEff interrupt.

4. The application can temporarily mask this interrupt by writing to the GINTMSK.GINNakEffMsk bit.

- GINTMSK.GINNakEffMsk = 1’b0

5. Whenever the application is ready to exit the Global OUT NAK mode, it must clear the DCTL.SGOUTNak bit. This also clears the GINTSTS.GOUTNakEff interrupt.

- DCTL.CGOUTNak = 1’b1

6. If the application has masked this interrupt earlier, it must be unmasked as follows:

- GINTMSK.GINNakEffMsk = 1’b1

[Disabling an OUT Endpoint]

The application must use this sequence to disable an OUT endpoint that it has enabled.

Application Programming Sequence

1. Before disabling any OUT endpoint, the application must enable Global OUT NAK mode in the core, as described in “Setting the Global OUT NAK”.

- DCTL.DCTL.SGOUTNak = 1’b1

2. Wait for the GINTSTS.GOUTNakEff interrupt

3. Disable the required OUT endpoint by programming the following fields.

- DOEPCTLn.EPDisable = 1’b1
- DOEPCTLn.SNAK = 1’b1

4. Wait for the DOEPINTn.EPDisabled interrupt, which indicates that the OUT endpoint is completely disabled. When the EPDisabled interrupt is asserted, the core also clears the following bits.

- DOEPCTLn.EPDisable = 1’b0
- DOEPCTLn.EPEnable = 1’b0

5. The application must clear the Global OUT NAK bit to start receiving data from other non-disabled OUT endpoints.

- DCTL.SGOUTNak = 1’b0

[Generic Non-Isochronous OUT Data Transfers without Thresholding]

This section describes a regular non-isochronous OUT data transfer (control, bulk, or interrupt).

Application Requirements

1. Before setting up an OUT transfer, the application must allocate a buffer in the memory to accommodate all data to be received as part of the OUT transfer, then program that buffer's size and start address (in DMA mode) in the endpoint-specific registers.

2. For OUT transfers, the Transfer Size field in the endpoint's Transfer Size register must be a multiple of the maximum packet size of the endpoint, adjusted to the DWORD boundary.

- transfer size[epnum] = n * (mps[epnum] + 4 – (mps[epnum] mod 4))
- packet count[epnum] = n
- n > 0

3. In DMA mode, the core stores a received data packet in the memory, always starting on a DWORD boundary. If the maximum packet size of the endpoint is not a multiple of 4, the core inserts byte pads at end of a maximum-packet-size packet up to the end of the DWORD.

4. On any OUT endpoint interrupt, the application must read the endpoint's Transfer Size register to calculate the size of the payload in the memory. The received payload size can be less than the programmed transfer size.

- Payload size in memory = application-programmed initial transfer size – core updated final transfer size
- Number of USB packets in which this payload was received = applicationprogrammed initial packet count – core updated final packet count

Internal Data Flow

1. The application must set the Transfer Size and Packet Count fields in the endpointspecific registers, clear the NAK bit, and enable the endpoint to receive the data.

2. Once the NAK bit is cleared, the core starts receiving data and writes it to the receive FIFO, as long as there is space in the receive FIFO. For every data packet received on the USB, the data packet and its status are written to the receive FIFO. Every packet (maximum packet size or short packet) written to the receive FIFO decrements the Packet Count field for that endpoint by 1.

- OUT data packets received with Bad Data CRC are flushed from the receive FIFO automatically.
- After sending an ACK for the packet on the USB, the core discards non-isochronous OUT data packets that the host, which cannot detect the ACK, re-sends. The application does not detect multiple back-to-back data OUT packets on the same endpoint with the same data PID. In this case the packet count is not decremented.
- If there is no space in the receive FIFO, isochronous or non-isochronous data packets are ignored and not written to the receive FIFO. Additionally, non-isochronous OUT tokens receive a NAK handshake reply.
- In all the above three cases, the packet count is not decremented because no data is written to the receive FIFO.

3. When the packet count becomes 0 or when a short packet is received on the endpoint, the NAK bit for that endpoint is set. Once the NAK bit is set, the isochronous or nonisochronous data packets are ignored and not written to the receive FIFO, and nonisochronous OUT tokens receive a NAK handshake reply.

4. After the data is written to the receive FIFO, either the application (in Slave mode) or the core's DMA engine (in External or Internal DMA mode), reads the data from the receive FIFO and writes it to external memory, one packet at a time per endpoint.

5. At the end of every packet write on the AHB to external memory, the transfer size for the endpoint is decremented by the size of the written packet.

6. The OUT Data Transfer Completed pattern for an OUT endpoint is written to the receive FIFO on one of the following conditions.

- The transfer size is 0 and the packet count is 0
- The last OUT data packet written to the receive FIFO is a short packet (0≤packet size< maximum packet size)

7. When either the application or the DMA pops this entry (OUT Data Transfer Completed),

- Transfer Completed interrupt is generated for the endpoint and the endpoint enable is cleared.

Application Programming Sequence

1. Program the DOEPTSIZn register for the transfer size and the corresponding packet count. Additionally, in DMA mode, program the DOEPWMAn register.

2. Program the DOEPCTLn register with the endpoint characteristics, and set the Endpoint Enable and ClearNAK bits.

- DOEPCTLn.EPEna = 1
- DOEPCTLn.CNAK = 1

3. In Slave mode, wait for the GINTSTS.Rx StsQ level interrupt and empty the data packets from the receive FIFO as explained in “Packet Read in Slave Mode”.

- This step can be repeated many times, depending on the transfer size.

4. Asserting the DOEPINTn.XferCompl interrupt marks a successful completion of the nonisochronous OUT data transfer.

5. Read the DOEPTSIZn register to determine the size of the received data payload.

[Generic non-Isochronous OUT Data Transfer with Thresholding]

This section describes a regular non-ISO OUT data transfer (Control/Bulk/Intr) when thresholding is enabled.

Application Requirements

Application requirements is the same as without thresholding.

Internal Data Flow

1. The application should set the transfer size and packet count fields in the endpoint specific registers, clear the NAK bit and enable the endpoint to receive the data.

2. Once the NAK bit is cleared, the core starts receiving the data and writes it into the receive FIFO, as long as there is threshold amount of space in the receive FIFO. For every threshold amount of data received on the USB, the data packet and the threshold status are written into the receive FIFO. At the end of a packet, a last threshold status and also a data update status is written into the receive FIFO.If it was the last packet of the transfer, then a transfer complete status is also written into the receive FIFO.On every packet (mps sized or short packet) written into the receive FIFO, the packet count field for that endpoint is decremented by 1.

- OUT data packets received with Bad Data CRC are flushed out of the receive FIFO automatically.The core also rewind the DMA pointers internally.
- non-ISO OUT data packet re-sent by the host, because the ACK was not seen by the host, will be discarded by the core, after sending an ACK for the packet on the USB. The application will not see multiple back to back data OUT packets on the same endpoint, with the same data PID. In this case the packet count is not decremented.
- If there is no space for at least threshold amount of data in the receive FIFO, the ISO/non-ISO data packets are ignored and not written into the receive FIFO. In addition, the non-ISO OUT tokens are responded with NAK handshake.

- If the core sees an overflow case (no space in the fifo in the middle of a packet reception), then the core stops writing the remaining data into the fifo and sends a NAK handshake on the USB. The core rewinds the fifo pointer to the threshold boundary, so that the portion of the threshold data that is in the fifo is flushed out.The core also rewinds the DMA pointers.The core also sets DOEPINTn.OutPktErr (This interrupt bit is mainly used for debug purpose).

In all the above cases, the packet count is not decremented because no data is written into the receive FIFO.

In High Speed, after the core has received a packet, the core sends a NYET handshake if the core does not find threshold amount of free space available in the FIFO.

3. When the packet count becomes 0 or when a short packet is received on the endpoint, the NAK bit for that endpoint is set. Once the NAK bit is set, the ISO/non-ISO data packets are ignored and not written into the receive FIFO and the non-ISO OUT tokens are responded with a NAK handshake.

4. The DMA engine will transfer data from the receive FIFO to the system memory as soon as it sees one threshold amount of data in the FIFO.

5. At the end of every packet write on the AHB into the external memory, the transfer size for the endpoint is decremented by the size of packet written into the memory.

6. The OUT Data Transfer Complete pattern for an OUT endpoint is written into the receive FIFO, on one of the following conditions.

- The last threshold is written into FIFO and the packet count is decremented to 0
- If it is a short packet and the core sees the end of packet within a threshold.

7. When this entry (OUT Data Transfer Complete) is popped out by the DMA engine, Transfer Complete interrupt for the endpoint is generated and the endpoint enable is cleared.

8. “Rewind OUT Data Transfer” pattern is written into the receive FIFO, on one of the following conditions

- On seeing Overflow condition.
- On seeing a CRC error.

9. When this entry (Rewind OUT Data Transfer) is popped out by the DMA engine, it does the DMA pointer rewind.

Application Programming Sequence

This sequence is the same as in non thresholding case.

[Generic Isochronous OUT Data Transfer without Thresholding]

This section describes a regular isochronous OUT data transfer.

Application Requirements

1. All the application requirements for non-isochronous OUT data transfers also apply to isochronous OUT data transfers

2. For isochronous OUT data transfers, the Transfer Size and Packet Count fields must always be set to the number of maximum-packet-size packets that can be received in a single microframe and no more. Isochronous OUT data transfers cannot span more than 1 microframe.

- $1 \leq$ packet count[epnum] ≤ 3

3. In Slave mode, when isochronous OUT endpoints are supported in the device, the application must read all isochronous OUT data packets from the receive FIFO (data and status) before the end of the periodic frame (GINTSTS.EOPF interrupt). In DMA mode, the application must guarantee enough bandwidth to allow emptying the isochronous OUT data packet from the receive FIFO before the end of each periodic frame.

4. To receive data in the following frame/microframe, an isochronous OUT endpoint must be enabled after the GINTSTS.EOPF and before the GINTSTS.SOF.

Internal Data Flow

1. The internal data flow for isochronous OUT endpoints is the same as that for nonisochronous OUT endpoints, but for a few differences.

2. When an isochronous OUT endpoint is enabled by setting the Endpoint Enable and clearing the NAK bits, the Even/Odd frame/microframe bit must also be set appropriately. The core receives data on a isochronous OUT endpoint in a particular microframe only if the following condition is met.

- DOEPCTLn.Even/Odd microframe = DSTS.SOFFN[0]

3. When either the application or the external/internal DMA completely reads an isochronous OUT data packet (data and status) from the receive FIFO, the core updates the DOEPTSIZn.Received DPID field with the data PID of the last isochronous OUT data packet read from the receive FIFO.

Application Programming Sequence

1. Program the DOEPTSIZn register for the transfer size and the corresponding packet count. When in DMA mode, also program the DOEPWMAn register.

2. Program the DOEPCTLn register with the endpoint characteristics and set the Endpoint Enable, ClearNAK, and Even/Odd frame/microframe bits.

- Endpoint Enable = 1
- CNAK = 1
- Even/Odd frame/microframe = (0: Even/1: Odd)

3. In Slave mode, wait for the GINTSTS.Rx StsQ level interrupt and empty the data packets from the receive FIFO as explained in “Packet Read in Slave Mode”.

- This step can be repeated many times, depending on the transfer size.

4. The assertion of the DOEPINTn.XferCompl interrupt marks the completion of the isochronous OUT data transfer. This interrupt does not necessarily mean that the data in memory is good.

- This interrupt can not always be detected for isochronous OUT transfers. Instead, the application can detect the GINTSTS.incomplete Isochronous OUT data interrupt. See “Incomplete Isochronous OUT Data Transfers”, for more details

5. Read the DOEPTSIZn register to determine the size of the received transfer and to determine the validity of the data received in the microframe. The application must treat the data received in memory as valid only if one of the following conditions is met.

- DOEPTSIZn.RxDPID = D0 and the number of USB packets in which this payload was received = 1
- DOEPTSIZn.RxDPID = D1 and the number of USB packets in which this payload was received = 2
- DOEPTSIZn.RxDPID = D2 and the number of USB packets in which this payload was received = 3
  • The number of USB packets in which this payload was received = App Programmed Initial Packet Count – Core Updated Final Packet Count

The application can discard invalid data packets.

[Generic Isochronous OUT Data Transfer with Thresholding]

This section describes a regular isochronous OUT data transfer.

Application Requirements

There is no change in this section from a non-thresholding mode.

Internal Data Flow

1. The internal data flow for isochronous OUT Endpoints when thresholding is enabled, is the same as that for the non isochronous OUT endpoints when thresholding is enabled, but for a few differences.

2. If MAC sees an overflow condition when writing a packet, it stops writing.The current threshold amount of data that is being written into the receive FIFO is flushed out at the end of packet.This will eventually result in GINTSTS.incomplete ISO OUT Data interrupt. Refer to the section Incomplete isochronous OUT Data Transfers, for more details.

3. If MAC sees a CRC error for the receiving packet, the last threshold being written into the receive FIFO is flushed out. This will eventually result in GINTSTS.incomplete isochronous OUT Data interrupt. Refer to the section Incomplete isochronous OUT Data Transfers, for more details.

4. Assertion of DOEPINTn.XferCompl interrupt marks a completion of the isochronous OUT Data Transfer. This interrupt may not necessarily mean that data in the memory is good data.

5. Read the DOEPTSIZn register, to find out the size of the received transfer and to find out the validity of the data received in the microframe. The application should treat the data received into the memory as valid only if one of the following conditions is met. Invalid data packets may be discarded by the application.

- DOEPTSIZn.RxDPID = D0 and Number of USB Packets in which this payload was received = 1
- DOEPTSIZn.RxDPID = D1 and Number of USB Packets in which this payload was received = 2
- DOEPTSIZn.RxDPID = D2 and Number of USB Packets in which this payload was received = 3

• Number of USB Packets in which this payload was received = App Programmed Initial Packet Count - Core Updated Final Packet Count

[Incomplete Isochronous OUT Data Transfers]

This section describes the application programming sequence when isochronous OUT data packets are dropped inside the core.

Internal Data Flow

1. For isochronous OUT endpoints, the DOEPINTn.XferCompl interrupt possibly is not always asserted. If the core drops isochronous OUT data packets, the application could fail to detect the DOEPINTn.XferCompl interrupt under the following circumstances.

- When the receive FIFO cannot accommodate the complete ISO OUT data packet, the core drops the received ISO OUT data. In thresholding this is same as overflow.
- When the isochronous OUT data packet is received with CRC errors
- When the isochronous OUT token received by the core is corrupted
- When the application is very slow in reading the data from the receive FIFO

2. When the core detects an end of periodic frame before transfer completion to all isochronous OUT endpoints, it asserts the GINTSTS.incomplete Isochronous OUT data interrupt, indicating that a DOEPINTn.XferCompl interrupt is not asserted on at least one of the isochronous OUT endpoints. At this point, the endpoint with the incomplete transfer remains enabled, but no active transfers remains in progress on this endpoint on the USB.

Application Programming Sequence

1. Asserting the GINTSTS.incomplete Isochronous OUT data interrupt indicates that in the current microframe, at least one isochronous OUT endpoint has an incomplete transfer.

- If this occurs because isochronous OUT data is not completely emptied from the endpoint, the application must ensure that the DMA or the application empties all isochronous OUT data (data and status) from the receive FIFO before proceeding.

- When all data is emptied from the receive FIFO, the application can detect the DOEPINTn.XferCompl interrupt. In this case, the application must re-enable the endpoint to receive isochronous OUT data in the next microframe, as described in "Generic Isochronous OUT Data Transfer without Thresholding".

2. When it receives a GINTSTS.incomplete Isochronous OUT data interrupt, the application must read the control registers of all isochronous OUT endpoints (DOEPCTLn) to determine which endpoints had an incomplete transfer in the current microframe. An endpoint transfer is incomplete if both the following conditions are met.

- DOEPCTLn.Even/Odd microframe bit = DSTS.SOFFN[0]
- DOEPCTLn.Endpoint Enable = 1

3. The previous step must be performed before the GINTSTS.SOF interrupt is detected, to ensure that the current microframe number is not changed.

4. For isochronous OUT endpoints with incomplete transfers, the application must discard the data in the memory and disable the endpoint by setting the DOEPCTLn.Endpoint Disable bit.

5. Wait for the DOEPINTn.Endpoint Disabled interrupt and enable the endpoint to receive new data in the next microframe as explained in "Generic Isochronous OUT Data Transfer without Thresholding".

- Because the core can take some time to disable the endpoint, the application possibly is not able to receive the data in the next microframe after receiving bad isochronous data.

[Stalling a Non-Isochronous OUT Endpoint]

This section describes how the application can stall a non-isochronous endpoint.

1. Put the core in the Global OUT NAK mode, as described in "Setting the Global OUT NAK".

2. Disable the required endpoint, as described in "Disabling an OUT Endpoint".

- When disabling the endpoint, instead of setting the DOEPCTL.SNAK bit, set DOEPCTL.STALL = 1.
  - The Stall bit always takes precedence over the NAK bit.

3. When the application is ready to end the STALL handshake for the endpoint, the DOEPCTLn.STALL bit must be cleared.

4. If the application is setting or clearing a STALL for an endpoint due to a SetFeature.Endpoint Halt or ClearFeature.Endpoint Halt command, the Stall bit must be set or cleared before the application sets up the Status stage transfer on the control endpoint.

[Examples]

This section describes and depicts some fundamental transfer types and scenarios.

Slave Mode Bulk OUT Transaction

Figure below depicts the reception of a single Bulk OUT Data packet from the USB to the AHB and describes the events involved in the process.

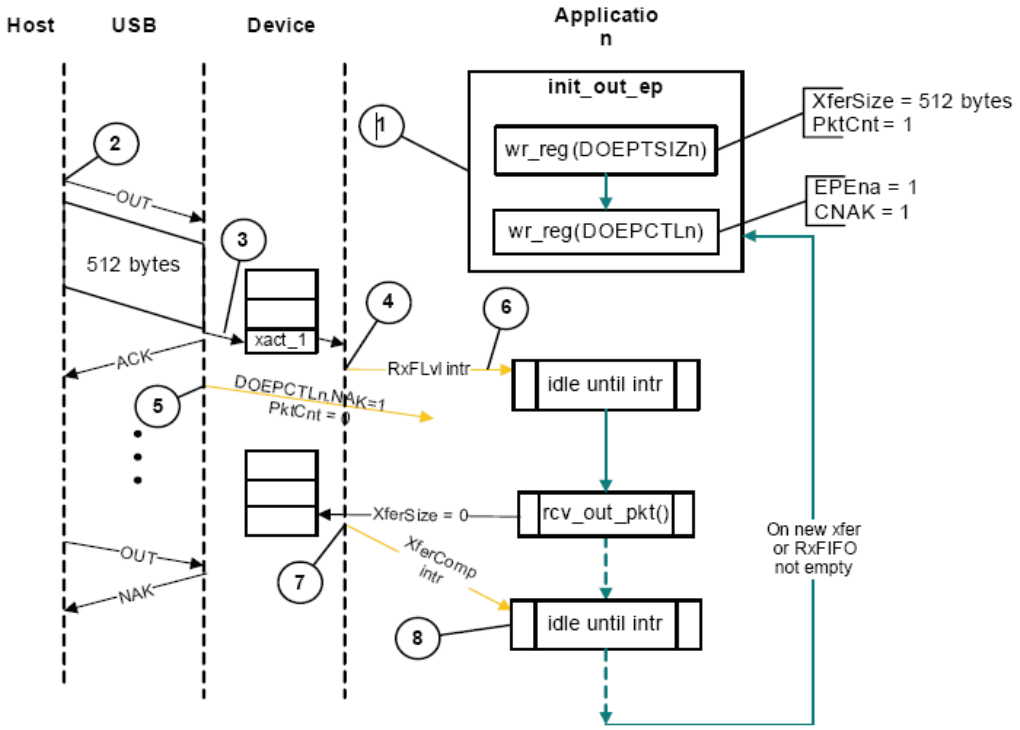


Figure 4.22 Slave Mode Bulk OUT Transaction

1. After a SetConfiguration/SetInterface command, the application initializes all OUT endpoints by setting DOEPCTLn.CNAK = 1 and DOEPCTLn.EPEna = 1, and setting a suitable XferSize and PktCnt in the DOEPTSIZn register.

2. Host attempts to send data (OUT token) to an endpoint.

3. When the core receives the OUT token on the USB, it stores the packet in the RxFIFO because space is available there.

4. After writing the complete packet in the RxFIFO, the core then asserts the GINTSTS.RxFLvl interrupt.

5. On receiving the PktCnt number of USB packets, the core sets the NAK bit for this endpoint internally to prevent it from receiving any more packets.

6. The application processes the interrupt and reads the data from the RxFIFO.

7. When the application has read all the data (equivalent to XferSize), the core generates a DOEPINTn.XferCompl interrupt.

8. The application processes the interrupt and uses the setting of the DOEPINTn.XferCompl interrupt bit to determine that the intended transfer is complete.

4.7.2.2 IN Data Transfers

This section describes IN data transfer details.

[Packet Write in Slave Mode]

This section describes how the application writes data packets to the endpoint FIFO in Slave mode.

1. The application can either choose polling or interrupt mode.

- In polling mode, application monitors the status of the endpoint transmit data FIFO, by reading the DTXFSTSn register, to determine, if there is enough space in the data FIFO.
- In interrupt mode, application waits for the DIEPINTn.TxFEmp interrupt and then reads the DTXFSTSn register, to determine, if there is enough space in the data FIFO.
- To write a single non-zero length data packet, there must be space to write the entire packet is the data FIFO.
- For writing zero length packet, application must not look for FIFO space.

2. Using one of the above mentioned methods, when the application determines that there is enough space to write a transmit packet, the application must first write into the endpoint control register, before writing the data into the data FIFO. The application, typically must do a read modify write on the DIEPCTLn, to avoid modifying the contents of the register, except for setting the Endpoint Enable bit.

The application can write multiple packets for the same endpoint, into the transmit FIFO, if space is available. For periodic IN endpoints, application must write packets only for one microframe. It can write packets for the next periodic transaction, only after getting transfer complete for the previous transaction.

[Setting IN Endpoint NAK]

Internal Data Flow

1. When the application sets the IN NAK for a particular endpoint, the core stops transmitting data on the endpoint, irrespective of data availability in the endpoint's transmit FIFO.

- Non-isochronous IN tokens receive a NAK handshake reply
- Isochronous IN tokens receive a zero-data-length packet reply

2. The core asserts the DIEPINTn.IN NAK Effective interrupt in response to the DIEPCTL.Set NAK bit.

3. Once this interrupt is seen by the application, the application can assume that the endpoint is in IN NAK mode. This interrupt can be cleared by the application by setting the DIEPCTLn. Clear NAK bit.

Application Programming Sequence

1. To stop transmitting any data on a particular IN endpoint, the application must set the IN NAK bit. To set this bit, the following field must be programmed.

- DIEPCTLn.SetNAK = 1'b1

2. Wait for assertion of the DIEPINTn.NAK Effective interrupt. This interrupt indicates the core has stopped transmitting data on the endpoint.

3. The core can transmit valid IN data on the endpoint after the application has set the NAK bit, but before the assertion of the NAK Effective interrupt.

4. The application can mask this interrupt temporarily by writing to the DIEPMSK.NAK Effective bit.

- DIEPMSK.NAK Effective = 1'b0

5. To exit Endpoint NAK mode, the application must clear the DIEPCTLn.NAK status. This also clears the DIEPINTn.NAK Effective interrupt.

- DIEPCTLn.ClearNAK = 1'b1

6. If the application masked this interrupt earlier, it must be unmasked as follows:

- DIEPMSK.NAK Effective = 1'b1

[IN Endpoint Disable]

Use the following sequence to disable a specific IN endpoint (periodic/non-periodic) that has been previously enabled.

Application Programming Sequence

1. In Slave mode, the application must stop writing data on the AHB, for the IN endpoint to be disabled.

2. The application must set the endpoint in NAK mode. Refer to the section Setting the Endpoint NAK

- DIEPCTLn.SetNAK = 1'b1

3. Wait for DIEPINTn.NAK Effective interrupt.

4. Set the following bits in the DIEPCTLn register for the endpoint that must be disabled.

- DIEPMSK.NAK Effective = 1'b1
- DIEPCTLn.Endpoint Disable = 1
- DIEPCTLn.SetNAK = 1

5. Assertion of DIEPINTn.Endpoint Disabled interrupt indicates that the core has completely disabled the specified endpoint. Along with the assertion of the interrupt, the core also clears the following bits.

- DIEPCTLn.EPEnable = 1'b0
- DIEPCTLn.EPDisable = 1'b0

6. The application must read the DIEPTSIZn register for the periodic IN EP, to calculate how much data on the endpoint was transmitted on the USB.

7. The application must flush the data in the Endpoint transmit FIFO, by setting the following fields in the GRSTCTL register.

- GRSTCTL.TxFIFONum = Endpoint Transmit FIFO Number
- GRSTCTL.TxFFlush = 1

The application must poll the GRSTCTL register, until the TxFFlush bit is cleared by the core, which indicates the end of flush operation. To transmit new data on this endpoint, the application can re-enable the endpoint at a later point.

[Generic Non-periodic IN Data Transfers without Thresholding]

This section describes a regular non periodic IN data transfer when transmit thresholding is not enabled.

Application Requirements

1. Before setting up an IN transfer, the application must ensure that all data to be transmitted as part of the IN transfer is part of a single buffer, and must program the size of that buffer and its start address (in DMA mode) to the endpoint-specific registers.

2. For IN transfers, the Transfer Size field in the Endpoint Transfer Size register denotes a payload that constitutes multiple maximum-packet-size packets and a single short packet. This short packet is transmitted at the end of the transfer.

- To transmit a few maximum-packet-size packets and a short packet at the end of the transfer:
  • Transfer size[epnum] = n * mps[epnum] + sp
  (where n is an integer $\geq$ 0, and 0 $\leq$ sp < mps[epnum])
  • If (sp > 0), then packet count[epnum] = n + 1.
  Otherwise, packet count[epnum] = n
- To transmit a single zero-length data packet:
  • Transfer size[epnum] = 0
  • Packet count[epnum] = 1
- To transmit a few maximum-packet-size packets and a zero-length data packet at the end of the transfer, the application must split the transfer in two parts. The first sends maximum-packet-size data packets and the second sends the zero-length data packet alone.
  • First transfer: transfer size[epnum] = n * mps[epnum]; packet count = n;
  • Second transfer: transfer size[epnum] = 0; packet count = 1;

3. In DMA mode, the core fetches an IN data packet from the memory, always starting at a DWORD boundary. If the maximum packet size of the IN endpoint is not a multiple of 4, the application must arrange the data in the memory with pads inserted at the end of a maximum-packet-size packet so that a new packet always starts on a DWORD boundary.

4. Once an endpoint is enabled for data transfers, the core updates the Transfer Size register. At the end of IN transfer or Endpoint Disabled interrupt, the application must read the Transfer Size register to determine how much data posted in the transmit FIFO was already sent on the USB.

- Data fetched into transmit FIFO = Application-programmed initial transfer size – core-updated final transfer size

- Data transmitted on USB = (application-programmed initial packet count – Core updated final packet count) * mps[epnum]
- Data yet to be transmitted on USB = (Application-programmed initial transfer size – data transmitted on USB)

Internal Data Flow

1. The application must set the Transfer Size and Packet Count fields in the endpointspecific registers and enable the endpoint to transmit the data.

2. In Slave mode, the application must also write the required data to the transmit FIFO for the endpoint. In DMA mode, the core fetches the data from memory according to the application setting for the endpoint.

3. Every time a packet is written into the transmit FIFO, either by the core's internal DMA (in DMA mode) or the application (in Slave Mode), the transfer size for that endpoint is decremented by the packet size. The data is fetched from the memory (DMA/
Application), until the transfer size for the endpoint becomes 0. After writing the data into the FIFO, the "number of packets in FIFO" count is incremented (this is a 3-bit count, internally maintained by the core for each IN endpoint transmit FIFO. The maximum number of packets maintained by the core at any time in an IN endpoint FIFO is eight). For zero-length packets, a separate flag is set for each FIFO, without any data in the FIFO.

4. Once the data is written to the transmit FIFO, the core reads it out upon receiving an IN token. For every non-isochronous IN data packet transmitted with an ACK handshake, the packet count for the endpoint is decremented by one, until the packet count is zero. The packet count is not decremented on a TIMEOUT.

5. For zero length packets (indicated by an internal zero length flag), the core sends out a zero-length packet for the IN token and decrements the Packet Count field.

6. If there is no data in the FIFO for a received IN token and the packet count field for that endpoint is zero, the core generates a IN Tkn Rcvd When FIFO Empty Interrupt for the endpoint, provided the endpoint NAK bit is not set. The core responds with a NAK handshake for non-isochronous endpoints on the USB.

7. When the transfer size is 0 and the packet count is 0, the transfer complete interrupt for the endpoint is generated and the endpoint enable is cleared.

Application Programming Sequence

1. Program the DIEPTSIZn register with the transfer size and corresponding packet count. In DMA mode, also program the DIEPWMAn register.

2. Program the DIEPCTLn register with the endpoint characteristics and set the CNAK and Endpoint Enable bits. In DMA mode, ensure that the NextEp field is programmed so that the core fetches the data for IN endpoints in the correct order. See "Non-periodic IN Endpoint Sequencing" for details.

3. When transmitting non-zero length data packet in slave mode, the application must poll the DTXFSTSn register (where n is the FIFO number associated with that endpoint) to determine whether there is enough space in the data FIFO. The application can optionally use DIEPINTn.TxFEmp before writing the data.

[Generic non-periodic IN Data Transfers with Thresholding]

This section describes a regular non periodic IN data transfer when transmit thresholding is enabled.

Application Requirements

Application requirements are the same as those for DMA mode with no thresholding.

Internal Data Flow

1. The application should set the transfer size and packet count fields in the Endpoint pecific registers, and enable the endpoint to transmit the data.

2. The core fetches threshold amount of data for one endpoint before switching to the next in a round robin fashion with equal fairness. The priority is given to periodic endpoints. non-periodic endpoint data is fetched only if data for the periodic endpoints has been fetched or if the currently active token on the USB is a non-periodic one.The core will not switch, and continue to fetch till the complete packet, if it finds that the currently active IN token on the USB is for that particular endpoint and the FIFO does not have the omplete packet.

3. In response to an IN token on the USB, the core starts transmitting data, if the MAC finds t least threshold amount of data in the FIFO for that particular endpoint.

4. With each threshold amount of data written into the FIFO, the transfer size for that endpoint is decremented by the threshold size, except for the last packet. For the last packet, the transfer size is not decremented by the core. After writing the first threshold amount of data into the FIFO, the “number of packets in FIFO” count is incremented. For zero-length packets, a separate flag is set for that endpoint FIFO, without any data in the FIFO. This count is internally maintained by the core and is decremented when a full packet has been read out of the FIFO.

5. If the MAC sees an underrun case, where there is not enough data in the FIFO, the core will corrupt the data (invert the CRC) on the USB. The core will internally flush the FIFO, rewinding the DMA pointers and re fetch the packet(s).Also DIEPINTn.TxfifoUndrn bit will be set as an indication to the application.

6. For every non-ISO data IN packet transmitted, with an ACK handshake, the packet count for the endpoint is decremented by 1, until the packet count becomes zero. The packet count is not decremented on a TIMEOUT or underrun condition.
7. If the zero length flag in the FIFO is set (internally set by the core, when the application enables and endpoint for zero length), then core sends out zero length packet for the IN token and decrements the packet count field.
8. If there is no data (or partial threshold data) in the FIFO for a received IN token, and the packet count field for that endpoint is 0, the core generates a IN Tkn Rcvd When FIFO Empty Interrupt for the endpoint. The core responds with a NAK handshake for the non-ISO endpoints on the USB.

9. If the core does not receive a handshake after sending the packet (TIMEOUT), the core nternally flush the FIFO, rewind the DMA pointers and re-fetch the packet(s).

10. When the packet count is zero, the transfer complete interrupt for the endpoint is generated, and then the endpoint enable and transfer size are both cleared by the core.

Application Programming Sequence

1. Program the DIEPTSIZn register, for the transfer size and the corresponding packet count and the DIEPWMAn register.

2. Program the DIEPCTLn register, with the Endpoint Characteristics and set the CNAK and the Endpoint Enable bit.Also specify the Tx FIFO number in the DIEPCTLn.TXFNum field.

3. Assertion of DIEPINTn.XferCompl interrupt marks the successful completion of the nonperiodic IN transfer. Read to DIEPTSIZn register should indicate, a transfer size = 0 and packet count = 0, indicating all the data is transmitted on the USB.

[Generic Periodic IN Data Transfers without Thresholding]

This section describes a typical Periodic IN data transfer when thresholding is not enabled. Application Requirements

1. Application requirements 1, 2, 3, and 4 of “Generic Non-periodic IN Data Transfers without Thresholding” also apply to periodic IN data transfers, except for a slight modification of Requirement 2.

- The application can only transmit multiples of maximum-packet-size data packets or multiples of maximum-packet-size packets, plus a short packet at the end. To transmit a few maximum-packet-size packets and a short packet at the end of the transfer, the following conditions must be met.
  • transfer size[epnum] = n * mps[epnum] + sp
  (where n is an integer ≥ 0, and 0 ≤ sp < mps[epnum])
  • If (sp > 0), packet count[epnum] = n + 1
  Otherwise, packet count[epnum] = n;
  • mc[epnum] = packet count[epnum]
- The application cannot transmit a zero-length data packet at the end of transfer. It can transmit a single zero-length data packet by it self. To transmit a single zero-length data packet.
  • transfer size[epnum] = 0

- packet count[epnum] = 1
- mc[epnum] = packet count[epnum]

2. The application can only schedule data transfers 1 microframe at a time.

- (DIEPTSIZn.MC−1) * DIEPCTLn.MPS ≤ DIEPTSIZn.XferSiz ≤ DIEPTSIZn.MC * DIEPCTLn.MPS
- DIEPTSIZn.PktCnt = DIEPTSIZn.MC
- If DIEPTSIZn.XferSiz < DIEPTSIZn.MC * DIEPCTLn.MPS, the last data packet of the transfer is a short packet.

3. The application can schedule data transfers for multiple microframes, only if multiples of max packet sizes (up to 3 packets), should be transmitted every microframe. This is can be done, only when the core is operating in DMA mode. This is not a recommended mode of operation though.

- ((n*DIEPTSIZn.MC) - 1)*DIEPCTLn.MPS <= DIEPTSIZn.Transfer Size <= n*DIEPTSIZn.MC*DIEPCTLn.MPS
- DIEPTSIZn.Packet Count = n*DIEPTSIZn.MC
- n is the number of microframes for which the data transfers are scheduled

Data Transmitted per microframe in this case would be DIEPTSIZn.MC*DIEPCTLn.MPS, in all the microframes except the last one. In the microframe “n”, the data transmitted would be (DIEPTSIZn.TransferSize - (n-1)*DIEPTSIZn.MC*DIEPCTLn.MPS)

4. For Periodic IN endpoints, the data must always be prefetched 1 microframe ahead for transmission in the next microframe. This can be done, by enabling the Periodic IN endpoint 1 (micro)frame ahead of the (micro)frame in which the data transfer is scheduled.

5. The complete data to be transmitted in the (micro)frame must be written into the transmit FIFO (either by the application or the DMA), before the Periodic IN token is received. Even when 1 DWORD of the data to be transmitted per microframe is missing in the transmit FIFO when the Periodic IN token is received, the core behaves as when the FIFO was empty. When the transmit FIFO is empty,

- A zero data length packet would be transmitted on the USB for ISO IN endpoints
- A NAK handshake would be transmitted on the USB for INTR IN endpoints

6. For a High Bandwidth IN endpoint with three packets in a microframe, the application can program the endpoint FIFO size to be 2*max_pkt_size and have the third packet load in after the first packet has been transmitted on the USB.

Internal Data Flow

1. The application must set the Transfer Size and Packet Count fields in the endpointspecific registers and enable the endpoint to transmit the data.

2. In Slave mode, the application must also write the required data to the associated transmit FIFO for the endpoint. In DMA mode, the core fetches the data for the endpoint from memory, according to the application setting.

3. Every time either the core’s internal DMA (in DMA mode) or the application (in Slave mode) writes a packet to the transmit FIFO, the transfer size for that endpoint is decremented by the packet size. The data is fetched from DMA or application memory until the transfer size for the endpoint becomes 0.

4. When an IN token is received for an periodic endpoint, the core transmits the data in the FIFO, if available. If the complete data payload (complete packet, in dedicated FIFO mode) for the microframe is not present in the FIFO, then the core generates an IN Tkn Rcvd When TxF Empty Interrupt for the endpoint.

- A zero-length data packet is transmitted on the USB for isochronous IN endpoints
- A NAK handshake is transmitted on the USB for interrupt IN endpoints

5. The packet count for the endpoint is decremented by 1 under the following conditions:

- For isochronous endpoints, when a zero- or non-zero-length data packet is transmitted
- For interrupt endpoints, when an ACK handshake is transmitted

6. When the transfer size and packet count are both 0, the Transfer Completed interrupt for the endpoint is generated and the endpoint enable is cleared.

7. At the “Periodic frame Interval” (controlled by DCFG.PerFrint), when the core finds nonempty any of the isochronous IN endpoint FIFOs scheduled for the current (micro)frame non-empty, the core generates a GINTSTS.incompISOIN interrupt.

Application Programming Sequence (Transfer Per Microframe)

1. Program the DIEPTSIZn register. In DMA mode, also program the DIEPWMAn register.

2. Program the DIEPCTLn register with the endpoint characteristics and set the CNAK and Endpoint Enable bits.

3. In Slave mode, write the data to be transmitted in the next microframe to the transmit FIFO as described in “Periodic Packet Write in Slave Mode: Shared FIFO”.

4. Asserting the DIEPINTn.In Token Rcvd When TxF Empty interrupt indicates that either the DMA or application has not yet written all data to be transmitted to the transmit FIFO.

- If the interrupt endpoint is already enabled when this interrupt is detected, ignore the interrupt. If it is not enabled, enable the endpoint so that the data can be transmitted on the next IN token attempt.
- If the isochronous endpoint is already enabled when this interrupt is detected, see “Incomplete Isochronous IN Data Transfers” for more details.

5. The core handles timeouts internally on interrupt IN endpoints programmed as periodic endpoints, or when Dedicated FIFO operation is used, without application intervention. The application, thus, never detects a DIEPINTn.TimeOUT interrupt for periodic interrupt IN endpoints.

6. Asserting the DIEPINTn.XferCompl interrupt with no DIEPINTn.In Tkn Rcvd When TxF Empty interrupt indicates the successful completion of an isochronous IN transfer. A read to the DIEPTSIZn register must indicate transfer size = 0 and packet count = 0, indicating all data is transmitted on the USB.

7. Asserting the DIEPINTn.XferCompl interrupt, with or without the DIEPINTn.In Tkn Rcvd When TxF Empty interrupt, indicates the successful completion of an interrupt IN transfer. A read to the DIEPTSIZn register must indicate transfer size = 0 and packet count = 0, indicating all data is transmitted on the USB.

8. Asserting the GINTSTS.incomplete Isochronous IN Transfer interrupt with none of the aforementioned interrupts indicates the core did not receive at least 1 periodic IN token in the current microframe.

- For isochronous IN endpoints, see “Incomplete Isochronous IN Data Transfers” for more details.

[Generic Periodic IN Data Transfers with Thresholding]

This section describes a typical Periodic IN data transfer when thresholding is enabled.

Application Requirements

Application requirements are the same as that for DMA mode with no thresholding.

Internal Data Flow

1. The application should set the transfer size and packet count fields in the Endpoint Specific registers, and enable the endpoint to transmit the data.

2. The core fetches threshold amount of data for one endpoint before switching to the next in a round robin fashion with equal fairness. The priority is given to periodic endpoints. The core will not switch, and continue to fetch till the complete packet, if it finds that the currently active IN token on the USB side is for that particular endpoint and the FIFO does not have the complete packet.

3. In response to an IN token on the USB, the core starts transmitting, if the MAC finds at least threshold amount of data in the FIFO for that particular endpoint.

4. After a full packet has been written into the FIFO, the transfer size for that endpoint is decremented by the packet size (or less). After writing the first threshold amount of data into the FIFO, the “number of packets in FIFO” count is incremented. For zero length packets, a separate flag will be set in the FIFO, without any data in the FIFO. This count is internally maintained b y the core and is decremented when a full packet has been read out of the FIFO.

5. If the MAC sees an underrun case, where there is not enough data in the FIFO, the core will corrupt the data (invert the CRC) on the USB. For isochronous endpoints, Underrun interrupt (DIEPINTn.TxFifoUndrn) is generated by the core for this endpoint. For interrupt endpoints, the core will flush the FIFO, rewind the DMA pointers and re-fetch the data.

6. If the core sees a timeout or an underrun condition for an interrupt endpoint, the core flushes the fifo, rewinds the DMA pointers and re-fetches the packet. No interrupt is generated to the application.

7. When an IN token is received for an periodic endpoint, the core transmits the data in the FIFO, if the core sees at least threshold amount of data. If the threshold amount of data for the packet is not present in the FIFO, the core generates a IN Tkn Rcvd When TxF Empty Interrupt for the endpoint.

- A zero data length packet would be transmitted on the USB for ISO IN endpoints
- A NAK handshake would be transmitted on the USB for INTR IN endpoints

8. The packet count for the endpoint is decremented by 1, on the following conditions

- When a zero or non zero data length for ISO endpoints
- When an ACK handshake is transmitted for INTR endpoints

9. The packet count is not decremented when the core has to corrupt a packet because of underrun condition.
10. When the transfer size is 0 and the packet count is 0, the transfer complete interrupt for the endpoint is generated and the endpoint enable is cleared.

Application Programming Sequence (Transfer Per Microframe)

1. Program the DIEPTSIZn register and in addition, in DMA mode program the DIEPWMAn register.

2. Program the DIEPCTLn register, with the Endpoint Characteristics and set the CNAK bit and the Endpoint Enable bit.Also specify the Tx FIFO number in the DIEPCTLn.TXFNum field.

3. Assertion of DIEPINTn.In Token Rcvd When TxF Empty interrupt indicates that the complete data to be transmitted is not written into the transmit FIFO.

- If the INTR endpoint is already enabled, when this interrupt is seen, ignore the interrupt. If the IN Endpoint in Dedicated Tx FIFO config is not enabled, enable the endpoint, so that the data could be transmitted on the next attempt of the IN token.
- If the ISO endpoint is already enabled, when this interrupt is seen, refer to the section Incomplete ISO IN Data Transfers, for more details.

4. TimeOUT on Interrupt IN endpoints, is handled by the core internally, without any application intervention. The application never sees any interrupt for timeout.

5. Assertion of DIEPINTn.XferCompl interrupt without any DIEPINTn.In Tkn Rcvd when TxF Empty interrupt, marks the successful completion of the ISO IN transfer. Read to DIEPTSIZn register should indicate, a transfer size = 0 and packet count = 0, indicating all the data is transmitted on the USB.

6. Assertion of DIEPINTn.XferCompl interrupt with/without any DIEPINTn.In Tkn Rcvd when TxF Empty interrupt, marks the successful completion of the INTR IN transfer. Read to DIEPTSIZn register should indicate, a transfer size = 0 and packet count = 0, indicating all the data is transmitted on the USB.

7. Assertion of GINTSTS.incomplete ISO IN Transfer interrupt with out any of the previously mentioned interrupts indicates that at least 1 periodic IN token was not received by the core, in the current microframe.

- For ISO IN endpoints, refer to the section Incomplete ISO IN Data Transfers, for more details.

8. Assertion of DIEPINTn.TxFifoUndrn interrupt indicate that there was an underrun condition for the isochronous transaction in the current microframe. The application may choose ignore this interrupt, as this will eventually result in GINTSTS.Incomplete isochronous IN interrupt at the end of periodic frame. The application can also choose to service this interrupt. If they choose to do so, then they can save some time in re-enabling the endpoint for the next microframe. In response to this interrupt,

- Disable the endpoint (Check section Disabling IN Endpoint in Dedicated Tx FIFO config).
- Application reads Endpoint DIEPSIZn register to see how much data is transferred.
- Re-enable the endpoint for the next microframe.

[Incomplete Isochronous IN Data Transfers]

This section describes what the application must do on an incomplete isochronous IN data transfer.

Internal Data Flow

1. An isochronous IN transfer is treated as incomplete in one of the following conditions.

- The core receives a corrupted isochronous IN token on at least one isochronous IN endpoint. In this case, the application detects a GINTSTS.incomplete Isochronous IN Transfer interrupt.
- The application or DMA is slow to write the complete data payload to the transmit FIFO and an IN token is received before the complete data payload is written to the FIFO. In this case, the application detects a DIEPINTn.IN Tkn Rcvd When TxFIFO Empty interrupt. The application can ignore this interrupt, as it eventually results in a GINTSTS.incomplete Isochronous IN Transfer interrupt at the end of periodic frame.
  • The core transmits a zero-length data packet on the USB in response to the received IN token.
- If thresholding is enabled and there was an underrun condition, the core also generates a DINEPINTn.TxfifoUndrn interrupt. The application can ignore this interrupt, which eventually results in a GINTSTS.incomplete ISO IN Transfer interrupt.

2. In either of the aforementioned cases, in Slave mode, the application must stop writing the data payload to the transmit FIFO as soon as possible.

3. The application must set the NAK bit and the disable bit for the endpoint. In DMA mode, the core automatically stops fetching the data payload when the endpoint disable bit is set.

4. The core disables the endpoint, clears the disable bit, and asserts the Endpoint Disable interrupt for the endpoint.

Application Programming Sequence

1. The application can ignore the DIEPINTn.IN Tkn Rcvd When TxFIFO empty interrupt on any isochronous IN endpoint, as it eventually results in a GINTSTS.incomplete Isochronous IN Transfer interrupt. The application can also ignore DIEPINTn.TxfifoUndrn interrupt when thresholding is enabled.

2. Assertion of the GINTSTS.incomplete Isochronous IN Transfer interrupt indicates an incomplete isochronous IN transfer on at least one of the isochronous IN endpoints.

3. The application must read the Endpoint Control register for all isochronous IN endpoints to detect endpoints with incomplete IN data transfers.

4. In Slave mode, the application must stop writing data to the Periodic Transmit FIFOs associated with these endpoints on the AHB.

5. In both modes of operation, program the following fields in the DIEPCTLn register to disable the endpoint.

- DIEPCTLn.SetNAK = 1
- DIEPCTLn.Endpoint Disable = 1

6. The DIEPINTn.Endpoint Disabled interrupt's assertion indicates that the core has disabled the endpoint.

- At this point, the application must flush the data in the associated transmit FIFO or overwrite the existing data in the FIFO by enabling the endpoint for a new transfer in the next microframe. To flush the data, the application must use the GRSTCTL register.

[Stalling Non-Isochronous IN Endpoints]

This section describes how the application can stall a non-isochronous endpoint.

Application Programming Sequence

1. Disable the IN endpoint to be stalled. Set the Stall bit as well.

- DIEPCTLn.Endpoint Disable = 1, when the endpoint is already enabled
- DIEPCTLn.STALL = 1
- The Stall bit always takes precedence over the NAK bit

2. Assertion of the DIEPINTn.Endpoint Disabled interrupt indicates to the application that the core has disabled the specified endpoint.

3. The application must flush the Non-periodic or Periodic Transmit FIFO, depending on the endpoint type. In case of a non-periodic endpoint, the application must re-enable the other non-periodic endpoints, which do not need to be stalled, to transmit data.

4. Whenever the application is ready to end the STALL handshake for the endpoint, the DIEPCTLn.STALL bit must be cleared.

5. If the application sets or clears a STALL for an endpoint due to a SetFeature.Endpoint Halt command or ClearFeature.Endpoint Halt command, the Stall bit must be set or cleared before the application sets up the Status stage transfer on the control endpoint.

Special Case: Stalling the Control OUT Endpoint

The core must stall IN/OUT tokens if, during the Data stage of a control transfer, the host sends more IN/OUT tokens than are specified in the SETUP packet. In this case, the application must to enable DIEPINTn.INTknTXFEmp and DOEPINTn.OUTTknEPdis interrupts during the Data stage of the control transfer, after the core has transferred the amount of data specified in the SETUP packet. Then, when the application receives this interrupt, it must set the STALL bit in the corresponding endpoint control register, and clear this interrupt.

[Examples]

Slave Mode Bulk IN Transaction

1. The host attempts to read data (IN token) from an endpoint.

2. On receiving the IN token on the USB, the core returns a NAK handshake, because no data is available in the transmit FIFO.

3. To indicate to the application that there was no data to send, the core generates a DIEPINTn.IN Token Rcvd When TxFIFO Empty interrupt.

4. When data is ready, the application sets up the DIEPTSIZn register with the Transfer Size and Packet Count fields.
5. The application writes one maximum packet size or less of data to the Non-periodic TxFIFO.

6. The host reattempts the IN token.

7. Because data is now ready in the FIFO, the core now responds with the data and the host ACKs it.

8. Because the XferSize is now zero, the intended transfer is complete. The device core generates a DIEPINTn.XferCompl interrupt.

9. The application processes the interrupt and uses the setting of the DIEPINTn.XferCompl interrupt bit to determine that the intended transfer is complete.

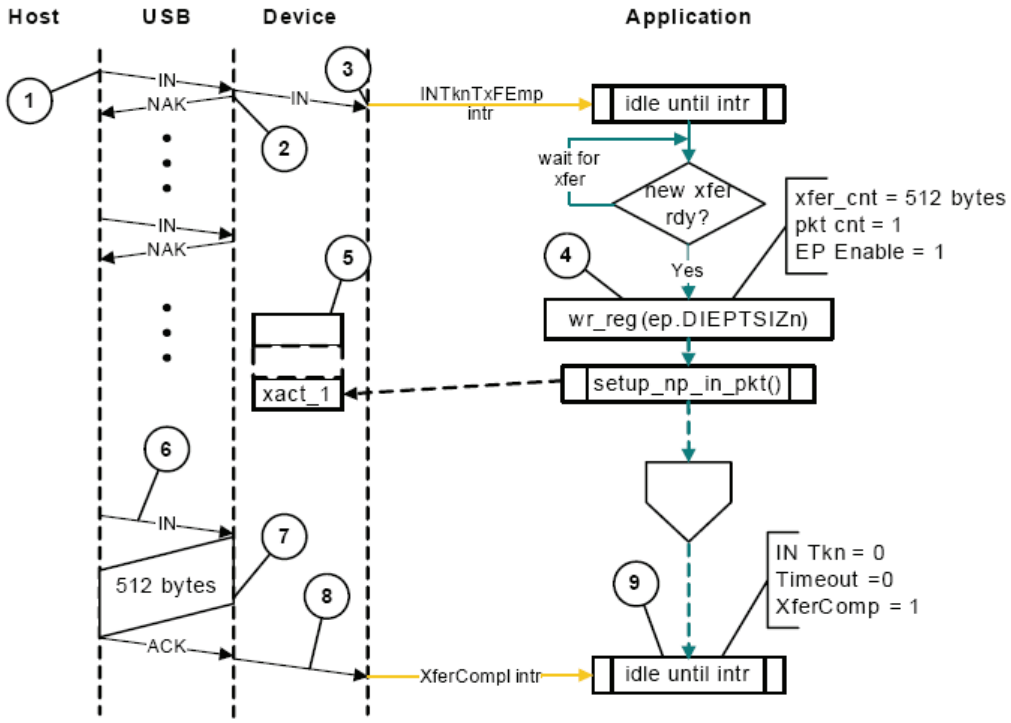


Figure 4.23 Slave Mode Bulk IN Transaction

Slave Mode Bulk IN Transfer (Pipelined Transaction)

1. The host attempts to read data (IN token) from an endpoint.

2. On receiving the IN token on the USB, the core returns a NAK handshake, because no data is available in the transmit FIFO.

3. To indicate that there was no data to send, the core generates an DIEPINTn.InTkn Rcvd When TxFIFO Empty interrupt.

4. When data is ready, the application sets up the DIEPTSIZn register with the transfer size and packet count.

5. The application writes one maximum packet size or less of data to the Non-periodic TxFIFO.

6. The host reattempts the IN token.

7. Because data is now ready in the FIFO, the core responds with the data, and the host ACKs it.

8. When the TxFIFO level falls below the halfway mark, the core generates a GINTSTS.NonPeriodic TxFIFO Empty interrupt. This triggers the application to start writing additional data packets to the FIFO.

9. A data packet for the second transaction is ready in the TxFIFO.

10. A data packet for third transaction is ready in the TxFIFO while the data for the second packet is being sent on the bus.
11. The second data packet is sent to the host.

12. The last short packet is sent to the host.

13. Because the last packet is sent and XferSize is now zero, the intended transfer is complete.

The core generates a DIEPINTn.XferCompl interrupt.

14. The application processes the interrupt and uses the setting of the DIEPINTn.XferCompl interrupt bit to determine that the intended transfer is complete.

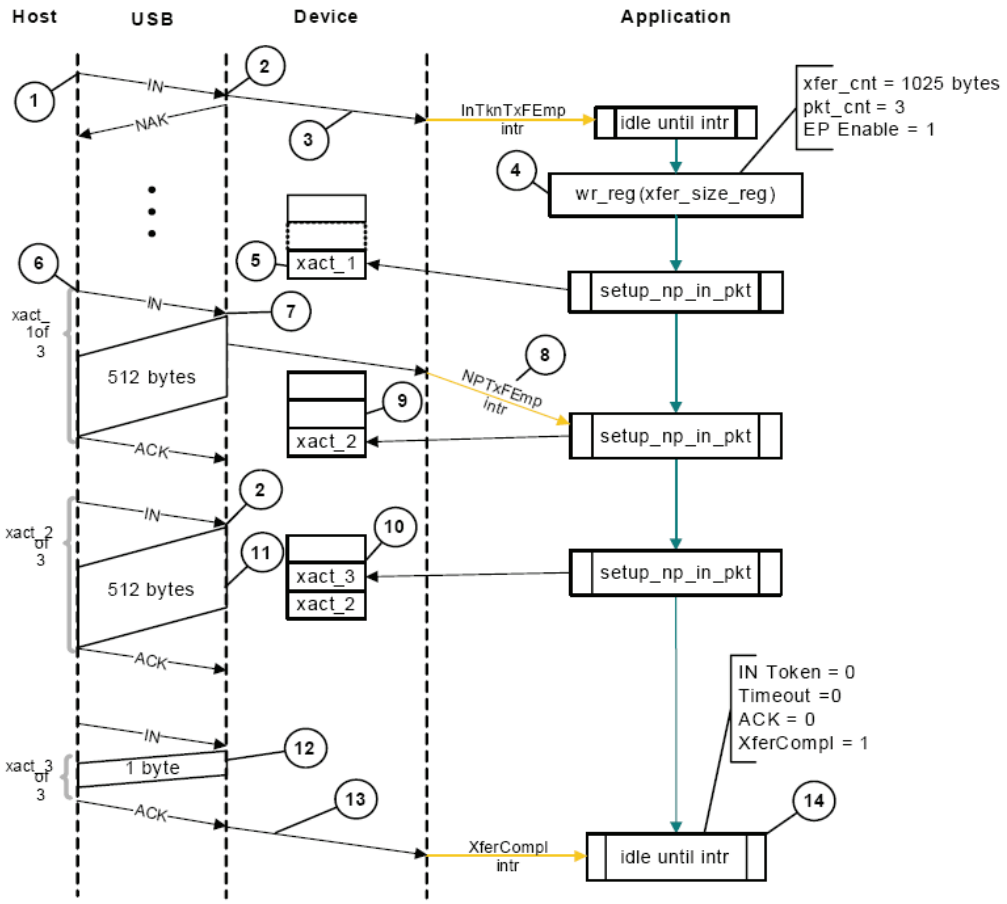


Figure 4.24 Slave Mode Bulk IN Transfer (Pipelined Transaction)

Slave Mode Bulk IN Two-Endpoint Transfer

1. The host attempts to read data (IN token) from endpoint 1.

2. On receiving the IN token on the USB, the core returns a NAK handshake, because no data is available in the transmit FIFO for endpoint 1, and generates a DIE PINT1.InTkn Rcvd When TxFIFO Empty interrupt.

3. The application processes the interrupt and initializes DIEPTSIZ1 register with the Transfer Size and Packet Count fields. The application starts writing the transaction data to the transmit FIFO.

4. The application writes one maximum packet size or less of data for endpoint 1 to the Non-periodic TxFIFO.

5. Meanwhile, the host attempts to read data (IN token) from endpoint 2.

6. On receiving the IN token on the USB, the core returns a NAK handshake, because no data is available in the transmit FIFO for endpoint 2, and the core generates a DIEPINT2.InTkn Rcvd When TxFIFO Empty interrupt.

7. Because the application has completed writing the packet for endpoint 1, it initializes the DIEPTSIZ2 register with the Transfer Size and Packet Count fields. The application starts writing the transaction data into the transmit FIFO for endpoint 2.

8. The host repeats its attempt to read data (IN token) from endpoint 1.

9. Because data is now ready in the TxFIFO, the core returns the data, which the host ACKs.

10. Meanwhile, the application has initialized the data for the next two packets in the TxFIFO (ep2.xact1 and ep1.xact2, in order).

11. The host repeats its attempt to read data (IN token) from endpoint 2.

12. Because endpoint 2's data is ready, the core responds with the data (ep2.xact_1), which the host ACKs.

13. Meanwhile, the application has initialized the data for the next two packets in the TxFIFO (ep2.xact2 and ep1.xact3, in order). The application has finished initializing data for the two endpoints involved in this scenario.

14. The host repeats its attempt to read data (IN token) from endpoint 1.

15. Because data is now ready in the FIFO, the core responds with the data, which the host ACKs.

16. The host repeats its attempt to read data (IN token) from endpoint 2.

17. With data now ready in the FIFO, the core responds with the data, which the host ACKs.

18. With the last packet for endpoint 2 sent and its XferSize now zero, the intended transfer is complete. The core generates a DIEPINT2.XferCompl interrupt for this endpoint.

19. The application processes the interrupt and uses the setting of the DIEPINT2.XferCompl interrupt bit to determine that the intended transfer on endpoint 2 is complete.

20. The host repeats its attempt to read data (IN token) from endpoint 1 (last transaction).

21. With data now ready in the FIFO, the core responds with the data, which the host ACKs.

22. Because the last endpoint one packet has been sent and XferSize is now zero, the intended transfer is complete. The core generates a DIEPINT1.XferCompl interrupt for this endpoint.

23. The application processes the interrupt and uses the setting of the DIEPINT1.XferCompl interrupt bit to determine that the intended transfer on endpoint 1 is complete.

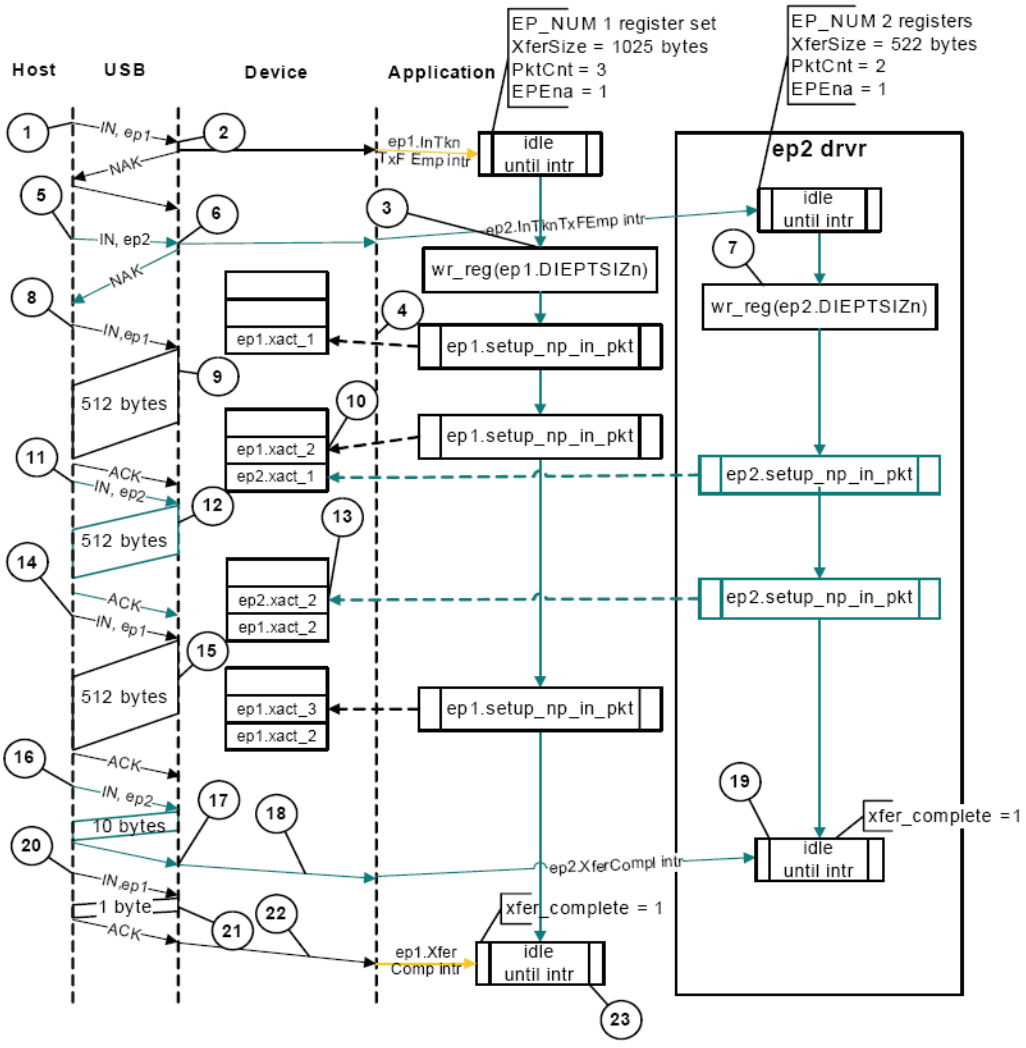


Figure 4.25 Slave Mode Bulk IN Two-Endpoint Transfer

Bulk IN Stall

1. The application has scheduled an IN transfer on receiving the DIEPINTn.InTknRcvd When TxFIFO Empty interrupt.

2. When the transfer is in progress, the application must force a STALL on the endpoint. This could be because the application has received a SetFeature.Endpoint Halt command. The application sets the Stall bit, disables the endpoint and waits for the DIEPINTn.Endpoint Disabled interrupt. This generates STALL handshakes for the endpoint on the USB.
3. On receiving the interrupt, the application flushes the Non-periodic Transmit FIFO and clears the DCTL.GlobalINNPNAK bit.

4. On receiving the ClearFeature.Endpoint Halt command, the application clears the Stall bit.

5. The endpoint behaves normally and the application can re-enable the endpoint for new transfers.

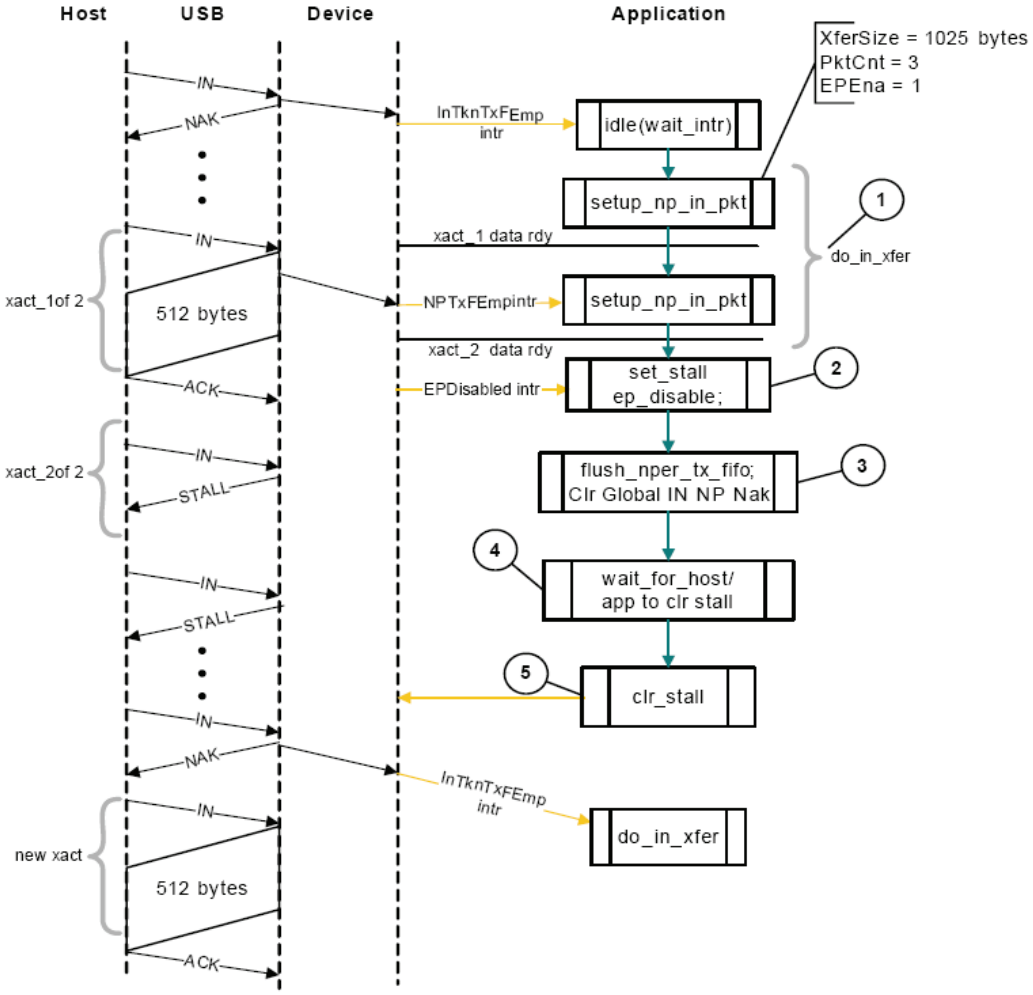


Figure 4.26 Bulk IN Stall

Bulk IN DMA mode with Thresholding

1. The host attempts to read data(IN token) from an endpoint

2. On receiving the IN token on the USB bus, the core sends back a NAK handshake because no data is available in the Transmit FIFO

3. To indicate to the application that there was no data to send, the core generates a DIEPINTn.IN Token Rcvd When TxF Empty interrupt.

4. When data is ready, the application sets up the DIEPTSIZn register with the TransferSize and Packet Count fields and enables the endpoint.

5. On seeing the endpoint enabled, the internal DMA engine start fetching the first threshold amount of data.

6. DMA engine fetching the second threshold.

7. The host re-attempts the IN token, and core start sending the data because it sees at least threshold amount of data in the FIFO.

8. The DMA engine starts fetching the third threshold after it sees threshold amount of space.

9. The MAC sees an underrun condition because of FIFO being empty in the middle of the packet, stops the packet and corrupt the CRC.

10. DMA engine starts to fetch the last threshold for that packet but it is late and will eventually result in underrun.

11. No handshake response from the host, because the packet was corrupted. Core rewinds the pointers and flush the FIFO.

12. The core starts to re-fetch the packet, staring with the first threshold.

13. The host re-attempts the IN token, and the core starts sending the data, since the core detects at least the threshold amount of data in the FIFO.

14. The core fetches the last threshold in time.

15. The core receives the handshake from host and Because the XferSize is now zero, the intended transfer is complete. Device core generates a DIEPINTn.XferCompl interrupt.

16. The application processes the interrupt and uses the setting of DIEPINTn.XferCompl interrupt bit to determine that the intended transfer is complete.

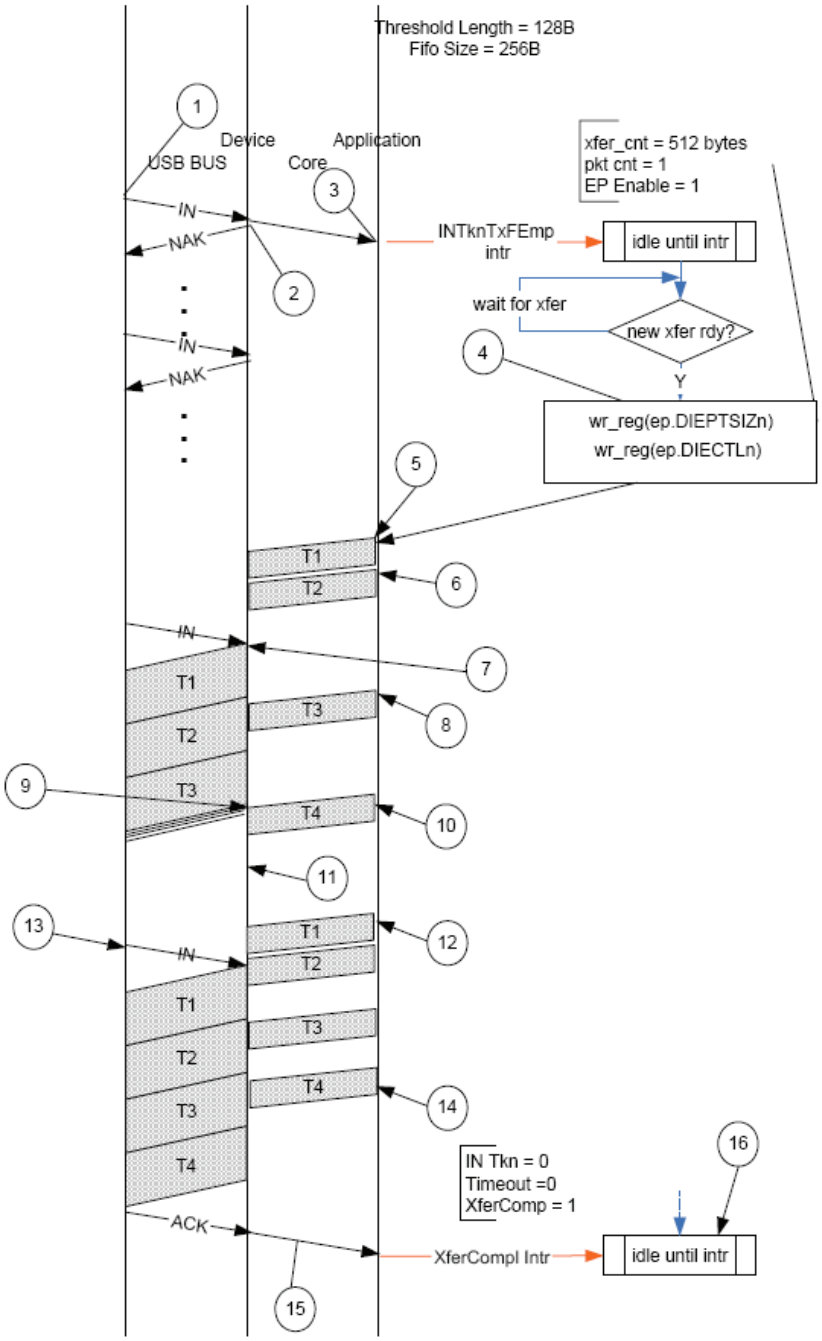


Figure 4.27 Bulk IN DMA mode with Thresholding

Isochronous IN DMA Mode with Thresholding

The FIFO size is assumed to be 512 Bytes and the threshold length is 128 Bytes.

1. Application enabled the isochronous IN endpoint for Odd microframe.

2. Core starts fetching the first 2 thresholds.

3. Core starts sending data out in response to the IN token.

4. Core sees an underrun condition, because the third threshold was not fetched in time.

5. Core generates DIEPINTn.TXFifoUnderrun interrupt (In this case, application ignores this interrupt).

6. At the End of Periodic Frame Interval, core generates GINTSTS.IncompIISO interrupt.

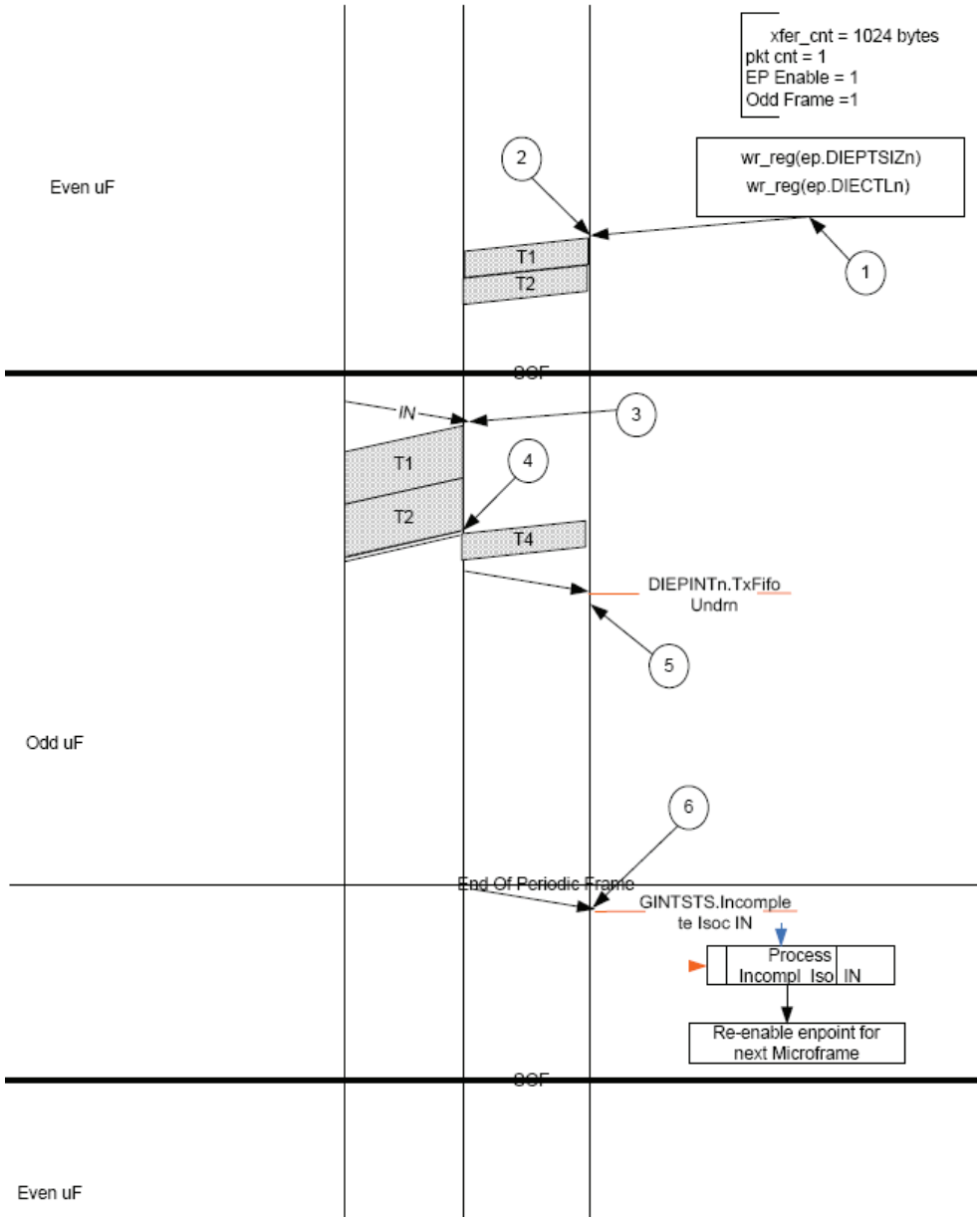


Figure 4.28 Isochronous IN DMA Mode with Thresholding

4.7.2.3 Control Transfers

This section describes the various types of control transfers.

[Control Write Transfers (SETUP, Data OUT, Status IN)]

This section describes control write transfers.

Application Programming Sequence

1. Assertion of the DOEPINTn.SETUP Packet interrupt indicates that a valid SETUP packet has been transferred to the application. See “SETUP Transactions” for more details. At the end of the Setup stage, the application must reprogram the DOEPTSIZn.SUPCnt field to 3 to receive the next SETUP packet.

2. If the last SETUP packet received before the assertion of the SETUP interrupt indicates a data OUT phase, program the core to perform a control OUT transfer as explained in “Generic Non-Isochronous OUT Data Transfers without Thresholding”. In DMA mode, the application must reprogram the DOEPWMAn register to receive a control OUT data packet

to a different memory location.

3. In a single OUT data transfer on control endpoint 0, the application can receive up to 64 bytes. If the application is expecting more than 64 bytes in the Data OUT stage, the application must re-enable the endpoint to receive another 64 bytes, and must continue to do so until it has received all the data in the Data stage.

4. Assertion of the DOEPINTn.Transfer Compl interrupt on the last data OUT transfer indicates the completion of the data OUT phase of the control transfer.

5. On completion of the data OUT phase, the application must do the following.

- To transfer a new SETUP packet in DMA mode, the application must re-enable the control OUT endpoint as explained in section in section “SETUP Transactions”.
  • DOEPCTLn.EPEna = 1’b1
- To execute the received Setup command, the application must program the required registers in the core. This step is optional, based on the type of Setup command received.

6. For the status IN phase, the application must program the core as described in “Generic Non-periodic IN Data Transfers without Thresholding” to perform a data IN transfer.

7. Assertion of the DIEPINTn.Transfer Compl interrupt indicates completion of the status IN phase of the control transfer.

8. The previous step must be repeated until the DIEPINTn.Transfer Compl interrupt is detected on the endpoint, marking the completion of the control write transfer.

[Control Read Transfers (SETUP, Data IN, Status OUT)]

This section describes control write transfers.

Application Programming Sequence

1. Assertion of the DOEPINTn.SETUP Packet interrupt indicates that a valid SETUP packet has been transferred to the application. See “SETUP Transactions” for more details. At the end of the Setup stage, the application must reprogram the DOEPTSIZn.SUPCnt field to 3 to receive the next SETUP packet.

2. If the last SETUP packet received before the assertion of the SETUP interrupt indicates a data IN phase, program the core to perform a control IN transfer as explained in “Generic Non-periodic IN Data Transfers without Thresholding”.

3. On a single IN data transfer on control endpoint 0, the application can transmit up to 64 bytes. To transmit more than 64 bytes in the Data IN stage, the application must re-enable the endpoint to transmit another 64 bytes, and must continue to do so, until it has transmitted all the data in the Data stage.

4. The previous step must be repeated until the DIEPINTn.Transfer Compl interrupt is detected for every IN transfer on the endpoint.

5. The DIEPINTn.Transfer Compl interrupt on the last IN data transfer marks the completion of the control transfer’s Data stage.

6. To perform a data OUT transfer in the status OUT phase, the application must program the core as described in “SETUP and OUT Data Transfers”.

- The application must program the DCFG.NZStsOUTHShk handshake field to a proper setting before transmitting an data OUT transfer for the Status stage.
- In DMA mode, the application must reprogram the DOEPWMAn register to receive the control OUT data packet to a different memory location.

7. Assertion of the DOEPINTn.Transfer Compl interrupt indicates completion of the status OUT phase of the control transfer. This marks the successful completion of the control read transfer.

- To transfer a new SETUP packet in DMA mode, the application must re-enable the control OUT endpoint as explained in “SETUP Transactions”.
  • DOEPCTLn.EPEna = 1’b1

[Two-Stage Control Transfers (SETUP/Status IN)]

This section describes two-stage control transfers.

Application Programming Sequence

1. Assertion of the DOEPINTn.SetUp interrupt indicates that a valid SETUP packet has been transferred to the application. See “SETUP Transactions” for more detail. To receive the next SETUP packet, the application must reprogram the DOEPTSIZn.SUPCnt field to 3 at the end of the Setup stage.

2. Decode the last SETUP packet received before the assertion of the SETUP interrupt. If the packet indicates a two-stage control command, the application must do the following.

- To transfer a new SETUP packet in DMA mode, the application must re-enable the control OUT endpoint. See “SETUP Transactions” for details.
  - DOEPCTLn.EPEna = 1’b1
- Depending on the type of Setup command received, the application can be required to program registers in the core to execute the received Setup command.

3. For the status IN phase, the application must program the core described in “Generic Non-periodic IN Data Transfers without Thresholding” to perform a data IN transfer.

4. Assertion of the DIEPINTn.Transfer Compl interrupt indicates the completion of the status IN phase of the control transfer.

5. The previous step must be repeated until the DIEPINTn.Transfer Compl interrupt is detected on the endpoint, marking the completion of the two-stage control transfer.

Example: Two-Stage Control Transfer

1. SETUP packet #1 is received on the USB and is written to the receive FIFO, and the core responds with an ACK handshake. This handshake is lost and the host detects a timeout.

2. The SETUP packet in the receive FIFO results in a GINTSTS.RxFLvl interrupt to the application, causing the application to empty the receive FIFO.

3. SETUP packet #2 on the USB is written to the receive FIFO, and the core responds with an ACK handshake.
4. The SETUP packet in the receive FIFO sends the application the GINTSTS.RxFLvl interrupt and the application empties the receive FIFO.

5. After the second SETUP packet, the host sends a control IN token for the status phase. The core issues a NAK response to this token, and writes a Setup Stage Done entry to the receive FIFO. This entry results in a GINTSTS.RxFLvl interrupt to the application, which empties the receive FIFO. After reading out the Setup Stage Done DWORD, the core asserts the DOEPINTn.SetUp packet interrupt to the application.

6. On this interrupt, the application processes SETUP Packet #2, decodes it to be a two-stage control command, and clears the control IN NAK bit.

- DIEPCTLn.CNAK = 1

7. When the application clears the IN NAK bit, the core interrupts the application with a DIEPINTn.INTknTXFEmp. On this interrupt, the application enables the control IN endpoint with a DIEPTSIZn.XferSize of 0 and a DIEPTSIZn.PktCnt of 1. This results in a zero-length data packet for the status IN token on the USB.

8. At the end of the status IN phase, the core interrupts the application with a DIEPINTn.XferCompl interrupt.

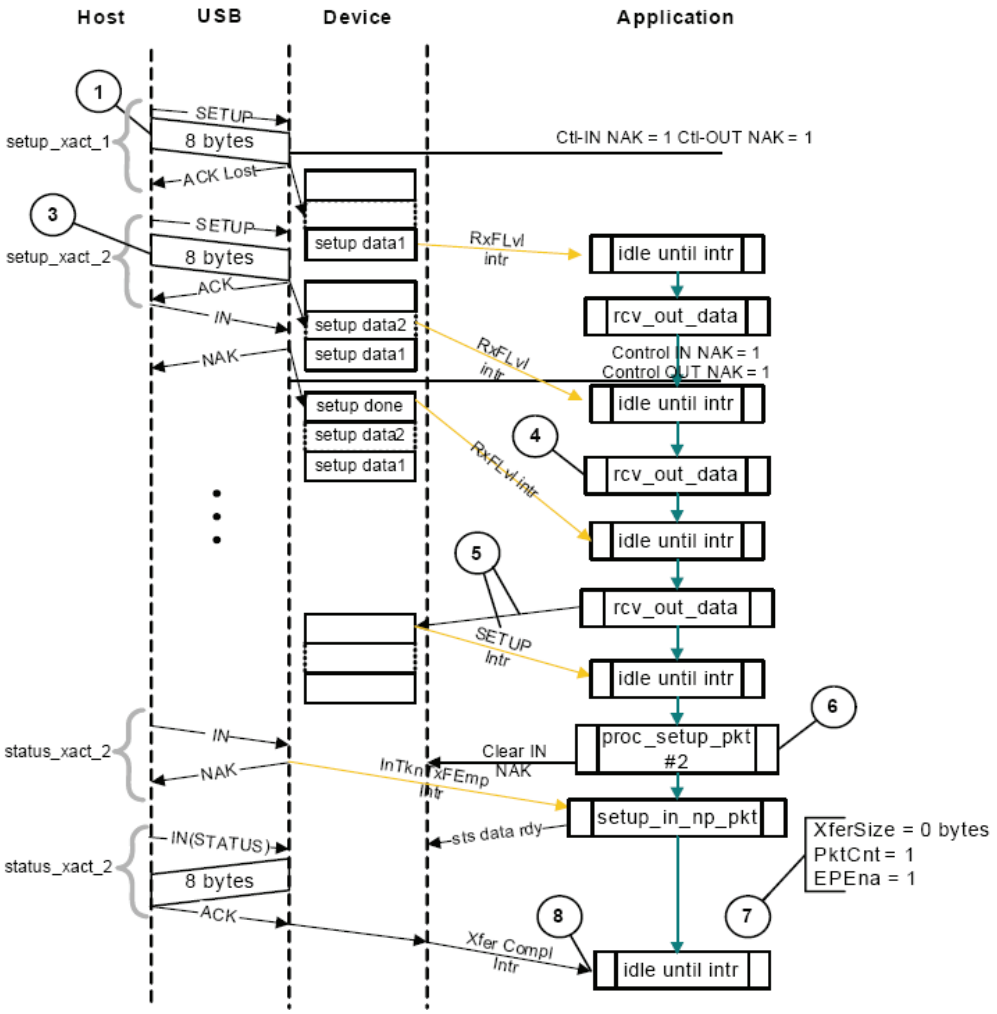


Figure 4.29 Two-Stage Control Transfer

4.7.3 Handling Babble Conditions

If the otg core receives a packet that is larger than the maximum packet size for that endpoint, the core stops writing data to the Rx buffer and waits for the end of packet (EOP). When the core detects the EOP, it flushes the packet in the Rx buffer and does not send any response to the host.

If the core continues to receive data at the EOF2 (the end of frame 2, which is very close to SOF), the core generates an early_suspend interrupt (GINTSTS.ErlySusp). On receiving this interrupt, the application must check the erratic_error status bit (DSTS.ErrticErr). If this bit is set, the application must take it as a long babble and perform a soft reset.

4.7.4 Worst Case Response Time

When the otg core acts as a device, there is a worst case response time for any tokens that follow an isochronous OUT. This worst case response time depends on the AHB clock frequency.

The core registers are in the AHB domain, and the core does not accept another token before updating these register values. The worst case is for any token following an isochronous OUT, because for an isochronous transaction, there is no handshake and the next token could come sooner. This worst case value is 7 PHY clocks when the AHB clock is the same as the PHY clock. When AHB clock is faster, this value is smaller.

If this worst case condition occurs, the core responds to bulk/interrupt tokens with a NAK and drops isochronous and SETUP tokens. The host interprets this as a timeout condition for SETUP and retries the SETUP packet. For isochronous transfers, the incomplISOCIN and incomplISOCOUT interrupts inform the application that isochronous IN/OUT packets were dropped.

4.7.5 Choosing the Value of GUSBCFG.USBTrdTim

The value in GUSBCFG.USBTrdTim is the time it takes for the MAC, in terms of PHY clocks after it has received an IN token, to get the FIFO status, and thus the first data from PFC (Packet FIFO Controller) block. This time involves the synchronization delay between the PHY and AHB clocks. The worst case delay for this is when the AHB clock is the same as the PHY clock. In this case, the delay is 5 clocks. If the PHY clock is running at 60 MHz and the AHB is running at 30 MHz, this value is 9 clocks.

Once the MAC receives an IN token, this information (token received) is synchronized to the AHB clock by the PFC (the PFC runs on the AHB clock). The PFC then reads the data from the SPRAM and writes it into the dual clock source buffer. The MAC then reads the data out of the source buffer (4 deep).

If the AHB is running at a higher frequency than the PHY, the application can use a smaller value for GUSBCFG.USBTrdTim. Next figure explains the 5-clock delay. This diagram has the following signals:

- tkn_rcvd: Token received information from MAC to PFC
- dynced_tkn_rcvd: Doubled sync tkn_rcvd, from pclk to hclk domain
- spr_read: Read to SPRAM
- spr_addr: Address to SPRAM
- spr_rdata: Read data from SPRAM
- srcbuf_push: Push to the source buffer
- srcbuf_rdata: Read data from the source buffer. Data seen by MAC

The application can use the following formula to calculate the value of GUSBCFG.USBTrdTim:

4 * AHB Clock + 1 PHY Clock = (2 clock sync + 1 clock memory address + 1 clock memory data from sync RAM) + (1 PHY Clock (next PHY clock MAC can sample the 2-clock FIFO output)

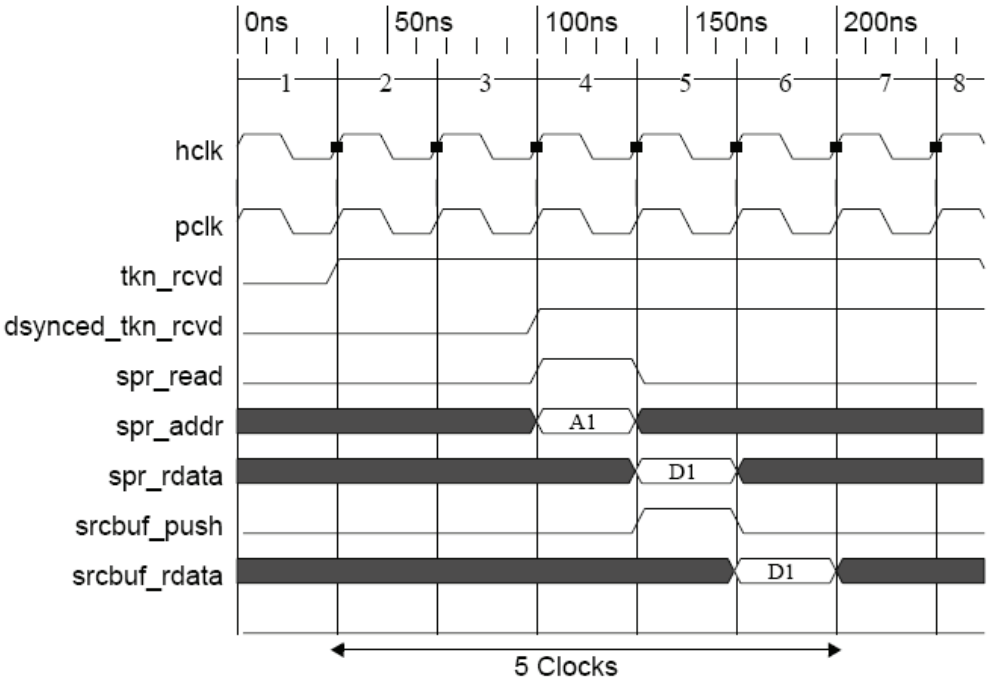


Figure 4.30 USBTrdTim Max Timing Case

4.8 Scatter-Gather DMA Mode

4.8.1 Overview

4.8.2 Scatter/Gather DMA Mode

When the Scatter/Gather DMA mode is enabled data buffers are presented through descriptor structures

1. The application prepares the descriptors, and sets the bit DIEPCTLn/DOEPCTLn.EPEna.

2. DMA fetches the corresponding descriptor (initially determined by DIEPDMAn/DOEPDMAn).

3. DMA internally sets the transfer size from descriptor back to DIEPTSIZn/DOEPTSIZn.

Note The registers DIEPTSIZn/DOEPTSIZn must not be written by the application in Scatter/Gather DMA mode

From this point, the current OTG flow executes. Once the transfer size data is moved by DMA, the DMA checks for further links in the descriptor chain. If this is the last descriptor, the DMA sets the DIOEPINTn.XferCompl interrupt. If there are further active links, the DMA continues to process them. In Scatter/Gather DMA mode, the core implements a true scatter-gather memory distribution in which data buffers are scattered over the system memory. Each endpoint memory structure is implemented as a contiguous list of descriptors, in which each descriptor points to a data buffer of predefined size. In addition to the buffer pointer (1 DWORD), the descriptor also has a status quadlet (1 DWORD). When the list is implemented as a ring buffer, the list processor switches to the first element of the list when it encounters last bit. All endpoints (control, bulk, interrupt, and isochronous) implement these structures in memory.

Note The descriptors are stored in continuos locations. For example descriptor 1 is stored in 32'h0000_0000, descriptor 2 is stored in 32'h0000_0008, descriptor 3 in 32'h0000_0010 and so on. The descriptors are always DWORD aligned.

4.8.3 SPRAM Requirements

For each endpoint the current descriptor pointer and descriptor status are cached to avoid additional requests to system memory. To save gates, these are stored in SPRAM. In addition DIEPDMAn/DOEPDMAn registers are implemented in SPRAM instead of flops.

4.8.3.1 Descriptor Memory Structures

The descriptor memory structures are displayed in Figure 1.31

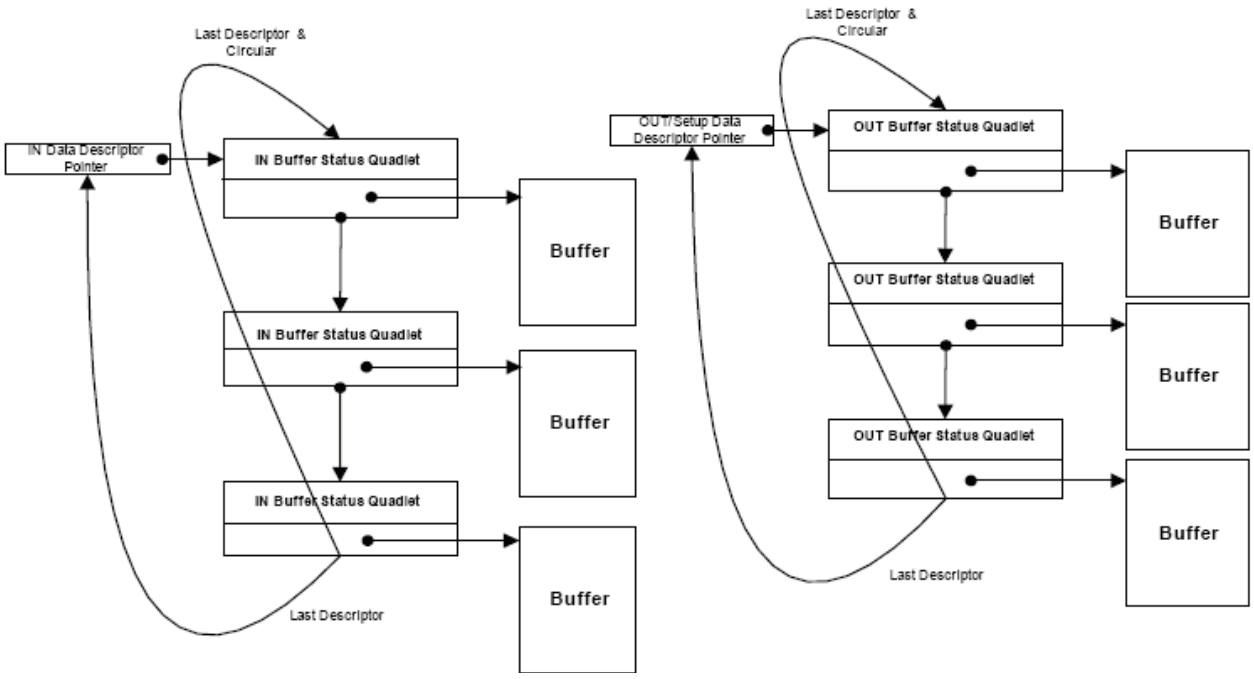


Figure 4.31 Descriptor Memory Structures

4.8.3.2 OUT Data Memory Structure

All endpoints that support OUT direction transactions (endpoints that receive data from the USB host), must implement a memory structure with the following characteristics:

❖ Each data buffer must have a descriptor associated with it to provide the status of the buffer. The buffer itself contains only raw data.

❖ Each buffer descriptor is two quadlets in length.

When the buffer status of the first descriptor is host Ready, the DMA fetches and processes its data buffer; otherwise the DMA optionally skips to the next descriptor until it reaches the end of the descriptor chain. The buffers to which the descriptor points hold packet data for non-isochronous endpoints and frame (FS)/μframe (HS) data for isochronous endpoints.

Host Ready -indicates that the descriptor is available for the DMA to process.

DMA Busy -indicates that the DMA is still processing the descriptor.

DMA Done -indicates that the buffer data transfer is complete.

Host Busy -indicates that the application is processing the descriptor.

The OUT data memory structure is shown in Figure 1.32, which shows the definition of status quadlet bits for non-ISO and ISO end points.

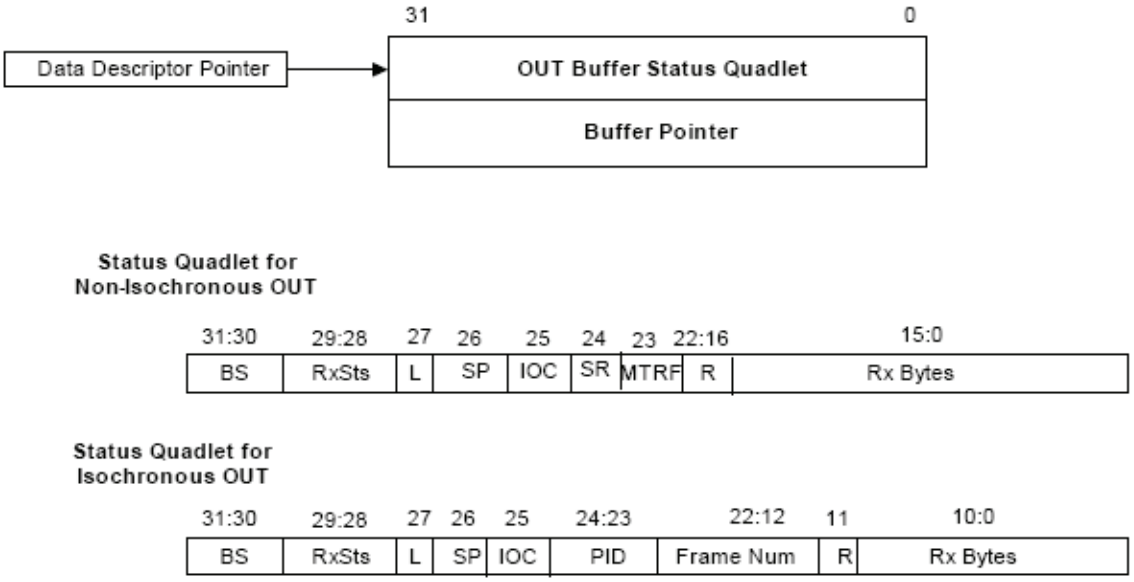


Figure 4.32 Out Data Memory Structure

The status quadlet interpretation depends on the end point type field (DOEPCTLn.EPType) for the corresponding end point. For example, if an end point is OUT and periodic, then the status quadlet is interpreted as Status Quadlet for Isochronous OUT. Table 6-3 displays the OUT Data Memory Structure fields.

Note Note that some fields change depending on the mode

Table 4.14 OUT Data Memory Structure Values

Bit	Name	Descriptiom.	
BS[31:30]	Buffer Status	This 2-bit value describes data buffer status. Possible options are: • 2'b00: Host Ready • 2'b01: DMA Busy • 2'b10: DMA Done • 2'b11: Host Busy Application sets to Host Ready if the descriptor is ready or to Host Busy if the descriptor is not ready. Core sets to DMA busy if the descriptor is being serviced or to DMA Done if the transfer finished associated with the descriptor. The application needs to make these bits as 2'b00 (Host Ready) as a last step after preparing the entire descriptor ready.Once the software makes these bits as Host Ready then it must not alter the descriptor until DMA completes	
Rx Sts [29:28]	Receive Status	This 2-bit value describes the status of the received data. Core updates this when the descriptor is closed. This reflects whether OUT data has been received correctly or with errors. BUFERR is set by the core when AHB error is encountered during buffer access. BUFERR is set by the core after asserting AHBErr for the corresponding end point.The possible combinations are: • 2'b00: Success, No AHB errors • 2'b01: Reserved • 2'b10: Reserved • 2'b11: BUFERR	
L [27]	Last	Set by the application, this bit indicates that this descriptor is the last one in the chain. Note - L Bit is interpreted by the core even when BS value is other than Host ready. For example, BNA is set, the core keeps traversing all the descriptors until it encounters a descriptor whose L bit is set after which the core disables the corresponding endpoint.	
SP[26]	Short Packet	Set by the Core, this bit indicates that this descriptor closed after short packet. When reset it indicates that the descriptor is closed after requested amount of data is received.	
IOC[25]	Interrupt On complete	Set by the application, this bit indicates that the core must generate a transfer complete interrupt(XferCompl) after this descriptor is finished.	
[24]		**Non Isochronous Out Bit**: SR[24] **Bit ID**: Setup Packet Received Set by the Core, this bit indicates that this buffer holds 8 bytes of setup data. There is only one setup packet per descriptor. On reception of a setup packet, the descriptor is closed and the corresponding endpoint is disabled after SETUP_COMPLETE status is seen in the Rx fifo. The core puts a SETUP_COMPLETE status into the Rx FIFO when it sees the first IN/OUT token after the SETUP packet for that particular endpoint. However, if the L bit of the descriptor is set, the endpoint is disabled and the descriptor is closed irrespective of the SETUP_COMPLETE status. The application has to re-enable for receiving any OUT data for the control transfer. (It also need to reprogram the descriptor start address) Note - Because of the above behavior, the core can receive any number of back to back setup packets and one descriptor for every setup packet is used.	**Isochronous Out** **Bit: PID [24:23]** **Bit ID**: ISO Received Data PID Set by the Core. This field is for only high-speed isochronous transactions, and indicates the data PID for an isochronous receive packet. • 2'b00: The packet contained in this descriptor is received with a data PID of DATA0 . • 2'b10: The packet contained in this descriptor is received with a data PID of DATA1. • 2'b01: The packet contained in this descriptor is received with a data PID of DATA2. • 2'b11: The packet contained in this descriptor is received with a data PID of MDATA. For Full-Speed transactions, this field is reserved and the core writes 2'b00.
[23]			

<table>
<tr><td></td><td></td><td>Non Isochronous Out
Bit: MTRF[23]
Bit ID: Multiple Transfer
Set by the application, this bit indicates the Core can continue processing the list after it encountered last descriptor.This is to support multiple transfers without application intervention. Reserved for ISO OUT and Control OUT endpoints.</td><td></td></tr>
<tr><td>[22:16]</td><td></td><td>Non Isochronous Out
Bit: [22:12]
Bit ID: Reserved</td><td rowspan="2">Isochronous Out
Bit: Frame Number [22:12]
Bit ID: Frame number
The 11-bit frame number in which the current ISO-OUT packet is received. For HS, the 11-bit is the concantenation of [7:0] of 1 ms frame number and [2:0] of uframe number, i.e {[7:0],[2:0]}.
For FS, the 11-bit corresponds to full speed frame number.</td></tr>
<tr><td>[15:12]</td><td></td><td rowspan="2">Non Isochronous Out
Bit: Rx Bytes [15:0]
Bit ID: Received number of bytes remaining
This 16-bit value can take values from 0 to (64K-1) bytes, depending on the transfer size of data received from the USB host. The application programs the expected transfer size. When the descriptor is done this indicates remainder of the transfer size. Here, Rx Bytes must be in terms of multiple of MPS for the corresponding end point.
The MPS for the various packet types are as follows:
• Control
• LS - 8 bytes
- FS - 8,16,32,64 bytes
- HS - 64 bytes
• Bulk
- FS - 8,16,32,64 bytes
- HS - 512 bytes
• Interrupt
- LS - up to 8 bytes
- FS - up to 64 bytes
- HS - up to 1024 bytes

Note: In case of Interrupt packets, the MPS may not be a multiple of 4. If the MPS in an interrupt packet is not a multiple of 4, then a single interrupt packet corresponds to a single descriptor. If MPS is a multiple of 4 for an interrupt packet, then a single descriptor can have multiple MPS packets.</td><td>Isochronous Out
Bit: 11
Bit ID: Reserved</td></tr>
<tr><td>[11]</td><td></td><td></td></tr>
<tr><td>[10:0]</td><td></td><td></td><td>Isochronous Out
Bit: Rx Bytes [10:0]
Bit ID: Received number of bytes
This 11-bit value can take values from 0 to (2K-1) bytes, depending on the packet size of data received from the USB host.Application programs the expected transfer size. When the descriptor is done this indicates remainder of the transfer size. The maximum payload size of each ISO packet as per USB specification 2.0 is as follows.
• FS - up to 1023 bytes
• HS - up to 1024 bytes

Note: A Value of 0 indicates zero bytes of data, 1 indicates 1 byte of data and so on.</td></tr>
</table>

Table 4.15 displays the matrix of L bit and MTRF bit options.

Table 4.16 OUT - L Bit and MTRF Bit

L Bit	MTRF bit	Functionality
1	1	Continue to process the list after the last descriptor encountered. Use DOEPDMAn as next descriptor. The Endpoint is not disabled.
1	0	For non-Isochronous endpoints, Stop processing list after last descriptor encountered. The application intervenes and programs the list pointer into DOEPDMAn register when a list is created in a new location otherwise enables the endpoint. Start processing when the endpoint is enabled again with DOEPDMAn register pointing to start of list. For Isochronous endpoints, the DMA engine always goes back to the base descriptor address after the last descriptor.
0	1	After processing the current descriptor go to next descriptor. If a short packet OR zero length packet is received disable the endpoint and a transfer complete interrupt is generated irrespective of IOC bit setting for that descriptor.
0	0	After processing the current descriptor go to next descriptor. If a short packet OR zero length packet is received disable the endpoint and a transfer complete interrupt is generated irrespective of IOC bit setting for that descriptor.

Table 4.17 displays the out buffer pointer field description.
Note For Bulk and Interrupt End Points, if MTRF bit is set for the last descriptor in a list, then all the descriptors in that list need to have their MTRF bit set.

Table 4.18 OUT Buffer Pointer

Bit	Name	Functionality
31:0	Buffer Address	The Buffer pointer field in the descriptor is 32 bits wide and contains the address where the received data is to be stored in the system memory. The starting buffer address must be DWORD aligned. The buffer size must be also DWORD aligned.

4.8.3.3 Isochronous OUT

- ❖ The application must create one descriptor per packet.
- ❖ End point is not disabled by the core based on L bit. The DMA always goes back to the base descriptor address after the last descriptor.
- ❖ The bit MTRF is not applicable.

4.8.3.4 Non-Isochronous OUT

- ❖ The core uses one descriptor per setup packet.
- ❖ The core closes the descriptor after receiving a short packet.
- ❖ Bit combinations for L and MTRF appear in Table 6-4.
- ❖ Multiple Interrupt packets in the same buffer is allowed only if the MPS is multiple of 4.

4.8.3.5 IN Data Memory Structure

All endpoints that support IN direction transactions (transmitting data to the USB host) must implement the following memory structure. Each buffer must have a descriptor associated with it. The application fills the data buffer, updates its status in the descriptor, and enables the endpoint. The DMA fetches this descriptor and processes it, moving on in this fashion until it reaches the end of the descriptor chain. The buffer to which the descriptor points to hold packet data for non-isochronous endpoints and frame (FS)/microframe (HS) data for isochronous endpoints. The definition of status quadlet bits for non-periodic and periodic end points are as shown in the figure 1.33. The status quadlet interpretation depends on the end point type field (DIEPCTLn.EPType) for the corresponding end point. For example, if an end point is IN and periodic, then the status quadlet is interpreted as “Status Quadlet for Isochronous IN”. The IN data memory structure is shown in Figure 1.33.

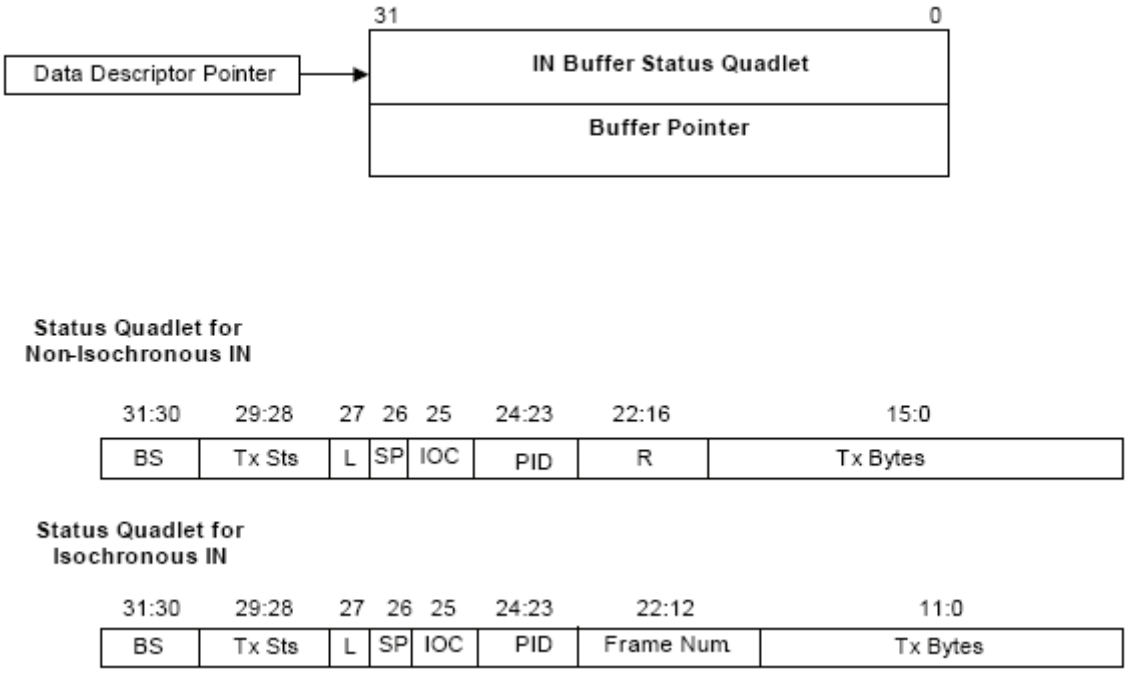


Figure 4.33 IN Data Memory Structure

Table 1.19 displays the IN Data Memory Structure fields.

Table 4.19 IN Data Memory Structure Values

Bit	Name	Descriptiom.	
BS[31:30]	Buffer Status	This 2-bit value describes data buffer status. Possible options are: • 2'b00: Host Ready • 2'b01: DMA Busy • 2'b10: DMA Done • 2'b11: Host Busy The application needs to make these bits as 2'b00 (Host Ready) as a last step after preparing the entire descriptor ready.Once the software makes these bits as HostReady then it must not alter the descriptor until DMA done	
Tx Sts [29:28]	Transmit Status	The status of the transmitted data. This reflects if the IN data has been transmitted correctly or with errors. BUFERR is set by core when there is a AHB error during buffer access. When iIgnrFrmNum is not set, BUFFLUSH is set by the core when • the core is fetching data pertaining to the current frame (N) and finds that the frame has incremented (N+1) during the data fetch • or • when it fetches a descriptor for which the frame number has already elapsed. The possible combinations are: • 2'b00: Success, No AHB errors • 2'b01: Reserved • 2'b10: Reserved • 2'b11: BUFERR Note: Also, irrespective of IgnrFrmNum value, in case of Isochronous IN endpoints, when threshold is enabled, if an underflow occurs when the corresponding buffer data is being accessed, the current descriptor is closed with a BUFFLUSH status.	
L [27]	Last	When set by the application, this bit indicates that this descriptor is the last one in the chain.	
SP[26]	Short Packet	When set, this bit indicates that this descriptor points to a short packet or a zero length packet. If there is more than one packet in the descriptor, it indicates that the last packet is a short packet or a zero length packet.	
IOC[25]	Interrupt On complete	When set by the application, this bit indicates that the core must generate a transfer complete interrupt after this descriptor is finished.	
[24]		**Non Isochronous Out Bit**: SR[24] **Bit** : Reserved[24:16] **Bit ID**: Reserved	**Isochronous Out** **Bit:** PID [24:23] **Bit ID**: Number of packets per frame This 2-bit value indicates the number of packets per µSOF (microframe) for isochronous IN transfers during high-speed operation. The application must program these bits in the descriptor (these bits must be the same for all descriptors of the same µSOF) such that the core returns an isochronous packet with an appropriate data PID per frame. • 2'b00: Reserved. The application should not program this value. • 2'b01: 1 packet per microframe, Data0 is the starting PID • 2'b10: 2 packets per microframe, Data1 is the starting PID • 2'b11: 3 packets per microframe, Data2 is the starting PID These bits are reserved for full-speed operation.
[24:23]		**Non Isochronous Out** **Bit**: MTRF[23] **Bit ID**: Multiple Transfer Set by the application, this bit indicates the Core can continue processing the list after it encountered last descriptor.This is to support multiple transfers without application intervention. Reserved for ISO OUT and Control OUT endpoints.	
[22:12]			

[15:12]		**Non Isochronous In** **Bit**: Tx bytes [15:0] **Bit ID**: Number of bytes to be transmitted This 16-bit value can take values from 0 to (64K-1) bytes, indicating the number of bytes of data to be transmitted to the USB host. **Note**: A Value of 0 indicates zero bytes of data, 1 indicates 1 byte of data and so on	**Isochronous In** **Bit:** Frame Number [22:12] **Bit ID**: Frame Number Frame number in which the current packet must be transmitted.The 11-bit is the concantenation of [7:0] of 1 ms frame number and [2:0] of uframe number i.e {[7:0],[2:0]}. For FS, this field must correspond to the 11-bit full speed frame number.
[11:0]			**Isochronous In Bit:** **Tx bytes [11:0]** **Bit ID**: Number of bytes to transmit Tx bytes [11:0] Number of bytes to be transmitted This 12-bit value can take values from 0 to (4K-1) bytes, indicating the number of bytes of data to be transmitted to the USB host. **Note**: A Value of 0 indicates zero bytes of data, 1 indicates 1 byte of data and so on.

Table 4.201 displays the matrix of L bit and MTRF bit options.

Table 4.21 IN - L Bit, SP Bit and MTRF Bit

L Bit	SP bit	Tx Bytes	Functionality
0	1	Multiple of endpoint maximum packet size	Transmit a zero length packet after the last packet
0	1	Not multiple of maximum packet size	Send short packet at the end after normal packets are sent out. Then move onto next descriptor
0	1	0	Transmit zero length packet. Then move on to next descriptor.
0	0	Multiple of endpoint maximum packet size	Send normal packets and then move to next descriptor.
0	0	Not a multiple of maximum packet size	Transmit the normal packets and concatenate the remaining bytes with next buffer from the next descriptor. This combination is valid only for bulk end points.
0	0	0	
1	1	Multiple of endpoint maximum packet size	Transmit a zero length packet after the last packet If this IN descriptor is for a ISO endpoint, then move onto the first descriptor in the list. If this IN descriptor is for a non-ISO endpoint, then stop processing this list and disable the corresponding end point.
1	1	Not multiple of maximum packet size	Send short packet after sending the normal packets If this IN descriptor is for a ISO endpoint, move onto the first descriptor in the list. If this IN descriptor is for a non-ISO endpoint, then stop processing this list and disable the corresponding end point.
1	1	0	Transmit zero length packet

			If this IN descriptor is for a ISO endpoint, move onto the first descriptor in the list. If this IN descriptor is for a non-ISO endpoint, then stop processing this list and disable the corresponding end point.
1	0	Multiple of endpoint maximum packet size	Send normal packets If this IN descriptor is for a ISO endpoint, Move onto the first descriptor in the list after current transfer done. If this IN descriptor is for a non-ISO endpoint, then stop processing the list and disable the corresponding end point.
1	0	Not multiple of maximum packet size.	Invalid. The behavior of the core is undefined for these values.
1	0	0	invalid. The behavior of the core is undefined for these values.

The descriptions provided for the different combinations in Table 6-7 depend on the previous descriptor L, SP, and Tx Bytes values. Consider table 1.22

Table 4.22 IN – Buffer Pointer

DESC NO	L bit	SP bit	Txbytes	Description
1	0	0	520	Send a normal packet of size 512, and concatenate the remaining 8 bytes with the next descriptor's buffer data
2	0	1	512	For this combination of L,SP and TxBytes, as per the above table, we need to send a zero length packet instead of a short packet. However, a normal packet followed by a short packet of length 8-bytes is sent. This is to illustrate the context dependency based on previous descriptor L,SP and TxByte combinations.

Table 1.23 displays the IN buffer pointer field description.

Table 4.23 IN Buffer Pointer

Bit	Bit ID	Description
31-0	Buffer Address	The Buffer pointer field in the descriptor is 32 bits wide and contains the address where the transmit data is stored in the system memory. The address can be non- DWORD aligned.

4.8.3.6 Descriptor Update Interrupt Enable Modes

If IOC bit is set for a descriptor and if the corresponding Transfer Completed Interrupt Mask (XferComplMask) is unmasked, this interrupt (DIOEPINTn.XferCompl) is asserted while closing that descriptor.

4.8.3.7 DMA Arbitration in Scatter/Gather DMA Mode

The arbiter grants receive higher priority than transmit. Within transmit, the priority is as follows.

- ❖ The highest priority is given to periodic endpoints. The periodic endpoints are serviced in a round robin fashion.
- ❖ The non periodic endpoints are serviced after the periodic scheduling interval has elapsed. The duration of the periodic scheduling interval is programmable, as specified by register bits DCFG[25:24]. When the periodic interval is active, the periodic endpoints are given priority.
- ❖ Amongst the periodic endpoints, the priority is round robin.
- ❖ Amongst the non periodic endpoints, the Global Multi Count field in the Device Control Register (DCTL) specifies the number of packets that need to be serviced for that end point before moving to the next endpoint.

The arbiter disables an endpoint and moves on to the next endpoint in the following scenarios as well, for all the endpoint types:

- ❖ Descriptor Fetch and AHB Error occurs.
- ❖ Buffer Not Available (BNA), such as when buffer status is Host busy.
- ❖ AHB Error during Descriptor update stage and Data transfer stage.

4.8.3.8 Buffer Data Access on AHB in Scatter/Gather DMA Mode

The buffer address whose data needs to be accessed in the system memory can be non DWORD aligned for transmit.
For buffer data read, the core arranges the buffer data to form a quadlet internally before populating the TXFIFO within the core as per the following scenarios

- ❖ The packet starts in a non DWORD aligned address, the core does two reads on AHB before appending the relevant bytes to form a quadlet internally. Hence the core stores the bytes before pushing to the TXFIFO.
- ❖ The packet ends in a non DWORD aligned address and it is not the end of the buffer or expected transfer, the core may switch to service another end point and come back to service the initial end point. In this case, the core reads the same DWORD location again and then samples only the relevant bytes. This eliminates the storage of the bytes for the initial end point.

For buffer data write, the core always performs DWORD accesses.

4.8.4 Control Transfer Handling

Control transfers (3-Stage Control R/WR or 2-Stage), can be handled effectively in the Descriptor-Based Scatter/Gather DMA mode by following the procedure explained in this section. By following this procedure the application is able to handle all normal control transfer flow and any of the following abnormal cases.

- ❖ More than one SETUP packet (back to back)□ Host could send any number of SETUP packets back to back, before sending any IN/OUT token. In this case, the application is suppose to take the last SETUP packet, and ignore the others.
- ❖ More OUT/IN tokens during data phase than what is specified in the wlength field□ If the host sends more OUT/IN data tokens than what is specified in the wlength field of the SETUP data, then the device must STALL.
- ❖ Premature SETUP packet during data/status phase□ Device application must be able to handle this SETUP packet and ignore the previous control transfer.
- ❖ Lost ACK for the last data packet of a Three-Stage Control Read Status Stage.

4.8.5 Interrupt Usage for Control Transfers

The application checks the following OUT interrupts status bits for the proper decoding of control transfers.

- ❖ DIEPINTn.XferCompl (Transfer complete, based on IOC bit in the descriptor)
- ❖ DIEPINTn.InTknTxfEmp (In token received when Tx FIFO is empty)
- ❖ DOEPINTn.XferCompl (Transfer complete, based on IOC bit in the descriptor)
- ❖ DOEPINTn.SetUp (Setup Complete interrupt, generated when the core receives IN/OUT token after a SETUP packet.
- ❖ DOEPINTn.StsPhseRcvd (Status phase received interrupt (Also called SI), generated when host has switched to status phase of a Control Write transfer).

The core performs some optimization of these interrupt settings, when it sees multiple interrupt bits need to be set for OUT endpoints. This reduces the number of valid combinations of interrupts and simplifies the application.

The core gives priority for DOEPINTn.XferCompl over DOEPINTn.SetUp and DOEPINTn.StsPhseRcvd (SI) interrupts. When setting the XferCompl interrupts, it clears the SetUP and SI interrupt bits.

- ❖ The core gives priority to DOEPINTn.SI interrupt over DOEPINTn.SetUp. When setting DOEPINTn.StsPhseRcvd (SI), the core clears DOEPINTn.SetUp interrupt bit.

Based on this, the application needs only to decode the combinations of interrupts for OUT endpoints shown in Table 1.24

Table 4.24 IN – Buffer Pointer

StsPhseRcvd (SI)	SetUp (SPD)	XferComl (IOC)	Description	Template Used
0	0	1	Core has updated the OUT descriptor. Check the “SR”(Setup Received) bit in the descriptor to see if the data is for a SETUP or OUT transaction.	Case A
0	1	0	Setup Phase Done Interrupt for the previously decoded SETUP packet.	Case B
0	1	1	The core has updated the OUT descriptor for a SETUP packet, and the core is indicating a SETUP complete status also.	Case C
1	0	0	Host has switched to Status phase of a Control OUT transfer	Case D
1	0	1	Core has updated the OUT descriptor. Check the SR”(Setup Received) bit in the descriptor to see if the data is for a SETUP or OUT transaction. Also, the host has already switched to Control Write Status phase.	Case E

4.8.6 Application Programming Sequence

This section describes the application programming sequence to take care of normal and abnormal Control transfer scenarios.
All the control transfer cases can be handled by five separate descriptor lists. The descriptor lists are shown in Figure 1.34

- ❖ Three lists are for SETUP. The SETUP descriptors also take data for the Status stage of Control Read.
- ❖ The first two (index 0 and 1) act in a ping-pong fashion.
- ❖ The third list is an empty list, linked to one of the OUT descriptors when premature SETUP comes during the data/status phase.
- ❖ Two lists are for IN and OUT data respectively.
- ❖ Figure 6-39 on page 445 displays setup_index 0, 1, and 2 as elements of array of pointers called setup_index. The first two elements of this array point to SETUP descriptors. The third element of this array is initially a NULL pointer, but is eventually linked to a SETUP descriptor. These array elements could also point to a descriptor for Control Read Status phase.

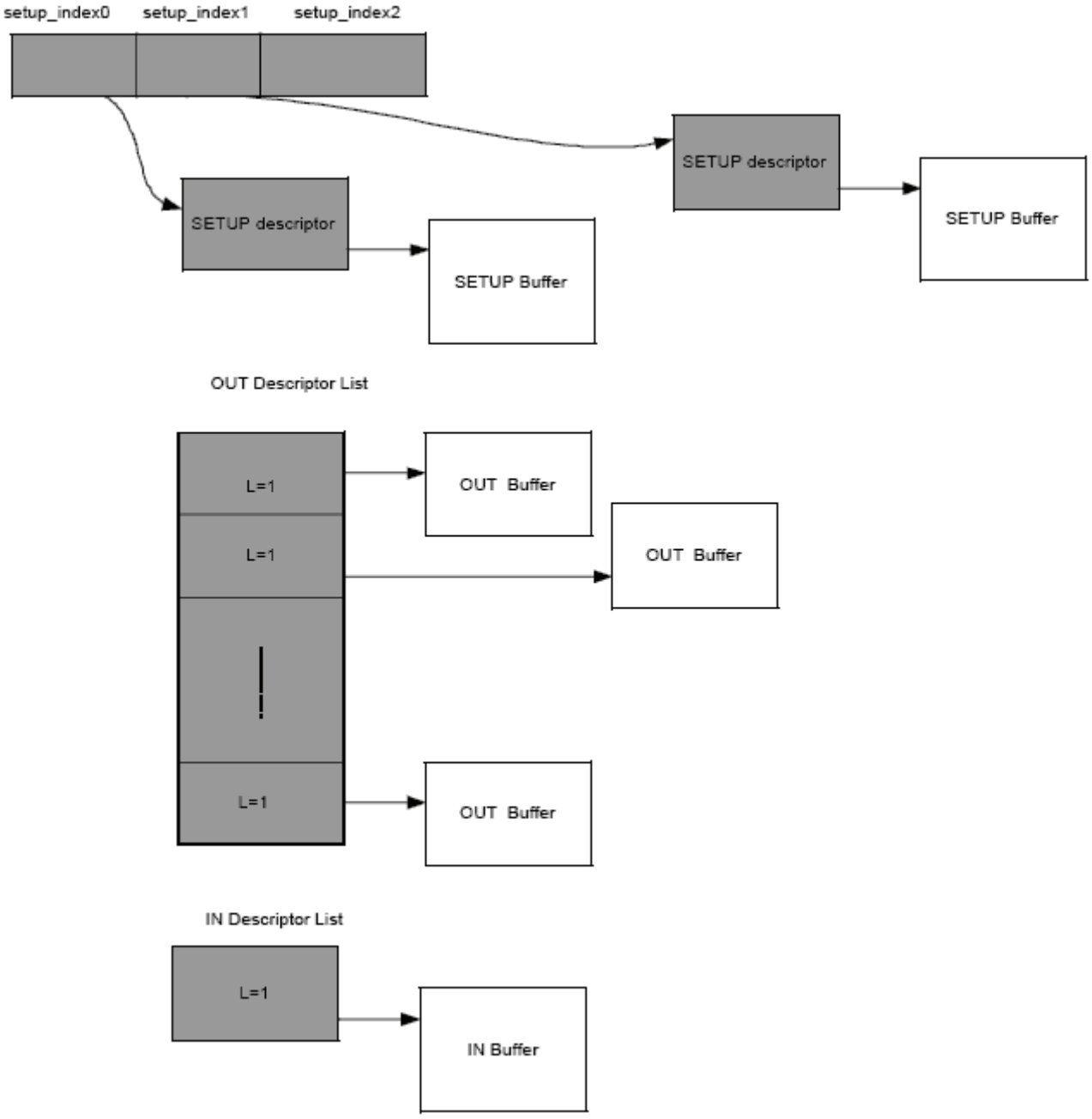


Figure 4.34 Descriptor Lists for Handling Control Transfers

The following are the steps that need to be followed by the application driver.

1. **Set up Desc for SETUP/Ctrl-Rd-Sts -**Setup2 descriptor lists in memory for taking in SETUP packets. Each of this list must have only one descriptor, with the descriptor fields set to the following

   - Rx_bytes Set it to Max packet size of the control endpoint.
   - IOC =1.
   - MTRF=0.
   - L=1.

2. **Enable DMA** If current setup_index =0, then setup_index=1. The application ping-pongs between these two descriptors. Program the address of the current setup descriptor (specified by setup_index) to DOEPDMAn. Write to DOEPCTLn with the following fields.

   - DOEPCTL.MPS Max Packet size of the endpoint
   - DOEPCTL.EPEna Set to 1 to enable the DMA for the endpoint.

3. **Wait for Interrupt-**Wait for OUT endpoint interrupt (GINTSTS.OEPInt). Then read the corresponding DOEPINT.

4. If Control Read Data Stage in progress

- Case A: Check SR bit (In this case SR bit is set, because the host cannot send OUT at this point. If it sends OUT it is NAKed. **GOTO** Step 24.
- Case B: **GOTO** Step 26.
- Case C:-Check SR bit (In this case SR bit is set because host cannot send OUT packets without SETUP at this stage). **GOTO** Step 24.
- Case D: Cannot happen at this stage because SI cannot come alone without a SETUP, at this stage.
- Case E: Indicates that host has switched to another SETUP (Three-Stage control write) and then has switched to status phase without and data phase (core clears SUP with SI in this case). Decode SETUP packet and if ok, **GOTO** Step 11. else If Ctrl Write Status Stage in progress OR Two-Stage Status Stage in progress
- Case A: Check SR bit (In this case SR bit is set, because the host cannot send OUT at this point. If it sends OUT it is NAKed.) **GOTO** Step 24.
- Case B: (Could happen for Two-Stage Ctrl Transfer.) **GOTO** Step 26.
- Case C: **GOTO** Step 24.
- Case D: Clear SI interrupt and wait Step 3.
- Case E: Cannot happen at this stage. *else*
- Case A: **GOTO** Check Desc.
- Case B: Normally, this does not occur at this stage. Either IOC comes first or IOC comes with SUP (Case C).
- Case C: **GOTO** Check Desc.
- Case D: Cannot happen at this point.
- Case E: If SR==1, Indicates Three stage control Transfer SETUP and that the host has switched to status phase. Decode the SETUP packet and **Goto** Step 11.
- Check Desc Read the Descriptor status quadlet corresponding to the setup_index and check the SR field. (Application might also want to check the BS and RxSts fields and take necessary actions if there is any abnormalities). If SR field is 1 **GOTO** Step 5 (If Step Step 20 is active, terminate it). If SR field is 0 **GOTO** Step 22 (Control Rd Status phase) (This must also terminate Step 20).

5. **Decode SETUP-**Decode the SETUP packet. If it is a Three-Stage Control Write, **GOTO** Step 20. If it is a Three-Stage Control Read, **GOTO** Step 15. If it is a Two-Stage Control transfer, **GOTO** Step 11 (Same as Status stage for 3-Stage Control Write).

6. **Desc list for Ctrl Wr data—**Setup descriptor list for Control write data phase. This must be based on the Wlength field in the SETUP data. The descriptors in the list must be setup such that there must be one descriptor per packet. Each of these descriptors must have the control fields set as follows.

- Rx_Bytes: Set to the Max Packet Size of the control Endpoint.
- IOC = 1
- MTRF = 0.
- L=1.
- At this point we are not enabling and clearing the NAK for the IN endpoint for status phase.This is because, the status phase for Control Write can be ACKed only after decoding the complete data for the data phase. **GOTO** Step 7.

7. **Enable DMA for Ctrl Wr Data:** Write the start address of this list to DOEPDMAn. Program the DOEPCTLn with the following bits set

- DOEPCTL.MPS: Max Packet size of the endpoint
- DOEPCTL.EPEna: Set to 1 to enable the DMA for the endpoint. **GOTO** Step 8.

8. **Wait for Ctrl Wr Data Interrupt:** Wait for OUT endpoint interrupt (GINTSTS.OEPInt). Then read the corresponding DOEPINTn.

- Case A: check the SR field. Also clear DOEPINTn.XferCompl by writing to DOEPINTn.(Application might also want to check the BS and RxSts fields and take necessary actions if there is any abnormalities). If SR field is 0 **GOTO** Step 9.If SR field is 1, **GOTO** Step 23**.** (This indicates that the host has switched to a new control transfer).
- Case B: **GOTO** Step 25.
- Case C: **GOTO** Step 23**.** (This indicates that the host has switched to a new control transfer).
- Case D: Host has switched to status phase. Decode the data received so far. **GOTO** Step 10.

- ✦ Case E Check SR bit. If SR==0, decode the data received so far. GOTO Step 10. If SR==1, decode the SETUP packet and Goto Step10.

9. **Check Desc-**If it's not the last packet of data phase, Re-enable the endpoint and clear the Nak. This is because the core sets NAK after receiving each OUT packet for control write data phase. This is to allow application to STALL in case the host sends more data than what is specified in the Wlength field. **GOTO** Step 8. Re-enabling and clearing the NK involves the following steps.

- ✦ Write to DOEPDMA with the new descriptor address.
- ✦ Write to DOEPCTLn with the following fields.
  - ✧ DOEPCTL.MPS Max Packet size of the endpoint
  - ✧ DOEPCTL.CNAK Set to 1 to clear the NAK.
  - ✧ DOEPCTL.EPEna Set to 1 to enable the DMA for the endpoint. If it is the last packet of the data phase, **GOTO** Step 10.

10. **STALL Extra Bytes—**Write to DOEPCTLn with Stall set so that the core could STALL any further OUT tokens from host.If the received Bytes so far is greater than what is specified in Wlength field OR is there were any unsupported commands in the data phase, then write to DIEPCTLn with the Stall bit set so that the Status phase could be Stalled.(The STALL bit is automatically cleared by the core with the next SETUP). **GOTO** Step 11.

11. **Disc list for Ctrl Wr Sts—**The following two process must run in parallel. This is because, we are preparing for the status phase (IN) of Control write but at the same time the host could send another SETUP. So IN and OUT descriptor list must be ready. a. Do Step 2— Step 5 (This is for handling SETUP or Ctrl Wr Status). If the OUT DMA is already enabled (OUT DMA was enabled for data phase of Three-Stage Control Write, but there was a premature status phase), GOTO Step 3. b. Setup descriptor list for Status phase IN, depending on the data in the status phase. Normally it is always a zero length packet. c. Tx_Bytes—Size of status phase, d. BS—Host Ready, e. L=1. f. IOC=1. g. SP=1 (Depending on the Tx_Bytes). h. Write to DIEPDMAn with the start address of the descriptor.Write to DIEPCTLn clear the NAK and enable the endpoint. Flush the corresponding TX FIFO. i. If SI has not been received in the data stage prior to the status stage, then wait for SI before clearing the NAK(DIEPCTLn.CNAK=1) j. DIEPCTLn.EpEna=1. k. **GOTO** Step 12.

12. **Wait for Interrupt—**Wait for IN endpoint interrupt (GINTSTS.IEPInt).

13. If IN endpoint INterrupt, and DIEPINTn.XferCompl, then **GOTO** Step 14.

14. **Check Desc—**Read the Status field of the descriptor. Check Tx_bytes in the descriptor.(Application might also want to check the BS and RxSts fields and take necessary actions if there is any abnormalities). This is end of Three-Stage Control Write OR Two-Stage Control transfer. We are now ready for the next control transfer (Already taken care by process "a" is Step 11**.**

15. **Desc for Ctrl Rd Data—**The following two steps must be run in parallel. This is because, we are preparing for Data phase of Control read, but at the same time, the host could abnormally abort this control transfer and send a SETUP, or switch to status phase.

16. Do Step 2— Step 5 (This is for handling SETUP and also Control Read Status phase.).

17. Setup descriptor list for Data phase IN, depending on the WLength field in the SETUP data. You could setup single descriptor OR multiple descriptors. If it is multiple descriptors, ensure that IOC for the last descriptor is set.

- ✧ Tx_Bytes Size of data phase (Wlength field).
- ✧ BS Host Ready ✧ L=1. ✧ IOC=1. It is mandatory to set the IOC when it is the last descriptor.
- ✧ SP=1 (Depending on the Tx_Bytes). ✧ Write to DIEPDMAn with the start address of the descriptor list.
- ✧ Write to DIEPCTLn clear the NAK and enable the endpoint.
- ✧ Flush the corresponding TX FIFO.
- ✧ DIEPCTLn.MPS = Max_packet size of the endpoint,
- ✧ DIEPCTLn.CNAK=1 only if SPD already set (Case C in Step 3).
- ✧ Also set the DOEPCTLn.CNAK for the corresponding OUT endpoint after SPD because a premature status stage (OUT) can come which must be acked.
- ✧ DIEPCTLn.EpEna=1.
- ✧ **GOTO** Step 18.

18. **Wait for Interrupt** Wait for IN endpoint interrupt (GINTSTS.IEPInt)

19. If IN endpoint interrupt, read the corresponding DIEPINTn and if XferCompl is set **GOTO** Step 20.

20. **Check_Desc**—Wait for the DIEPINTn.IOC interrupt. Go to Step 21.

21. **Set_Stall—**Write to DIEPCTLn with STALL bit set. (The STALL bit is automatically cleared by the core with the next SETUP). The function of this process initiated in step Step 15 is over, and must be terminated. The next control transfer is already taken care by the process that is running from Step 2.

22. **Ctrl Rd Sts Desc Check—**Read the descriptor to check the Rxbytes and also check the SP field. The Three-Stage control Read is complete here. **GOTO** Step 2, in preparation for the next SETUP.

23. The unexpected SETUP packet now received during the control write data phase, is sitting in the descriptor allocated for Data. Link this to the setup descriptor pointer. setup_desc_index = 2. Point setup_desc_index to the current OUT descriptor (which has the SETUP). **GOTO** Step 5.

24. Disable IN Endpoint DMA. Core flushes the corresponding Tx FIFO in order to flush the data that was meant for Control Write Status phase OR Control Read data phase. If Step 12 or Step 18 is active, terminate it. **GOTO** Step .

25. Read Modify write DOEPCTLn to clear the NAK. Then **GOTO** Step 8 again.

- ✦ DOEPCTLn.CNAK Set to 1 to clear the NAK.

26. Read Modify write DIEPCTLn to clear the NAK. Then **GOTO** Step 3 again.

- ✦ DIEPCTLn.CNAK Set to 1 to clear the NAK.

27. Read Modify write DIEPCTLn to clear the NAK. Then Step 12 again

- ✦ DIEPCTLn.CNAK Set to 1 to clear the NAK.
- ✦ DOEPCTLn.CNAK:Set to 1 to clear the NAK for the out endpoint. This clears the NAK to accept status stage data in case of control read.

4.8.7 Internal Data Flow

This section explains the cores internal data flow for control transfers.

4.8.7.1 Three-Stage Control Write

Figure 6-40 on page 451 displays the core behavior for Three-Stage control write transfers.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint. Additionally, the clearing of the NAK bit is blocked by the core until the following SPD or SI is read by the application and cleared.

2. The DMA detects the RxFIFO as non-empty and does the following:

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Application clears NAK for the data phase, after receiving DOEPINTn.SetUp interrupt.

7. The core ACKs and the next OUT token because the NAK has been cleared (provided there is enough space in the RxFIFO.

8. DMA detects the OUT packet in RxFIFO and starts transferring the OUT packet to the system memory.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the OUT packet from the RxFIFO to the buffer pointed by the descriptor.

- ✦ Close descriptor with DMA_DONE status.

9. The core NAKs the next OUT token because the core internally sets the NAK after every control write data phase packets. This is to allow application to Stall any extra tokens.

10. The core generates DOEPINT.XferCompl after closing the OUT descriptor (Step 8).

11. Application clear NAK on receiving DOEPINTn.XferCompl interrupt.

12. Host starts the Status phase by sending the IN token which is NAKed by the core. The core push DATA_PHASE_DONE status into the RxFIFO.

13. The core generates DOEPINTn.XferCompl for the last OUT packet transfer to system memory.

14. The core generates DOEPINT.StsPhsRcvd interrupt after the DMA has popped the DATA_PHASE_DONE status from the RxFIFO.

15. Application clears the NAK and enables the IN endpoint for status phase.

16. The core starts fetching the data for the Status phase

- ✦ Fetch the descriptor pointed by DIEPDMA.
- ✦ Fetch the packet (if size >0) to Tx fifo.
- ✦ Close the descriptor with DMA_DONE status
- ✦ The core generates DIEPINTn.XferCompl interrupt after closing the descriptor.

17. The core sends out data in response to the Status Phase IN token.

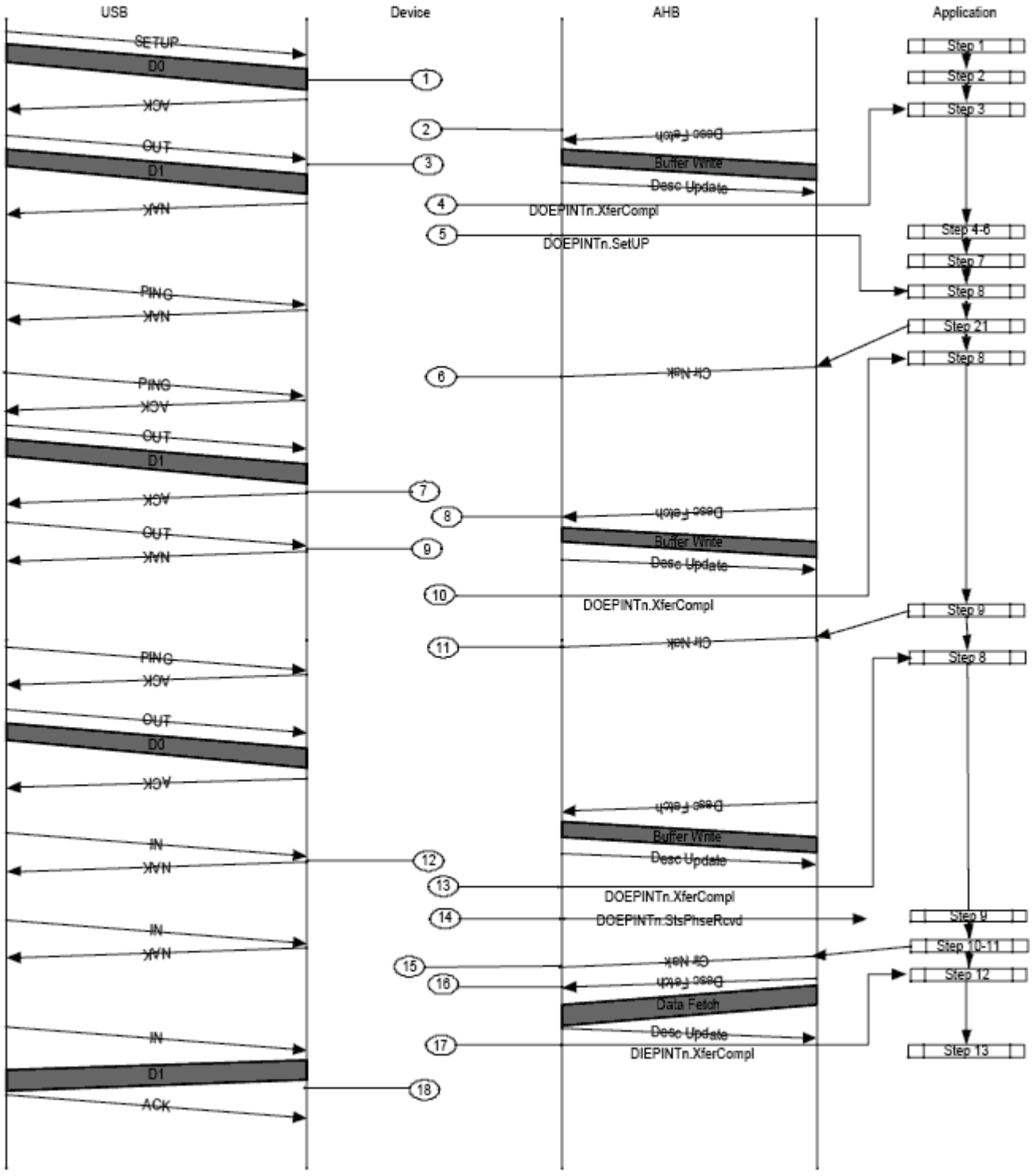


Figure 4.35 Three-Stage Control Write

In this example, it is assumed that the data phase consists of 2 packets, and the application allocates these two packets in a single buffer.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.

- ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Data phase IN tokens are NAKed until this point because the NAK has not yet been cleared by the application.

7. The core starts fetching the IN data after the application enables IN DMA (In this example it is assumed that multiple packets are in the same buffer. But it could also be in different buffers). This involves the following steps

- ✦ Fetch the descriptor pointed by DIEPDMA.
- ✦ Fetch the data into the corresponding Tx FIFO.
- ✦ Close the descriptor with DMA_DONE status...s

8. The application clears the NAK after receiving the setup complete (DOEPINT.SetUp) interrupt. The application also clears NAK of the OUT End point to accept the status phase.

9. After all the data has been fetched for the descriptor (Step 7), core generates DIEPINT.XferCompl interrupt.

10. The core sends data in response to the IN token for the data phase.

11. The core sends out the last packet of the IN data phase.

12. The core ACKs the status phase.

13. The core generates DOEPINTn.XferCompl interrupt after transferring the data received for the status phase to system memory.

Two-Stage Control Transfer

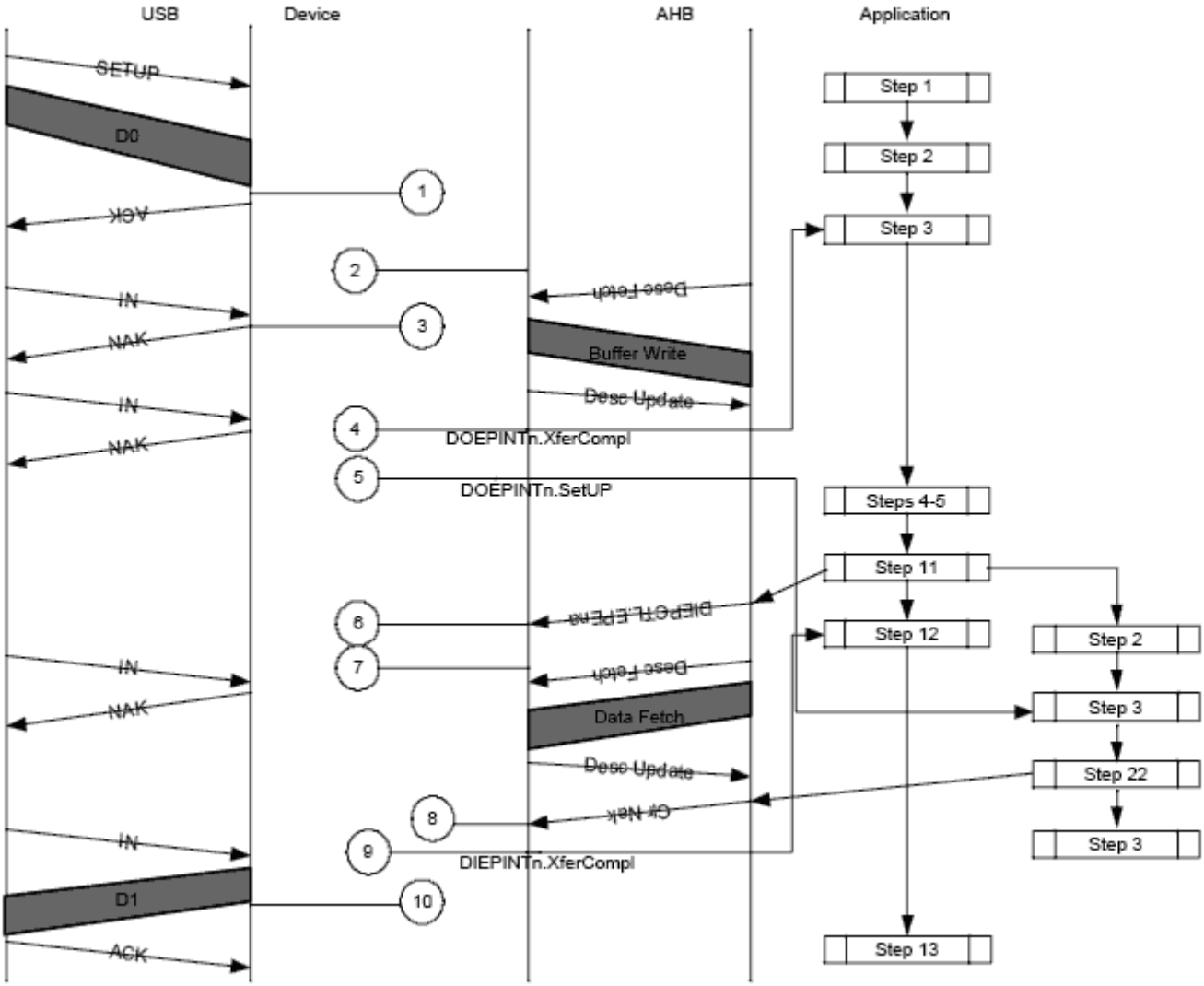


Figure 4.36 Two-Stage Control Write

This example shows the core behavior for a Two-Stage Control transfer.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.
   - ✦ The core receives the status phase IN token, which it NAKs. Core also pushes SETUP_COMPLETE status into the Rx FIFO.
   - ✦ The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2)
   - ✦ The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

3. Application enables the IN endpoint for status phase.

4. The core starts fetching the descriptor for IN endpoint.

5. Application clears the NAK for IN endpoint after getting the DOEPINTn.SetUP interrupt (Step 4).

6. The core generates DIEPINTn.XferCompl after updating the descriptor after IN data fetch.

7. The core sends out data for the status phase IN token from host.

4.8.7.2 Back to Back SETUP During Control Write

This example shows the core receiving 2 Back to Back SETUP tokens fo3 Three-Stage Control write.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

- ✦ Fetch the descriptor pointed by DOEPMA.
- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.
- ✦ The core receives another SETUP, and pushes the data into the Rx FIFO, also sets the NAK.
- ✦ The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The DMA detects the RxFIFO as non-empty (because of the 2nd SETUP packet) and does the following.

- ✦ Fetch the descriptor pointed by DOEPMA.
- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.
- ✦ The core generates DOEPINT.XferCompl interrupt after having transferred the second SETUP packet into memory (Step 6)
- ✦ The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

5. Application clears NAK for the data phase, after receiving DOEPINTn.SetUp interrupt.

6. The core ACKs the next OUT/Ping token after the NAK has been cleared by the application.

7. The DMA detects the RxFIFO as non-empty (because of the OUT packet) and does the following.

- ✦ Fetch the descriptor pointed by DOEPMA.
- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.
- ✦ The core generates DOEPINT.XferCompl interrupt after having transferred the OUT packet into memory (Step 11) and closing the descriptor.

The remaining steps are similar to Steps 11-18 of “Application Programming Sequence”. This example shows the core behavior for a Two-Stage Control transfer

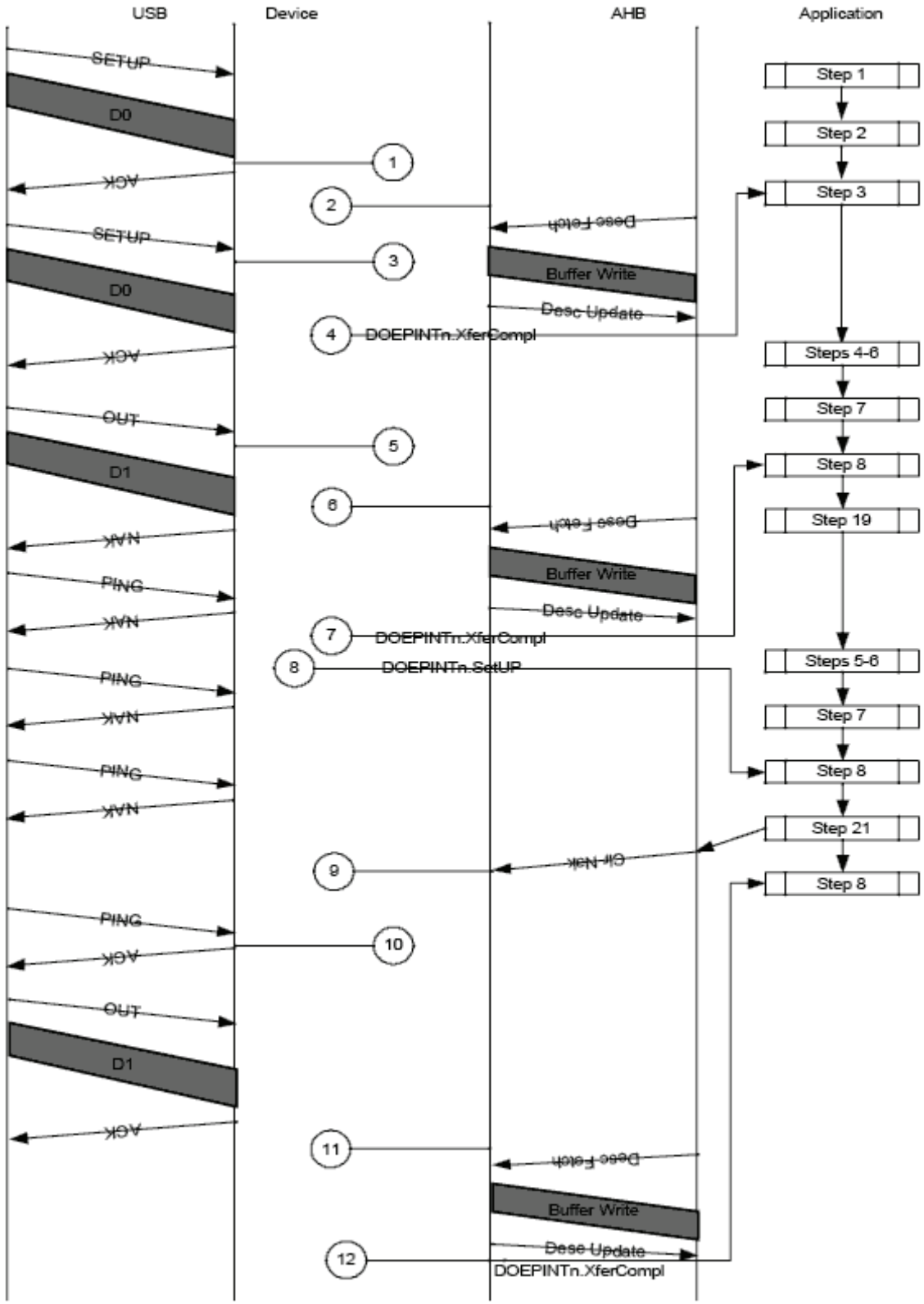


Figure 4.37 Back-to-Back SETUP Packet Handling During Control Write

4.8.7.3 Back-to-Back SETUPs During Control Read

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

- ✦ Fetch the descriptor pointed by DOEPMA.
- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.
- ✦ The core receives another SETUP, and pushes the data into the Rx FIFO, also sets the NAK.

3. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

4. Host sends IN token for the data phase which is NAKed by the core, because NAK is set in Setp3. The core pushes SETUP_COMPLETE status into RxFIFO.

5. After the application has re-enabled the OUT DMA (Application flow Step 2) core detects RxFIFO as non-empty because of the second SETUP packet and does the following.

- ✦ Fetch the descriptor pointed by DOEPMA.
- ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
- ✦ Close the descriptor with DMA_DONE status.
- ✦ The core generates DOEPINTn.XferCompl interrupt after having transferred the SETUP packet into memory (Step6).
- ✦ The core starts fetching data for IN endpoint because the IN endpoint was enabled by application in Step-14.

6. On seeing DOEPINTn.XferCompl (Step 7) and finding that it is a SETUP packet, application disables the endpoint in Step 20.

7. The core generates DOEPINTn.SetUP (Setup complete) interrupt after popping the SETUP_COMPLETE status from the RxFIFO.\

8. The core generates endpoint disabled interrupt (as a result of application setting disable bit in step 9).

9. The core generates DIEPINTn.XferCompl after completing the IN data fetch and updating the descriptor.

10. application clears NAK after seeing setup_complete interrupt (generated in Step 10.

The flow after this is same as steps 9 - 13 of “Internal Data Flow”

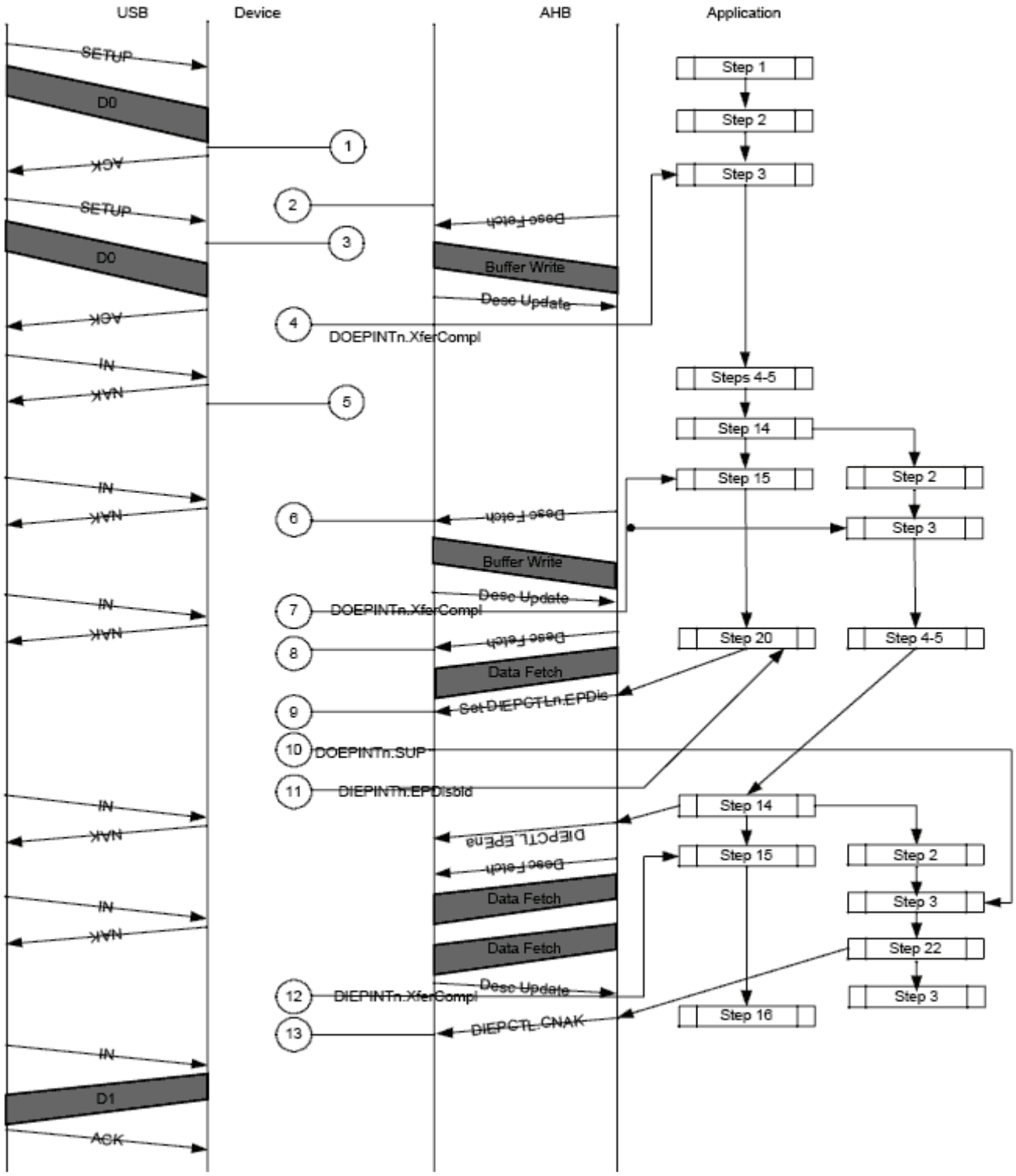


Figure 4.38 Back-to-Back SETUP During Control Read

4.8.7.4 Extra Tokens During Control Write Data Phase

This example assumes a three-stage control write transfer with only Wlength field in the SETUP indicating only 1 packet in the data phase. But the host sends an additional OUT packets which the core STALLs.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2)
.
5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Application clears NAK for the data phase, after receiving DOEPINTn.SetUp interrupt.

7. The core ACKs and the next OUT token because the NAK has been cleared (provided there is enough space in the RxFIFO.

8. DMA starts transferring the OUT packet to the system memory.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the OUT packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close descriptor with DMA_DONE status.
   - ✦ The core generates DOEPINTn.XferCompl interrupt after having transferred the OUT packet to the system memory. Since there were only one packet in the data phase, the data phase is complete here.
   - ✦ The core initially NAK s the extra tokens send by the host, because the core internally sets NAK after each OUT packet for the data phase of control write.

9. Application sets STALL to stall any extra tokens.

10. The core stalls the next OUT/PING token.

11. Host switches to next control transfer, core ACKs the SETUP. This SETUP packet is transferred to the system memory buffer originally allocated for Status phase.

12. The core generates DOEPINTn.XferCompl interrupt after transferring the SETUP packet to the system memory.

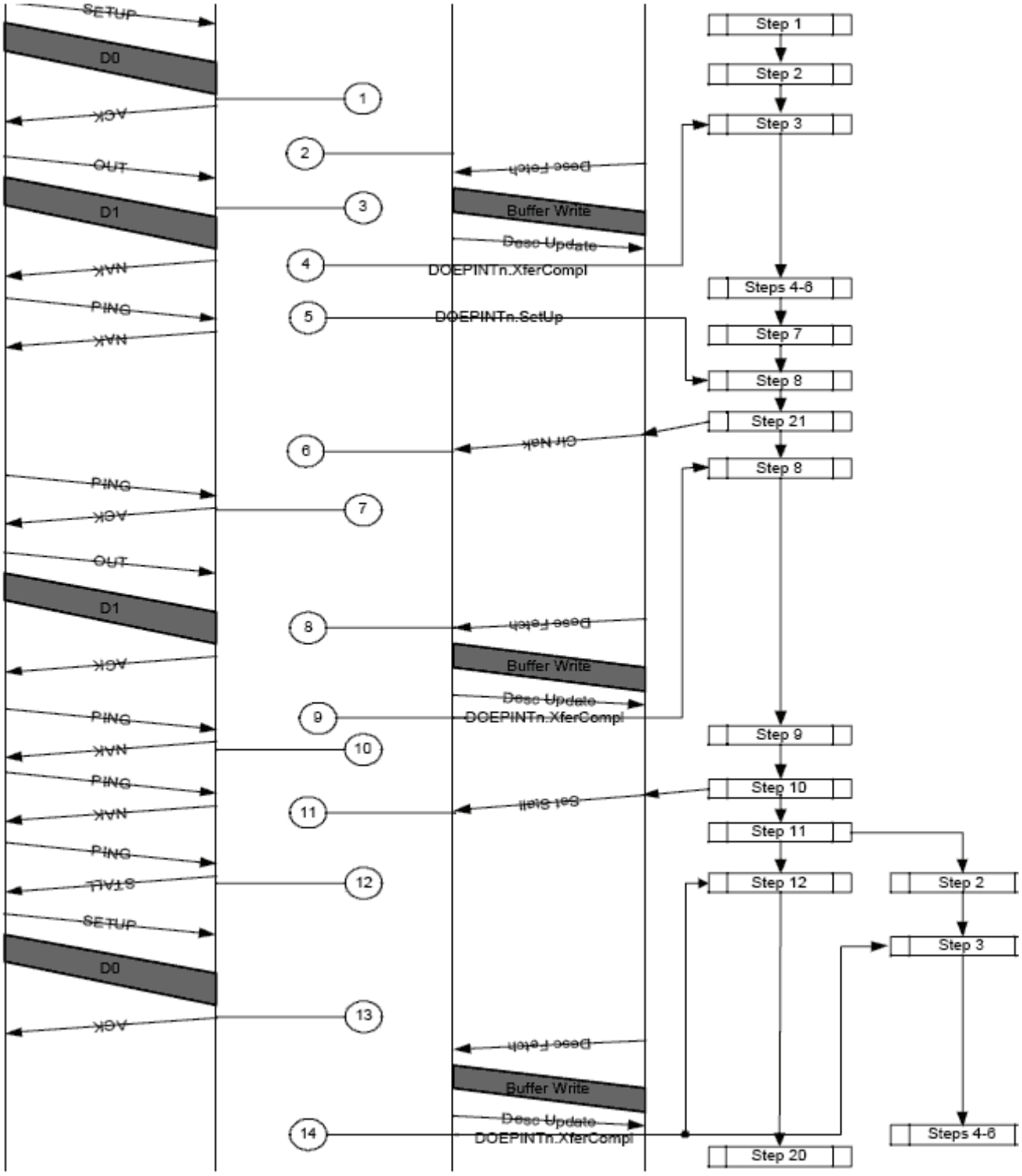


Figure 4.39 Extra Tokens During Control Write Data Phase

4.8.7.5 Extra Tokens During Control Read Data Phase

In this example, it is assumed that the data phase consists of 2 packets, and the application allocates these two packets in a single buffer. After the data phase is complete and the two packets have been transferred, the core sends an extra IN token and then the application sets Stall.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO. Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory(Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Data phase IN tokens are NAKed until this point because the NAK has not yet been cleared by the application.

7. The core starts fetching the IN data after the application enables IN DMA (In this example it is assumed that multiple packets are in the same buffer. But it could also be in different buffers). This involves the following steps

   - ✦ Fetch the descriptor pointed by DIEPDMA.
   - ✦ Fetch the data into the corresponding Tx FIFO.
   - ✦ Close the descriptor with DMA_DONE status.

8. The application clear the NAK after receiving the setup complete (DOEPINT.SetUp) interrupt. Set the Stall bit after all the Data has been pushed in the FIFO

9. After all the data has been fetched for the descriptor (Step 7), core generates DIEPINT.XferCompl interrupt.

10. The core sends data in response to the IN token for the data phase.

11. The core sends out the last packet of the IN data phase.

12. Host sends an extra token.

13. The core Stalls the IN token and also automatically Stalls the Status phase if the Host switches to the Status phase.

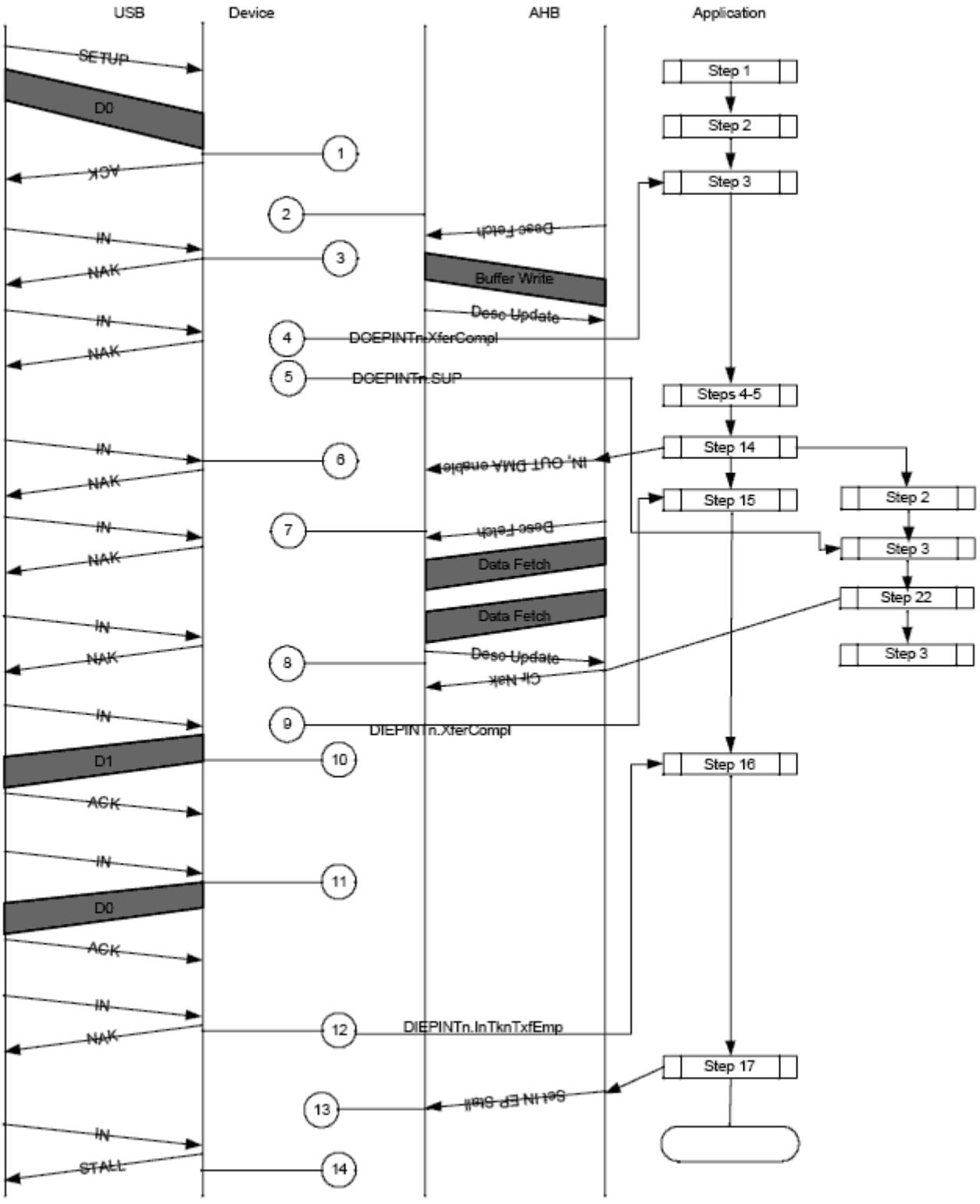


Figure 4.40 Extra IN Tokens During Control Read Data Phase

4.8.7.6 Premature SETUP During Control Write Data Phase

This example shows a Three-Stage Control Write transfer with host sending a premature Control Write SETUP packet during the data phase.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.
   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. The core receives a SETUP packet during the data phase. This is an unexpected SETUP packet.On receiving this SETUP, the SETUP data is pushed into the RxFIFO and the core again sets NAK on both IN and OUT endpoints of the control endpoint (NAK was already set because of the first SETUP packet received).

7. Application decodes the previous DOEPINT.SetUp interrupt and clears the NAK, unaware of the fact that there is another SETUP packet sitting in the RxFIFO for the same control endpoint. On seeing this condition, core does not allow clearing of the NAK bit, and masks the clearing of NAK. The core takes this decision based on the fact that a SETUP_COMPLETE status is pending in the RXFIFO.

8. The DMA detects the RxFIFO as non-empty (because of the unexpected SETUP) and does following

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.
   - ✦ The core NAKs the data phase OUT token because NAK bit clearing by the application did not take effect (as explained in Step 7).
   - ✦ The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 8).
   - ✦ The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status (for the unexpected SETUP packet received) out of the RxFIFO.

9. Application clears the NAK after decoding the latest SETUP packet. This time, the core does not mask the clearing of the NAK because there are no more SETUP_COMPLETE status sitting in the RxFIFO.

10. The core ACKs the next OUT/PING token of the data phase. 11. DMA starts transferring the OUT packet to the system memory.

    - ✦ Fetch the descriptor pointed by DOEPMA.
    - ✦ Transfer the OUT packet from the RxFIFO to the buffer pointed by the descriptor.
    - ✦ Close descriptor with DMA_DONE status.
    - ✦ The core generates DOEPINTn.XferCompl interrupt after having transferred the OUT packet to the system memory.
    - ✦ The remaining steps are similar to Steps 11-18 of “Application Programming Sequence”

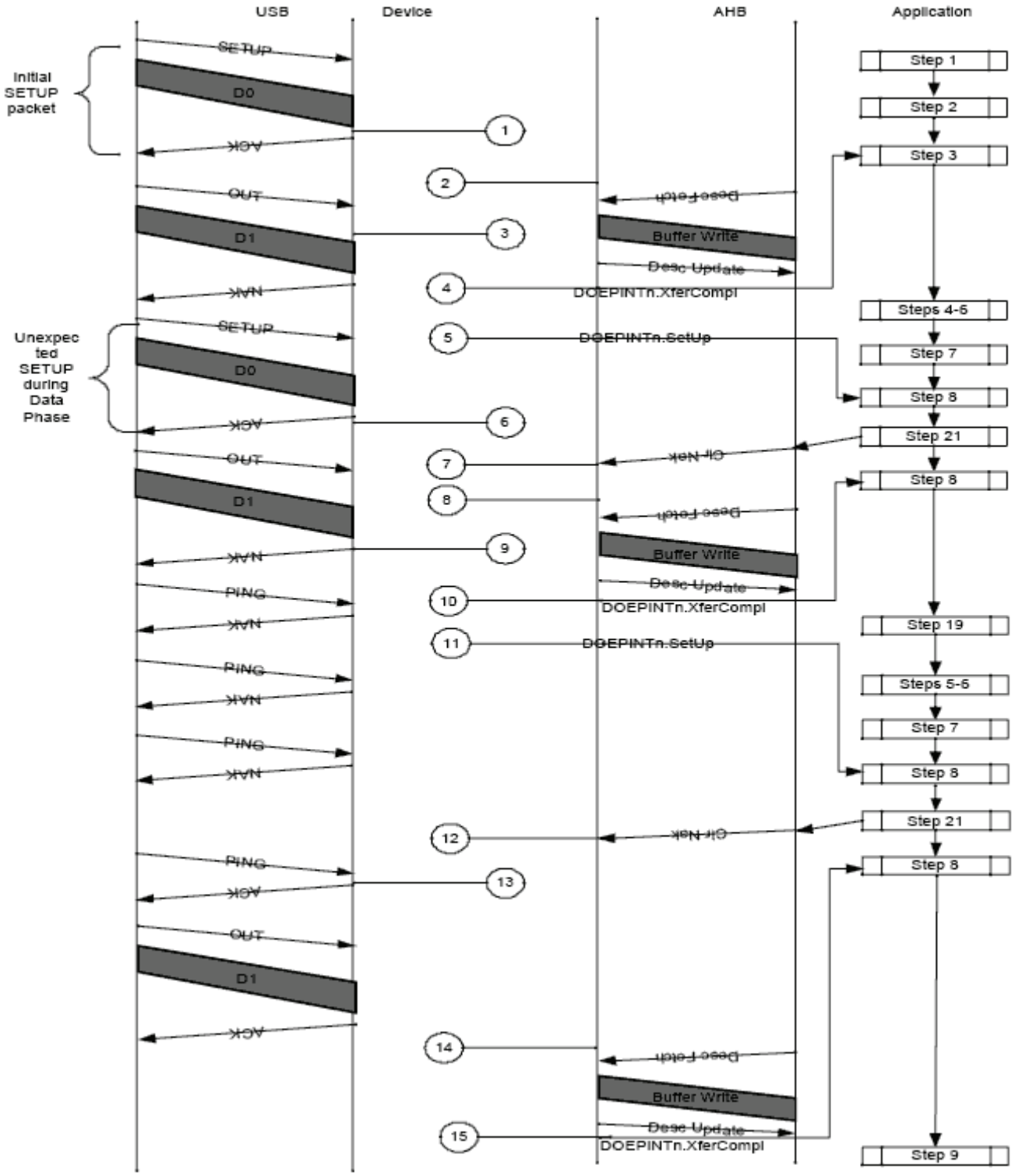


Figure 4.41 Premature SETUP During Control Write Data Phase

4.8.7.7 Premature SETUP During Control Read Data Phase

In this example, it is assumed that the data phase consists of 2 packets, and the application allocates these two packets in a single buffer. The host switches to a new control read command after having send two IN tokens during the data phase.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase IN token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase IN tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory(Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Host switches to a new Control transfer by sending a SETUP token. This is the premature SETUP packet. Core sets NAK on both IN and OUT control endpoints.

7. The core fetch the data for IN control endpoint after application enables the IN endpoint.

8. The core push SETUP_COMPLETE status into Rx FIFO on seeing the IN token for data phase.

9. Application clears NAK as a result of DOEPINT.SetUP (Setup complete) interrupt generated in Step 5. But core masks this clearing of setup_complete interrupt, because there is already one SETUP packet sitting in the Rx FIFO.

10. The core generates DIEPINTn.XFERCompl after closing the IN endpoint descriptor (for Step 7)

11. The core generates DOEPINTn.XferCompl after transferring the premature SETUP packet to system memory and closing the descriptor.

12. The core generates SETUP complete interrupt.

13. Application enables IN endpoint DMA for data phase.

14. The core fetches descriptor and data for IN endpoint.

15. Application clears IN endpoint NAK after receiving DOEPINTn.SetUP (Setup complete) interrupt. This time, the core does not mask the clearing of the Nak because NO SETUP packet is remaining in the Rx FIFO.

16. The core generates DIEPINTn.XferCompl interrupt after fetching the data and closing the descriptor. The remaining steps are same as steps 11 to 13 of “Internal Data Flow”.

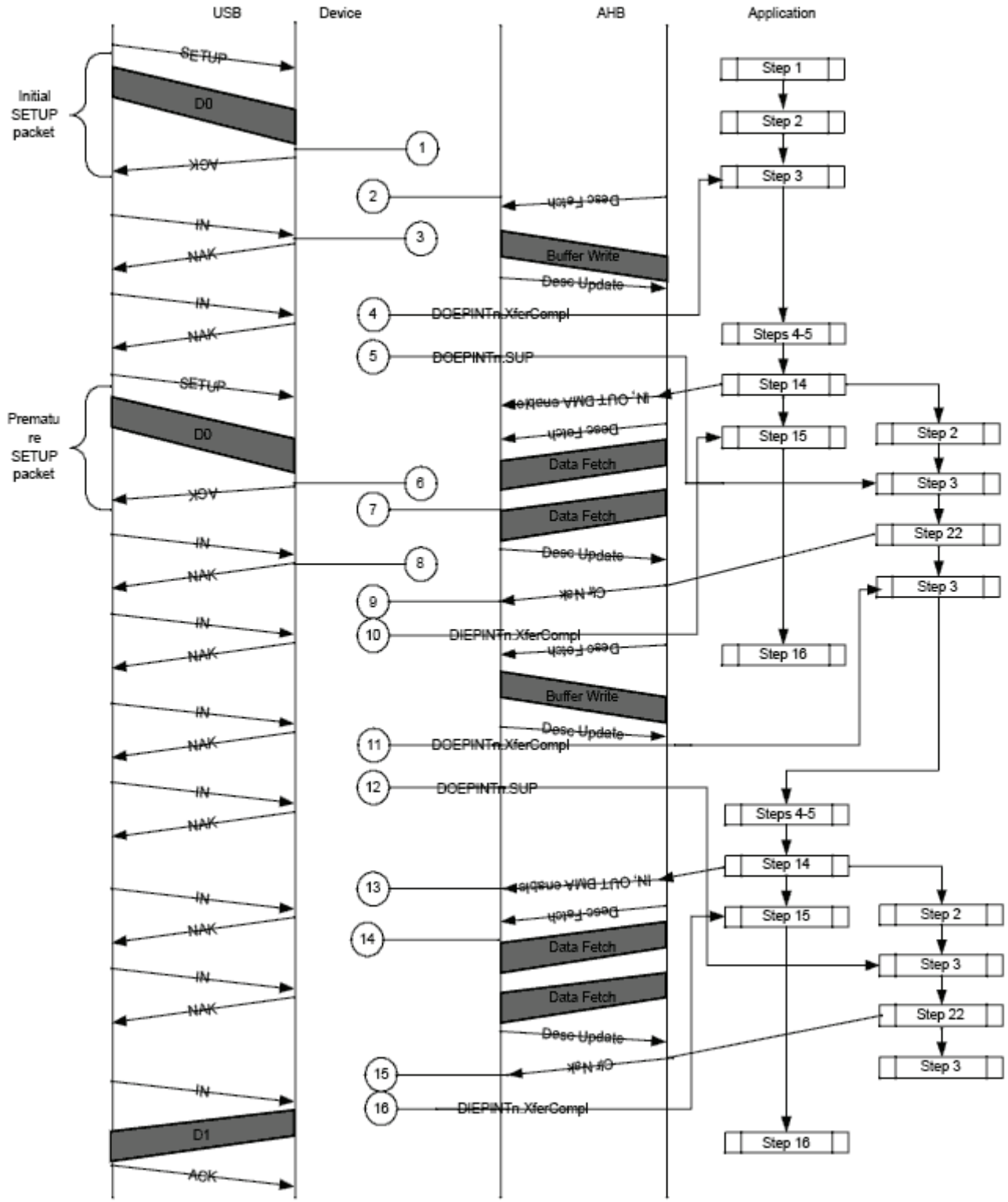


Figure 4.42 Premature SETUP During Control Read Data Phase

6.5.7.10Premature Status During Control Write

This example assumes a Three-Stage control write transfer with only Wlength field in the SETUP indicating two packets in the data phase. But the host switch to data phase after the first packet of the data phase is complete.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase OUT token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase OUT tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Application clears NAK for the data phase, after receiving DOEPINTn.SetUp interrupt (Step 5).

7. The core ACKs and the next OUT token because the NAK has been cleared (provided there is enough space in the RxFIFO.

8. DMA sees TxFIFO non empty and starts transferring the OUT packet to the system memory.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the OUT packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close descriptor with DMA_DONE status.

9. Host switch to status phase (IN token) without completing the data phase.

10. The core generates DOEPINTn.XferComp after closing the descriptor after the data fetch.

11. Application sets up descriptor, enables IN endpoint and clear NAK.

12. The core starts to fetch the descriptor and data for the status phase once application has enabled the IN endpoint.

13. The core generates DIEPINTn.XferCompl after doing the data fetch and the descriptor update (step 12)

14. The core sends data out in response to status phase IN token.

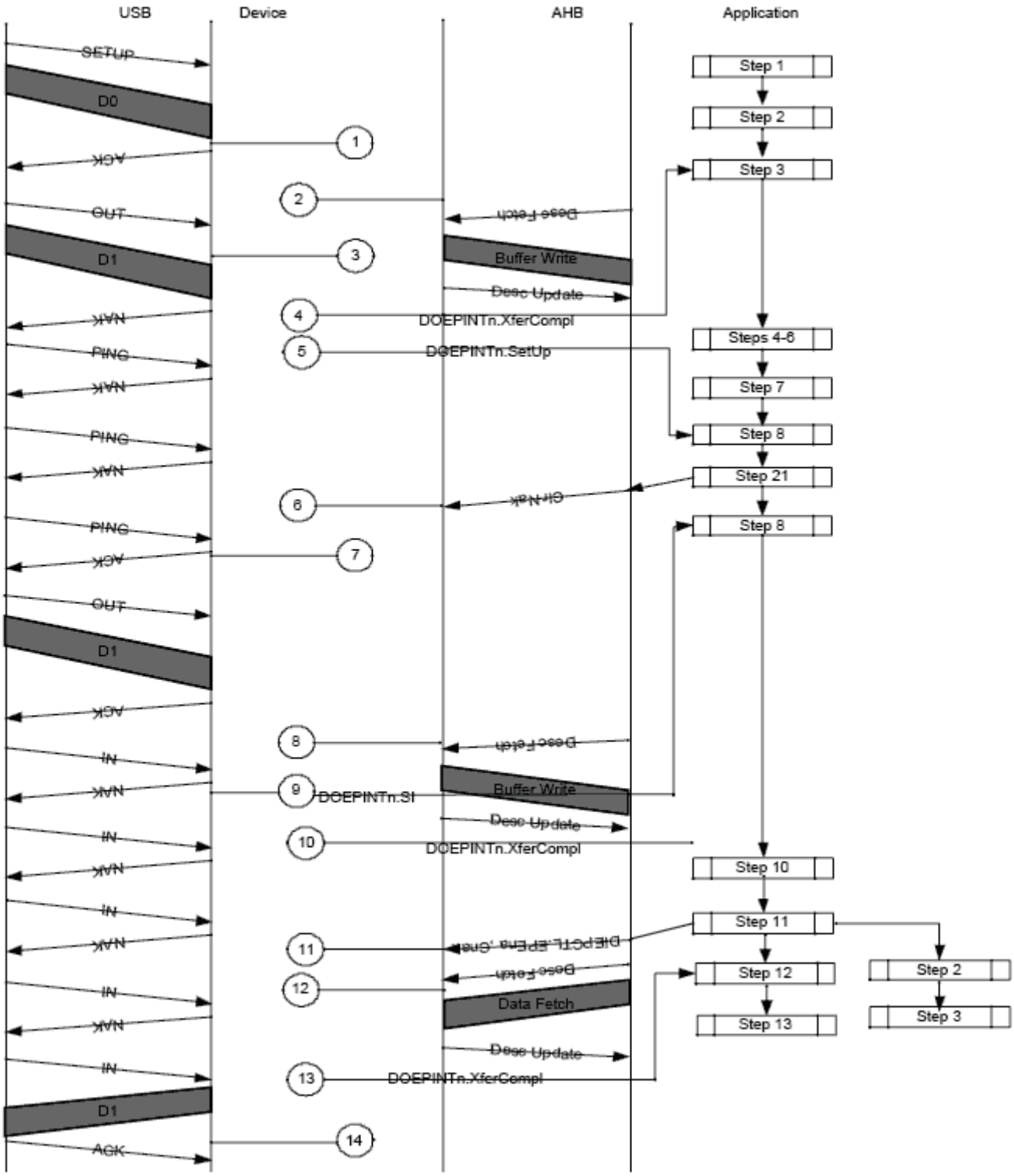


Figure 4.43 Premature Status Phase During Control Write

4.8.7.8 Premature Status During Control Read

In this example, it is assumed that the data phase consists of two packets, and the application allocates these two packets in a single buffer. After one packet in the data phase, host switches to status phase.

1. On receiving SETUP, the data is pushed into the Rx FIFO and the core sets NAK on both IN and OUT endpoint of that control endpoint.

2. The DMA detects the RxFIFO as non-empty and does the following.
   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.

3. On receiving the first data phase IN token after the SETUP, the core push a SETUP_COMPLETE status into the RxFIFO.Core NAKs the data phase IN tokens because of the NAK set on receiving the SETUP packet.

4. The core generates DOEPINT.XferCompl interrupt after having transferred the SETUP packet into memory (Step 2).

5. The core generates DOEPINT.SetUp interrupt after the DMA has popped the SETUP_COMPLETE status out of the RxFIFO.

6. Data phase IN tokens are NAKed until this point because the NAK has not yet been cleared by the application.

7. The core starts fetching the IN data after the application enables IN DMA (In this example it is assumed that multiple packets are in the same buffer. But it could also be in different buffers). This involves the following steps

   - ✦ Fetch the descriptor pointed by DIEPDMA.
   - ✦ Fetch the data into the corresponding Tx FIFO.
   - ✦ Close the descriptor with DMA_DONE status.

8. Application clear the NAK after receiving the setup complete (DOEPINT.SetUp) interrupt.

9. After all the data has been fetched for the descriptor (Step 7), core generates DIEPINT.XferCompl interrupt.

10. The core sends data in response to the IN token for the data phase.

11. Host switches to status phase and sends the status phase OUT token. Core ACks the OUT packet because the NAK has already been cleared.

12. The DMA detects the RxFIFO as non-empty (because of the status phase data) and does following.

   - ✦ Fetch the descriptor pointed by DOEPMA.
   - ✦ Transfer the SETUP packet from the RxFIFO to the buffer pointed by the descriptor.
   - ✦ Close the descriptor with DMA_DONE status.
   - ✦ The core generates DOEPINTn.XferCompl after transferring the status phase data to system memory and closing The descriptor.

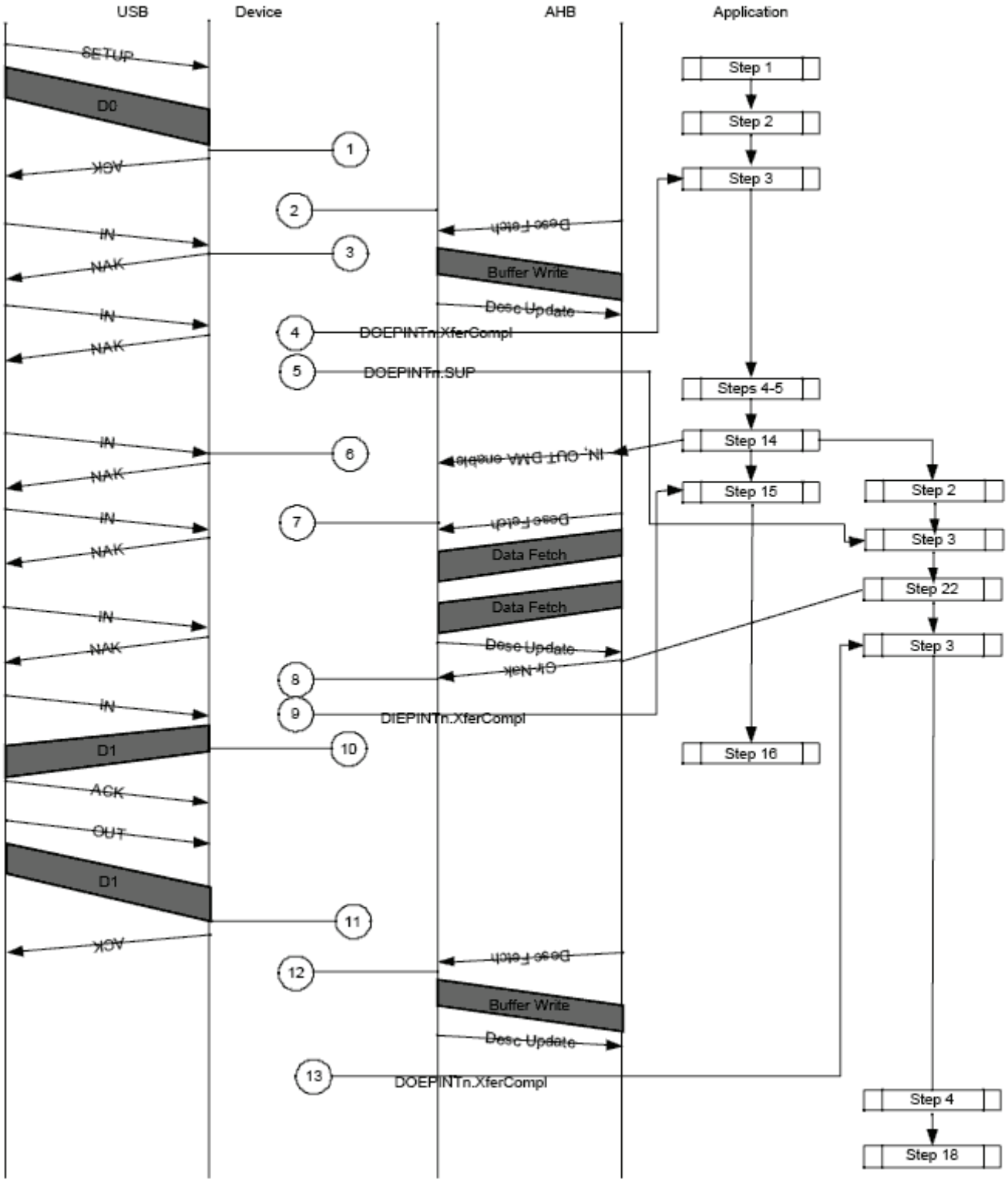


Figure 4.44 Premature Status Phase During Control Read

4.8.7.9 Lost ACK During Last Packet of Control Read

This is similar to the previous section. Figure 6-51 shows this.

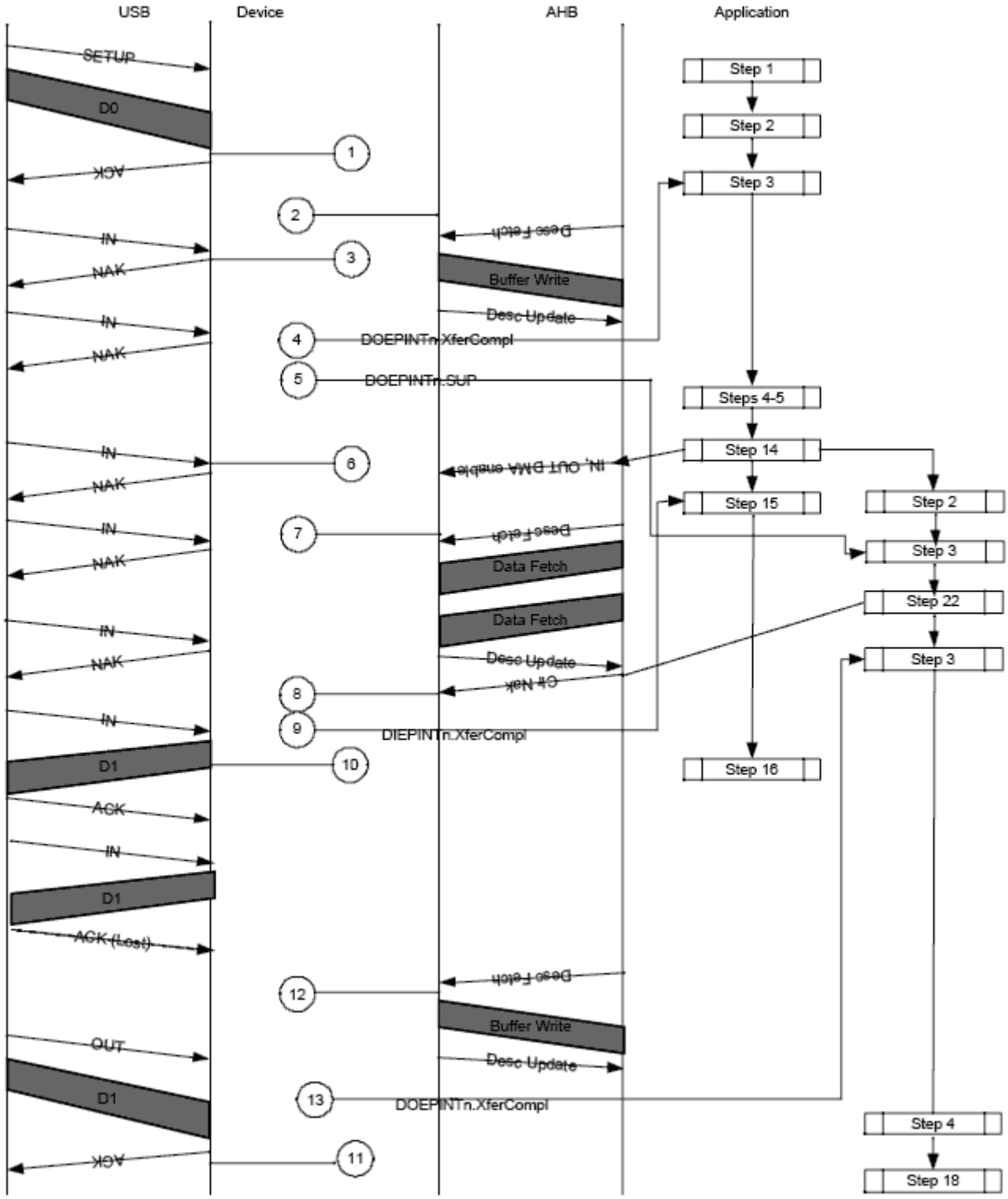


Figure 4.45 Lost ACK During Last Packet of Control Read

4.9 Bulk Transfer Handling in Scatter/Gather DMA Mode

The following section describes the Bulk transfer handling in Scatter/Gather DMA mode.

4.9.1 Bulk IN Transfer Data Transaction in Scatter-Gather DMA Mode

4.9.1.1 Interrupt usage

The following interrupts are of relevance.

1. DIEPINTn.XferCompl (Transfer complete, based on IOC bit in the descriptor)
2. DIEPINTn.BNA (Buffer Not Available)

4.9.1.2 Application Programming Sequence

This section describes the application programming sequence for Bulk IN transfer scenarios.

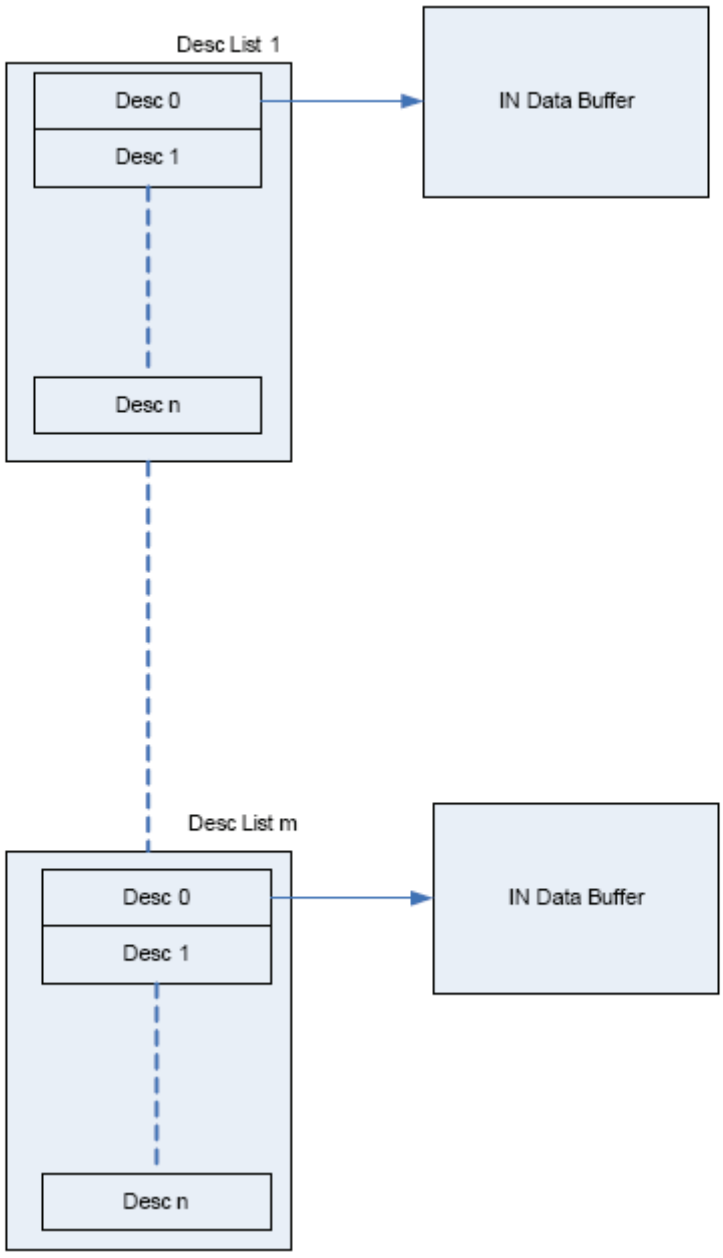


Figure 4.46 IN Descriptor List

1. Prepare Descriptor(s):

2. The application creates descriptor list(s) in the system memory pertaining to an Endpoint.

3. Each descriptor list may have up to *n* descriptors and there may be up to *m* descriptor lists.

4. Application may choose to set the IOC bit of the corresponding descriptor. If the IOC is set for the last descriptor of the list, the core generates DIEPINTn.XferCompl interrupt after the entire list is processed.

5. Program DIEPDMAn:

   a. Application programs the base address of the descriptor in the corresponding IN Endpoint DIEPDMAn register.

6. Enable DMA:

   a. Application programs the corresponding endpoint DIEPCTLn register with the following

      i. DIEPCTLn.MPS—Max Packet size of the endpoint
      ii. DIEPCTLn.CNAK—Set to 1 to clear the NAK
      iii. DIEPCTLn.EPEna—Set to 1 to enable the DMA for the endpoint.

7. Wait for Interrupt:
   a. On reception of DIEPINTn.XferCompl, application must check the Buffer status and Tx Status field of the descriptor to ascertain that the descriptor closed normally.

DIEPINTn.BNA interrupt gets generated by the core when it encounters a descriptor in the list whose Buffer Status field is not Host Ready. In this case, the application is suppose to read the DIEPDMAn register to ascertain the address for which the BNA interrupt is asserted to take corrective action.

4.9.1.3 Internal Flow

Bulk IN Transfers

The core handles Bulk IN transfers internally as functionally depicted in Figure 6-53 on page 475 (Non ISO IN Descriptor/Data Processing). Figure 6-54 depicts this flow.

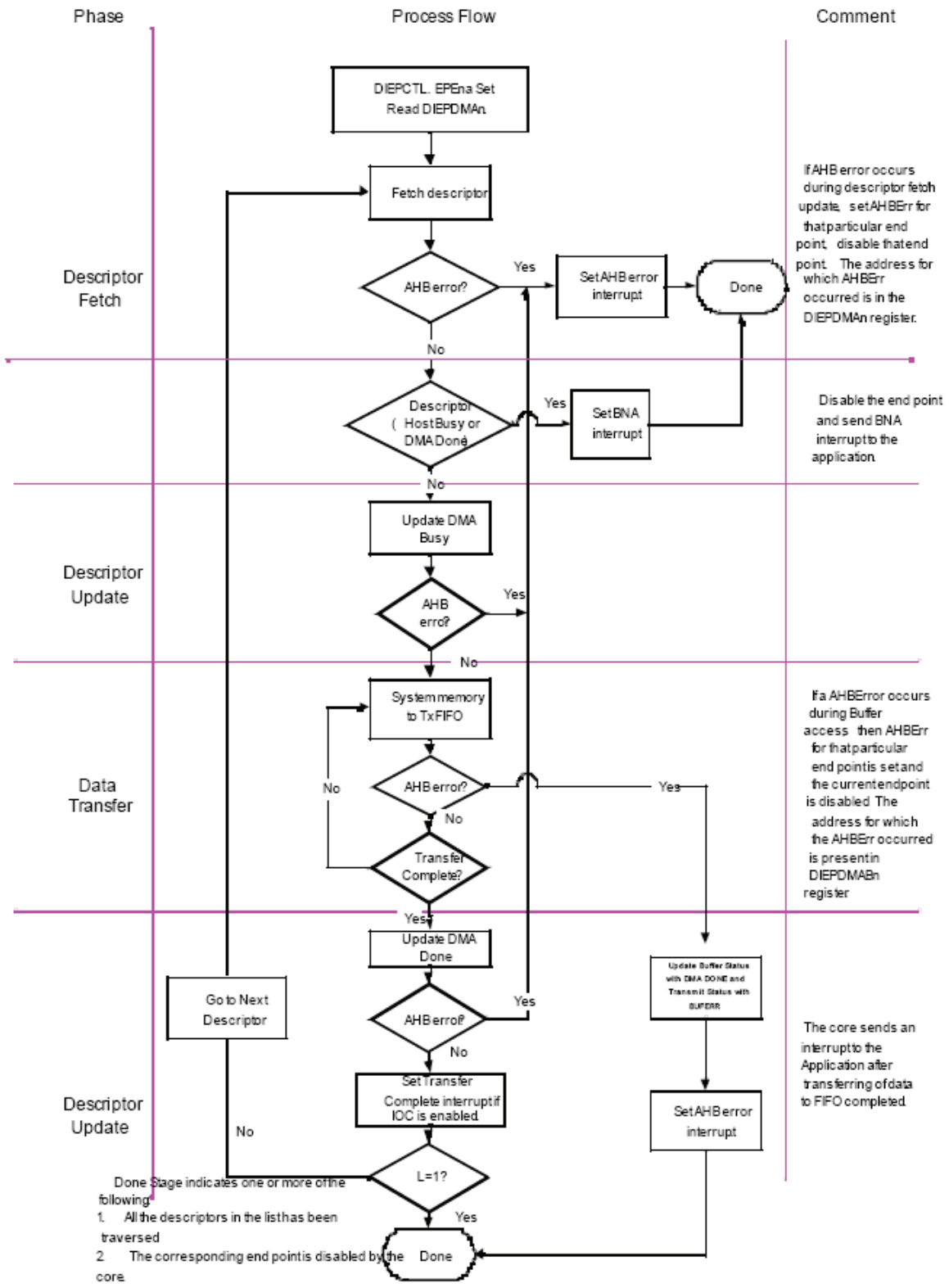
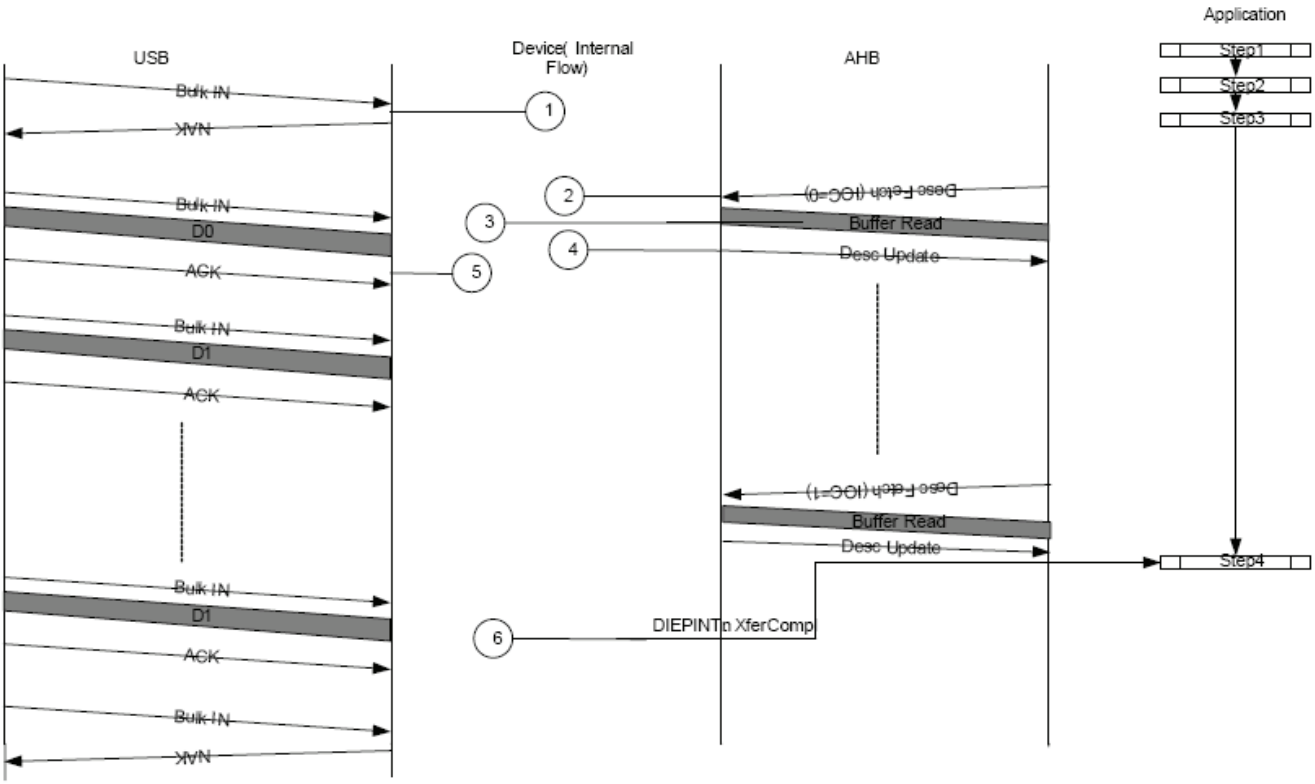


Figure 4.47 Non ISO IN Descriptor/Data Processing



1. When a BULK IN token is received on an end point before the corresponding DMA is enabled, (DIEPCTLn.EPEna = 1'b0), it is NAKed on USB.

2. As a result of application enabling the DMA for the corresponding end point (DIEPCTLn.EPEna=1), the core fetches the descriptor and processes it.

3. The DMA fetches the data from the system memory and populates its internal FIFO with this data.

4. After fetching all the data from a descriptor, the core closes the descriptor with a DMA_DONE status.

5. On reception of BULK IN tokens on USB, data is sent to the USB Host.

6. After the last descriptor in the chain is processed, the core generates DIEPINTn.XferCompl interrupt provided the IOC bit for the last descriptor is set.

4.9.2 Bulk OUT Data Transaction in Scatter-Gather Mode

4.9.2.1 Interrupt Usage

The following interrupts are of relevance.

1. DOEPINTn.XferCompl (Transfer complete, based on IOC bit in the descriptor)

2. DOEPINTn.BNA (Buffer Not Available)

4.9.2.2 Application Programming Sequence

This section describes the application programming sequence to take care of Bulk OUT transfer scenarios.

4.9.2.3 Application Programming Sequence

This section describes the application programming sequence to take care of Bulk OUT transfer scenarios.

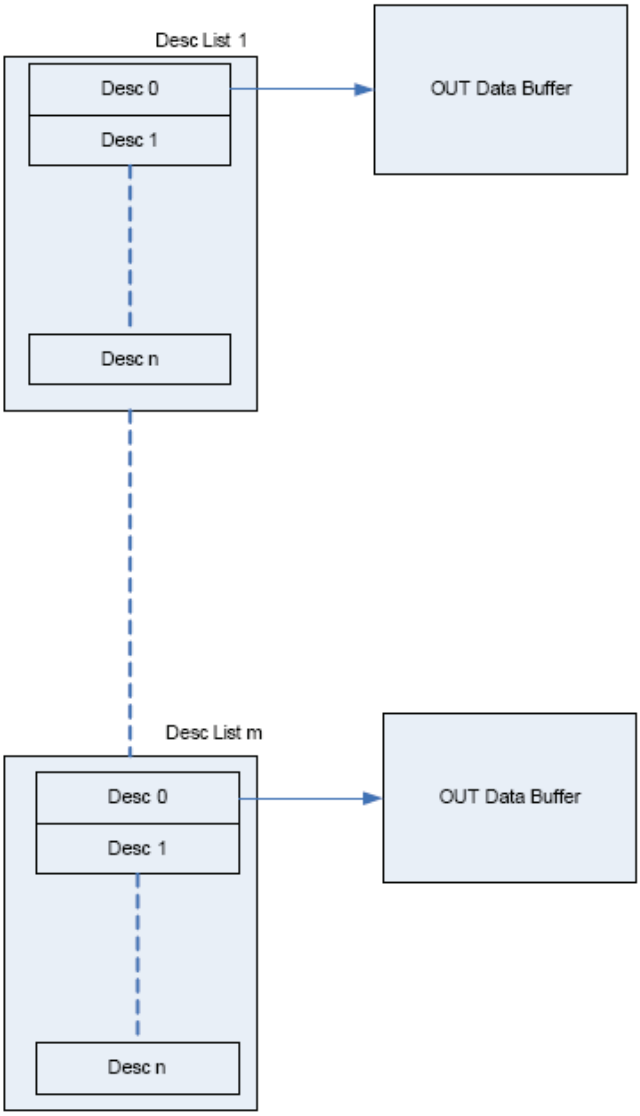


Figure 4.48 OUT Descriptor List

1. Prepare Descriptor(s):

2. The application creates descriptor list(s) in the system memory pertaining to an Endpoint.

3. Each descriptor list may have up to *n* descriptors and there may be up to *m* descriptor lists.

4. Application may choose to set the IOC bit of the corresponding descriptor. If the IOC is set for the last descriptor of the list, the core generates DOEPINTn.XferCompl interrupt after the entire list is processed.

    a. Based on L bit and MTRF bit combinations, the core may disable the end point. Refer to Table 6-3 on page 433 for bit field descriptions.

5. Program DOEPDMAn:

    a. Application programs the base address of the descriptor in the corresponding OUT Endpoint DOEPDMAn register.

6. Enable DMA:

    a. Application programs the corresponding endpoint DOEPCTLn register with the following

        i. DOEPCTL.MPS—Max Packet size of the endpoint
        ii. DOEPCTL.CNAK—Set to 1 to clear the NAK

iii. DOEPCTL.EPEna—Set to 1 to enable the DMA for the endpoint.

7. Wait for Interrupt:

a. On reception of DOEPINTn.XferCompl, application must check the Buffer status and Rx Status field of the descriptor to ascertain that the descriptor closed normally.

DOEPINTn.BNA interrupt gets generated by the core when it encounters a descriptor in the list whose Buffer Status field is not Host Ready. In this case, the application is suppose to read the DOEPDMAn register to ascertain the address for which the BNA interrupt is asserted to take corrective action.

4.9.2.4 Internal Flow

The core handles Bulk OUT transfers internally as depicted in Figure 6-56. Figure 6-57 also diagrams this flow.

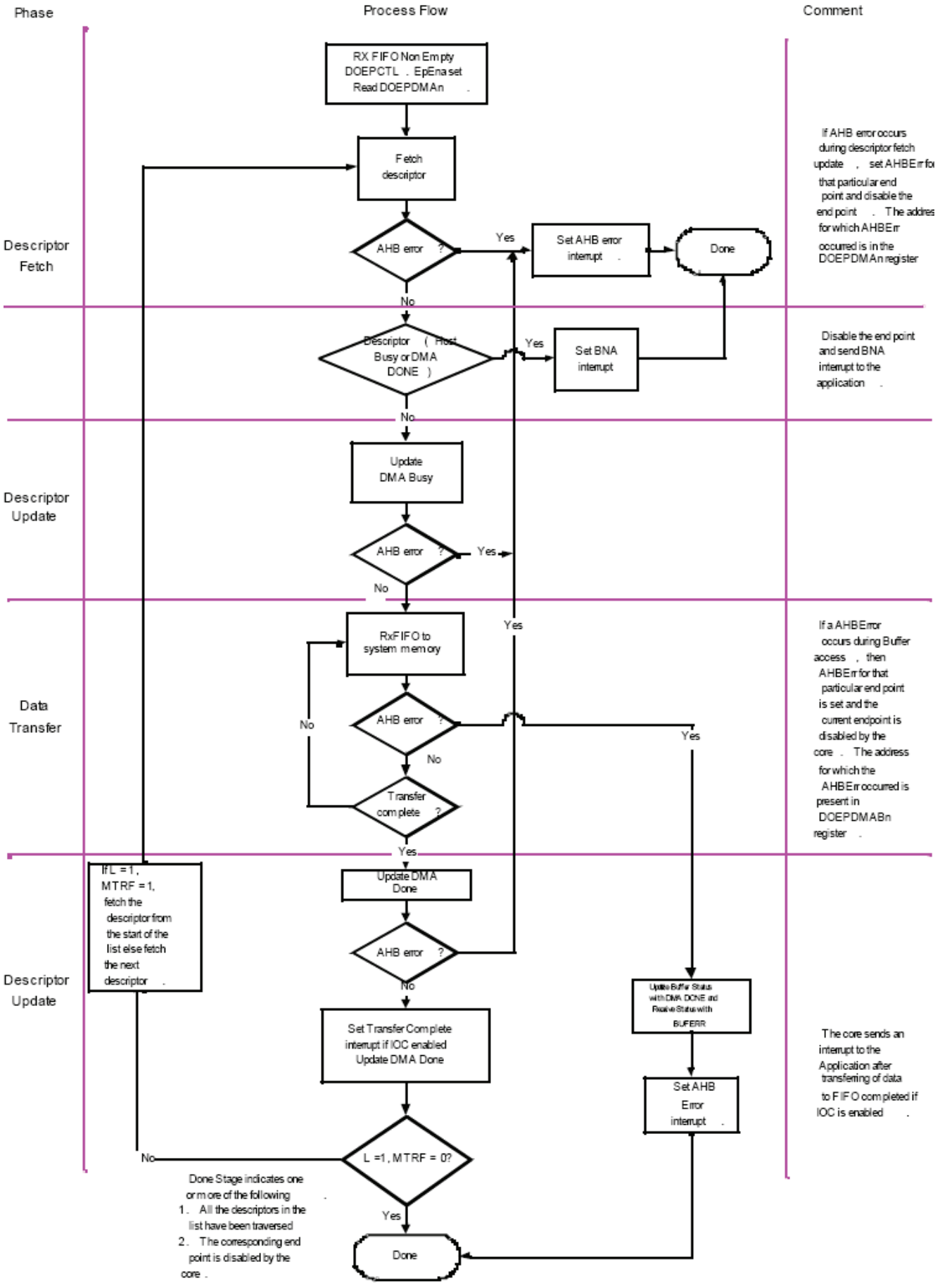

Phase
Process Flow
Comment
Descriptor Fetch
Descriptor Update
Data Transfer
Descriptor Update
RX FIFO Non Empty DOEPCTL . EpEna set Read DOEPDMAn .
Fetch descriptor
AHB error ?
Yes
Set AHB error interrupt .
Done
No
Descriptor (Host Busy or DMA DONE)
Yes
Set BNA interrupt
No
Update DMA Busy
AHB error ?
Yes
No
RxFIFO to system memory
AHB error ?
Yes
No
Transfer complete ?
Yes
Update DMA Done
AHB error ?
No
Set Transfer Complete interrupt if IOC enabled Update DMA Done
L =1 , MTRF = 0?
No
Yes
Done
If L = 1 , MTRF = 1, fetch the descriptor from the start of the list else fetch the next descriptor .
Update Buffer Status with DMA DONE and Receive Status with BUFERR
Set AHB Error interrupt .
Done Stage indicates one or more of the following .
1 . All the descriptors in the list have been traversed
2 . The corresponding end point is disabled by the core .
If AHB error occurs during descriptor fetch update , set AHBErr for that particular end point and disable the end point . The address for which AHBErr occurred is in the DOEPDMAn register
Disable the end point and send BNA interrupt to the application .
If a AHBError occurs during Buffer access , then AHBErr for that particular end point is set and the current endpoint is disabled by the core . The address for which the AHBErr occurred is present in DOEPDMABn register .
The core sends an interrupt to the Application after transferring of data to FIFO completed if IOC is enabled .

Figure 4.49 Non ISO OUT Descriptor/Data Buffer Processing

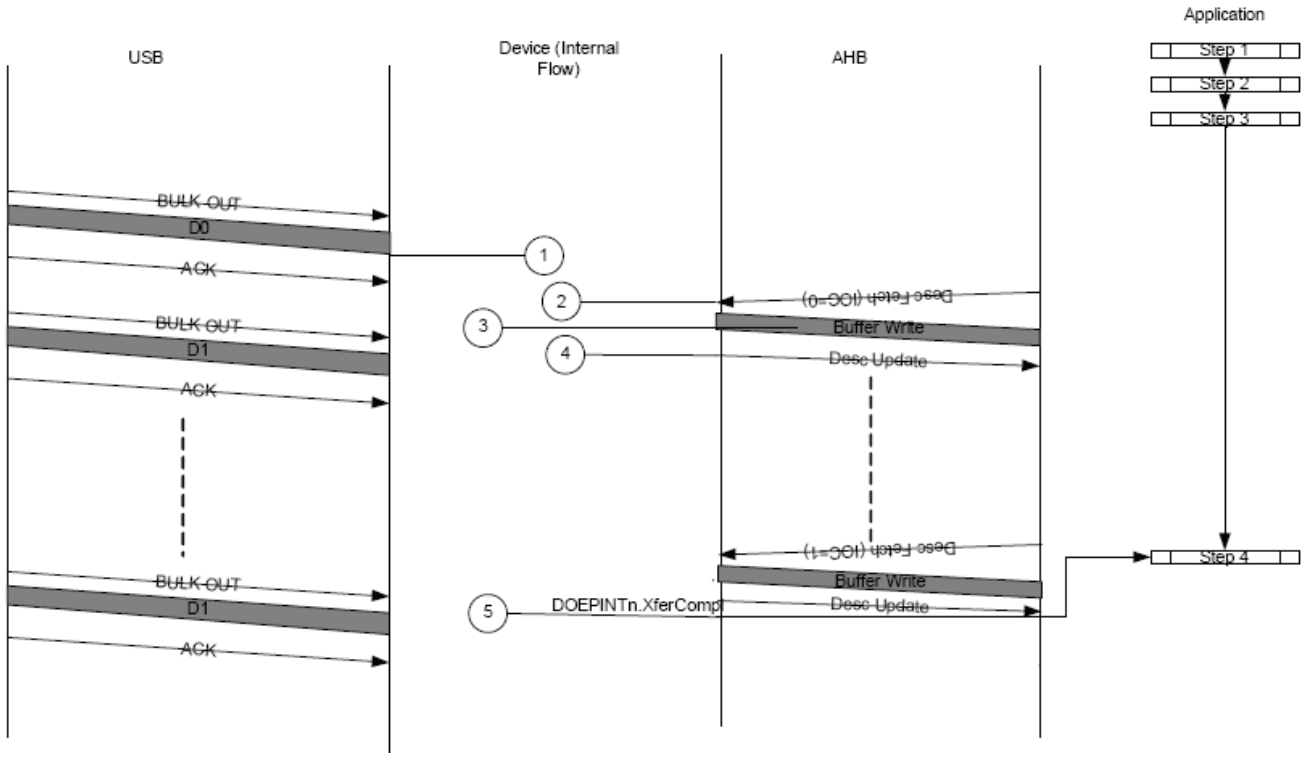


Figure 4.50 Bulk OUT Transfers

1. When a BULK OUT token is received on an end point, the core stores the received data internally in a FIFO.

2. As a result of application enabling the DMA for the corresponding end point (DOEPCTLn.EPEna=1), the core fetches the descriptor and processes it.

3. The DMA transfers the data from the internal FIFO to system memory.

4. After transferring all the data from the FIFO, the core closes the descriptor with a DMA_DONE status.

5. After the last descriptor in the chain is processed, the core generates DOEPINTn.XferCompl interrupt provided the IOC bit for the last descriptor is set.

4.10 Interrupt Transfer Handling in Descriptor-Based Scatter/Gather DMA Mode

4.10.1 Interrupt IN Data Transaction in Scatter/Gather DMA Mode

Application programming for Interrupt IN transfers is as with the Bulk IN transfer sequence.
The core handles Interrupt IN transfers internally in the same way it handles Bulk IN transfers

4.10.2 Interrupt OUT Transfer

Application programming for Interrupt OUT transfers is as with the Bulk OUT transfer sequence.
The core handles Interrupt OUT transfers internally in the same way it handles Bulk OUT Transfers

4.11 Isochronous Transfer Handling in Descriptor-Based Scatter/Gather DMA Mode

4.11.1 Isochronous IN Transfer

The application programming for Isochronous IN transfers is in the same manner as Bulk IN transfer sequence.
The following behavior is of importance while working with Isochronous IN end points

DCTL.IgnrFrmNum = 1'b1

The way the core handles Isochronous IN transfers internally in the same way as it handles Bulk IN Transfers.

DCTL.IgnrFrmNum = 1'b0

The core closes the descriptor and clears the corresponding fetched data in the FIFO if the USB (micro)frame number to which the descriptor belongs is elapsed.

4.11.1.1 Isochronous Transfers in Scatter/Gather (Descriptor DMA) Mode

This topic includes descriptions of both isochronous IN and OUT transfers

Isochronous IN

In the case of ISO IN After descriptor is fetched, the frame number field M is compared with current USB frame number N.

If the frame number in the fetched descriptor is already elapsed (M<N) then the descriptor is closed with status changed to DMA Done.

If the frame number in the fetched descriptor is for future (N>M+1) then the descriptor is left untouched. The Core suspends and re-look at this descriptor contents in the next frame/microframe.

- ❖ If the frame number in the fetched descriptor is for current or next frame (N=M or M+1) then the descriptor is further processed as per the flow chart. At the end of data transfer from memory to TxFIFO the above check must be performed. And if the data fetch finished in the subsequent frame, data must be flushed and descriptor must be closed (DMA Done) with BUFFLUSH status.

- ❖ For ISO IN, the application creates a series of descriptors (D,D+1,D+2.....) for a given periodic end point corresponding to successive frames (N,N+1,N+2.....).

Note The series of descriptors does not correspond to the series of frames in the same order.

For example, D and D + 1 may correspond to N, D + 2 may correspond to N + 1 and so on except in the case where the application can create more than one descriptor for the same microframe. The core fetches the descriptor and compares the frame/μframe number field with the current frame/μframe number.
If the fetched descriptor corresponds to a frame which has already elapsed, the core updates the descriptor with DMA Done Buffer status and proceeds to the next descriptor.
If the next descriptor fetched indicates that it corresponds to frame number N or N + 1, it services it. In the process of fetching the descriptors, if the core determines that the descriptor corresponds to a future frame/μframe (> N + 1), it does not service the descriptor in that frame/μframe. Instead, it moves on to the next periodic endpoint or non-periodic endpoint without disabling the current periodic endpoint. It revisits this endpoint in the next frame/μframe and repeats the process.

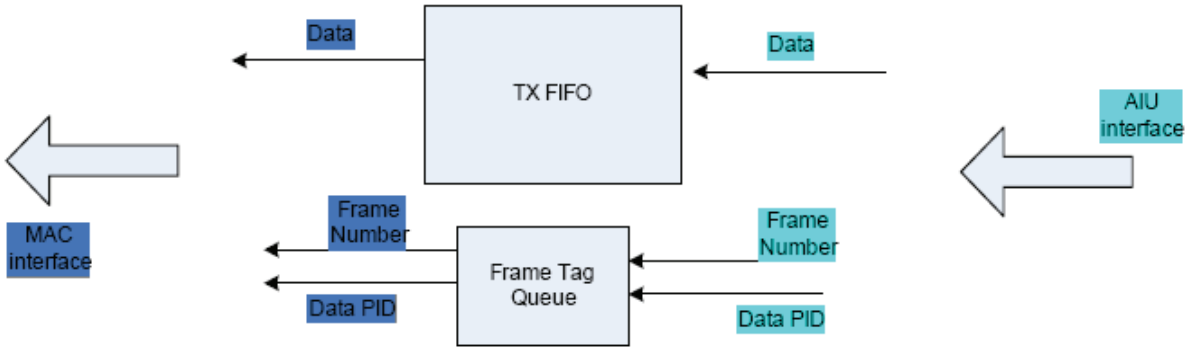


Figure 4.51 ISO IN Data Flow

Application Programming Sequence

This section describes the application programming sequence for Isochronous IN transfer scenarios.

Prepare Descriptor(s):

The application creates descriptor list(s) in the system memory pertaining to an Endpoint. Each descriptor list may have up to *n* descriptors and there may be up to *m* descriptor lists. Application may choose to set the IOC bit of the corresponding descriptor. If the IOC is set for the last descriptor of the list, the core generates DIEPINTn.XferCompl interrupt after the entire list is processed.

1. Program DIEPDMAn: a. Application programs the base address of the descriptor in the corresponding IN Endpoint DIEPDMAn register.

2. Enable DMA:

a. Application programs the corresponding endpoint DIEPCTLn register with the following
i. DIEPCTLn.MPS—Max Packet size of the endpoint
ii. DIEPCTLn.CNAK—Set to 1 to clear the NAK
iii. DIEPCTLn.EPEna—Set to 1 to enable the DMA for the endpoint.

3. Wait for Interrupt:
a. On reception of DIEPINTn.XferCompl, application must check the Buffer status and Tx Status field of the d escriptor to ascertain that the descriptor closed normally.

DIEPINTn.BNA interrupt gets generated by the core when it encounters a descriptor in the list whose Buffer Status field is not Host Ready. In this case, the application is suppose to read the DIEPDMAn register to ascertain the address for which the BNA interrupt is asserted to take corrective action.

4.11.1.2 Internal Flow

The core handles isochronous IN transfers internally as functionally depicted in Figure 6-59. Figure 6-60 on page 485 also diagrams this flow.

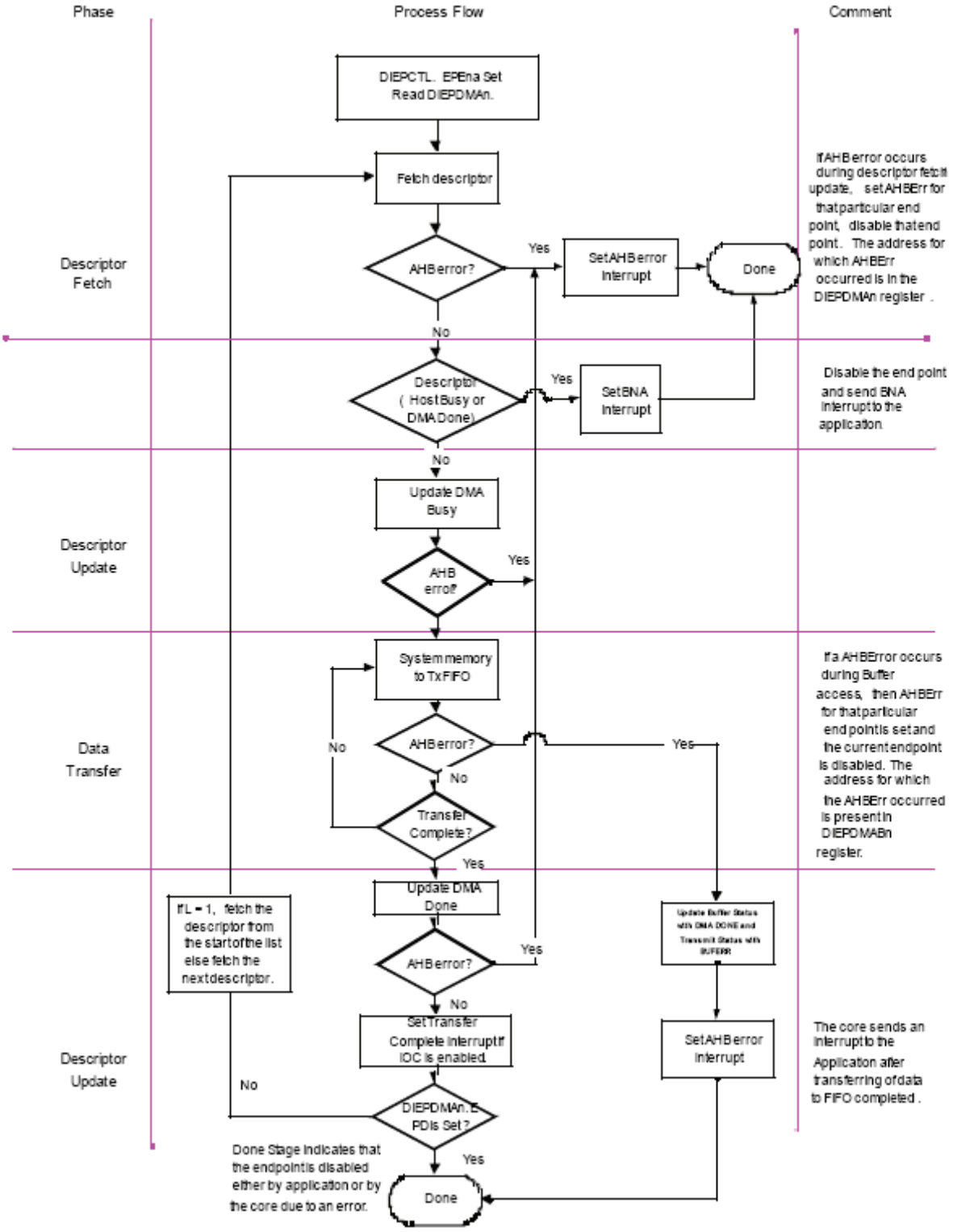


Figure 4.52 ISO IN Descriptor/Data Processing

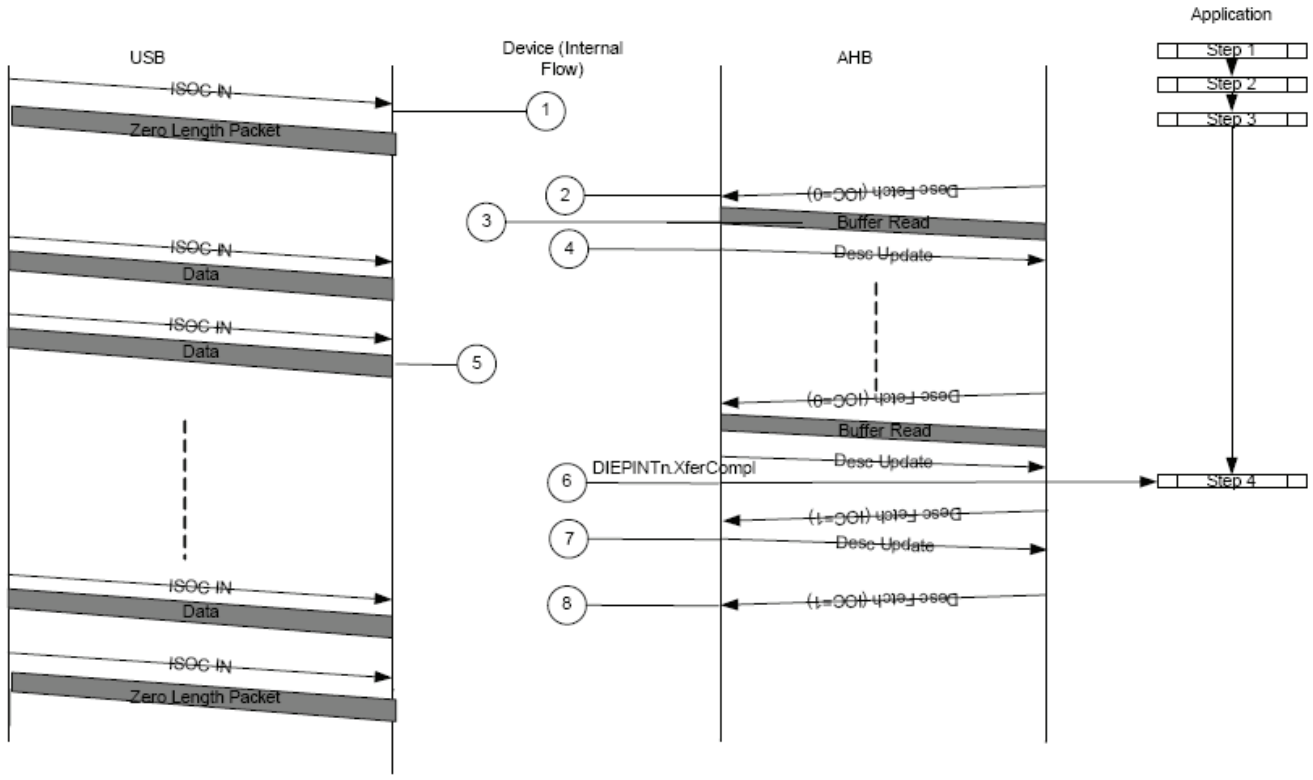


Figure 4.53 Isochronous IN Transfers

1. When an Isochronous IN token is received on an end point before the corresponding DMA is enabled, (DIEPCTLn.EPEna = 1'b0), zero length packet is sent on USB.

2. As a result of application enabling the DMA for the corresponding end point (DIEPCTLn.EPEna=1), the core fetches the descriptor. If the descriptor belongs to the current or the next USB frame number, the core processes it.

3. The DMA fetches the data pointed by the above descriptor from the system memory and populates its internal FIFO with this data.

4. After fetching all the data, the core closes the descriptor with a DMA_DONE status.

5. On reception of Isochronous IN tokens on USB, data is sent to the USB Host.

6. After the last descriptor in the chain is processed, the core generates DIEPINTn.XferCompl interrupt provided the IOC bit for the last descriptor is set.

7. When the DMA fetches a descriptor whose USB frame number has been already elapsed, it closes that descriptor with a DMA_DONE status without fetching the data for that descriptor.

8. When the DMA fetches a descriptor which has a future USB frame number, it does not service it in the current context. It services it in the future.

4.11.2 Isochronous OUT Transfer

The application programming for isochronous out transfers is in the same manner as Bulk OUT transfer sequence, except that the application creates only 1 packet per descriptor for an isochronous OUT endpoint. The core handles isochronous OUT transfers internally in the same way it handles Bulk OUT transfers, and as depicted in Figure 6-61.

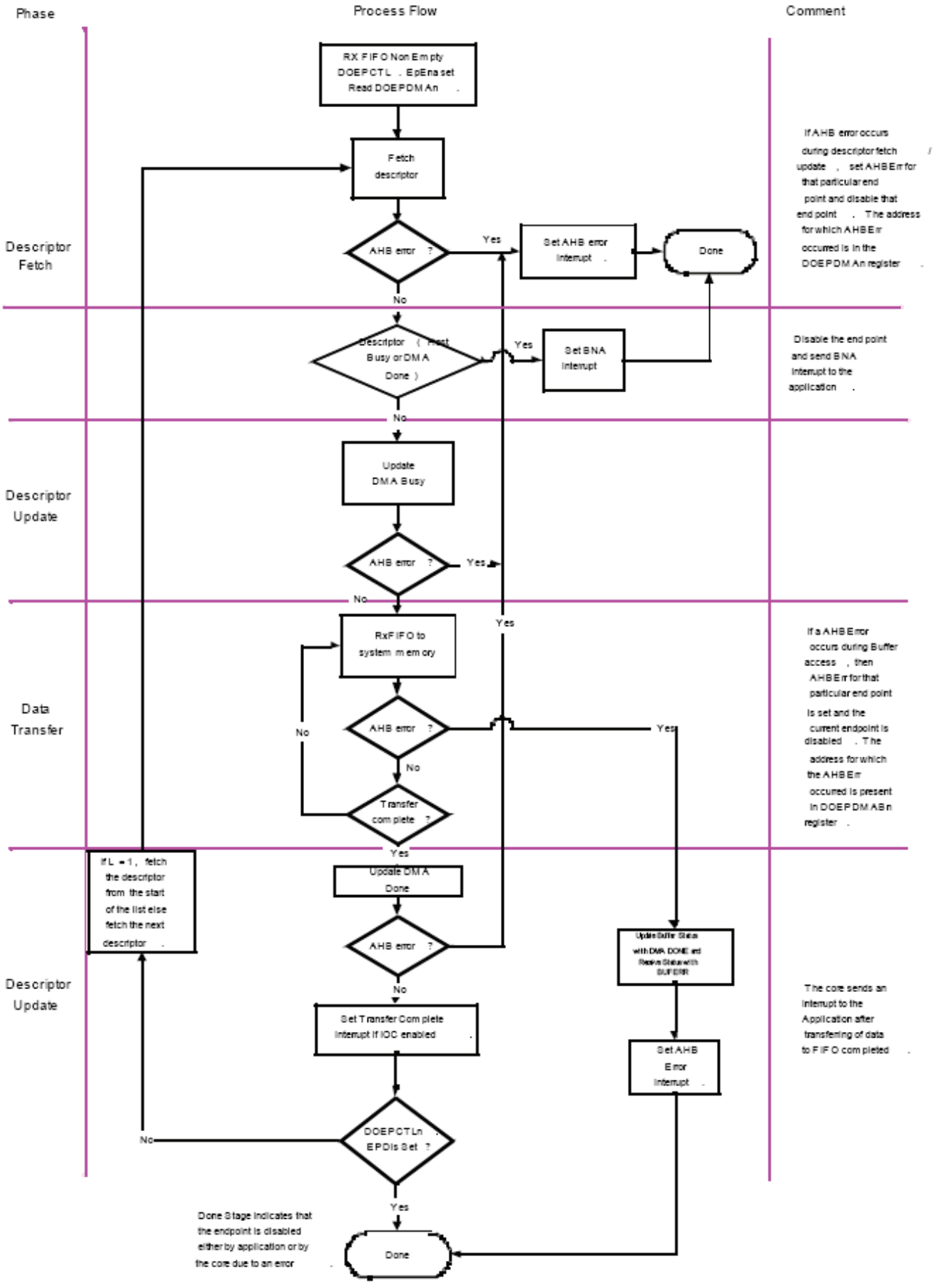


Figure 4.54 Isochronous OUT Descriptor/Data Buffer Processing

Isochronous OUT

For ISO OUT transactions, the core transfers the packets from the Rx FIFO to the system memory and updates the frame number field of the descriptor with the frame number in which the packet was received.The frame number for which data is received is extracted from the Receive Status queue and written back to the descriptor.

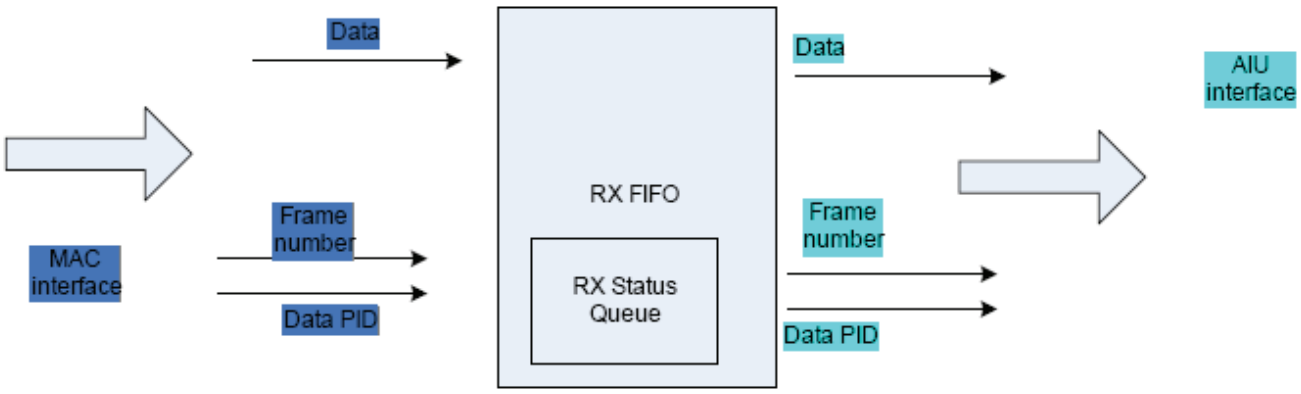


Figure 4.55 ISO Out Data Flow

Note Incomplete Isochronous Interrupt (GINTSTS.incomplete) is not generated in Scatter/Gather DMA mode. Received isochronous packets are sent unmodified to the application memory, with the corresponding frame number updated in the descriptor status.

4.12 OTG Programming Model

The otg core is an OTG device supporting HNP and SRP. When the core is connected to an “A” plug, it is referred to as an A-device. When the core is connected to a “B” plug it is referred to as a B-device. In Host mode, the otg core turns off VBUS to conserve power. SRP is a method by which the B-device signals the A-device to turn on VBUS power. A device must perform both data-line pulsing and VBUS pulsing, but a host can detect either data-line pulsing or VBUS pulsing for SRP. HNP is a method by which the B-device negotiates and switches to host role. In Negotiated mode after HNP, the B-device suspends the bus and reverts to the device role.

4.12.1 A-Device Session Request Protocol

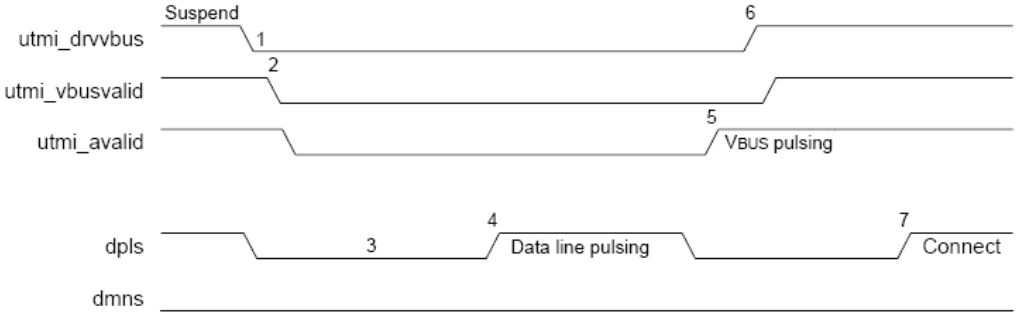


Figure 4.56 A-Device SRP

The application must set the SRP-Capable bit in the Core USB Configuration register. This enables the otg core to detect SRP as an A-device.

1. To save power, the application suspends and turns off port power when the bus is idle by writing the Port Suspend and Port Power bits in the Host Port Control and Status register.

2. PHY indicates port power off by deasserting the utmi_vbusvalid signal.

3. The device must detect SE0 for at least 2 ms to start SRP when VBUS power is off.

4. To initiate SRP, the device turns on its data line pull-up resistor for 5 to 10 ms. The otg core detects data-line pulsing.

5. The device drives VBUS above the A-device session valid (2.0 V minimum) for VBUS pulsing. The otg core interrupts the application on detecting SRP. The Session Request Detected bit is set in Global Interrupt Status register (GINTSTS.SessReqInt).

6. The application must service the Session Request Detected interrupt and turn on the Port Power bit by writing the Port Power bit in the Host Port Control and Status register. The PHY indicates port power-on by asserting utmi_vbusvalid signal.
7. When the USB is powered, the device connects, completing the SRP process.

4.12.2 B-Device Session Request Protocol

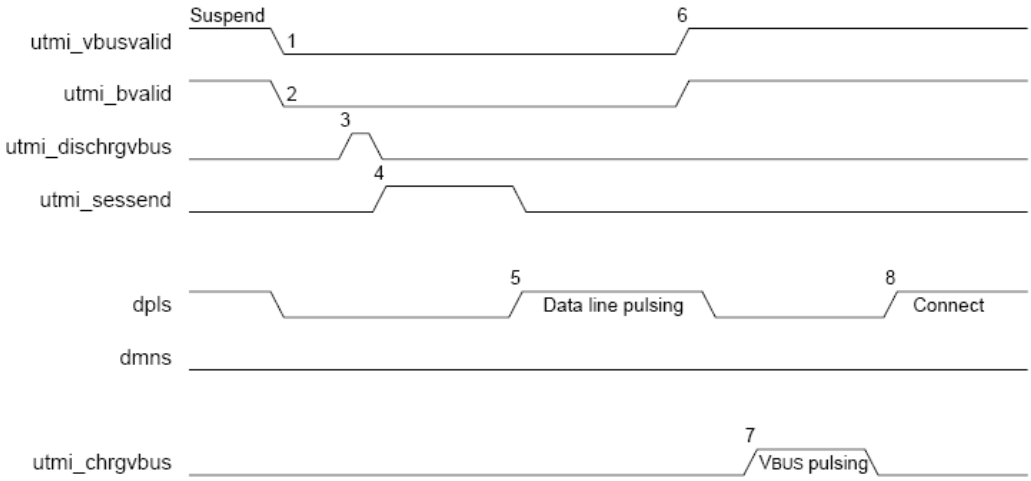


Figure 4.57 B-Device SRP

The application must set the SRP-Capable bit in the Core USB Configuration register. This enables the otg core to initiate SRP as a B-device. SRP is a means by which the otg core can request a new session from the host.

1. To save power, the host suspends and turns off port power when the bus is idle. PHY indicates port power off by deasserting the utmi_vbusvalid signal. The otg core sets the Early Suspend bit in the Core Interrupt register after 3 ms of bus idleness. Following this, the otg core sets the USB Suspend bit in the Core Interrupt register.

2. The PHY indicates the end of the B-device session by deasserting the utmi_bvalid signal.

3. The otg core asserts the utmi_dischrgvbus signal to indicate to the PHY to speed up VBUS discharge.

4. The PHY indicates the session's end by asserting the utmi_sessend signal. This is the initial condition for SRP. The otg core requires 2 ms of SE0 before initiating SRP. For a USB 1.1 full-speed serial transceiver, the application must wait until VBUS discharges to 0.2 V after GOTGCTL.BSesVld is deasserted. This discharge time can be obtained from the transceiver vendor and varies between different transceivers.

5. The application initiates SRP by writing the Session Request bit in the OTG Control and Status register. The otg core perform data-line pulsing followed by VBUS pulsing.

6. The host detects SRP from either the data-line or VBUS pulsing, and turns on VBUS. The PHY indicates VBUS power-on by asserting utmi_vbusvalid.

7. The otg core performs VBUS pulsing by asserting utmi_chrgvbus. The host starts a new session by turning on VBUS, indicating SRP success. The otg core interrupts the application by setting the Session Request Success Status Change bit in the OTG Interrupt Status register. The application reads the Session Request Success bit in the OTG Control and Status register.

8. When the USB is powered, the otg core connects, completing the SRP process.

4.12.3 A-Device Host Negotiation Protocol

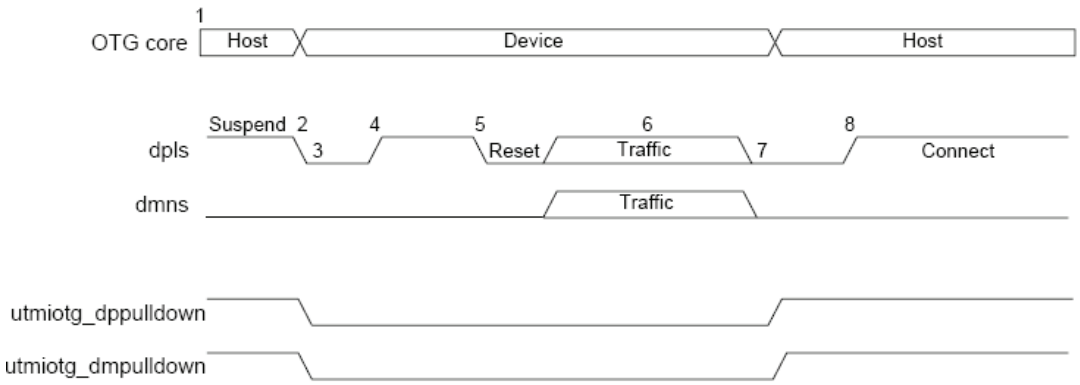


Figure 4.58 A-Device HNP

HNP switches the USB host role from the A-device to the B-device. The application must set the HNP-Capable bit in the Core USB Configuration register to enable the otg core to perform HNP as an A-device.

1. The otg core sends the B-device a SetFeature b_hnp_enable descriptor to enable HNP support. The B-device's ACK response indicates that the B-device supports HNP. The application must set Host Set HNP Enable bit in the OTG Control and Status register to indicate to the otg core that the B-device supports HNP.

2. When it has finished using the bus, the application suspends by writing the Port Suspend bit in the Host Port Control and Status register.

3. When the B-device observes a USB suspend, it disconnects, indicating the initial condition for HNP. The B-device initiates HNP only when it must switch to the host role; otherwise, the bus continues to be suspended. The otg core sets the Host Negotiation Detected interrupt in the OTG Interrupt Status register, indicating the start of HNP.

4. The otg core deasserts the utmiotg_dppulldown and utmiotg_dmpulldown signals to indicate a device role. The PHY enable the D + pull-up resistor indicates a connect for B-device. The application must read the Current Mode bit in the OTG Control and Status register to determine Device mode operation.

5. The B-device detects the connection, issues a USB reset, and enumerates the otg core for data traffic.

6. The B-device continues the host role, initiating traffic, and suspends the bus when done. The otg core sets the Early Suspend bit in the Core Interrupt register after 3 ms of bus idleness. Following this, the otg core sets the USB Suspend bit in the Core Interrupt register.

7. In Negotiated mode, the otg core detects the suspend, disconnects, and switches back to the host role. The otg core asserts the utmiotg_dppulldown and utmiotg_dmpulldown signals to indicate its assumption of the host role. The otg core sets the Connector ID Status Change interrupt in the OTG Interrupt Status register. The application must read the connector ID status in the OTG Control and Status register to determine the otg core's operation as an A-device. This indicates the completion of HNP to the application. The application must read the Current Mode bit in the OTG Control and Status register to determine Host mode operation.

8. The B-device connects, completing the HNP process.

4.12.4 B-Device Host Negotiation Protocol

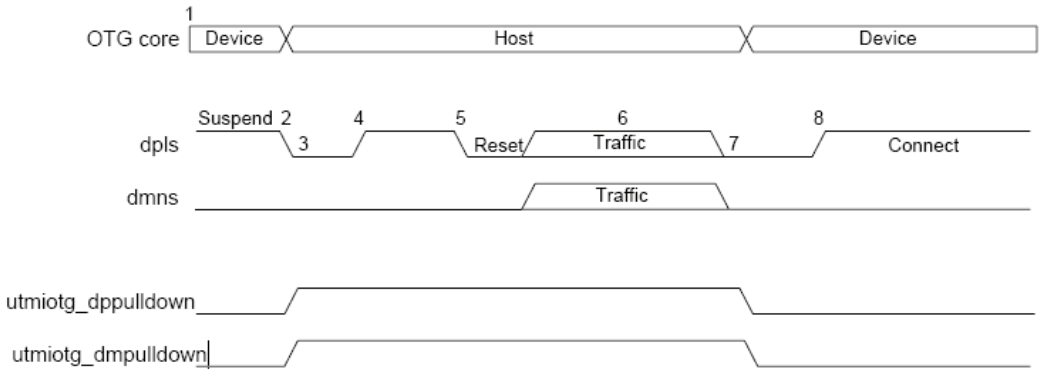


Figure 4.59 B-Device HNP

HNP switches the USB host role from B-device to A-device. The application must set the HNP-Capable bit in the Core USB Configuration register to enable the otg core to perform HNP as a B-device.

1. The A-device sends the SetFeature b_hnp_enable descriptor to enable HNP support. The otg core's ACK response indicates that it supports HNP. The application must set the Device HNP Enable bit in the OTG Control and Status register to indicate HNP support. The application sets the HNP Request bit in the OTG Control and Status register to indicate to the otg core to initiate HNP.

2. When it has finished using the bus, the A-device suspends by writing the Port Suspend bit in the Host Port Control and Status register. The otg core sets the Early Suspend bit in the Core Interrupt register after 3 ms of bus idleness. Following this, the otg core sets the USB Suspend bit in the Core Interrupt register.

3. The otg core disconnects and the A-device detects SE0 on the bus, indicating HNP. The otg core asserts the utmiotg_dppulldown and utmiotg_dmpulldown signals to indicate its assumption of the host role.

4. The A-device responds by activating its D + pull-up resistor within 3 ms of detecting SE0. The otg core detects this as a connect. The otg core sets the Host Negotiation Success Status Change interrupt in the OTG Interrupt Status register, indicating the HNP status. The application must read the Host Negotiation Success bit in the OTG Control and Status register to determine host negotiation success. The application must read the Current Mode bit in the Core Interrupt register (GINTSTS) to determine Host mode operation.

5. The otg core issues a USB reset and enumerates the A-device for data traffic.

6. The otg core continues the host role of initiating traffic, and when done, suspends the bus by writing the Port Suspend bit in the Host Port Control and Status register.

7. In Negotiated mode, when the A-device detects a suspend, it disconnects and switches back to the host role. The otg core deasserts the utmiotg_dppulldown and utmiotg_dmpulldown signals to indicate the assumption of the device role. The application must read the Current Mode bit in the Core Interrupt (GINTSTS) register to determine the Host mode operation.

8. The otg core connects, completing the HNP process.

4.12.5 Clock Gating

You can use clock gating to reduce power consumption when the USB is suspended or the session is not valid. The PHY turns off the PHY clock for as long as the core asserts the suspend_n signal to the PHY. The AHB clock to the HSOTG internal modules can also be gated by writing to the Gate hclk bit in the Power and Clock Gating Control register. The following sections show the procedures you must follow to use the clock gating feature.

4.12.5.1 Host Mode Suspend and Resume With Clock Gating

1. The application sets the Port Suspend bit in the Host Port CSR, and the core drives a USB suspend.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, the core gates the hclk (hclk_gated) to AHB-domain modules other than the BIU.

3. The core remains in Suspend mode.

4. The application clears the Gate hclk and Stop PHY Clock bits, and the PHY clock is generated.

5. The application sets the Port Resume bit, and the core starts driving Resume signaling.

6. The application clears the Port Resume bit after at least 20 ms.

7. The core is in normal operating mode.

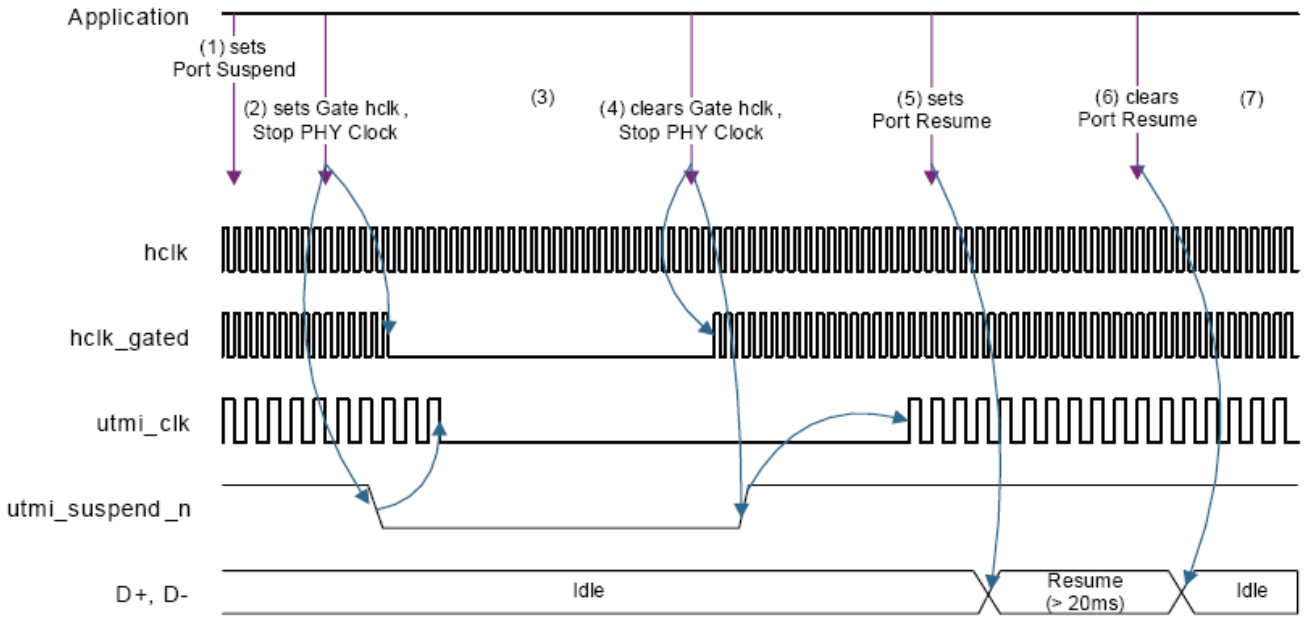


Figure 4.60Host Mode Suspend and Resume With Clock Gating

4.12.5.2 Host Mode Suspend and Remote Wakeup With Clock Gating

Sequence of operations

1. The application sets the Port Suspend bit in the Host Port CSR, and the core drives a USB suspend.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

3. The core remains in Suspend mode.

4. The Remote Wakeup signaling from the device is detected. The core deasserts the suspend_n signal to the PHY to generate the PHY clock. The core generates a Remote Wakeup Detected interrupt.

5. The application clears the Gate hclk and Stop PHY Clock bits. The core sets the Port Resume bit.

6. The application clears the Port Resume bit after at least 20 ms.

7. The core is in normal operating mode.

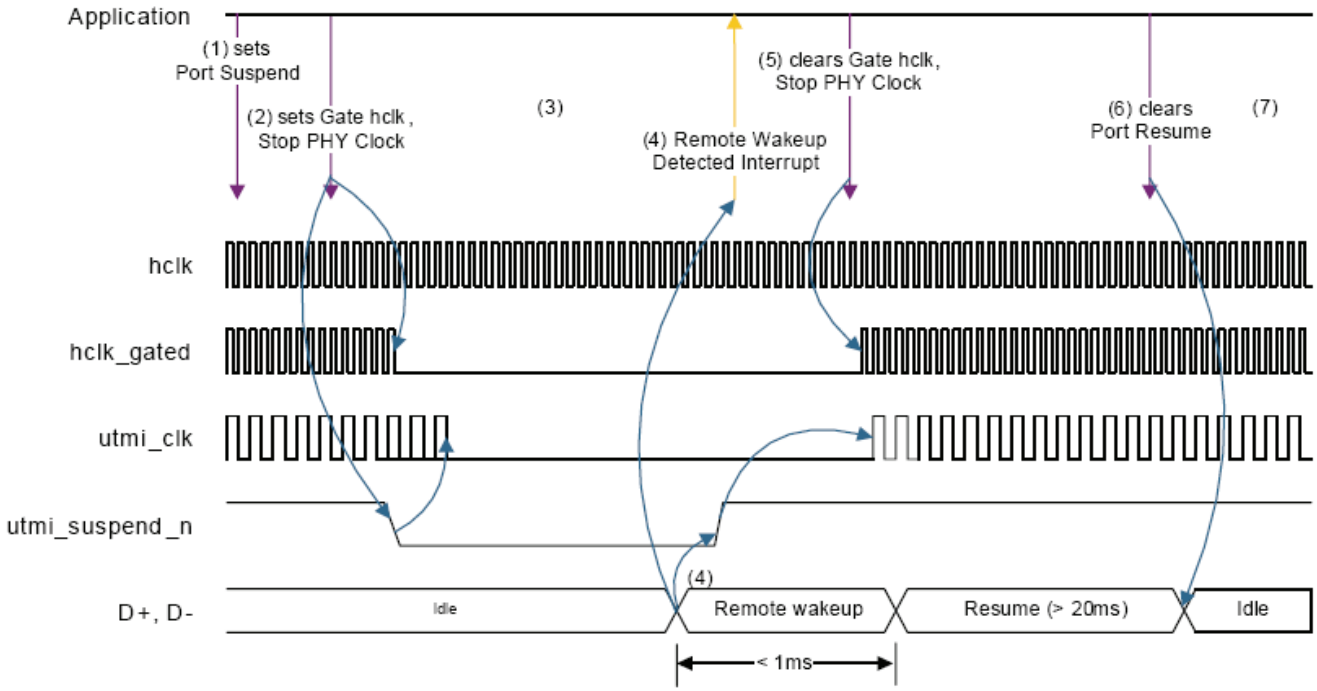


Figure 4.61Host Mode Suspend and Remote Wakeup With Clock Gating

4.12.5.3 Host Mode Session End and Start With Clock Gating

Sequence of operations

1. The application sets the Port Suspend bit in the Host Port CSR, and the core drives a USB suspend.

2. The application clears the Port Power bit. The core turns off VBUS.

3. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

4. The core remains in Low-Power mode.

5. The application clears the Gate hclk bit and the application clears the Stop PHY Clock bit to start the PHY clock.

6. The application sets the Port Power bit to turn on VBUS.

7. The core detects device connection and drives a USB reset.

8. The core is in normal operating mode.

4.12.5.4 Host Mode Session End and SRP With Clock Gating

Sequence of operations

1. The application sets the Port Suspend bit in the Host Port CSR, and the core drives a USB suspend.

2. The application clears the Port Power bit. The core turns off VBUS.

3. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating

Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

4. The core remains in Low-Power mode.

5. SRP (data line pulsing) from the device is detected. The core deasserts the suspend_n signal to the PHY to generate the PHY clock. An SRP Request Detected interrupt is generated.

6. The application clears the Gate hclk bit and the Stop PHY Clock bit.

7. The core sets the Port Power bit to turn on VBUS.

8. The core detects device connection and drives a USB reset.

9. The core is in normal operating mode.

4.12.5.5 Device Mode Suspend and Resume With Clock Gating

Sequence of operations

1. The core detects a USB suspend and generates a Suspend Detected interrupt.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

3. The core remains in Suspend mode.

4. The Resume signaling from the host is detected. The core deasserts the suspend_n signal to the PHY to generate the PHY clock. A Resume Detected interrupt is generated.

5. The application clears the Gate hclk bit and the Stop PHY Clock bit.

6. The host finishes Resume signaling.

7. The core is in normal operating mode.

4.12.5.6 Device Mode Suspend and Remote Wakeup With Clock Gating

Sequence of operations

1. The core detects a USB suspend and generates a Suspend Detected interrupt.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

3. The core remains in Suspend mode.

4. The application clears the Gate hclk bit and the Stop PHY Clock bit.

5. The application sets the Remote Wakeup bit in the Device Control register, the core starts driving Remote Wakeup signaling.

6. The host drives Resume signaling.

7. The core is in normal operating mode.

4.12.5.7 Device Mode Session End and Start With Clock Gating

Sequence of operations

1. The core detects a USB suspend, and generates a Suspend Detected interrupt. The host turns off VBUS.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

3. The core remains in Low-Power mode.

4. The new session is detected (bsessvld is high). The core deasserts the suspend_n signal to the PHY to generate the PHY clock. A New Session Detected interrupt is generated.

5. The application clears the Gate hclk and Stop PHY Clock bits.

6. The core detects USB reset.

7. The core is in normal operating mode

4.12.5.8 Device Mode Session End and SRP With Clock Gating

Sequence of operations

1. The core detects a USB suspend, and generates a Suspend Detected interrupt. The host turns off VBUS.

2. The application sets the Stop PHY Clock bit in the Power and Clock Gating Control register, the core asserts the suspend_n signal to the PHY, and the PHY clock stops. The application sets the Gate hclk bit in the Power and Clock Gating Control register, and the core gates the hclk (hclk_ctl) to AHB-domain modules other than the BIU.

3. The core remains in Low-Power mode.

4. The application clears the Gate hclk and Stop PHY Clock bits.

5. The application sets the SRP Request bit, and the core drives data line and VBUS pulsing.

6. The host turns on VBUS, detects device connection, and drives a USB reset.

7. The core is in normal operating mode.

4.13 Miscellaneous Topics

4.13.1 Data FIFO RAM Allocation

If Dynamic FIFO Sizing is enabled in the core, the External RAM must be allocated among different FIFOs in the core before any transactions can start. The application must follow this procedure every time it changes core FIFO RAM allocation.

The application must allocate data RAM per FIFO based on the AHB's operating frequency, the PHY Clock frequency, the available AHB bandwidth, and the performance required on the USB. Based on the above mentioned criteria, the application must provide a table as described below with RAM sizes for each FIFO in each mode.

4.13.1.1 Device Mode

[Dedicated Tx FIFO Operation]

When allocating data RAM for FIFOs in Device mode when dedicated TX FIFO is used, keep in mind these factors:

1. Receive FIFO RAM Allocation:

- RAM for SETUP Packets: 4 * n + 6 locations must be Reserved in the receive FIFO to receive up to n SETUP packets on control endpoints, where n is the number of control endpoints the device core supports. The core does not use these locations, which are Reserved for SETUP packets, to write any other data.
- 1 location for Global OUT NAK
- Status information is written to the FIFO along with each received packet. Therefore, a minimum space of (Largest Packet Size / 4) + 1 must be allotted to receive packets. If a high-bandwidth endpoint is enabled, or multiple isochronous endpoints are enabled, then at least two (Largest Packet Size / 4) + 1 spaces must be allotted to receive back-to-back packets. Typically, two (Largest Packet Size / 4) + 1 spaces are recommended so that when the previous packet is being transferred to AHB, the USB can receive the subsequent packet. If AHB latency is high, you must allocate enough space to receive multiple packets. This is critical to prevent dropping any isochronous packets.

- Along with each endpoint's last packet, transfer complete status information is also pushed to the FIFO. Typically, one location for each OUT endpoint is recommended.

2. Transmit FIFO RAM Allocation:

- The minimum RAM space required for each IN Endpoint Transmit FIFO is the maximum packet size for that particular IN endpoint.
- The more space allocated in the transmit IN Endpoint FIFO results in a better performance on the USB and can hide the latencies on the AHB.

FIFO Name	Data RAM Size
Receive dta FIFO	rx_fifo_size. This must include RAM for setup packets, OUT endpoint control information and data OUT packets, as mentioned earlier.
Transmit FIFO 0	Tx_fifo_size[0]
Transmit FIFO 1	Tx_fifo_size[1]
….	…
Transmit FIFO i	Tx_fifo_size[i]

With this information at hand, the following registers must be programmed as follows:

1. Receive FIFO Size Register (GRXFSIZ)

- GRXFSIZ.Receive FIFO Depth = rx_fifo_size;

2. Device IN Endpoint Transmit FIFO0 Size Register (GNPTXFSIZ)

- GNPTXFSIZ.non-periodic Transmit FIFO Depth = tx_fifo_size[0];
- GNPTXFSIZ.non-periodic Transmit RAM Start Address = rx_fifo_size;

3. Device IN Endpoint Transmit FIFO#1 Size Register (DIEPTXF1)

- DIEPTXF1. Transmit RAM Start Address = GNPTXFSIZ.FIFO0 Transmit RAM Start Address + tx_fifo_size[0];

4. Device IN Endpoint Transmit FIFO#2 Size Register (DIEPTXF2)

- DIEPTXF2.Transmit RAM Start Address = DIEPTXF1.Transmit RAM Start Address + tx_fifo_size[1];

5. Device IN Endpoint Transmit FIFO#i Size Register (DIEPTXFi)

- DIEPTXFm.Transmit RAM Start Address = DIEPTXFi-1.Transmit RAM Start Address + tx_fifo_size[i-1];

6. The transmit FIFOs and receive FIFO must be flushed after the RAM allocation is done, for the proper functioning of the FIFOs.

- GRSTCTL.TxFNum = 5’h10
- GRSTCTL.TxFFlush = 1’b1
- GRSTCTL.RxFFlush = 1’b1
- The application has to wait until the TxFFlush bit and the RxFFlush bits are cleared, before performing any operation on the core.

[Dedicated Tx FIFO operation with Thresholding]

1. Receive FIFO RAM allocation

- RAM for Setup Packets: 7*n + 6 locations have to be Reserved in the receive FIFO, to receive up to “n” setup packets on control endpoints, where “n” is the number of control endpoints supported by the device core. These locations Reserved for Setup Packets are not used by the core, to write any other data.

- 1 location for Global OUT NAK
- It is recommended to have an Rx FIFO space for two thresholds. Along with each received threshold, a status information is also written in to the FIFO. With the last threshold of a packet, two status DWORDs are written into the FIFO. With the last packet of a transfer, transfer complete status information is written into the FIFO. So worst case , a minimum of 2*(Rx_threshold length/4 + 4) space is needed to be allocated to receive a packet.

2. Transmit FIFO RAM Allocation:

- The minimum RAM space required for each IN Endpoint Transmit FIFO is min(2*Tx_threshold_length, endpoint_max_pkt_size).
- The more space allocated in the transmit IN Endpoint FIFO results in a better performance on the USB and can hide the latencies on the AHB.

If high AHB latencies results in underrun error very often, then there is a possibility that the Host might disable this endpoint because of Errors in the packet (typically when there is error in 3 consecutive packets). So thresholding must be enabled only when the AHB latency is not very high.

4.13.1.2 Host Mode

With this information at hand, the following registers must be programmed as follows:

1. Receive FIFO Size Register (GRXFSIZ)

- GRXFSIZ.RxFDep = rx_fifo_size;

2. Non-periodic Transmit FIFO Size Register (GNPTXFSIZ)

- GNPTXFSIZ.NPTxFDe = tx_fifo_size[0];
- GNPTXFSIZ.NPTxFStAddr = rx_fifo_size;

3. Host Periodic Transmit FIFO Size Register (HPTXFSIZ)

- HPTXFSIZ.PTxFSize = tx_fifo_size[1];
- HPTXFSIZ.PTxFStAddr = GNPTXFSIZ.NPTxFStAddr + tx_fifo_size[0];

4. The transmit FIFOs and receive FIFO must be flushed after RAM allocation for proper FIFO function.

- GRSTCTL.TxFNum = 5’h10
- GRSTCTL.TxFFlush = 1’b1
- GRSTCTL.RxFFlush = 1’b1
- The application must wait until the TxFFlush bit and the RxFFlush bits are cleared before performing any operation on the core.

4.13.1.3 Calculating the Total FIFO Size for OTG

[Dedicated FIFO Mode with No Thresholding]

The otg RxFIFO is shared between the host and device. The Host TxFIFOs are also shared with Device IN endpoint TxFIFOs 0 through n.

There are three ways to calculate the total FIFO size. The total FIFO size will depend on whether you support high-bandwidth transfers with high AHB latency.

Method 1

Use this method if you are using the following conditions:

- Minimum FIFO depth allocation
- No support for high-bandwidth endpoints
- The FIFO must equal at least one MaxPacketSize (MPS).

Device RxFIFO = (4 * number of control endpoints + 6) + ((largest USB packet used / 4) + 1 for status information) + (2 * number of OUT endpoints) + 1 for Global NAK

Note : Include the control OUT endpoint in the “number of OUT endpoints.”

Host RxFIFO = (largest USB packet used / 4) + 1 for status information + 1 transfer complete

Host Non-Periodic TxFIFO = largest non-periodic USB packet used / 4

Host Periodic TxFIFO = largest periodic USB packet used / 4

Device IN Endpoint TxFIFOs (a separate FIFO is allocated to each IN endpoint) = IN Endpoints Max packet Size / 4

OTG Total RAM = (Device RxFIFO or Host RxFIFO; choose the largest one) + {((Host Non-Periodic TxFIFO + Host peiodic TxFIFO) OR Device IN Endpoint TxFIFO #0 + #1 + #2 + #n); choose the largest one}

Example

The maximum packet size (MPS) is 1,024 bytes for a periodic USB packet and 512 bytes for a non-periodic USB packet. There are three OUT endpoints, three IN endpoints, and one control endpoints. One of the IN endpoints is isochronous.

Device RxFIFO = (4 * 1 + 6) + ((1,024 / 4) +1) + (2 * 4) + 1 = 276

Host RxFIFO = ((1,024 / 4) + 1) + 1 = 258

Host Non-Periodic TxFIFO = (512 / 4) = 128

Host Periodic TxFIFO = (1,024 / 4) = 256

Device IN Endpoint TxFIFO:

FIFO #0= (64/ 4) = 16 (Assuming this is used for EP0)
FIFO #1= (512 / 4) = 128
FIFO #2 = (512/4) = 128
FIFO #3= (1,024 / 4) = 256 (Assuming this is used for Isochronous).

OTG Total RAM = max(276,258) + max(384,528) = 804.

Method 2

Use this method if you are using the recommended minimum FIFO depth allocation with support for high-bandwidth endpoints. This FIFO allocation enables the core to transfer a packet on the USB while the previous (next) packet is simultaneously transferred to the AHB.

This FIFO allocation improves the core’s performance .

Device RxFIFO = (4 * number of control endpoints + 6) + 2 * ((largest USB packet used / 4) + 1) + (2 * number of OUT endpoints) + 1

Note: Include the control OUT endpoint in the “number of OUT endpoints.”

Host RxFIFO = 2 * ((largest USB packet used / 4) + 1 + 1)

Host Non-Periodic TxFIFO = 2 * (largest non-periodic USB packet used / 4)

Host Periodic TxFIFO = 2 * (largest periodic USB packet used / 4)

Device IN Endpoint-Specific TxFIFOs (a separate FIFO is allocated to each endpoint) = 2*(max_pkt_size for the endpoint)/4.

OTG Total RAM = (Device RxFIFO or Host RxFIFO; choose the largest one) + {((Host Non-Periodic TxFIFO + Host peiodic TxFIFO) OR Device IN Endpoint TxFIFO #0 + #1 + #2 + #n); choose the largest one }

Example

The MPS is 1,024 bytes for a periodic USB packet and is 512 bytes for a non-periodic USB packet. There are three OUT endpoints, three IN endpoints, and one control endpoint. .One of the IN endpoint is Isochronous and the maximum number of periodic data packets per transfer for Endpoint 3 is 3.

Device RxFIFO = (4 * 1 + 6) + 2 * ((1,024 / 4) + 1) + (2 * 4) + 1 = 533

Host RxFIFO = 2 * ((1,024 / 4) + 1 + 1) = 516

Host Non-Periodic TxFIFO = 2 * (512 / 4) = 256

Host Periodic TxFIFO = 2 * (1,024 / 4) = 512

Device IN Endpoint TxFIFO:

FIFO #0 = 2*(64 / 4) = 32(Assuming used for Control Endpoint)
FIFO #1 = 2*(512 / 4) = 256
FIFO #2 = 2*(512 / 4) = 256
FIFO #3 = 2*(1024 / 4) = 256 * 2= 512 (Assuming used for High Bandwidth Endpoint)

OTG total RAM = max(533,516) + max(768,1056)= 1589

Method 3

Use this method if you are using the recommended FIFO depth allocation that supports high-bandwidth endpoints and high AHB latency.

Notes: x = (AHB latency + time to transfer largest packet on AHB) / time to transfer largest packet on USB. The value of x is an integer. Any fractional value is rounded to the nearest integer. For example: x = 20 µs / 17,039 µs = 1.17 µs = 2 µs.

Device RxFIFO = (4 * number of control endpoints + 6) + (x + 1) * ((largest USB packet used / 4) + 1) + (2 * number of OUT endpoints) + 1

Note : Include the control OUT endpoint in the “number of OUT endpoints.”

Host RxFIFO = (x + 1) * ((largest USB packet used / 4) + 1 + 1)

Host Non-Periodic TxFIFO = (x + 1) * (largest non-periodic USB packet used / 4)

Host Periodic TxFIFO = (x + 1) * (largest periodic USB packet used / 4)

Device IN Endpoint-Specific TxFIFOs (a separate FIFO is allocated to each endpoint) = (x+1)*(max_pkt_size for the endpoint)/4e

OTG Total RAM = (Device RxFIFO or Host RxFIFO; choose the largest one) + {((Host Non-Periodic TxFIFO + Host peiodic TxFIFO) OR Device IN Endpoint TxFIFO #0 + #1 + #2 + #n); choose the largest one}

For example, the otg core is operating as a device and receives three isochronous backto-back packets (high bandwidth). The RxFIFO is configured for only 2 MPS (2 kB). On the USB, it takes 2.08 ns (high speed) to transfer 1 bit of data. To transfer 1,024 bytes (8,192 bits), it takes approximately 17,039 µs (2.08 ns * 8,192) to complete the packet transfer from the USB to RxFIFO. After the full packet is in the RxFIFO (assume you are using DMA mode), the DMA begins to transfer the data packet from the RxFIFO to system memory. The AHB latency plus the time to transfer 1,024 bytes on the AHB is 20 µs.

A break-down of the sequence is as follows:

1. First isochronous packet from USB to RxFIFO = 17,039 µs.

2. DMA begins to fetch data from RxFIFO to memory; time = 0.

3. Second isochronous packet from USB to RxFIFO = 17,039 µs.

4. Because there is an extra 1 MPS RxFIFO, the core can receive the next packet regardless of whether the DMA has completed the transfer on the AHB.

5. Third isochronous packet from USB to RxFIFO = 17,039 µs.

If the DMA transfer on the AHB for the first packet is still not complete after 17,039 µs, no FIFO space is available for the third transaction. In this case, the AHB latency is 20 µs; therefore, the transfer of the first packet still has not completed. The OTG core does not issue an interrupt for the transfer completed for the third packet and the transfer times out on the USB side. To compensate for the high AHB latency, you can configure the RxFIFO for 3 more MPS.

Example

The MPS for a periodic USB packet is 1,024 bytes and 512 bytes for a non-periodic USB packet. There are three OUT endpoints, three IN endpoints, and one control endpoint. One of the IN Endpoint is Isochronous and the maximum number of periodic data packets per transfer for Endpoint 3 is 3. The AHB latency plus the time to transfer 1,024 bytes on the AHB is 20 µs.

Device RxFIFO = (4 * 1 Host Periodic TxFIFO = (2 + 1) * (1,024 / 4) = 768

Host RxFIFO = (2 + 1) * ((1,024 / 4) + 1 + 1) = 774

Host Non-Periodic TxFIFO = (2 + 1) * (512 / 4) = 384

Device IN ENdpoint TxFIFO:

FIFO #0 = (2+1) * (64/4) = 48 (Assuming to be used for Control Endpoint)
FIFO #1 = (2+1) * (512 / 4) = 384
FIFO #2 = (2+1) * (512 / 4) = 384
FIFO #3 = 3* (1,024 / 4) = 256 * 3= 768 (Assuming to be used for Isochronous Endpoint)

OTG total RAM = max(790,774) + max(1152, 1584) = 2374+ 6) + (2 + 1) * ((1,024 / 4) +1) + (2 * 4) + 1 =790

[Dedicated FIFO Mode with thresholding]

The main intention of threshold support are the following

- To have a smaller FIFO size
- To have faster DMA response.

Thresholding is supported only in device mode. So if your device does not have many IN endpoints (not more than 2) OR if your calculated total device FIFO depth without thresholding is not more than the required host FIFO depth, then you are not going to save FIFO space much by enabling thresholding.

It is strongly recommended that you enable dynamic FIFO sizing when thresholding is enabled.

Device RxFIFO = (7 * number of control endpoints + 6) + 2 * ((min(largest USB packet used, Rx_threshold_length) / 4) + 4) + 1 for Global NAK.

Host RxFIFO = (largest USB packet used / 4) + 1 for status information + 1 transfer complete

Host Non-Periodic TxFIFO = largest non-periodic USB packet used / 4

Host Periodic TxFIFO = largest periodic USB packet used / 4

Device IN Endpoint TxFIFOs (a separate FIFO is allocated to each IN endpoint) = min(2 x Transmit Threshold size, IN Endpoints Max packet Size) / 4

OTG Total RAM = (Device RxFIFO or Host RxFIFO; choose the largest one) + {((Host Non-Periodic TxFIFO + Host peiodic TxFIFO) OR Device IN Endpoint TxFIFO #0 + #1 + #2 + #n); choose the largest one}

Example

The maximum packet size (MPS) is 1,024 bytes for a periodic USB packet and 512 bytes for a non-periodic USB packet. There are three OUT endpoints, three IN endpoints, and one control endpoints. One of the IN endpoints is isochronous. The Rx threshold size is 128 bytes and the Tx threshold size is also 128 bytes.

Device RxFIFO = (7 * 1 + 6) + 2* ((128 / 4) +4) + 1 = 86

Host RxFIFO = ((1,024 / 4) + 1) + 1 = 258

Host Non-Periodic TxFIFO = (512 / 4) = 128

Host Periodic TxFIFO = (1,024 / 4) = 256

Device IN Endpoint TxFIFO:

FIFO #0= (64/ 4) = 16 (Assuming this is used for EP0)
FIFO #1= (128/ 4) = 32
FIFO #2 = (128/4) = 32
FIFO #3= (128 / 4) = 32(Asuming this is used for Isochronous).

OTG Total RAM = (Device RxFIFO or Host RxFIFO; choose the largest one) + ((Host Non-Periodic TxFIFO + Host periodic TxFIFO) OR Device IN Endpoint TxFIFO #0 + #1 + #2 + #n; choose the largest one)

OTG_TOTAL_RAM = max(258,86) + max((128+256),(16+32+32+32)) = 258 + 384 = 642.

4.13.2 Dynamic FIFO Allocation

The application can change the RAM allocation for each FIFO during the operation of the core.

Host Mode

In Host mode, before changing FIFO data RAM allocation, the application must determine the following.

- All channels are disabled
- All FIFOs are empty

Once these conditions are met, the application can reallocate FIFO data RAM as explained in “Data FIFO RAM Allocation”.

After reallocating the FIFO data RAM, the application must flush all FIFOs in the core using the GRSTCTL.TxFIFO Flush and GRSTCTL.RxFIFO Flush fields. Flushing is required to reset the pointers in the FIFOs for proper FIFO operation after reallocation.

Device Mode

In Device mode, before changing FIFO data RAM allocation, the application must determine the following.

- All IN and OUT endpoints are disabled
- NAK mode is enabled in the core on all IN endpoints
- Global OUT NAK mode is enabled in the core
- All FIFOs are empty

Once these conditions are met, the application can reallocate FIFO data RAM as explained in “Data FIFO RAM Allocation”. When NAK mode is enabled in the core, the core responds with a NAK handshake on all tokens received on the USB, except for SETUP packets.

After the reallocating the FIFO data RAM, the application must flush all FIFOs in the core using the GRSTCTL.TxFIFO Flush and GRSTCTL.RxFIFO Flush fields. Flushing is required to reset the pointers in the FIFOs for proper FIFO operation after reallocation.

4.13.3 Core Interrupt Handler

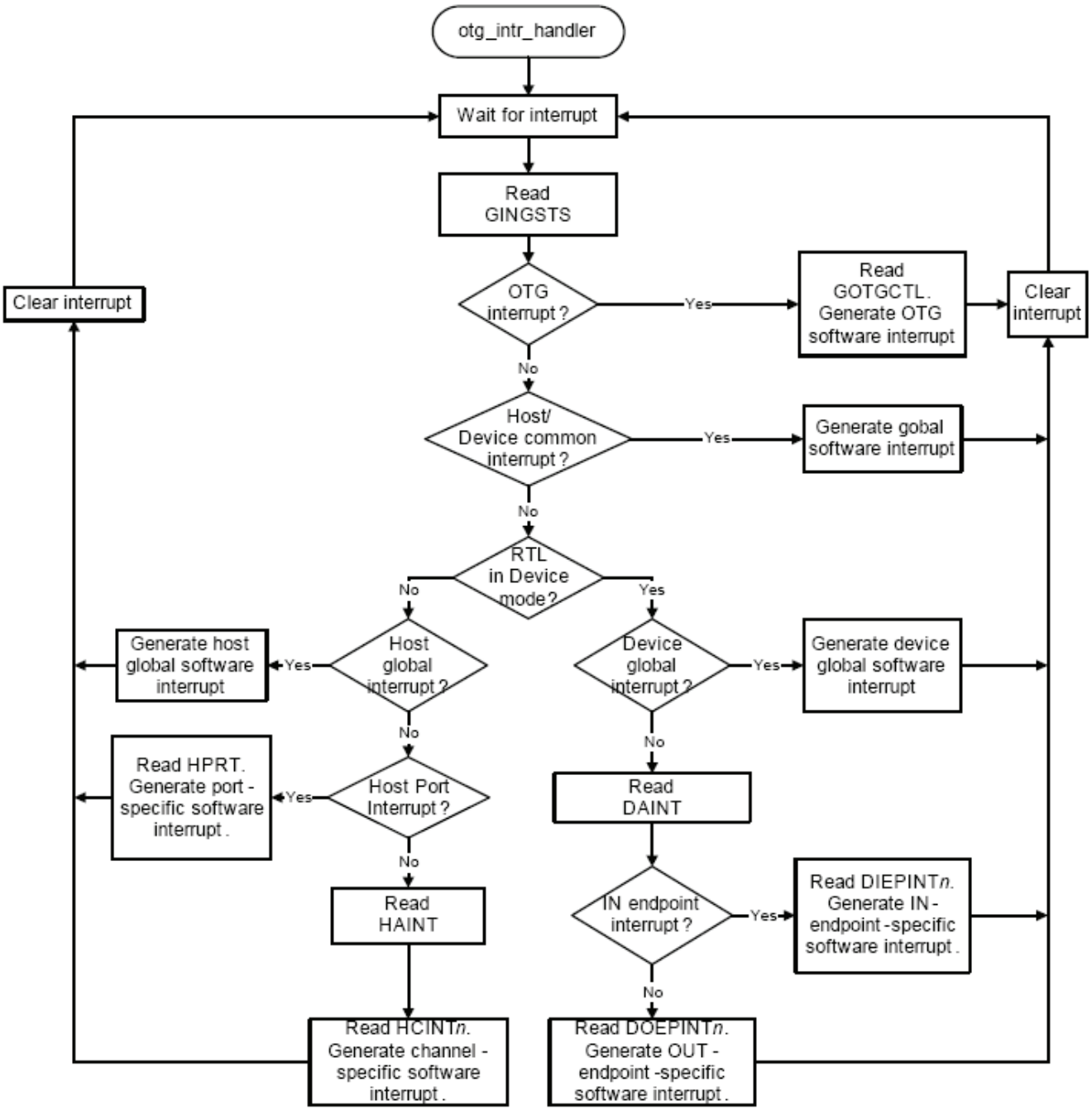


Figure 4.62 Core Interrupt Handler

5 Memory Stick Host Controller

5.1 Overview

This is a host controller with 32-bit CPU interface that supports Memory Stick Ver. 1.x and Memory Stick PRO-HG Duo. This conforms to “Memory Stick Standard Format Specifications ver. 1.4x” and “Memory Stick Standard Memory Stick PRO-HG Duo Format Specifications ver. 1.0x”

The memory stick host controller has the following features.

- Data transmit/receive FIFO (64 bits x 4)
- Hardware CRC Generator
- Memory Stick Serial Clock (Serial : 20MHz-MAX, Parallel : 60MHz-MAX)
- Burst Transfer via On-Chip DMA Controller

Listed below are major memory stick device and modules supported by this host controller. Memory Stick XC Series are not supported.

- Non Copyright Protected Memory Stick Media
- MagicGate Memory Stick Media
- MagicGate Memory Stick Media with Parallel Transfer Support
- Memory Stick PRO Media
- Memory Stick Micro Media
- Memory Stick PRO-HG Duo Media
- Memory Stick I/O Expansion Module
- Memory Stick PRO I/O Expansion Module+

The host controller is called “smshc_i” standing for **S**ony **M**emory **S**tick **H**ost **C**ontroller with **I-CON**. The smshc_i is a host controller supporting memory stick, equipped with a 32-bit AHB interface.

The smshc_i is composed of two blocks, smshc and I-CON. The former performs communications with memory stick while the latter is a sequencer block which automatically controls the smshc. Each block is mapped to an address which can be accessed by Host CPU via AHB interface.

The I-CON makes it possible to automatically manage the command control sequence for memory stick on behalf of a host CPU. This is how the workload imposed on the host CPU can be reduced. The Figure 5.1 shows this architecture simply.

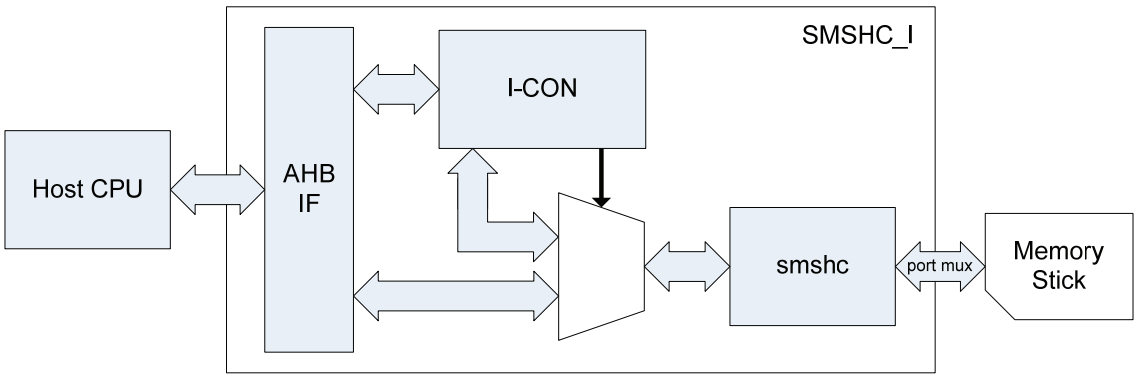


Figure 5.1 The Components in SMSHC_I

<u>smshc Block</u>

- The smshc can be controlled by either a host CPU or the I-CON.
  However, when it is controlled by one of the two, the other one can not control it.
- The smshc issues TPCs for communicating with memory stick.
- The smshc can select one of the following three interfaces for the communication with

memory stick; serial interface, 4-bit parallel interface, or 8-bit parallel interface.

- The smshc can transfer data up to 2048 bytes. That is the maximum size transferable by a single TPC.
- The smshc supports interrupts through 8-channel DATA lines from memory stick.
- FIFO, or [MSHC-Data Register] (64 bits × 4), is built in the smshc.
- Only when the I-CON is not used, the smshc can output a DMA transfer request for [MSHC-Data Register]. The smshc enables a host CPU to set whether or not a DMA transfer request can be output per TPC.

I-CON Block

- The I-CON automatically controls the smshc and can perform command control sequences to memory stick. A host CPU sets command control sequences for memory stick to [Instruction Queue], a buffer for instructions, as microcode.
- The I-CON runs instructions specified by microcode one after another in order.
- A FIFO for sending/receiving general data [General Data FIFO] is built in the I-CON.
- I-CON is equipped with a memory interface for [PageBuffer], a buffer which sends/receives page data.
- I-CON is equipped with a memory interface for [Instruction Queue], an instruction buffer.
- 2 channels are built in the I-CON for data transfer requests.
  (1) Data transfer requests for [PageBuffer]
  (2) Data transfer requests for [General Data FIFO]
- How the I-CON notifies a data transfer request can be changed with a host CPU. Refer to [System Register].

Memory Stick Signals and External Port Mux

Memory Stick Interface signals are as follows:

- SCLK : Clock output signal to Memory Stick device
- BS : Bus State signal of Memory Stick device
- DATA[7:0] : Data in/output signal for Memory Stick device

The bus state(BS) is classified into four states depending on the attributes and the transfer directions of data on DATA[7:0]. They include BS0 where no packet communication is performed, and BS1 to BS3 where packet communication is being performed. Each data output timing is controlled according to the bus states. The bus states from BS1 to BS3 are defined as a single packet, and each communication should be completed with a packet.

Table 5.1 Bus State

State	BS	Description	
BS0	Low	INT Transfer State : A state in which no packet communication is performed and an INT (interrupt) signal is sent from Memory Stick to a host controller.	
BS1	High	TPC Transfer State : A state in which a TPC is transferred from the host controller to the Memory Stick in order to start packet communication.	
BS2	Low	Read packet	Write packet
		Handshake State : The host controller waits for a RDY signal from the Memory Stick.	Data Transfer State : The host controller transfers data to the Memory Stick.
BS3	High	Read packet :	Write packet :
		Data Transfer State : Memory Stick transfers data to the host controller.	Handshake State : The host controller waits for a RDY signal from Memory Stick.

There are 8 channels for Memory Stick interface in NVS2310. Each Memory Stick interface signal is mapped to one port of them.

Block Diagram

The Figure 5.2 shows the block diagram of memory stick host controller.

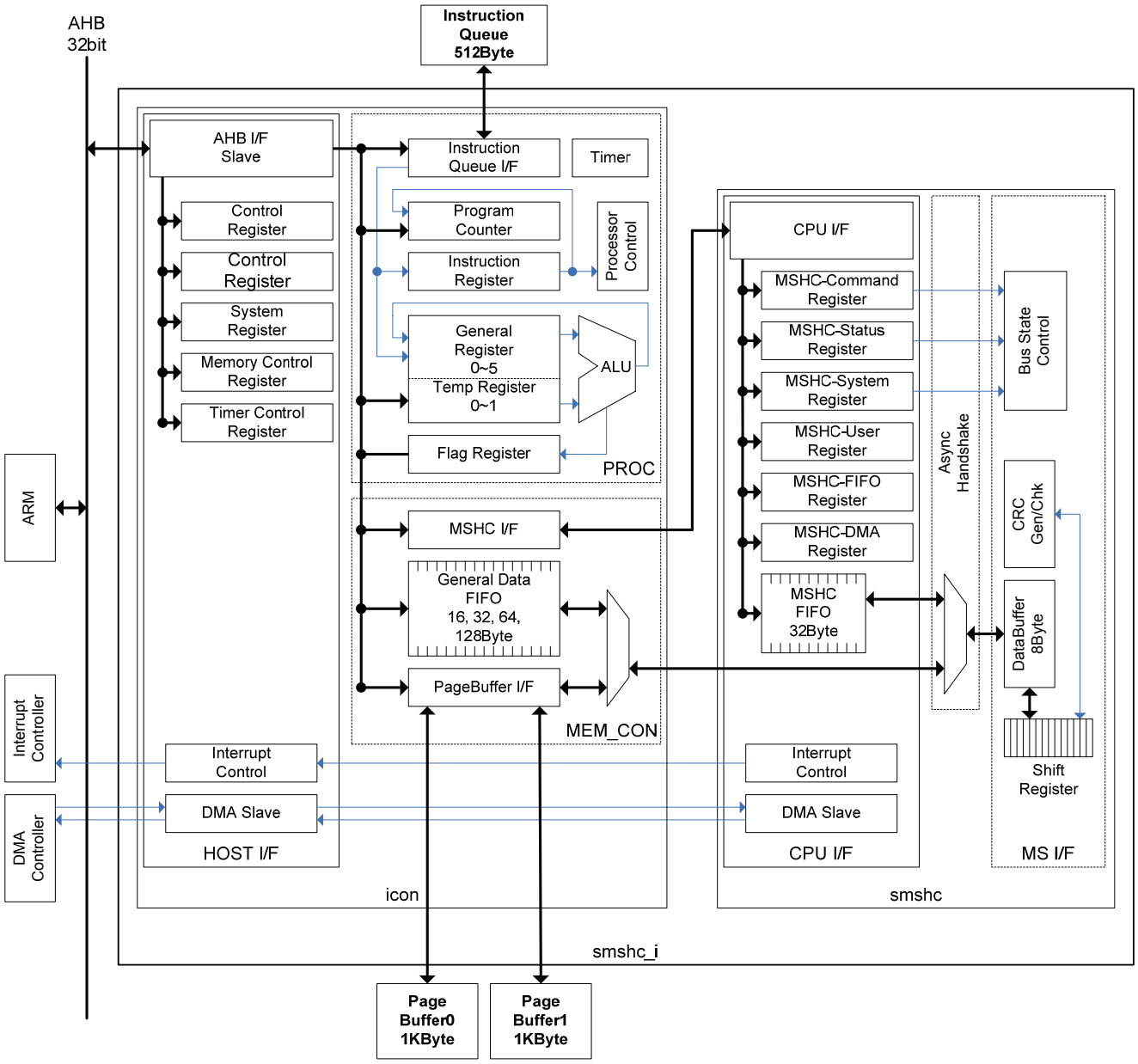


Figure 5.2 Memory Stick Host Controller Block Diagram

5.2 Register Descriptions

Table 5.2 SMSHC_I Register Map (Base Address = 0xB0840000)

Address	Name				Type	Default
	[31:24]	[23:16]	[15:8]	[7:0]		
0x00	Control Register		Program Counter Register		R/W	0x0070_1000
0x04	System Register		-		R/W	0x0800_XXXX
0x08	Flag Register		-		R	0x4000_XXXX
0x0C	Memory Control Register		-		R/W	0x0001_7000
0x10	General Register0		General Register1		R/W	0x8000_9000
0x14	General Register2		General Register3		R/W	0xA000_B000
0x18	General Register4		General Register5		R/W	0xC000_D000
0x1C	Timer Register		-		R	0xE000_XXXX
0x20	Instruction Queue				R/W	0xXXXX_XXXX
	LSB(Least Significant Byte) First Mode					
	[15:0]		[31:16]			
	MSB(Most Significant Byte) First Mode					
	[31:16]		[15:0]			
0x24	General Data FIFO				R/W	0x0000_0000
	LSB(Least Significant Byte) First Mode					
	[7:0]	[15:8]	[23:16]	[31:24]		
	MSB(Most Significant Byte) First Mode					
	[31:24]	[23:16]	[15:8]	[7:0]		
0x28	PageBuffer				R/W	0xXXXX_XXXX
	LSB(Least Significant Byte) First Mode					
	[7:0]	[15:8]	[23:16]	[31:24]		
	MSB(Most Significant Byte) First Mode					
	[31:24]	[23:16]	[15:8]	[7:0]		
0x2C	Version Register		-		R	0xXXXX_XXXX
0x30	MSHC-Command Register		-		R/W	0x0000_XXXX
0x34	MSHC-Data Register				R/W	0x0000_0000
	LSB(Least Significant Byte) First Mode					
	[7:0]	[15:8]	[23:16]	[31:24]		
	MSB(Most Significant Byte) First Mode					
	[31:24]	[23:16]	[15:8]	[7:0]		
0x38	MSHC-Status Register		-		R	0x1000_XXXX
0x3C	MSHC-System Register		-		R/W	0x20A5_XXXX
0x40	MSHC-User Custom Register		-		R	0x0220_XXXX
0x44	MSHC-FIFO Control Register		-		R/W	0x0001_XXXX
0x4C	MSHC-DMA Control Register		-		R/W	0x0000_XXXX

Table 5.3 PORTCFG Register Map (Base Address = 0xB0870100)

Name	Address	Type	Reset	Description
PORTCFG	0x00	R/W	0x00000000	Port Configuration Register

Table 5.4 PORTDLY Register Map (Base Address = 0xB0870104)

Name	Address	Type	Reset	Description
PORTDLY	0x00	R/W	0x00000000	Port Output Delay Register

Control / Program Counter Register 0xB0840000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								GDIR	GRPN		PDIR	INTC	INT	0	START
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0			1	0				PRGC							

Field	Name	RW	Reset	Description
31-24	-	R/W	-	Reseved
23	GDIR	R/W	0x0	**General Data FIFO Direction** 0: CPU ← [General Data FIFO] ← Reading data from memory stick 1: CPU → [General Data FIFO] → Writing data to memory stick
22-21	GRPN	R/W	0x3	**General Register for PageBuffer Data Number** 00: [General Register0] 01: [General Register1] 10: [General Register2] 11: No writing to [PageBuffer]
20	PDIR	R/W	0x1	**PageBuffer Direction** 0: CPU ← [PageBuffer] ← Reading data from memory stick 1: CPU → [PageBuffer] → Writing data to memory stick
19	INTC	R/W	0x0	**INT Clear** 0: Not clear 1: Interrupt Clear, Automatically back to 0 after clearing the interrupt
18	INT	R	0x0	**Interrupt (Read Only)** 0: An interrupt has not occurred in the I-CON block. 1: An interrupt has occurred in the I-CON block.
17	-	-	-	Reseved
16	START	R/W	0x0	**Start** 0: Do not start self-run. 1: The I-CON can be a self-run state. The address for starting to proceed an instruction is specified by the PRGC bits in [Program Counter Register].
15-8	-	-	-	Reserved
7-0	PRGC	R/W	0x00	**Program Counter** These bits are an address pointer for writing/reading/executing each instruction to/from/in [Instruction Queue]. To write an instruction, the address to start writing from shall be first set to PRGC. After the instruction has been written, these bits can be automatically incremented.

System Register 0xB0840004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAIM1	DMAIM0	DMAE1	DMAE0	DMASL[1:0]		GDSZ[1:0]		DMACH	PDSZ[2:0]			0	INTE	0	SRST
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-															

Field	Name	RW	Reset	Description
31	DMAIM1	R/W	0x0	**DMA1 Interrupt Mode** 0: Disable 1: This bit enables the occurrence of an interrupt for a data transfer request (Channel1).
30	DMAIM0	R/W	0x0	**DMA0 Interrupt Mode** 0: Disable 1: This bit enables the occurrence of an interrupt for a data transfer request (Channel0).
29	DMAE1	R/W	0x0	**DMA1 Enable** 0: Disable 1: This bit enables outputting the DMA transfer request signal for a data transfer request (Channel1).
28	DMAE0	R/W	0x0	**DMA0 Enable** 0: Disable 1: This bit enables outputting the DMA transfer request signal for a data transfer request (Channel0).
27-26	DMASL	R/W	0x2	**DMA Select** If the number of data buffers to be used is 1 (DMACH =0), these bits select the data buffer where a data transfer request occurs. 00: General Data FIFO 01: PageBuffer 1X: MSHC-Data Register
25-24	GDSZ	R/W	0x0	**General Data FIFO DMA Slice Size** This bits specify the transfer data size of a single data transfer for [PageBuffer]. GDSZ / Slice Size / Capacity of General Data FIFO (16Byte, 32Byte, 64Byte, 128Byte) 00b / 4Byte / o / O / o / o 01b / 16Byte / x / O / o / o 10b / 32Byte / x / X / o / o 11b / 64Byte / x / x / x / o
23	DMACH	R/W	0x0	**DMA Channel Number** This bit specifies the number of data buffers to be used. 0 : 1 kind 1 : 2 kinds
22-20	PDSZ	R/W	0x0	**PageBuffer DMA Slice Size** This bits specify the transfer data size of a single data transfer for [PageBuffer]. 000b : 4Byte 001b : 16Byte 010b : 32Byte 011b : 64Byte 100b : 128Byte 101b : 256Byte 110b : Reserved 111b : Reserved
19	-	-	-	Reserved
18	INTE	R/W	0x0	**INT Enable** 0: Disable 1: This bit enables outputting an interrupt.
17	-	-	-	Reserved
16	SRST	R/W	0x0	**Soft Reset** By setting 1 to this bit, the I-CON block performs a synchronous reset. After the synchronous reset, the bit will be returned to 0.
15-0	-	-	-	Reserved

The Table 5.5 below shows how to specify a data buffer for DMA transfer. DMA requests for [MSHC-Data Register] are output by setting DMASL=2'b1x only.

Table 5.5 How to Specify a Data Buffer for DMA Transfers

DMACH	DMASL		Channel 0	Channel 1
	[1]	[0]		
0	0	0	[General Data FIFO]	Not Used
0	0	1	[PageBuffer]	Not Used
0	1	X	[MSHC-Data Register]	Not Used
1	X	X	[General Data FIFO]	[PageBuffer]

Flag Register 0xB0840008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MIEF	1	0				DMAF1	DMAF0	FLG	HLTF	ITOF	STPF	EXTS			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-															

All bits of this register are read-only.

Field	Name	RW	Reset	Description
31	MIEF	R	0x0	**MSINT Error Flag** This bit becomes 1 if an error is detected while checking an INT status with the Instruction 2 to send/receive through [PageBuffer] in Section 8.3.3 TPC Instructions. It is judged that an error has occurred in case of any other interrupt cause than BREQ. It will be cleared to 0 when the START bit of [Control Register] changes from 0 to 1.
30-26	-	-	-	Reserved
25	DMAF1	R	0x0	**DMA1 Request Flag** This bit remains 1 while there is a request through “Data Transfer Request Channel 1.” When DMAE1 of [System Register] is 0, it will be cleared to 0 as soon as data of the slice size has been read/written. However, if there happens a fraction of data smaller than the slice size, this bit will not be cleared to 0 until the fraction is completely read/written. While DMAE1 bit of [System Register] is high, DMAF1 will be cleared, if acknowledge input to DMA channel 1 is set.
24	DMAF0	R	0x0	**DMA0 Request Flag** This bit remains 1 while there is a request through “Data Transfer Request Channel 0.” When DMAE0 of [System Register] is 0, it will be cleared to 0 as soon as data of the slice size has been read/written. However, if there happens a fraction of data smaller than the slice size, this bit will not be cleared to 0 until the fraction is completely read/written. While DMAE0 bit of [System Register] is high, DMAF0 will be cleared, if acknowledge input to DMA channel 0 is set.
23	FLG	R	0x0	**FLAG** This flag is used during a self-run. When a Compare instruction is executed, this bit answers 1 for true and 0 for false. On the other hand, when a TPC instruction is executed this bit becomes 1 when a communication error with memory stick occurs. It will be cleared to 0 when the START bit of [Control Register] changes from 0 to 1.
22	HLTF	R	0x0	**Halt Flag** This bit becomes 1 when a Halt instruction is completely processed. It will be cleared to 0 when the START bit of [Control Register] changes from 0 to 1.
21	ITOF	R	0x0	**INT Timeout Flag** This bit becomes 1 when a timeout occurs during a Wait instruction or a TPC instruction. It will be cleared to 0 when the START bit of [Control Register] changes from 0 to 1.
20	STPF	R	0x0	**Stop Flag** While an instruction is being processed (the START bit of [Control Register] is 1), if the process is attempted to stop (0 is set to the START bit), this STPF bit becomes 1 when the stop process is completed. This bit is cleared to 0 when the START bit of [Control Register] changes from 0 to 1.
19-16	EXTS	R	0x0	**Exit Status** When a Halt instruction is completed, the exit status will be reflected on these bits. They will be cleared to 0 when the START bit of [Control Register] changes from 0 to 1.
15-0	-	-	-	Reserved

Memory Control Register 0xB084000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GFVD								0		PBBC		PBCR	GFCR	PBFUL	PBEMP
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	1	1	1	GFDN											

Field	Name	RW	Reset	Description
31-24	GFVD	R	0x0	**General Data FIFO Valid Data Count (Read-Only)** These bits specify the number of valid data in [General Data FIFO]. (unit : byte)
23-22	-	-	-	Reserved
21-20	PBBC	R/W	0x0	**PageBuffer Bank Change (R/W)** These bits change a bank size of [PageBuffer] 00b : 256Byte 01b : Reserved 10b : 1KByte 11b : Reserved
19	PBCR	R/W	0x0	**PageBuffer Clear (R/W)** [PageBuffer] can be initialized by setting 1 to this bit. It automatically returns to 0 after having initialized [PageBuffer].
18	GFCR	R/W	0x0	**General Data FIFO Clear (R/W)** [General Data FIFO] can be initialized by setting 1 to this bit. It automatically returns to 0 after having initialized [General Data FIFO].
17	PBFUL	R/O	0x0	**PageBuffer Full (Read-Only)** If [PageBuffer] is a double-buffer structure, this bit becomes 1 when the data size in one of the buffers accessed from a CPU is the same as the PBBC size. When PBFUL=1, writing a value to [PageBuffer] is prohibited.
16	PBEMP	R	0x1	**PageBuffer Empty (Read-Only)** If [PageBuffer] is a double-buffer structure, this bit becomes 1 when the data size in one of the buffers accessed from a CPU is 0. When PBEMP=1, reading a value from [PageBuffer] is prohibited.
15-12	-	-	-	Reserved
11-0	GFDN	R/W	0x00	**General Data FIFO DMA Transfer Data Number (R/W)** These bits specify the number of accesses when writing data from a host CPU or a DMA controller to [General Data FIFO] during a self-run. (GFDN = Total data / 4Byte) Even when GDIR = 1, if there is no data written from the host CPU or the DMA controller to [General Data FIFO] during a self-run, “0” shall be set to these bits. When GDIR = 0 and a self-run is underway, “0” shall be set to these bits.

General Register 0/1 0xB0840010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
General Register 0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
General Register 1															

Field	Name	RW	Reset	Description
31-16	General Register 0	R/W	0x8000	**General Register** Guaranteed Operation of General Register [General Register] is used to mainly perform such functions as follows. -To store immediate values of a Load instruction -To store an expected value for comparing with an immediate value on a Compare instruction -To be used as a loop counter, in which a value is pre-incremented/pre-decremented, on a Compare instruction -To set the total number of pages written into [PageBuffer] on a self-run -To store the register values read/written from/to registers of the smshc If a value has been written to this register through a CPU during a self-run (when START=1), no proper operation is guaranteed.
15-0	General Register 1	R/W	0x9000	

General Register 2/3 — 0xB0840014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
General Register 2															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
General Register 3															

Field	Name	RW	Reset	Description
31-16	General Register 2	R/W	0xA000	**General Register** Guaranteed Operation of General Register [General Register] is used to mainly perform such functions as follows. -To store immediate values of a Load instruction -To store an expected value for comparing with an immediate value on a Compare instruction -To be used as a loop counter, in which a value is pre-incremented/pre-decremented, on a Compare instruction -To set the total number of pages written into [PageBuffer] on a self-run -To store the register values read/written from/to registers of the smshc If a value has been written to this register through a CPU during a self-run (when START=1), no proper operation is guaranteed.
15-0	General Register 3	R/W	0xB000	

General Register 4/5 — 0xB0840018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
General Register 4															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
General Register 5															

Field	Name	RW	Reset	Description
31-16	General Register 4	R/W	0xC0000	**General Register** Guaranteed Operation of General Register [General Register] is used to mainly perform such functions as follows. -To store immediate values of a Load instruction -To store an expected value for comparing with an immediate value on a Compare instruction -To be used as a loop counter, in which a value is pre-incremented/pre-decremented, on a Compare instruction -To set the total number of pages written into [PageBuffer] on a self-run -To store the register values read/written from/to registers of the smshc If a value has been written to this register through a CPU during a self-run (when START=1), no proper operation is guaranteed.
15-0	General Register 5	R/W	0xD0000	

Timer Register **0xB084001C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1	1	1	TIMER												
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-															

Field	Name	RW	Reset	Description
31-29	-	-	-	Reserved
28-16	TIMER	R	0x000	**TIMER (Read-Only)** These bits can work by using a Wait instruction or setting a timeout with a TPC instruction. The default value will be kept in these bits until another value is set by microcode. After microcode is completed, they will remain as 0. If these bits indicate any other value than 0, it means that it is counting. If a value is set here by microcode, it will start to count down toward 0. When it reaches 0, an event will occur. If waiting for an INT from memory stick through a Wait instruction, the ITOF bit of [Flag Register] becomes 1 as soon as the counted value of these bits reaches 0. Even before the counted value reaches 0, it becomes 0 when an INT occurs from memory stick. In the case of a simple Wait, it stops counting when it reaches 0, and moves on to a next instruction. Table 5.6 shows the values set to TimeCount and those set to the actual TIMER bit through a TPC instruction or a Wait instruction. No timeout will be set if 0000b has been set to TimeCount through a TPC instruction.
15-0	-	-	-	Reserved

Table 5.6 The Values Set to TimeCount and TIMER

TimeCount (in microcode)	Waiting Time (ms) (TIMER/Cycle)	TimeCount (in microcode)	Waiting Time (ms) (TIMER/Cycle)
0000b	0	1000b	1500/Cycle
0001b	2/Cycle	1001b	2000/Cycle
0010b	8/Cycle	1010b	2250/Cycle
0011b	10/Cycle	1011b	3000/Cycle
0100b	15/Cycle	1100b	4500/Cycle
0101b	20/Cycle	1101b	5250/Cycle
0110b	150/Cycle	1110b	6000/Cycle
0111b	200/Cycle	1111b	7000/Cycle

The Waiting Times shown below could be shorter by 1ms at most depending on timing when DVEN rises.
Cycle: DVEN cycle [kHz]

e.g.
In the case that 1 kHz clock has been input to DVEN and the value set in TimeCount is 0010b:
Waiting Time = 8 ÷ 1 = 8 msec

Instruction Queue 0xB0840020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INST0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INST1															

Field	Name	RW	Reset	Description
31-16	INST0	R/W	0xXXXX	**Instruction Queue** This is a buffer for writing instructions in the Instruction Queue, which can store up to 256 words. It is capable of writing instructions of 256 words. Although the length of an instruction word is 16 bits, it is possible to transfer data from a host both by every 32 and 16 bits. If [Program Counter Register] holds an even-number address, this buffer is accessible both by every 32 and 16 bits. On the other hand, if [Program Counter Register] holds an odd-number address, this buffer is accessible only by every 16bits. Writing/executing an instruction is always performed to the address indicated by the PRGC bits of [Program Counter Register]. Therefore, an appropriate address shall be set in PRGC bits when an instruction is written/executed. When this register is read, the instruction written in the address pointed by PRGC bits will be simultaneously read. - This register shall be read while the START bit of [Control Register] remains 0. - Reading this register is prohibited while START bit is 1.
15-0	INST1	R/W	0xXXXX	

General Data FIFO 0xB0840024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GDFIFO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GDFIFO															

Field	Name	RW	Reset	Description
31-0	GDFIFO	R/W	0x00000000	**General Data FIFO** This is a FIFO for sending/receiving general data. Selectable capacities include 16, 32, 64, and 128 bytes. One arbitrary size among them shall be chosen when implemented to an LSI. - While a self-run is being halted, the access size from a CPU is variable among 1, 2, or 4 bytes depending on the HSIZE terminal. - During a self-run, the access size from a CPU or a DMA controller is fixed to 4 bytes. - The data size ranges from 1 to 2048 bytes (selectable by every byte) to be sent/received by a single TPC through [General Data FIFO]. - The data size ranges from 1 to 16,380 bytes to be sent by a single self-run (due to the limitation of GFDN bit width. - The data size is limitless to be received by a single Self-run.

PageBuffer **0xB0840028**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PBUF															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PBUF															

Field	Name	RW	Reset	Description
31-0	PBUF	R/W	0xXXXXXXXX	**PageBuffer** This is a buffer for sending/receiving page data which capacity is 2048 bytes. - The access size from a CPU or a DMA controller is fixed to 4 bytes. - The maximum data size available to be sent/received by a single TPC through [PageBuffer] is 2048 bytes. - The data size available to be sent/received by a single TPC through [PageBuffer] is selectable from 256, 512, 768, 1024,1280, 1536, 1792, or 2048 bytes. - A bank size, which can divide one of the transfer data sizes, should be set to the PBBC bit of [Memory Control Register]. - PageBuffer cannot be cleared during a self-run.

Table 5.7 Conversion of a [PageBuffer] Bank

PageBuffer Capacity	Conversion of a [PageBuffer] bank (PBBC of [Memory Control Register]	
2048 KByte	How a [PageBuffer] bank is converted varies depending on each TPC.	
	TPC for a 512-byte transfer	: The bank conversion is 256 bytes.
	TPC for a 2048-byte transfer	: The bank conversion is 1024 bytes.

Version Register **0xB084002C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VER3				VER2				VER1				VER0			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Field	Name	RW	Reset	Description
31-16	VER3-VER0	R	0x0000	**Version (Read-Only)** The version of the smshc_i is represented to the bits from VER3 to VER0 by default. How a version number is represented is as follows. e.g. Version 2 5 2 c ver3 ver2 ver1 ver0 VER3: 2h VER2: 5h VER1: 2h VER0: Ch (When the value is 0h, the version number does not have a, b, c ...)

MSHC-Command Register 0xB0840030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TPC				DSL	DSZ										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Field	Name	RW	Reset	Description
31-28	TPC	R/W	0x0	**Transfer Protocol Code** These bits specify a TPC (4 bits).
27	DSL	R/W	0x0	**Data Select** This bit selects a send/receive buffer used for communicating with memory stick. 0: Sending/receiving data to/from memory stick through a FIFO of the smshc block 1: Sending/receiving data to/from memory stick through [PageBuffer] or [General Data FIFO]
26-16	DSZ	R/W	0x000	**Data Size** These bits specify the length of data to be sent/received. The selectable length ranges from 1 to 2048 bytes. However, it is fixed to 2048 bytes when DSZ=0.
15-0	-	-	-	Reserved

Table 5.8 Transfer Protocol Code

TPC	Transfer Protocol Code	
0x2	MS_RD_LDATA	Read Long Data Command
0x3	MS_RD_SDATA	Read Short Data Command
0x4	MS_RD_REG	Read Register Command
0x7	MS_GET_INT	Get Interrupt Command
0xD	MS_WR_LDATA	Write Long Data Command
0xC	MS_WR_SDATA	Write Short Data Command
0xB	MS_WR_REG	Write Register Command
0x8	MS_SET_RW_REG_ADRS	Set R/W Register Address Command.
0xE	MS_SET_CMD	Set Command
0x9	MS_EX_SET_CMD	Exit Set Command

A communication to memory stick can start by writing a value to this register.

Once the communication with memory stick has started, the RDY bit of [MSHC-Status Register] becomes 0. While the communication is underway, the RDY bit of [MSHC-Status Register] remains 0. The same bit will become 1 when the communication is completed.

Writing a value to this register is prohibited during a communication with memory stick (while the RDY bit of [MSHC-Status Register] is 0).

MSHC-Data Register 0xB0840034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSDATA															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSDATA															

Field	Name	RW	Reset	Description
31-0	MSDATA	R/W	0x00000000	**MSHC-Data** This bit is the access register to a FIFO of the smshc block which can be initialized by setting 1 to the FCLR bit of [MSHC-FIFO Control Register]. The minimum unit size is 8 bytes to access the data in a FIFO of the smshc block. In the case that the data is smaller than 8 bytes, the lower bytes of the data will be invalid. When the internal FIFO is used to transfer data, the unit size is 8 bytes. In the case the actual data is smaller than 8 bytes (GET_INT and SET_RW_REG_ADRS) or indivisible by 8, the data size shall be made a multiple number of 8. 8-bit and 16-bit accesses are prohibited.

MSHC-Status Register 0xB0840038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	DRQ	MSINT	RDY	0	0	CRC	TOE	CED	ERR	BRQ	MSINT7	MSINT6	MSINT5	MSINT4	CNK
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-															

Field	Name	RW	Reset	Description
31	-	-	-	Reserved
30	DRQ	R	0x0	**DMA Request** This bit will be set when a data transfer request occurs. This bit can be reset by accessing [MSHC-Data Register]. 1: A data transfer request exists. 0: No data transfer request exists.
29	MSINT	R	0x0	**MS I/F Interrupt** This bit is set 1 by receiving an interrupt request INT from memory stick. When the SRAC bit of [MSHC-System Register] is 0, the causes of an interrupt request INT from the memory stick will be set to such bits of this register as CED, ERR, BRQ, CNK, MSINT7, MSINT6, MSINT5, and MSINT4. The MSINT bit will be reset by writing a value to [MSHC-Command Register]. 1: An interrupt request INT has been received from memory stick. 0: No interrupt request INT has been received from memory stick yet.
28	RDY	R	0x1	**Ready** This bit is set 1 when a communication with memory stick is completed. If any error occurs while a protocol is being processed, the error status will be reflected on such bits as CRC and TOE. This bit will be reset by writing any value to [MSHC-Command Register]. 1: Ready to receive a command, or a protocol completed. 0: Not ready to receive a command because a communication with memory stick is underway.
27-26	-	-	-	Reserved
25	CRC	R	0x0	**CRC Error** If a CRC error occurs, 1 is set to this bit when the protocol is completed. This bit will be cleared by writing any value to [MSHC-Command Register]. 1: A CRC error occurs when receiving data. 0: Normal end
24	TOE	R	0x0	**Timeout Error** When a busy state continues longer than the number of clocks set in the BSY bits of [MSHC-System Register], it is considered that some malfunction has happened to memory stick, and the protocol will be consequently terminated. Then, 1 is set to this bit. This bit will be cleared by writing any value to [MSHC-Command Register]. 1: A RDY timeout error occurs at a handshake period in a communication with memory stick. 0: Normal end
23	CED	R	0x0	**MS Command End** This bit is effective only at the 4-bit and 8-bit parallel interface mode. When a value is set to MSINT bit, the CED bit of the INT register on memory stick will be simultaneously reflected on this bit. This bit will be cleared by writing any value to [MSHC-Command Register].
22	ERR	R	0x0	**MS Error** This bit is effective only at the 4-bit and 8-bit parallel interface mode. When a value is set to MSINT bit, the ERR bit of the INT register on memory stick will be simultaneously reflected on this bit. This bit will be cleared by writing any value to [MSHC-Command Register].
21	BRQ	R	0x0	**MS Data Buffer Request** This bit is effective only at the 4-bit and 8-bit parallel interface mode. When a value is set to MSINT bit, the BREQ bit of the INT register on memory stick will be simultaneously reflected on this bit. This bit will be cleared by writing any value to [MSHC-Command Register].
20-17	MSINT7 to 4	R	0x0	**MS Interrupt 7 to 4** This bits are effective only at the 8-bit parallel interface mode.

				These bits can be expanded to indicate the causes of an interrupt from memory stick. When a value is set to MSINT bit, the causes of an interrupt status indicated by the bits from DATA7 to DATA4 on memory stick will be simultaneously reflected on these bits. These bits will be cleared by writing any value to [MSHC-Command Register].
16	CNK	R	0x0	**MS Command No Acknowledge** This bit is effective only at the 4-bit and 8-bit parallel interface mode. When a value is set to MSINT bit, the CMDNK bit of the INT register on memory stick will be simultaneously reflected on this bit. This bit will be cleared by writing any value to [MSHC-Command Register].
15-0	-	-	-	Reserved

MSHC-System Register **0xB084003C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RST	INTEN	MSIEN	DRQSL	INTCLR	0			SRAC	EIGHT	MSPIO1	MSPIO0	NOCRC	BSY		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-															

Field	Name	RW	Reset	Description
31	RST	R/W	0x0	**Synchronous Reset** When 1 is written to this bit, the smshc block will be rest. It will be cleared to 0 after the internal resetting is completed.
30	INTEN	R/W	0x0	**XINT Enable** 1: Enables the smshc block to output an interrupt request. 0: Disables the smshc block to output an interrupt request.
29	MSIEN	R/W	0x1	**MS Interrupt Enable** 1: Enables receiving an interrupt request from memory stick. 0: Disables receiving an interrupt request from memory stick.
28	DRQSL	R/W	0x0	**DRQ Select** 1: Enables the outputting of XINT when the smshc block requests data. (DRQ=1) 0: Disables the outputting of XINT when the smshc block requests data. (DRQ=1)
27	INTCLR	R/W	0x0	**INT Clear** An interrupt from the smshc block is cleared by setting 1 to this bit. This bit is cleared to 0 after the interrupt has been cleared.
26-24	-	-	-	Reserved
23	SRAC	R/W	0x1	**Serial Access Mode** 1: Serial interface mode 0: Parallel interface mode
22	EIGHT	R/W	0x0	**Eight Line Parallel Access Mode** 0: 4-bit parallel interface mode 1: 8-bit parallel interface mode
21	MSPIO1	R/W	0x1	**Memory Stick Parallel I/O No.1** The value of MSPIO1 is output from an external terminal, mspio[1]. When the MSPIO1 bit of [MSHC-System Register] is updated, the MSPIO1 bit of [MSHC-User Custom Register] is also updated. 0: Outputs Low from mspio[1] (Default) 1: Outputs High from mspio[1]
20	MSPIO0	R/W	0x0	**Memory Stick Parallel I/O No.0** The value of MSPIO0 is output from an external terminal, mspio[0]. When the MSPIO0 bit of [MSHC-System Register] is updated, the MSPIO0 bit of [MSHC-User Custom Register] is also updated. 0: Outputs Low from mspio[0] (Default) 1: Outputs High from mspio[0]
19	NOCRC	R/W	0x0	**No CRC** When 1 is set to this bit, a write protocol is executed without any CRC data (16 bits) attached to the end of data to be sent. For a read protocol, checking a CRC is performed as usual no matter what the value set to this bit is. This bit shall be 0 unless otherwise specified. 1: Outputting a CRC off

				0: Outputting a CRC on
18-16	BSY	R/W	0x5	**Busy Count** These bits set a RDY timeout period. The maximum BSY period for waiting an RDY signal from memory stick can be calculated by multiplying the value set in this bit x 4 SCLKs. When BSY=0, no timeout is detected. To wake up a Memory Stick Ver. 1.X, this bit shall be set 0 before starting a protocol. This is because the RDY timeout is usually longer when waking up a Memory Stick Ver. 1.X. While a protocol is being executed, the value in this bit shall not change.
15-0	-	-	-	Reserved

Table 5.9 Interface Mode Configuration

SRAC	EIGHT	Interface Mode
0	0	4-bit parallel interface mode
0	1	8-bit parallel interface mode
1	0	Serial interface mode
1	1	Prohibited to be set

MSHC-User Custom Register **0xB0840040**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSPIO[7:2]						MSPIO1	MSPIO0	0		EMP	FUL	0			NDLT
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-															

Field	Name	RW	Reset	Description
31-26	MSPIO[7:2]	R	0x00	**Memory Stick Parallel I/O** The value of MSPIO[7:2] is output from an external terminal, mspio[7:2].
25	MSPIO1	R	0x1	**Memory Stick Parallel I/O No.1** The value of MSPIO1 is output from an external terminal, mspio[1]. When the MSPIO1 bit of [MSHC-User Custom Register] is updated, the MSPIO1 bit of [MSHC-System Register] is also updated. 0: Outputs Low from mspio[1] 1: Outputs High from mspio[1] (Default)
24	MSPIO0	R	0x0	**Memory Stick Parallel I/O No.0** The value of MSPIO0 is output from an external terminal, mspio[0]. When the MSPIO0 bit of [MSHC-User Custom Register] is updated, the MSPIO0 bit of [MSHC-System Register] is also updated. 0: Outputs Low from mspio[0] (Default) 1: Outputs High from mspio[0]
23-22	-	-	-	Reserved
21	EMP	R	0x1	**FIFO Empty** When 1 is set to the FCLR bit of [MSHC-FIFO Control Register], this bit is also set 1. 1: FIFO of the smshc block is empty. 0: Data exists in FIFO of the smshc block.
20	FUL	R	0x0	**FIFO Full** When 1 is set to the FCLR bit of [MSHC-FIFO Control Register], this bit is reset to 0. 1: FIFO of the smshc block is full. 0: FIFO of the smshc block is not full.
19-17	-	-	-	Reserved
16	NDLT	R	0x0	**Next Data Latch Timing** Default = 0 (normal data latch) 0: Latches data at a rising edge right after the data input timing (Default) 1: Latches data at a rising edge two cycles after the data input timing
15-0	-	-	-	Reserved

MSHC-FIFO Control Register 0xB0840044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					AFCLR	FCLR	FDIR	0						DRSZ	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-															

Field	Name	RW	Reset	Description
31-27	-	-	-	Reserved
26	AFCLR	R/W	0x0	**Auto FIFO Clear** The FIFO data of the smshc block will be automatically initialized by setting 1 to this bit. When both a protocol and a data transfer are completed, this bit will be initialized.
25	FCLR	R/W	0x0	**FIFO Clear** The FIFO data in the smshc block will be initialized by setting 1 to this bit. It will be cleared to 0 when initializing the FIFO data is completed.
24	FDIR	R/W	0x0	**FIFO Direction** When a protocol starts, the value of TPC[3] of [MSHC-Command Register] will be reflected on this bit. 1: The direction of FIFO in the smshc block shall be the direction for sending data. (CPU → FIFO of the smshc block → memory stick) 0: The direction of FIFO in the smshc block shall be the direction for receiving data. (CPU ← FIFO of the smshc block ← memory stick)
23-18	-	-	-	Reserved
17-16	DRSZ	R/W	0x1	**Data Request Size** These bits set a slice size for a data transfer of FIFO in the smshc block.
15-0	-	-	-	Reserved

Table 5.10 DMA Slice Size Configuration

DRSZ[1:0]	DMA Slice Size
00b	4Byte
01b	8Byte
10b	16Byte
11b	Prohibited to be set

MSHC-DMA Control Register 0xB084004C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMAON															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-															

Field	Name	RW	Reset	Description
31-16	DMAON	R/W	0x0001	**DMA Request Enable** These bits enable outputting a DMA transfer request signal depending on each value set in the TPC bit of [MSHC-Command Register]. 0: When a data transfer request occurs, xdrq is not asserted. 1: When a data transfer request occurs, xdrq is asserted. e.g. DMAON[15]=1: DMA transfer request signal will be valid when issuing TPC = 1111b. DMAON[13]=1: DMA transfer request signal will be valid when issuing TPC = 1101b (WRITE_LONG_DATA). DMAON[7]=0: DMA transfer request signal will be invalid when issuing TPC = 0111b (GET_INT).
15-0	-	-	-	Reserved

PORTCFG Register **0xB0870100**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	BM			DVEN	-										
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-													PORT		

Field	Name	RW	Reset	Description
31	-	-	-	Reserved
30	BM	R/W	0x0	**Byte Order Mode Selection** Select Byte order 0:LSB(Least Significant Byte) first mode (default) 1:MSB(Most Significant Byte) first mode
29-28	-	-	-	Reserved
27	DVEN	R/W	0x0	**DVEN** Timer count enable signal. (1KHz to 2KHz) The maximum frequency will be half of HCLK frequency. For more about the operation of this bit, refer to the description of [Timer Register].
26-3	-	-	-	Reserved
2-0	PORT	R/W	0x0	**Port Number** It specifies the port number to be used by the memory stick host controller. (0~5)

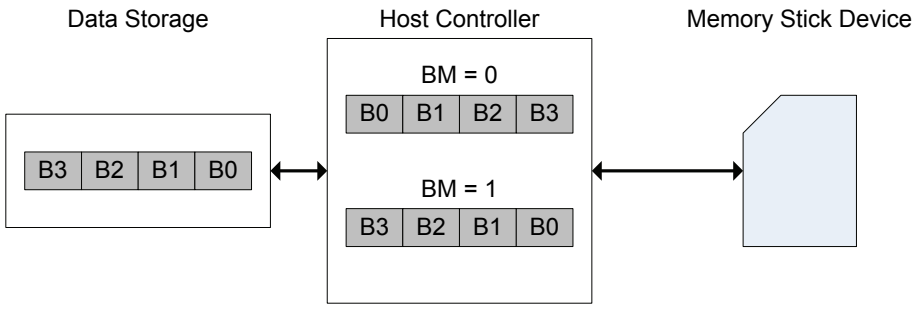


Figure 5.3 Memory Stick Byte Order Mode

PORTDLY Register 0xB0870104

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-		DLY_OEN			DLY_BS			DLY_DATA7			DLY_DATA6			DLY_DATA5	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DLY_DATA5		DLY_DATA4		DLY_DATA3			DLY_DATA2			DLY_DATA1			DLY_DATA0		

Field	Name	RW	Reset	Description
31-30	-	-	-	Reserved
29-27	DLU_OEN	R/W	0x0	**OEN Output Delay Configuration** It controls output delays of all Memory Stick output signal. 3 bits are assigned and it ranges from 0 to 7. 0: no delay 1~7: about max. 500ps* DLY_OEN
27-24	DLY_BS	R/W	0x0	**BS Output Delay Configuration** It controls output delays of each Memory Stick BS signal. 3 bits are assigned and it ranges from 0 to 7. 0: no delay 1~7: about max. 500ps* DLY_BSn
23-0	DLY_DATA 0 - DLY_DATA 7	R/W	0x0	**DATA[7:0] Output Delay Configuration** It controls output delays of each Memory Stick DATA signal. 3 bits are assigned and it ranges from 0 to 7. 0: no delay 1~7: about max. 500ps* DLY_DATAn

5.3 Operation

5.3.1 Operation during Reset

When 1 is set to the SRST bit of [System Register], the registers and operational sequences inside the I-CON block will be initialized. After this synchronous reset is completed, the SRST bit will be automatically cleared to 0.When 1 is set to the RST bit of [MSHC-System Register], the registers and operational sequences inside the smshc block will be initialized. After this synchronous reset is completed, the RST bit will be automatically cleared to 0.
If a reset is performed during a communication with memory stick, the bus state of host controller might disagree to that of the memory stick. In addition, when such a reset occurs in the middle of a communication with memory stick, the power supply of the memory stick shall be also reset.

5.3.2 Operation Flow

The smshc_i can achieve a communication with memory stick by starting up the smshc and issuing a TPC. There are two methods to start up the smshc; one is through a host CPU while the other one is through the I-CON.

5.3.3 Communication Method to Start up smshc through Host CPU

A TPC and the data size to be sent are written to [MSHC-Command Register] by a CPU. This is how the smshc can start communicating with memory stick (issuing a TPC).

The basic flow is as follows:

Writes values to [MSHC-Command Register]
(1) Accesses [MSHC-Data Register] (Write / Read)
(2) Completes a TPC ([MSHC-Status Register] RDY = 1)

Described below are the basic explanation of write TPCs and read TPCs.

Write TPC

In the case that a write TPC is issued, the smshc communicates with memory stick if a TPC, the size of data to be transferred, and the data itself are set by a host CPU. The smshc block automatically reverses the TPC in BS1, adds CRC data in BS2, and detects a BSY/RDY signal in BS3. Transferring data can be achieved using [MSHC-Data Register] (FIFO) or using either [PageBuffer] or [General Data FIFO]. The selection of a method to write data can be set to the DSL bit of [MSHC-Command Register].
When a TPC is completed, the RDY bit of [MSHC-Status Register] will be set.

After having written data to [MSHC-Command Register], writing any value to the following registers is prohibited until the TPC is completed ([MSHC-Status Register] RDY = 0); [MSHC-Command Register], [MSHC-System Register], [MSHC-FIFO Control Register], and [MSHC-DMA Request Control Register]. If written, proper operation cannot be guaranteed.
To transfer data through [PageBuffer] or [General Data FIFO], microcode with a TPC instruction, which utilizes [PageBuffer] or [General Data FIFO] respectively, shall be executed. Neither [PageBuffer] nor [General Data FIFO] can be used without a self-run.

- When [MSHC-Data Register] is used, data of the FIFO size (32 bytes) can be stored in [MSHC-Data Register] in advance.
- If no transfer data can be prepared in time, SCLK (Memory Stick I/F clock) will stop to wait for the data to be transferred.
- If no RDY signal is returned from the memory stick resulting in a timeout during BS3, not only the TOE bit of [MSHC-Status Register] but also the RDY bit will be set. The duration of the RDY timeout can be set at the BSY bits of [MSHC-System Register].

Read TPC

In the case that a read TPC is issued, the smshc communicates with memory stick if a TPC and the size of data to be transferred are set by a host CPU. The smshc automatically reverses the TPC in BS1, detects a BSY/RDY signal in BS2, and checks CRC data in BS3. Reading data can be achieved using [MSHC-Data Register](FIFO) or using either [PageBuffer] or [General Data FIFO]. The selection of a method to read data can be set to the DSL bit of [MSHC-Command Register].
When a TPC is completed, the RDY bit of [MSHC- Status Register] will be set.

After having written a value to [MSHC-Command Register], writing any value to the following registers is prohibited until the TPC is completed ([MSHC-Status Register] RDY = 0); [MSHC-Command Register], [MSHC-System Register], [MSHC-FIFO Control Register], and [MSHC-DMA Request Control Register]. If written, proper operation cannot be guaranteed.
To transfer data through [PageBuffer] or [General Data FIFO], microcode with a TPC instruction, which utilizes [PageBuffer] or [General Data FIFO] respectively, shall be executed. Neither [PageBuffer] nor [General Data FIFO] can be used without a self-run.

- If no transfer data can be prepared in time, SCLK (Memory Stick I/F clock) will stop to wait for the data to be transferred.
- If no RDY signal is returned from the memory stick resulting in a timeout during BS2, not only the TOE bit of [MSHC-Status Register] but also the RDY bit will be set. The duration of a RDY timeout can be set at the BSY bits of [MSHC-System Register].
- During BS3, if a CRC calculation of data results in an inconsistency, not only the RDY bit of [MSHC-Status Register] but also the CRC bit will be set. This announces the lack of data reliability.

Shown below is an example of a communication with Memory Stick PRO. This example illustrates the flow of a READ_DATA command. A host CPU directly accesses the smshc registers (0x30 to 0x4C). This method makes it easier to issue TPCs, however, more workload will be imposed on the host CPU.

The ▭ portion shows a single cycle of TPC issuance or the process for an interrupt from Memory Stick PRO.

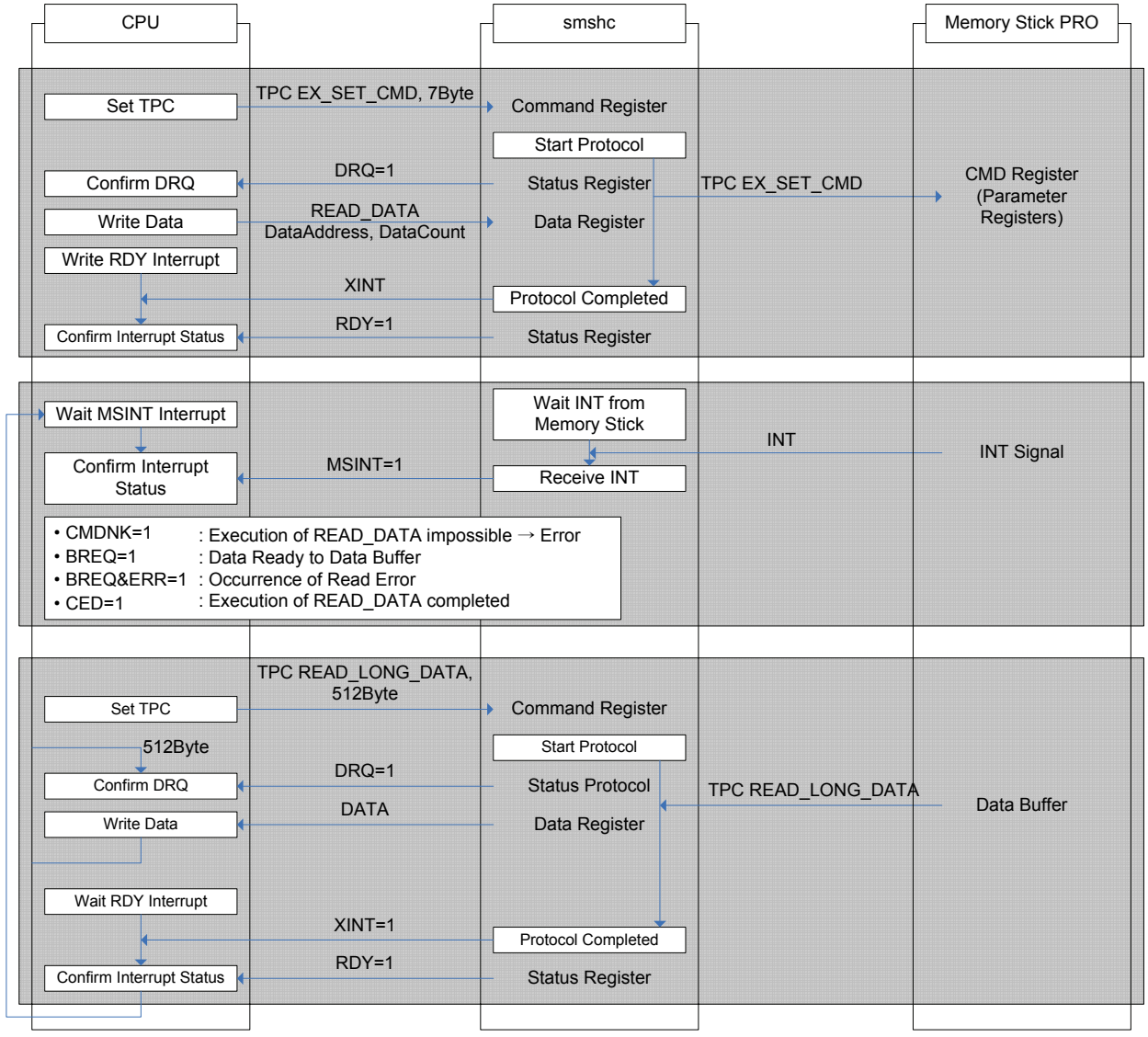


Figure 5.4 Communication Example for Starting up smshc through CPU

5.3.4 Communication Method to Start up smshc through I-CON

The I-CON accesses the smshc registers (0x30 to 0x4C). The details of instructions for starting up the smshc are all gathered into the I-CON as microcode. The I-CON can start up the smshc and handle interrupts on behalf of a host CPU. This method can decrease the workload imposed on the host CPU, however, microcode are required.
Shown below is an example of a communication with Memory Stick PRO. This example illustrates the flow of a READ_DATA command.

The ▭ portion shows a single cycle of TPC issuance or the process for an interrupt from Memory Stick PRO.

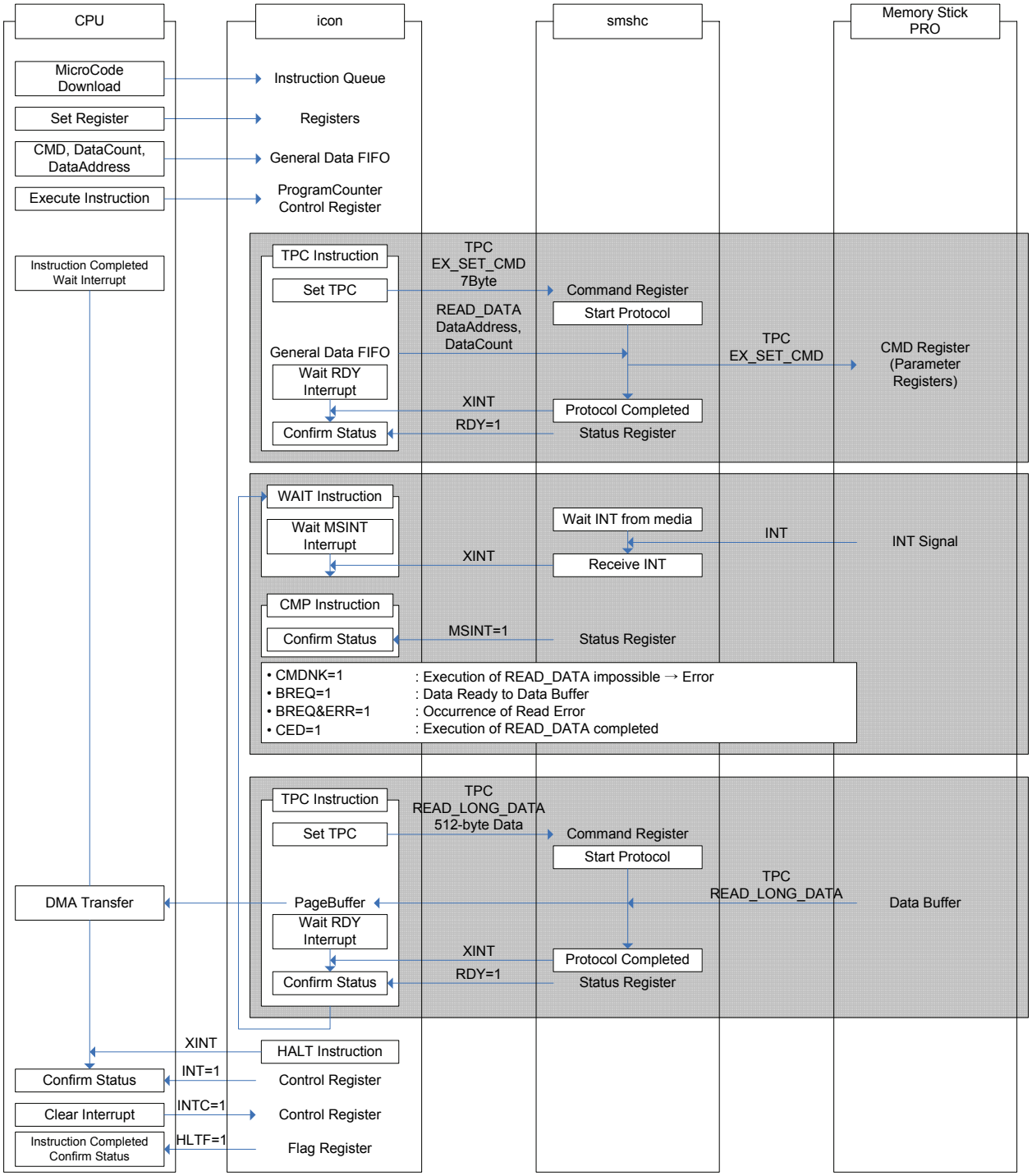


Figure 5.5 Communication Example for Starting up smshc through I-CON

5.3.5 Interface Mode Switching Sequence

The host controller supports three interface mode, serial, 4-bit parallel, 8-bit parallel. The interface mode on the host controller shall be switched after the interface mode on the memory stick has been successfully completed through a WRITE_REG TPC.(a RDY signal is received.) The sequence of switching interface mode is described in this section.

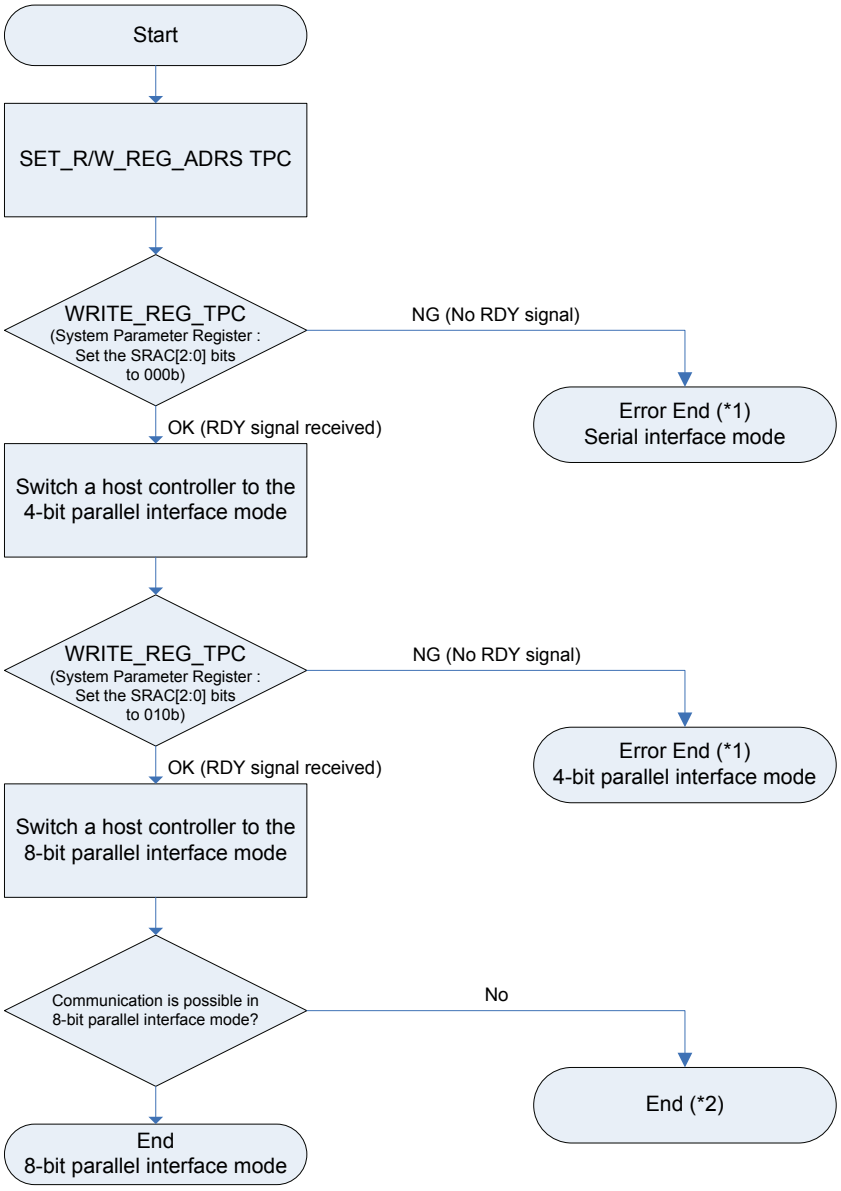


Figure 5.6 Sequence to Switch Interface Mode

Switching Interface Mode through CPU

This test accesses the smshc controller's registers, and issues TPCs after switching to a 4-bit parallel interface mode. Specifically, it issues SET_RW_REG_ADRS as a write TPC and GET_INT as a read TPC; checks for an interrupt after a TPC transfer is complete; clears an interrupt when it occurs; and checks RDY, TOE and CRC of the MSHC-Status Register. The setting flow (1)~(14) is for 4-bit interface mode and (15)~(26) is for checking the previous settings .

- Enable external interrupts to occur from smshc_i.

(1) Set the INTEN bit of the MSHC-System Register to enable external interrupts to occur from smshc_i.

- Switch to 4-bit parallel interface.

(2) In the MSHC-Command Register, set as follows: TPC=SET_RW_REG_ADRS and DSZ=0x4.
(3) Continue to read the DRQ bit of the MSHC-Status Register until it is set to 1.
(4) Write 8-byte data (0x0101_1001, 0x0000_0000) to the MSHC-Data Register.
(5) Wait for an XINT interrupt to occur.
(6) Read the MSHC-Status Register to make sure everything is completed successfully.
(7) Set the INTCLR bit of the MSHC-System Register to clear the XINT interrupt.
(8) In the MSHC-Command Register, set as follows: TPC=WRITE_REG and DSZ=0x1.
(9) Continue to read the DRQ bit of the MSHC-Status Register until it is set to 1.
(10) Write 8-byte data (0x0000_0000, 0x0000_0000) to the MSHC-Data Register.
(11) Wait for an XINT interrupt to occur.
(12) Read the MSHC-Status Register to make sure everything is completed successfully.
(13) Set the INTCLR bit of the MSHC-System Register to clear the XINT interrupt.
(14) In the MSHC-System Register, set as follows: SRAC=0.

Additional steps below are to be done in 8-bit parallel interface setting mode after 4-bit interface mode settings.

- Switch to 8-bit parallel interface.

(15) In the MSHC-Command Register, set as follows: TPC=WRITE_REG and DSZ=0x1.
(16) Continue to read the DRQ bit of the MSHC-Status Register until it is set to 1.
(17) Write 8-byte data (0x4000_0000, 0x0000_0000) to the MSHC-Data Register.
(18) Wait for an XINT interrupt to occur.
(19) Read the MSHC-Status Register to make sure everything is completed successfully.
(20) Set the INTCLR bit of the MSHC-System Register to clear the XINT interrupt.
(21) In the MSHC-System Register, set as follows: SRAC=0 and EIGHT=1.

Following steps (22)~(33) checks if mode switching is completed or not.

- Transfer a write TPC.

(22) In the MSHC-Command Register, set as follows: TPC=SET_RW_REG_ADRS and DSZ=0x4.
(23) Continue to read the DRQ bit of the MSHC-Status Register until it is set to 1.
(24) Write arbitrary 8-byte data to the MSHC-Data Register.
(25) Wait for an XINT interrupt to occur.
(26) Read the MSHC-Status Register to make sure everything is completed successfully.
(27) Set the INTCLR bit of the MSHC-System Register to clear the XINT interrupt.

- Transfer a read TPC.

(28) In the MSHC-Command Register, set as follows: TPC=GET_INT and DSZ=0x1.
(29) Continue to read the DRQ bit of the MSHC-Status Register until it is set to 1.
(30) Read the MSHC-Data Register. The expected value is as follows: 0x8000_0000, 0x0000_0000.
(31) Wait for an XINT interrupt to occur.
(32) Read the MSHC-Status Register to make sure everything is completed successfully.
(33) Set the INTCLR bit of the MSHC-System Register to clear the XINT interrupt.

Switching Interface Mode through I-CON - 8bit Parallel Interface Mode

To operate with I-CON, microcode must be prepared in [Instruction Queue]. If START bit of [Control Register] is set, the microcode is read and executed in the ordere saved. The detailed setting flow described in the table below.

Table 5.11 Microcode for Interface Mode Switching

Program Counter	Microcode Instruction	Description
0x0000	0x2C20	TPC instruction to send an immediate value; uses [Data Register] of smshc.
0x0001	0x8004	SET_RW_REG_ADRS | SIZE=0x4
0x0002	0x0101	RD_ADRS=0x1; RD_SIZE=0x1
0x0003	0x1001	WR_ADRS=0x10; WR_SIZE=0x1
0x0004	0x2720	Instruction to send a TPC to memory stick using [General Data FIFO]
0x0005	0xB001	WRITE_REG | SIZE=0x1
0x0006	0x60BC	Stores the value of [General Register0] to the [System Register] of smshc.
0x0007	0x0001	HALT instruction; EXIT Code=0x1

In switching interface mode through CPU, mode setting data, to be transferred to external memory stick, would be moved from common memory space with a help from DMA controller. The common memory space should be kept occupied previously. 4-bit parallel interface mode setting sequence is as follows.

- Enable external interrupts to occur from smshc_i.
- Write microcode instructions to the Instruction Queue. (refer to Table 5.11)
- Write in the common memory space the data to be sent by WRITE_REG to switch to a 4-bit parallel interface.
- Perform settings to start DMA0.

  (34) The slice size is 4 bytes, and therefore memory-to-peripheral transfer shall be performed once.
- Start smshc_i (to switch to 4-bit parallel interface).

  (35) Set the Memory Control Register as follows: GFDN=0x1.
  (36) Write 0x6005_0000 in the General Data Registers 0 to 1.
  (37) Write 0x0000_0000 in General Data Registers 2 to 3.
  (38) Set the System Register as follows: DMAE0=1, DMACH=0, DMASL = 00b, GDSZ=00b, INTE=1 and DMAE1=1.
  (39) Set Program Counter Register as follows: PRGC=0x00.
  (40) Set Control Register as follows: GDIR=1 and START=1.
  (41) Wait for an XINT interrupt to occur. DMA data transfer is being performed...
  (42) Check the Flag Register to make sure data is set as follows: HLTF=1, EXTS=0x1.
  (43) Set the Control Register as follows: INTC=1.

Additional steps below are to be done in 8-bit parallel interface setting mode after 4-bit interface mode settings.

- Write in the Shared SRAM the data to be sent by WRITE_REG to switch to an 8-bit parallel interface.
- Perform settings to start DMA0.

  (44) The slice size is 4 bytes, and therefore memory-to-peripheral transfer shall be performed once.
- Start smshc_i (to switch to 8-bit parallel interface).

  (45) Set the Memory Control Register as follows: GFDN=0x1 and GFCR=1.
  (46) Write 0x6045_0000 in General Data Registers 0 to 1.
  (47) Write 0x0000_0000 in General Data Registers 2 to 3.
  (48) Set the System Register as follows: DMAE0=1, DMACH=0, DMASL = 00b, GDSZ=00b, INTE=1 and DMAE1=1.
  (49) Set Program Counter Register as follows: PRGC=0x00.
  (50) Set the Control Register as follows: GDIR=1 and START=1.

(51) Wait for an XINT interrupt to occur. DMA data transfer is being performed.
(52) Check the Flag Register to make sure data is set as follows: HLTF=1, FLG=1 and EXTS=0x1.
(53) Set the Control Register as follows: INTC=1.

5.4 Timing Diagram

5.4.1 Serial Interface Mode

TPC Transfer State (BS1)

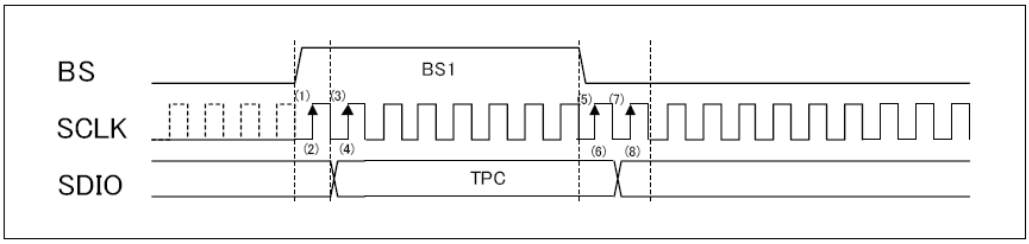


Figure 5.7 Serial Protocol TPC Transfer State (BS1)

Operation	Description
Host	The host controller outputs High at point (1) on BS above, and starts the output of SCLK before point (2). Starts output the MSB of a TPC on the SDIO at point (3). Switches the BS to Low at the same time the LSB of the TPC is output at point (5). BS2 starts from point (7).
Memory Stick	The Memory Stick detects BS as High at point (2) above, and SDIO is put in a High Impedance (input) state between point (2) and point (3). Receive the MSB of the TPC at the next point (4). Detects the BS as Low at the same time the LSB of the TPC is received at point (6) and starts BS2 from point (7).

Data Transfer State (Read Packet : BS3)

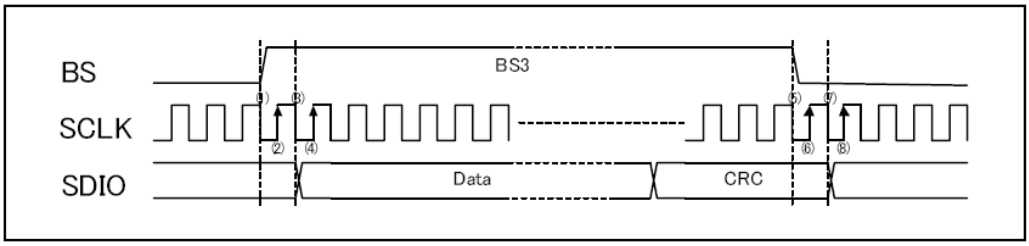


Figure 5.8 Serial Protocol Data Transfer State (BS3)

Operation	Description
Host	In a Read Packet, after switching the BS to High at point (1) above, the host controller receives the MSB of the first data at point (4). After switching the BS to Low at point (5), the LSB of the CRC is received at point (6) and the Read Packet completes.
Memory Stick	Detects the Bs as High at point (2) above, and starts outputting the MSB of the first data at point (3). After outputting the LSB of the CRC at point (5), the BS is detected as Low and then enters into BS0 from point (7).

Data Transfer State (Write Packet : BS2)

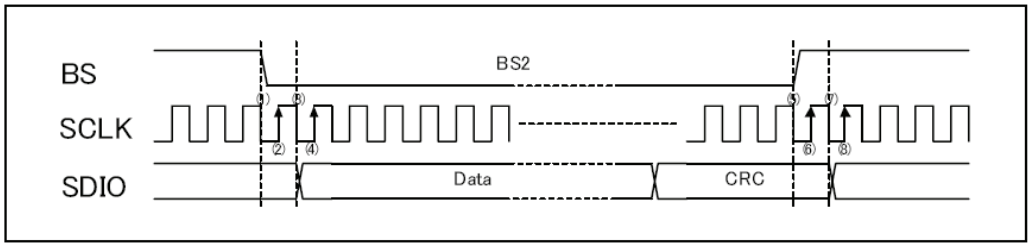


Figure 5.9 Serial Protocol Data Transfer State (BS2)

Operation	Description
Host	For a Write Packet, after switching the BS to Low at point (1) above, transferring the MSB of the first data is started at point (3) with a 1 SCLK delay. Transfers the 16-bit CRC for all the data immediately after the LSB of the last data. Switches the BS to High at the same time the LSB of the CRC is sent at point (5), and then starts detecting the RDY signal of BS3 from point (8).
Memory Stick	The Bs is detected as Low at point (2) above, and receiving of the signal will start with the MSB of the first data at point (4). The BS is detected as High simultaneously to receiving the LSB of the CRC at point (6) and then starts outputting the BSY/RDY signal of BS3 from point (7).

Handshake State (Read Packet : BS2 / Write Packet : BS3)

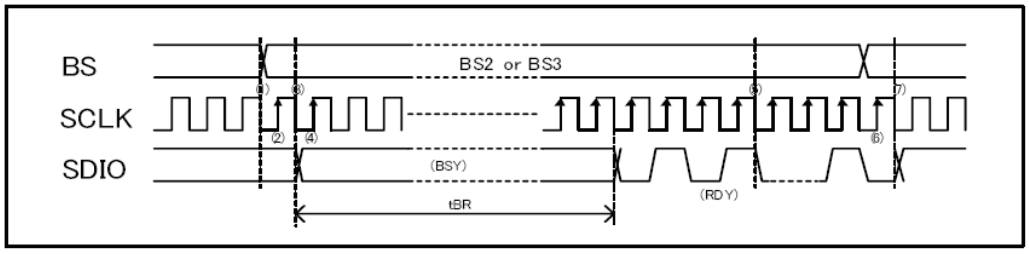


Figure 5.10 Serial Protocol Handshake State

Operation	Description
Host	After switching the BS at point (1) above, starts to detect the RDY signal from point (4). After receiving the RDY signal, which reversing for each SCLK, for more than or equal to 4 SCLKs, switches the BS signal at the falling edge of the SCLK after point (5) and the next Bus State begins with point (7).
Memory Stick	After detecting the switch of Bus State at point (2) above, starts to output of the BSY/RDY signal from point (3). After the Memory Stick has completed its internal processing and outputs the RDY signal, it detects the switch of Bs at point (6) and then the next BS starts from point (7).

INT Transfer State (BS0)

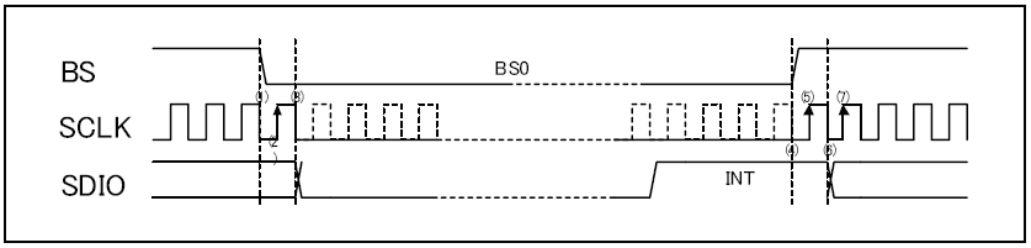


Figure 5.11 Serial Protocol INT Transfer State

Operation	Description
Host	SCLK can be halted after the point (3) above, if necessary. (SCLK is Low) SDIO shall be in a Hi-Z state (input) after point (3). If an INT signal (SDIO is High) if detected during the INT Transfer State period, the host controller shall confirm the cause of interrupt by read out INT Register of a Memory Stick.
Memory Stick	Detects BS as Low at point (2) above, and starts the INT Transfer State at point (3). SDIO shall be in the output status after point (3), and outputs Low until the INT signal is output. The cause of interrupt shall be reflected to the INT Register and the INT signal (SDIO is High) shall be output for the period of the INT Transfer State.

5.4.2 4-bit Parallel Interface Mode

TPC Transfer State (BS1)

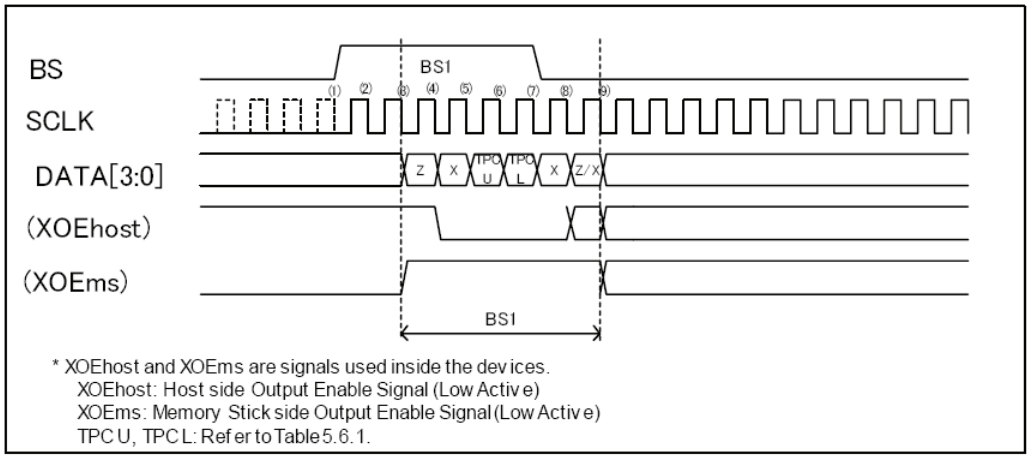


Figure 5.12 4-bit Parallel Protocol TPC Transfer State

Operation	Description
Host	It outputs the BS as High at the above point (1), and starts the provision of SCLK before point (2). It starts the output of the upper 4 bits of the TPC on DATA[3:0] at point (5). Switches the BS to Low at point (7) and then starts BS2 from point (9).
Memory Stick	It detects the BS as High at the above point (2), and receives the upper 4 bits of the TPC at point (6). Receives the lower 4 bits at point (7). The BS is detected as Low at point (8), and starts BS2 from point (9).

Data Transfer State (Read Packet : BS3)

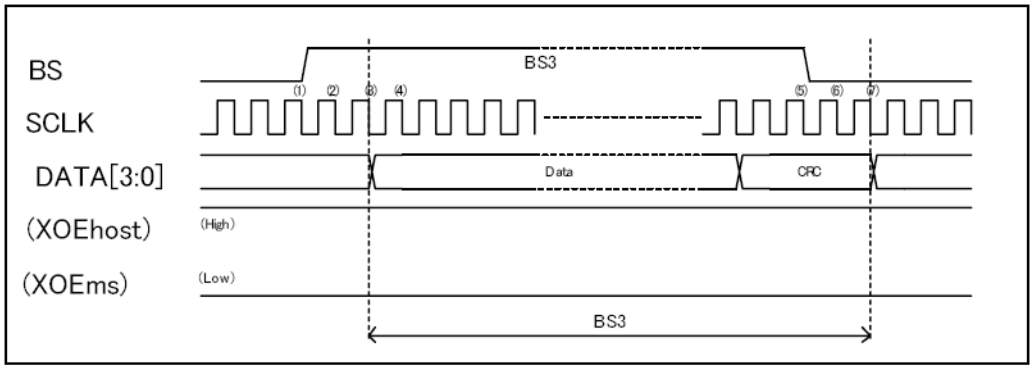


Figure 5.13 4-bit Parallel Protocol Data Transfer State (BS3)

Operation	Description
Host	For a Read Packet, after switching the BS to High at point (1) above, receiving the signal will start with the upper 4 bits of the first data at point (4), with a delay of 3 SCLKs. After switching the BS to Low at point (5), the lower 4 bits of the 16 bit CRC are received at point (7), and the Read Packet completes.
Memory Stick	The BS is detected as High at point (2) above, and the upper 4 bits of the first data are output at point (3). At point (6), the Bs is detected as Low and the lower 4 bits of the 16-bit CRC are output. The transition to BS0 is made at point (7).

Data Transfer State (Write Packet : BS2)

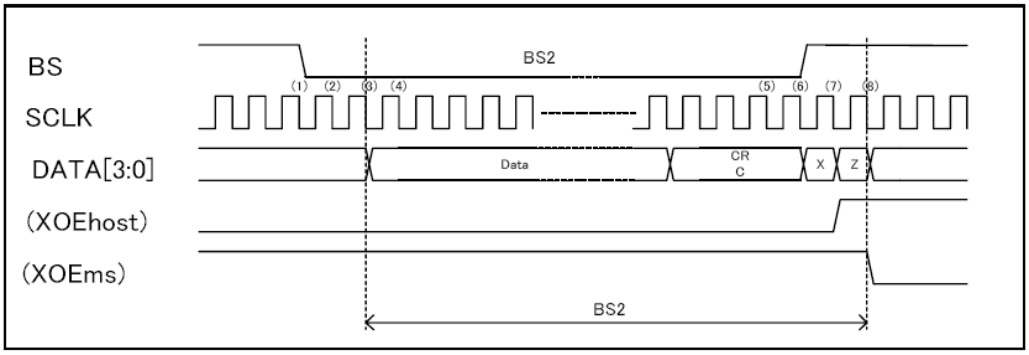


Figure 5.14 4-bit Parallel Protocol Data Transfer State (BS2)

Operation	Description
Host	For a Write Packet, after switching the BS to Low at point (1) above, starts to transfer the signal with the upper 4 bits of the first data at point (3), with a delay of 2 SCLKs. The 16-bit CRC for all of the data is transferred from the next SCLK after the last data. The BS is switched to High at point (6).
Memory Stick	The BS is detected as Low at point (2) above, and starts to receive the signal with the upper 4 bits of the first data at point (4). After receiving the lower 4 bits of the 16-bit CRC at point (6), the BS is detected as High at point (7), and the BSY signal or RDY signal of BS3(Handshake State) starts outputting from point (8).

Handshake State (Read Packet : BS2 / Write Packet : BS3)

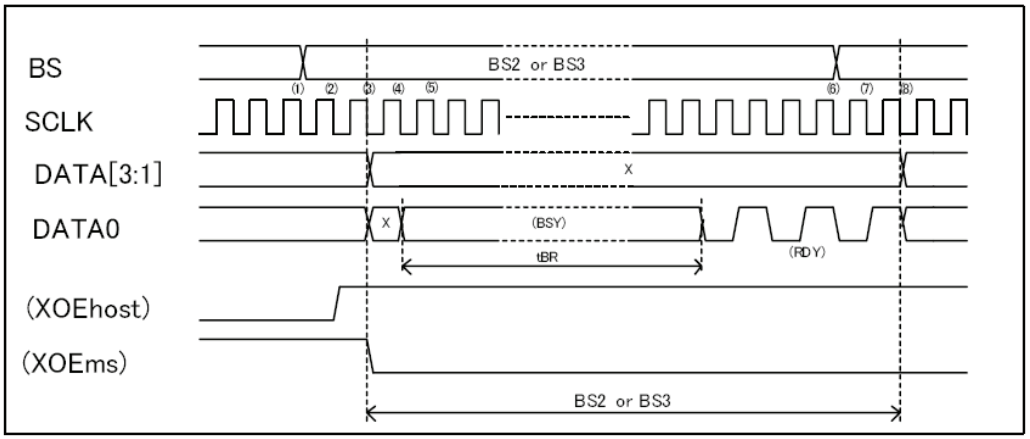


Figure 5.15 4-bit Parallel Protocol Handshake State

Operation	Description
Host	After switching the BS at point (1) above, the detection of a RDY signal is started from point (5). After receiving the RDY signal, which is reversed for each SCLK, for more than or equal to 4 SCLKs, the BS signal is switched at the falling edge of the SCLK after point (6), and the next Bus State begins at point (8).
Memory Stick	After detecting the switch of BS at point (2) above, BSY/RDY signal are output at point (3). After outputs the RDY signal by the completion of internal processing, the Memory Stick detects the switching of BS at point (7), and starts the next Bus State at point (8).

INT Transfer State (BS0)

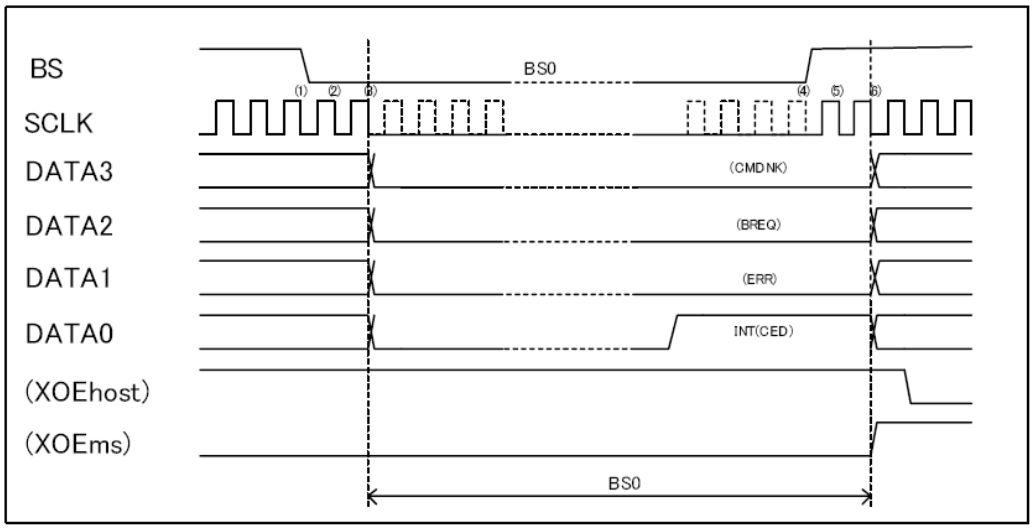


Figure 5.16 4-bit Parallel Protocol INT Transfer State

Operation	Description
Host	SCLK can be halted after point (3). If one of DATA[3:0] becomes High in the INT Transfer State, it will be detected as an INT signal. Confirms the interrupt factor from signals lines in High Level.
Memory Stick	It detects BS0 at point (2) above, and starts the INT transfer State at point (3). After point (3), DATA[3:0] shall be in the output status, and output 0 until the INT signals are output. The cause of interrupt shall be reflected to the INT Register and the INT signals are output on DATA[3:0]

5.4.3 8-bit Parallel Interface Mode

TPC Transfer State (BS1)

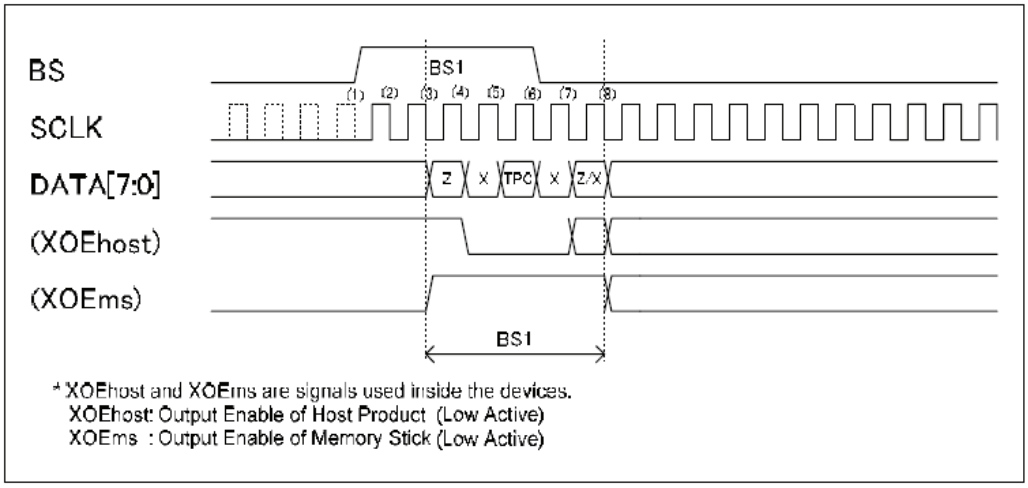


Figure 5.17 8-bit Parallel Protocol TPC Transfer State

Operation	Description
Host	A host controller outputs the BS as High at (1) above, and starts to supply the SCLK before (2). It outputs a TPC on the DATA[7:0] at (5), switches the BS to Low at (6), and starts BS2 from (8).
Memory Stick	Memory Stick detects that the BS is High at (2) and receives the TPC at (6). It detects that the BS is Low at (7), and starts BS2 from (8).

Data Transfer State (Read Packet : BS3)

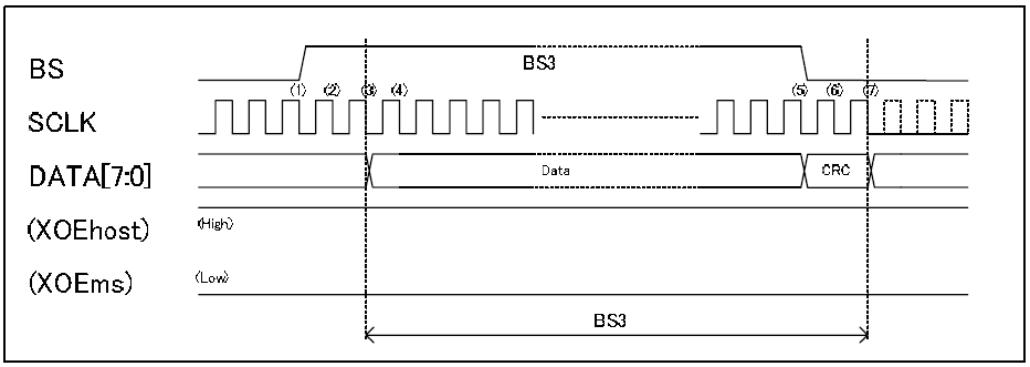


Figure 5.18 8-bit Parallel Protocol Data Transfer State (BS3)

Operation	Description
Host	In a read packet, after switching the BS to High at (1) above, the host controller starts to receive data at (4) which is 3 SCLKs behind from (1). After switching the BS to Low at (5), the host controller receives the lower 8 bits out of a 16-bit CRC at (7) so that the read packet will end.
Memory Stick	Memory Stick detects the BS as High at (2), and starts to output data from (3). It detects the BS as Low at (6) and shifts the state to BS0 at (7).

Data Transfer State (Write Packet : BS2)

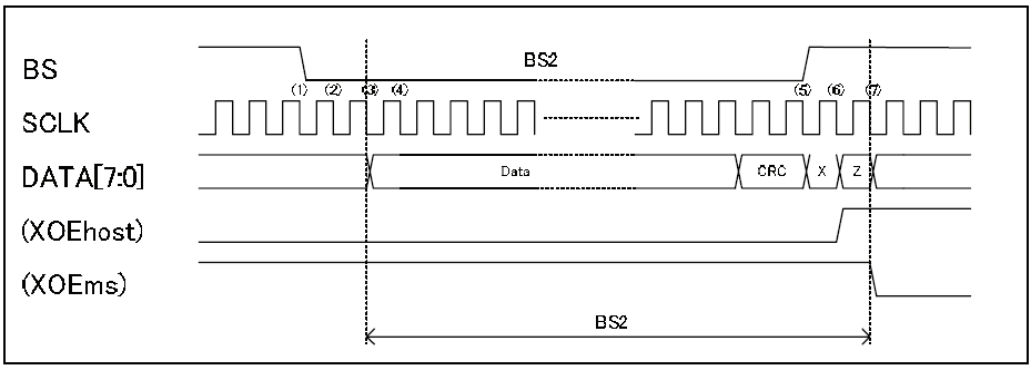


Figure 5.19 8-bit Parallel Protocol Data Transfer State (BS2)

Operation	Description
Host	In a write packet, after switching the BS to Low at (1) above, the host controller starts to transfer data at (3) which is 2 SCLKs behind from (1). The host controller transfers the 16-bit CRC for all of the data from the next SCLK after the last data and switches the BS to High at (5).
Memory Stick	The Memory Stick detects the BS as Low at (2), and starts to receive data at (4). After receiving the lower 8 bits out of a 16-bit CRC at (5), the Memory Stick detects the BS as High at (6), and starts to output s BSY signal or a RDY signal in BS3 (a handshake state) from (7).

Handshake State (Read Packet : BS2 / Write Packet : BS3)

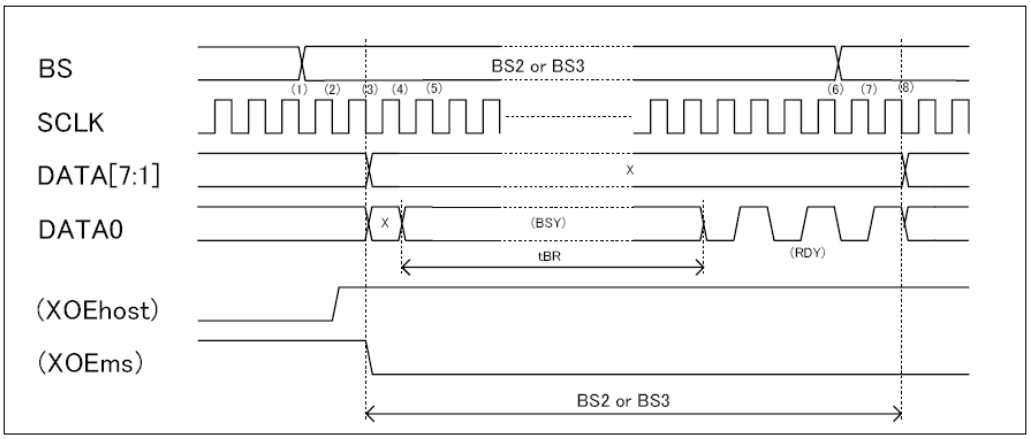


Figure 5.20 8-bit Parallel Protocol Handshake State

Operation	Description
Host	After switching the BS at (1) above, the host controller starts to detect a RDY signal from (5). After receiving the RDY signal the host controller switches the BS signal at a falling edge of the SCLK after (6), and starts the next bus state from (8).
Memory Stick	At (2), Memory Stick detects that the BS has been switched, and then it starts to output a BSY signal or a RDY signal from (3). The Memory Stick outputs a RDY signal after its internal process has completed. It then detects that the BS has been switched again at (7), and starts the next bus state from (8).

INT Transfer State (BS0)

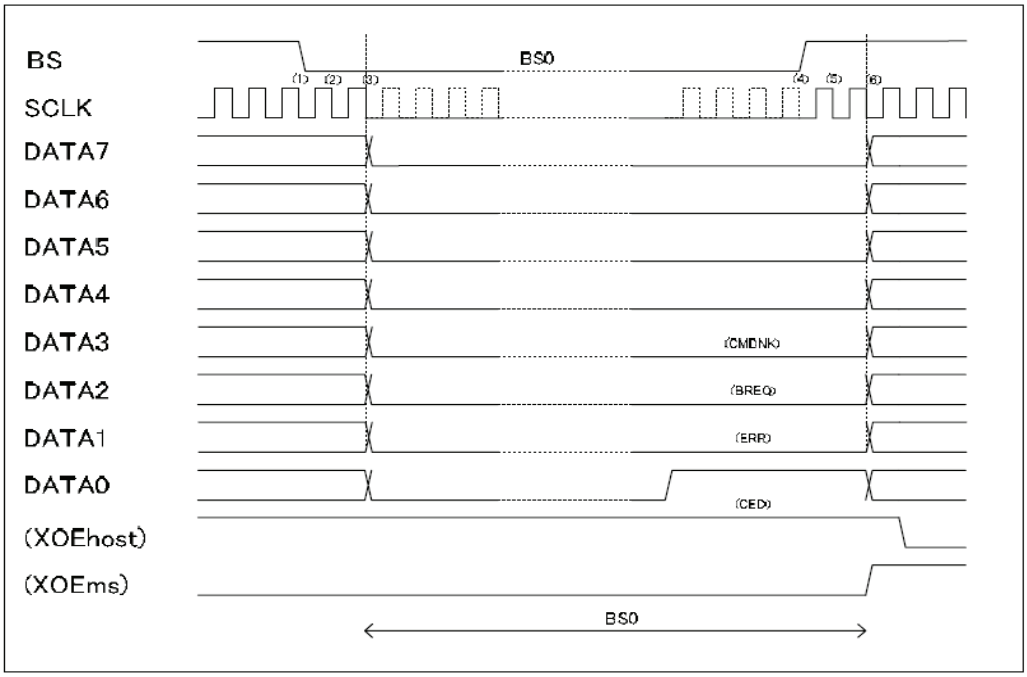


Figure 5.21 8-bit Parallel Protocol INT Transfer State

Operation	Description
Host	A host controller can stop the SCLK after (3). If any of the DATA[3:0] becomes High in this bus state, data is detected as an INT signal. The host controller identifies the interrupt factor through the signal line with High.
Memory Stick	Memory Stick detects BS0 at (2) above, and starts an INT transfer state from (3). After (3), the Memory Stick sets the output mode on the DATA[7:0], and outputs Low until an INT signal is output. The cause of an interrupt is reflected to the INT register and the INT signal is output on DATA[3:0] (High).

5.5 I-CON Block

The main function of the I-CON implemented on the smshc_i is to control the smshc on behalf of a host CPU. This chapter explains about the control of the smshc, self-run, microcode instructions, and data transfer.

5.5.1 Control of smshc

The I-CON performs a self-run. As the “Terminology” page says at the very beginning of this document that “self-run” is a state in which the I-CON performs microcode and controls the smshc, its flow can be detailed as follows:

A host CPU writes microcode to [Instruction Queue].
(54) The host CPU sets 1 to the START bit of [Control Register].
(55) The I-CON will be in a self-run mode. It reads, decodes, and executes microcode.
(fetch → decode → execute)

The above tasks performed by a self-run include a control of the smshc.

5.5.2 Self-run

Starting of Self-run

The I-CON starts instructions specified by microcode by writing 1 to the START bit of [Control Register] through a host CPU. The address where the instruction starts from can be specified by the PRGC bits of [Program Counter Register]. To communicate with memory stick using a self-run, the INTEN bit of [MSHC-System Register] shall remain 1.
The following operations are prohibited during a self-run.

- Writing data to an address which does not belong to any of Group A to E listed below
- Changing a bit which belongs to neither Group A nor B
- Reading neither Group F nor G

Group A to E represent as below.

A : START and INTC bits of [Control Register]
B : SRST bit of [System Register]
C : [Program Counter Register]
D : [General Data FIFO]
E : [PageBuffer]
F : [Instruction Queue]
G : [MSHC Register]

Halt of Self-run

During a self-run, if a halt instruction is executed, or such a bit as MIEF, FLG, or ITOF of [Flag Register] changes from 0 to 1 due to a communication problem, the self-run automatically stops and an interrupt occurs. After the halt of the self-run, the START bit of [Control Register] returns to 0.

Termination of Self-run

If 0 is written to the START bit of [Control Register] while it is 1, a self-run of the I-CON can be terminated on purpose as soon as the instruction underway at the moment is completed. When the self-run is terminated, [Program Counter Register] also stops and indicates the address of an instruction next to the one that has been just completed. After the ongoing instruction is completed, the STPF bit of [Flag Register] becomes 1 and an interrupt occurs due to the aborted self-run.

- Even though 1 is written back to the START bit after the termination process has been completed, there is no guarantee that it can lead a resumption of the self-run.
- It requires an asynchronous reset or a synchronous reset to start it over.

5.5.3 Instruction

Overview

Every instruction is 16-bit variable. The value shown in parentheses attached to each instruction represents its length. (1 word is equivalent to 16 bits.)
The structure of each instruction code is as follows.

3 bits (Instruction ID)	3bits (Instruction word length -1)	10 bits (varies depending on each instruction)

MSB **LSB**

Table 5.12 Instruction ID

Instruction ID	Instruction	Description
000	Wait / Halt	A simple wait for a specified period. To wait for an INT from memory stick To halt
001	TPC	To send an specified TPC to communicate with memory stick
010	Load	To load an immediate value or the value of an [MSHC Register] onto [General Register]
011	Store	To store the value of [General Register] or an immediate value into [MSHC Registers] or [PageBuffer]
100	Compare	To compare between immediate value and that of [General Register] or [MSHC Registers]
101	Jump	True/false/unconditional jump Where to jump is specified by an absolute address/relative address
110	-	-
111	Nop	No Operation

Wait/Halt Instructions (1 Word)

This type of instructions waits for an INT from memory stick, conducts a wait for a predetermined period, and halts an instruction (Halt). For setting up TimeCount, refer to [Timer Register (0x1C)].

Instruction Code	Note	Description
000_000_11_xxxx_tttt	t:TimeCount	To wait for an INT from memory stick (with a timeout) In case of a timeout, 1 is set to the ITOF bit of [Flag Register] and the instruction is terminated.
000_000_01_xxxx_xxxx		To wait for an INT from memory stick (without a timeout)
000_000_10_xxxx_tttt	t:TimeCount	A simple wait for a specified period.
000_000_00_xxxx_CCCC	C:ExitCode	To halt an instruction Consequently, ExitCode will be set to the EXTS bit of [Flag Register].

TPC Instructions (2 to 8 Words)

A TPC instruction issues a TPC to the smshc. There are seven types of TPC instruction.

(56) Instruction 1 to send a TPC through [MSHC-Data Register]
→ Data to be sent is recorded in microcode
(57) Instruction to receive a TPC through [MSHC-Data Register]
→ Data to be received is read by a Load instruction through [MSHC-Data Register].
(58) Instruction to send/receive a TPC through [General Data FIFO]
→ A data transfer request occurs during a self-run.
(59) Instruction 1 to send/receive a TPC through [PageBuffer]
→ A data transfer request occurs during a self-run.

(60) Instruction 2 to send a TPC through [MSHC-Data Register]
→ Data to be sent is 5-byte data including the value of [General Register]
(61) Instruction 3 to send a TPC through [MSHC-Data Register]
→ Data to be sent is 9-byte data including the value of [General Register].
(62) Instruction 2 to send/receive through [PageBuffer]
→ A data transfer request occurs during a self-run.

A TPC timeout can be determined by the lower 4 bits of an instruction code (tttt). However, if tttt is 0000b, there should not be any TPC timeout. For how to specify TimeCount, refer to [Timer Register (0x1C)]. How to handle a TPC communication error can be specified by the E bit of an instruction code.
E=0: ignore communication errors.
E=1: In case of an error occurrence, a self-run is terminated.

No.	Instruction Code	Note	Description
(1)	001_LLL_000_xE_xttttt	L:Length t:TimeCount T:TPC S:DataSize	To send an immediate value to memory stick through [MSHC-Data Register] The maximum size is 12 bytes for the data to be sent as an immediate value.
(2)	001_001_001_xE_xtttt TTTT_0SSS_SSSS_SSSS	t: TimeCount T: TPC S:DataSize [Byte]	To wait for an INT from memory stick (without a timeout)
(3)	001_001_110_xE_xtttt TTTT_1SSS_SSSS_SSSS	t: TimeCount T: TPC S: DataSize [Byte]	To send/receive data to/from memory stick through [General Data FIFO] For the data size to be sent/received, refer to [General Data FIFO (0x24)].
(4)	001_001_111_xE_xtttt TTTT_1SSS_SSSS_SSSS	t: TimeCount T: TPC S: DataSize [Byte]	To send/receive data to/from memory stick through [PageBuffer] Data size to be sent: 512 bytes, or 2 Kbytes Data size to be received: 512 bytes, 2 Kbytes, or any integer multiple of 4 bytes
(5)	001_001_010_xE_xtttt TTTT_0SSS_SSSS_SSSS	t: TimeCount T: TPC S: DataSize [Byte]	Block Address send mode To send data to memory stick through [MSHC-Data Register] To send the total 5-byte data of {0x00,[GeneralRegister2,3]}. This instruction code is used to make a Logical/physical transportation table.
(6)	001_001_011_xE_xtttt TTTT_0SSS_SSSS_SSSS	t: TimeCount T: TPC S: DataSize [Byte]	Expanded Block Address send mode To send data to memory stick through [MSHC-Data Register] To send the total 9-byte data of {0x00,[General Register2,3,4,5]}.
(7)	001_001_100_xE_xtttt TTTT_1SSS_SSSS_SSSS	t: TimeCount T: TPC S: DataSize [Byte]	Expanded TPC instruction To send/receive data to/from memory stick through [PageBuffer] This instruction code is used to receive an INT from memory stick, to check the interrupt state, and to send a TPC. Data size to be sent: 512 bytes, or 2 Kbytes Data size to be received: 512 bytes or 2 Kbytes

During the process of an expanded TPC instruction, a self-run is terminated regardless of the E bit when a communication error occurs through GET_INT TPC (TOC =1 or CRC = 1 in [MSHC-Status Register]). Regardless of the E bit, a communication error will be reflected to the FLG bit of [Flag Register].

Load Instructions (1 to 2 Words)

A load instruction writes specified data to General Registers. There are two types of load instructions.

(63) Instruction to read a value from one of MSHC Registers (0x30 to 0x4F) and write it to one of [General Register0 to 5]
(64) Instruction to read an immediate value of microcode and writes it to one of [General Register0 to 5]

Only when reading [MSHC-Data Register] (0x34), an access width can be selected from either 4 bytes or 2 bytes.

If 4 bytes is selected, the upper 2 bytes of [MSHC-Data Register] are written to one of [General Register0, 2, 4] while the lower 2 bytes of [MSHC-Data Register] to the corresponding [General Register1, 3, 5].

No.	Instruction Code	Note	Description
(1)	010_000_RR_1AAA_AAAW	R: Register Number, A: I/O address, W: Access Width (Refer to Table 5.16 , Table 5.17, Table 5.18.)	To write the values of an specified [MSHC Register] to a designated [General Register]
(2)	010_001_RR_0xxx_xxxx (immediate)	R: Register Number (Refer to Table 5.16)	To write immediate values to a designated [General Register]

Store Instructions (1 to 2 Words)

Store instruction writes the values of a General Register or the immediate values of microcode to a specified place. There are three types of store instructions.

(65) To read the values of a General Register, and write them to [PageBuffer]
(66) To read the values of a General Register, and write them to an [MSHC Register]
(67) To read the immediate values of microcode, and write them to [PageBuffer]

Only when writing values to [MSHC-Data Register], the access width can be selected from either 4 bytes or 2 bytes.

If 4 bytes is selected, any value among [General Register0, 2, 4] is written to the upper 2 bytes of [MSHC-Data Register] while the value of the corresponding [General Register1, 3, 5] to the lower 2 bytes of [MSHC-Data Register].

In case of writing values to any other register than [MSHC-Data Register], the access width shall be 2 bytes.
In case of writing values to [PageBuffer], the access width shall be always 4 bytes.
In case of writing values to [PageBuffer], a store instruction is allowed only when the PDIR bit of [Control Register] is 1.

No.	Instruction Code	Note	Description
(1)	011_000_RR_1AAA_AAAW	R: Register Number, A: I/O address, W: Access Width (Refer to Table 5.16 , Table 5.17, Table 5.18.)	To write the values of a [General Register] to [PageBuffer] To write the values of a [General Register] to an specified [MSHC Register]
(2)	011_001_xx_0010_1000 (immediate)		To write immediate values to [PageBuffer]

Table 5.13 Access Flag Set to [General Register4,5]

Instruction Code	Description
011_000_01_1111_1100	To set an access flag to [General Register4, 5]
011_000_00_1111_1100	To cancel an access flag on [General Register4, 5]

Compare Instructions (2 to 3 Words)

This type of instructions compares an expected value and that of a General Register or an MSHC Register, and reflects the result to the FLG bit of [Flag Register]. When comparing with the value of an MSHC Register, the value shall be first loaded to a General Register before the comparison.

If loaded from [MSHC-Data Register], the access width can be selected from either 4 bytes or 2 bytes. However, even when 4 bytes is selected, it is only the upper 2 bytes that are to be compared.

To compare values effectively, each bit can be masked. 1 shall be set to all the bits to be masked. The results from the comparison will be reflected to the FLG bit of [Flag Register]. When true, the bit is set to 1 while it is set to 0 for false.

In addition, a pre-increment/pre-decrement by 1 or a pre-increment/pre-decrement by 4 to the value of a General Register is available to perform a loop process. In the case of a pre-increment/pre-decrement by 1, the register value wraps around

when it overflows or underflows. If two General Registers, General Register A and General Register B as shown in the table below, are compared, a pre-increment/pre-decrement by 1 or a pre-increment/pre-decrement by 4 is executed to General Register A.

PPP	Description
000	No action
010	No action
100	+ 1 to the value of a [General Register] before executing microcode
110	- 1 to the value of a [General Register] before executing microcode
001	No action
011	No action
101	+ 4 to the value of a [General Register] before executing microcode
111	- 4 to the value of a [General Register] before executing microcode

Instruction Code	Note	Description
100_001_RR_0PPP_0xxx (Expected values)	R: Register Number (Refer to Table 5.16 , Table 5.17, Table 5.18.)	To compare an immediate value and that of [General Register] (No bit mask exists)
100_001_RR_1AAA_AAAW (Expected values)	R: Register Number A: I/O address W: Access Width (Refer to Table 5.16 , Table 5.17, Table 5.18.)	To compare an immediate value and that of [General Register] after automatically loaded (No bit mask exists)
100_010_RR_0PPP_0xxx (Expected values) (Bit mask)	R: Register Number (Refer to Table 5.16)	To compare an immediate value and that of [General Register] (Bit masks exist)
100_010_RR_1AAA_AAAW (Expected values) (Bit mask)	R: Register Number A: I/O address W: Access Width (Refer to Table 5.16 , Table 5.17, Table 5.18.)	To compare an immediate value and that of [General Register] after automatically loaded (Bit masks exist)
100_000_AA_0PPP_1xBB	A: [General Register-A] B: [General Register-B] (Refer to Table 5.16)	To compare two [General Register] one another (No bit mask exists) Pre-increment /Pre-decrement are executed to [General Register-A].
100_001_AA_0PPP_1xBB (Bit mask)	A: [General Register-A] B: [General Register-B] (Refer to Table 5.16)	To compare two [General Register] one another (Bit masks exist) Pre-increment /Pre-decrement are executed to [General Register-A].

Jump Instructions (1 Word)

Whether a true jump, false jump, or unconditional jump is performed depending on the FLG bit value of [Flag Register].
The address for a jump can be selected from either an absolute address or relative address. A relative address is represented with a 2's complement.

R=0: absolute address
R=1: relative address (address represented with a 2's complement)

Instruction Code	Note	Description
101_000_00_RAAA_AAAA	A: Address	True for a jump
101_000_01_RAAA_AAAA	A: Address	False for a jump
101_000_10_RAAA_AAAA	A: Address	Unconditional

The range of an address is between 0 and 127 for an absolute address while between -64 and +63 for a relative address.

Table 5.14 How to Specify Absolute Address

RAAA_AAAA	Instruction Queue Address Pointer (Absolute Address)
0000_0000	0
0000_0001	1
0000_0010	2
....	....
0111_1101	125
0111_1110	126
0111_1111	127

Table 5.15 How to Specify Relative Address

RAAA_AAAA	Instruction Queue Range of Address Pointer (Relative Address)
1011_1111	+63
1011_1110	+62
1011_1101	+61
....	....
1000_0001	+`
1000_0000	0
1111_1111	-1
....	....
1100_0010	-62
1100_0001	-63
1100_0000	-64

Nop Instructions (1 Word)

No operation

Instruction Code	Note	Description
111_000_xx_xxxx_xxxx		No operation

Table 5.16 Setting of [General Register]

Register Number	Register	Note
00	[General Register0]	
01	[General Register1]	
10	[General Register2]	When the access flag to [General Register4, 5] has been set, [General Register4] is accessed. The access flag can be set by a store instruction.
11	[General Register3]	When the access flag to [General Register4, 5] has been set, [General Register5] is accessed. The access flag can be set by a store instruction.

Table 5.17 Setting of I/O Address

Register Number	Register	Note
0x28	[PageBuffer]	[PageBuffer] can be specified only by a store instruction.
0x30	[MSHC-Command Register]	
0x34	[MSHC-Data Register]	
0x38	[MSHC-Status Register]	
0x3C	[MSHC-System Register]	
0x40	[MSHC-User Custom Register]	
0x44	[MSHC_FIFO Control Register]	
0x48	Reserved	An access is prohibited.
0x4C	[MSHC-DMA Control Register]	

Table 5.18 Setting of Access Width

W	Access Width
0	2 bytes
1	4 bytes

Table 5.19 [General Register] Accessed when Access Width Is 1

Register Number	[General Register] to be accessed	Note
00	[General Register0,1]	
01	-	Setting is prohibited.
10	[General Register2,3]	When the access flag to [General Register4, 5] has been set, [General Register4,5] is accessed.
11	-	Setting is prohibited.

5.5.4 Data Transfer

Direction of Data Transfer

During a self-run (START=1 of [Control Register]) of the smshc_i, the data transfer direction of [PageBuffer] and [General Data FIFO] cannot be changed.

- The data transfer direction of [General Data FIFO] is specified by the GDIR bit of [Control Register].
- The data transfer direction of [PageBuffer] is specified by the PDIR bit of [Control Register].

Therefore, during a self-run (START=1);
Changing GDIR is prohibited.
Changing PDIR is prohibited.

Also, while a self-run is halted (START=0);
Changing GDIR is prohibited if any data remains in [General Data FIFO].
Changing PDIR is prohibited if any data remains in [PageBuffer].

Number of Data Transfers

1. Number of DMA transfers of data written to [General Data FIFO]

When data is written to [General Data FIFO] from a host CPU or a DMA controller during a self-run, the number of transfers shall be set to the GFDN bits of [Memory Control Register].

- The transfer unit is 4 bytes.
- Also refer to [General Data FIFO (0x24)].

2. Total number of pages written to [PageBuffer]

When data is written to [PageBuffer] from a host CPU or a DMA controller during a self-run, the total number of pages shall be set to [General Register] specified by GRPN.

- The transfer unit is 512 bytes. However, if the size of data transferred to memory stick is not divisible by 512, it shall be rounded up to a multiple that can be divided by 512.
- Also refer to [PageBuffer (0x28)].

Slice Size

To access [General Data FIFO] and [PageBuffer] from a host CPU or a DMA controller during a self-run, the data size shall be only a multiple of Word (32bit). Listed below are the slice sizes.

- "Slice Size" is the size of transfer data per XDRQ assertion. It shall be set depending on the specifications of a DMA controller.
- If the slice size of the smshc_i does not agree with that of a DMA controller, a proper operation cannot be achieved.

Table 5.20 DMA Slice Size of [General Data FIFO]

GDSZ[1:0]	Slice Size	Capacity of General Data FIFO			
		16 bytes	32 bytes	64 bytes	128 bytes
00	4 bytes	o	o	o	o
01	16 bytes	x	o	o	o
10	32 bytes	x	x	o	o
11	128 bytes	x	x	x	o

When the data of a fraction smaller than a slice size is read/written from/to [General Data FIFO];

- In case of DMAE0/1=1: The data transfer request is cleared as soon as DMA acknowledgement[Channel0/1] is asserted after the fraction data has been read/written.
- In case of DMAE0/1=0: The data transfer request is cleared after the fraction data has been read/written.

Table 5.21 DMA Slice Size of [PageBuffer]

PDSZ[2:0]	Slice Size
000	4 bytes
001	16 bytes
010	32 bytes
011	64 bytes
100	128 bytes
101	256 bytes
110	Reserved
111	Reserved

Data of a fraction smaller than a slice size:

To write it to [PageBuffer];

- Prohibited. The data size shall be always equal to that of the slice size.

To read it from [PageBuffer];

- In case of DMAE0/1=1: The data transfer request is cleared as soon as DMA acknowledgements[Channel0/1] are asserted after the fraction data has been read.
- In case of DMAE0/1=0: The data transfer request is cleared after the fraction data has been read.

Data Transfer Request of General Data FIFO

1. Reading Data from [General Data FIFO]

When GDIR = 0, data is being read from [General Data FIFO].

- During a self-run:
  When data stored in [General Data FIFO] reaches the volume of the slice size, a data transfer request occurs.
  On the other hand, while data stored in [General Data FIFO] remains smaller than the slice size, no data transfer request occurs.

After a self-run, if any data is left over in [General Data FIFO], a data transfer request occurs.
Once the data transfer has been completed, no more data transfer request occurs.

2. Writing Data to [General Data FIFO]

When GDIR=1, data is being written to [General Data FIFO].

- During a self-run:
  If there remains a space equivalent to the slice size or larger in [General Data FIFO], and also the number of transfers is less than the value set in the GFDN bits, a data transfer request occurs. When the number of transfers specified in the GFDN bits of [Memory Control Register] is completed, no more data transfer request will occur.
- While a self-run is being halted:
  No data transfer request occurs.

Data Transfer Request of PageBuffer

1. Reading Data from [PageBuffer]

When PDIR = 0, data is being read from [PageBuffer].

- During a self-run:
  When data stored in [PageBuffer] reaches the volume of the PBBC size, a data transfer request occurs.

Until data of the PBBC size has been completely read, data transfer requests will occur multiple times depending on a slice size. If the data written in [PageBuffer] is smaller than the PBBC size, no data transfer request will occur.

After a self-run, if any data is left over in [PageBuffer], a data transfer request occurs.
Once the data transfer request is completed, no more data transfer request will occur.

2. Writing Data to [PageBuffer]

When PDIR=1, data is being written to [PageBuffer].

- During a self-run:
  If there remains a space equivalent to the PBBC size or larger in [PageBuffer], and also the number of transferred pages is less than the total number of transfer pages, a data transfer request occurs.

Data transfer requests for writing data of the PBBC size will occur multiple times depending on a slice size.
Once the total transfer page data has been transferred, no more Data transfer request occurs.
A total data volume written to [PageBuffer] shall be a multiple of the PBBC size ×2.
If data to be transferred is not a multiple of the PBBC size ×2, it shall be so adjusted by adding arbitrary data to the end of the data.
Such added data is likely to be left over in [PageBuffer] when a self-run is completed. In this case, an initialization is required through a reset of I-CON or the PBCR bit of [Memory Control Register].

- While a self-run is being halted:
  No Data transfer request occurs.

6 GMAC

6.1 Overview

The DWC Ether MAC 10/100/1000 Universal, commonly referred to as GMAC-UNIV in this document, enables a host to transmit and receive data over Ethernet in compliance with the IEEE 802.3-2002 standard. The GMAC-UNIV can have four major configurations: GMAC core only with native interface (GMACCORE), GMAC with transaction layer (GMAC-MTL), GMAC with native DMA (GMAC-DMA), and GMAC with AHB-interfaced DMA (GMAC-AHB).

The GMAC-UNIV provides an optimized (with respect to gate count and latency), configurable, flexible product to meet the needs of various applications and customers, and supports a multitude of industry standard interfaces to the PHY, in addition to the default Gigabit Media Independent Interface (GMII)/Media Independent Interface (MII) defined in the IEEE 802.3 specifications. The GMAC-UNIV can be used in number of applications such as switches, network interface cards, etc. The GMAC-AHB is designed to interface to the industry standard AMBA High-Performance Bus (AHB) on the application side.

The GMAC-UNIV is compliant to the following standards:

- IEEE 802.3-2002 for Ethernet MAC, GMII
- IEEE 1588-2002 standard for precision networked clock synchronization
- AMBA 2.0 for AHB Master/Slave ports
- RGMII specification from HP/Marvell for RGMII

6.2 System Overview

6.2.1 System-Level Block Diagram

A system-level block diagram is shown in Figure 1.1.

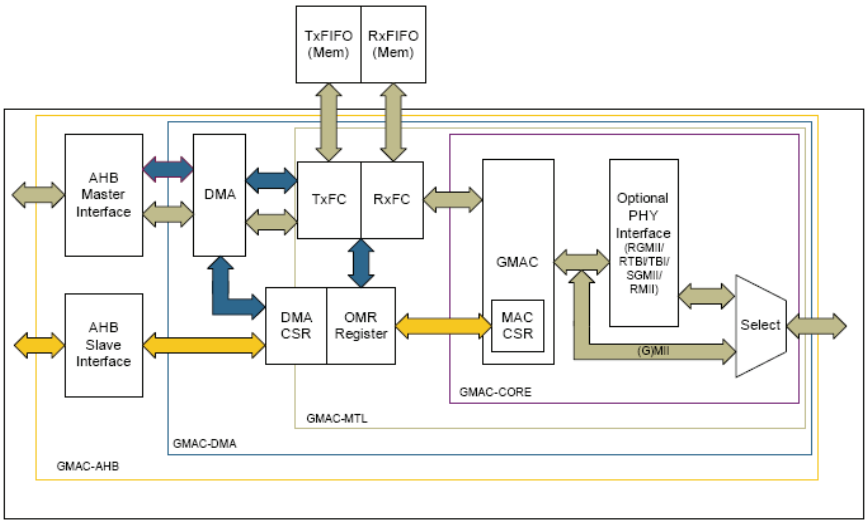


Figure 6.1 GMAC-UNIV Block Diagram

6.2.2 Interface

The GMAC-AHB transfers data to system memory through the AHB master interface. This interface can be removed for a non-AHB system, and the subsystem will have a direct native FIFO-type interface.

The host CPU uses the default 32-bit AHB Slave interface to access the GMAC subsystem's Control and Status registers (CSRs). There is an option to select an APB port for CSR access instead of the AHB Slave port. For non-AHB systems, the AHB/APB slave modules can be removed and the native 32-bit Read/Write bus is provided for CPU accesses.

The GMAC-UNIV supports any one or a combination of the following PHY interfaces:

- Gigabit Media Independent Interface (GMII)/Media Independent Interface (MII)
- Reduced GMII (RGMII)

6.2.3 Transmit and Receive FIFOs

The Transmit FIFO (Tx FIFO) buffers data read from system memory by the DMA before transmission by the GMAC Core. Similarly, the Receive FIFO (Rx FIFO) stores the Ethernet frames received from the line until they are transferred to system memory by the DMA. These are asynchronous FIFOs, as they also transfer the data between the application clock and the GMAC line clocks.

Both FIFOs are required to be two-ported RAM of configurable depth (35/68/133 bits wide for 32/64/128 data bus widths).

In the GMAC-MTL configuration, an additional two-port RAM (width configurable from 12 to 15 bits) is needed to store the frame lengths of received frames in the Rx FIFO. The depth of this frame-length FIFO depends on the maximum number of frames that can be stored in the Rx FIFO. This frame-length FIFO is optional, and is disabled by default.

6.3 Features List

The GMAC-UNIV includes the following features, listed by category.

6.3.1 GMAC Core Features

- Supports 10/100/1000-Mbps data transfer rates with the following PHY interfaces
  - IEEE 802.3-compliant GMII/MII (default) interface to communicate with an external Gigabit/Fast Ethernet PHY
  - RGMII interface to communicate with an external gigabit PHY
- Supports both full-duplex and half-duplex operation
  - Supports CSMA/CD Protocol for half-duplex operation
  - Supports packet bursting and frame extension in 1000 Mbps half-duplex operation
  - Supports IEEE 802.3x flow control for full-duplex operation
  - Optional forwarding of received pause control frames to the user application in full-duplex operation
  - Back-pressure support for half-duplex operation
  - Automatic transmission of zero-quanta pause frame on deassertion of flow control input in fullduplex operation
- Preamble and start-of-frame data (SFD) insertion in Transmit, and deletion in Receive paths
- Automatic CRC and pad generation controllable on a per-frame basis
- Options for Automatic Pad/CRC Stripping on receive frames
- Programmable frame length to support Standard or Jumbo Ethernet frames with sizes up to 16 KB
- Programmable InterFrameGap (40-96 bit times in steps of 8)
- Supports a variety of flexible address filtering modes:
  - Up to 31 additional 48-bit perfect (DA) address filters with masks for each byte
  - Up to 31 48-bit SA address comparison check with masks for each byte
  - 64-bit Hash filter (optional) for multicast and uni-cast (DA) addresses
  - Option to pass all multicast addressed frames
  - Promiscuous mode support to pass all frames without any filtering for network monitoring
  - Passes all incoming packets (as per filter) with a status report
- Separate 32-bit status returned for transmission and reception packets
- Supports IEEE 802.1Q VLAN tag detection for reception frames
- Separate transmission, reception, and control interfaces to the Application
- Configurable big endian and little endian support for transmission and reception data paths
- Supports 32/64/128-bit data transfer interface on the system-side
- Complete network statistics (optional) with RMON/MIB Counters (RFC2819/RFC2665)
- MDIO Master interface (optional) for PHY device configuration and management
- Optional module for detection of LAN wake-up frames and AMD Magic Packet frames
- Optional Receive module for checksum off-load for received IPv4 and TCP packets encapsulated by the Ethernet frame
- Optional Enhanced Receive module for checking IPv4 header checksum and TCP, UDP, or ICMP checksum encapsulated in IPv4 or IPv6 datagrams.
- Optional module to support Ethernet frame time stamping as described in IEEE 1588-2002. Sixty-four-bit time stamps are given in each frame's transmit or receive status.

6.3.2 DMA Block Features

The DMA block exchanges data between the MTL block and host memory. A set of registers (DMA CSR) to control DMA operation is accessible by the host.

DMA features include:

- 32/64/128-bit data transfers
- Single-channel Transmit and Receive engines
- Fully synchronous design operating on a single system clock (except for CSR module, when a separate CSR clock is configured)
- Optimization for packet-oriented DMA transfers with frame delimiters
- Byte-aligned addressing for data buffer support
- Dual-buffer (ring) or linked-list (chained) descriptor chaining
- Descriptor architecture, allowing large blocks of data transfer with minimum CPU intervention; each descriptor can transfer up to 8 KB of data
- Comprehensive status reporting for normal operation and transfers with errors
- Individual programmable burst size for Transmit and Receive DMA Engines for optimal host bus utilization
- Programmable interrupt options for different operational conditions
- Per-frame Transmit/Receive complete interrupt control
- Round-robin or fixed-priority arbitration between Receive and Transmit engines
- Start/Stop modes
- Separate ports for host CSR access and host data interface

6.3.3 Transaction layer (MTL) Features

The MTL block consists of two sets of FIFOs: a Transmit FIFO with programmable threshold capability, and a Receive FIFO with a configurable threshold (default of 64 bytes).

MTL features include:

- 32-, 64-, or 128-bit Transaction Layer block providing a bridge between the application and the GMAC-CORE
- Single-channel Transmit and Receive engines
- Data transfers executed using simple FIFO-protocol
- Synchronization for all clocks in the design (Transmit, Receive and system clocks)
- Optimization for packet-oriented transfers with frame delimiters
- Four Separate ports for system-side and GMAC-CORE-side transmission and reception
- Two 2-port RAM-based asynchronous FIFOs with synchronous/asynchronous Read and Write operation with respect to the Read and Write clocks (one for transmission and one for reception)
- FIFO instantiation outside the top-level module to facilitate memory testing/instantiation
- Supports 128-, 256-, or 512-byte, or 1-, 2-, 4-, 8-, 16-, or 32-KB receive FIFO depths on reception.
- Optional interface to indicate the length of a received frame at the top of the MTL Rx FIFO in the GMAC-MTL configuration
- Programmable burst-length support for starting a burst up to half the size of the MTL Rx and Tx FIFO in the GMAC-MTL configuration
- Receive Status vectors inserted into the Receive FIFO after the EOF transfer enables multiple-frame storage in the Receive FIFO without requiring another FIFO to store those frames' Receive Status.
- Configurable Receive FIFO threshold (default fixed at 64 bytes) in Cut-Through mode
- Option to filter all error frames on reception and not forward them to the application in Store-and- Forward mode
- Option to forward under-sized good frames
- Supports statistics by generating pulses for frames dropped or corrupted (due to overflow) in the Receive FIFO
- Supports 256- or 512-byte, or 1-, 2-, 4-, 8-, or 16-KB FIFO depth on transmission
- Supports Store and Forward mechanism for transmission to the GMAC core
- Supports threshold control for transmit buffer management
- Supports configurable number of frames to be stored in FIFO at any time. The default is 2 frames (fixed) with internal DMA, and up to 8 frames in GMAC-MTL configuration.
- Automatic generation of PAUSE frame control or backpressure signal to the GMAC core based on Receive FIFO-fill (threshold configurable) level.
- Handles automatic retransmission of Collision frames for transmission
- Discards frames on late collision, excessive collisions, excessive deferral and underrun conditions
- Software control to flush Tx FIFO
- Data FIFO RAM chip-select disabled when inactive, to reduce power consumption
- Optional module to calculate and insert IPv4 header checksum and TCP, UDP, or ICMP checksum in

frames transmitted in Store-and-Forward mode.

6.3.4 AMBA Interface Features

- Interfaces with the application via the AHB
- AHB Slave interface (32-, 64-, or 128-bit) for CSR access, in which only 32-bit or less (byte, half-word) accesses are possible
- Option for a 32-bit APB port for CSR access instead of an AHB Slave port
- Supports 32-, 64-, or 128-bit data on the AHB Master port
- Supports Split, Retry, and Error AHB responses in the AHB Master interface
- Does not generate Split, Retry, or Error responses in the AHB Slave interface (compatible with AHB-Lite)
- Configurable for Little- or Big-Endian modes
- Supports all AHB burst types in the AHB Slave Interface
- Software can select the type of AHB burst (fixed or indefinite burst) in the AHB Master interface.
- Option to select address-aligned bursts from AHB master port

6.3.5 Monitoring, Test, and Debugging Support Features

- Supports internal loopback on the GMII/MII for debugging
- DMA states (Tx and Rx) given as status bits
- Application Abort status bits
- MMC (RMON) module in the GMAC core
- Current Tx/Rx Buffer pointer as status registers
- Current Tx/Rx Descriptor pointer as status registers

6.4 Register Descriptions

Table 6.1 DMA Register Map (Base Address = 0xB0821000)

Name	Address	BITS	RW	Reset	Description
BMODE	0x00	32	R/W	0x00020101	Bus mode register
TPD	0x04	32	R/W	0x00000000	Transmit poll demand register
RPD	0x08	32	R/W	0x00000000	Receive poll demand register
RDLA	0x0C	32	R/W	0x00000000	Receive descriptor list address register
TDLA	0x10	32	R/W	0x00000000	Transmit descriptor list address reigster
STS	0x14	32	R	-	Status register
OPMODE	0x18	32	R/W	0x00000000	Operation mode register
IE	0x1C	32	R/W	0x00000000	Interrupt enable register
MFBOC	0x20	32	R/C	0x00000000	Missed frame and buffer overflow counter register
CHTD	0x48	32	R	-	Current host transmit descriptor register
CHRD	0x4C	32	R	-	Current host receive descriptor register
CHTBA	0x50	32	R	-	Current host transmit buffer address register
CHRBA	0x54	32	R	-	Current host receive buffer address register

Table 6.2 GMAC Register Map (Base Address = 0xB0820000)

Name	Address	BITS	RW	Reset	Description
MACC	0x000	32	R/W	0x00000000	MAC configuration register
MACFF	0x004	32	R/W	0x00000000	MAC frame filter
HTH	0x008	32	R/W	0x00000000	Hash table high register
HTL	0x00C	32	R/W	0x00000000	Hash table low register
GMIIA	0x010	32	R/W	0x00000000	GMII address register
GMIID	0x014	32	R/W	0x00000000	GMII data register
FC	0x018	32	R/W	0x00000000	Flow control register
VLANT	0x01C	32	R/W	0x00000000	VLAN tag register
VERSION	0x020	32	R/W	0x0000--34	Version register
RWFF	0x028	32	R/W	0x00000000	Remote wake-up frame register
PMTCS	0x02C	32	R/W	0x00000000	PMT control and status
IRQS	0x038	32	R	-	Interrupt status register
IRQM	0x03C	32	R/W	0x00000000	Interrupt mask register
MACA0H	0x040	32	R/W	0x0000FFFF	MAC address0 high register
MACA0L	0x044	32	R/W	0xFFFFFFFF	MAC address0 low register
MACA1H	0x048	32	R/W	0x0000FFFF	MAC address1 high register
MACA1L	0x04C	32	R/W	0xFFFFFFFF	MAC address1 low register
MACA2H	0x050	32	R/W	0x0000FFFF	MAC address2 high register
MACA2L	0x054	32	R/W	0xFFFFFFFF	MAC address2 low register
MACA3H	0x058	32	R/W	0x0000FFFF	MAC address3 high register
MACA3L	0x05C	32	R/W	0xFFFFFFFF	MAC address3 low register

MACA4H	0x060	32	R/W	0x0000FFFF	MAC address4 high register
MACA4L	0x064	32	R/W	0xFFFFFFFF	MAC address4 low register
MACA5H	0x068	32	R/W	0x0000FFFF	MAC address5 high register
MACA5L	0x06C	32	R/W	0xFFFFFFFF	MAC address5 low register
MACA6H	0x070	32	R/W	0x0000FFFF	MAC address6 high register
MACA6L	0x074	32	R/W	0xFFFFFFFF	MAC address6 low register
MACA7H	0x078	32	R/W	0x0000FFFF	MAC address7 high register
MACA7L	0x07C	32	R/W	0xFFFFFFFF	MAC address7 low register
MACA8H	0x080	32	R/W	0x0000FFFF	MAC address8 high register
MACA8L	0x084	32	R/W	0xFFFFFFFF	MAC address8 low register
MACA9H	0x088	32	R/W	0x0000FFFF	MAC address9 high register
MACA9L	0x08C	32	R/W	0xFFFFFFFF	MAC address9 low register
MACA10H	0x090	32	R/W	0x0000FFFF	MAC address10 high register
MACA10L	0x094	32	R/W	0xFFFFFFFF	MAC address10 low register
MACA11H	0x098	32	R/W	0x0000FFFF	MAC address11 high register
MACA11L	0x09C	32	R/W	0xFFFFFFFF	MAC address11 low register
MACA12H	0x0A0	32	R/W	0x0000FFFF	MAC address12 high register
MACA12L	0x0A4	32	R/W	0xFFFFFFFF	MAC address12 low register
MACA13H	0x0A8	32	R/W	0x0000FFFF	MAC address13 high register
MACA13L	0x0AC	32	R/W	0xFFFFFFFF	MAC address13 low register
MACA14H	0x0B0	32	R/W	0x0000FFFF	MAC address14 high register
MACA14L	0x0B4	32	R/W	0xFFFFFFFF	MAC address14 low register
MACA15H	0x0B8	32	R/W	0x0000FFFF	MAC address15 high register
MACA15L	0x0BC	32	R/W	0xFFFFFFFF	MAC address15 low register
RGMIIS	0x0D8	32	R	-	RGMII status register
MMC_CNTRL	0x100	32	RW	0x00000000	MMC control establishes the operating mode of MMC
MMC_INTR_RX	0x104	32	R/C	-	MMC receive interrupt maintains the interrupt generated from all of the receive statistic counters.
MMC_INTR_TX	0x108	32	R/C	-	MMC transmit interrupt maintains the interrupt generated form all of the transmit statistic counters.
MMC_INTR_MASK_RX	0x10C	32	RW	0x00000000	MMC receive interrupt mask maintains the mask for the interrupt generated form all of the receive statistic counters
MMC_INTR_MASK_TX	0x110	32	RW	0x00000000	MMC transmit interrupt mask maintains the mask for the interrupt generated form all of the transmit statistic counters
TXOCTETCOUNT_GB	0x114	32	R	-	Number of bytes transmitted, exclusive of preamble and retried bytes, in good and bad frames.
TXFRAMECOUNT_GB	0x118	32	R	-	Number of good and bad frames transmitted, exclusive of retried frames.
TXBROADCASTFRAMES_G	0x11C	32	R	-	Number of good broadcast frames transmitted.
TXMULTICASTFRAMES_G	0x120	32	R	-	Number of good multicast frames transmitted.
TX64OCTETS_GB	0x124	32	R	-	Number of good and bad frames transmitted with length 64 bytes, exclusive of preamble and retried frames.
TX65TO127OCTETS_GB	0x128	32	R	-	Number of good and bad frames transmitted with length between 65 and 127 (inclusive) bytes, exclusive of preamble and retried frames.
TX128TO255OCTETS_GB	0x12C	32	R	-	Number of good and bad frames transmitted with length between 128 and 255 (inclusive) bytes, exclusive of preamble and retried frames.
TX256TO511OCTETS_GB	0x130	32	R	-	Number of good and bad frames transmitted with length between 256 and 511 (inclusive) bytes, exclusive of preamble and retried frames.
TX512TO1023OCTETS_GB	0x134	32	R	-	Number of good and bad frames transmitted with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble and retried frames.
TX1024TOMAXOCTETS_GB	0x138	32	R	-	Number of good and bad frames transmitted with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.
TXUNICASTFRAMES_GB	0x13C	32	R	-	Number of good and bad unicast frames transmitted.
TXMULTICASTFRAMES_GB	0x140	32	R	-	Number of good and bad multicast frames transmitted.
TXBROADCASTFRAMES_GB	0x144	32	R	-	Number of good and bad broadcast frames transmitted.
TXUNDERFLOWERROR	0x148	32	R	-	Number of frames aborted due to frame underflow error.
TXSINGLECOL_G	0x14C	32	R	-	Number of successfully transmitted frames after a single collision in Half-duplex mode.

TXMULTICOL_G	0x150	32	R	-	Number of successfully transmitted frames after more than a single collision in Half-duplex mode.
TXDEFERRED	0x154	32	R	-	Number of successfully transmitted frames after a deferral in Half-duplex mode.
TXLATECOL	0x158	32	R	-	Number of frames aborted due to late collision error.
TXEXESSCOL	0x15C	32	R	-	Number of frames aborted due to excessive (16) collision errors.
TXCARRIERERROR	0x160	32	R	-	Number of frames aborted due to carrier sense error (no carrier or loss of carrier).
TXOCTETCOUNT_G	0x164	32	R	-	Number of bytes transmitted, exclusive of preamble, in good frames only.
TXFRAMECOUNT_G	0x168	32	R	-	Number of good frames transmitted.
TXEXCESSDEF	0x16C	32	R	-	Number of frames aborted due to excessive deferral error (deferred for more than two max-sized frame times).
TXPAUSEFRAMES	0x170	32	R	-	Number of good PAUSE frames transmitted.
TXVLANFRAMES_G	0x174	32	R	-	Number of good VLAN frames transmitted, exclusive of retried frames.
RXFRAMECOUNT_GB	0x180	32	R	-	Number of good and bad frames received.
RXOCTETCOUNT_GB	0x184	32	R	-	Number of bytes received, exclusive of preamble, in good and bad frames.
RXOCTETCOUNT_G	0x188	32	R	-	Number of bytes received, exclusive of preamble, only in good frames.
RXBROADCASTFRAMES_G	0x18C	32	R	-	Number of good broadcast frames received.
RXMULTICASTFRAMES_G	0x190	32	R	-	Number of good multicast frames received.
RXCRCERROR	0x194	32	R	-	Number of frames received with CRC error.
RXALIGNMENTERROR	0x198	32	R	-	Number of frames received with alignment (dribble) error. Valid only in 10/100 mode.
RXRUNTERROR	0x19C	32	R	-	Number of frames received with runt (<64 bytes and CRC error) error.
RXJABBERERROR	0x1A0	32	R	-	Number of giant frames received with length (including CRC) greater than 1,518 bytes (1,522 bytes for VLAN tagged) and with CRC error. If Jumbo Frame mode is enabled, then frames of length greater than 9,018 bytes (9,022 for VLAN tagged) are considered as giant frames.
RXUNDERSIZE_G	0x1A4	32	R	-	Number of frames received with length less than 64 bytes, without any errors.
RXOVERSIZE_G	0x1A8	32	R	-	Number of frames received with length greater than the maxsize (1,518 or 1,522 for VLAN tagged frames), without errors.
RX64OCTETS_GB	0x1AC	32	R	-	Number of good and bad frames received with length 64 bytes, exclusive of preamble.
RX65TO127OCTETS_GB	0x1B0	32	R	-	Number of good and bad frames received with length between 65 and 127 (inclusive) bytes, exclusive of preamble.
RX128TO255OCTETS_GB	0x1B4	32	R	-	Number of good and bad frames received with length between 128 and 255 (inclusive) bytes, exclusive of preamble.
RX256TO511OCTETS_GB	0x1B8	32	R	-	Number of good and bad frames received with length between 256 and 511 (inclusive) bytes, exclusive of preamble.
RX512TO1023OCTETS_GB	0x1BC	32	R	-	Number of good and bad frames received with length between 512 and 1,023 (inclusive) bytes, exclusive of preamble.
RX1024TOMAXOCTETS_GB	0x1C0	32	R	-	Number of good and bad frames received with length between 1,024 and maxsize (inclusive) bytes, exclusive of preamble and retried frames.
RXUNICASTFRAMES_G	0x1C4	32	R	-	Number of good unicast frames received.
RXLENGTHERROR	0x1C8	32	R	-	Number of frames received with length error (Length type field ≠ frame size), for all frames with valid length field.
RXOUTOFRANGETYPE	0x1CC	32	R	-	Number of frames received with length field not equal to the valid frame size (greater than 1,500 but less than 1,536).
RXPAUSEFRAMES	0x1D0	32	R	-	Number of missed received frames due to FIFO

					overflow. This counter is not present in the GMAC-CORE configuration.
rxfifooverflow	0x1D4	32	R	-	Number of missed received frames due to FIFO overflow. This counter is not present in the GMAC-CORE configuration.
rxvlanframes_gb	0x1D8	32	R	-	Number of good and bad VLAN frames received.
rxwatchdogerror	0x1DC	32	R	-	Number of frames received with error due to watchdog timeout error (frames with a data load larger than 2,048 bytes).
mmc_ipc_intr_mask_rx	0x200	32	R/W	0x00000000	MMC IPC Receive Checksum Offload Interrupt Mask maintains the mask for the interrupt generated from the receive IPC statistic counters.
mmc_ipc_intr_rx	0x208	32	R/C	0x00000000	MMC Receive Checksum Offload Interrupt maintains the interrupt that the receive IPC statistic counters generate.
rxipv4_gd_frms	0x210	32	R	-	Number of good IPv4 datagrams received with the TCP, UDP, or ICMP payload
rxipv4_hdrerr_frms	0x214	32	R	-	Number of IPv4 datagrams received with header (checksum, length, or version mismatch) errors
rxipv4_nopay_frms	0x218	32	R	-	Number of IPv4 datagram frames received that did not have a TCP, UDP, or ICMP payload processed by the Checksum engine
rxipv4_frag_frms	0x21C	32	R	-	Number of good IPv4 datagrams with fragmentation
rxipv4_udsbl_frms	0x220	32	R	-	Number of good IPv4 datagrams received that had a UDP payload with checksum disabled
rxipv6_gd_frms	0x224	32	R	-	Number of good IPv6 datagrams received with TCP, UDP, or ICMP payloads
rxipv6_hdrerr_frms	0x228	32	R	-	Number of IPv6 datagrams received with header errors (length or version mismatch)
rxipv6_nopay_frms	0x22C	32	R	-	Number of IPv6 datagram frames received that did not have a TCP, UDP, or ICMP payload. This includes all IPv6 datagrams with fragmentation or security extension headers
rxudp_gd_frms	0x230	32	R	-	Number of good IP datagrams with a good UDP payload. This counter is not updated when the rxipv4_udsbl_frms counter is incremented.
rxudp_err_frms	0x234	32	R	-	Number of good IP datagrams whose UDP payload has a checksum error
rxtcp_gd_frms	0x238	32	R	-	Number of good IP datagrams with a good TCP payload
rxtcp_err_frms	0x23C	32	R	-	Number of good IP datagrams whose TCP payload has a checksum error
rxicmp_gd_frms	0x240	32	R	-	Number of good IP datagrams with a good ICMP payload
rxicmp_err_frms	0x244	32	R	-	Number of good IP datagrams whose ICMP payload has a checksum error
rxipv4_gd_octets	0x250	32	R	-	Number of bytes received in good IPv4 datagrams encapsulating TCP, UDP, or ICMP data. (Ethernet header, FCS, pad, or IP pad bytes are not included in this counter or in the octet counters listed below).
rxipv4_hdrerr_octets	0x254	32	R	-	Number of bytes received in IPv4 datagrams with header errors (checksum, length, version mismatch). The value in the Length field of IPv4 header is used to update this counter.
rxipv4_nopay_octets	0x258	32	R	-	Number of bytes received in IPv4 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv4 header's Length field is used to update this counter.
rxipv4_frag_octets	0x25C	32	R	-	Number of bytes received in fragmented IPv4 datagrams. The value in the IPv4 header's Length field is used to update this counter.
rxipv4_udsbl_octets	0x260	32	R	-	Number of bytes received in a UDP segment that had the UDP checksum disabled. This counter does not count IP Header bytes.
rxipv6_gd_octets	0x264	32	R	-	Number of bytes received in good IPv6 datagrams encapsulating TCP, UDP or ICMPv6 data

rxipv6_hdrerr_octets	0x268	32	R	-	Number of bytes received in IPv6 datagrams with header errors (length, version mismatch). The value in the IPv6 header's Length field is used to update this counter.
rxipv6_nopay_octets	0x26C	32	R	-	Number of bytes received in IPv6 datagrams that did not have a TCP, UDP, or ICMP payload. The value in the IPv6 header's Length field is used to update this counter.
rxudp_gd_octets	0x270	32	R	-	Number of bytes received in a good UDP segment. This counter (and the counters below) does not count IP header bytes.
rxudp_err_octets	0x274	32	R	-	Number of bytes received in a UDP segment that had checksum errors
rxtcp_gd_octets	0x278	32	R	-	Number of bytes received in a good TCP segment
rxtcp_err_octets	0x27C	32	R	-	Number of bytes received in a TCP segment with checksum errors
rxicmp_gd_octets	0x280	32	R	-	Number of bytes received in a good ICMP segment
rxicmp_err_octets	0x284	32	R	-	Number of bytes received in an ICMP segment with checksum errors
TSC	0x700	32	R/W	0x00000000	Time stamp control register
SSI	0x704	32	R/W	0x00000000	Sub-second increment register
TSH	0x708	32	R	-	Time stamp high register
TSL	0x70C	32	R	-	Time stamp low register
TSHU	0x710	32	R/W	0x00000000	Time stamp high update register
TSLU	0x714	32	R/W	0x00000000	Time stamp low update register
TSA	0x718	32	R/W	0x00000000	Time stamp addend register
TTH	0x71C	32	R/W	0x00000000	Target time high register
TTL	0x720	32	R/W	0x00000000	Target time low register
MACA16H	0x800	32	R/W	0x0000FFFF	MAC address16 high register
MACA16L	0x804	32	R/W	0xFFFFFFFF	MAC address16 low register
MACA17H	0x808	32	R/W	0x0000FFFF	MAC address17 high register
MACA17L	0x80C	32	R/W	0xFFFFFFFF	MAC address17 low register
MACA18H	0x810	32	R/W	0x0000FFFF	MAC address18 high register
MACA18L	0x814	32	R/W	0xFFFFFFFF	MAC address18 low register
MACA19H	0x818	32	R/W	0x0000FFFF	MAC address19 high register
MACA19L	0x81C	32	R/W	0xFFFFFFFF	MAC address19 low register
MACA20H	0x820	32	R/W	0x0000FFFF	MAC address20 high register
MACA20L	0x824	32	R/W	0xFFFFFFFF	MAC address20 low register
MACA21H	0x828	32	R/W	0x0000FFFF	MAC address21 high register
MACA21L	0x82C	32	R/W	0xFFFFFFFF	MAC address21 low register
MACA22H	0x830	32	R/W	0x0000FFFF	MAC address22 high register
MACA22L	0x834	32	R/W	0xFFFFFFFF	MAC address22 low register
MACA23H	0x838	32	R/W	0x0000FFFF	MAC address23 high register
MACA23L	0x83C	32	R/W	0xFFFFFFFF	MAC address23 low register
MACA24H	0x840	32	R/W	0x0000FFFF	MAC address24 high register
MACA24L	0x844	32	R/W	0xFFFFFFFF	MAC address24 low register
MACA25H	0x848	32	R/W	0x0000FFFF	MAC address25 high register
MACA25L	0x84C	32	R/W	0xFFFFFFFF	MAC address25 low register
MACA26H	0x850	32	R/W	0x0000FFFF	MAC address26 high register
MACA26L	0x854	32	R/W	0xFFFFFFFF	MAC address26 low register
MACA27H	0x858	32	R/W	0x0000FFFF	MAC address27 high register
MACA27L	0x85C	32	R/W	0xFFFFFFFF	MAC address27 low register
MACA28H	0x860	32	R/W	0x0000FFFF	MAC address28 high register
MACA28L	0x864	32	R/W	0xFFFFFFFF	MAC address28 low register
MACA29H	0x868	32	R/W	0x0000FFFF	MAC address29 high register
MACA29L	0x86C	32	R/W	0xFFFFFFFF	MAC address29 low register
MACA30H	0x870	32	R/W	0x0000FFFF	MAC address30 high register
MACA30L	0x874	32	R/W	0xFFFFFFFF	MAC address30 low register
MACA31H	0x878	32	R/W	0x0000FFFF	MAC address31 high register
MACA31L	0x87C	32	R/W	0xFFFFFFFF	MAC address31 low register

6.4.1 DMA Registers

Bus mode Register (BMODE) **0xB0821000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-						AAL	4xPBL	USP	RPBL						FB
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PR		PBL						-	DSL					DA	SWR

The Bus Mode register establishes the bus operating modes for the DMA.

Field	Name	RW	Reset	Description
25	AAL	R/W	0x0	Address-Aligned Beats. When this bit is set high and the FB bit equals 1, the AHB interface generates all bursts aligned to the start address LS bits. If the FB bit equals 0, the first burst (accessing the data buffer's start address) is not aligned, but subsequent bursts are aligned to the address. This bit is valid only in GMAC-AHB configuration, and reserved (RO with default value 0) in all other configurations.
24	4xPBL	R/W	0x0	4xPBL Mode When set high, this bit multiplies the PBL value programmed (bits [22:17] and bits [13:8]) four times. Thus the DMA will transfer data in to a maximum of 4, 8, 16, 32, 64 and 128 beats depending on the PBL value.
23	USP	R/W	0x0	Use Separate PBL When set high, it configures the RxDMA to use the value configured in bits [22:17] as PBL while the PBL value in bits [13:8] is applicable to TxDMA operations only. When reset to low, the PBL value in bits [13:8] is applicable for both DMA engines.
22-17	RPBL	R/W	0x01	RxDMA PBL These bits indicate the maximum number of beats to be transferred in one RxDMA transaction. This will be the maximum value that is used in a single block Read/Write. The RxDMA will always attempt to burst as specified in RPBL each time it starts a Burst transfer on the host bus. RPBL can be programmed with permissible values of 1, 2, 4, 8, 16, and 32. Any other value will result in undefined behavior. These bits are valid and applicable only when USP is set high.
16	FB	R/W	0x0	Fixed Burst This bit controls whether the AHB Master interface performs fixed burst transfers or not. When set, the AHB will use only SINGLE, INCR4, INCR8 or INCR16 during start of normal burst transfers. When reset, the AHB will use SINGLE and INCR burst transfer operations.
15-14	PR	R/W	0x0	PRx:Tx priority ratio. RxDMA requests given priority over TxDMA requests in the following ratio. This is valid only when the DA bit is reset. 00: 1:1 01: 2:1 10: 3:1 11: 4:1
13-8	PBL	R/W	0x01	Programmable Burst Length These bits indicate the maximum number of beats to be transferred in one DMA transaction. This will be the maximum value that is used in a single block Read/Write. The DMA will always attempt to burst as specified in PBL each time it starts a Burst transfer on the host bus. PBL can be programmed with permissible values of 1, 2, 4, 8, 16, and 32. Any other value will result in undefined behavior. When USP is set high, this PBL value is applicable for TxDMA transactions only. The PBL values have the following limitations. The maximum number of beats (PBL) possible is limited by the size of the Tx FIFO and Rx FIFO in the MTL layer and the data bus width on the DMA. The FIFO has a constraint that the maximum beat supported is half the depth of the FIFO. For different data bus widths and FIFO sizes, the valid PBL range (including x4 mode) is provided in the following table. If the PBL is common for both transmit and receive DMA, the minimum Rx FIFO and Tx FIFO depths must be considered. Do not program out-of-range PBL values, because the system may not behave properly.
6-2	DSL	R/W	0x0	Descriptor Skip Length This bit specifies the number of Word/Dword/Lword (depending on 32/64/128-bit bus) to skip between two unchained descriptors. The address skipping starts from the end of current descriptor to the start of next descriptor. When DSL value

				equals zero, then the descriptor table is taken as contiguous by the DMA, in Ring mode.
1	DA	R/W	0x0	DMA Arbitration scheme 0: Round-robin with Rx:Tx priority given in bits [15:14] 1: Rx has priority over Tx
0	SWR	R/W	0x0	Software Reset When this bit is set, the MAC DMA Controller resets all GMAC Subsystem internal registers and logic. It is cleared automatically after the reset operation has completed in all of the core clock domains. Read a 0 value in this bit before re-programming any register of the core.

Transmit Poll Demand Register (TPD) 0xB0821004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TPD															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPD															

The Transmit Poll Demand register enables the Transmit DMA to check whether or not the current descriptor is owned by DMA. The Transmit Poll Demand command is given to wake up the TxDMA if it is in Suspend mode. The TxDMA can go into Suspend mode due to an Underflow error in a transmitted frame or due to the unavailability of descriptors owned by Transmit DMA. You can give this command anytime and the TxDMA will reset this command once it starts re-fetching the current descriptor from host memory.

Field	Name	RW	Reset	Description
31-0	TDP	R/W	0x00000000	Transmit Poll Demand When these bits are written with any value, the DMA reads the current descriptor pointed to by Current Host Transmit Descriptor Register. If that descriptor is not available (owned by Host), transmission returns to the Suspend state and DMA Status Register[2] is asserted. If the descriptor is available, transmission resumes.

Receive Poll Demand Register (RPD) 0xB0821008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RPD															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RPD															

The Receive Descriptor List Address register points to the start of the Receive Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word/Dword/Lword-aligned (for 32/64/128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to Receive Descriptor List Address Register is permitted only when reception is stopped. When stopped, Receive Descriptor List Address Register must be written to before the receive Start command is given.

Field	Name	RW	Reset	Description
31-0	RPD	R/W	0x00000000	Receive Poll Demand When these bits are written with any value, the DMA reads the current descriptor pointed to by Current Host Receive Descriptor Register. If that descriptor is not available (owned by Host), reception returns to the Suspended state and Status Register[7] is not asserted. If the descriptor is available, the Receive DMA returns to active state.

Receive Descriptor List Address Register (RDLA) 0xB082100C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDLA-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDLA															

The Receive Descriptor List Address register points to the start of the Receive Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word/Dword/Lword-aligned (for 32/64/128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LS bits low. Writing to Receive Descriptor List Address Register is permitted only when reception is stopped. When stopped, Receive Descriptor List Address Register must be written to before the receive Start command is given.

Field	Name	RW	Reset	Description
31-0	RDLA	R/W	0x00000000	Start of Receive List This field contains the base address of the First Descriptor in the Receive Descriptor list. The LSB bits [1/2/3:0] for 32/64/128-bit bus width) will be ignored and taken as all-zero by the DMA internally. Hence these LSB bits are Read Only.

Transmit Descriptor List Address Register (TDLA) 0xB0821010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TDLA															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDLA															

The Transmit Descriptor List Address register points to the start of the Transmit Descriptor List. The descriptor lists reside in the host's physical memory space and must be Word/DWORD/LWORD-aligned (for 32/64/128-bit data bus). The DMA internally converts it to bus width aligned address by making the corresponding LSB to low. Writing to Transmit Descriptor List Address Register is permitted only when transmission has stopped. When stopped, Transmit Descriptor List Address Register can be written before the transmission Start command is given.

Field	Name	RW	Reset	Description
31-0	TDLA	R/W	0x00000000	Start of Transmit List This field contains the base address of the First Descriptor in the Transmit Descriptor list. The LSB bits [1/2/3:0] for 32/64/128-bit bus width) will be ignored and taken as all-zero by the DMA internally. Hence these LSB bits are Read Only.

Status Register (STS) 0xB0821014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-		TTI	GPI	GMI	GLI	EB			TS			RS			NIS
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AIS	ERI	FBI	-		ETI	RWT	RPS	RU	RI	UNF	OVF	TJT	TU	TPS	TI

The Status register contains all the status bits that the DMA reports to the host. Status Register and is usually read by the Software driver during an interrupt service routine or polling. Most of the fields in this register cause the host to be interrupted. Status Register bits are not cleared when read. Writing 1'b1 to (unreserved) bits in Status Register[16:0] clears them and writing 1'b0 has no effect. Each field (bits[16:0]) can be masked by masking the appropriate bit in Interrupt Enable Register.

Field	Name	RW	Reset	Description
29	TTI	R	-	Time-Stamp Trigger Interrupt This bit indicates an interrupt event in the GMAC core's Time Stamp Generator block. The software must read the GMAC core's Interrupt Status register, clearing its source (Bit 9), to reset this bit to 1'b0. When this bit is high, the interrupt signal from the GMAC subsystem (sbd_intr_o) is high.
28	GPI	R	-	GMAC PMT Interrupt This bit indicates an interrupt event in the GMAC core's PMT module. The software must read the corresponding registers in the GMAC core to get the exact cause of interrupt and clear its source to reset this bit to 1'b0. The interrupt signal from the GMAC subsystem (sbd_intr_o) is high when this bit is high.
27	GMI	R	-	GMAC MMC Interrupt This bit reflects an interrupt event in the MMC module of the GMAC core. The software must read the corresponding registers in the GMAC core to get the exact cause of interrupt and clear the source of interrupt to make this bit as 1'b0. The interrupt signal from the GMAC subsystem (sbd_intr_o) is high when this bit is high.
26	GLI	R	-	GMAC Line interface Interrupt This bit reflects an interrupt event in the GMAC Core's PCS or RGMII interface block. The software must read the corresponding registers in the GMAC core to get the exact cause of interrupt and clear the source of interrupt to make this bit as 1'b0. The interrupt signal from the GMAC subsystem (sbd_intr_o) is high when this bit is high.
25-23	EB	R	-	Error Bits These bits indicate the type of error that caused a Bus Error (error response on the AHB interface). Valid only with Fatal Bus Error bit (Status Register[13]) set. This field does not generate an interrupt. Bit 23 1'b1 Error during data transfer by TxDMA 1'b0 Error during data transfer by RxDMA Bit 24 1'b1 Error during read transfer 1'b0 Error during write transfer Bit 25 1'b1 Error during descriptor access 1'b0 Error during data buffer access
22-20	TS	R	-	Transmit Process State These bits indicate the Transmit DMA FSM state. This field does not generate an interrupt. 3'b000: Stopped; Reset or Stop Transmit Command issued. 3'b001: Running; Fetching Transmit Transfer Descriptor. 3'b010: Running; Waiting for status. 3'b011: Running; Reading Data from host memory buffer and queuing it to transmit buffer (Tx FIFO). 3'b100, 3'b101: Reserved for future use. 3'b110: Suspended; Transmit Descriptor Unavailable or Transmit Buffer Underflow. 3'b111: Running; Closing Transmit Descriptor.
19-17	RS	R	-	Receive Process State These bits indicate the Receive DMA FSM state. This field does not generate an interrupt. 3'b000: Stopped: Reset or Stop Receive Command issued. 3'b001: Running: Fetching Receive Transfer Descriptor.

				3'b010: Reserved for future use. 3'b011: Running: Waiting for receive packet. 3'b100: Suspended: Receive Descriptor Unavailable. 3'b101: Running: Closing Receive Descriptor. 3'b110: Reserved for future use. 3'b111: Running: Transferring the receive packet data from receive buffer to host memory.
16	NIS	R/C	-	Normal Interrupt Summary Normal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in DMA Interrupt Enable Register: Status Register[0]: Transmit Interrupt Status Register[2]: Transmit Buffer Unavailable Status Register[6]: Receive Interrupt Status Register[14]: Early Receive Interrupt Only unmasked bits affect the Normal Interrupt Summary bit. This is a sticky bit and must be cleared (by writing a 1 to this bit) each time a corresponding bit that causes NIS to be set is cleared.
15	AIS	R/C	-	Abnormal Interrupt Summary Abnormal Interrupt Summary bit value is the logical OR of the following when the corresponding interrupt bits are enabled in DMA Interrupt Enable Register: Status Register[1]:Transmit Process Stopped Status Register[3]:Transmit Jabber Timeout Status Register[4]: Receive FIFO Overflow Status Register[5]: Transmit Underflow Status Register[7]: Receive Buffer Unavailable Status Register[8]: Receive Process Stopped Status Register[9]: Receive Watchdog Timeout Status Register[10]: Early Transmit Interrupt Status Register[13]: Fatal Bus Error Only unmasked bits affect the Abnormal Interrupt Summary bit. This is a sticky bit and must be cleared each time a corresponding bit that causes AIS to be set is cleared.
14	ERI	R/C	-	Early Receive Interrupt This bit indicates that the DMA had filled the first data buffer of the packet. Receive Interrupt Status Register[6] automatically clears this bit.
13	FBI	R/C	-	Fatal Bus Error Interrupt This bit indicates that a bus error occurred, as detailed in [25:23]. When this bit is set, the corresponding DMA engine disables all its bus accesses.
10	ETI	R/C	-	Early Transmit Interrupt This bit indicates that the frame to be transmitted was fully transferred to the MTL Transmit FIFO.
9	RWT	R/C	-	Receive Watchdog Timeout This bit is asserted when a frame with a length greater than 2,048 bytes is received (10,240 when Jumbo Frame mode is enabled).
8	RPS	R/C		Receive Process Stopped This bit is asserted when the Receive Process enters the Stopped state.
7	RU	R/C		Receive Buffer Unavailable This bit indicates that the Next Descriptor in the Receive List is owned by the host and cannot be acquired by the DMA. Receive Process is suspended. To resume processing Receive descriptors, the host should change the ownership of the descriptor and issue a Receive Poll Demand command. If no Receive Poll Demand is issued, Receive Process resumes when the next recognized incoming frame is received. Status Register[7] is set only when the previous Receive Descriptor was owned by the DMA.
6	RI	R/C		Receive Interrupt This bit indicates the completion of frame reception. Specific frame status information has been posted in the descriptor. Reception remains in the Running state.
5	UNF	R/C		Transmit Underflow This bit indicates that the Transmit Buffer had an Underflow during frame transmission. Transmission is suspended and an Underflow Error TDES0[1] is set.
4	OVF	R/C		Receive Overflow This bit indicates that the Receive Buffer had an Overflow during frame reception. If the partial frame is transferred to application, the overflow status is set in RDES0[11].
3	TJT	R/C		Transmit Jabber Timeout

				This bit indicates that the Transmit Jabber Timer expired, meaning that the transmitter had been excessively active. The transmission process is aborted and placed in the Stopped state. This causes the Transmit Jabber Timeout TDES0[14] flag to assert.
2	TU	R/C		Transmit Buffer Unavailable This bit indicates that the Next Descriptor in the Transmit List is owned by the host and cannot be acquired by the DMA. Transmission is suspended. Bits[22:20] explain the Transmit Process state transitions. To resume processing transmit descriptors, the host should change the ownership of the bit of the descriptor and then issue a Transmit Poll Demand command.
1	TPS	R/C		Transmit Process Stopped This bit is set when the transmission is stopped.
0	TI	R/C		Transmit Interrupt This bit indicates that frame transmission is finished and TDES1[31] is set in the First Descriptor.

Operation Mode Register (OPMODE) 0xB0821018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-					DT	RSF	DFF	RFA[2]	RFD[2]	TSF	FTF	-			TTC[2]
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TTC[1:0]		ST	RFD		RFA		EFC	FEF	FUF	-	RTC		OSF	SR	-

The Operation Mode register establishes the Transmit and Receive operating modes and commands. Operation Mode Register should be the last CSR to be written as part of DMA initialization. This register is also present in the GMAC-MTL configuration with bits 24, 13, 2, and 1 unused and reserved.

Field	Name	RW	Reset	Description
26	DT	R/W	0x0	Disable Dropping of TCP/IP Checksum Error Frames When this bit is set, the core does not drop frames that only have errors detected by the Receive Checksum Offload engine. Such frames do not have any errors (including FCS error) in the Ethernet frame received by the MAC but have errors in the encapsulated payload only. When this bit is reset, all error frames are dropped if the FEF bit is reset. If the Full Checksum Offload engine (Type 2) is disabled, this bit is reserved (RO with value 1'b0).
25	RSF	R/W	0x0	Receive Store and Forward When this bit is set, the MTL only reads a frame from the Rx FIFO after the complete frame has been written to it, ignoring RTC bits. When this bit is reset, the Rx FIFO operates in Cut-Through mode, subject to the threshold specified by the RTC bits.
24	DFF	R/W	0x0	Disable Flushing of Received Frames When this bit is set, the RxDMA does not flush any frames due to the unavailability of receive descriptors/buffers as it does normally when this bit is reset. This bit is reserved (and RO) in GMAC-MTL configuration.
23	RFA[2]	R/W	0x0	MSB of Threshold for Activating Flow Control If the GMAC-UNIV is configured for an Rx FIFO depth of 8 KB or more, this bit (when set) provides additional threshold levels for activating the Flow Control in both Half- Duplex and Full-Duplex modes. This bit (as Most Significant Bit) along with the RFA (bits [10:9]) give the following thresholds for activating flow control. 100: Full minus 5 KB 101: Full minus 6 KB 110: Full minus 7 KB 111: Reserved This bit is reserved (and RO) if the Rx FIFO is 4 KB or less deep.
22	RFD[2]	R/W	0x0	MSB of Threshold for Deactivating Flow Control If the GMAC-UNIV is configured for an Rx FIFO depth of 8 KB or more, this bit (when set) provides additional threshold levels for deactivating the Flow Control in both Half- Duplex and Full-Duplex modes. This bit (as Most Significant Bit) along with the RFD (bits [12:11]) give the following thresholds for deactivating flow control. 100: Full minus 5 KB 101: Full minus 6 KB 110: Full minus 7 KB 111: Reserved This bit is reserved (and RO) if the Rx FIFO is 4 KB or less deep.
21	TSF	R/W	0x0	Transmit Store and Forward When this bit is set, transmission starts when a full frame resides in the MTL

				Transmit FIFO. When this bit is set, the TTC values specified in Operation Mode Register[16:14] are ignored. This bit should be changed only when transmission is stopped.
20	FTF	R/W	0x0	Flush Transmit FIFO When this bit is set, the transmit FIFO controller logic is reset to its default values and thus all data in the Tx FIFO is lost/flushed. This bit is cleared internally when the flushing operation is completed fully. The Operation Mode register should not be written to until this bit is cleared.
16-14	TTC	R/W	0x0	Transmit Threshold Control These three bits control the threshold level of the MTL Transmit FIFO. Transmission starts when the frame size within the MTL Transmit FIFO is larger than the threshold. In addition, full frames with a length less than the threshold are also transmitted. These bits are used only when the TSF bit (Bit 21) is reset 000: 64 001: 128 010: 192 011: 256 100: 40 101: 32 110: 24 111: 16
13	ST	R/W	0X0	Start/Stop Transmission Command When this bit is set, transmission is placed in the Running state, and the DMA checks the Transmit List at the current position for a frame to be transmitted. Descriptor acquisition is attempted either from the current position in the list, which is the Transmit List Base Address set by Transmit Descriptor List Addreess Register, or from the position retained when transmission was stopped previously. If the current descriptor is not owned by the DMA, transmission enters the Suspended state and Transmit Buffer Unavailable (Status Register[2]) is set. The Start Transmission command is effective only when transmission is stopped. If the command is issued before setting DMA Transmit Descriptor List Address Register, then the DMA behavior is unpredictable. When this bit is reset, the transmission process is placed in the Stopped state after completing the transmission of the current frame. The Next Descriptor position in the Transmit List is saved, and becomes the current position when transmission is restarted. The stop transmission command is effective only the transmission of the current frame is complete or when the transmission is in the Suspended state.
12-11	RFD	R/W	0x0	Threshold for deactivating flow control (in both HD and FD) These bits control the threshold (Fill-level of Rx FIFO) at which the flow-control is deasserted after activation. 00: Full minus 1 KB 01: Full minus 2 KB 10: Full minus 3 KB 11: Full minus 4 KB Note that the deassertion is effective only after flow control is asserted. If the Rx FIFO is 8 KB or more, an additional bit (RFD[2]) is used for more threshold levels as described in bit [22].
10-9	RFA	R/W	0x0	Threshold for activating flow control (in both HD and FD) These bits control the threshold (Fill level of Rx FIFO) at which flow control is activated. 00: Full minus 1 KB 01: Full minus 2 KB 10: Full minus 3 KB 11: Full minus 4 KB Note that the above only applies to Rx FIFOs of 4 KB or more when the EFC bit is set high. If the Rx FIFO is 8 KB or more, an additional bit (RFA[2]) is used for more threshold levels as described in bit [23].
8	EFC	R/W	0x0	Enable HW flow control When this bit is set, the flow control signal operation based on fill-level of Rx FIFO is enabled. When reset, the flow control operation is disabled. This bit is not used (reserved and always reset) when the Rx FIFO is less than 4 KB.
7	FEF	R/W	0x0	Forward Error Frames When this bit is reset, the Rx FIFO drops frames with error status (CRC error, collision error, GMII_ER, giant frame, watchdog timeout, overflow). However, if the frame's start byte (write) pointer is already transferred to the read controller side (in Threshold mode), then the frames are not dropped. Note that in GMAC-

				MTL configuration in which the Frame Length FIFO is also enabled during coreKit configuration, the Rx FIFO drops the error frames if that frame's start byte is not transferred (output) on the ARI bus. When FEF is set, all frames except runt error frames are forwarded to the DMA.
6	FUF	R/W	0x0	Forward Undersized Good Frames When set, the Rx FIFO will forward Undersized frames (frames with no Error and length less than 64 bytes) including pad-bytes and CRC). When reset, the Rx FIFO will drop all frames of less than 64 bytes, unless it is already transferred due to lower value of Receive Threshold (e.g., RTC = 01).
4-3	RTC	R/W	0x0	Receive Threshold Control These two bits control the threshold level of the MTL Receive FIFO. Transfer (request) to DMA starts when the frame size within the MTL Receive FIFO is larger than the threshold. In addition, full frames with a length less than the threshold are transferred automatically. Note that value of 11 is not applicable if the configured Receive FIFO size is 128 bytes. These bits are valid only when the RSF bit is zero, and are ignored when the RSF bit is set to 1 00: 64 01: 32 10: 96 11: 128
2	OSF	R/W	0x0	Operate on Second Frame When this bit is set, this bit instructs the DMA to process a second frame of Transmit data even before status for first frame is obtained.
1	SR	R/W	0x0	Start/Stop Receive When this bit is set, the Receive process is placed in the Running state. The DMA attempts to acquire the descriptor from the Receive list and processes incoming frames. Descriptor acquisition is attempted from the current position in the list, which is the address set by DMA Receive Descriptor List Address Register or the position retained when the Receive process was previously stopped. If no descriptor is owned by the DMA, reception is suspended and Receive Buffer Unavailable (Status Register[7]) is set. The Start Receive command is effective only when reception has stopped. If the command was issued before setting DMA Receive Descriptor List Address Register, DMA behavior is unpredictable. When this bit is cleared, RxDMA operation is stopped after the transfer of the current frame. The next descriptor position in the Receive list is saved and becomes the current position after the Receive process is restarted. The Stop Receive command is effective only when the Receive process is in either the Running (waiting for receive packet) or in the Suspended state.

Interrupt Enable Register (IE) 0xB082101C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															NIE
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
AIE	ERE	FBE	-		ETE	RWE	RSE	RUE	RIE	UNE	OVE	THE	TUE	TSE	TIE

The Interrupt Enable register enables the interrupts reported by Status Register. Setting a bit to 1'b1 enables a corresponding interrupt. After a hardware or software reset, all interrupts are disabled.

Field	Name	RW	Reset	Description
16	NIE	R/W	0x0	Normal Interrupt Summary Enable When this bit is set, a normal interrupt is enabled. When this bit is reset, a normal interrupt is disabled. This bit enables the following bits Status Register[0]: Transmit Interrupt Status Register[2]: Transmit Buffer Unavailable Status Register[6]: Receive Interrupt Status Register[14]: Early Receive Interrupt
15	AIE	R/W	0x0	Abnormal Interrupt Summary Enable When this bit is set, an Abnormal Interrupt is enabled. When this bit is reset, an Abnormal Interrupt is disabled. This bit enables the following bits Status Register[1]: Transmit Process Stopped Status Register[3]: Transmit Jabber Timeout Status Register[4]: Receive Overflow Status Register[5]: Transmit Underflow Status Register[7]: Receive Buffer Unavailable Status Register[8]: Receive Process Stopped

				Status Register[9]: Receive Watchdog Timeout Status Register[10]: Early Transmit Interrupt Status Register[13]: Fatal Bus Error
14	ERE	R/W	0x0	Early Receive Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Interrupt Enable Register[16]), Early Receive Interrupt is enabled. When this bit is reset, Early Receive Interrupt is disabled.
13	FBE	R/W	0x0	Fatal Bus Error Enable When this bit is set with Abnormal Interrupt Summary Enable (Interrupt Enable Register[15]), the Fatal Bus Error Interrupt is enabled. When this bit is reset, Fatal Bus Error Enable Interrupt is disabled.
10	ETE	R/W	0x0	Early Transmit Interrupt Enable When this bit is set with an Abnormal Interrupt Summary Enable (Interrupt Enable Register[15]), Early Transmit Interrupt is enabled. When this bit is reset, Early Transmit Interrupt is disabled.
9	RWE	R/W	0x0	Receive Watchdog Timeout Enable When this bit is set with Abnormal Interrupt Summary Enable (Interrupt Enable Register[15]), the Receive Watchdog Timeout Interrupt is enabled. When this bit is reset, Receive Watchdog Timeout Interrupt is disabled.
8	RSE	R/W	0x0	Receive Stopped Enable When this bit is set with Abnormal Interrupt Summary Enable (Interrupt Enable Register[15]), Receive Stopped Interrupt is enabled. When this bit is reset, Receive Stopped Interrupt is disabled.
7	RUE	R/W	0x0	Receive Buffer Unavailable Enable When this bit is set with Abnormal Interrupt Summary Enable (Interrupt Enable Register[15]), Receive Buffer Unavailable Interrupt is enabled. When this bit is reset, the Receive Buffer Unavailable Interrupt is disabled.
6	RIE	R/W	0x0	Receive Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Interrupt Enable Register[16]), Receive Interrupt is enabled. When this bit is reset, Receive Interrupt is disabled.
5	UNE	R/W	0x0	Underflow Interrupt Enable When this bit is set with Abnormal Interrupt Summary Enable (Interrupt Enable Register[15]), Transmit Underflow Interrupt is enabled. When this bit is reset, Underflow Interrupt is disabled.
4	OVE	R/W	0x0	Overflow Interrupt Enable When this bit is set with Abnormal Interrupt Summary Enable (Interrupt Enable Register[15]), Receive Overflow Interrupt is enabled. When this bit is reset, Overflow Interrupt is disabled
3	TJE	R/W	0x0	Transmit Jabber Timeout Enable When this bit is set with Abnormal Interrupt Summary Enable (Interrupt Enable Register[15]), Transmit Jabber Timeout Interrupt is enabled. When this bit is reset, Transmit Jabber Timeout Interrupt is disabled.
2	TUE	R/W	0x0	Transmit Buffer Unavailable Enable When this bit is set with Normal Interrupt Summary Enable (Interrupt Enable Register[16]), Transmit Buffer Unavailable Interrupt is enabled. When this bit is reset, Transmit Buffer Unavailable Interrupt is disabled.
1	TSE	R/W	0x0	Transmit Stopped Enable When this bit is set with Abnormal Interrupt Summary Enable (Interrupt Enable Register[15]), Transmission Stopped Interrupt is enabled. When this bit is reset, Transmission Stopped Interrupt is disabled.
0	TIE	R/W	0x0	Transmit Interrupt Enable When this bit is set with Normal Interrupt Summary Enable (Interrupt Enable Register[16]), Transmit Interrupt is enabled. When this bit is reset, Transmit Interrupt is disabled.

Missed Frame and Buffer Overflow Counter Register (MFBOC) 0xB0821020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-			OFF	NFMA											OFMF
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NFMH															

The DMA maintains two counters to track the number of missed frames during reception. This register reports the current value of the counter. The counter is used for diagnostic purposes. Bits[15:0] indicate missed frames due to the host buffer being unavailable. Bits[27:17] indicate missed frames due to buffer overflow conditions (MTL and GMAC) and runt frames (good frames of less than 64 bytes) dropped by the MTL.

Field	Name	RW	Reset	Description
28	OFF	R	-	Overflow bit for FIFO Overflow Counter
27-17	NFMA	R		Indicates the number of frames missed by the application. This counter is incremented each time the MTL asserts the sideband signal mtl_rxoverflow_o. The counter is cleared when this register is read with mci_be_i[2] at 1'b1.
16	OFMF	R	-	Overflow bit for Missed Frame Counter
15-0	NFMH	R		Indicates the number of frames missed by the controller due to the Host Receive Buffer being unavailable. This counter is incremented each time the DMA discards an incoming frame. The counter is cleared when this register is read with mci_be_i[0] at 1'b1.

Current Host Transmit Descriptor Register (CHTD) 0xB0821048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CHTD															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHTD															

The Current Host Transmit Descriptor register points to the start address of the current Transmit Descriptor read by the DMA.

Field	Name	RW	Reset	Description
31-0	CHTD	R	-	Host Transmit Descriptor Address Pointer Cleared on Reset. Pointer updated by DMA during operation.

Current Host Receive Descriptor Register (CHRD) 0xB082104C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CHRD															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHRD															

The Current Host Receive Descriptor register points to the start address of the current Receive Descriptor read by the DMA.

Field	Name	RW	Reset	Description
31-0	CHRD	R	-	Host Receive Descriptor Address Pointer Cleared on Reset. Pointer updated by DMA during operation.

Current Host Transmit Buffer Address Register (CHTBA) 0xB0821050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CHTBA															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHTBA															

The Current Host Transmit Buffer Address register points to the current Transmit Buffer Address being read by the DMA.

Field	Name	RW	Reset	Description
31-0	CHTBA	R	-	Host Transmit Buffer Address Pointer Cleared on Reset. Pointer updated by DMA during operation.

Current Host Receive Buffer Address Register (CHRBA) 0xB0821054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CHRBA															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CHRBA															

The Current Host Receive Buffer Address register points to the current Receive Buffer address being read by the DMA.

Field	Name	RW	Reset	Description
31-0	CHRBA	R	-	Host Receive Buffer Address Pointer Cleared on Reset. Pointer updated by DMA during operation.

6.4.2 GMAC Registers

MAC Configuration Register (MACC) 0xB0820000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-							TC	WD	JD	BE	JE	IFG			DCRS
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PS	FES	DO	LM	DM	IPC	DR	LUD	ACS	BL		DC	TE	RE	-	

The MAC Configuration register establishes receive and transmit operating modes.

Field	Name	RW	Reset	Description
24	TC	R/W	0x0	Transmit Configuration in RGMII/SGMII When set, this bit enables the transmission of duplex mode, link speed, and link up/down information to the PHY in the RGMII/SGMII ports. The details of this feature are explained in “Reduced Gigabit Media Independent Interface” on page 128 and “Serial Gigabit Media Independent Interface” on page 135. When this bit is reset, no such information is driven to the PHY. This bit is reserved (and RO) if neither RGMII nor SGMII PHY port is selected during core configuration.
23	WD	R/W	0x0	Watchdog Disable When this bit is set, the GMAC disables the watchdog timer on the receiver, and can receive frames of up to 16,384 bytes. When this bit is reset, the GMAC allows no more than 2,048 bytes (10,240 if JE is set high) of the frame being received and cuts off any bytes received after that.
22	JD	R/W	0x0	Jabber Disable When this bit is set, the GMAC disables the jabber timer on the transmitter, and can transfer frames of up to 16,384 bytes. When this bit is reset, the GMAC cuts off the transmitter if the application sends out more than 2,048 bytes of data (10,240 if JE is set high) during transmission.
21	BE	R/W	0x0	Frame Burst Enable When this bit is set, the GMAC allows frame bursting during transmission in GMII Half- Duplex mode. This bit is reserved (and RO) in 10/100 Mbps only or Full-Duplex-only configurations
20	JE	R/W	0x0	Jumbo Frame Enable When this bit is set, GMAC allows Jumbo frames of 9,018 bytes (9,022 bytes for VLAN tagged frames) without reporting a giant frame error in the receive frame status.
19-17	IFG	R/W	0x0	Inter-Frame Gap These bits control the minimum IFG between frames during transmission. 000: 96 bit times 001: 88 bit times 010: 80 bit times …. 111: 40 bit times Note that in Half-Duplex mode, the minimum IFG can be configured for 64 bit times (IFG = 100) only. Lower values are not considered. In 1000-Mbps mode, the minimum IFG supported is 64 bit times (and above) in the GMAC-CORE configuration and 80 bit times (and above) in other system configurations.
16	DCRS	R/W	0x0	Disable Carrier Sense During Transmission When set high, this bit makes the MAC transmitter ignore the (G)MII CRS signal during frame transmission in Half-Duplex mode. This request results in no errors generated due to Loss of Carrier or No Carrier during such transmission. When this bit is low, the MAC transmitter generates such errors due to Carrier Sense and will even abort the transmissions. This bit is reserved (and RO) when the core is selected for Full-Duplex-only operation during core configuration.

15	PS	R/W	0x0	Disable Carrier Sense During Transmission When set high, this bit makes the MAC transmitter ignore the (G)MII CRS signal during frame transmission in Half-Duplex mode. This request results in no errors generated due to Loss of Carrier or No Carrier during such transmission. When this bit is low, the MAC transmitter generates such errors due to Carrier Sense and will even abort the transmissions. This bit is reserved (and RO) when the core is selected for Full-Duplex-only operation during core configuration.
14	FES	R/W	0x0	Speed Indicates the speed in Fast Ethernet (MII) mode: 0: 10 Mbps 1: 100 Mbps This bit is reserved (RO) by default and is enabled only when RMII/RGMII/SGMII is enabled during configuration. This bit generates link speed encoding when TC (Bit 24) is set in RGMII/SGMII mode. This signal can optionally be driven as an output signal (mac_speed_o[0]) if the core is configured with an RMII, SGMII, or RGMII PHY interface selected.
13	DO	R/W	0x0	Disable Receive Own When this bit is set, the GMAC disables the reception of frames when the gmii_txen_o is asserted in Half-Duplex mode. When this bit is reset, the GMAC receives all packets that are given by the PHY while transmitting. This bit is not applicable if the GMAC is operating in Full-Duplex mode. This bit is reserved (RO with default value) if the GMAC is configured for Full-Duplex-only operation during configuration.
12	LM	R/W	0x0	Loopback Mode When this bit is set, the GMAC operates in loopback mode at GMII/MII. The (G)MII Receive clock input (clk_rx_i) is required for the loopback to work properly, as the Transmit clock is not looped-back internally.
11	DM	R/W	0x0	Duplex Mode When this bit is set, the GMAC operates in a Full-Duplex mode where it can transmit and receive simultaneously. This bit is RO with default value of 1'b1 in Full-Duplex-only configuration.
10	IPC	R/W	0x0	Checksum Offload When this bit is set, the GMAC calculates the 16-bit one's complement of the one's complement sum of all received Ethernet frame payloads. It also checks whether the IPv4 Header checksum (assumed to be bytes 25–26 or 29–30 (VLAN-tagged) of the received Ethernet frame) is correct for the received frame and gives the status in the receive status word. The GMAC core also appends the 16-bit checksum calculated for the IP header datagram payload (bytes after the IPv4 header) and appends it to the Ethernet frame transferred to the application (when Type 2 COE is deselected). When this bit is reset, this function is disabled. When Type 2 COE is selected, this bit, when set, enables IPv4 checksum checking for received frame payloads' TCP/UDP/ICMP headers. When this bit is reset, the COE function in the receiver is disabled and the corresponding PCE and IP HCE status bits are always cleared. If the IP Checksum Offload feature is not enabled during coreKit configuration, this bit is reserved (RO with default value).
9	DR	R/W	0x0	Disable Retry When this bit is set, the GMAC will attempt only 1 transmission. When a collision occurs on the GMII/MII, the GMAC will ignore the current frame transmission and report a Frame Abort with excessive collision error in the transmit frame status. When this bit is reset, the GMAC will attempt retries based on the settings of BL. This bit is applicable only to Half-Duplex mode and is reserved (RO with default value) in Full- Duplex-only configuration.
8	LUD	R/W	0x0	Link Up/Down Indicates whether the link is up or down during the transmission of configuration in RGMII/SGMII interface: 0: Link Down 1: Link Up This bit is reserved (RO with default value) and is enabled when RGMII/SGMII is enabled during configuration.
7	ACS	R/W	0x0	Automatic Pad/CRC Stripping When this bit is set, the GMAC strips the Pad/FCS field on incoming frames only if the length's field value is less than or equal to 1,500 bytes. All received frames with length field greater than or equal to 1,501 bytes are passed to the application without stripping the Pad/FCS field. When this bit is reset, the GMAC will pass all incoming frames to the Host unmodified.
6-5	BL	R/W	0x0	Back-Off Limit The Back-Off limit determines the random integer number (r) of slot time delays (4,096 bit times for 1000 Mbps and 512 bit times for 10/100 Mbps) the GMAC waits before rescheduling a transmission attempt during retries

				after a collision. This bit is applicable only to Half-Duplex mode and is reserved (RO) in Full-Duplex-only configuration. 00: k = min (*n*, 10) 01: k = min (*n*, 8) 10: k = min (*n*, 4) 11: k = min (*n*, 1), where *n* = retransmission attempt. The random integer *r* takes the value in the range 0 ≤ *r* < 2^k
4	DC	R/W	0x0	Deferral Check When this bit is set, the deferral check function is enabled in the GMAC. The GMAC will issue a Frame Abort status, along with the excessive deferral error bit set in the transmit frame status when the transmit state machine is deferred for more than 24,288 bit times in 10/100-Mbps mode. If the Core is configured for 1000 Mbps operation, or if the Jumbo frame mode is enabled in 10/100-Mbps mode, the threshold for deferral is 155,680 bits times. Deferral begins when the transmitter is ready to transmit, but is prevented because of an active CRS (carrier sense) signal on the GMII/MII. Defer time is not cumulative. If the transmitter defers for 10,000 bit times, then transmits, collides, backs off, and then has to defer again after completion of back-off, the deferral timer resets to 0 and restarts. When this bit is reset, the deferral check function is disabled and the GMAC defers until the CRS signal goes inactive. This bit is applicable only in Half-Duplex mode and is reserved (RO) in Full-Duplex-only configuration.
3	TE	R/W	0x0	Transmitter Enable When this bit is set, the transmit state machine of the GMAC is enabled for transmission on the GMII/MII. When this bit is reset, the GMAC transmit state machine is disabled after the completion of the transmission of the current frame, and will not transmit any further frames.
2	RE	R/W	0x0	Receiver Enable When this bit is set, the receiver state machine of the GMAC is enabled for receiving frames from the GMII/MII. When this bit is reset, the GMAC receive state machine is disabled after the completion of the reception of the current frame, and will not receive any further frames from the GMII/MII.

MAC Frame Filter Register (MACFF) 0xB0820004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					HPF	SAF	SAIF	PCF		DBF	PM	DAIF	HMC	HUC	PR

The MAC Frame Filter register contains the filter controls for receiving frames. Some of the controls from this register go to the address check block of the MAC, which performs the first level of address filtering. The second level of filtering is performed on the incoming frame, based on other controls such as Pass Bad Frames and Pass Control Frames.

Field	Name	RW	Reset	Description
10	HPF	R/W	0x0	Hash or Perfect Filter When set, this bit configures the address filter to pass a frame if it matches either the perfect filtering or the hash filtering as set by HMC or HUC bits. When low and if the HUC/HMC bit is set, the frame is passed only if it matches the Hash filter. This bit is reserved (and RO) if the Hash filter is not selected during core configuration.
9	SAF	R/W	0x0	Source Address Filter Enable The GMAC core compares the SA field of the received frames with the values programmed in the enabled SA registers. If the comparison matches, then the SAMatch bit of RxStatus Word is set high. When this bit is set high and the SA filter fails, the GMAC drops the frame. When this bit is reset, then the GMAC Core forwards the received frame to the application and with the updated SA Match bit of the RxStatus depending on the SA address comparison.
8	SAIF	R/W	0x0	SA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the SA address comparison. The frames whose SA matches the SA registers will be marked as failing the SA Address filter. When this bit is reset, frames whose SA does not match the SA registers will be marked as failing the SA Address filter.
7-6	PCF	R/W	0x0	Pass Control Frames These bits control the forwarding of all control frames (including unicast and multicast PAUSE frames). Note that the processing of PAUSE control frames depends only on RFE of Flow Control Register[2] 0x: GMAC filters all control frames from reaching the application 10: GMAC forwards all control frames to application even if they fail the Address Filter 11: GMAC forwards control frames that pass the Address Filter.
5	DBF	R/W	0x0	Disable Broadcast Frames When this bit is set, the AFM module filters all incoming broadcast frames. When this bit is reset, the AFM module passes all received broadcast frames.
4	PM	R/W	0x0	Pass All Multicast When set, this bit indicates that all received frames with a multicast destination address (first bit in the destination address field is '1') are passed. When reset, filtering of multicast frame depends on HMC bit.
3	DAIF	R/W	0x0	DA Inverse Filtering When this bit is set, the Address Check block operates in inverse filtering mode for the DA address comparison for both unicast and multicast frames. When reset, normal filtering of frames is performed.
2	HMC	R/W	0x0	Hash Multicast When set, MAC performs destination address filtering of received multicast frames according to the hash table. When reset, the MAC performs a perfect destination address filtering for multicast frames, that is, it compares the DA field with the values programmed in DA registers. If Hash Filter is not selected during coreKit configuration, this bit is reserved (and RO).
1	HUC	R/W	0x0	Hash Unicast When set, MAC performs destination address filtering of unicast frames according to the hash table. When reset, the MAC performs a perfect destination address filtering for unicast frames, that is, it compares the DA field with the values programmed in DA registers. If Hash Filter is not selected during coreKit configuration, this bit is reserved (and RO).
0	PR	R/W	0x0	Promiscuous Mode When this bit is set, the Address Filter module passes all incoming frames regardless of its destination or source address. The SA/DA Filter Fails status bits of the Receive Status Word will always be cleared when PR is set.

Hash Table High Register (HTH) 0xB0820008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HTH															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HTH															

The 64-bit Hash table is used for group address filtering. For hash filtering, the contents of the destination address in the incoming frame is passed through the CRC logic, and the upper 6 bits of the CRC register are used to index the contents of the Hash table. The most significant bit determines the register to be used (Hash Table High/Hash Table Low), and the other 5 bits determine which bit within the register. A hash value of 5b'00000 selects Bit 0 of the selected register, and a value of 5b'11111 selects Bit 31 of the selected register. For example, if the DA of the incoming frame is received as 0x1F52419CB6AF (0x1F is the first byte received on GMII interface), then the internally calculated 6-bit Hash value is 0x2C and the HTH register bit[12] is checked for filtering. If the DA of the incoming frame is received as 0xA00A98000045, then the calculated 6- bit Hash value is 0x07 and the HTL register bit[7] is checked for filtering.
If the corresponding bit value of the register is 1’b1, the frame is accepted. Otherwise, it is rejected. If the PM (Pass All Multicast) bit is set in Mac Frame Filter Register, then all multicast frames are accepted regardless of the multicast hash values. If the Hash Table register is configured to be double-synchronized to the (G)MII clock domain, the synchronization is triggered only when Bits[31:24] (in Little-Endian mode) or Bits[7:0] (in Big-Endian mode) of the Hash Table High/Low registers are written to. Please note that consecutive writes to these register should be performed only after at least 4 clock cycles in the destination clock domain when double-synchronization is enabled. The Hash Table High register contains the higher 32 bits of the Hash table.

Field	Name	RW	Reset	Description
31-0	HTH	R/W	0x00000000	Hash Table High This field contains the upper 32 bits of Hash table.

Hash Table Low Register (HTL) 0xB082000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HTL															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HTL															

The Hash Table Low register contains the lower 32 bits of the Hash table. Both Hash Table High Register and Hash Table Low Register and corresponding HMC and HUC bits in Filter Register are reserved if the Hash Filter Function is disabled during coreKit configuration.

Field	Name	RW	Reset	Description
31-0	HTL	R/W	0x00000000	Hash Table Low This field contains the lower 32 bits of Hash table.

GMII Address Register (GMIIA) 0xB0820010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PA					GR					-	CR			GW	GB

The GMII Address register controls the management cycles to the external PHY through the management interface.

Field	Name	RW	Reset	Description
15-11	PA	R/W	0x00	Physical Layer Address This field tells which of the 32 possible PHY devices are being accessed.
10-6	GR	R/W	0x00	GMII Register These bits select the desired GMII register in the selected PHY device.
4-2	CR	R/W	0x0	CSR Clock Range The CSR Clock Range selection determines the clk_csr_i frequency and is used to decide the frequency of the MDC clock: **Selection clk_csr_i MDC Clock** 000 60-100 MHz clk_csr_i/42 001 100-150 MHz clk_csr_i/62 010 20-35 MHz clk_csr_i/16 011 35-60 MHz clk_csr_i/26 100 150-250 MHz clk_csr_i/102 101 250-300 MHz clk_csr_i/122 110, 111 Reserved
1	GW	R/W	0x0	GMII Write When set, this bit tells the PHY that this will be a Write operation using the GMII Data register. If this bit is not set, this will be a Read operation, placing the data in the GMII Data register.
0	GB	R/W	0x0	GMII Busy This bit should read a logic 0 before writing to GMII Address Register and GMII Data Register. This bit must also be set to 0 during a Write to GMII Address Register. During a PHY register access, this bit will be set to 1'b1 by the Application to indicate that a Read or Write access is in progress. GMII Data Register should be kept valid until this bit is cleared by the GMAC during a PHY Write operation. The GMII Data Register is invalid until this bit is cleared by the GMAC during a PHY Read operation. The GMII Address Register should not be written to until this bit is cleared.

GMII Data Register (GMIID) 0xB0820014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GMIID															

The GMII Data register stores Write data to be written to the PHY register located at the address specified in GMII Address Register. GMII Data Register also stores Read data from the PHY register located at the address specified by GMII Address Register.

Field	Name	RW	Reset	Description
15-0	GMIID	R/W	0x0000	GMII Data This contains the 16-bit data value read from the PHY after a Management Read operation or the 16-bit data value to be written to the PHY before a Management Write operation.

Flow Control Register (FC) 0xB0820018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PT															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								DZPQ	-	PLT		UP	RFE	TFE	FCB/ BPA

The Flow Control register controls the generation and reception of the Control (Pause Command) frames by the GMAC's Flow control module. A Write to a register with the Busy bit set to '1' triggers the Flow Control block to generate a Pause Control frame. The fields of the control frame are selected as specified in the 802.3x specification, and the Pause Time value from this register is used in the Pause Time field of the control frame. The Busy bit remains set until the control frame is transferred onto the cable. The Host must make sure that the Busy bit is cleared before writing to the register.

Field	Name	RW	Reset	Description
31-16	PT	R/W	0x0000	Pause Time This field holds the value to be used in the Pause Time field in the transmit control frame. If the Pause Time bits is configured to be double-synchronized to the (G)MII clock domain, then consecutive writes to this register should be performed only after at least 4 clock cycles in the destination clock domain.
7	DZPQ	R/W	0x0	Disable Zero-Quanta Pause When set, this bit disables the automatic generation of Zero-Quanta Pause Control frames on the deassertion of the flow-control signal from the FIFO layer (MTL or external sideband flow control signal sbd_flowctrl_i/mti_flowctrl_i). When this bit is reset, normal operation with automatic Zero-Quanta Pause Control frame generation is enabled.
5-4	PLT	R/W	0x0	Pause Low Threshold This field configures the threshold of the PAUSE timer at which the input flow control signal mti_flowctrl_i (or sbd_flowctrl_i) is checked for automatic retransmission of PAUSE Frame. The threshold values should be always less than the Pause Time configured in Bits[31:16]. For example, if PT = 100H (256 slot-times), and PLT = 01, then a second PAUSE frame is automatically transmitted if the mti_flowctrl_i signal is asserted at 228 (256 – 28) slot-times after the first PAUSE frame is transmitted. **Selection Threshold** 00 Pause time minus 4 slot times 01 Pause time minus 28 slot times 10 Pause time minus 144 slot times 11 Pause time minus 256 slot times Slot time is defined as time taken to transmit 512 bits (64 bytes) on the GMII/MII interface.
3	UP	R/W	0x0	Unicast Pause Frame Detect When this bit is set, the GMAC will detect the Pause frames with the station's unicast address specified in MAC Address0 High Register and MAC Address0 Low Register, in addition to the detecting Pause frames with the unique multicast address. When this bit is reset, the GMAC will detect only a Pause frame with the unique multicast address specified in the 802.3x standard.
2	RFE	R/W	0x0	Receive Flow Control Enable When this bit is set, the GMAC will decode the received Pause frame and disable its transmitter for a specified (Pause Time) time. When this bit is reset, the decode function of the Pause frame is disabled.
1	TFE	R/W	0x0	Transmit Flow Control Enable In Full-Duplex mode, when this bit is set, the GMAC enables the flow control operation to transmit Pause frames. When this bit is reset, the flow control operation in the GMAC is disabled, and the GMAC will not transmit any Pause frames. In Half-Duplex mode, when this bit is set, the GMAC enables the back-pressure operation. When this bit is reset, the backpressure feature is disabled.
0	FCB/BPA	R/W	0x0	Flow Control Busy/Backpressure Activate This bit initiates a Pause Control frame in Full-Duplex mode and activates the backpressure function in Half-Duplex mode if TFE bit is set. In Full-Duplex mode, this bit should be read as 1'b0 before writing to the Flow Control register. To initiate a Pause control frame, the Application must set this bit to 1'b1. During a transfer of the Control Frame, this bit will continue to be set to signify that a frame transmission is in progress. After the completion of Pause control frame transmission, the GMAC will reset this bit to 1'b0. The Flow Control register should not be written to until this bit is cleared. In Half-Duplex mode, when this bit is set (and TFE is set), then backpressure is

				asserted by the GMAC Core. During backpressure, when the GMAC receives a new frame, the transmitter starts sending a JAM pattern resulting in a collision. This control register bit is logically OR'ed with the mti_flowctrl_i input signal for the backpressure function. When the GMAC is configured to Full-Duplex mode, the BPA is automatically disabled.

VLAN Tag Register (VLANT) 0xB082001C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															ETV
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VL															

The VLAN Tag register contains the IEEE 802.1Q VLAN Tag to identify the VLAN frames. The MAC compares the 13th and 14th bytes of the receiving frame (Length/Type) with 16'h8100, and the following 2 bytes are compared with the VLAN tag; if a match occurs, it sets the received VLAN bit in the receive frame status. The legal length of the frame is increased from 1518 bytes to 1522 bytes. If the VLAN Tag register is configured to be double-synchronized to the (G)MII clock domain, then consecutive writes to these register should be performed only after at least 4 clock cycles in the destination clock domain.

Field	Name	RW	Reset	Description
16	ETV	R/W	0x0	Enable 12-Bit VLAN Tag Comparison When this bit is set, a 12-bit VLAN identifier, rather than the complete 16-bit VLAN tag, is used for comparison and filtering. Bits[11:0] of the VLAN tag are compared with the corresponding field in the received VLAN-tagged frame. When this bit is reset, all 16 bits of the received VLAN frame's fifteenth and sixteenth bytes are used for comparison.
15-0	VL	R/W	0x0000	VLAN Tag Identifier for Receive Frames This contains the 802.1Q VLAN tag to identify VLAN frames, and is compared to the fifteenth and sixteenth bytes of the frames being received for VLAN frames. Bits[15:13] are the User Priority, Bit[12] is the Canonical Format Indicator (CFI) and bits[11:0] are the VLAN tag's VLAN Identifier (VID) field. When the ETV bit is set, only the VID (Bits[11:0]) is used for comparison. If VL (VL[11:0] if ETV is set) is all zeros, the GMAC does not check the fifteenth and sixteenth bytes for VLAN tag comparison, and declares all frames with a Type field value of 0x8100 to be VLAN frames.

Version Register (VERSION) 0xB0820020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UDV								SV							

The Version register's contents identify the version of the core. This register contains two bytes, one of which Synopsys uses to identify the core release number, and the other of which you set during coreKit configuration.

Field	Name	RW	Reset	Description
15-8	UDV	R	-	User-defined version (configured with coreKit)
7-0	SV	R	0x34	Synopsys-defined version (3.4)

Interrupt Status Register (IRQS) 0xB0820038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-						TS	-	MMCC	MMCT	MMCR	MMCI	PMT	PANC	PLSC	RGMII

The Interrupt Status register contents identify the events in the GMAC-CORE that can generate interrupt. Note that all the interrupt events are generated only when the corresponding optional feature is selected during coreKit configuration and enabled during operation. Hence, these bits are reserved when the corresponding features is not present in the core.

Field	Name	RW	Reset	Description
9	TS	R/C	-	Time Stamp Interrupt Status This bit is set high when the system time value equals or exceeds the value specified in the Target Time High and Low registers. This bit is cleared when this register is read. This bit is only active when IEEE1588 time stamping is enabled without an external time stamp input selected. In all other modes, this bit is reserved.
7	MMCC	R		MMC Receive Checksum Offload Interrupt Status This bit is set high whenever an interrupt is generated in the MMC Receive Checksum Offload Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. This bit is only valid when the optional MMC module and Checksum Offload Engine (Type 2) are selected during configuration.
6	MMCT	R		MMC Transmit Interrupt Status This bit is set high whenever an interrupt is generated in the MMC Transmit Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. This bit is only valid when the optional MMC module is selected during configuration.
5	MMCR	R		MMC Receive Interrupt Status This bit is set high whenever an interrupt is generated in the MMC Receive Interrupt Register. This bit is cleared when all the bits in this interrupt register are cleared. This bit is only valid when the optional MMC module is selected during configuration.
4	MMCI	R		MMC Interrupt Status This bit is set high whenever any of bits 7:5 is set high and cleared only when all of these bits are low. This bit is valid only when the optional MMC module is selected during configuration.
3	PMT	R		PMT Interrupt Status This bit is set whenever a Magic packet or Wake-on-LAN frame is received in Power- Down mode This bit is cleared when both bits[6:5] are cleared due to a read operation to the PMT Control and Status register. This bit is valid only when the optional PMT module is selected during configuration
2	PANC	R		PCS Auto-Negotiation Complete This bit is set when the Auto-negotiation is completed in the TBI/RTBI/SGMII PHY interface. This bit is cleared when the user makes a read operation to the AN Status register. This bit is valid only when the optional TBI/RTBI/SGMII PHY interface is selected during configuration and operation
1	PLSC	R		PCS Link Status Changed This bit is set due to any change in Link Status in the TBI/RTBI/SGMII PHY interface. This bit is cleared when the user makes a read operation the AN Status register. This bit is valid only when the optional TBI/RTBI/SGMII PHY interface is selected during configuration and operation
0	RGMII	R		RGMII Interrupt Status This bit is set due to any change in value of the Link Status of RGMII interface (Bit 3 of RGMII Status Register in “RGMII Status Register”. This bit is cleared when the user makes a read operation the RGMII Status register. This bit is valid only when the optional RGMII PHY interface is selected during configuration and operation.

Interrupt Mask Register (IRQM) 0xB082003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-						TS	-					PMT	PANC	PLSC	RGMII

The Interrupt Mask Register bits enables the user to mask the interrupt signal due to the corresponding event in the Interrupt Status Register. The interrupt signal is sbd_intr_o in GMAC-AHB and GMAC-DMA configuration while the interrupt signal is mci_intr_o in the GMAC-MTL and GMAC-CORE configuration.

Field	Name	RW	Reset	Description
9	TS	R/W	0x0	Time Stamp Interrupt Mask When set, this bit disables time stamp interrupt generation. This bit is only active when IEEE1588 time stamping is enabled without an external time stamp input selected. In all other modes, this bit is reserved.
3	PMT	R/W	0x0	PMT Interrupt Mask This bit when set, will disable the assertion of the interrupt signal due to the setting of PMT Interrupt Status bit in Interrupt Register.
2	PANC	R/W	0x0	PCS AN Completion Interrupt Mask This bit when set, will disable the assertion of the interrupt signal due to the setting of PCS Auto-negotiation complete bit in Interrupt Register caused due to the completion of Auto-negotiation event.
1	PLSC	R/W	0x0	PCS Link Status Interrupt Mask This bit when set, will disable the assertion of the interrupt signal due to the setting of PCS Link-status changed bit in Interrupt Register caused due to change in link-status event.
0	RGMII	R/W	0x0	RGMII Interrupt Mask This bit when set, will disable the assertion of the interrupt signal due to the setting of RGMII Interrupt Status bit in Interrupt Register.

MAC Address0 High Register (MACA0H) 0xB0820040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MO	-														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A[47:32]															

The MAC Address0 High register holds the upper 16 bits of the 6-byte first MAC address of the station. Note that the first DA byte that is received on the (G)MII interface corresponds to the LS Byte (Bits [7:0]) of the MAC Address Low register. For example, if 0x112233445566 is received (0x11 is the first byte) on the (G)MII as the destination address, then the MacAddress0 Register [47:0] is compared with 0x665544332211. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in Little-Endian mode) or Bits[7:0] (in Big-Endian mode) of the MAC Address Low Register are written to. Please note that consecutive writes to this Address Low Register should be performed only after at least 4 clock cycles in the destination clock domain for proper synchronization updates.

Field	Name	RW	Reset	Description
31	MO	RO	0x1	Always 1.
15-0	A[47:32]	R/W	0xFFFF	MAC Address0 [47:32] This field contains the upper 16 bits (47:32) of the 6-byte first MAC address. This is used by the MAC for filtering for received frames and for inserting the MAC address in the Transmit Flow Control (PAUSE) Frames.

MAC Address0 Low Register (MACA0L) 0xB0820044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
A[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
A[15:0]															

The MAC Address0 Low register holds the lower 32 bits of the 6-byte first MAC address of the station.

Field	Name	RW	Reset	Description
31-0	A[31:0]	R/W	0xFFFFFFFF	MAC Address0 [31:0] This field contains the lower 32 bits of the 6-byte first MAC address. This is used by the MAC for filtering for received frames and for inserting the MAC address in the Transmit Flow Control (PAUSE) Frames

MAC Address1 High Register (MACA1H) 0xB0820048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AE	SA	MBC						-							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
A[47:32]															

The MAC Address1 High register holds the upper 16 bits of the 6-byte second MAC address of the station. If the MAC address registers are configured to be double-synchronized to the (G)MII clock domains, then the synchronization is triggered only when Bits[31:24] (in Little Endian mode) or Bits[7:0] (in Big Endian mode) of the MAC Address Low Register are written to. Consecutive writes to this Address Low Register must be performed only after at least 4 clock cycles in the destination clock domain for proper synchronization updates.

Field	Name	RW	Reset	Description
31	AE	R/W	0x0	Address Enable When this bit is set, the Address filter module uses the second MAC address for perfect filtering. When reset, the address filter module will ignore the address for filtering.
30	SA	R/W	0x0	Source Address When this bit is set, the MAC Address1[47:0] is used to compare with the SA fields of the received frame. When this bit is reset, the MAC Address1[47:0] is used to compare with the DA fields of the received frame.
29-24	MBC	R/W	0x00	Mask Byte Control These bits are mask control bits for comparison of each of the MAC Address bytes. When set high, the GMAC core does not compare the corresponding byte of received DA/SA with the contents of MAC Address1 registers. Each bit controls the masking of the bytes as follows Bit 29: MAC Address1 High Register[15:8] Bit 28: MAC Address1 High Register[7:0] Bit 27: MAC Address1 Low Register[31:24] … Bit 24: MAC Address1 Low Register[7:0]
15-0	A[47:32]	R/W	0xFFFF	MAC Address1 [47:32] This field contains the upper 16 bits (47:32) of the 6-byte second MAC address.

MAC Address 1 Low Register (MACA1L) 0xB082004C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
A[32:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A[15:0]															

The MAC Address1 Low register holds the lower 32 bits of the 6-byte second MAC address of the station.

Field	Name	RW	Reset	Description
31-0	A[31:0]	R/W	0xFFFFFFFF	MAC Address1 [31:0] This field contains the lower 32 bits of the 6-byte second MAC address. The content of this field is undefined until loaded by the Application after the initialization process.

Note *)

- The descriptions for registers MAC Address2 High Register through MAC Address31 High Register are the same as for the 18 MAC Address1 High Registe
- The descriptions for registers MAC Address2 Low Register through MAC Address31 Low Register are the same as for the Register 19 MAC Address1 Low Register.

RGMII Status Register (RGMIIS) 0xB08200D8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												LSTS	LSPD		LMOD

The RGMII Status register indicates the status signals received by the RGMII (whichever is selected at reset) from the PHY. This register is optional and is present only when the core is configured for RGMIII PHY interfaces.

Field	Name	RW	Reset	Description
3	LSTS	R		Link Status. Indicates whether the link is up (1'b1) or down (1'b0).
2-1	LSPD	R		Link Speed. Indicates the current speed of the link: 00: 2.5 MHz 01: 25 MHz 10: 125 MHz
0	LMOD	R		Link Mode. Indicates the current mode of operation of the link: 1'b0: Half-Duplex mode 1'b1: Full-Duplex mode

7 Cipher

7.1 Overview

For the security of the network, the cipher encrypts data or decrypts the encrypted data. AES, DES, and Multi2 are algorithms built in the cipher of NVS2310.

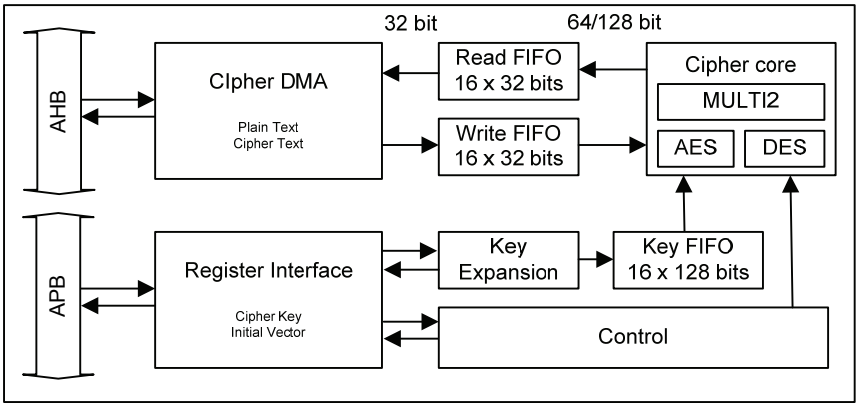


Figure 7.1 Cipher Engine Block Diagram

As for the basic operations of AES, DES, and MUTI2, a key is applied to a disclosed basic algorithm for encryption and decryption. At this time, the key is the same as the key value used in encryption or decryption. If encrypted data is decrypted with another key, a completely different data value is outputted.

Once a key value is defined and data is entered, AES, DES, and MULTI2 encrypts or decrypts this. The cipher of NVS2310 internally has DMA module and using this it uploads the data and stores the data. Since there is no other interface except for DMA, DMA setting is essential in order for the cipher to operate.

7.2 Feature

Specifically, AES supports 128/192/256 key mode and DES supports Single DES, Double DES and Triple DES. In addition, Triple DES supports 2 key mode and 3 key mode.

In overall, the cipher engine supports ECB(Electronic Codebook), CBC(Cipher Block Chaning), CFB(Cipher Feedback), OFB(Output Feedback), CTR(Counter) mode in AES, DES, and MULTI2.

7.3 Register Descriptions

Table 7.1 Cipher Register Map (Base Address = 0xB0870000)

Name	Offset	BITS	RW	Reset	Description
CTRL	0x00	32	R/W	0x00000000	Cipher control register
TXBASE	0x04	32	R/W	0x00000000	TX base address register
RXBASE	0x08	32	R/W	0x00000000	RX base address register
PACKET	0x0C	32	R/W	0x00000000	Packet register
DMACTR	0x10	32	R/W	0x00000000	DMA control register
DMASTR	0x14	32	R	-	DMA status register
IRQCTR	0x18	32	R/W	0x00000000	Interrupt control register
BLKNUM	0x1C	32	R	-	Block count register
ROUND	0x20	32	R/W	0x00000000	Round register
KEY0	0x40	32	R/W	0x00000000	Key0 register
KEY1	0x44	32	R/W	0x00000000	Key1 register
KEY2	0x48	32	R/W	0x00000000	Key2 register
KEY3	0x4C	32	R/W	0x00000000	Key3 register
KEY4	0x50	32	R/W	0x00000000	Key4 register
KEY5	0x54	32	R/W	0x00000000	Key5 register
KEY6	0x58	32	R/W	0x00000000	Key6 register
KEY7	0x5C	32	R/W	0x00000000	Key7 register
KEY8	0x60	32	R/W	0x00000000	Key8 register
KEY9	0x64	32	R/W	0x00000000	Key9 register
IV0	0x68	32	R/W	0x00000000	Initial vector0 register
IV1	0x6C	32	R/W	0x00000000	Initial vector1 register
IV2	0x70	32	R/W	0x00000000	Initial vector2 register
IV3	0x74	32	R/W	0x00000000	Initial vector3 register

7.3.1 Cipher Registers

Control Register (CTRL) **0xB0870000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WCLR	RCLR	BCLR	IVLD	KEYLD	KEYLEN		DESMODE		OPMODE			PRT	ENC	SELECT	

Field	Name	RW	Reset	Description
15	WCLR	W	0x0	Clear transmit FIFO counter '1' for clear
14	RCLR	W	0x0	Clear receive FIFO counter '1' for clear
13	BCLR	W	0x0	Clear Block counter '1' for clear
12	IVLD	W	0x0	Initial vector load '1' for load
11	KEYLD	W	0x0	Key data load '1' for load
10-9	KEYLEN	R/W	0x0	Select the key length in AES. 2,3: 256 bits key length 1: 192 bits key length 0: 128 bits key length
8-7	DESMODE	R/W	0x0	Select the mode in DES. 3: triple DES 3 key mode 2: triple DES 2 key mode 1: double DES mode 0: single DES mode
6-4	OPMODE	R/W	0x0	Select the operation mode. 4,5,6,7: counter(CTR) 3: Output Feedback(OFB) 2: Cipher Feedback(CFB) 1: Cipher Block Chaning(CBC) 0: Electronic Codebook(ECB)
3	PRT	R/W	0x0	Select the parity bit location of the key in DES. 1: MSB is a parity bit. 0: LSB is a parity bit.
2	ENC	R/W	0x0	Select Encryption or Description. 1: encryption 0: description
1-0	SELECT	R/W	0x0	Select the algorithm of the encryption. 2,3: multi2 1: DES 0: AES

TX Base Register (TXBASE) **0xB0870004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TXBASE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXBASE															

Field	Name	RW	Reset	Description
31-0	TXBASE	R/W	0x00000000	TX base address register

RX Base Register (RXBASE) 0xB0870008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXBASE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXBASE															

Field	Name	RW	Reset	Description
31-0	RXBASE	R/W	0x00000000	RX base address register

Packet Register (PACKET) 0xB087000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-			COUNT												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-			SIZE												

Field	Name	RW	Reset	Description
28-16	COUNT	R/W	0x0000	Packet numver information (COUNT + 1)
12-0	SIZE	R/W	0x0000	Pack size information

DMA Control Register (DMACTR) 0xB0870010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DE	-		END	-										TXAM	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXAM			-										PCLK	-	EN

Field	Name	RW	Reset	Description
31	DE	R/W	0x0	DMA request enable 1: enable 0: disable
28	END	R/W	0x0	Byte endian mode register 1: big-endian 0: little-endian
17-16	TXAM	R/W	0x0	TX addressing mode 2,3: Single packet 1: Fixed address(base) 0: Multiple Packet
15-14	RXAM	R/W	0x0	RX addressing mode 2,3: Single packet 1: Fixed address(base) 0: Multiple Packet
2	PCLK	W	0x0	Clear TX/RX packet counter ‘1’ for clear
0	EN	R/W	0x0	DMA enable register 1: enable 0: disable

DMA Status Register (DMASTS) 0xB0870014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-			RXPCNT												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-			TXPCNT												

Field	Name	RW	Reset	Description
28-16	RXPCNT	R	0x0000	Receive packet count register
12-0	TXPCNT	R	0x0000	Transmit packet count register

IRQ Control Register (IRQCTR) 0xB0870018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-		ISD	ISP	-							IRQS	-		IED	IEP
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-			IRQPCNT												

Field	Name	RW	Reset	Description
29	ISD	R/C	0x0	IRQ status for “Done interrupt” When the number of packets which DMA transfers/receives is equal to the number of packets specified in PACKET register, it is issued. When writing 1, it is cleared.
28	ISP	R/C	0x0	IRQ status for “Packet interrupt” It is issued every IRQPCNT packets which DMA transfers/receives When writing 1, it is cleared.
20	IRQS	R/W	0x0	IRQ select register 1: receiving 0: transmitting
17	IED	R/W	0x0	IRQ enable for “Done interrupt” 1: enable 0: disable
16	IEP	R/W	0x0	IRQ enable for “Packet interrupt” 1: enable 0: disable
12-0	IRQPCNT	R/W	0x0000	IRQ packet count register

Blocknum Count Register (BLKNUM) 0xB087001C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BLKNUM															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BLKNUM															

Field	Name	RW	Reset	Description
31-0	BLKNUM	RO	-	Check the number of the current block count.

Round Register (ROUND) 0xB0870020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ROUND															

Field	Name	RW	Reset	Description
15-0	ROUND	R/W	0x0000	Select Round in Multi2. (ROUND/8)

KEY0-9 Register (KEY0-9) 0xB0870040-0xB0870064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY															

Field	Name	RW	Reset	Description
31-0	KEY	R/W	0x00000000	Key register

There are 10 key registers and their key sizes vary depending on the selected encryption algorithm. If key setting is completed for these registers, 1 should be set for key load in the control register in order to generate a proper key corresponding to each algorithm. Key load is automatically cleared to 0 after one time key generation. The registers depending on the algorithms are as below.

Table 7.2 Key Map Depending on Cipher Engine

Cipher	mode	KEY0	KEY1	KEY2	KEY3	KEY4	KEY5	KEY6	KEY7	KEY8	KEY9
AES	128 bits key	KEY[127:0]				Not use					
	192 bits key	KEY[191:0]						Not use			
	256 bits key	KEY[255:0]								Not use	
DES	1 des	KEY[63:0]		Not use							
	2 des	KEY[127:0]				Not use					
	3 des 2key	KEY[127:0]				Not use					
	3 des 3key	KEY[191:0]						Not use			
Multi2	-	DATA KEY[63:0]		SYSTEM KEY[255:0]							

IV0-3 Register (IV0-3) 0xB0870068-0xB0870074

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IV															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IV															

There are four initial vector registers in total and the initial vector size varies from the selected encryption algorithm. Only in AES, all 4 registers are used. In DES and Multi2, only IV0 and IV1 are used. The initial vector registers are all used when OPMODE of control register is not ECB. Once iv setting is completed, iv load of control register should be set to 1 in order for the initial vector registers to operate properly.

Field	Name	RW	Reset	Description
31-0	IV	R/W	0x00000000	Initial vector register

8 HSIO BUS Configuration Registers

The HSIO BUS Configuration block has several registers named as SLEEP, SWRESET, MEM_PDN, MEM_SLN, ETHER_CFG, USBOTG, USBOTG_CFG0/1/2/3 and HSIO_A2X.

Table 8.1 HSIO BUS Configuration Register Map (Base Address = 0xB0880000)

Name	Address	Reset	Description
HCLKMASK0	0x00	0x00000000	Module Clock Mask Register 0
SWRESET0	0x04	0x00000000	Module Software Reset Register 0
Reserved	0x08	-	-
Reserved	0x0C	-	-
USBOTG	0x10	0x00000000	Refer to USB OTG Configuration Register (OTGCR) in "4.2 Register Description for USB 2.0 OTG Controller"USB OTG Configuration Register (OTGCR)USB OTG Configuration Register (OTGCR)USB OTG Configuration Register (OTGCR)USB OTG Configuration Register (OTGCR).
USBOTG_CFG0	0x14	0x00002930	Refer to USB PHY Configuration Register0 (UPCR0) in "4.2 Register Description for USB 2.0 OTG Controller"
USBOTG_CFG1	0x18	0x00007343	Refer to USB PHY Configuration Register1 (UPCR1) in "4.2 Register Description for USB 2.0 OTG Controller"
USBOTG_CFG2	0x1C	0x00005100	Refer to USB PHY Configuration Register3 (UPCR3) in "4.2 Register Description for USB 2.0 OTG Controller"
USBOTG_CFG3	0x20	0x00000005	Refer to USB PHY Configuration Register3 (UPCR3) in "4.2 Register Description for USB 2.0 OTG Controller
ETHER_CFG	0x30	0x00000000	ETHER CFG Register
HSIO_A2X	0x54	0x00000000	HSIOBUS AHB2AXI Control Register

HSIO BUS HCLK MASK Register 0(HCLKMASK0) **0xB0880000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-								HCLKMASK0							

Field	Name	RW	Reset	Description
31-8	-		0	Reserved
7-0	HCLKMASK0	R/W	0x0	Enable signals of HCLK clock gating logic AHB HCLKs, supplied to each block, are controlled by clock enable signal. 0: enable clock 1: disable clock The bit position indicates each sub-block which is controlled by Bit 6: CIPHER Bit 5: MSTICK Bit 4: GDMA Bit 3: USBOTG Bit 2: GMAC Bit 0: PWB (Prefetch/Write Buffer)

HSIO BUS SW RESET0(SWRESET0) 0xB0880004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-								SWRESET0							

Field	Name	RW	Reset	Description
31-8	-		0	Reserved
7-0	SWRESET0	R/W	0x0	Enable signals of SWRESET logic AHB SWRESETs, supplied to each block, are controlled by SWRESET enable signal. 0 : disable SWRESET 1 : enable SWRESET The bit position indicates each sub-block which is controlled by Bit 6: CIPHER Bit 5: MSTICK Bit 4: GDMA Bit 3: USBOTG Bit 2: GMAC Bit 0: PWB (Prefetch/Write Buffer)

ETHER CFG Register (ETHER_CFG) 0xB0880030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										TXCLK_SEL		PHY_INFSEL			FCTRL

Field	Name	RW	Reset	Description
31-6	-	-	-	Reserved
5-4	TXCLK_SEL	R/W	0x0	**TX clock selection** TXCLK_SEL[1] = 1 -> manual selection TXCLK_SEL[1] = 0 -> auto selection When manual selection TXCLK_SEL[0] = 1 -> use ckc clock TXCLK_SEL[0] = 0 -> use gmac phy clock
3-1	PHY_INFSEL	R/W	0x0	**PHY Interface Select** These pins select one of the GMAC's multiple PHY interfaces. This is sampled only during reset assertion (rst_clk_csr_n), and ignored after that. This input is removed from the port list when the core is configured for a single PHY interface. • 000: GMII/MII • 001: RGMII • All others: Reserved
0	FCTRL	R/W	0x0	**Sideband Flow Control** When set high, instructs the GMAC to transmit PAUSE Control frames in Full-duplex mode. In Half-duplex mode, the GMAC enables the back-pressure function until this signal is made low again.This signal is an optional port applicable to only GMAC-AHB, GMAC-DMA and GMAC-MTL configuration.

HSIO BUS AHB2AXI Control Register (HSIO_A2X) 0xB0880054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-										A2XMOD1			A2XMOD0		

Field	Name	RW	Reset	Description
31-6	-		0	Reserved
5-3	A2XMOD1	R/W	0x7	HSIO BUS to Memory Controller interface1 control register It affects read/write operation from following bus master to memory A2XMOD1[0] : flushs prefetch buffer when bus state is IDLE or WRITE. A2XMOD1[1] : stop reading prefetch buffer untill WRITE operation is completed. A2XMOD1[2] : not used
2-0	A2XMOD0	R/W	Ox7	HSIO BUS to Memory Controller interface0 control register It affects read/write operation from following bus master to memory A2XMOD0[0] : flushs prefetch buffer when bus state is IDLE or WRITE. A2XMOD0[1] : stop reading prefetch buffer untill WRITE operation is completed. A2XMOD0[2] : not used

PART7 – DISPLAY BUS
NVS2310

Rev. 1.02

Jun 01, 2011

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial Release
2011-04-22	1.01	∗ Correct the register table and document format. ∗ Correct LCLKDIV register description
2011-06-01	1.02	∗ Correct the mis-spelled chip name.

TABLE OF CONTENTS

Contents

Figures

Tables

1 Introduction

NVS2310 DDIBUS is a dedicated bus system for displaying or capturing video and image raw data. DDIBUS provides not only traditional and essential video interfaces such as LCD interface and NTSC/PAL encoder, bus also advanced features like HDMI, video/image scaler, and video quality enhancer as post processor for displaying device. Separated from main bus, DDIBUS provides connectivity between internal blocks with sufficient bandwidth.

1.1 Feature

- LCD Interface
  - Progressive or Interlaced Digital Video Output
  - Support of STN LCD
  - Support of CCIR-601/656
  - Support of TFT LCD
    - 16/18/24 bits output formats
  - 16-Level Image Overlay & Chroma-Keying OSD
  - Programmable Timing for Different Display Panels
  - Supports the Picture-In-Picture
  - Color Space Converter
  - Three 256-Level Color Look-Up Tables
  - Dual-LCD Controller
- Memory-to-memory Scaler
  - YUV4:2:2, YUV4:2:0 (8bits), RGB444, RGB454, RGB555, RGB565
  - Support of Direct Interface with LCD interface
  - Support Rolling Mode
- NTSC/PAL Encoder Composite Output
  - NTSC-M/4.43, PAL-B/D/G/H/I/M/N/Combination N
  - 10-bit DAC
- HDMI
  - HDMI 1.3, HDCP 1.1, DVI 1.0 Complaint
  - Supports Video format:
    480p @59.94Hz/60Hz, 576p@50Hz
    720p @50Hz/59.94Hz/60Hz
    1080i @50Hz/59.94Hz/60Hz
    1080p @50Hz/59.94Hz/60Hz
  - Other various formats up to 148.5 MHz Pixel Clock
  - Supports Color Format : 4:4:4 RGB/YCbCr , 4:2:2 YCbCr 12-bit mode
  - Pixel Repetition (Up to x4)
  - Supports Bit Per Color : 8bit, 10bit ,12bit (16bit is not supported)
  - Dedicated block for CEC function
  - Supports Audio Sample packets, DSD packets, HBR packets and DST packets for audio
  - Integrated HDCP Encryption Engine for Video/Audio content protection
  - Not include DDC (recommend to use separate I2C)
- Video/Image Quality Enhancer
  - De-interlacing
    - Edge-based/Motion-adaptive
    - File-mode processing
  - Noise Reduction
    - Spatio-temporal Engines
  - Edge Enhancing
  - Noise Measurement
  - Histogram Measurement
  - Contrast Enhancing
  - Color Gamut Mapping

2 Bus Architecture

Figure 2.1 shows internal hardware architecture of DDIBUS.

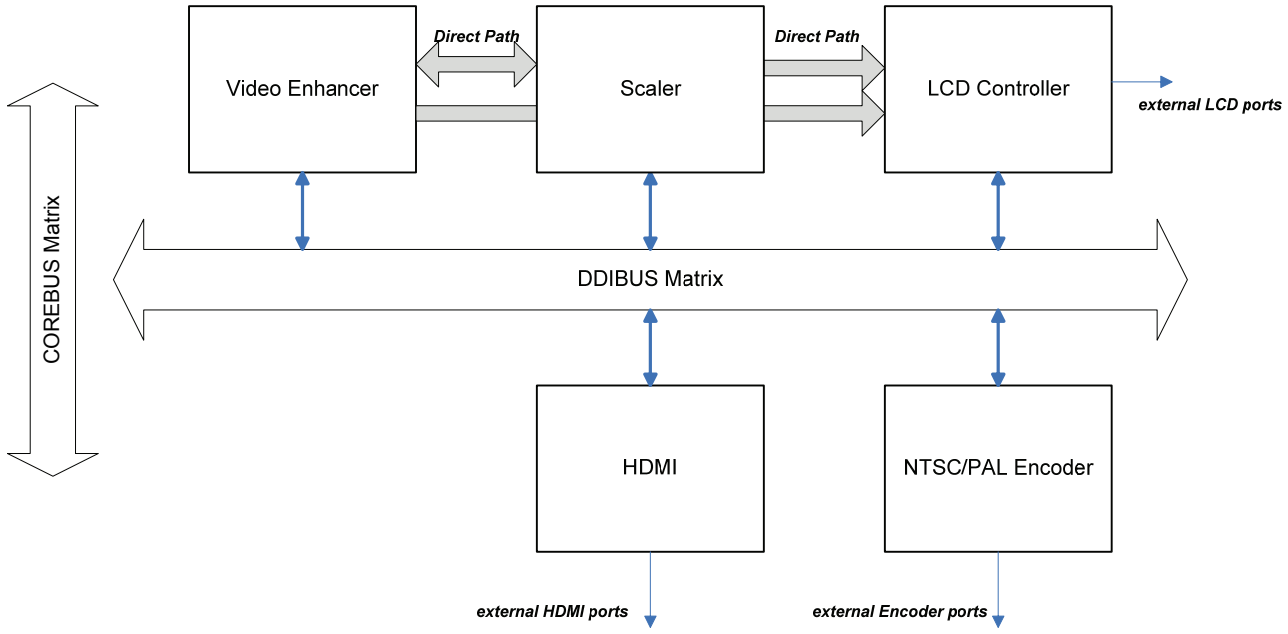


Figure 2.1 DDIBUS Hardware Architecture

3 Address and Register Map

The NVS2310 has various peripherals for specific video/image display/capture controllers or on-chip hardware components. These peripherals can be configured appropriately by its own registers that can be accessed through specially allocated address. These address maps are represented in the following table.

Refer to corresponding sections for detail information of each peripheral.

Base Address		Peripherals
0xB0A00000	0xB0A00000	LCD Controller 0
	0xB0A00400	LCD LUT 0
	0xB0A04000	LCD Controller 1
	0xB0A04400	LCD LUT 1
0xB0A10000		Scaler 0 (HPF)
0xB0A20000		Scaler 1
0xB0A40000	0xB0A40000	NTSC Encoder
	0xB0A40800	NTSC Encoder Controller
0xB0A50000		DDIC Controller
0xB0A51000		DDI Configuration
0xB0A52000		VIQE
0xB0A54000		HDMI Controller

4 LCD Controller

4.1 Overview

The LCD Controller (LCDC) is used to send out image data from system memory to LCD (RGB interface type), NTSC/PAL encoder, or HDMI. The LCDC provides all the necessary control signals to interface directly to mono STN, color STN, TFT panels, NTSC/PAL encoders, and HDMI. NVS2310 has 2 independent LCDCs. Both LCDC0 and LCDC1 can manipulate 3 input images.

The features of the LCDC are as follows.

supports Thin Film Transistor(TFT) color displays with 8-bit, 16-bit, 18-bit, 24-bit interface
supports STN displays with 4 or 8-bit interface
1, 2 or 4 bits per pixel(bpp) displays for mono STN
8(332) /16 bpp(444dummy) color displays for color STN
16, 18, 24 bpp true-color non-palettized color displays for color TFT
resolution programmable up to 4096 * 4096
programmable timing for different display panels
NTSC/PAL digital video encoder interface (CCIR601/656 interface)
Supports color lookup table.
Supports the overlay and alpha blending (2 overlay and 1 original image)
Supports the image up scaling (x2, x3, x4, x8)
Supports the Picture-In-Picture using the picture from the external device

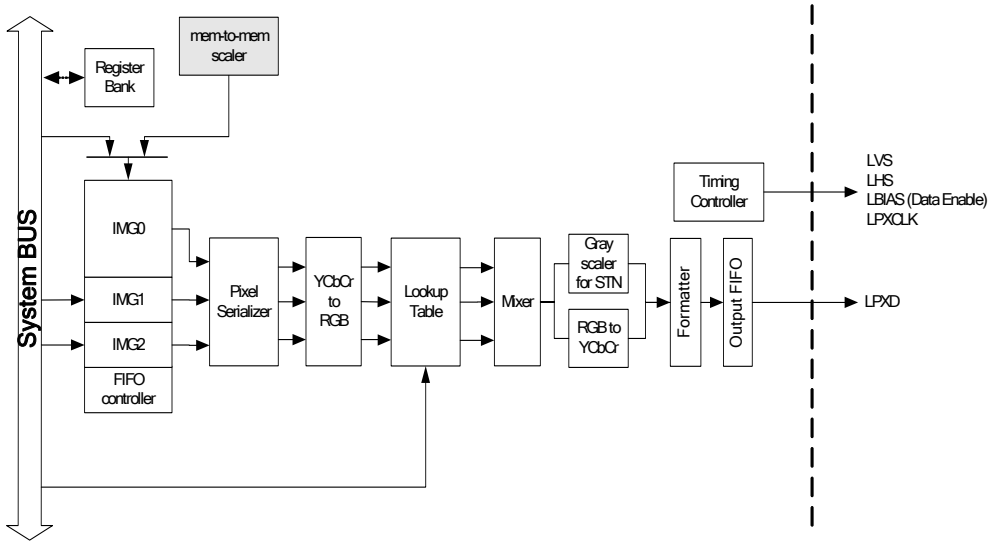


Figure 4.1 LCD Controller Block Diagram

The following key parameters can be programmed:

horizontal front and back porch
horizontal synchronization pulse width
number of panel clocks per line
vertical front and back porch
vertical synchronization pulse width
vertical synchronization pulse delay for STN mode
number of lines per panel
signal polarity
panel clock frequency
AC panel bias
bits-per-pixel
display type, STN mono/color or TFT
STN 4 or 8-bit interface mode
NTSC/PAL, Interlace/Non-interlace mode

HDMI, Interlace/Non-interlace mode
LVDS
Overlay/alpha blending mode
Image up scale ratio

The raw image sources stored in frame buffer are transferred to the LCDC's input FIFO, on a demand basis, using the AMBA AHB master interface.

The LCDC starts the DMA data transfer after it has been initialized and enabled. The DMA automatically performs burst word (64-bit) transfers, filling empty entries of the input FIFO. The data in the FIFO are fetched one entry at a time, and each 64-bit data is unpacked into appropriate pixel data formats (1, 2, 4, 8, or 16bpp) according to the raw image data format information. The frame buffer is in an off-chip memory area used to supply enough encoded pixel values to fill the entire screen one or more times. The pixel data buffer contains one encoded pixel values for each of the pixels present on the screen. The number of pixel data values depends on the size of the screen. The LCDC can fetch up to three raw image sources simultaneously and mixes them for making one frame image.

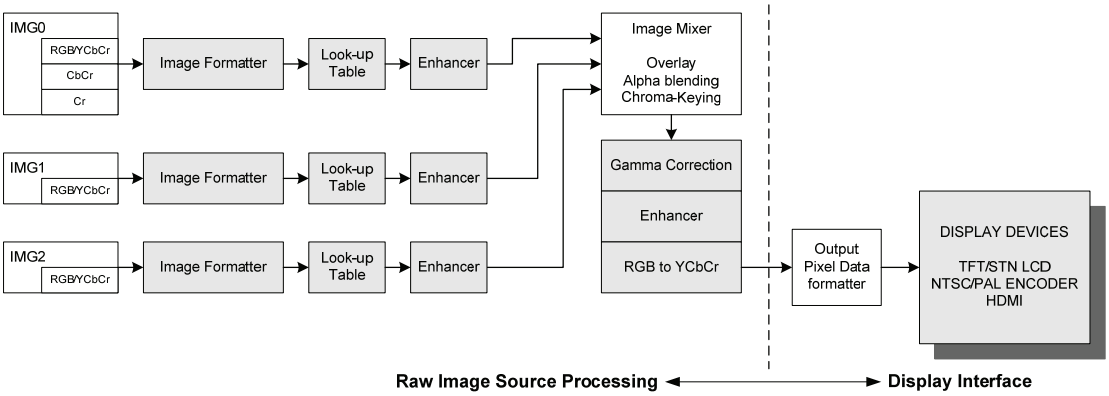


Figure 4.2 Overall Image Data Flow

4.2 Raw Image Source Processing

4.2.1 Image Formats and Relation Between Images and Layers

For processing the raw image sources, the LCDC defines a virtual display. It is shown in Figure 4.3. Display size, which consists of “Display Width” and “Display Height”, is determined by
LDS register. Actually, it becomes the active display size of the connected LCD or NTSC/PAL encoder.

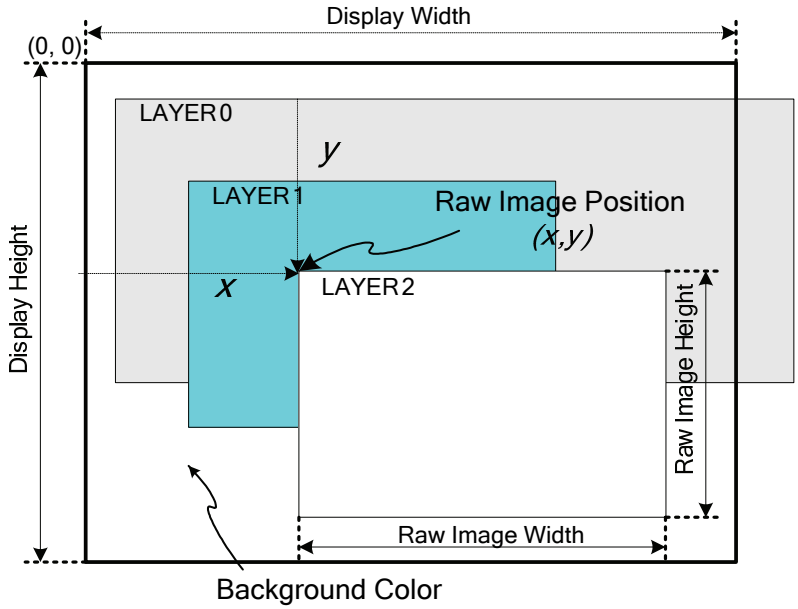


Figure 4.3 Virtual Display in the LCDC

The LCDC can have up to 3 raw image sources, which are named as Image 0, Image 1, and Image 2. IEN bits of each LI0C, LI1C, and LI2C register are used for enabling the corresponding raw image source. It can display simultaneously up to 3 layers, which are named as LAYER0, LAYER1, and LAYER2. LAYER2 is the top layer. Each layer is allocated for one of Image 0, Image 1, and Image 2. It is controlled by OP bits of LI0C register. Refer to the following table.

OP	LAYER2	LAYER1	LAYER0
0	Image 0	Image 1	Image 2
1	Image 0	Image 2	Image 1
2	Image 1	Image 0	Image 2
3	Image 2	Image 0	Image 1
4	Image 1	Image 2	Image 0
5*	Image 2	Image 1	Image 0

*) Default: OP = 5

Figure 4.4 Relationship between Layers and OP Bit

The area which is not overlapped by layers is filled with the background color. It is determined by the LBC register
The properties of a layer are determined by those of the corresponding raw image source; size, pixel data format, base address, position in the virtual display, etc.

BR=0	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
1BPP	P31	P30	P29	P28	P27	P26	P25	P24	P23	P22	P21	P20	P19	P18	P17	P16
2BPP	P15		P14		P13		P12		P11		P10		P9		P8	
4BPP	P7				P6				P5				P4			
8BPP	P3								P2							
BR=0	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
1BPP	P15	P14	P13	P12	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
2BPP	P7		P6		P5		P4		P3		P2		P1		P0	
4BPP	P3				P2				P1				P0			
8BPP	P1								P0							

(a) The bitmap for 1bpp, 2bpp, 4bpp, 8bpp when BR is ‘0’

BR=1	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1BPP	P24	P25	P26	P27	P28	P29	P30	P31	P16	P17	P18	P19	P20	P21	P22	P23
2BPP	P12		P13		P14		P15		P8		P9		P10		P11	
4BPP	P6				P7				P4				P5			
8BPP	P3								P2							
BR=0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1BPP	P8	P9	P10	P11	P12	P13	P14	P15	P0	P1	P2	P3	P4	P5	P6	P7
2BPP	P4		P5		P6		P7		P0		P1		P2		P3	
4BPP	P2				P3				P0				P1			
8BPP	P1								P0							

(b) The bitmap for 1bpp, 2bpp, 4bpp, 8bpp when BR is '1'

RGB	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGB332	R1			G1			B1		R0			G0			B0	
RGB444	A0				R0				G0				B0			
RGB565	R0					G0				B0						
RGB555	A0	R0				G0				B0						

(c) The bitmap for RGB332, RGB444, RGB565, and RGB555 formats.

RGB	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RGB666	IGNORED								A0						R0	
RGB888	A0								R0							
RGB	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGB666	R0				G0				B0							
RGB888	G0								B0							

(d) The bitmap for RGB666 and RGB888 formats.

Sequential	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
422SEQ	Cr								Y1							
Sequential	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
422SEQ	Cb								Y0							

(e) The bitmap for YCbCr 4:2:2 sequential format. The Cb and Cr affect to the Y0 and Y1.

Separated	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Y	Y1								Y0							
Cb	Cb1								Cb0							
Cr	Cr1								Cr0							

(f) The bitmap for YCbCr 4:2:0 and 4:2:2 separated format. The Cb0 and Cr0 affect to the Y0 and. The Cb1 and Cr1 affect to the Y2s and Y3s.

Interleaved	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Y	Y1								Y0							
C(type 0)	Cr0								Cb0							
C(type 1)	Cb0								Cr0							

(g) The bitmap for YCbCr 4:2:0 and 4:2:2 interleaved format.

Figure 4.5 Supported Pixel Data Format (BPP bits of LInC register)

The pixel-data formats supported in the LCDC are shown in Figure 4.5. They are determined by the LI*n*C registers (*n=0, 1, or 2*). LI0C, LI1C, and LI2C are for Image 0, Image 1, and Image 2 respectively. 1bpp, 2bpp, and 4bpp have no color information and can be arranged in big-endian pixel order as well as little-endian pixel order. Others have color information and should be arranged in little-endian pixel order only.

Image 1 and Image 2 do not support YCbCr4:2:0 and YCbCr4:2:2, but support "sequential YCbCr4:2:2". If the pixel data format of a raw image source is YCbCr4:2:0 or YCbCr4:2:2, it needs three base addresses, which are specified by the LI*n*BA0, LI*n*BA1, and LI*n*BA2 register. Otherwise, it needs one base address, which is specified by the LI*n*BA0 register. Thus, when the pixel data format of Image 0 is YCbCr4:2:0 or YCbCr4:2:2, the base address of Y, Cb, and Cr image should be set to LI0BA0, LI0BA1, and LI0BA2 respectively. But, because Image 1 and Image 2 do not support YCbCr4:2:0 and YCbCr4:2:2, there are only LI1BA0 and LI2BA0 register.

Though a raw image source is not RGB888 format in RGB color space, it is internally converted to RGB888 as appending additional bits. When this conversion is occurred, the PD bits of LI*n*C determine whether the additional bits are filled with zero or MSB. The difference between them is shown in Figure 4.6.

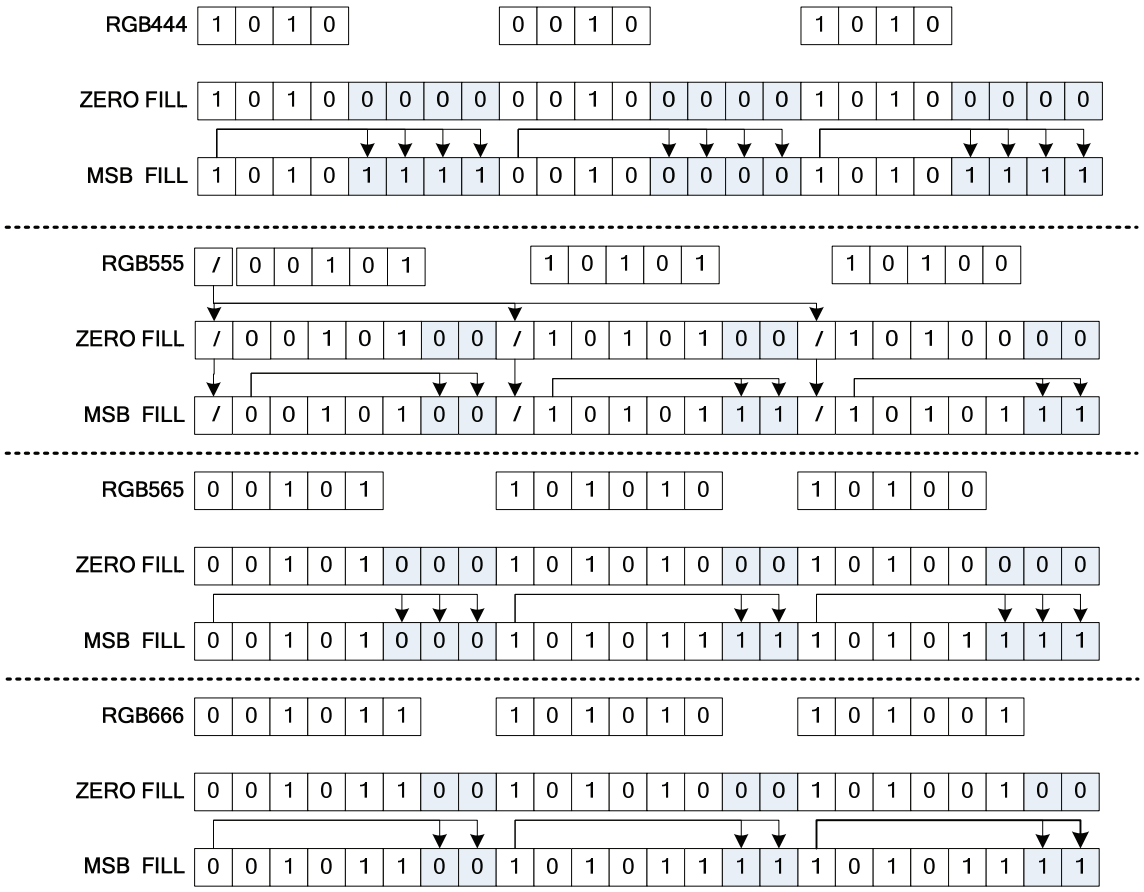


Figure 4.6 RGB2RGB888 Conversion

The origin of the virtual display is the top-left corner ((0, 0) in Figure 4.3) and “Raw Image Position” is the location of the top-left corner of a raw image within the virtual display. It is determined by the LI*n*P registers. “Raw Image Size”, which consists of “Raw Image Width” and “Raw Image Height”, is determined by the LI*n*S registers. Though the raw image sources are bigger than the size of the virtual display and/or the part of them exits the outside of the virtual display, the LCDC truncates an overlong part automatically.

And the LCDC has a simple scaler for each raw image source. It is controlled by LI*n*SC registers. But, to get the high-quality image, SCALER should be used instead of it.

Refer to Memory To Memory Scaler for more information about the image scaling.

4.2.2 Image Width, Height and Offset

The definition of frame window and image window is shown in the following figure. The frame window means that the raw image source and the image window is the window area on the frame window to be displayed. The “FP” stands for frame window pixel and “WP” stands for window pixel. The “FP(0,N)” is the pixel at the end of first line. All the pixels of the frame window should be on the linear and continuous address space.

The base address of LCD controller named as “LI*n*BA0” is the address of “WP(0,0)” and the image offset named as “LI*n*O” is the address difference between “FP(0,0)” and FP(1,0)”. ***The base address is byte-aligned and the image offset should be word-aligned”, the details are described in the explanation of the corresponding register.***

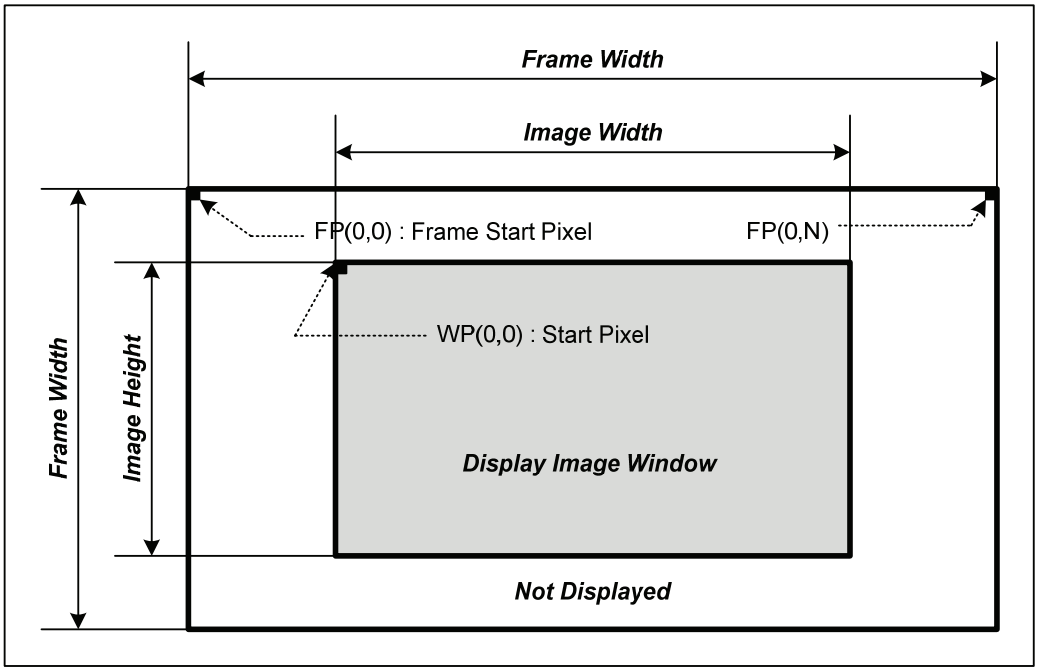


Figure 4.7 Definitions of Frame Window and Display Window

4.2.3 Conversion Between YCbCr Color Space and RGB Color Space

The internal operation in the LCD controller is designed for RGB color space. So, if the image source has the YCbCr color space, the source image should be converted to RGB color space. And if YCbCr color space is needed by display device, the internal graphic image should be converted to YCbCr color space.

The nominal ranges of “YCbCr” and “RGB” can be one of “0 ~ 255” or “16 ~ 235”. It is called “***Computer System Color***” and “***Studio Color***” for the 0 ~ 255 nominal range and 16 ~ 235 nominal range correspondingly.

The equations for color conversion can be dependent on the color types of source image and destination image.

4.2.4 Look-Up Table for Each Image Sources

The LCDC has the “24 bits x 256 entries” look-up table and it can be connected to one of Image 0, Image 1, and Image 2. Inputs of this table are RGB888 or YCbCr444 format as shown Figure 4.2. Each input pixel data addresses the table entry and the new pixel value is generated from the corresponding entry. Its entry consists of three 8-bit color values. The overall structure of Look-Up Table is shown in Figure 4.8. This table is controlled by LUT bits of the LIC2 register.

The table entries should be initialized before the LCDC uses it. Their address space is from 0xB0A00400 to 0xB0A007FC.

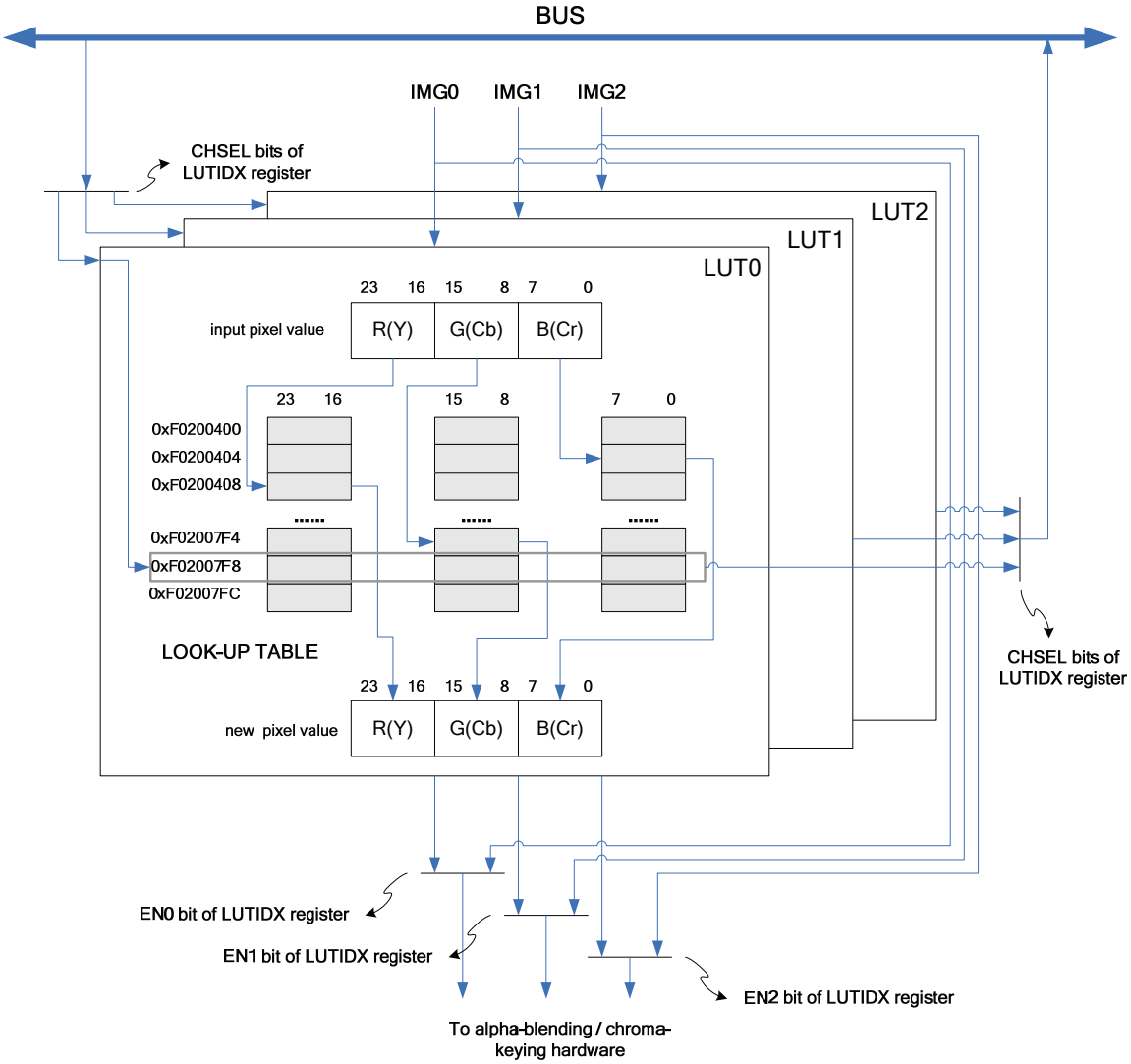


Figure 4.8 LCD Color Lookup Table

4.2.5 Alpha-blending with Multiple Layers

To process the overlapped region between layers, the LCDC supports the alpha-blending and the chroma-keying. When the alpha-blending and the chroma-keying are disabled, the top layer image is only shown in the overlapped region. Thus, LAYER1 is only shown in the overlapped region between the LAYER0 and the LAYER1.

The alpha-blending combines the overlay image (top layer) with the main image (bottom layer) according to the alpha value. The alpha value is specified by alpha bits in the pixel data of the overlay image and the alpha registers. The AEN2 bit of LCTRL register determines whether the LAYER2 enables the alpha-blending and the AEN1 bit determines whether LAYER1 enables the alpha-blending. Whether the alpha bits in the pixel data or alpha-registers are used as the alpha-value depends on the value of the SA1 bit of the LK1 register for the LAYER1 and the SA2 bit of the LK2 register for the LAYER2.

If the overlay image consists of the pixel data without the alpha value, you should use alpha-registers as the alpha-value (SA2 = 1 and/or SA1 = 1). The LAYER2 uses the A20 bits of LK2 register as the alpha-value and LAYER1 uses the A10 bits of LK2 register.

4.2.6 Chroma-Keying with Multiple Layers

Usually, a raw image source is the rectangular shape. Chroma-keying is used for clipping the arbitrary shape from a raw image source. In other words, chroma-keying disables the overlay process for selected pixels. They are identified by control registers, which are LK1, LK2. The LK1 is for the LAYER1 and the LK2 is for the LAYER2.

The overlay disabled pixel becomes the transparent pixel; the pixel of background layer is shown in the display.

For LAYER1, whether a pixel is overlaid or not depends on

$$\begin{aligned} OverlayDisabledPixel = ((&(Pixel(RorY)\ \&\ LKM1.MKR1) == (LK1.KR1\ \&\ LKM1.MKR1))\ \&\\ &((Pixel(GorU)\ \&\ LKM1.MKG1) == (LK1.KG1\ \&\ LKM1.MKG1))\ \&\\ &((Pixel(BorV)\ \&\ LKM1.MKB1) == (LK1.KB1\ \&\ LKM1.MKB1)) \end{aligned}$$

$$\begin{aligned} OverlayEnabledPixel = ((&(Pixel(RorY)\ \&\ LKM1.MKR1) != (LK1.KR1\ \&\ LKM1.MKR1))\ |\\ &((Pixel(GorU)\ \&\ LKM1.MKG1) != (LK1.KG1\ \&\ LKM1.MKG1))\ |\\ &((Pixel(BorV)\ \&\ LKM1.MKB1) != (LK1.KB1\ \&\ LKM1.MKB1)) \end{aligned}$$

For LAYER2, whether a pixel is overlaid or not depends on

$$\begin{aligned} OverlayDisabledPixel = ((&(Pixel(RorY)\ \&\ LKM2.MKR1) == (LK2.KR1\ \&\ LKM2.MKR1))\ \&\\ &((Pixel(GorU)\ \&\ LKM2.MKG1) == (LK2.KG1\ \&\ LKM2.MKG1))\ \&\\ &((Pixel(BorV)\ \&\ LKM2.MKB1) == (LK2.KB1\ \&\ LKM2.MKB1)) \end{aligned}$$

$$\begin{aligned} OverlayEnabledPixel = ((&(Pixel(RorY)\ \&\ LKM2.MKR1) != (LK2.KR1\ \&\ LKM2.MKR1))\ |\\ &((Pixel(GorU)\ \&\ LKM2.MKG1) != (LK2.KG1\ \&\ LKM2.MKG1))\ |\\ &((Pixel(BorV)\ \&\ LKM2.MKB1) != (LK2.KB1\ \&\ LKM2.MKB1)) \end{aligned}$$

4.2.7 Gamma Correction Function

Gamma Correction Function uses 16 correction values for each channel (GR0~15, GG0~15 and GB0~15). Each correction values are consisted of 4 bits and are represented two's complements format (value range: -8 ~ +7). Gamma correction function uses 17 correction points (16bands).

Nth Correction Point:
N<8 : ((256/N), (256/N)+LGR[N]*4)
N=8 : (32, 32+((LGR[7]+LGR[8])/2))
N>9 : (256/(N-1), (256/(N-1))+LGR[N-1]*4)

Gamma Correction Function is shown in Figure 4.9. Some examples are shown in Figure 4.10 and Figure 4.11.

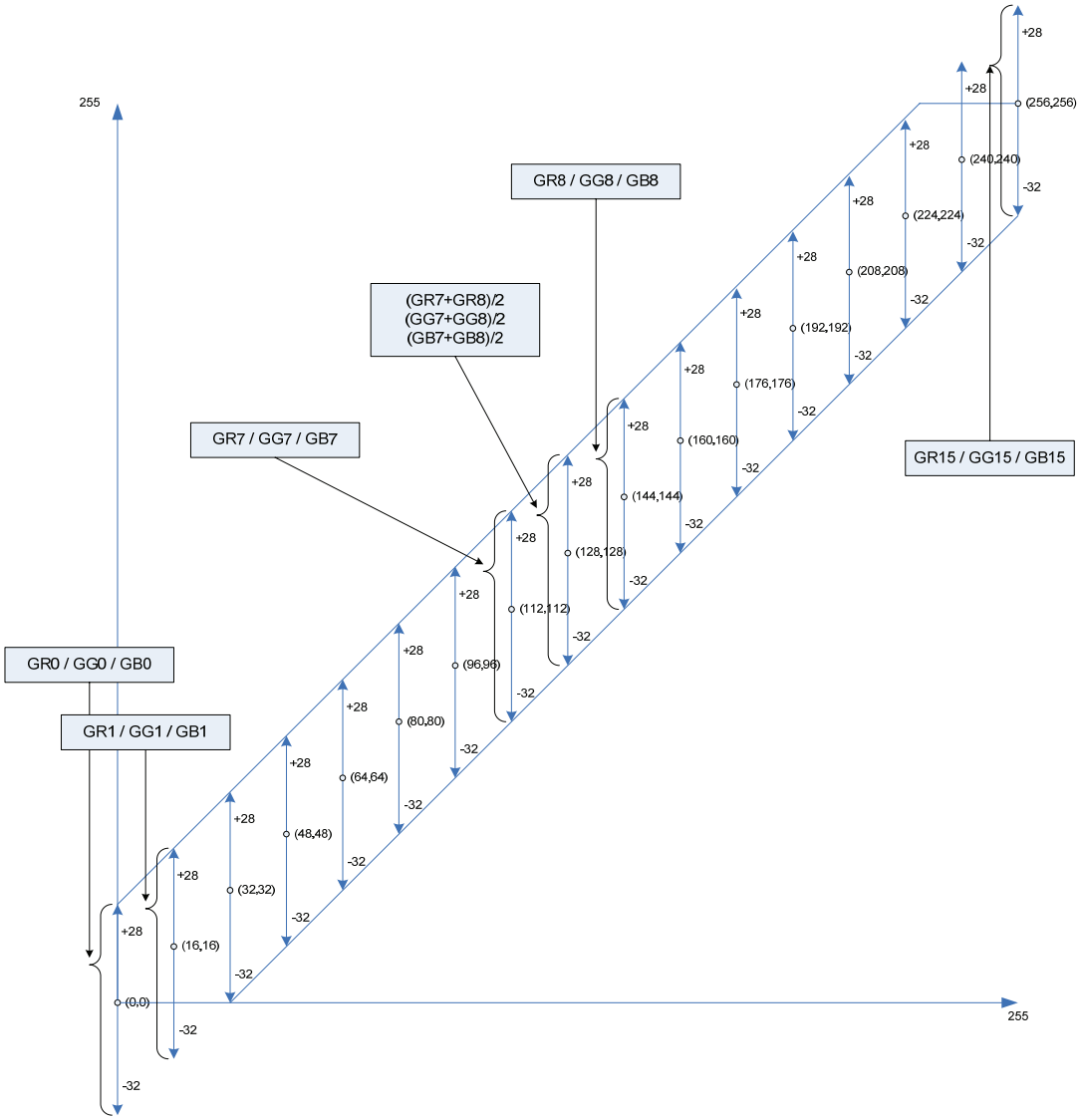


Figure 4.9 LCD Gamma Correction Function

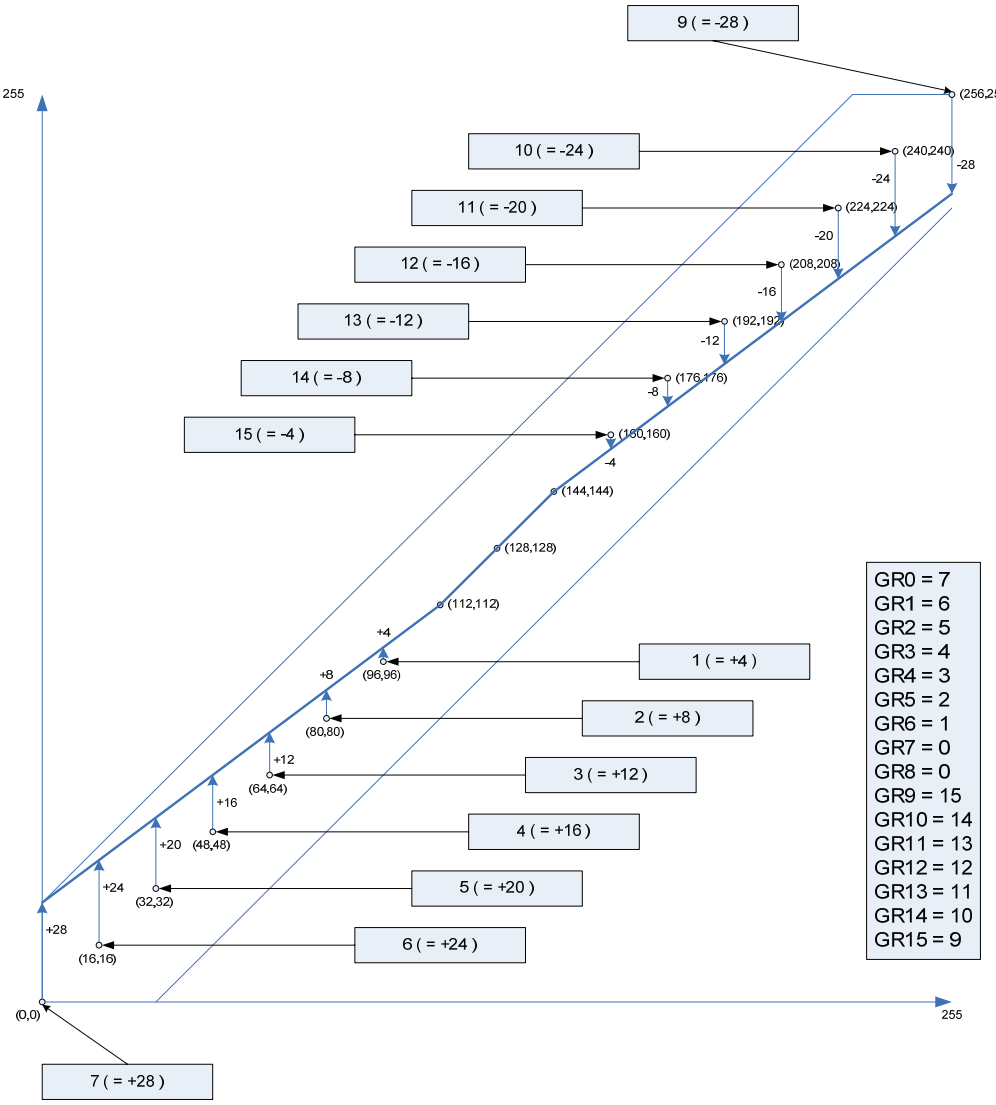


Figure 4.10 LCD Gamma Correction Function Example 1

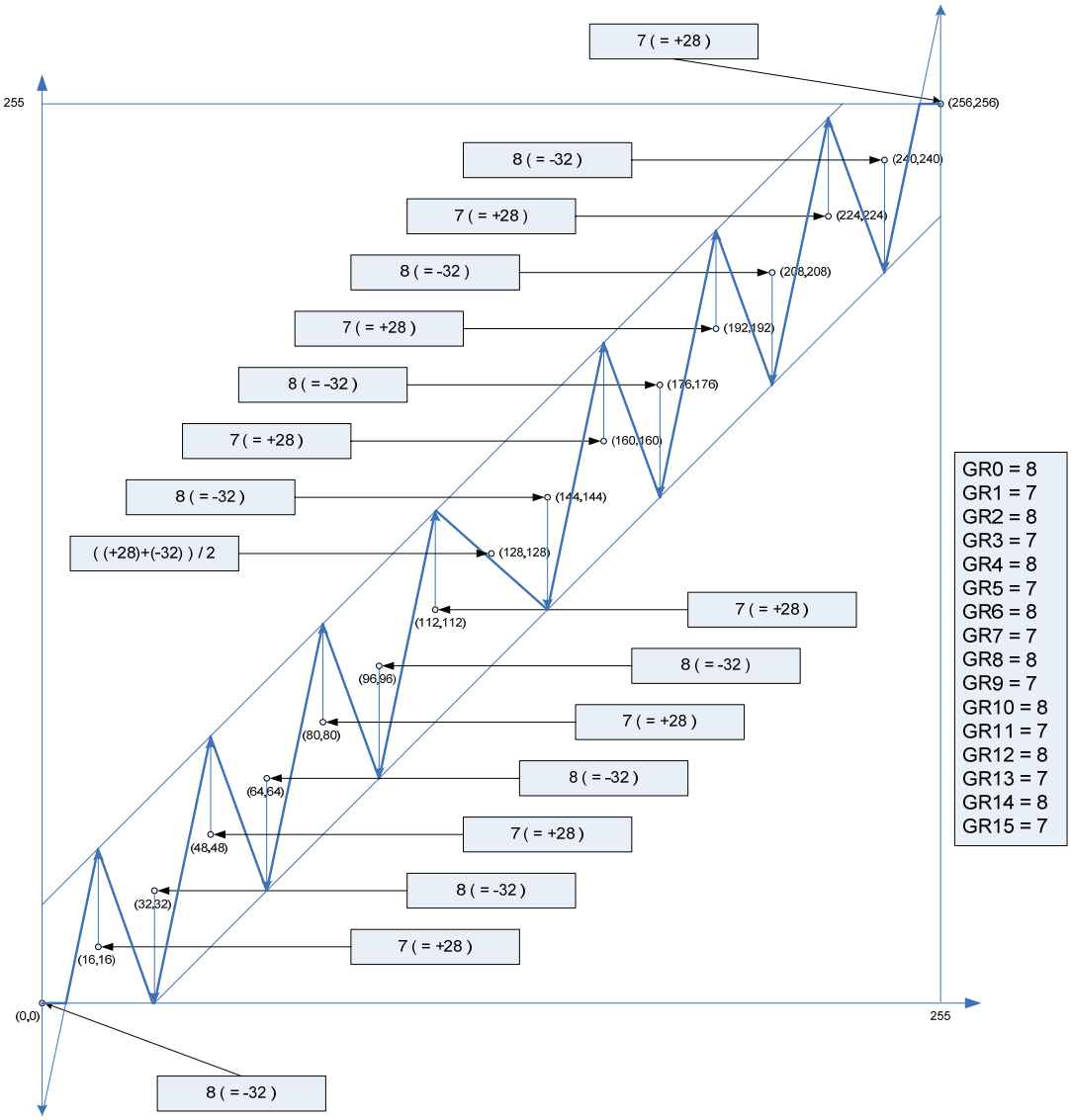


Figure 4.11 LCD Gamma Correction Function Example 2

4.2.8 Contrast and Brightness Adjust Function

Output_value = MAX(MIN((input_value – 128) x contrast + 128 + brightness, 255), 0)

4.2.9 RGB Dithering Function

In the case that LCD output format is RGB555, RGB565 or RGB666, RGB Ditherer can be used.
RGB Ditherer is based on “Pattern Dithering Algorithm” and pattern matrix can be configurable with register setting.

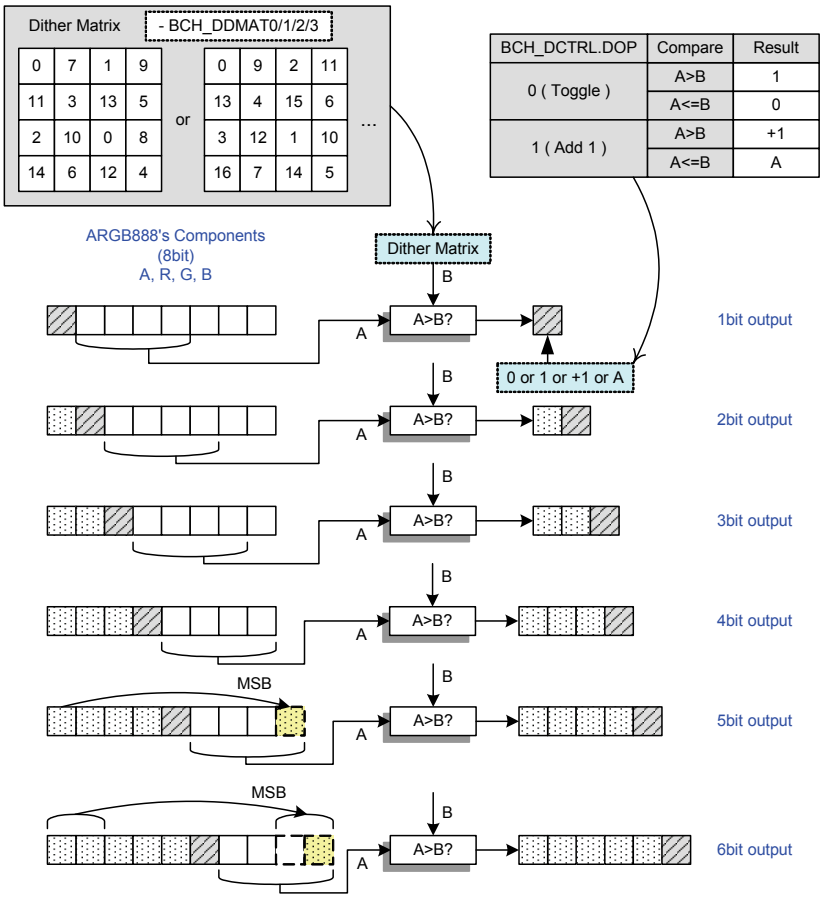


Figure 4.12 LCD RGB Dithering Operation

4.3 Display Interface

4.3.1 STN-LCD

The LCDC generates VSYNC, HSYNC, PXCLK, ACBIAS, and PXDATA signals for STN LCD driver.

The output data format for STN LCD is determined by the interface data width and the color depth of STN LCD. The interface data width is determined by PXDW bits of
LCTRL register. 4bits and 8bits are only supported for STN LCD. The color depth is BPP bits of
LCTRL register and it is not related to a raw image pixel data format.

The timing diagram for STN mode is shown in Figure 4.13. VSYNC and HSYNC pulse are controlled by the configurations of the LPC field of LHTIME and FLC field of LVTIME1 and LVTIME2. Each field is related to the LCD size and display mode.

In 1bpp, 2bpp, 4bpp:

$$LPC = (HorizontalDisplaySize)/(PixelDataWidth) - 1$$

In 8pp and 16bpp (RGB):

$$LPC = (3 \times HorizontalDisplaySize)/(PixelDataWidth) - 1$$

Pixel data width is determined by PXDW of LCTRL register. In the case of STN LCD mode, it must be 4 or 8-bit width.
ACBIAS signal is used by an LCD driver in the STN LCD module to alternate the polarity of the row and column voltage used to turn the pixel on and off. It is controlled by the ACDIV field of LCLKDIV register:

$$f_{ACBIAS} = f_{HSYNC}/(2 \times ACDIV)$$

PXCLK frequency is determined by the CLKDIV field of LCLKDIV register as follows. The minimum value of CLKDIV is 3 in STN mode.

$$f_{PXCLK} = f_{LCLK}/(2 \times CLKDIV) \quad (1)$$

VSYNC frequency is related to the field of FPW, LSWC, LEWC, LPC, and FLC as well as HCLK and PXCLK.

$$f_{VSYNC} = f_{PXCLK}/[\{(LPW+1)+(LPC+1)+(LSWC+1)+(LEWC+1)\} \times \{(FLC+1)+(FPW+1)\}]$$

Therefore, if FR is the required refresh rate, f_{PXCLK_REQ}, which is the required PXCLK, is as following.

$$f_{PXCLK\_REQ} = FR \times [\{(LPW+1)+(LPC+1)+(LSWC+1)+(LEWC+1)\} \times \{(FLC+1)+(FPW+1)\}] \quad (2)$$

The LCDC contains look-up registers for STN LCD with RGB332 or 2BPP to support palletized color. They are LLUTR, LLUTG, and LLUTB. When RGB332 STN LCD is used, LLUTR register allows any 8 red levels to be selected from 16 possible red levels. LLUTG register is for green color and LLUTB register is for blue color. But, when 2BPP STN LCD is used, LLUTG register is only used. Refer to Table 4.1. When STN LCD type is not RGB332 or 2BPP, look-up registers are not used. STN LCD type is determined by BPP bits of LCTRL register.

Table 4.1 STN LCD Palette Address

Pixel Value	Red Palette	Green Palette	Blue Palette
0	LLUTR[3:0]	LLUTG[3:0]	LLUTG[3:0]
1	LLUTR[7:4]	LLUTG[7:4]	LLUTG[7:4]
2	LLUTR[11:8]	LLUTG[11:8]	LLUTG[11:8]
3	LLUTR[15:12]	LLUTG[15:12]	LLUTG[15:12]
4	LLUTR[19:16]	LLUTG[19:16]	N/A
5	LLUTR[23:20]	LLUTG[23:20]	N/A
6	LLUTR[27:24]	LLUTG[27:24]	N/A
7	LLUTR[31:28]	LLUTG[31:28]	N/A

The LCDC contains the dithering pattern registers for STN LCD: a 48-bit modulo 7 dithering pattern register (LDP7L and LDP7H), a 32-bit modulo 5 dithering pattern register (LDP5), a 16-bit modulo 4 dithering pattern register (LDP4), and a 16-bit modulo 3(LDP3) dithering pattern register. These dithering pattern registers can contain the programmable pre-dithered pattern values for each duty cycle ratio.

The LDP7H and LDP7L contain 5 pre-dithered patterns for 1/7, 3/7, 4/7, 5/7, and 6/7 duty cycle rate. Each field of LDP7H and LDP7L is 7-bit long. The LDP5 has 4 pre-dithered pattern fields for 1/5, 2/5, 3/5, and 4/5 duty cycle rate. Each field of LDP5 is 5-bit long. The LDP4 has 3 pre-dithered pattern fields for 1/4, 1/2(=2/4), and 3/4 duty cycle rate, and each field is 4-bit long. Likewise, the LDP3 has 2 fields for 1/3 and 2/3 duty cycle rate with 3-bit length.

Note that the pre-dithered data for 1 and 0 is not defined in the dithering pattern register, because these values are implemented with VDD and VSS condition. The pre-dithered value is pixel value or palletized color value according to STN LCD type. Therefore, if BPP bits of LCTRL register represent RGB332 or 2BPP, pre-dithered value is palletized color. Otherwise, it is pixel value.

Table 4.2 STN LCD Dithering Pattern Register map

Pre-dithered value	Dithering register	Duty cycle ratio
0	all 0s	0
1	DP1_7	1/7
2	DP1_5	1/5
3	DP1_4	1/4
4	DP1_3	1/3
5	DP2_5	2/5
6	DP3_7	3/7
7	DP2_4	1/2
8	DP4_7	4/7
9	DP3_5	3/5
10	DP2_3	2/3
11	DP5_7	5/7
12	DP3_4	3/4
13	DP4_5	4/5
14	DP6_7	6/7
15	all 1s	1

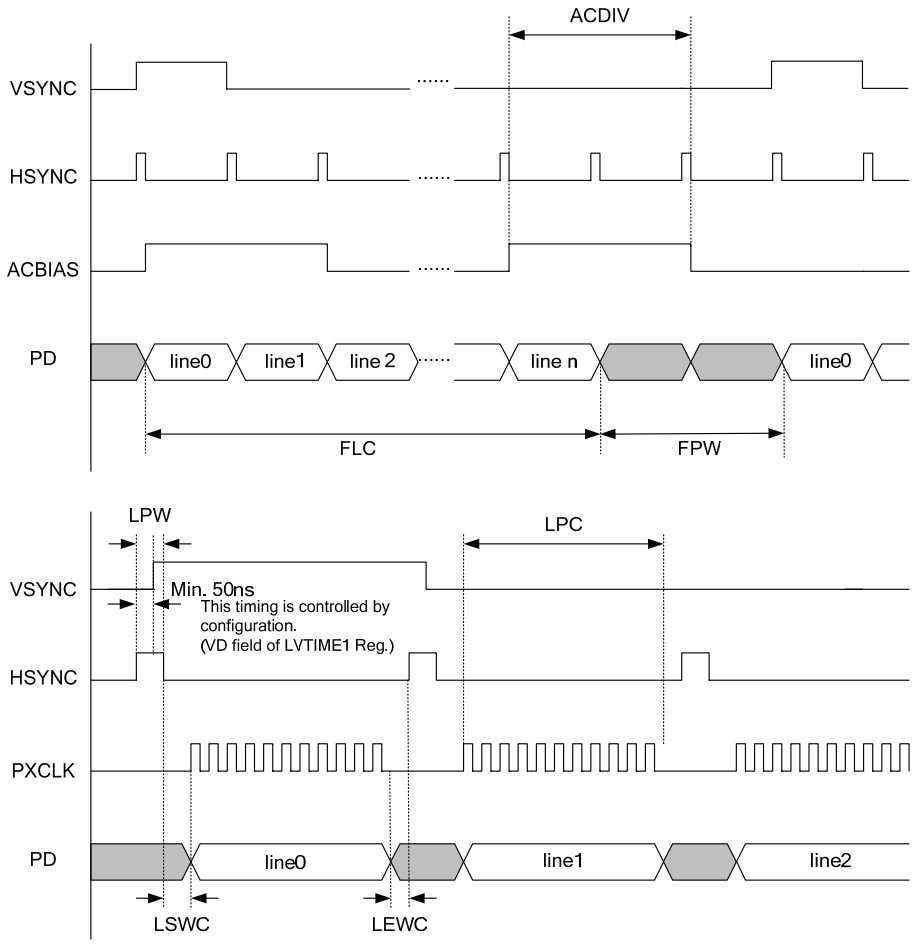


Figure 4.13 STN Mode Timing Diagram

EXAMPLE)

For a monochrome STN LCD, 4-bit interface panel, 4 pixels are captured by the panel in every panel clock cycle. Table 4.3 gives the major registers to be programmed for supporting 4-bit interface STN LCD. LCLK and Refresh rate are examples only. And LSWC, LEWC, LPW, and FPW are STN LCD panel dependent.

LCLK = 20 MHz, Refresh rate = 60 Hz
PXDW* = 0 (4bits), BPP* = 2 (4bpp) , DP* = 0 (one pixel data per one pixel cycle)
NI = 1, TV* = 0, TFT* = 0, STN* = 1
LSWC* = LEWC* = LPW* = FPW* = 1 (STN LCD dependent)

Table 4.3 Monochrome STN LCD (4bits, 1BPP) example

Width (pixel)	Height (pixel)	LPC*	FLC*	DHSIZE*	DVSIZE*	f_{PXCLK_REQ}**	CLKDIV*	f_{PXCLK}***
160	160	39	159	160	160	0.393	25	0.4
160	200	39	199	160	200	0.491	20	0.5
320	200	79	199	320	200	0.973	10	1

*) control registers to be programmed. **) Refer to expression (2). ***) Refer to expression (1).

4.3.2 TFT-LCD

The LCDC supports 16, 18, 24 bpp true-color non-palletized color displays for color TFT LCD. It generates the control signals for LCD driver such as, VSYNC, HSYNC, PXCLK, PXDEN (ACBIAS) and PXDATA. The timing diagram of TFT mode is shown in Figure 4.14

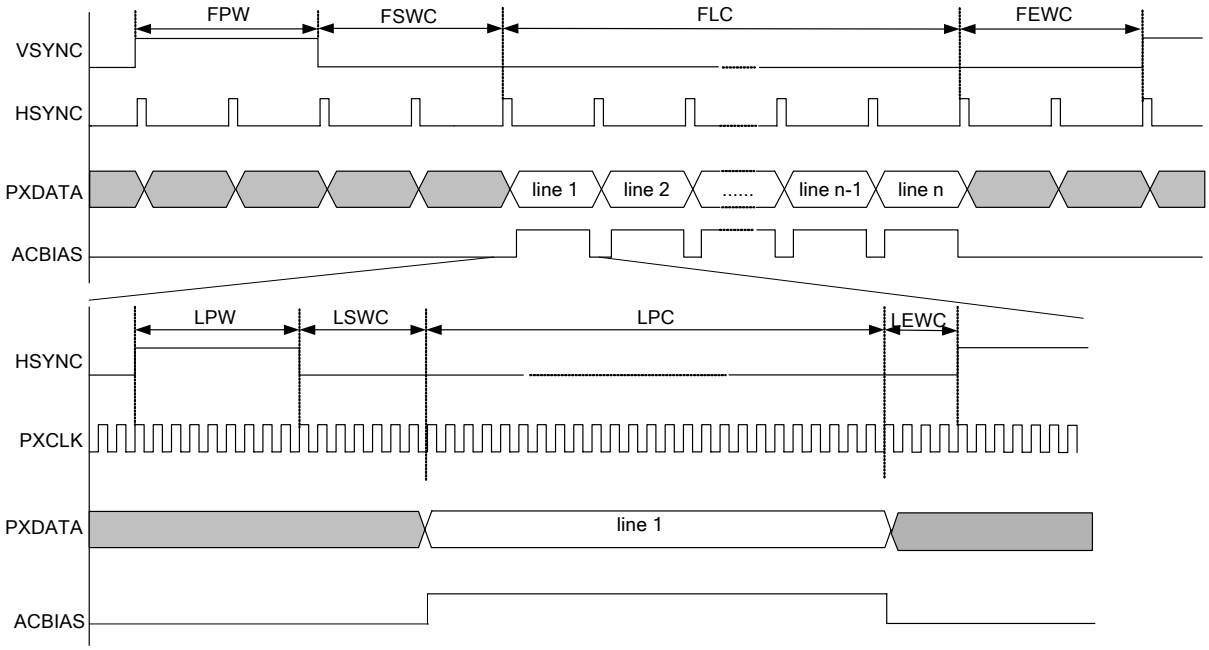


Figure 4.14 TFT Mode Timing Diagram

The VSYNC and HSYNC frequency is controlled by the LPC and FLC field.

$$LPC = (HorizontalDisplaySize) - 1$$
$$FLC = (VerticalDisplaySize) - 1$$

And PXCLK frequency is determined by the CLKDIV value.

$$f_{PXCLK} = f_{LCLK} / (2 \times CLKDIV) \quad (3)$$

The frequency of VSYNC signal is the frame rate. So the frame rate can be calculated as follows:

$$f_{HSYNC} = \frac{f_{PXCLK}}{\{(LPW+1)+(LPC+1)+(LSWC+1)+(LEWC+1)\}}$$

$$f_{VSYNC} = \frac{f_{HSYNC}}{\{(FPW+1)+(FPC+1)+(FSWC+1)+(FEWC+1)\}}$$

Therefore, if FR is the required refresh rate in TFT mode, f_{PXCLK_REQ}, which is the required PXCLK, is as following.

$$f_{PXCLK\_REQ} = FR \times \{(FPW+1)+(FPC+1)+(FSWC+1)+FEWC+1)\} \times \{(LPW+1)+(LPC+1)+(LSWC+1)+(LEWC+1)\} \quad (4)$$

Example)

For TFT LCD(RGB565), if LCLK = 80MHz and Refresh rate = 60Hz,

PXDW* = 0x4, YCbCr* = 0, BPP* = 0x4, DP* = 0, NI* = 1, TV* = 0, TFT* = 1, STN* = 0
LSWC* = LEWC* = LPW* = 3 (TFT LCD dependent)
FSWC* = FEWC* = FPW* = 1 (TFT LCD dependent)

Table 4.4 TFT LCD (RGB565) Example

Width (pixel)	Height (pixel)	LPC*	FLC*	DHSIZE*	DVSIZE*	F_{PXCLK_REQ}**	CLKDIV*	f_{PXCLK}***
176	220	175	219	176	220	2.55	15	2.67
240	320	239	319	240	320	4.93	8	5
640	480	639	479	640	480	19.01	2	20

*) control registers to be programmed. **) Refer to expression (4). ***) Refer to expression (3).

4.3.3 NTSC/PAL Interface

The LCDC can generate the control signals for 8-bit or 16-bit NTSC/PAL encoder. The supporting mode is CCIR601/656 interlace/non-interlace.

For NTSC/PAL interface, TV bit of
LCTRL register must be set. Registers used in this mode are similar to those in TFT mode except for LVTIME1 and LVTIME2 registers; LVTIME1 is for odd field and LVTIME2 is for Even field. And these registers value is not based on HSYNC, but based on half of HSYNC. For example, if FPW of LVTIME1 is 3, pulse width of VSYNC on odd field is not 4 HSYNC cycles, but 2 HSYNC cycles. And if FPW of LVTIME1 is 4, it is 2.5 HSYNC cycles.

Interlace/Non-interlace mode can be configured by NI bit of
LCTRL register. Figure 4.15 and Figure 4.16 show the timing diagram of NTSC and PAL interlace mode. In non-interlace mode, odd field sync signals are repeated instead.

The CCIR656 mode can be configured by 656 bit of
LCTRL register. This mode uses 8 interface signals which have SYNC information as well as 8-bit pixel data. Thus, the additional sync signals are not needed. The output pixel clock must be 27 MHz. Figure 4.17 shows the embedded sync information in the CCIR656 mode.

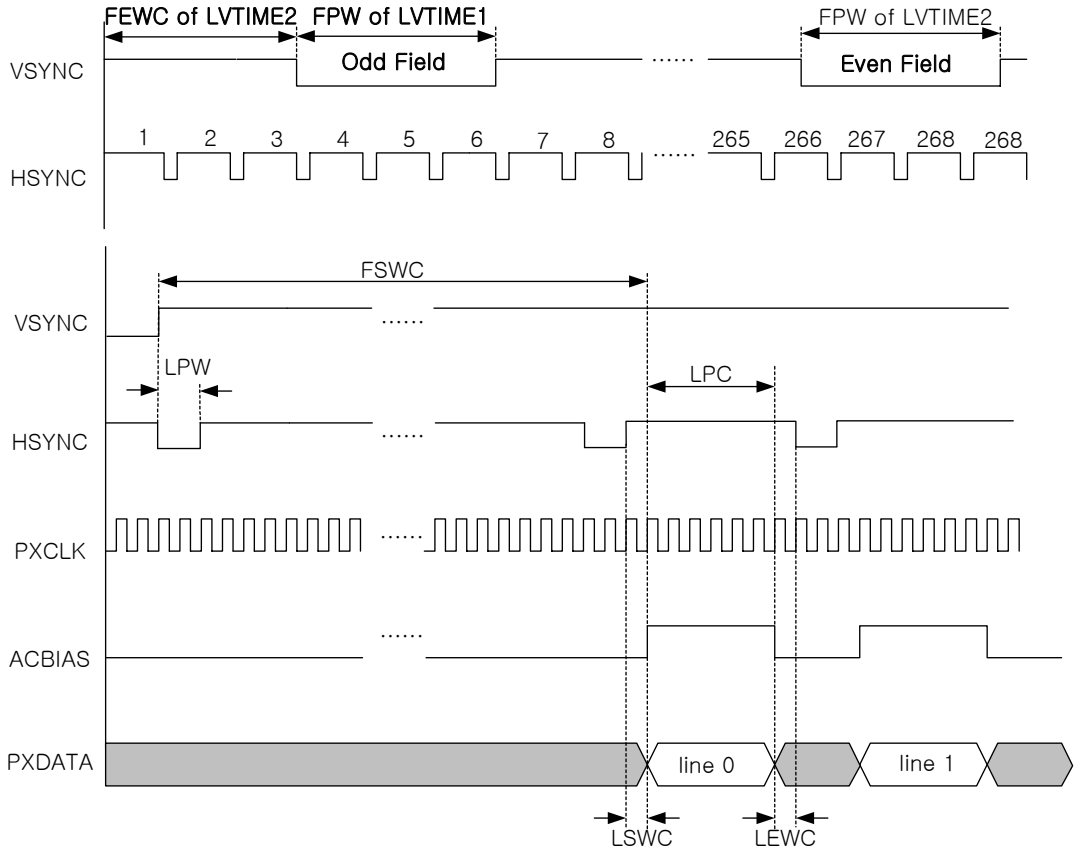


Figure 4.15 NTSC Interlace Mode Timing Diagram

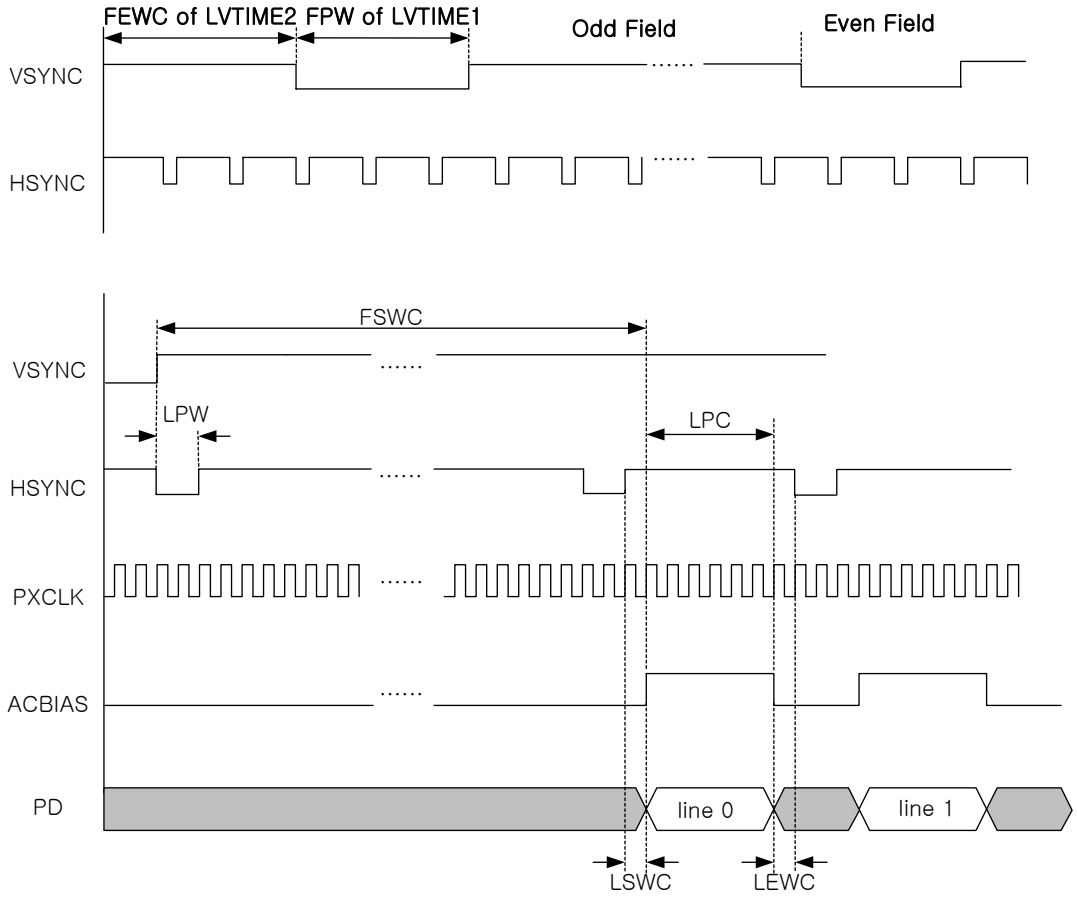
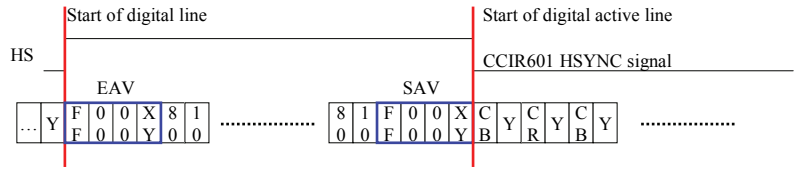


Figure 4.16 PAL Interlace Mode Timing Diagram



	8-bit data							
	D7 (MSB)	D6	D5	D4	D3	D2	D1	D0
preamble	1	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0
status	1	F	V	H	P3	P2	P1	P0

•Status word define
- •F=‘0’ for field 1, ‘1’ for field 2
- •V=‘1’ during vertical blanking
- •H=‘0’ at SAV, ‘1’ at EAV

•Protection bits
- •P3=V xor H
- •P2=F xor H
- •P1=F xor V
- •P0=F xor V xor H

* HSYNC is active low mode

Figure 4.17 CCIR656 Embedded sync. Information

In NTSC mode, one line consists of 1716 pixel clock cycles. The horizontal blanking area occupies 276 pixel clock cycles, and 8 pixel clock cycles within the horizontal blanking is SAV and EAV code. Active area occupies1440 pixel clock cycles.

One frame is composed of 525 lines and is divided into two fields, odd and even.

In PAL mode, one line consists of 1728 pixel clock cycles. The horizontal blanking area occupies 286 pixel clock cycles, and 8 pixel clock cycles within the horizontal blanking is SAV, EAV code. Active area occupies 1440 pixel clock cycles. One frame is composed 625 lines and is divided into two fields, odd and even.

4.4 Register Description

Table 4.5 LCDC Register Map (Base Address = 0xB0A00000/0xB0A04000)

Name	Address	Type	Reset	Description
LCTRL	0x000	R/W	0x0080004A	LCD Control Register
LBC	0x004	R/W	0x00000000	LCD Background Color Register
LCLKDIV	0x008	R/W	0x00000000	LCD Clock Divider Register
LHTIME1	0x00C	R/W	0x00000000	LCD Horizontal Timing Register 1
LHTIME2	0x010	R/W	0x00000000	LCD Horizontal Timing Register 2
LVTIME1	0x014	R/W	0x00000000	LCD Vertical Timing Register 1
LVTIME2	0x018	R/W	0x00000000	LCD Vertical Timing Register 2
LVTIME3	0x01C	R/W	0x00000000	LCD Vertical Timing Register 3
LVTIME4	0x020	R/W	0x00000000	LCD Vertical Timing Register 4
LLUTR	0x024	R/W	0x00000000	LCD Lookup Register for Red
LLUTG	0x028	R/W	0x00000000	LCD Lookup Register for Green
LLUTB	0x02C	R/W	0x00000000	LCD Lookup Register for Blue
LDP7L	0x030	R/W	0x6D2B5401	LCD Modulo 7 Dithering Pattern (low) Register
LDP7H	0x034	R/W	0x0000003F	LCD Modulo 7 Dithering Pattern (high) Register
LDP5	0x038	R/W	0x1d0b0610	LCD Modulo 5 Dithering Pattern Register
LDP4	0x03C	R/W	0x00000768	LCD Modulo 4 Dithering Pattern Register
LDP3	0x040	R/W	0x00000034	LCD Modulo 3 Dithering Pattern Register
LCP1	0x044	R/W	0x000000FF	LCD Clipping Register1
LCP2	0x048	R/W	0x000000FF	LCD Clipping Register2
LDS	0x04C	R/W	0x00000000	LCD Display Size Register
LSTATUS	0x050	R/CLR	0x00000000	LCD Status Register
LIM	0x054	R/W	0x0000FFFF	LCD Interrupt Register.
LGR0	0x058	R/W	0x00000000	LCD Gamma Correction Register 0 for Red Color
LGR1	0x05C	R/W	0x00000000	LCD Gamma Correction Register 1 for Red Color
LGG0	0x060	R/W	0x00000000	LCD Gamma Correction Register 0 for Green Color
LGG1	0x064	R/W	0x00000000	LCD Gamma Correction Register 1 for Green Color
LGB0	0x068	R/W	0x00000000	LCD Gamma Correction Register 0 for Blue Color
LGB1	0x06C	R/W	0x00000000	LCD Gamma Correction Register 1 for Blue Color
LENH	0x070	R/W	0x00000020	LCD Color Enhancement Register
Reserved	0x074	R/W	0x000000C2	Reserved
DITH0	0x078	R/W	0x00000000	LCD RGB dithering pattern Register
DITH1	0x07C	R/W	0x00000000	LCD RGB dithering pattern Register
LI0C	0x080	R/W	0x00000000	LCD Image 0 Control Register
LI0P	0x084	R/W	0x00000000	LCD Image 0 Position Register
LI0S	0x088	R/W	0x00000000	LCD Image 0 Size Register
LI0BA0	0x08C	R/W	0x00000000	LCD Image 0 Base Address 0 Register.
LI0CA	0x090	R/W	0x00000000	LCD Image 0 Current Address Register.
LI0BA1	0x094	R/W	0x00000000	LCD Image 0 Base Address 1 Register
LI0BA2	0x098	R/W	0x00000000	LCD Image 0 Base Address 2 Register
LI0O	0x09C	R/W	0x00000000	LCD Image 0 Offset Register
LI0SR	0x0A0	R/W	0x00000000	LCD Image 0 Scale ratio
LI0A	0x0A4	R/W	0x00000000	LCD Image 0 Alpha Configuration Register
LI0KR	0x0A8	R/W	0x00000000	LCD Image 0 Keying Register for RED or LUMA(Y)
LI0KG	0x0AC	R/W	0x00000000	LCD Image 0 Keying Register for BLUE or CHROMA(Cb)
LI0KB	0x0B0	R/W	0x00000000	LCD Image 0 Keying Register for GREEN or CHROMA(Cr)
LI0EN	0x0B4	R/W	0x00000020	LCD Image 0 Enhancement Register
LI1C	0x0B8	R/W	0x00000000	LCD Image 1 Control Register
LI1P	0x0BC	R/W	0x00000000	LCD Image 1 Position Register
LI1S	0x0C0	R/W	0x00000000	LCD Image 1 Size Register
LI1BA0	0x0C4	R/W	0x00000000	LCD Image 1 Base Address 0 Register.
LI1CA	0x0C8	R/W	0x00000000	LCD Image 1 Current Address Register.
LI1BA1	0x0CC	R/W	0x00000000	Not Used
LI1BA2	0x0D0	R/W	0x00000000	Not Used

Name	Address	Type	Reset	Description
LI1O	0x0D4	R/W	0x00000000	LCD Image 1 Offset Register
LI1SR	0x0D8	R/W	0x00000000	LCD Image 1 Scale ratio-
LI1A	0x0DC	R/W	0x00000000	LCD Image 1 Alpha Configuration Register
LI1KR	0x0E0	R/W	0x00000000	LCD Image 1 Keying Register for RED or LUMA(Y)
LI1KG	0x0E4	R/W	0x00000000	LCD Image 1 Keying Register for BLUE or CHROMA(Cb)
LI1KB	0x0E8	R/W	0x00000000	LCD Image 1 Keying Register for GREEN or CHROMA(Cr)
LI1EN	0x0EC	R/W	0x00000020	LCD Image 1 Enhancement Register
LI2C	0x0F0	R/W	0x00000000	LCD Image 2 Control Register
LI2P	0x0F4	R/W	0x00000000	LCD Image 2 Position Register
LI2S	0x0F8	R/W	0x00000000	LCD Image 2 Size Register
LI2BA0	0x0FC	R/W	0x00000000	LCD Image 2 Base Address 0 Register.
LI2CA	0x100	R/W	0x00000000	LCD Image 2 Current Address Register.
LI2BA1	0x104	R/W	0x00000000	Not Used
LI2BA2	0x108	R/W	0x00000000	Not Used
LI2O	0x10C	R/W	0x00000000	LCD Image 2 Offset Register
LI2SR	0x110	R/W	0x00000000	LCD Image 2 Scale ratio
LI2A	0x114	R/W	0x00000000	LCD Image 2 Alpha Register
LI2KR	0x118	R/W	0x00000000	LCD Image 2 Keying Register for RED or LUMA(Y)
LI2KG	0x11C	R/W	0x00000000	LCD Image 2 Keying Register for BLUE or CHROMA(Cb)
LI2KB	0x120	R/W	0x00000000	LCD Image 2 Keying Register for GREEN or CHROMA(Cr)
LI2EN	0x124	R/W	0x00000020	LCD Image 2 Enhancement Register
LUTIDX	0x128	R/W	0x00000070	Lookup Table index Register
LCDLUT	0x400 – 0x7FC	W	-	LCD Lookup Table

LCTRL (LCD Control Registers) 0xB0A00000/0xB0A04000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EVP	EVS	R2YMD		0	Reserved	GEN	656	CKG	BPP<2:0>			PXDW<3:0>			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ID	IV	IH	IP	CLEN	R2Y	DP	NI	TV	OPT	STN	EVSEL	OVP			LEN

Field	Name	RW	Reset	Description
31	EVP	R/W	0	**External VSYNC Polariy** 1'b0 Direct Input 1'b1 Inverted Input
30	EVS	R/W	0	**External VSYNC Enable** 1'b0 Disable 1'b1 Enable

Field	Name	RW	Reset	Description
29-28	R2YMD	R/W	0	**RGB to YCbCr Conversion Option** 2'b00 The range for "***RGB***" is 16~235, "***Studio Color***" **** Y = 0.299*R + 0.587G + 0.114B*** **** Cb = -0.172*R – 0.339*G + 0.511*B + 128*** **** Cr = 0.511*R – 0.428*G – 0.083*B + 128*** The result is "***Studio Color***" – Normally SDTV. 2'b01 The range for "***RGB***" is 0~255, "***Computer System Color***" **** Y = 0.257*R + 0.504*G + 0.098*B + 16*** **** Cb = -0.148*R – 0.291*G + 0.439*B + 128*** **** Cr = 0.439*R – 0.368*G – 0.071*B + 128*** The result is "***Studio Color***" – Normally SDTV. 2'b10 The range for "***RGB***" is 16~235, "***Studio Color***".. **** Y = 0.213*R + 0.715*G + 0.072*B*** **** Cb = -0.117*R – 0.394*G + 0.511*B + 128*** **** Cr = 0.511*R – 0.464*G – 0.047*B + 128*** The result is "***Studio Color***" – Normally HDTV. 2'b11 The range for "***RGB***" is 0~255, "***Computer System Color***". **** Y = 0.183*R + 0.614*G + 0.062*B + 16*** **** Cb = -0.101*R – 0.338*G + 0.439*B + 128*** **** Cr = 0.439*R – 0.399*G – 0.040*B + 128*** The result is "***Studio Color***" – Normally HDTV. Notice : The coefficients of the above table are approximated for fixed point calculation.
26	Reserved	R/W	0	**Reserved** This filed should be 0.
25	GEN	R/W	0	**Gamma Correction Enable Bit** 1'b0 Disable 1'b1 Enable
24	656	R/W	0	**CCIR 656 Mode** 1'b0 Disable (CCIR 601 Mode) 1'b1 Enable (CCIR656 Mode)
23	CKG	R/W	0	**Clock Gating Enable for Timing Generator** 1'b0 Pixel Clock cannot be gated. If output buffer is empty, the display image can be trembled. 1'b1 Pixel clock can be gated when output buffer is empty. If output buffer is empty, the pixel clock holds on the previous status.
22-20	BPP	R/W	1	**Bit Per Pixel for STN-LCD** 0 1bbp 1 2bpp 2 4bpp 3 RGB332 4 RGB444 5-7 Reserved Notice : BPP is only for STN LCD.

Field	Name	RW	Reset	Description
19-16	PXDW	R/W	0	**Pixel Data Width** 0 4 bits : Only for STN LCD 1 8 bits : Only for STN LCD 2 8 bits : RGB Stripe Type * Odd/Even Line : R – G – B – R – G – B … 3 16bits : RGB565 Format 4 16bits : RGB555 Format 5 18bits : RGB666 Format 6 8 bits : YCbCr Format * Cb – Y – Cr – Y 7 8 bits : YCbCr Format * Cr – Y – Cb – Y 8 16 bits : YCbCr Format * {Y,Cb} – {Y,Cr} 9 16 bits : YCbCr Format * {Y,Cr} – {Y,Cb} 10 8 bits : RGB Delta Type * Odd(1st) Line : R – G – B – R – G – B … * Even(2nd) Line : G – B – R – G – B – R … 11 8 bits : RGB Delta Type * Odd(1st) Line : G – B – R – G – B – R … * Even(2nd) Line : R – G – B – R – G – B … 12 24 bits : RGB888 Format 13 8 bits : RGB Dummy Format * Odd/Even Line : R – G – B – Dummy … 14 16bits : RGB666 Format * R[17:12], G[11:6], B[5:0] * [15:0] : First Data * [17:16] : Second Data 15 16bits : RGB888 Format * R[23:16], G[15:8], B[7:0] * [15:0] : First Data * [23:16] : Second Data Notice : Refer to Figure 4.18 on page 4-30 for more information about PXDW.
15	ID	R/W	0	**Inverted Data Enable (ACBIAS pin)** 0 Active high 1 Active low
14	IV	R/W	0	**Inverted Vertical Sync** 0 active high 1 active low
13	IH	R/W	0	**Inverted Horizontal Sync** 0 active high 1 active low
12	IP	R/W	0	**Inverted Pixel Clock** 0 Pixel data is driven on the rising edge of pixel clock. 1 Pixel data is driven on the falling edge of pixel clock.
11	CLEN	R/W	0	**Clipping Enable** 0 Disable 1 Enable
10	R2Y	R/W	0	**RGB to YCbCr Converter Enable for OUTPUT** 0 Disable 1 Output pixel data is converted to YCbCr format.
9	DP	R/W	0	**Double Pixel Data** 0 One pixel data per 1 PXCLK cycle is output 1 One pixel data per 2 PXCLK cycle is output.
8	NI	R/W	0	**Non-interlace** 0 Interlace mode Odd field timing control : LVTME1, LVTME2 Even field timing control : LVTIME3, LVTIME4 1 Non-interlace mode (progressive mode)
7	TV	R/W	0	**TV mode** 0 Normal mode 1 TV Mode Values of LVTIMEn registers are based on HSYNC cycle/2.

Field	Name	RW	Reset	Description
6	OPT	R/W	1	**LCD DMA Operation mode** 0 8 burst mode 1 16 burst mode
5	STN	R/W	0	**STN LCD mode** 0 Normal mode (for TFT/TV) 1 STN LCD mode
4	EVSEL	R/W	0	**External VSYNC Select** 0 External VSYNC In case of LCD Controller 0 : EVS0 In case of LCD Controller 1 : EVS1 1 Internal Generated VSYNC In case of LCD Controller 0 : VSYNC from LCD Controller 1 In case of LCD Controller 1 : VSYNC from LCD Controller 0
3-1	OVP	R/W	5	**Overlay Priority** 0 (top) Image 0 - Image 1 - Image 2 (bottom) 1 (top) Image 0 - Image 2 - Image 1 (bottom) 2 (top) Image 1 - Image 0 - Image 2 (bottom) 3 (top) Image 2 - Image 0 - Image 1 (bottom) 4 (top) Image 1 - Image 2 - Image 0 (bottom) 5 (top) Image 2 - Image 1 - Image 0 (bottom) Notice : The OVP field determines the relations between image channels and layers.
0	LEN	R/W	0	**LCD Controller Enable** 0 Disable 1 Enable Notice : If LEN was written ‘1’ followed by being written by ‘0’, only 1 frame would be displayed and the display controller would be stopped.

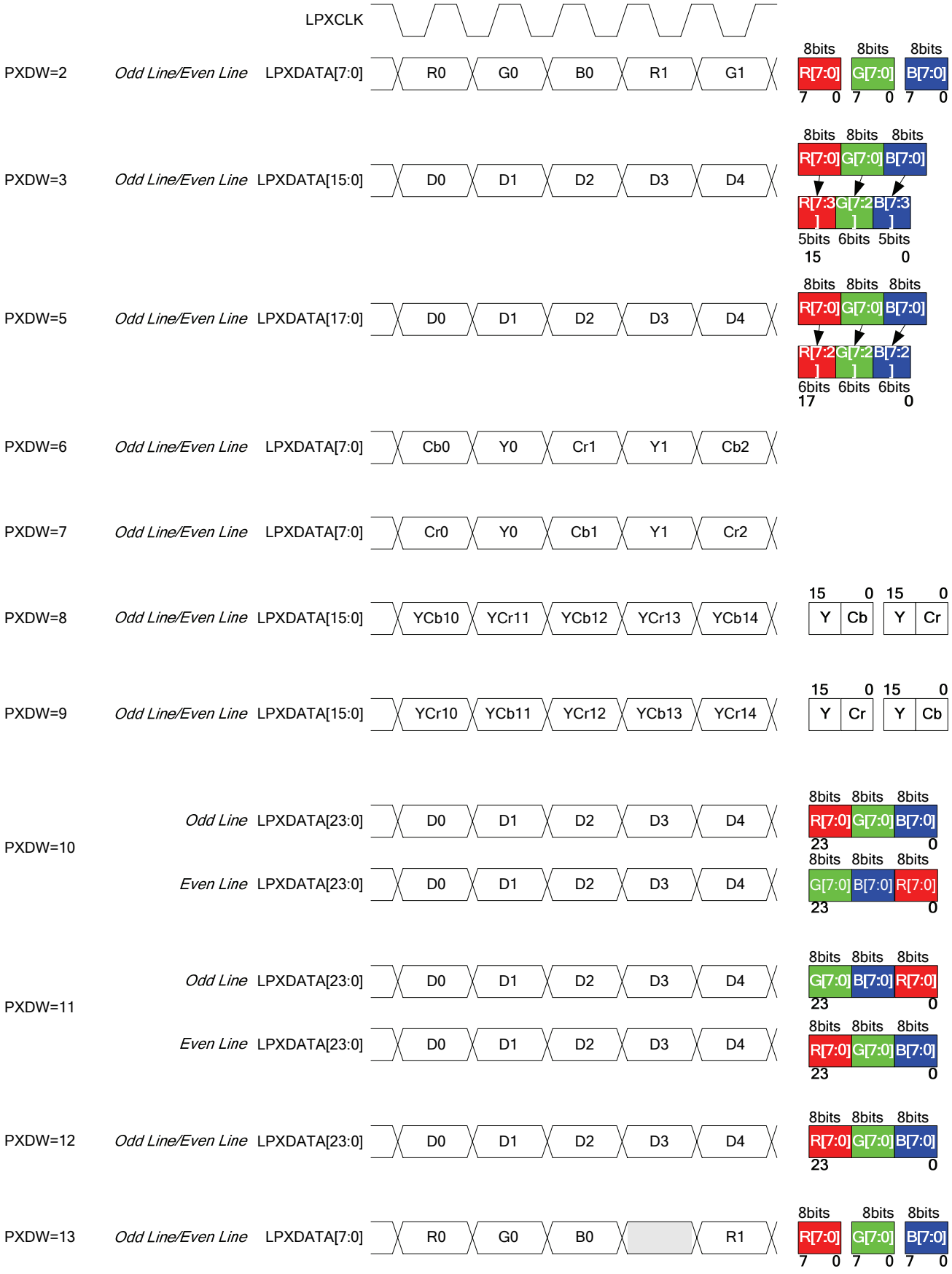


Figure 4.18 Output Pixel Data Format

LBC (LCD Background Color) 0xB0A00004/0xB0A04004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BGV	0							BG2							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BG1								BG0							

Field	Name	RW	Reset	Description
31	BGV	R/W	0	**Background Layer Data Valid Bit for Alpha-blending** 0 The background data(BG0,BG1,BG2) are discarded for alpha-blending 1 The alpha-blending for background data(BG0,BG1,BG2) was applied.
23-16	BG2	R/W	0	**Background Layer Color 2 (Y/Blue)**
15-8	BG1	R/W	0	**Background Layer Color 1 (Cb/Green)**
7-0	BG0	R/W	0	**Background Layer Color 0 (Cr/Red)**

LCLKDIV (LCD Clock Divider Register) 0xB0A00008/0xB0A04008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								ACDIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								PXCLKDIV							

Field	Name	RW	Reset	Description
23-16	ACDIV	R/W	0	AC bias clock divisor (STN only) The number of line clock cycle to count between each toggle of AC_BIAS pin
7-0	PXCLKDIV	R/W	0	Pixel clock divider. Note that programming CLKDIV less than 3 is illegal for STN LCD. PXCLK = LCLK / (PXCLKDIV+1) (PXCLKDIV=0~255)

LHTIME1 (LCD Horizontal Timing Register 1) 0xB0A0000C/0xB0A0400C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							LPW								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0		LPC													

Field	Name	RW	Reset	Description
24-16	LPW	R/W	0	Line pulse width
13-0	LPC	R/W	0	Line pulse count is the number of pixel clock cycles in each line minus 1 on the screen. TFT/NTSC(16bit)/PAL(16bit) : active horizontal pixel – 1 Color STN : (3 * Horizontal display size / pixel width) Mono STN : (Horizontal display size / pixel width) - 1

LHTIME2 (LCD Horizontal Timing Register 2) 0xB0A00010/0xB0A04010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							LSWC								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0							LEWC								

Field	Name	RW	Reset	Description
24-16	LSWC	R/W	0	Line start wait clock is the number of dummy pixel clock cycles minus 1 to be inserted from the start of each horizontal line of pixels.
8-0	LEWC	R/W	0	Line end wait clock is the number of dummy pixel clock cycles minus 1 to be inserted before the end of each horizontal line of pixels

LVTIME1 (LCD Vertical Timing Register 1) 0xB0A00014/0xB0A04014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VDB					0	VDF				FPW					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0		FLC													

Field	Name	RW	Reset	Description
31-27	VDB	R/W	0	Back porch VSYNC delay Delay cycle is -10 to 10cycle delay by PXLCLK. When TV mode, VDB value is equal to VDF. Ex) if VD=5, VSYNC delay is 5 cycle delay by HSYNC.
25-22	VDF	R/W	0	Front porch of VSYNC delay Delay cycle is 0 to 10 cycle delay by PXLCLK. Ex) if VD=5, VSYNC delay is 5 cycle delay by HSYNC.
21-16	FPW	R/W	0	TFT/TV : Frame pulse width is the pulse width of frame clock (VSYNC). STN : N/A
13-0	FLC	R/W	0	Frame line count is the number of lines in each frame on the screen.

Refer to Figure 4.13, Figure 4.14, and Figure 4.15.

LVTIME2 (LCD Vertical Timing Register 2) 0xB0A00018/0xB0A04018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							FSWC								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0							FEWC								

Field	Name	RW	Reset	Description
24-16	FSWC	R/W	0	TFT/TV: Frame start wait cycle is the number of lines to be inserted at the end of each frame. STN : FSWC is N/A. If FSWC[0] is set, VSYNC signal starts on negative falling edge of HSYNC.
8-0	FEWC	R/W	0	TFT/TV: Frame end wait cycle is the number of lines to be inserted at the beginning of each frame. STN extra dummy lines between the end and beginning of frame.

LVTIME3 (LCD Vertical Timing Register 3) 0xB0A0001C/0xB0A0401C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										FPW					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0		FLC													

Field	Name	RW	Reset	Description
21-16	FPW	R/W	0	-
13-0	FLC	R/W	0	-

If NI of LCTRL is 0, LVTIME3 and LVTIME4 are for even field. Otherwise, LVTIME3 and LVTIME4 must be equal to LVTIME1 and LVTIME2, respectively.

LVTIME4 (LCD Vertical Timing Register 4) 0xB0A00020/0xB0A04020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							FSWC								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0							FEWC								

Field	Name	RW	Reset	Description
24-16	FSWC	R/W	0	-
8-0	FEWC	R/W	0	-

If NI of LCTRL is 0, LVTIME3 and LVTIME4 are for even field. Otherwise, LVTIME3 and LVTIME4 must be equal to LVTIME1 and LVTIME2, respectively.

LLUTR (LCD Lookup Register for RED) 0xB0A00024/0xB0A04024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLUTR<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLUTR<15:0>															

Field	Name	RW	Reset	Description
31-0	LLUTR	R/W	0	-

This register is used for supporting palletized color STN-LCD. It is divided into 8 nibbles. The passive color mode uses a lookup table register, which allows any 8 red levels to be selected out of the 16 possible red levels. The most significant 3-bit of 8-bit encoded pixel addresses 8 red palette locations. Note that LLUTR register is only used in STN-LCD mode.

LLUTG (LCD Lookup Register for GREEN) 0xB0A00028/0xB0A04028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LLUTG<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLUTG<15:0>															

Field	Name	RW	Reset	Description
31-0	LLUTG	R/W	0	-

This register is used for supporting palletized color STN-LCD. It is divided into 8 nibbles. The passive color mode uses a lookup table register, which allows any 8 green levels to be selected out of the 16 possible green levels. The most significant 3-bit of 8-bit encoded pixel addresses 8 green palette locations. Note that LLUTG register is only used in STN-LCD mode.

LLUTB (LCD Lookup Register for BLUE) 0xB0A0002C/0xB0A0402C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLUTB<15:0>															

Field	Name	RW	Reset	Description
15-0	LLUTB	R/W	0	-

This register is used for supporting palletized color STN-LCD. It is divided into 4 nibbles. The passive color mode uses a lookup table register, which allows any 4 blue levels to be selected out of the 16 possible blue levels. The most significant 2-bit of 8-bit encoded pixel addresses 4 blue palette locations. Note that LLUTB register is only used in STN-LCD mode.

LDP7L (LCD Modulo 7 Dithering Pattern (Low) Register) 0xB0A00030/0xB0A04030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	DP5_7							0	DP4_7						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	DP3_7							0	DP1_7						

Field	Name	RW	Reset	Description
30-24	DP5_7	R/W	0x6D	-
22-16	DP4_7	R/W	0x2B	-
14-8	DP3_7	R/W	0x54	-
6-0	DP1_7	R/W	0x01	-

LDP7H (LCD Modulo 7 Dithering Pattern (High) Register) 0xB0A00034/0xB0A04034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									DP6_7						

Field	Name	RW	Reset	Description
6-0	DP6_7	R/W	0x3F	-

LDP5 (LCD Modulo 5 Dithering Pattern Register) 0xB0A00038/0xB0A04038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			DP4_5					0			DP3_5				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0			DP2_5					0			DP1_5				

Field	Name	RW	Reset	Description
28-24	DP4_5	R/W	0x1D	-
20-16	DP3_5	R/W	0x0B	-
12-8	DP2_5	R/W	0x06	-
4-0	DP1_5	R/W	0x10	-

LDP4 (LCD Modulo 4 Dithering Pattern Register) 0xB0A0003C/0xB0A0403C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0				DP3_4				DP2_4				DP1_4			

Field	Name	RW	Reset	Description
11-8	DP3_4	R/W	0x7	-
7-4	DP2_4	R/W	0x6	-
3-0	DP1_4	R/W	0x8	-

LDP3 (LCD Modulo 3 Dithering Pattern Register) 0xB0A00040/0xB0A04040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0									DP2_3			0	DP1_3		

Field	Name	RW	Reset	Description
6-4	DP2_3	R/W	0x3	-
2-0	DP1_3	R/W	0x4	-

Theses dithering pattern registers are only used by STN LCD. It is recommended that reset values are used.

Pre-dithered value	Dithering register	Duty cycle ratio
0	all 0s	0
1	DP1_7	1/7
2	DP1_5	1/5
3	DP1_4	1/4
4	DP1_3	1/3
5	DP2_5	2/5
6	DP3_7	3/7
7	DP2_4	1/2
8	DP4_7	4/7
9	DP3_5	3/5
10	DP2_3	2/3
11	DP5_7	5/7
12	DP3_4	3/4
13	DP4_5	4/5
14	DP6_7	6/7
15	all 1s	1

LCP1 (LCD Clipping Register 1) 0xB0A00044/0xB0A04044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								CLP1L							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								CLP1							

Field	Name	RW	Reset	Description
23-16	CLP1L	R/W	0	Clipping Y/R below this value. (standard value is 0)
7-0	CLP1	R/W	0	Clipping Y/R upper this value. (standard value is 255)

LCP2 (LCD Clipping Register 2) 0xB0A00048/0xB0A04048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								CLP2L							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								CLP2							

Field	Name	RW	Reset	Description
23-16	CLP2L	R/W	0	Clipping CHROMA/G/B below this value. (standard value is 0)
7-0	CLP2	R/W	0	Clipping CHROMA/G/B upper this value. (standard value is 255)

LCP1 and LCP2 may be used not to interpret pixel data as embedded sync signal in the CCIR-656 interface.

LDS (LCD Display Size Register) 0xB0A0004C/0xB0A0404C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			VSIZE												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			HSIZE												

Field	Name	RW	Reset	Description
28-16	VSIZE	R/W	0	Horizontal size : number of active pixel in a line
12-0	HSIZE	R/W	0	Vertical size : number of active lines

LSTATUS (LCD Status Register) 0xB0A00050/0xB0A04050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VS	BUSY	EF	DEOF	I0EOF	I1EOF	I2EOF	0								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0			IE2F	IE2R	IE1F	IE1R	IE0F	IE0R	DEF	DER	DD	RU	VSF	VSR	FU

Field	Name	RW	Reset	Description
31	VS	R	0	Monitoring vertical sync.- Read-Only register It has the same logic level with LVSYNC port.
30	BUSY	R	0	Busy signal During operation, is can be read by 1. If LEN is disabled, it will be read by 0 after current frame has been displayed.
29	EF	R	0	It indicates that the type of current frame(field) - even-field (read only) 0 Odd field or frame 1 Even field or frame
28	DEOF	R	0	DMA End
27	I0EOF	R	0	Image 0 End
26	I1EOF	R	0	Image 1 End
25	I2EOF	R	0	Image 2 End
12	IE2F	R/Clear	0	Image 2 end-of-frame falling edge flag*
11	IE2R	R/Clear	0	Image 2 end-of-frame rising edge flag*
10	IE1F	R/Clear	0	Image 1 end-of-frame falling edge flag*
9	IE1R	R/Clear	0	Image 1 end-of-frame rising edge flag*
8	IE0F	R/Clear	0	Image 0 end-of-frame falling edge flag*
7	IE0R	R/Clear	0	Image 0 end-of-frame rising edge flag*
6	DEF	R/Clear	0	DMA end-of-frame falling edge flag*
5	DER	R/Clear	0	DMA end-of-frame rising edge flag*
4	DD	R/Clear	0	Disable done If LEN is disabled, it will be set after current frame has been displayed.
3	RU	R/Clear	0	Register update flag* It indicates that all registers programmed are applied to current frame data.
2	VSF	R/Clear	0	VS falling flag*
1	VSR	R/Clear	0	VS rising flag*
0	FU	R/Clear	0	LCD output fifo under-run flag*

*) The flags should be cleared by writing '1' to the corresponding bit.

LIM (LCD Interrupt Masking Register) 0xB0A00054/0xB0A04054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0			IE2F	IE2R	IE1F	IE1R	IE0F	IE0R	DEF	DER	DD	RU	VSF	VSR	FU

Field	Name	RW	Reset	Description
12	IE2F	R/W	1	Image 2 end-of-frame falling edge interrupt mask
11	IE2R	R/W	1	Image 2 end-of-frame rising edge interrupt mask
10	IE1F	R/W	1	Image 1 end-of-frame falling edge interrupt mask
9	IE1R	R/W	1	Image 1 end-of-frame rising edge interrupt mask
8	IE0F	R/W	1	Image 0 end-of-frame falling edge interrupt mask
7	IE0R	R/W	1	Image 0 end-of-frame rising edge interrupt mask
6	DEF	R/W	1	DMA end-of-frame falling edge interrupt mask
5	DER	R/W	1	DMA end-of-frame rising edge interrupt mask
4	DD	R/W	1	Disable done interrupt mask
3	RU	R/W	1	Register update interrupt mask
2	VSF	R/W	1	VS falling interrupt mask
1	VSR	R/W	1	VS rising interrupt mask
0	FU	R/W	1	LCD output fifo under-run interrupt mask

0: enable interrupt 1: mask out

LGR0 (LCD Gamma Correction Register 0 for Red Color) 0xB0A00058/0xB0A04058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GR7				GR6				GR5				GR4			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GR3				GR2				GR1				GR0			

Field	Name	RW	Reset	Description
31-28	GR7	R/W	0	-
27-24	GR6	R/W	0	-
23-20	GR5	R/W	0	-
19-16	GR4	R/W	0	-
15-12	GR3	R/W	0	-
11-8	GR2	R/W	0	-
7-4	GR1	R/W	0	-
3-0	GR0	R/W	0	-

LGR1 (LCD Gamma Correction Register 1 for Red Color) 0xB0A0005C/0xB0A0405C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GR15				GR14				GR13				GR12			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GR11				GR10				GR9				GR8			

Field	Name	RW	Reset	Description
31-28	GR15	R/W	0	-
27-24	GR14	R/W	0	-
23-20	GR13	R/W	0	-
19-16	GR12	R/W	0	-
15-12	GR11	R/W	0	-
11-8	GR10	R/W	0	-
7-4	GR9	R/W	0	-
3-0	GR8	R/W	0	-

LGG0 (LCD Gamma Correction Register 0 for Green Color) 0xB0A00060/0xB0A04060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GG7				GG6				GG5				GG4			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GG3				GG2				GG1				GG0			

Field	Name	RW	Reset	Description
31-28	GG7	R/W	0	-
27-24	GG6	R/W	0	-
23-20	GG5	R/W	0	-
19-16	GG4	R/W	0	-
15-12	GG3	R/W	0	-
11-8	GG2	R/W	0	-
7-4	GG1	R/W	0	-
3-0	GG0	R/W	0	-

LGG1 (LCD Gamma Correction Register 1 for Green Color) 0xB0A00064/0xB0A04064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GG15				GG14				GG13				GG12			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GG11				GG10				GG9				GG8			

Field	Name	RW	Reset	Description
31-28	GG15	R/W	0	-
27-24	GG14	R/W	0	-
23-20	GG13	R/W	0	-
19-16	GG12	R/W	0	-
15-12	GG11	R/W	0	-
11-8	GG10	R/W	0	-
7-4	GG9	R/W	0	-
3-0	GG8	R/W	0	-

LGB0 (LCD Gamma Correction Register 0 for Blue Color) 0xB0A00068/0xB0A04068

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GB7				GB6				GB5				GB4			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GB3				GB2				GB1				GB0			

Field	Name	RW	Reset	Description
31-28	GB7	R/W	0	-
27-24	GB6	R/W	0	-
23-20	GB5	R/W	0	-
19-16	GB4	R/W	0	-
15-12	GB3	R/W	0	-
11-8	GB2	R/W	0	-
7-4	GB1	R/W	0	-
3-0	GB0	R/W	0	-

LGB1 (LCD Gamma Correction Register 1 for Blue Color) 0xB0A0006C/0xB0A0406C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GB15				GB14				GB13				GB12			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GB11				GB10				GB9				GB8			

Field	Name	RW	Reset	Description
31-28	GB15	R/W	0	-
27-24	GB14	R/W	0	-
23-20	GB13	R/W	0	-
19-16	GB12	R/W	0	-
15-12	GB11	R/W	0	-
11-8	GB10	R/W	0	-
7-4	GB9	R/W	0	-
3-0	GB8	R/W	0	-

LENH (LCD Color Enhancement Register) 0xB0A00070/0xB0A04070

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								HUE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BRIGHTNESS								CONTRAST							

Field	Name	RW	Reset	Description
23-16	HUE	R/W	0	Hue Calibration Register – 2's complement signed value * -30 ~ 30 degree * 0x80 for -30 degree * 0x00 for 0 degree for default value * 0x7F for about 30 degree
15-8	BRIGHTNESS	R/W	0	Brightness Calibration Register – 2's complement signed value * -128 ~ 128 value * 0x80 for -128 offset * 0x00 for 0 offset * 0x7F for 127 offset
7-0	CONTRAST	R/W	0x20	Contrast Calibration Register – 2's complement signed value * -4 ~ 4 * 0x80 for -4.0 value * 0xFF for -1.0 value * 0x20 for 1.0 value * 0x7F for about 4.0 value

DITH0 (LCD Dithering Coefficient Register) 0xB0A00078/0xB0A04078

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	DITH13			0	DITH12			0	DITH11			0	DITH10		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	DITH03			0	DITH02			0	DITH01			0	DITH00		

Field	Name	RW	Reset	Description
30-28	DITH13	R/W	0	LCD Dithering Pattern Matrix (1,3)
26-24	DITH12	R/W	0	LCD Dithering Pattern Matrix (1,2)
22-20	DITH11	R/W	0	LCD Dithering Pattern Matrix (1,1)
18-16	DITH10	R/W	0	LCD Dithering Pattern Matrix (1,0)
14-12	DITH03	R/W	0	LCD Dithering Pattern Matrix (0,3)
10-8	DITH02	R/W	0	LCD Dithering Pattern Matrix (0,2)
6-4	DITH01	R/W	0	LCD Dithering Pattern Matrix (0,1)
2-0	DITH00	R/W	0	LCD Dithering Pattern Matrix (0,0)

DITH1 (LCD Dithering Coefficient Register) 0xB0A0007C/0xB0A0407C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	DITH33			0	DITH32			0	DITH31			0	DITH30		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	DITH23			0	DITH22			0	DITH21			0	DITH20		

Field	Name	RW	Reset	Description
30-28	DITH33	R/W	0	LCD Dithering Pattern Matrix (3,3)
26-24	DITH32	R/W	0	LCD Dithering Pattern Matrix (3,2)
22-20	DITH31	R/W	0	LCD Dithering Pattern Matrix (3,1)
18-16	DITH30	R/W	0	LCD Dithering Pattern Matrix (3,0)
14-12	DITH23	R/W	0	LCD Dithering Pattern Matrix (2,3)
10-8	DITH22	R/W	0	LCD Dithering Pattern Matrix (2,2)
6-4	DITH21	R/W	0	LCD Dithering Pattern Matrix (2,1)
2-0	DITH20	R/W	0	LCD Dithering Pattern Matrix (2,0)

LI0C,LI1C,LI2C (Control Registers for Each Image) **0xB0A00080, 0xB0A000B8, 0xB0A000F0**
0xB0A04080, 0xB0A040B8, 0xB0A040F0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTL	AEN	CEN	IEN	SRC	AOPT		ASEL	CINTPL	0						UPD
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD	0				Y2RMD		Y2R	BR	0		FMT				

Field	Name	RW	Reset	Description
31	INTL	R/W	0	**DMA Interlace Mode** 0 Progressive Data Fetch 1 Interlace Data Fetch * Line Order is 1,2,3,4,5,6 … * Line Order of odd frame : 1,3,5,… * Line Order of even frame : 2,4,6,…
30	AEN	R/W	0	**Alpha-blending Function for Each Image** 0 Disabled 1 Enabled
29	CEN	R/W	0	**Chroma-Keying Function for Each Image** 0 Disabled 1 Enabled
28	IEN	R/W	0	**Image Displaying Function for Each Image** 0 Disabled 1 Enabled
27	SRC	R/W	0	**Image Source Select** 0 Image data from the memory 1 Image data from the on-the-fly path
26-25	AOPT	R/W	0	**Alpha-blending Option Selection Bits** 0 The alpha value is not changed. (100% ~ 0.39% transparency) 1 The alpha value is added by '1' (99.61% ~ 0% transparency) 2 The alpha value is not changed when msb of alpha is '0' and added by '1' when most significant bit of alpha value is '1' (100% ~ 0% transparency) 3 Reserved
24	ASEL	R/W	0	**Image Displaying Function for Each Image** 0 A10 bits and A11 are used as alpha value 1 Alpha bits in the pixel are used as alpha value.
23	CINTPL	R/W	0	Chroma (Cb/Cr) Interpolation 0 Disable 1 Enable
16	UPD	R/W	0	**Image Register settings Update** 0 Do not update 1 Update next frame Notice : This field is automatically clear at every start of frame.
15	PD	R/W	0	**Bit padding** 0 The extra lower bits are padded by '0'. 1 The extra lower bits are padded by most significant bit. Notice : Though raw image source is not RGB888 format, it is internally converted to RGB888 as appending additional bits. When this conversion is occurred, these padding bits are determined by PD bit (see Figure 4.19)

Field	Name	RW	Reset	Description
10-9	Y2RMD	R/W	0	**YCbCr to RGB Conversion Option** 0 The range for "YCbCr" is 16 ~ 235, "Studio Color". **** R = Y + 1.371 * (Cr – 128)*** **** G = Y + 0.336 * (Cb – 128) – 0.698 * (Cr – 128)*** **** B = Y + 1.732 * (Cb – 128)*** The result is "Studio Color" – Normally SDTV. 1 The range for "YCbCr" is 16 ~ 235, "Studio Color". **** R = 1.164 * (Y – 16) + 1.596 * (Cr – 128)*** **** G = 1.164 * (Y – 16) – 0.391 * (Cb – 128) – 0.813 * (Cr – 128)*** **** B = 1.164 * (Y – 16) + 2.018 * (Cb – 128)*** The result is "Computer System Color" – Normally SDTV. 2 The range for "YCbCr" is 16 ~ 235, "Studio Color". **** R = Y + 1.540 * (Cr – 128)*** **** G = Y – 0.183 * (Cb – 128) – 0.459 * (Cr – 128)*** **** B = Y + 1.816 * (Cb – 128)*** The result is "Studio Color" – Normally HDTV. 3 The range for "YCbCr" is 16 ~ 235, "Studio Color". **** R = 1.164 * (Y – 16) + 1.793 * (Cr – 128)*** **** G = 1.164 * (Y – 16) – 0.213 * (Cb – 128) – 0.534 * (Cr – 128)*** **** B = 1.164 * (Y – 16) + 2.115 * (Cb – 128)*** The result is "Computer System Color" – Normally HDTV. Notice : The coefficients of the above table are approximated for fixed point calculation.
8	Y2R	R/W	0	**YCbCr to RGB Conversion Enable Bit** 0 Disable 1 Enable
7	BR	R/W	0	**Bit Reverse** 0 Little-endian pixel data 1 Big-endian pixel data Notice : BR is only used when BPP is 1, 2, or 4 bpp
4-0	FMT	R/W	0	**Image Format** 0 1bpp indexed color 1 2bpp indexed color 2 4bpp indexed color 3 8bpp indexed color 4 ~ 7 Reserved for future use 8 RGB332 – 1 bytes aligned : R[7:5],G[4:2],B[1:0] 9 RGB444 – 2 bytes aligned : A[15:12],R[11:8],G[7:4],B[3:0] 10 RGB565 – 2 bytes aligned : R[15:11],G[10:5],B[4:0] 11 RGB555 – 2 bytes aligned : A[15],R[14:10],G[9:5],B[4:0] 12 RGB888 – 4 bytes aligned : A[31:24],R[23:16],G[15:8],B[7:0] 13 RGB666 – 4 bytes aligned : A[23:18],R[17:12],G[11:6],B[5:0] 14 ~ 23 Reserved for future use 24 YCbCr 4:2:0 separated format* 25 YCbCr 4:2:2 separated format* 26 YCbCr 4:2:2 sequential format 27 Reserved for future use 28 YCbCr 4:2:0 interleaved type 0 format* 29 YCbCr 4:2:0 interleaved type 1 format* 30 YCbCr 4:2:2 interleaved type 0 format* 31 YCbCr 4:2:2 interleaved type 1 format* Notice : The "A", "R", "G" and "B" stand for alpha value, red, green and blue color correspondingly. The alpha value to be applied is determined by ASEL bit. If ASEL is zero, the alpha value equals to "A" with extra bit extended. *) Not Supported for Image 1 and 2

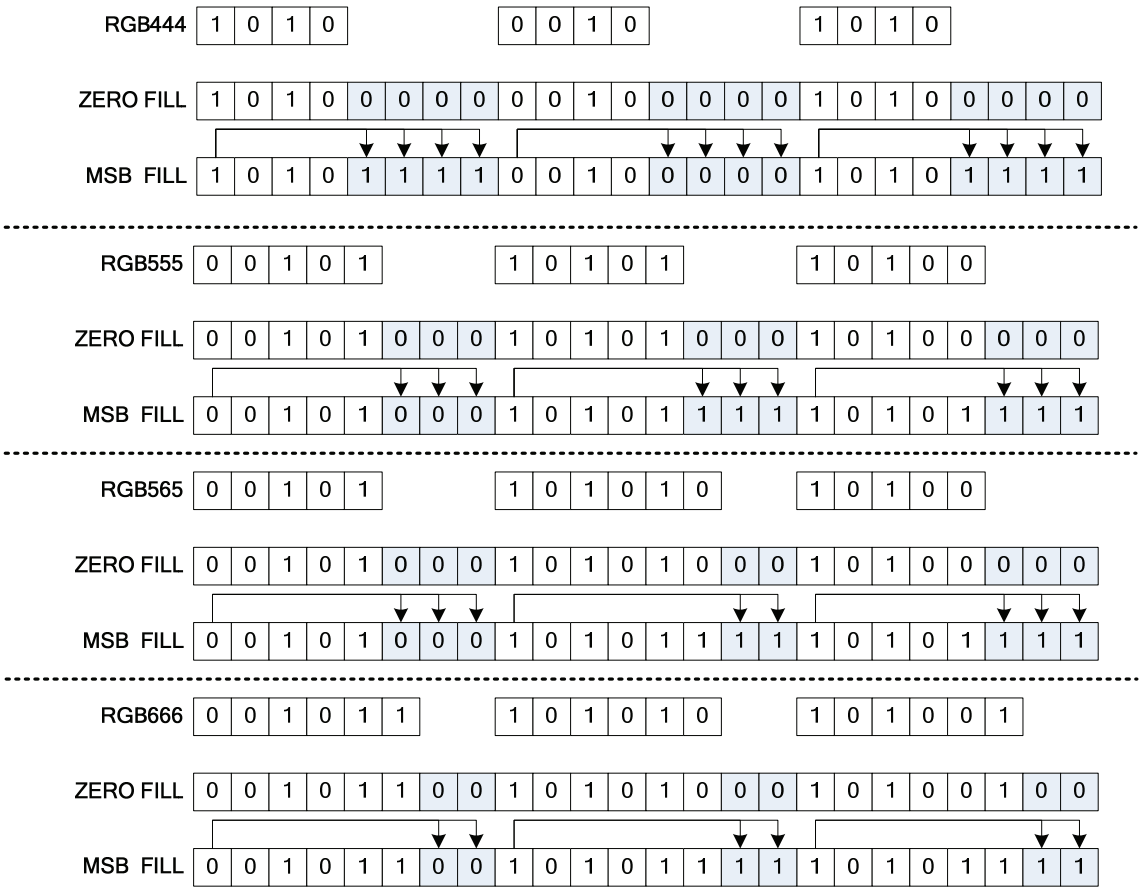


Figure 4.19 Bit Padding

LI0P,LI1P,LI2P (Start Position Registers for Each Image) 0xB0A00084, 0xB0A000BC, 0xB0A000F4 0xB0A04084, 0xB0A040BC, 0xB0A040F4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			POSY												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			POSX												

Field	Name	RW	Reset	Description
28-16	POSY	R/W	0	Y position to display.
12-0	POSX	R/W	0	X position to display.

LI0S,LI1S,LI2S (Image Size Registers for Each Image Channel) 0xB0A00088, 0xB0A000C0, 0xB0A000F8 0xB0A04088, 0xB0A040C0, 0xB0A040F8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0			HEIGHT												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0			WIDTH												

Field	Name	RW	Reset	Description
28-16	HEIGHT	R/W	0	Height of each Image
12-0	WIDTH	R/W	0	Width of each image

FMT	Image Format	Pixel Width Constraint
0	1bpp indexed color	64 pixels
1	2bpp indexed color	16 pixels
2	4bpp indexed color	8 pixels
3	8bpp indexed color	4 pixels
4 ~ 7	Reserved for future use	
8	RGB332	4 pixels
9	RGB444	2 pixels
10	RGB565	2 pixels
11	RGB555	2 pixels
12	RGB888	1 pixels
13	RGB666	1 pixels
14 ~ 23	Reserved for future use	
24	YCbCr 4:2:0 separated format	8 pixels
25	YCbCr 4:2:2 separated format	8 pixels
26	YCbCr 4:2:2 sequential format	2 pixels
27	Reserved for future use	
28	YCbCr 4:2:0 interleaved type 0 format	4 pixels
29	YCbCr 4:2:0 interleaved type 1 format	4 pixels
30	YCbCr 4:2:2 interleaved type 0 format	4 pixels
31	YCbCr 4:2:2 interleaved type 1 format	4 pixels

Important Notice :
The value of “WIDTH” should be integer multiple of the number of pixels described in the “Pixel Width Constraint” field in the above table. The limitation can be changed by “X_SCALE” in the following LI0SC, LI1SC, LI2SC register. The downscale ratio determined by “X_SCALE” should be multiplied by the upper constraint. For example, If the downscale ratio is 3 and FMT is 2, the pixel width limitation is changed from “8 pixels” to “24 pixels”

LI0BA0,LI1BA0,LI2BA0 (Base Address for Each Images) 0xB0A0008C, 0xB0A000C4, 0xB0A000FC 0xB0A0408C, 0xB0A040C4, 0xB0A040FC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE0															

Field	Name	RW	Reset	Description
31-0	BASE0	R/W	0	1st base address for each image If the image format is YCbCr separated type, it is the base address of Y.

LI0CA, LI1CA, LI2CA (Current Address for Each Images) 0xB0A00090, 0xB0A000C8, 0xB0A00100 0xB0A04090, 0xB0A040C8, 0xB0A04100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CURR															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURR															

Field	Name	RW	Reset	Description
31-0	CURR	R/W	0	The working address for 1st base address.

LI0BA1,LI1BA1,LI2BA1 (2nd Base Address for Each Images) 0xB0A00094, 0xB0A000CC, 0xB0A00104 0xB0A04094, 0xB0A040CC, 0xB0A04104

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE1															

Field	Name	RW	Reset	Description
31-0	BASE1	R/W	0	The 2nd base address for each image. If the image format is YCbCr separated type, it is the base address for Cb, otherwise it is not used. ***Notice : The LI1BA1 and LI2BA1 are not supported.***

LI0BA2,LI1BA2,LI2BA2 (3rd Base Address for Each Images) 0xB0A00098, 0xB0A000D0, 0xB0A00108 0xB0A04098, 0xB0A040D0, 0xB0A04108

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE2															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE2															

Field	Name	RW	Reset	Description
31-0	BASE2	R/W	0	The 3rd base address for each image. If the image format is YCbCr separated type, it is the base address for Cb, otherwise it is not used. ***Notice : The LI1BA2 and LI2BA2 are not supported.***

FMT	Image Format	Base Address Constraints
0	1bpp indexed color	8 pixels aligned (1byte aligned)
1	2bpp indexed color	4 pixels aligned (1byte aligned)
2	4bpp indexed color	2 pixels aligned (1byte aligned)
3	8bpp indexed color	1 pixels aligned (1byte aligned)
4 ~ 7	Reserved for future use	
8	RGB332	1 pixels aligned (1byte aligned)
9	RGB444	1 pixels aligned (2bytes aligned)
10	RGB565	1 pixels aligned (2bytes aligned)
11	RGB555	1 pixels aligned (2bytes aligned)
12	RGB888	1 pixels aligned (4bytes aligned)
13	RGB666	1 pixels aligned (4bytes aligned)
14 ~ 23	Reserved for future use	
24	YCbCr 4:2:0 separated format	2 pixels aligned (2bytes aligned)
25	YCbCr 4:2:2 separated format	2 pixels aligned (2bytes aligned)
26	YCbCr 4:2:2 sequential format	1 pixels aligned (2bytes aligned)
27	Reserved for future use	
28	YCbCr 4:2:0 interleaved type 0 format	2 pixels aligned (2bytes aligned)
29	YCbCr 4:2:0 interleaved type 1 format	2 pixels aligned (2bytes aligned)
30	YCbCr 4:2:2 interleaved type 0 format	2 pixels aligned (2bytes aligned)
31	YCbCr 4:2:2 interleaved type 1 format	2 pixels aligned (2bytes aligned)

LI0O,LI1O,LI2O (Offset Information Registers for Each Image) 0xB0A0009C, 0xB0A000D4, 0xB0A0010C 0xB0A0409C, 0xB0A040D4, 0xB0A0410C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OFFSET1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OFFSET0															

Field	Name	RW	Reset	Description
31-16	OFFSET1	R/W	0	The 2nd offset information for each image. It is the address offset in U or V channel of FIFO (FIFO1,2) ***Notice : Invalid for LI1O and LI2O***
15-0	OFFSET0	R/W	0	The 1st offset information for each image. It is the address offset in Y or RGB channel of FIFO (FIFO0).

LI0SC,LI1SC,LI2SC (Image Scaling Registers for Each Image) 0xB0A000A0, 0xB0A000D8, 0xB0A00110 0xB0A040A0, 0xB0A040D8, 0xB0A04110

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Y_SCALE				X_SCALE			

Field	Name	RW	Reset	Description
7-4	Y_SCALE	R/W	0	**Input Image Scaling** 0 No-Scale 1 Downscale by 2 2 Downscale by 3 3 Downscale by 4 4-6 Reserved 7 Downscale by 8 8 Reserved 9 Upscale by 2 10 Upscale by 3 11 Upscale by 4 12-14 Reserved 15 Upscale by 8 ***Notice : FMT 28, 29, 30 and 31 does NOT support scaling***
3-0	X_SCALE	R/W	0	

LI0A,LI1A,LI2A (Alpha Information Registers for Each Image) 0xB0A000A4, 0xB0A000DC, 0xB0A00114 0xB0A040A4, 0xB0A040DC, 0xB0A04114

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								A1							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								A0							

Field	Name	RW	Reset	Description
23-16	A1	R/W	0	These bits should be the same as value of “A0”.
7-0	A0	R/W	0	These are used as alpha value for each image when ASEL bit is ‘0’. ***Output Pixel = Main Pixel *(1- A0/256) + Overlay Pixel* (A0)***

LI0KR,LI1KR,LI2KR (Keying Registers for RED or LUMA(Y)) 0xB0A000A8, 0xB0A000E0, 0xB0A00118 0xB0A040A8, 0xB0A040E0, 0xB0A04118

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								KEYMASK							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								KEY							

Field	Name	RW	Reset	Description
23-16	KEYMASK	R/W	0	Key Mask Value for RED (or Y)
7-0	KEY	R/W	0	Key Value for RED (or Y)

LI0KG,LI1KG,LI2KG (Keying Registers for GREEN or CHROMA(Cb))

0xB0A000AC, 0xB0A000E4, 0xB0A0011C
0xB0A040AC, 0xB0A040E4, 0xB0A0411C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								KEYMASK							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								KEY							

Field	Name	RW	Reset	Description
23-16	KEYMASK	R/W	0	Key Mask Value for GREEN (or Cb)
7-0	KEY	R/W	0	Key Value for GREEN (or Cb)

LI0KB,LI1KB,LI2KB (Keying Registers for BLUE or CHROMA(Cr))

0xB0A000B0, 0xB0A000E8, 0xB0A00120
0xB0A040B0, 0xB0A040E8, 0xB0A04120

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								KEYMASK							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								KEY							

Field	Name	RW	Reset	Description
23-16	KEYMASK	R/W	0	Key Mask Value for BLUE (or Cr)
7-0	KEY	R/W	0	Key Value for BLUE (or Cr)

LI0EN,LI1EN,LI2EN (LCD Image Enhancement Register) 0xB0A000B4, 0xB0A000EC, 0xB0A00124 0xB0A040B4, 0xB0A040EC, 0xB0A04124

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								HUE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BRIGHTNESS								CONTRAST							

Field	Name	RW	Reset	Description
23-16	HUE	R/W	0	Hue Calibration Register – 2's complement signed value * -30 ~ 30 degree * 0x80 for -30 degree * 0x00 for 0 degree for default value * 0x7F for about 30 degree
15-8	BRIGHTNESS	R/W	0	Brightness Calibration Register – 2's complement signed value * -128 ~ 128 value * 0x80 for -128 offset * 0x00 for 0 offset * 0x7F for 127 offset
7-0	CONTRAST	R/W	0x20	Contrast Calibration Register – 2's complement signed value * -4 ~ 4 * 0x80 for -4.0 value * 0xFF for -1.0 value * 0x20 for 1.0 value * 0x7F for about 4.0 value

LUTIDX (Lookup Table index Register) 0xB0A00128/0xB0A04128

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0									EN2	EN1	EN0	0		CHSEL	

Field	Name	RW	Reset	Description
6	EN2	R/W	0	Enable Lookup Table for Image Channel 2
5	EN1	R/W	0	Enable Lookup Table for Image Channel 1
4	EN0	R/W	0	Enable Lookup Table for Image Channel 0
1-0	CHSEL	R/W	0	LUT Access channel select * 0 for LUT0 * 1 for LUT1 * 2 for LUT2

5 LCD System Interface

5.1 Overview

The LCD system interface (LCDSI) is used to send out the image data from the system memory to the LCD module which has 68/80-system interface. NVS2310 has 2 independent LCDSIs. The LCDSI can be accessed by both LCDC and the on-chip CPU. The setup time, hold time, and pulse width of the interface signals are programmable.

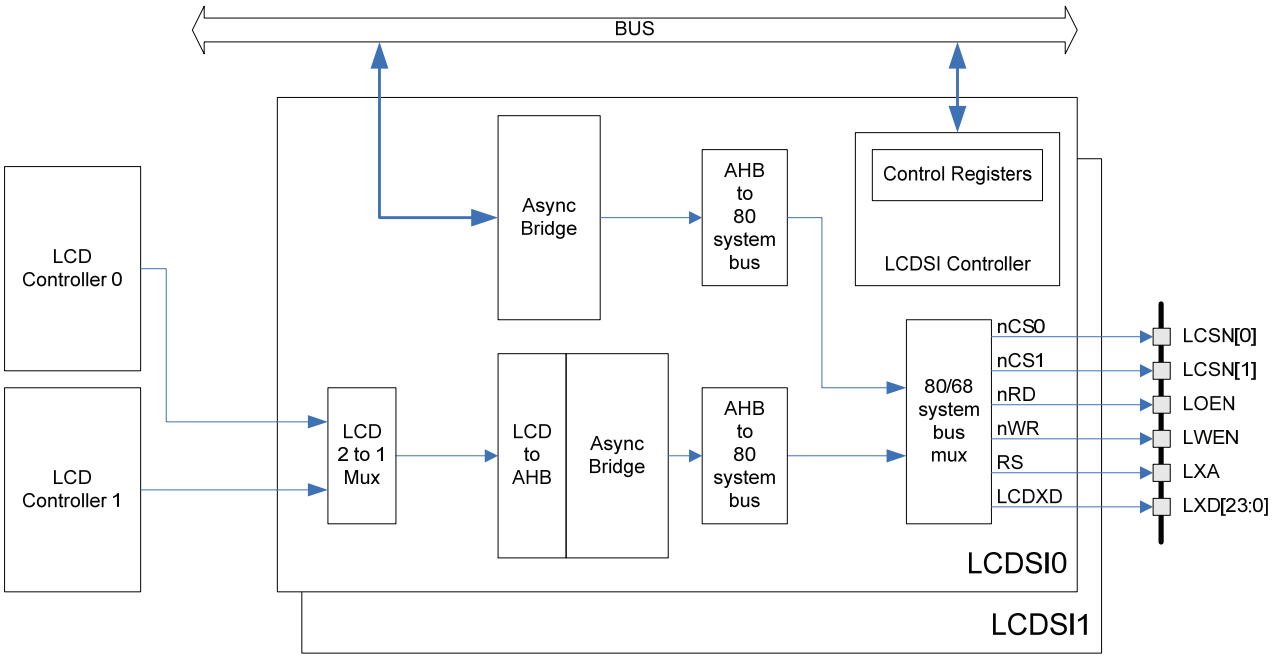


Figure 5.1 LCD System Interface Block Diagram

5.2 Operation

5.2.1 Reading/Writing operation through the on-chip CPU

The LCDSI allows the on-chip CPU to read from and write to an external LCD module, which has 68/80-system interface. If the on-chip CPU accesses LCDSI DATA0 register, then reading or writing operations are generated on the device connected to LCSN[0]. While these operations are executed, LXA is low (Figure 5.2 (a)). If the on-chip CPU accesses LCDSI DATA1 register, LXA is high (Figure 5.2 (b)). Similarly, to access the device connected to LCSN[1], the on-chip CPU must access LCDSI DATA2 or LCDSI DATA3 register. Note that LXA signal is irrelevant to LCDSI LnCTRL.RSP when the on-chip CPU access to LCDSI.

Timing and data width configuration about the LCDSI signals can be programmed via LCDSI CTRLn registers.

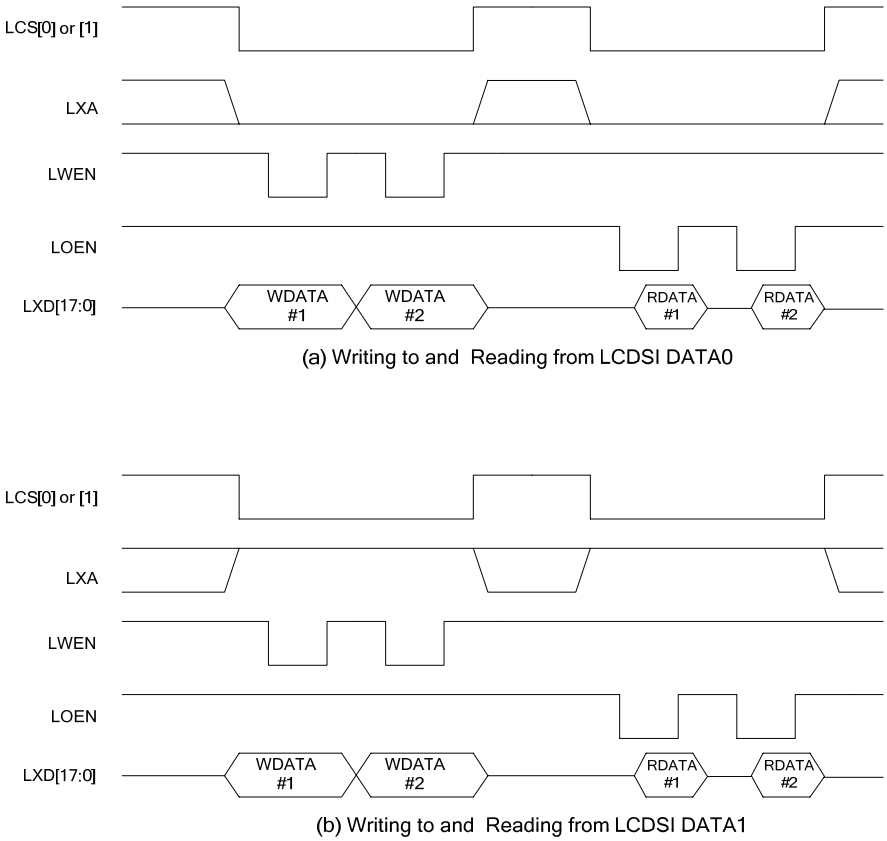


Figure 5.2 Writing / Reading Operation through on-chip CPU

5.2.2 Writing Operation Through LCD Controller

For converting LCDC control signals to 68/80-system interface signals, the LCDC must be configured to TFT mode. And LPXD must be RGB565 or RGB888 and driven at negative edge of LPXCLK. LACBIAS and LVSYNC polarity is also configured by LCDSI LnCTRL.IA and LCDSI LnCTRL.IVS and the values must be same with them of LCDC polarity registers. Refer to LCDC register set for more information.

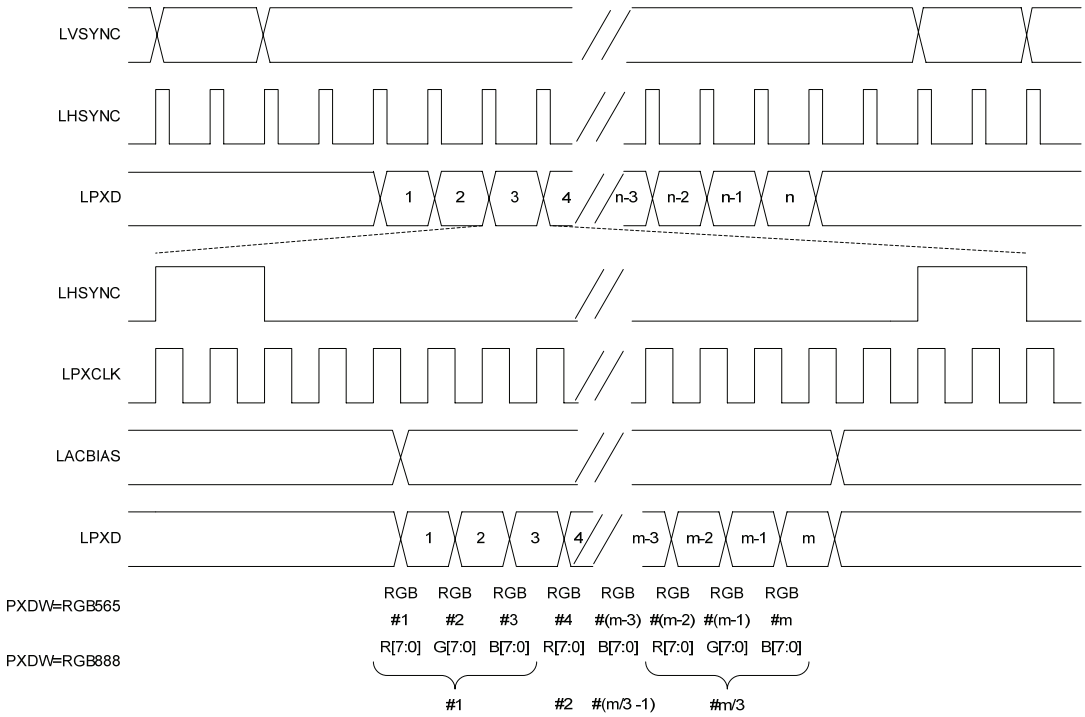


Figure 5.3 Example of LCDC Output Signals for LCDSI

LCDSI LnCTRL.CS, LnCTRL.RSP, LnCTRL.FMT, LnCTRL.IVS, and LnCTRL.IA register fields specify nCS, RS, and output pixel data format during operation and polarity of LVSYNC and LACBIAS signal. For example, if an LCD module is connected to nCS0 and requires RS signal to be low, LCDSI LnCTRL.CS and LCDSI LnCTRL.RSP must be set to 0.

And LCDSI output signals can be adjusted by programming LCDSI CTRL0[31:16] register. LCDXD is dependent on LCDC LPXD, LCDSI LnCTRL.FMT, and LCDSI CTRLn.WBW registers.

The following is the procedure that one frame data of LCDC is sent to LCD module through LCDSI.

(1) Set LCDSI LnCTRL register.
(2) Set the timing parameters for LCD module via LCDSI CTRLn registers.
(3) Set LCDC register for one frame data.
(4) Enable LCDC and Disable it sequentially.

If LCDC is not disabled at 4, LCDC output data are sent to LCD module through LCDSI continuously.

5.3 Register Descriptions

All control registers for LCDSI are listed in Table 5.1.

Table 5.1 LCDSI Register map(0xB0A08000/0xB0A0C000)

Name	Address	Type	Reset	Description
LCDSI CTRL0	0x00	R/W	0xA0229011	Control register for LCSN[0] when LXA=0
LCDSI CTRL1	0x04	R/W	0xA0429021	Control register for LCSN[0] when LXA=1
LCDSI CTRL2	0x08	R/W	0xA0129009	Control register for LCSN[0] when LXA=2
LCDSI CTRL3	0x0C	R/W	0xA0229011	Control register for LCSN[0] when LXA=3
LCDSI CTRL4	0x10	R/W	0xA0229011	Control register for LCSN[1] when LXA=0
LCDSI CTRL5	0x14	R/W	0xA0429021	Control register for LCSN[1] when LXA=1
LCDSI CTRL6	0x18	R/W	0xA0129009	Control register for LCSN[1] when LXA=2
LCDSI CTRL7	0x1C	R/W	0xA0229011	Control register for LCSN[1] when LXA=3
LCDSI BSWAP0	0x20	R/W	0xE4E4E4E4	Byte Swap Control register for LCSN[0] when LXA=0 or LXA=1
LCDSI BSWAP1	0x24	R/W	0xE4E4E4E4	Byte Swap Control register for LCSN[0] when LXA=2 or LXA=3
LCDSI BSWAP2	0x28	R/W	0xE4E4E4E4	Byte Swap Control register for LCSN[1] when LXA=0 or LXA=1
LCDSI BSWAP3	0x2C	R/W	0xE4E4E4E4	Byte Swap Control register for LCSN[1] when LXA=2 or LXA=3
LCDSI LCTRL	0x30	R/W	0x00000020	Control register for Writing operation through LCDC
LCDSI MODE	0x34	R/W	0x00000000	68/80 mode select
LCDSI STATUS	0x38	R	-	LCDSI status
LCDSI DATA0	0x40	R/W	-	If this register is read or written, reading or writing operations are generated on LCSN[0] while LXA = 0.
LCDSI DATA1	0x50	R/W	-	If this register is read or written, reading or writing operations are generated on LCSN[0] while LXA = 1.
LCDSI DATA2	0x60	R/W	-	If this register is read or written, reading or writing operations are generated on LCSN[0] while LXA = 2.
LCDSI DATA3	0x70	R/W	-	If this register is read or written, reading or writing operations are generated on LCSN[0] while LXA = 3.
LCDSI DATA0	0x80	R/W	-	If this register is read or written, reading or writing operations are generated on LCSN[1] while LXA = 0.
LCDSI DATA1	0x90	R/W	-	If this register is read or written, reading or writing operations are generated on LCSN[1] while LXA = 1.
LCDSI DATA2	0xA0	R/W	-	If this register is read or written, reading or writing operations are generated on LCSN[1] while LXA = 2.
LCDSI DATA3	0xB0	R/W	-	If this register is read or written, reading or writing operations are generated on LCSN[1] while LXA = 3.

LCDSI CTRLn **0xB0A08000, 0xB0A08004, 0xB0A08008, 0xB0A0800C**
0xB0A08010, 0xB0A08014, 0xB0A08018, 0xB0A0801C
0xB0A0C010, 0xB0A0C014, 0xB0A0C018, 0xB0A0C01C
0xB0A0C010, 0xB0A0C014, 0xB0A0C018, 0xB0A0C01C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BW[1]	WSTP			WPW									WHLD		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BW[0]	RSTP			RPW									RHLD		

Field	Name	RW	Reset	Description
31	BW[1]	R/W	0x1	**Bus Width** 0 LCDSI DATAn registers become 8 or 16-bit I/O devices. Refer to BW[0] 1 LCDSI DATAn registers become 32-bit I/O devices. Therefore, when the on-chip CPU accesses them with a 32-bit data operation, LCDSI generates 1 32-bit dat operation. But, upper 14bits are truncated because LXD is 18 bits.
30-28	WSTP	R/W	0x2	**Write Setup Time** N cycles are issued between the falling edge of LCSN and the falling edge of LWEN (Writing operation).
27-19	WPW	R/W	0x4/0x8/0x2/0x4 0x4/0x8/0x2/0x4	**Write Pulse Width** (N+1) cycles are issued between the falling edge of LWEN (or LOEN).
18-16	WHLD	R/W	0x2	**Write Hold Time** N cycles are issued between the rising edge of LWEN (or LOEN) and the rising edge of LCSN.
15	BW[0]	R/W	0x1	**Bus Width** This bit valid only when BW[1] = 1. 0 LCDSI DATAn register become 8-bit I/O devices. Therefore, when the on-chip CPU accesses them with a 32-bit data operation, LCDSI generates 4 8-bit data operations. 1 LCDSI DATAn registers become 16-bit I/O devices Therefore, when the on-chip CPU accesses them with 32-bit data operation, LCDSI generates 2 16-bit data operations.
14-12	RSTP	R/W	0x1	**Read Setup Time** N cycles are issued between the falling edge of LCSN and the falling edge of LOEN(Reading operation).
11-3	RPW	R/W	0x2/0x4/0x1/0x2 0x2/0x4/0x1/0x2	**Read Pulse Width** (N+1) cycles are issued between the falling edge of LWEN (or xLOEN) and the rising edge of LOEN.
2-0	RHLD	R/W	0x1	**Read Hold Time** N cycles are issued between the rising edge of LCSN.

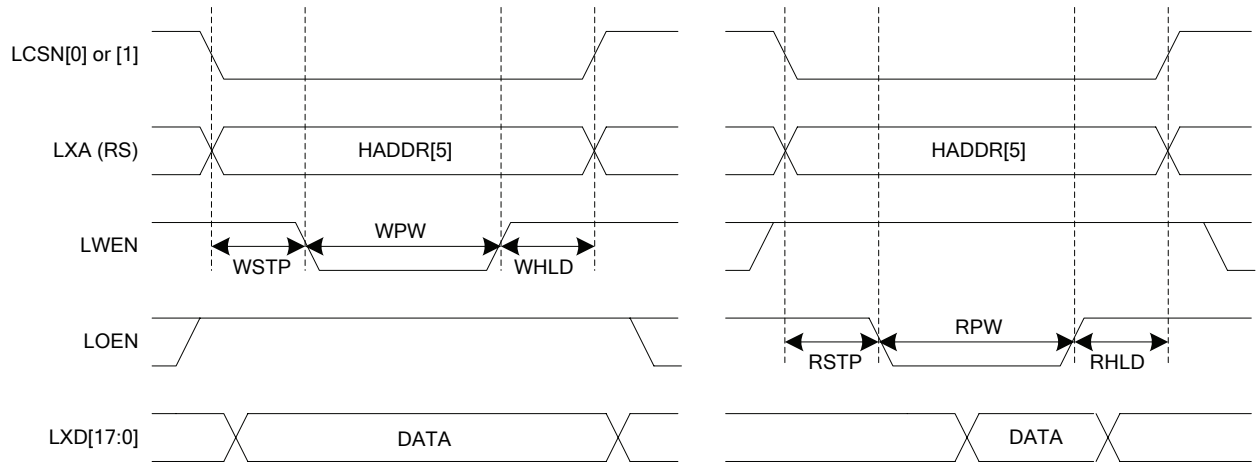


Figure 5.4 Timing Configuration of LCDSI Output Signals for the on-chip CPU Access

LCDSI BSWAP0 — 0xB0A08020/0xB0A0C020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WBS_C0X1								RBS_C0X1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WBS_C0X0								RBS_C0X0							

Field	Name	RW	Reset	Description
31-24	WBS_C0X1	R/W	0xE4E4	**Write Channel Byte Swap when C=0 and X=1** Refer Figure 5.5
23-16	RBS_C0X1	R/W	0xE4E4	**Read Channel Byte Swap when C=0 and X=1** Refer Figure 5.6
15-8	WBS_C0X0	R/W	0xE4E4	**Write Channel Byte Swap when C=0 and X=0** Refer Figure 5.5
7-0	RBS_C0X0	R/W	0xE4E4	**Read Channel Byte Swap when C=0 and X=0** Refer Figure 5.6

LCDSI BSWAP1 — 0xB0A08024/0xB0A0C024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WBS_C0X3								RBS_C0X3							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WBS_C0X2								RBS_C0X2							

Field	Name	RW	Reset	Description
31-24	WBS_C0X3	R/W	0xE4E4	**Write Channel Byte Swap when C=0 and X=3** Refer Figure 5.5
23-16	RBS_C0X3	R/W	0xE4E4	**Read Channel Byte Swap when C=0 and X=3** Refer Figure 5.6
15-8	WBS_C0X2	R/W	0xE4E4	**Write Channel Byte Swap when C=0 and X=2** Refer Figure 5.5
7-0	RBS_C0X2	R/W	0xE4E4	**Read Channel Byte Swap when C=0 and X=2** Refer Figure 5.6

LCDSI BSWAP2 — 0xB0A08028/0xB0A0C028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WBS_C1X1								RBS_C1X1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WBS_C1X0								RBS_C1X0							

Field	Name	RW	Reset	Description
31-24	WBS_C1X1	R/W	0xE4E4	**Write Channel Byte Swap when C=1 and X=1** Refer Figure 5.5
23-16	RBS_C1X1	R/W	0xE4E4	**Read Channel Byte Swap when C=1 and X=1** Refer Figure 5.6
15-8	WBS_C1X0	R/W	0xE4E4	**Write Channel Byte Swap when C=1 and X=0** Refer Figure 5.5
7-0	RBS_C1X0	R/W	0xE4E4	**Read Channel Byte Swap when C=1 and X=0** Refer Figure 5.6

LCDSI BSWAP3 **0xB0A0802C/0xB0A0C02C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WBS_C1X3								RBS_C1X3							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WBS_C1X2								RBS_C1X2							

Field	Name	RW	Reset	Description
31-24	WBS_C1X3	R/W	0xE4E4	**Write Channel Byte Swap when C=1 and X=3** Refer Figure 5.5
23-16	RBS_C1X3	R/W	0xE4E4	**Read Channel Byte Swap when C=1 and X=3** Refer Figure 5.6
15-8	WBS_C1X2	R/W	0xE4E4	**Write Channel Byte Swap when C=1 and X=2** Refer Figure 5.5
7-0	RBS_C1X2	R/W	0xE4E4	**Read Channel Byte Swap when C=1 and X=2** Refer Figure 5.6

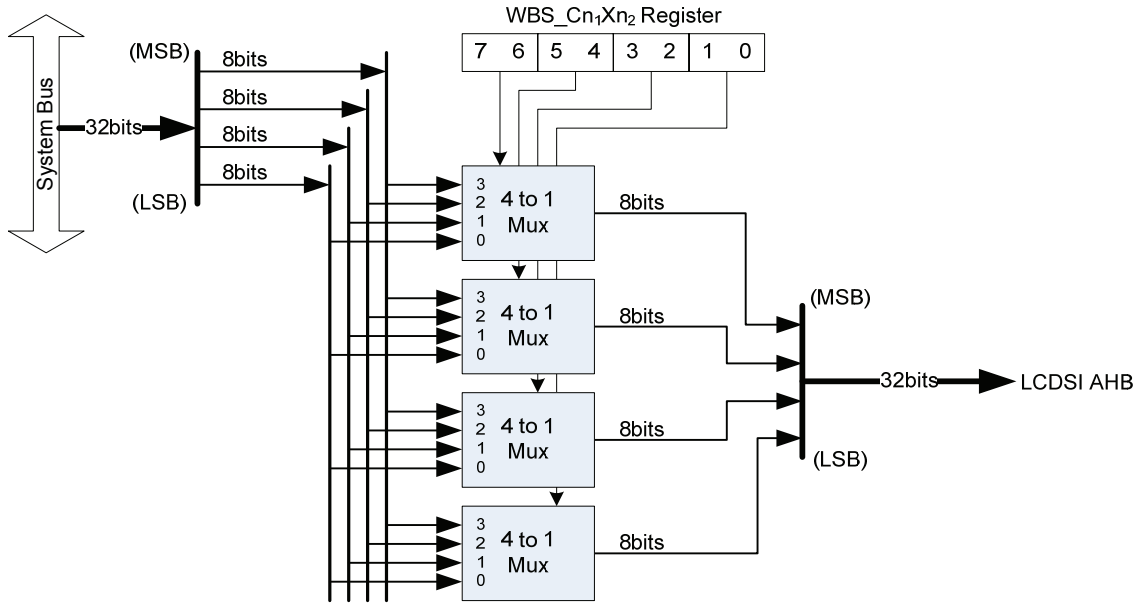


Figure 5.5 LCDSI (CPU to SI) Write Path

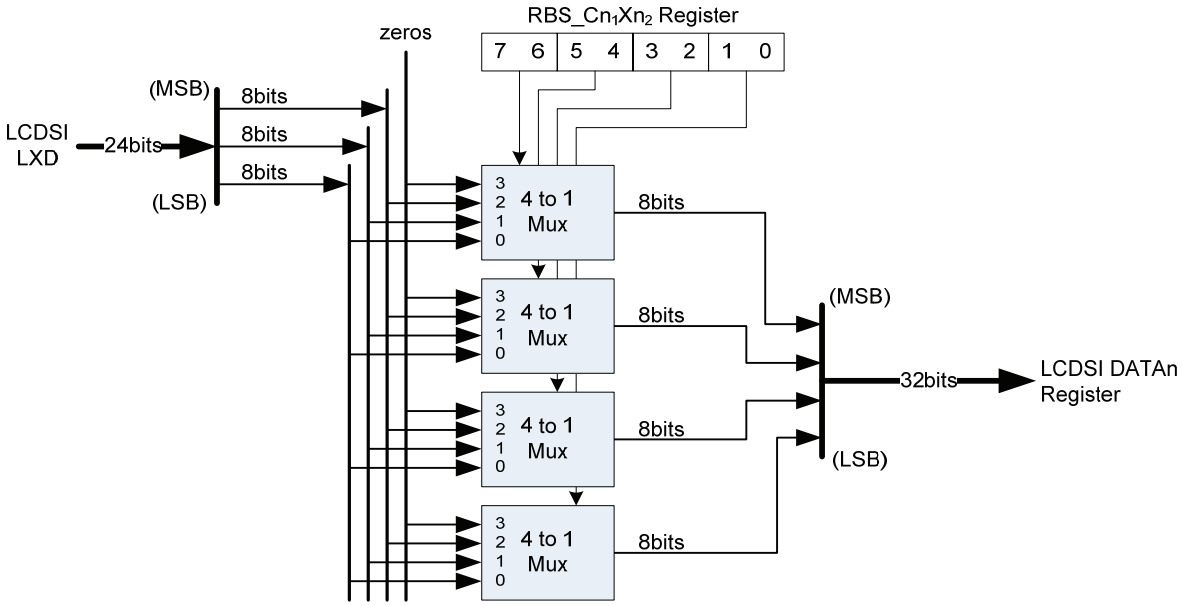


Figure 5.6 LCDSI (SI to Register) Read Path

LCDSI LCTRL

0xB0A08030/0xB0A0C030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IA	IVS	0					CS	XA		FMT				LSEL	EN

Field	Name	RW	Reset	Description
15	IA	R/W	-	**Inverse ACBIAS** 0 LACBIAS(Data Enable) signal of LCDC is active high. 1 LACBIAS(Data Enable) signal of LCDC is active low.
14	IVS	R/W	-	**Inverse VSYNC** 0 LVSYNC signal of LCDC is active high. 1 LVSYNC signal of LCDCL is active low.
8	CS	R/W	-	**Chip select** 0 LCSN[0] is used. 1 LCSN[1] is used.
7-6	XA	R/W	-	**Address select** 0 LXA (RS) is 2'b00 while LCSN[0] or LSN[1] is asserted. 1 LXA (RS) is 2'b01 while LCSN[0] or LSN[1] is asserted. 2 LXA (RS) is 2'b10 while LCSN[0] or LSN[1] is asserted. 3 LXA (RS) is 2'b11 while LCSN[0] or LSN[1] is asserted.
5-2	FMT	R/W	-	**Pixel Data Format** 0 8bit format LCDSI[7:0] <= LPD[7:0] 1~4 Reserved 5 32bit format LCDSI[11:0] <= LPD[23:12] LCDSI[15:12] <= 0 LCDSI[27:16] <= LPD[11:0] LCDSI[31:28] <= 0 6 32bit format LCDSI[7:0] <= LPD[23:16] LCDSI[15:8] <= 0 LCDSI[31:16] <= LPD[15:0] 7 32bit format LCDSI[15:0] <= LPD[23:8] LCDSI[23:16] <= LPD[7:0] LCDSI[31:24] <= 0 8 32bit format LCDSI[17:0] <= LPD[17:0] LCDSI[31:18] <= 0 9 32bit format LCDSI[15:0] <= LPD[17:2] LCDSI[17:16] <= LPD[1:0] LCDSI[31:18] <= 0 10 32bit format LCDSI[17:0] <= LPD[17:0] LCDSI[31:18] <= 0 11 32bit format LCDSI[8:0] <= LPD[8:0] LCDSI[15:9] <= 0 LCDSI[24:16] <= LPD[17:9] LCDSI[31:25] <= 0 12 32bit format LCDSI[15:0] <= LPD[15:0] LCDSI[31:16] <= 0 13 16bit format LCDSI[7:0] <= LPD[15:8] LCDSI[15:8] <= LPD[7:0] 14 16bit format LCDSI[15:0] <= LPD[15:0]
1	LSEL	R/W	-	**LCD Controller Select** 0 LCDC0 1 LCDC1

Field	Name	RW	Reset	Description
0	EN	R/W	-	**Enable** 0 Disable 1 Enable

FMT	LCDSI CTRLn.BW	LCDSI Output (LPXD)
32bit format	32bit (BW = 2 or 3)	1st Data : XD[23:0] = LCDSI[23:0]
	16bit (BW = 1)	1st Data : XD[15:0] = LCDSI[15:0] 2nd Data : XD[15:0] = LCDSI[31:16]
	8bit (BW = 0)	1st Data : XD[7:0] = LCDSI[7:0] 2nd Data : XD[7:0] = LCDSI[15:8] 3rd Data : XD[7:0] = LCDSI[23:16] 4th Data : XD[7:0] = LCDSI[31:24]
16bit format	32bit (BW = 2 or 3)	1st Data : XD[23:0] = LCDSI[15:0]
	16bit (BW = 1)	1st Data : XD[15:0] = LCDSI[15:0]
	8bit (BW = 0)	1st Data : XD[7:0] = LCDSI[7:0] 2nd Data : XD[7:0] = LCDSI[15:8]
8bit format	32bit (BW = 2 or 3)	1st Data : XD[23:0] = LCDSI[7:0]
	16bit (BW = 1)	1st Data : XD[15:0] = LCDSI[7:0]
	8bit (BW = 0)	1st Data : XD[7:0] = LCDSI[7:0]

LCDSI MODE 0xB0A08034/0xB0A0C034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0															MODE

Field	Name	RW	Reset	Description
0	MODE	R/W	0	LCDSI MODE 0 80-type system interface 1 68-type system interface

LCDSI STATUS 0xB0A08038/0xB0A0C038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0															OVR

Field	Name	RW	Reset	Description
0	OVR	R/W	-	**LCD Overrun** LCDSI input FIFO overflow 0 Normal 1 LCDC overrun

6 Memory To Memory Scaler

6.1 Overview

The block diagram of the Memory to Memory Scaler (MSC) is shown in the following figure. The hardware reads the source image and resizes it and finally writes the scaled image to the destination region. Specially, the hardware can interface to the LCD controller directly in progressive display output mode.

The maximum resolution of the MSC controller 0 and 1 is limited 4088x4088 and 2032x2032. And the output buffer sizes which can be used as the interface buffer for the on-the-fly mode are 512bytes and 4096bytes correspondingly. In the case of MSC controller 0, it can be a cause of the under-run to display with LCDC in the on-the-fly path because the output buffer is small relative to the display size."

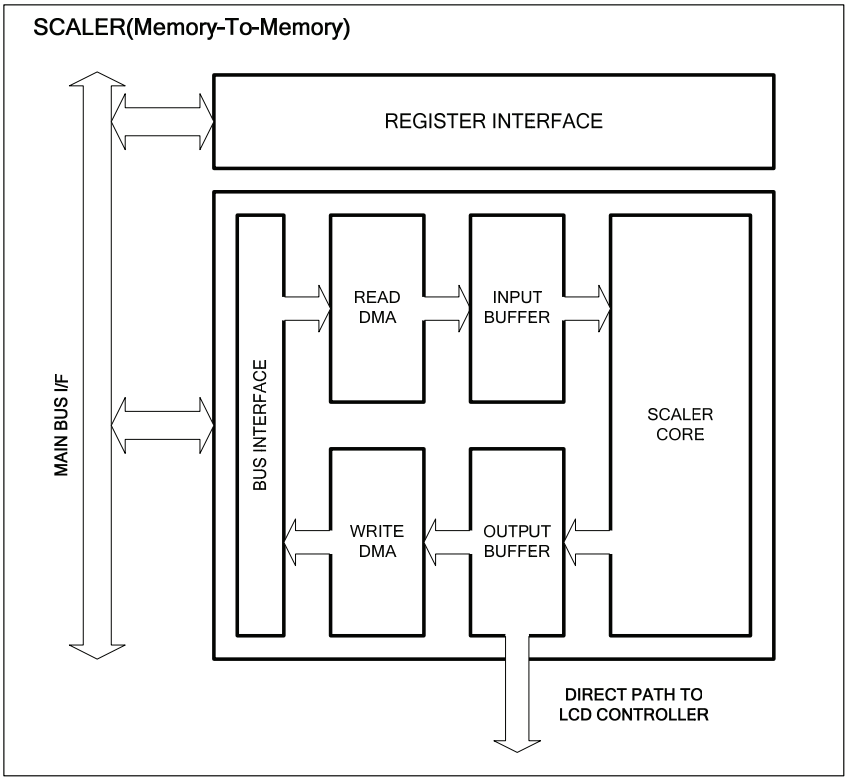


Figure 6.1 Scaler Block Diagram

6.2 Operation

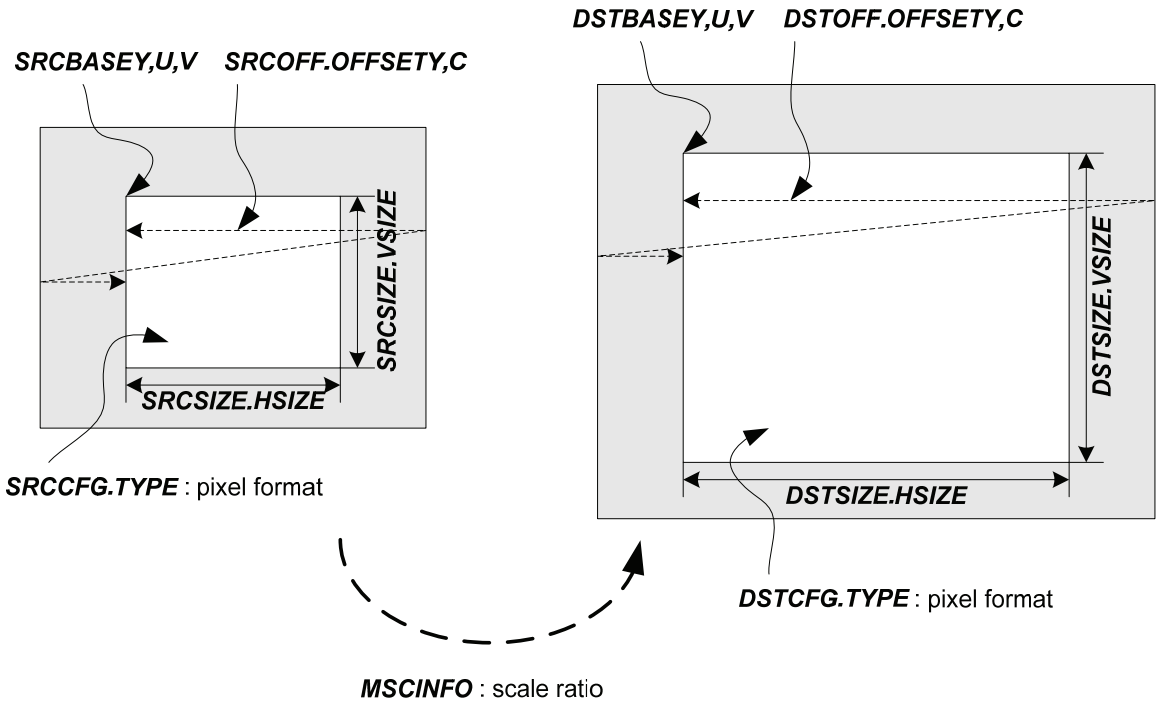


Figure 6.2 Memory to Memory Scaling Operation

The MSC reads the source image in the memory and resizes it and then stores the result image to the specified memory space or send it to the LCDC. Figure 6.2 shows that memory-to-memory scaling operation and the corresponding registers. The MSC has two modes to store the result image to the memory. One is the frame mode. The other is the rolling mode. The frame mode needs as much memory space as the whole result image size (Figure 6.3(a)). But, the rolling mode needs as much memory space as region which is specified by the rolling lines (Figure 6.3(b)).

In case of rolling mode, there are two methods the MSC informs the on-chip CPU the current status. One is to use the rolling line trigger level; the other is to use the middle line trigger level. Whenever the MSC stores one line to be scaled, the current rolling line counter (CRCNT register) increases by 1. When this counter reaches the rolling line trigger level (ROLLCNT of DSTRMCNT register), it is reset to 0 and addresses which specifies the location to store the result image return to destination base addresses (DSTBASEY, DSTBASEU, and DSTBASEV) and the MSC can issue the interrupt request. The MSC can also inform the on-chip CPU how many lines are stored from the destination base address. When the current rolling line counter reaches the middle line trigger level (MIDCNT of DSTRMCNT register), the MSC can issue the interrupt request. But, in this case, the rolling line counter is not reset to 0 and address does not return to the destination base address.

Additionally, the MSC can make scaling operation stopped temporarily. The middle line trigger level and the rolling line trigger level are used as the indicator. Whenever the current rolling line counter reaches middle line and/or the rolling line trigger level, the MSC pauses in the operation. To resume the operation, RLS bit of MSCCTR register is set to 1. To enable this function, RGSM and MGSM of MSCCTR register should be set to 1. RGSM is for the rolling line trigger level and MGSM is for the middle line trigger level.

To enable the rolling mode, REN bit of MSCCTR register should be set to 1. The rolling line trigger level is determined by ROLLCNT of DSTRMCNT register. To enable the middle line trigger level, MEN and REN bit of MSCCTR register should be set to 1 and the middle line trigger level is determined by MIDCNT of DSTRMCNT register. And to enable the corresponding interrupt, RIQREN and MIRQEN should be set to 1. They are for the rolling line trigger level interrupt and the middle line trigger level interrupt, respectively.

Finally, when the whole result image is stored to the memory completely, MSCSTR.BUSY bit is cleared to 0 and MSCSTR.RDY bit is set to 1. And these can be used as an interrupt request source.

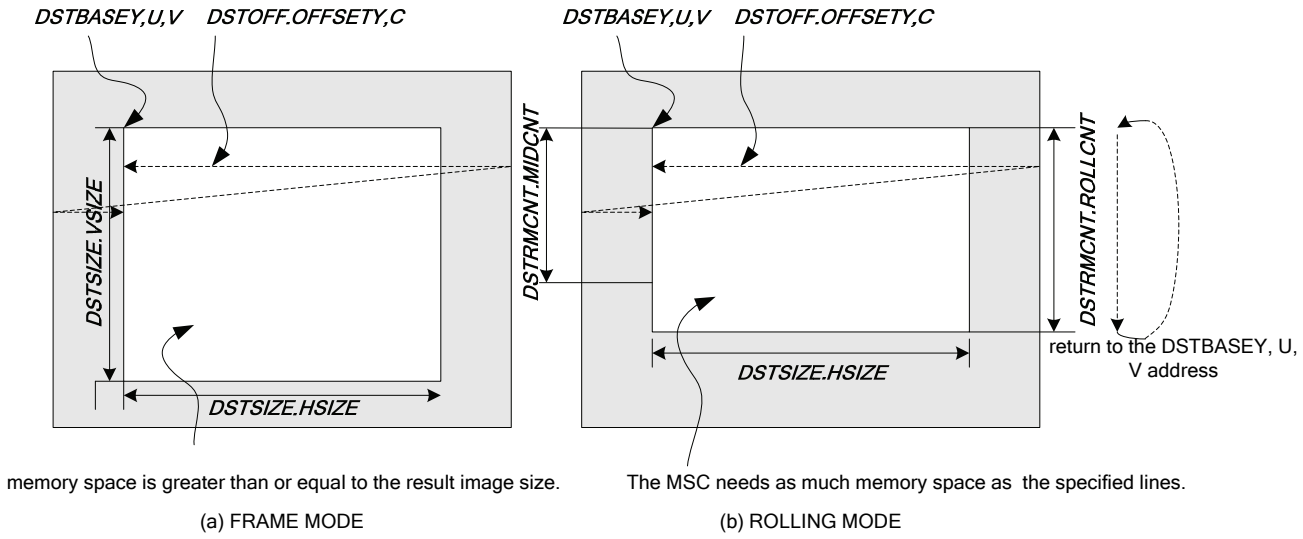


Figure 6.3 Storing the Result Image

6.3 Registers

Table 6.1 Scaler Registers (Base Address = 0xB0A10000/0xB0A2000)

Name	Addr	Type	Reset	Description
SRCBASEY	0x000	R/W	0x00000000	Scaler source base address for Y
SRCBASEU	0x004	R/W	0x00000000	Scaler source base address for U (Cb)
SRCBASEV	0x008	R/W	0x00000000	Scaler source base address for V (Cr)
SRCSIZE	0x00c	R/W	0x00000000	Source image size register
SRCOFF	0x010	R/W	0x00000000	Source image line offset register
SRCCFG	0x014	R/W	0x00000000	Source image configuration register
DSTBASEY	0x020	R/W	0x00000000	Scaler destination base address for Y
DSTBASEU	0x024	R/W	0x00000000	Scaler destination base address for U (Cb)
DSTBASEV	0x028	R/W	0x00000000	Scaler destination base address for V (Cr)
DST_SIZE	0x02c	R/W	0x00000000	Destination image size register
DSTOFF	0x030	R/W	0x00000000	Destination image line offset register
DSTCFG	0x034	R/W	0x00000000	Destination image configuration register
MSCINF	0x040	R/W	0x00000000	Scaling information register
MSCCTR	0x044	R/W	0x00000000	Scaler control register
MSCSTR	0x048	R/W	0x00000000	Scaler status register
HSTROBE	0x4C	R/W	0x000A0002	Horizontal Strobe Timing Control Register
DSTRMCNT	0x050	R/W	0x00000000	Destination Rolling Count Register
CRCNT	0x054	R	0x00000000	Destination Rolling Status Register
CLIP0	0x58	R/W	0x00ff00ff	RGB-to-YCbCr Clipping Configuration Register 0
CLIP1	0x5C	R/W	0x000000ff	RGB-to-YCbCr Clipping Configuration Register 1
VSTROBE	0x60	R/W	0x0000000a	Vertical Strobe Timing Control Register

The scaler post filter is used to sharpen the output image of the scaler. Note that the denoise 2D in the VIQE and this filter cannot be used simultaneously.

Table 6.2 Scaler Post Filter Register Map (Base Address = 0XB0A5B000)

Name	Offset	RW	Reset	Notes
PARAM0	0x40	R/W	0x12C83C28	Filter Parameters
PARAM1	0x44	R/W	0x743C0238	
PARAM2	0x48	R/W	0xFF408081	
PARAM3	0x4C	R/W	0x003801FC	
PARAM4	0x50	R/W	0x12641E14	
PARAM5	0x54	R/W	0x582D011C	
PARAM6	0x58	R/W	0xFF408080	
PARAM7	0x5C	R/W	0x002B01FE	
Reserved	0x60	R	0x00000000	These registers should not be changed.
	0x64	R/W	0x00000000	
	0x68	R	0x00000000	
	0x6C	R/W	0x00000000	
	0x70	R/W	0x00000160	
CTRL	0x74	R/W	0x00000080	Control register
SIZE	0x78	R/W	0x00000000	Image size of the output image of the scaler
Reserved	0x7C	R/W	0x00040000	This register should not be changed.

SRC Image Y Base Address (SRCBASEY) 0xB0A1/0xB0A20000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRCBASEY[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRCBASEY[15:2]															

Field	Name	RW	Reset	Description
31-2	SRCBASEY	R/W	-	Source Base Address Y for 4:2:0 or 4:2:2 Separate Mode Y for 4:2:2 Sequential Mode Y for 4:2:0 or 4:2:2 Interleaved Mode

SRC Image U Base Address (SRCBASEU) 0xB0A1/0xB0A20004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRCBASEU[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRCBASEU[15:2]															

Field	Name	RW	Reset	Description
31-2	SRCBASEU	R/W	-	Source Base Address for U Cb/Cr for 4:2:0 or 4:2:2 Interleaved Mode SRCBASEU[31:30] and SRCBASEY[31:30] should be the same value.

SRC Image V Base Address (SRCBASEV) 0xB0A1/0xB0A20008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SRCBASEV [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SRCBASEV[15:2]															

Field	Name	RW	Reset	Description
31-2	SRCBASEV	R/W	-	Source Base Address for V SRCBASEV[31:30] and SRCBASEY[31:30] should be the same value.

SRC Image Size (SRCSIZE) 0xB0A1/0xB0A2000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				VSIZE[11:0]											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				HSIZE[11:0]											

Field	Name	RW	Reset	Description
27-16	VSIZE	R/W	0	Vertical Image Size by pixel unit
11-0	HSIZE	R/W	0	Horizontal Image Size by pixel unit

SRC Image Offset (SRCOFF) 0xB0A1/0xB0A20010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				OFFSETC[11:0]											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				OFFSETY[11:0]											

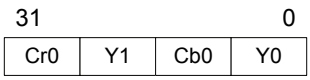
Field	Name	RW	Reset	Description
27-16	OFFSETC	R/W	0x0	Chrominance Address Offset
11-0	OFFSETY	R/W	0x0	Luminance Address Offset

SRC Image Configuration (SRCCFG0/1)

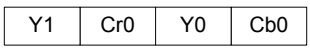
0xB0A1/B0A20014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSBF															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					WAITCNT				READY	INTM	INPATH	INTLV	TYPE		

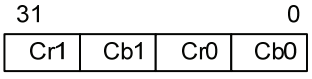
Field	Name	RW	Reset	Description
31	MSBF	R/W	0x00000000	When RGB color used, the LSB is extended by 0: zero 1: MSB(Most Significant Bit) * The RGB color was extended to 8bits and converted to YCbCr color.
10-8	WAITCNT	R/W	0x00000000	Ready Cycle Count Register ***This is invalid for this processor – should be zero***.
6	READY	R/W	0x00000000	Ready Control Register ***This is invalid for this processor – should be zero***.
5	INTM	R/W	0x00000000	Interleaved mode cb/cr order 0: Mode 0 1: Mode 1
4	INPATH	R/W	0x00000000	Input Data path Configuration Register ***This is invalid for this processor – should be zero***.
3	INTLV	R/W	0x00000000	1: TYPE=2 or 3 YUV420 interleaved YUV422 interleaved 0: TYPE=2 or 3 YUV420 separate YUV422 separate
2-0	TYPE	R/W	0x00000000	7 :RGB454 6: RGB444 5: RGB555 4: RGB565 3 : YUV420 separate 2 : YUV422 separate 1 : YUV422 sequential mode 1 0 : YUV422 sequential mode 0



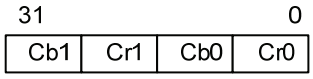
YCbCr4:2:2 Sequential Mode 0



YCbCr4:2:2 Sequential Mode 1



4:2:2/4:2:0 Chrominance Interleaved Mode 0



4:2:2/4:2:0 Chrominance Interleaved Mode 1

DST Image Y Base Address (DSTBASEY) 0xB0A1/0xB0A20020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DSTBASEY[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSTBASEY[15:2]															

Field	Name	RW	Reset	Description
31-2	DSTBASEY	R/W	-	Destination Base Address Y for 4:2:0 or 4:2:2 Separate Mode Y for 4:2:2 Sequential Mode Y for 4:2:0 or 4:2:2 Interleaved Mode

DST Image U Base Address (DSTBASEU) 0xB0A1/0xB0A20024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DSTBASEU[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSTBASEU[15:2]															

Field	Name	RW	Reset	Description
31-2	DSTBASEU	R/W	-	Destination Base Address for U Cb/Cr for 4:2:0 or 4:2:2 Interleaved Mode DSTBASEU[31:30] and DSTBASEY[31:30] should be the same value.

DST Image V Base Address (DSTBASEV) 0xB0A1/0xB0A20028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DSTBASEV[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSTBASEV[15:2]															

Field	Name	RW	Reset	Description
31-2	DSTBASEV	R/W	-	Destination Base Address for V DSTBASEV[31:30] and DSTBASEY[31:30] should be the same value.

DST Image Size (DSTSIZE) 0xB0A1/0xB0A2002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				VSIZE[11:0]											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				HSIZE[11:0]											

Field	Name	RW	Reset	Description
27-16	VSIZE	R/W	0	Vertical Image Size by pixel unit
11-0	HSIZE	R/W	0	Horizontal Image Size by pixel unit

DST Image Offset (DSTOTFF) 0xB0A1/0xB0A20030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				OFFSETC[11:0]											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				OFFSETY[11:0]											

Field	Name	RW	Reset	Description
27-16	OFFSETC	R/W	0x0	Chrominance Address Offset
11-0	OFFSETY	R/W	0x0	Luminance Address Offset

DST Image Configuration (DSTCFG) 0xB0A1/0xB0A20034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				COP	WAITCNT				RDY	INTM	PATH	INTLV	TYPE		

Field	Name	RW	Reset	Description
11	COP	R/W	0x0	Chrominance Writing Mode Register Defined for 4:2:0 Separate Mode '0' : Y0→U0→V0→Y1→Y2→U1 … '1' : Y0→U0→Y1→V0→Y2→U1 …
10-8	WAITCNT	R/W	0x0	Wait Cycle Count for RDY being '1'
6	RDY	R/W	0x0	Access Wait Control Register Valid for PATH being '1' '0' : Wait for "WAITCNT+1" cycles for LCD Access '1' : Wait until Output FIFO is not empty
5	INTM	R/W	0x0	Interleaved mode cb/cr order 0: Mode 0 1: Mode 1
4	PATH	R/W	0x0	Destination Type Register '0' : Memory The scaled image is written to the destination memory. '1' : LCD Channel 0 Direct Path The scaled image is written to the LCDC for IMG0.
3	INTLV	R/W	0x0	1: TYPE=2 or 3 YUV420 interleaved YUV422 interleaved 0: TYPE=2 or 3 YUV420 separate YUV422 separate
2-0	TYPE	R/W	0x0	Destination Type Register 7 :RGB454 6: RGB444 5: RGB555 4: RGB565 3 : YUV420 separate 2 : YUV422 separate 1 : YUV422 sequential mode 1 0 : YUV422 sequential mode 1 * Byte Sequence is same as SRCCFG

MSC Information (MSCINFO) 0xB0A1/0xB0A20040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		VRATIO[13:0]													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
		HRATIO[13:0]													

Field	Name	RW	Reset	Description
29-16	VRATIO	R/W	0x0	Vertical scale ratio VRATIO = 256 * SRCSIZE.VSIZE/DSTSIZE.VSIZE
13-0	HRATIO	R/W	0x0	Horizontal scale ratio HRATIO = 256 * SRCSIZE.HSIZE/DSTSIZE.HSIZE

MSC Control (MSCCTR) 0xB0A1/0xB0A20044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	TST		CKG				INPATH	REN	MEN			RLS		RGSM	MGSM
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R2YMD		Y2RMD		R2YCEN		OUIEN	RST	RIRQEN	MIRQEN	CONT	BP		BUSY	RDY	EN

Field	Name	RW	Reset	Description
30	TST	R/W	0x0	Should be zero for test purpose
28	CKG	R/W	0x0	Clock Gating Enable Signal 0b : Gating Enable 1b : Gating Disabled * This should be “ZERO”
24	INPATH	R/W	0x0	Read Path Configuration Register 0 : Read from Memory 1 : Read from On-the-fly Path
23	REN	R/W	0x0	Enable the rolling mode 0b : disable (frame mode) 1b : enable (rolling mode)
22	MEN	R/W	0x0	Enable the middle line counter 0b : disable 1b : enable
19	RLS	R/W	0x0	Release Stop Mode When scaling operation is stopped by “rolling line counter” or “middle line counter”, it is resumed by writing 1 to this bit. This bit is automatically cleared.
17	RGSM	R/W	0x0	Enable Stop Mode using the rolling line counter 0b : disable 1b : enable (REN should be also set to 1)
16	MGSM	R/W	0x0	Enable Stop Mode using the middle line counter 0b : disable 1b : enable (MEN and REN should be also set to 1)
15-14	R2YMD	R/W	0x0	RGB-to-YCbCr Conversion Mode
13-12	Y2RMD	R/W	0x0	YCbCr-to-RGB Conversion Mode
11	R2YCEN	R/W	0x0	Clipping Enable for RGB-to-YCbCr Conversion 0 : Disabled 1 : Enabled * Normally, ‘0’ is recommended.
9	OUIEN	R/W	0x0	Output Underrun IRQ Enable 0 : Disabled 1 : Enabled
8	RST	R/W	0x0	Internal state machine Reset Signal 0b : Not Reset 1b : Reset
7	RIRQEN	R/W	0x0	Enable an interrupt using the rolling line counter 0b : disable 1b : enable
6	MIRQEN	R/W	0x0	Enable an interrupt using the middle line counter

				0b : disable 1b : enable
5	CONT	R/W	0x0	Continuous Mode 0 : One-Time Mode 1 : Continuous Mode **** If the output of the scaler connected to On-The-Fly and the connected device requires continuously such as LCD controller, this should be "1".***
4	BP	R/W	0x0	Bypass Enable Signal (Test Purpose) 0b : Not-bypassed (Normal Operation) 1b : Bypassed * This should be "ZERO"
2	BUSY	R/W	0x0	1b : Busy Interrupt Enable
1	RDY	R/W	0x0	1b : Ready Interrupt Enable
0	EN	R/W	0x0	Enable Register 0b : Operation Disabled 1b : Operation Enabled

Y2RMD [10:9]	YCbCr to RGB Conversion Option
0	**** R = Y + 1.371 * (Cr – 128)*** **** G = Y + 0.336 * (Cb – 128) – 0.698 * (Cr – 128)*** **** B = Y + 1.732 * (Cb – 128)*** The range for "***YCbCr***" is 16 ~ 235, "***Studio Color***". The result is "***Studio Color***" – Normally SDTV.
1	**** R = 1.164 * (Y – 16) + 1.596 * (Cr – 128)*** **** G = 1.164 * (Y – 16) – 0.391 * (Cb – 128) – 0.813 * (Cr – 128)*** **** B = 1.164 * (Y – 16) + 2.018 * (Cb – 128)*** The range for "***YCbCr***" is 16 ~ 235, "***Studio Color***". The result is "***Computer System Color***" – Normally SDTV.
2	**** R = Y + 1.540 * (Cr – 128)*** **** G = Y – 0.183 * (Cb – 128) – 0.459 * (Cr – 128)*** **** B = Y + 1.816 * (Cb – 128)*** The range for "***YCbCr***" is 16 ~ 235, "***Studio Color***". The result is "***Studio Color***" – Normally HDTV.
3	**** R = 1.164 * (Y – 16) + 1.793 * (Cr – 128)*** **** G = 1.164 * (Y – 16) – 0.213 * (Cb – 128) – 0.534 * (Cr – 128)*** **** B = 1.164 * (Y – 16) + 2.115 * (Cb – 128)*** The range for "***YCbCr***" is 16 ~ 235, "***Studio Color***". The result is "***Computer System Color***" – Normally HDTV.

Notice !!! :
The coefficients of the above table are approximated for fixed point calculation.

R2YMD [29:28]	RGB to YCbCr Conversion Option
0	**** Y = 0.299*R + 0.587G + 0.114B*** **** Cb = -0.172*R – 0.339*G + 0.511*B + 128*** **** Cr = 0.511*R – 0.428*G – 0.083*B + 128*** The range for "***RGB***" is 16 ~ 235, "***Studio Color***". The result is "***Studio Color***" – Normally SDTV.
1	**** Y = 0.257*R + 0.504*G + 0.098*B + 16*** **** Cb = -0.148*R – 0.291*G + 0.439*B + 128*** **** Cr = 0.439*R – 0.368*G – 0.071*B + 128*** The range for "***RGB***" is 0 ~ 255, "***Computer System Color***" The result is "***Studio Color***" – Normally SDTV.
2	**** Y = 0.213*R + 0.715*G + 0.072*B*** **** Cb = -0.117*R – 0.394*G + 0.511*B + 128*** **** Cr = 0.511*R – 0.464*G – 0.047*B + 128*** The range for "***RGB***" is 16 ~ 235, "***Studio Color***".. The result is "***Studio Color***" – Normally HDTV.
3	**** Y = 0.183*R + 0.614*G + 0.062*B + 16*** **** Cb = -0.101*R – 0.338*G + 0.439*B + 128*** **** Cr = 0.439*R – 0.399*G – 0.040*B + 128*** The range for "***RGB***" is 0 ~ 255, "***Computer System Color***". The result is "***Studio Color***" – Normally HDTV.

Notice !!! :
The coefficients of the above table are approximated for fixed point calculation.

MSC Status (MSCSTR) **0xB0A1/0xB0A20048**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
											SCE	SCB	STS		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
							IOU	IR	IM	IBUSY	IRDY			BUSY	RDY

Field	Name	RW	Reset	Description
20	SCE	R	0x0	Scaler Error Flag (Test Purpose)
19	STB	R	0x0	Scaler Core Busy Flag (Test Purpose)
18-16	STS	R	0x0	Ready Status Signals (Test Purpose) [16] : Output Port Ready Status [17] : Input Port Ready Status [18] : Scaler Interface Ready Status
8	IOU	R/W	0	Output Underrun Interrupt Flag By writing '1', it would be cleared.
7	IM	R/W	0	Middle line interrupt Flag By writing '1', it would be cleared.
6	IR	R/W	0	Rolling line interrupt Flag By writing '1', it would be cleared.
5	IBUSY	R/W	0x0	Busy Interrupt Flag By writing '1', it would be cleared.
4	IRDY	R/W	0x0	Ready Interrupt Flag By writing '1', it would be cleared.
1	BUSY	R	0x0	Busy Status Register 0b : Not-Busy Status 1b : Busy Status
0	RDY	R	0x0	Ready Status Register 0b : Not-Ready 1b : Ready

Horizontal Strobe Timing Control Register (HSTROBE) 0xB0A1/0xB0A20048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HEN	HWAIT														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PEN									PWAIT						

Field	Name	RW	Reset	Description
31	HEN	R/W	0x0	Cycles are Determined 0 : Automatically Controlled by H/W 1 : Controlled by HWAIT In case of HEN = '0', the WaitCycle is 0x20 In case of HEN = '1', the WaitCycle is HWAIT.
30-16	HWAIT	R/W	0xA	Number of Cycles from End-of-Line to Start-of-Next-Line
15	PEN	R//W	0x0	Pixel-to-Pixel Wait Cycle Control Enable 0b : Automatically Controlled by H/W 1b : Controlled by PWAIT In case of PEN = '0', the WaitCycle is 1. 256 <= HSCALE : 1 1. 128 <= HSCALE <256 : 1 2. 64 <= VSCALE < 127 : 3 3. 32 <= VSCALE < 64 : 7 4. 16 <= VSCALE < 32 : 15 5. 0 <= VSCALE < 16 : INVALID In case of PEN = '1', the WaitCycle is PWAIT.
6-0	PWAIT	R//W	2	Number of Cycles from Current Pixel to Next Pixel

Vertical Strobe Timing Control Register (VSTROBE) 0xB0A1/0xB0A20060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VEN															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VWAIT															

Field	Name	RW	Reset	Description
31	VEN	R/W	0x0	Frame-to-Frame Wait Cycle Control Enable 0b : Automatically Controlled by H/W 1b : Controlled by V2VWAIT In case of VEN = '0', the WaitCycle is 1. 256 <= VSCALE : SRCHSIZE 2. 192 <= VSCALE < 256 : SRCHSIZE * 1.5 3. 128 <= VSCALE < 192 : SRCHSIZE * 2 4. 96 <= VSCALE < 128 : SRCHSIZE * 3 5. 64 <= VSCALE < 96 : SRCHSIZE * 4 6. 32 <= VSCALE < 64 : SRCHSIZE * 8 7. 16 <= VSCALE < 32 : SRCHSIZE * 16 8. 0 <= VSCALE < 16 : Not-Supported In case of VEN = '1', the WaitCycle is VWAIT.
15-0	VWAIT	R/W	0xA	Number of Cycles from Current Frame to Next Frame

DST Rolling/Middle Line Count (DSTRMCNT) 0xB0A1/0xB0A20050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				ROLLCNT [11:0]											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				MIDCNT[11:0]											

Field	Name	RW	Reset	Description
27-16	ROLLCNT	R/W	0x0	-
11-0	MIDCNT	R/W	0x0	-

Current Rolling Count (CRCNT) 0xB0A1/0xB0A20054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				C_RCNT[11:0]											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															

Field	Name	RW	Reset	Description
27-16	C_RCNT	R	0x0	Number of lines that are stored the result image from the base address.

RGB-to-YCbCr Conversion Clipping Register (CLIP0) 0xB0A1/0xB0A20058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLIPCBL								CLIPCBU							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLIPYL								CLIPYU							

Field	Name	RW	Reset	Description
31-24	CLIPCBL	R/W	0x00	Lower bound for the Cb
23-16	CLIPCBU	R/W	0xFF	Upper bound for the Cb
15-8	CLIPYL	R/W	0x00	Lower bound for the Y
7-0	CLIPYU	R/W	0xFF	Upper bound for the Y

RGB-to-YCbCr Conversion Clipping Register (CLIP1) 0xB0A1/0xB0A2005C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLIPCRL								CLIPCRU							

Field	Name	RW	Reset	Description
15-8	CLIPCRL	R/W	0x00	Lower bound for the Cr
7-0	CLIPCRU	R/W	0xFF	Upper bound for the Cr

PARAMn (n=0~7) **0xB0A5B040 ~0XB0A5B05C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PARAMn															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PARAMn															

Field	Name	RW	Reset	Description
31-0	PARAMn	R/W	PARAM0: 0x12C83C28 PARAM1: 0x743C0238 PARAM2: 0xFF408081 PARAM3: 0x003801FC PARAM4: 0x12641E14 PARAM5: 0x582D011C PARAM6: 0xFF408080 PARAM7: 0x002B01FE	Filter Parameters

These parameters should be set as one of the following strength levels. Initially, they are set as the strength level 2.

STRENGTH	PARAM0	PARAM1	PARAM2	PARAM3	PARAM4	PARAM5	PARAM6	PARAM7
0	-	0x3A3C0238	0x9B2040FF	0x001C01FE	-	0x2C2D011C	0x952040FF	0x001501FF
1	-	0x573C0238	0xFF2C60AC	0x002D01FD	-	0x422D011C	0xFF2C60AB	0x002201FF
2	-	0x743C0238	0xFF408081	0x003801FC	-	0x582D011C	0xFF408080	0x002B01FE
3	-	0x7F3C0238	0xFF468C75	0x003F01FC	-	0x602D011C	0xFF468C74	0x002F01FE
4	-	0x913C0238	0xFF4CA068	0x004901FB	-	0x6E2D011C	0xFF4CA067	0x003701FE

CTRL **0xB0A5B074**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN	UU	RESERVED													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								TY	RESERVED						

Field	Name	RW	Reset	Description
-	RESERVED	R/W	0	SHOULD BE ZERO
31	EN	R/W	0	0 : enable 1 : disable
30	UU	R/W	0	Urgent Update 0: The current registered parameters are updated the end of on-going frame. 1: The current registered parameters are updated immediately. This bit needs to be 0 usally.
7	TY	R/W	1	SHOULD BE ONE

SIZE **0xB0A5B078**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED					HEIGHT										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					WIDTH										

Field	Name	RW	Reset	Description
-	RESERVED	R/W	0	SHOULD BE ZERO
26-16	HEIGHT	R/W	0	Height of the scaler output image
10-0	WIDTH	R/W	0	Width of the scaler output image

7 NTSC / PAL Encoder Composite Output

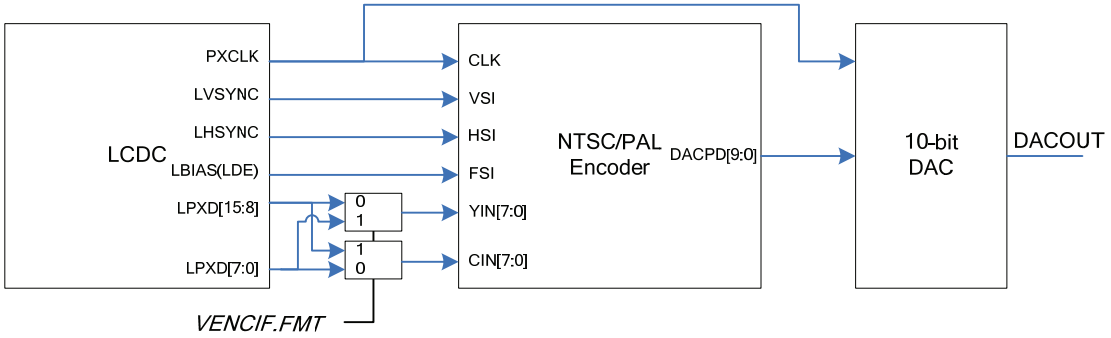
7.1 Overviews

The NTSC/PAL encoder meets all requirements of the ITU -R BT.470-3 specifications. This is designed to support all standards and variations of the NTSC and PAL encoding systems. This includes cross standards pseudo PAL and pseudo NTSC. This allows independent control of field rate, chroma subcarrier and the chroma encoding algorithm. Both the luma and chroma bandwidths can be varied to optimize for various data input conditions. Because input signals are generated by “LCD Interface”, LCD can not be displayed while NTSC/PAL composite output is enabled.

7.2 Features

- Encoding at square pixel or CCIR601 rates including flexible support for multiple clock frequencies
- Programmable Luma and Chroma bandwidth
- Programmable Saturation, Hue, Contrast and Brightness
- Supports all NTSC and PAL formats
  - NTSC-M/4.43, PAL-B/D/G/H/I/M/N/Combination N
- 8 bit or 16bit YUV input
  - ITU-R BT.601 4:2:2 16-bit Parallel Input
  - ITU-R BT.656 Parallel Input Format
- Controlled entry and exit of active video pixel and line
- Composite outputs.
- Outputs 10 bit to DAC
  - Maximum conversion rate : 27MSPS
  - Output Load Resistance : 37.5 ohm
  - Analog Output Range : 0.0 – 1.3V(Typical)

7.3 Selecting Operation Mode



To use the NTSC/PAL encoder, the LCDC should be configured as NTSC/PAL inter-face and output timing should be configured as the corresponding input timing of the NTSC.PAL encoder.
The NTSC/PAL encoder can operate at square pixel or 601 sampling rates for 50Hz or 60Hz standards. This is controlled using the PIXSEL and IFMT bits. Selecting the op-erating mode changes various internal timing locations to ensure that the final analog output signal meets appropriate standards. The operation frequencies and related in-formation are described below.

DESCRIPTION	SYMBOL	601 Sampling Clock		Square Pixel Clock	
		60Hz Field	50Hz Field	60Hz Field	50Hz Field
DAC Operation FREQ Pixel Data Rate	F_{CK27} F_{CK13}	27MHz 13.5MHz	27MHz 13.5MHz	24.5454MHz 12.2727MHz	29.5MHz 14.75MHz
Number of Clocks per Line	T_{HS}	1716	1728	1560	1888
Number of CK13(CK27) clocks for active video (Encoded pixels)	T_{AV}	720 (1440)	720 (1440)	640 (1280)	768 (1536)
Number of Lines per Field	L_{VS}	262.5	312.5	262.5	312.5
Default numbers of CK13 clocks from HS active edge	T_{FPX}	244	262	238	304

to first input data					

7.4 Input Timing

The 3 signals HSI, VSI and FSI are used to define the horizontal, vertical and field ref-erence points. The ISYNC[2:0] configuration register is used to select timing configura-tions. Once a timing mode is selected, each input has a polarity control (HSIP, VSIP and FSIP). These are used to define the “active” edge of the input sync. Finally, by programming the HOFFSET and VOFFSET bits, the desired pixel and line location for reset at the “active” edge is selected.

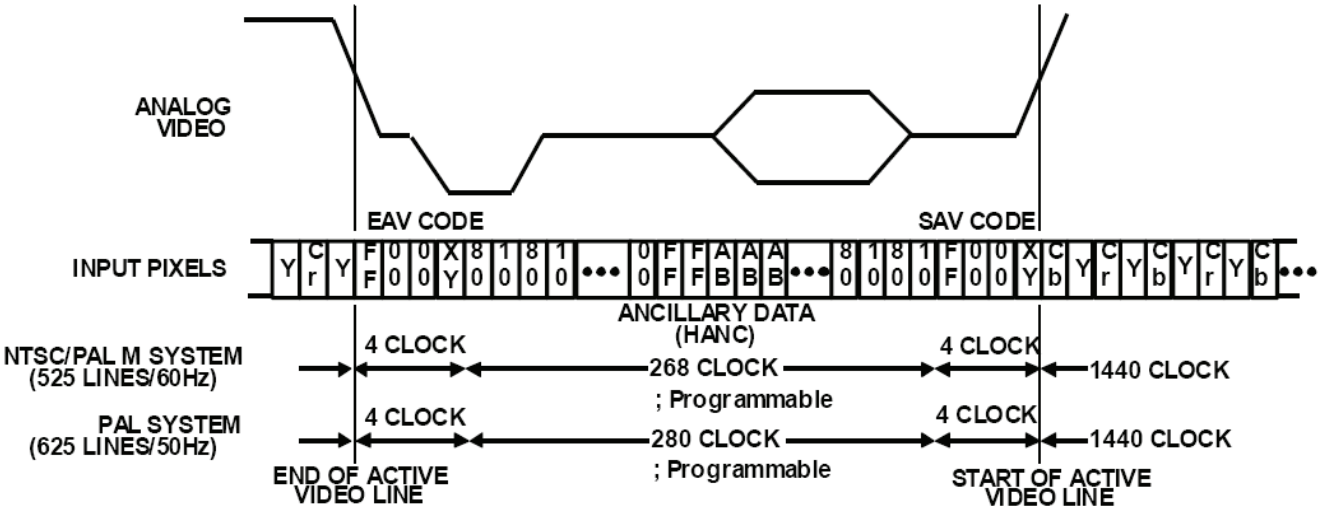


Figure 7.1 Digital Input Timing(ITU-R BT.656 8bit parallel Input)

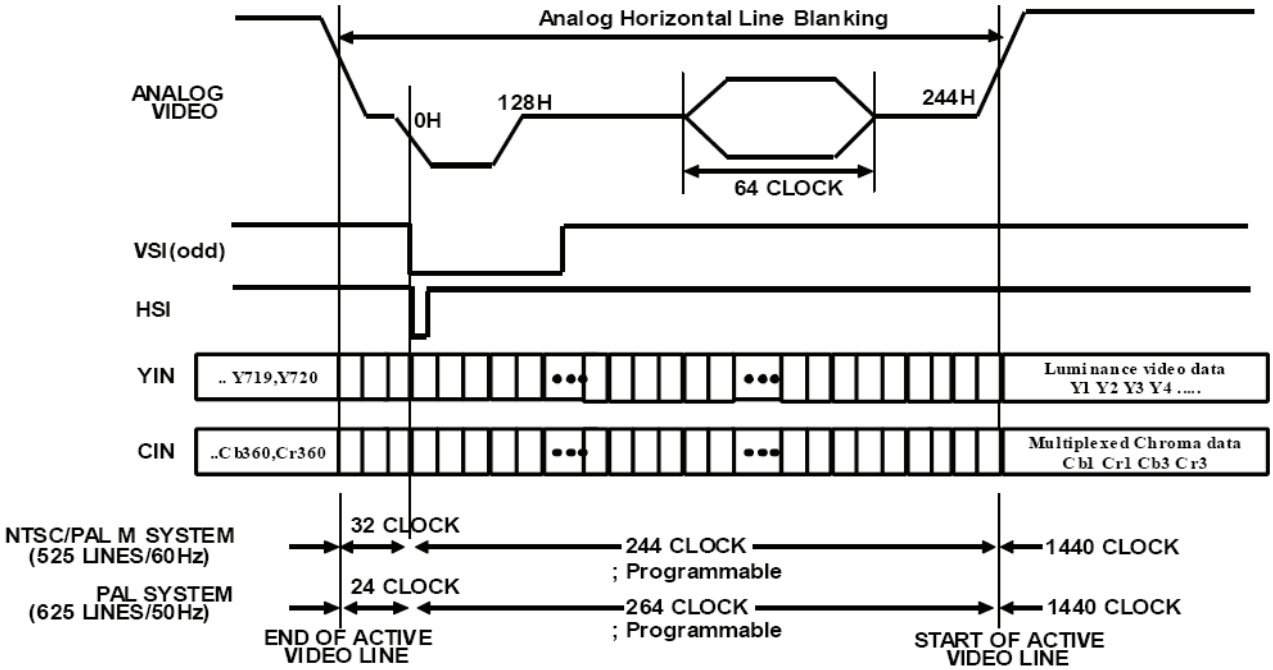


Figure 7.2 Digital Input Timing (ITU-R BT.601 4:2:2 16bit Parallel Input)

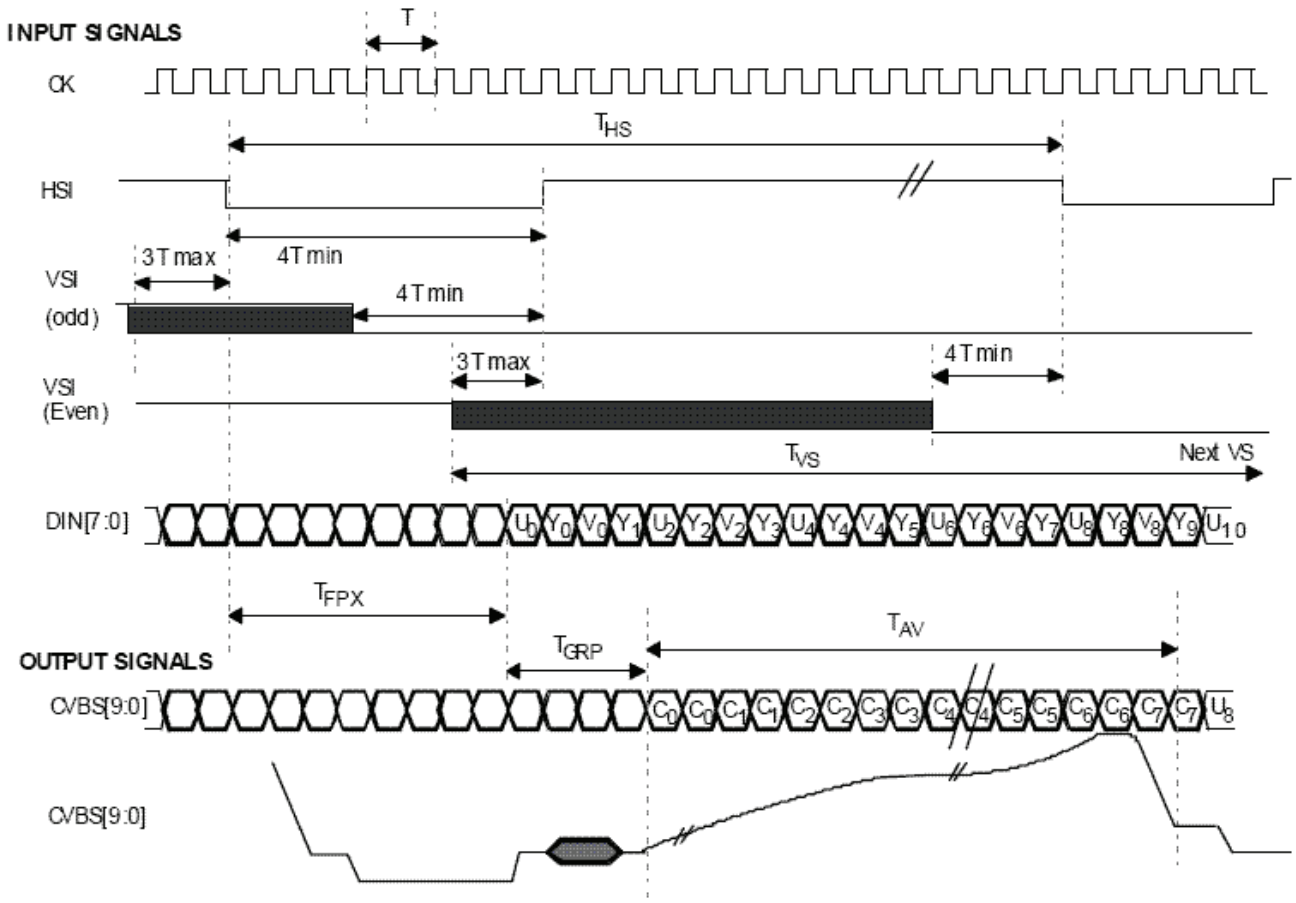


Figure 7.3 Example of Input Timing

7.5 Video Standard Selection

The NTSC/PAL encoder supports all NTSC and PAL standards throughout the world. The encoder supports independent control of the Chroma modulation frequency, selec-tion of phase alternating Line encoded chroma, and field frequency. In addition, for non SCH locked standards, the Chroma can be allowed to free run using the FDRST. If the FSCADJ register is set to any value other than '0', the SCH relationship will not be able to be maintained. For these setting it is also recommended that the FDRST bit be set to free run mode.

Table 7.1 STN LCD Palette Address

Requested output Format			Required Encoder Settings				
Format	Field Rate	FSC	IFMT	FSCSEL	PHALT	FDRST	PED
NTSC M	59.94Hz (525)	3.5795454	0	0	0	1	1/0
NTSC N	50 Hz (625)	3.5795454	1	0	0	0	1/0
PAL M	59.952 (525)	3.57561189	0	2	1	0	
PAL N	50 Hz (625)	3.58205625	1	3	1	0	
PAL B/G/H/I	50 Hz (625)	4.43361875	1	1	1	1	0
Pseudo PAL	60 Hz (525)	4.43361875	0	1	1	0	
Pseudo NTSC	50 Hz (625)	3.5795454	1	0	0	0	
NTSC 4.43	60Hz (525)	4.43361875	0	1	0	0	

7.6 Basic Video Adjustments

The standard video adjustments for Chroma and Luma are included. Chroma controls include Saturation and Hue (SAT HUE). Luma adjustments include Contrast and Brightness (CONTBRIGHT). An additional SCH control is included that changes the Chroma subcarrier phase relative to the horizontal sync. Note that this feature operates only when FDRST = 0.

7.7 Programmable Bandwidth

The data bandwidth for the Luma and chroma paths is shown in the following fre-quency plots. The YBW control allows 3 different settings to optimize the output band-width to the receiver. The same applies to chroma bandwidth using the CBW control.

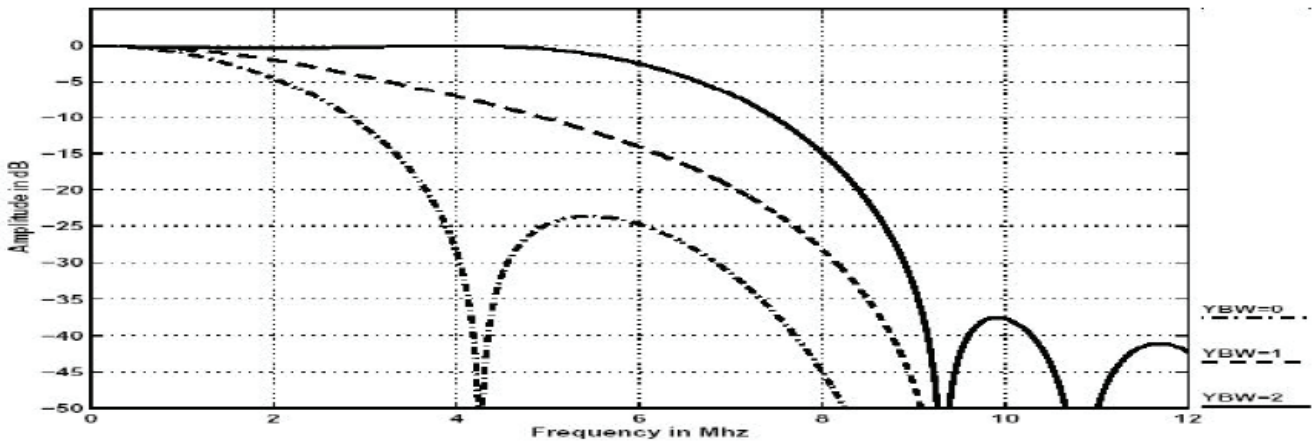


Figure 7.4 Luma Bandwidth

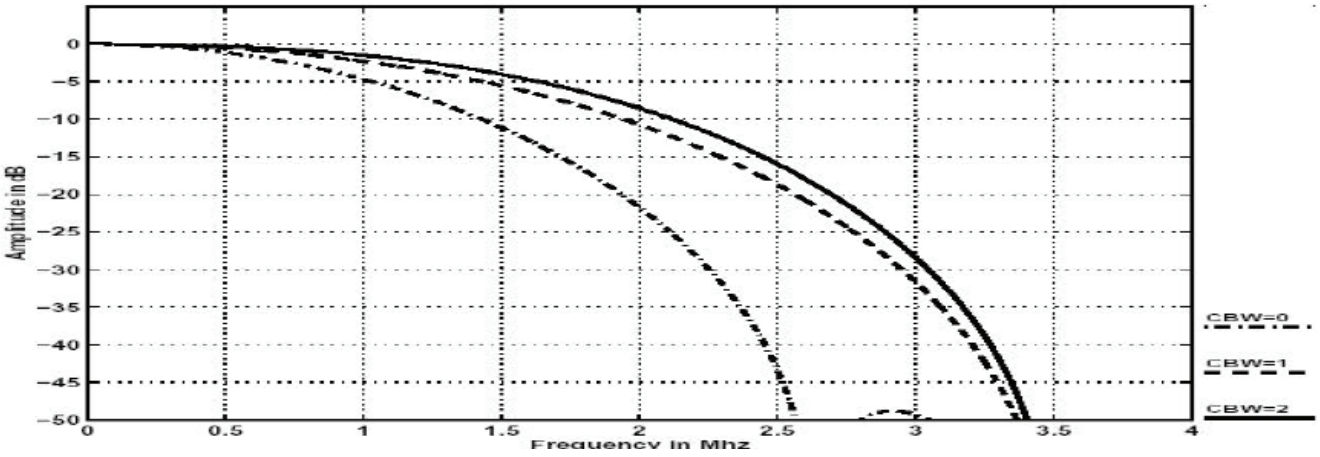


Figure 7.5 Chroma Bandwidth

7.8 Analog Video Output Configuration

The NVS2310 supports composite video. Refer to output configuration register DACSEL for further details. The DAC output levels and the associated digital codes are summarized below. DAC voltage assumes the standard 140IRE = 1v. Numbers are shown for NTSC type video with a pedestal.

Table 7.2 Summary of DAC voltage and Codes

Signal Level	CVBS / LUMA Value	IRE Value	DAC Voltage
Max output	1023	137.2	1.282v
100% White	810	100	1.015v
Black	282	7.37	353mv
Sync	12	-40	15mv
White - Blank	570	100	714mv delta
White - sync	798	140	1v delta
Color burst	228	40	285mv delta

7.9 Registers

STATA **0xB0A40000**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								REVID					FIELD		

Field	Name	RW	Reset	Description
7-5-	REVID	R	0	Current Revision Identification Number
2-0	FIELD	R	-	The current video field 0-3 or 4-7 for NTSC fields 1-4 0-7 for PAL fields 1-8

ECMDA **0xB0A40004**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								PWDENC	FDRST	FSCCELL		PED	PIXEL	IFMT	PHALT

Field	Name	RW	Reset	Description
7	PWDENC	R/W	0	Power Down Mode 0: Normal Operation 1 Power down mode for the entire digital logic circuits of encoder digital core.
6	FDRST	R/W	0	SCH Chroma-Luma locking control 0: Constant relation ship between color burst and horizontal sync maintained for appropriate video standards 1: Chroma is free running as compared to horizontal sync
5-4	FSCSEL	R/W	0	Modulation frequency of chroma output 0 Color subcarrier frequency = 3.57954545 MHz for NTSC 1 Color subcarrier frequency = 4.43361875 MHz for PAL - B,D,G,H,I,N 2 Color subcarrier frequency = 3.57561149 MHz for PAL - M 3 Color subcarrier frequency = 3.58205625 MHz for PAL –combination N
3	PED	R/W	0	Define input pedestal format 0 Video output has no pedestal*. 1 Video output has a pedestal.
2	PIXEL	R/W	0	Select Pixel sampling rate. 0 Input data is at 601 rates. * 1 Input data is at square pixel rates.
1	IFMT	R/W	0	Format of Output Data 0 525 Lines 1 625 Lines
0	PHALT	R/W	0	Phase Alternate control for PAL encoded chroma signal output 0 NTSC encoded color 1 PAL encoded color

ECMDB **0xB0A40008**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
											YBIBLK	CBW		YBW	

Field	Name	RW	Reset	Description
4	VBIBLK	R/W	0	VBI Blanking control 0 Input data is passed through for non vbi processing lines 1 Video data is forced to black level for vertical non vbi processed lines.
3-2	CBW	R/W	0	Chroma Bandwidth control 0 Low bandwidth 1 Medium bandwidth 2 High bandwidth 3 Not used, default to low bandwidth
1-0	YBW	R/W	0	Luma Bandwidth control 0 Low bandwidth 1 Medium bandwidth 2 High bandwidth 3 Not used, default to low bandwidth

GLK **0xB0A4000C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
											XT24	GLKEN	GLKE		GLKPL

Field	Name	RW	Reset	Description
4	XT24	R/W	0	24MHz Crystal input when high 0 27MHz Crystal input 1 24MHz Crystal input
3	GLKEN	R/W	0	Genlock reset control 0 Genlock reset disable 1 Genlock reset enable
2-1	GLKE	R/W	0	Chroma Fsc Generation frequency control 0,1 Chroma Fsc is generated from internal constants based on current user setting. 2 Fsc is adjusted based on external RTCO input 3 Fsc tracks non standard Encoder clock (CLKI) programmed frequencies.
0	GLKPL	R/W	0	Genlock mode PAL ID Polarity control 0 PAL bit input polarity is active high 1 PAL bit input polarity is active low

SCH **0xB0A40010**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								SCH							

Field	Name	RW	Reset	Description
7-0	SCH	R/W	0	Programs the Color burst phase relation to the sync tip. '0' is the nominal value. The 8 bit control covers the entire 360 range as a 2's compliment number.

HUE **0xB0A40014**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								HUE							

Field	Name	RW	Reset	Description
7-0	HUE	R/W	0	Programs the Active video Color burst phase relative to color burst. The 8 bit control covers the entire 360 range as a 2's compliment number.

SAT **0xB0A40018**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								SAT							

Field	Name	RW	Reset	Description
7-0	SAT	R/W	0	Controls the Active video Chroma gain relative to the color burst gain. Value is 2's compliment with '0' the nominal value.

CONT **0xB0A4001C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								CONT							

Field	Name	RW	Reset	Description
7-0	CONT	R/W	0	Controls Luma gain. Value is 2's compliment with '0' the nominal value.

BRIGHT **0xB0A40020**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								SCH							

Field	Name	RW	Reset	Description
7-0	BRIGHT	R/W	0	Controls Luma offset. Value is 2's compliment with '0' the nominal value.

FSC_ADJM **0xB0A40024**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								FSC_ADJM							

Field	Name	RW	Reset	Description
7-0	FSC_ADJM	R/W	0	FSC_ADJ[15:8]

FSC_ADJL **0xB0A40028**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								FSC_ADJ[7:0]							

Field	Name	RW	Reset	Description
7-0	FSC_ADJL	R/W	0	FSC_ADJ[7:0]

FSC_ADJ[7:0] allows the Pixel clock to be varied up to +/- 200ppm of its nominal value. This allows dot crawl adjustment.

This 16 bit signal is multiplied by 4 and added to the internal chroma frequency constant.

ECMDC **0xB0A4002C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								CSMDE	CSMD		RGBSYNC				

Field	Name	RW	Reset	Description
7	CSMDE	R/W	0	Composite sync CSYN Pin tri-state control 0 Composite sync mode not enable - Pin is tri-stated 1 Composite sync mode enable - Output reflexes mode defined by CSMD
6-5	CSMD	R/W	0	Composite sync CSYN Pin output control 0 Composite Sync 1 Keyed clamp 2 Keyed pulse 3 N/A
4-3	RGBSYNC	R/W	0	Enable RGBSYNC when output is configured for analog EGB modes 0 No sync on the RGB output signals 01 Sync on the RGB output signals 10 Sync on the G output signal

DACSEL **0xB0A40040**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												DACSEL			

Field	Name	RW	Reset	Description
7-4	-	R/W	0	SHOULD BE ZERO
3-0	DACSEL	R/W	0	Data type output selection for DAC. 0: DAC digital output is disabled, Output is code 0. 1: Data output in CVBS format – Composite Video others : N/A

DACPD **0xB0A40050**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															PD

Field	Name	RW	Reset	Description
7-1	-	R/W	0	SHOULD BE ZERO
0	PD	R/W	0	0 DAC Normal Operation 1 DAC Power down

ICNTL **0xB0A40080**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								FSIP	VSIP	HSIP	HSVSP	VSMD	ISYNC		

Field	Name	RW	Reset	Description
7	FSIP	R/W	0	Field polarity. It controls the polarity of any F or Field control as input or output. 0 Odd field (fields 1,3,5,7) active low. 1 Odd field active high.
6	VSIP	R/W	0	Vertical field polarity 0 Vertical sync active low. 1 Vertical sync active high.
5	HSIP	R/W	0	Horizontal sync polarity 0 Horizontal sync active low.

				1 Horizontal sync active high.
4	HSVSP	R/W	0	Horizontal and vertical sync latch enable 0 Enable to latch on the falling edge of VS and HS . 1 Enable to latch on the rising edge of VS and HS.
3	VSMD	R/W	0	Vertical Sync Output format 0 Odd Field VS and FS outputs aligned to video line Start. Even Field VS and FS output aligned to video line midpoint. 1 Odd Field VS and FS outputs aligned to video line Start. Even Field VS and FS outputs aligned to video line Start.
2-0	ISYNC	R/W	0	Timing configuration for Horizontal, Vertical and Field inputs. 0 Alignment input format from pin FSI. F --- Field timing from the FSI Pin directly. V --- Vertical timing from the FSI Pin rising or falling edge. H --- Horizontal timing from the FSI Pin rising or falling edge. 1 Alignment input format from pins HSI, VSI, FSI. F --- Field timing from the FSI pin. V --- Vertical timing from the VSI pin rising or falling edge. H --- Horizontal timing from the HSI pin rising or falling edge. 2 Alignment input format from pins HSI and VSI. F --- Field timing from the latched value of HSI and VSI rising or falling edge. V --- Vertical timing from the VSI pin rising or falling edge. H --- Horizontal timing from the HSI pin rising or falling edge. 3 Alignment input format from pins VSI and FSI. F --- Field timing from the FSI pin directly. V --- Vertical timing from the VSI pin rising or falling edge. H --- Horizontal timing from the VSI pin rising or falling edge. 4 Alignment input format from pin VSI Only. Note that there is no field information in this mode so the software counter reset is required for proper field identification. F --- Field timing from software reset only. V --- Vertical timing from the VSI pin rising or falling edge. H --- Horizontal timing from the VSI pin rising or falling edge. 5 Alignment is from embedded EAV,SAV codes (line by line). F --- Field timing from the SAV/EAV ‘F’ bit. V --- Vertical timing from the SAV/EAV ‘V’ bit. H --- Horizontal timing from the SAV/EAV ‘H’ bit. 6 Alignment is from embedded EAV,SAV codes (frame by frame). F --- Field timing from the SAV/EAV ‘F’ bit. V --- Vertical timing from the SAV/EAV ‘F’ bit. --- Horizontal timing from the SAV/EAV ‘F’ bit. 7 F,V,H Alignment is Free Running (Master Mode).

HVOFFST **0xB0A40084**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								INSEL				VOFFST	HOFFSET[10:8]		

Field	Name	RW	Reset	Description
7-6	INSEL	R/W	0x0	Digital input format select. 0: format is 16 bit YUV 4:2:2 data Sampled at 27MHz. 1: format is 16 bit YUV 4:2:2 data Sampled at 13.5MHz . 2: Input format is 8 bit YUV 4:2:2 data 27MHz data Samples at 13.5MHz. 3: Input is from TESTI port, same format as mode 2.
3	VOFFST	R/W	0x0	Vertical offset bit 8. Refer to VOFFST register. VOFFSET[8:0] determines Vertical alignment point for the VSI input. Programmed value is the “line number” that pixel counter is reset to when a VSI transition in input.
2-0	HOFFSET	R/W	0x0	Horizontal offset bit 8 ~ 10. Refer to HOFFST register. HOFFSET[10:0] determines Horizontal alignment point for the HSI input. Programmed value is the “pixel number” that pixel counter is reset to when a HIS transition in input.

HOFFST **0xB0A40088**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								HOFFSET[7:0]							

Field	Name	RW	Reset	Description
7-0	HOFFSET	R/W	0x0	-

Refer to the HVOFFST register description.

VOFFST **0xB0A4008C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								VOFFSET[7:0]							

Field	Name	RW	Reset	Description
7-0	VOFFSET	R/W	0x0	-

Refer to the HVOFFST register description.

HSVSO **0xB0A40090**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									VSOB	HSOB[10:8]			HSOE[10:8]		

Field	Name	RW	Reset	Description
6	VSOB	R/W	0	VSOB[8]. Programs the Leading edge of the Vertical sync Output. This signal is not used inside the core. It is provided a reference for master mode timing.
5-3	HSOB	R/W	0	HSOB[10:8] Programs the Leading edge of the Horizontal sync output. The HSO signal is not used.
2-0	HSOE	R/W	0	HSOE[10:8] Programs the Trailing edge of the Horizontal sync output. The HSO signal is not used inside the core. It is provided as a source for master timing.

HSOE **0xB0A40094**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								HSOE[7:0]							

Field	Name	RW	Reset	Description
7-0	HSOE	R/W	0x0	-

Refer to HSVSO register.

HSOB **0xB0A40098**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								HSOB [7:0]							

Field	Name	RW	Reset	Description
7-0	HSOB	R/W	0x0	-

Refer to HSVSO register.

VSOB **0xB0A4009C**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								VSOB [7:0]							

Field	Name	RW	Reset	Description
7-0	VSOB	R/W	0x0	-

VSOE **0xB0A400A0**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								VSOST		NOVRST	VSOE				

Field	Name	RW	Reset	Description
7-6	VSOST	R/W	0x0	Programs the Vertical sync relative location for Odd and Even Fields.
5	NOVRST	R/W	0x0	No Vertical Reset, working in conjunction with EPRGOUT. 0: Normal operation (interlaced output timing). 1: No vertical reset on every field.
2-0	HSOE	R/W	0x0	Programs the Trailing edge of the Vertical sync Output. This signal is not used inside the core. It is provided a reference for master mode timing.

VENCON **0xB0A40800**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															EN

Field	Name	RW	Reset	Description
31-1				RESERVED
0	EN	R/W	0x0	It determines whether output signals of the LCDC is connected to the NTSC/PAL encoder. 0: disable 1: Enable (connection)

VENCIF **0xB0A40804**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														MV	FMT

Field	Name	RW	Reset	Description
31-2	-	-	-	RESERVED
1	MV	R/W	0x0	RESERVED
0	FMT	R/W	0x0	It determines data bus connection type between LCDC and NTSC/PAL encoder. 0: LCDC PXDATA[7:0] is connected to NTSC/PAL CIN[7:0] LCDC PXDATA[15:0] is connected to NTSC/PAL YIN[7:0] 1: LCDC PXDATA[7:0] is connected to NTSC/PAL YIN[7:0] LCDC PXDATA[15:8] is connected to NTSC/PAL CIN[7:0]

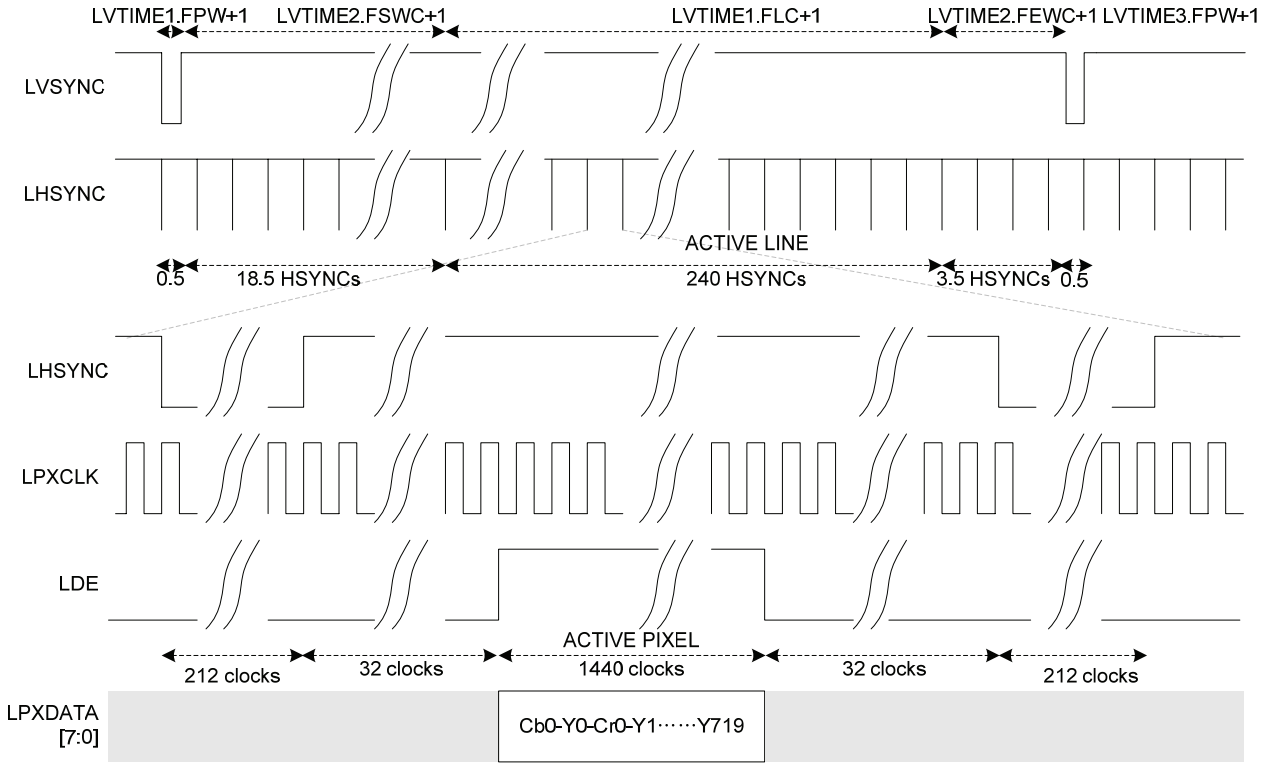
7.10 Example for NTSC/PAL Interface

We will strongly recommend you to use the values in the tables.

NTSC-M interface

The following tables show how to configure the LCDC and the NTSC/PAL encoder for NTSC. PXCLK of the LCDC should be configured as 27MHz.

LCDC register configuration for NTSC-M	
LDS.HSIZE = 720	LCTRL.TV = 1
LDS.VSIZE = 480	LCTRL.NI = 0
LHTIME1.LPC = 720 * 2 -1	LCTRL.DP = 0
LHTIME1.LPW = 212 - 1	LCTRL.PXDW = 6
LHTIME2.LSWC = 32 - 1	LCTRL.IH = 1
LHTIME2.LEWC = 32 - 1	LCTRL.IV =1
LVTIME1.FLC = 480 – 1	LCTRL IP = 1
LVTIME1.FPW = 1 - 1	
LVTIME2.FSWC = 37 -1	
LVTIME2.FEWC = 7 -1	
LVTIME3.FLC = 480 – 1	
LVTIME3.FPW = 1 – 1	
LVTIME4.FSWC = 38 – 1	
LVTIME4.FEWC = 6 - 1	



NTSC/PAL encoder register configuration for NTSC-M	
DACPD. PD = 0 (not power down)	HOFFST.HOFFSET = 0
ECMDA.PWDENC = 0	VOFFST.VOFFSET =1
ECMDA.FDRST = 1	
ECMDA.PHALT = 0	HVOFFST.HOFFSET = 0
ECMDA.IFMT = 0	HVOFFST.VOFFST = 0
ECMDA.PIXSEL = 0	HVOFFSET.INSEL = 2
ECMDA.PED = 1	
ECMDA.FSCSEL = 0	
ECMDB.VBIBLK = 0	
DACSEL.DACSEL = 1 (composite output)	
ICNTL.ISYNC = 2	
ICNTL.HSVSP = 1	
ICNTL.HSIP = 0	
ICNTL.VSIP = 1	
ICNTL.FSIP = 0	
VENCIF.FMT = 1	
VENCON.EN = 1	

PAL-B/G/H/I interface

The following tables show how to configure the LCDC and the NTSC/PAL encoder for PAL-B/G/H/I. PXCLK of the LCDC should be configured as 27MHz.

LCDC register configuration for PAL B/G/H/I	
LDS.HSIZE = 720	LCTRL.TV = 1
LDS.VSIZE = 576	LCTRL.NI = 0
LHTIME1.LPC = 720 * 2 -1	LCTRL.DP = 0
LHTIME1.LPW = 128 - 1	LCTRL.PXDW = 6
LHTIME2.LSWC = 138 - 1	LCTRL.IH = 1
LHTIME2.LEWC = 22 - 1	LCTRL.IV =1
LVTIME1.FLC = 576 – 1	LCTRL IP = 1
LVTIME1.FPW = 1 - 1	
LVTIME2.FSWC = 43 -1	
LVTIME2.FEWC = 5 -1	
LVTIME3.FLC = 576 – 1	
LVTIME3.FPW = 1 – 1	
LVTIME4.FSWC = 44 – 1	
LVTIME4.FEWC = 4 - 1	

NTSC/PAL encoder register configuration for PAL B/G/H/I	
DACPD. PD = 0 (not power down)	HOFFST.HOFFSET = 0
ECMDA.PWDENC = 0	VOFFST.VOFFSET =1
ECMDA.FDRST = 1	
ECMDA.PHALT = 1	HVOFFST.HOFFSET = 0
ECMDA.IFMT = 1	HVOFFST.VOFFST = 0
ECMDA.PIXSEL = 0	HVOFFSET.INSEL = 2
ECMDA.PED = 0	
ECMDA.FSCSEL = 1	
ECMDB.VBIBLK = 0	
DACSEL.DACSEL = 1 (composite output)	
ICNTL.ISYNC = 2	
ICNTL.HSVSP = 1	
ICNTL.HSIP = 0	
ICNTL.VSIP = 1	
ICNTL.FSIP = 0	
VENCIF.FMT = 1	
VENCON.EN = 1	

7.11 Copy Generation Management Systems

Copy Generation Management Systems (CGMS) is supported for 525/60 for Japan. Line 20 and 283 are used to transmit the CGMS data when the CGMS function is en-abled. The CGMS consists of a 2 bit start code and a 20 bit data. The 20 bits data, in-cluding 6 bits of Cyclic Redundancy Check, are loaded into the CGMS register via the host interface prior to being transmitted. The data are transmitted at a rate of 447.443 kHz per bit with the amplitude ranging from 0 IRE to 70 IRE.

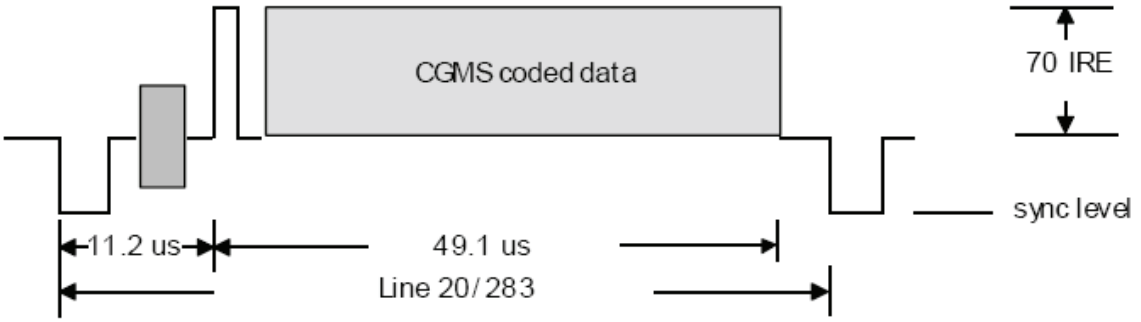


Figure 7.6 Copy Generation Management Systems

VSTAT **0xB0A400C0**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														CGRDY	

Field	Name	RW	Reset	Description
Others	-	-	-	RESERVED
1	CGRDY	R	0	Copy Generation Management System Status (READ ONLY) 0: CGMS is able to set. 1: CGMS is set already.

VCTRL **0xB0A400C4**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								0	0	0	0	0	CGOE	CGEE	0

Field	Name	RW	Reset	Description
Others	-	-	-	Should Be Zero
2	CGOE	R/W	0	Copy Generation Management System enable odd field. 0: CGMS is not enabled. 1: CGMS is enabled.
1	CGEE	R/W	0	Copy Generation Management System enable even field. 0: CGMS is not enabled. 1: CGMS is enabled.

CGMSA **0xB0A400E0**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										CGMS6	CGMS5	CGMS4	CGMS3	CGMS2	CGMS1

Field	Name	RW	Reset	Description
5	CGMS6	R/W	0x0	6th bit of CGMS data
4	CGMS5	R/W	0x0	5th bit of CGMS data
3	CGMS4	R/W	0x0	4th bit of CGMS data
2	CGMS3	R/W	0x0	3rd bit of CGMS data
1	CGMS2	R/W	0x0	2nd bit of CGMS data
0	CGMS1	R/W	0x0	1st bit of CGMS data

CGMSB **0xB0A400E4**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								CGMS14	CGMS13	CGMS12	CGMS11	CGMS10	CGMS9	CGMS8	CMGS7

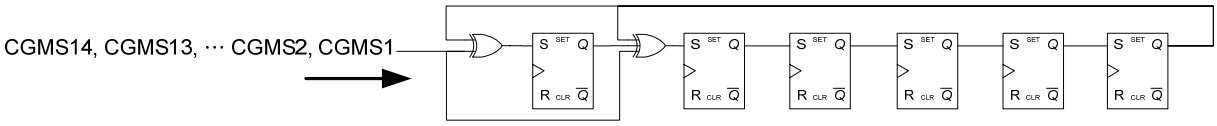
Field	Name	RW	Reset	Description
7	CGMS14	R/W	0x0	14th bit of CGMS data
6	CGMS13	R/W	0x0	13th bit of CGMS data
5	CGMS12	R/W	0x0	12th bit of CGMS data
4	CGMS11	R/W	0x0	11th bit of CGMS data
3	CGMS10	R/W	0x0	10nd bit of CGMS data
2	CGMS9	R/W	0x0	9st bit of CGMS data
1	CGMS8	R/W	0x0	8th bit of CGMS data
0	CGMS7	R/W	0x0	7th bit of CGMS data CGMS8 / CGMS7 / DESCRIPTION 0 / 0 / Copying permitted without restriction. 0 / 1 / One copy permitted. 1 / 0 / No more copies.(One copy has already been made.) 1 / 1 / No copying permitted.

CGMSC **0xB0A400E8**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										CGMS20	CGMS19	CGMS18	CGMS17	CGMS16	CGMS15

Field	Name	RW	Reset	Description
5	CGMS20	R/W	0x0	20th bit of CGMS data
4	CGMS19	R/W	0x0	19th bit of CGMS data
3	CGMS18	R/W	0x0	18th bit of CGMS data
2	CGMS17	R/W	0x0	17th bit of CGMS data
1	CGMS16	R/W	0x0	16th bit of CGMS data
0	CGMS15	R/W	0x0	15th bit of CGMS data

CGMSA, CGMSB, and CGMSC registers are 20bits CGMS data. Especially, CGMSC register is for the CRC for CGMSA and CGMSB and is manually set according to CGMSA and CGMSB value. Figure 7.7 shows example of the CRC calculator for CGMS. For example, when CGMSA=0x00 and CGMSB=0x03, CGMSC (CRC) should be set to 0x3A.



The CRC used is $X^6 + X+1$, all preset to "1".

Figure 7.7 Example: CRC calculator for CGMS

7.12 10-Bit DAC

10-bit DAC is used for composite video signal output.

7.12.1 HDMI Specific Features

- Resolution : 10-bit
- Maximum conversion rate : 27MSPS
- BGR (Internal / External)
- Power down mode
- Output load resistance : 37.5Ω
- Analog output range : 0.0 ~ 1.3V(Typical)
- Power supply : 3.0V single

7.12.2 Functional Description

The DAC is initially power on state. To make the DAC enter the power down mode, the PD bit of DACPD register in NTSC/PAL encoder should be set to 1

The DAC uses reference current to decide the 1LSB current size by dividing the reference current by 31 times. To adjust the full current output, you must decide the "R(IREF)" resister value(connected to IREF pin) and "VREF" voltage value(connected to VREF pin). Its voltage output can be obtained by connecting RLOAD (connected to DACOUT pin).

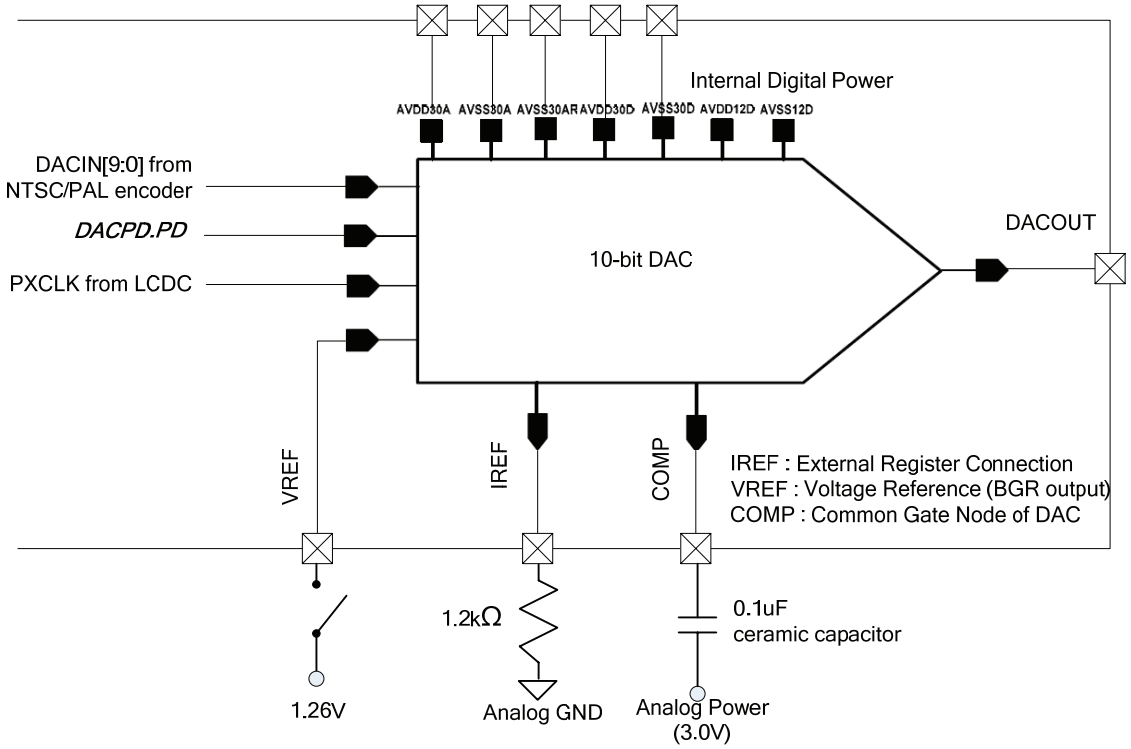


Figure 7.8 10-bit DAC Connection

The voltage output of DAC are decided by R(IREF), RLOAD, and V(VREF).

$$VO = \frac{V(VREF)}{R(IREF) \times 31} \sum_{i=0}^{9} (2^i \times DACIN[i]) * R_{LOAD}$$

For example, when the corresponding powers are applied to DAC power pins, VREF might be about 1.26V. If 1.2kΩ is connected to IREF pin and DACIN bits are all 1s and RLOAD is 37.5Ω, DACOUT is about 1.3V. in case R(IREF) is 1.5kΩ, DACOUT is about 1.04V. If you cannot change resister value connected to IREF pin, you can apply the corresponding voltage to VREF pin to adjust VO voltage.

8 HDMI

8.1 Overview

NVS2310 is comprised of an HDMI Tx controller with I2S/SPDIF input interface, CEC block and AES block for HDCP Key encryption.

8.1.1 HDMI Specific Features

- HDMI 1.3, HDCP 1.1, DVI 1.0 Complaint
- Supports Video format:
  480p @59.94Hz/60Hz, 576p@50Hz
  720p @50Hz/59.94Hz/60Hz
  1080i @50Hz/59.94Hz/60Hz
  1080p @50Hz/59.94Hz/60Hz
- Other various formats up to 148.5 MHz Pixel Clock
- Supports Color Format : RGB/YCbCr 4:2:2
- Pixel Repetition (Up to x4)
- Supports Bit Per Color : 8bit, 10bit ,12bit
- Dedicated block for CEC function
- Supports Audio Sample packets, DSD packets, HBR packets and DST packets for audio
- Integrated HDCP Encryption Engine for Video/Audio content protection
- Not include DDC (recommend to use separate I2C)

8.2 Architecture

NVS2310 HDMI controller has following interface.
- APB and I2C for register configuration
- Dedicated Video input interface
- Dedicated SPDIF input interface
- Dedicated Audio input interface for I2S

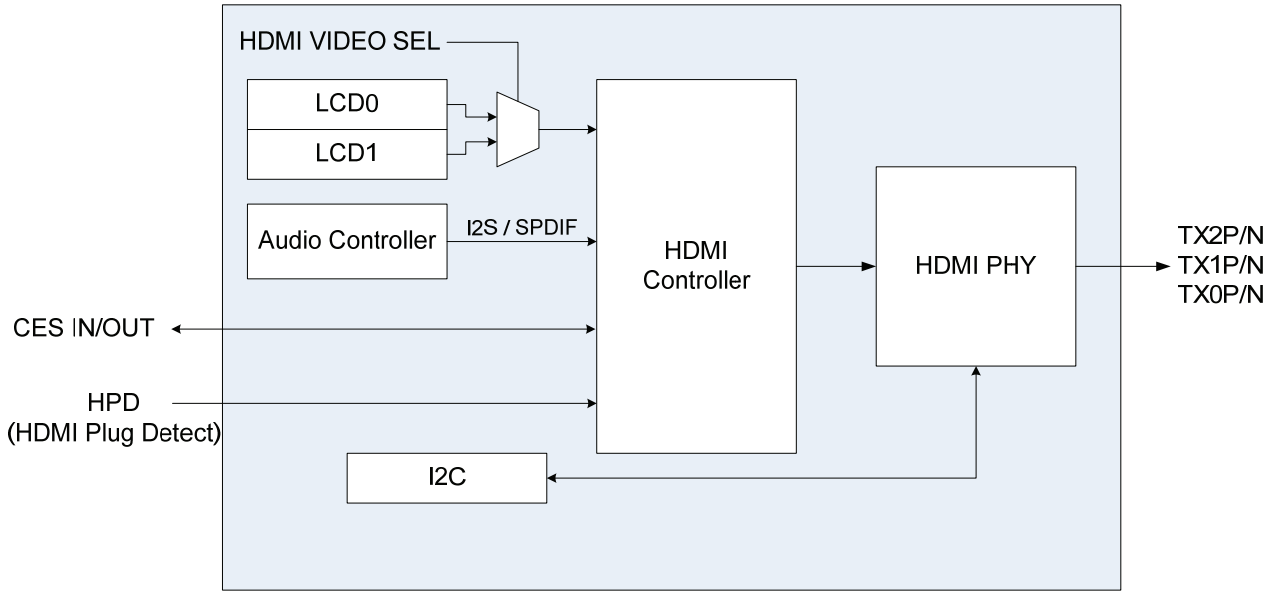


Figure 8.1 HDMI LINK and PHY

8.3 HDMI Controller (LINK)

8.3.1 HDCP KEY Management

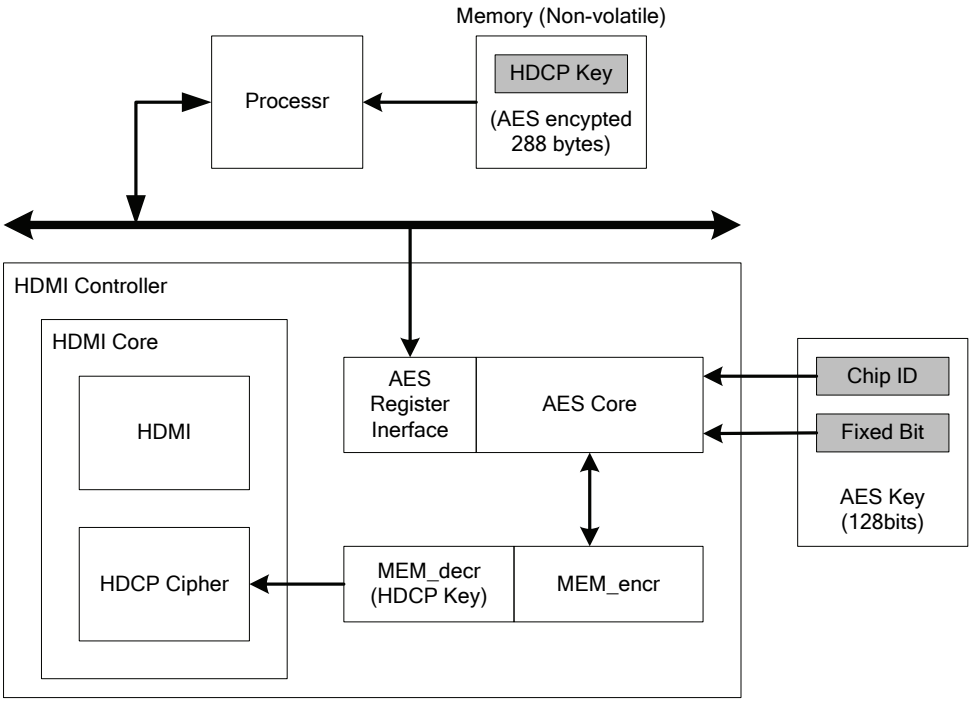


Figure 8.2 Block Diagram of HDCP Key Management

HDMI controller can decrypt AES-encrypted HDCP key. For HDMI with HDCP function, each device has unique HDCP key. In HDMI controller, HDCP key is stored outside of the HDMI controller at non-volatile memory. HDMI controller adopts AES encryption method to protect original HDCP key. Non-volatile memory does not have original raw-HDCP key, and without proper decryption the key in the non-volatile memory is useless.

In this situation, HDMI controller must have an additional AES key. This is needed to decrypt the AES encrypted HDCP key data. AES key can consist of variable chip id and fixed hardware value. However, this AES key combination can be different depending on the user configuration. Hacker may steal AES key and get an HDCP key. However, other device's HDCP key cannot be revealed because the Chip ID is different for each Chip. By this configuration HDCP key can be stored outside safely.

Core reads the AES-encrypted HDCP key once before starting HDMI and writes into mem_encr with the amount of decryption size desired. Then a command starts to decrypt encrypted data and the decrypted data are written into mem_decr. At that time the START flag goes down to notify the decryption is done.

AES core has 128-bit key that is embedded in hardwired fashion. It only supports 128-bit wide decryption. The first thing to do decryption is to load data into mem_encr. To load encryption data into encr_mem first sets AES_DATA_SIZE_H/L to decide how many bytes to decrypt and it should be 128-bit aligned because AES core is operating with 128bit data. And set AES_START to 1. If data is decrypted and full the mem_decr with decrypted data then the AES_START flag goes down to 0 to indicate the decryption and transfer are done. After checking that AES_START flag is 0, decrypted data can be used for HDCP key. HDCP cannot operate until the AES_START flag goes to 0.

8.3.2 Interface Protocol

8.3.2.1 Video Input Interface

Video input of HDMI controller is connected with NVS2310 LCD output. To use HDMI video output correctly, HDMI link setting, HDMI PHY setting and LCD setting should be consistent. LCD, HDMI link and HDMI PHY register setting values for some well-known video resolutions (for HDTV) are shown in chapter 8.6 .

8.3.2.2 Audio Input Interface

HDMI controller has two audio input ports (SPDIF Rx and I2S Rx). Each of audio inputs is connected with NVS2310 SPDIF Tx and NVS2310 I2S Tx.

Users can use SPDIF interface for two-channel Linear or Non-linear PCM Audio transmission and I2S interface for up-to eight channel Linear PCM or up-to six channel Super Audio CD audio transmission.

For I2S interface, users can specify the channel status block and the user bit information in registers, which does not inherently exist in I2S audio format.

8.3.2.3 HPD

HPD signal has two transactions: rising (plugged) and falling (unplugged) transition. Users can specify the stage number of the noise filter to reduce the possible glitches during transition.

8.3.2.4 CEC Interface

CEC is abbreviation of Consumer Electronics Control and is used for controlling devices connected to a one-wired “CEC bus”. The CEC output port is a bidirectional port, which should be pulled-up to 3.3V using external resistor and voltage supply. (i.e. an open-drain connection)

CEC devices send messages serially (MSB first) via CEC bus and the receiving device sends acknowledge bit by pulling the bus to low at ACK bit timing. For timing of each bit and structure of a message, refer to HDMI specification 1.3b, Supplement 1.(CEC specification)

8.3.2.5 AESKEY

HDMI controller has AES decryption function to support HDCP Key encryption in external HDCP Key configuration. Users can use the AESKEY to decrypt AES-encrypted HDCP Key. AESKEY value must be stable when AES decryption starts.

8.3.2.6 Interrupt Timing

HDMI controller generates level-triggered interrupts. Interrupts are masked in two levels: each submodule and controller. On the other hand, interrupt clear happens only in sub module for all interrupts except HPD. For HPD, every plug and unplug will generate, HPD plug interrupt and HPD unplug interrupt, respectively. When interrupt is masked on, interrupt asserts reflecting the current HPD plug status as shown in Figure 8.3 and Figure 8.4 .

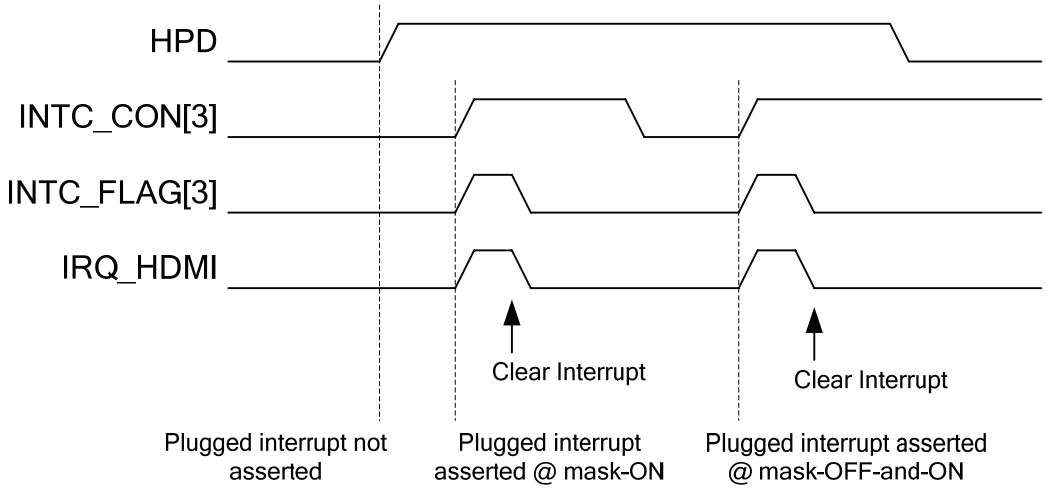


Figure 8.3 Timing Diagram for HPD Plug Interrupt

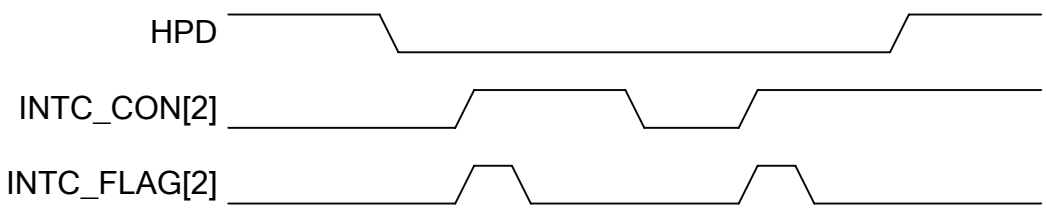


Figure 8.4 Timing Diagram for HPD Unplug Interrupt

8.4 HDMI Controller Register Description

Register Base	Offset	Description
CTRL_BASE	0xB0A54000	Controller register base address
HDMI_CORE_BASE	0xB0A55000	HDMI register base address
AES_BASE	0xB0A56000	AES register base address
SPDIF_BASE	0xB0A57000	SPDIF Receiver register base address
I2S_BASE	0xB0A58000	I2S Receiver register base address
CEC_BASE	0xB0A59000	CEC register base address

8.4.1 Control Registers

Name	Offset	Type	Description	Default
INTC_CON	0x0000	R/W	Interrupt Control Register	0x00
INTC_FLAG	0x0004	R/W	Interrupt Flag Register	0x00
AESKEY_VALID	0x0008	R	aeskey_valid Register	0x00
HPD	0x000C	R	HPD signal	0x00

INTC_CON (Interrupt Control Register) **0xB0A54000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								Pol	Global	I2S	CEC	HPDin	HPDout	SPDIF	HDCP

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	Pol	R/W	0	Interrupt Polarity 0 : Active high 1 : Active low
6	Global	R/W	0	0 : All interrupts are disabled 1 : Interrupts are enabled or disabled by INTC_CON5:0]
5	I2S	R/W	0	I2S interrupt enable 0 : Disabled 1 : Enabled
4	CEC	R/W	0	CEC interrupt enable 0 : Disabled 1 : Enabled
3	HPDin	R/W	0	HPD plugged interrupt enable 0 : Disabled 1 : Enabled
2	HPDout	R/W	0	HPD unplugged interrupt enable 0 : Disabled 1 : Enabled
1	SPDIF	R/W	0	SPDIF interrupt enable 0 : Disabled 1 : Enabled
0	HDCP	R/W	0	HDCP interrupt enable 0 : Disabled 1 : Enabled

INTC_FLAG (Interrupt Flag Register) 0xB0A54004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										I2S	CEC	HPDin	HPDout	SPDIF	HDCP

Field	Name	RW	Reset	Description
31-6	-	R	0	Reserved
5	I2S	R/W	0	I2S interrupt flag. (read only) 0 : not occurred 1 : interrupt occurred
4	CEC	R/W	0	CEC interrupt flag. (read only) 0 : not occurred 1 : interrupt occurred
3	HPDin	R/W	0	HPD plugged interrupt flag. If it is written by 1, it is cleared. 0 : not occurred 1 : HPD plugged
2	HPDout	R/W	0	HPD unplugged interrupt flag. If it is written by 1, it is cleared. 0 : not occurred 1 : HPD unplugged
1	SPDIF	R/W	0	SPDIF interrupt flag. (read only) 0 : not occurred 1 : interrupt occurred
0	HDCP	R/W	0	HDCP interrupt flag. (read only) 0 : not occurred 1 : interrupt occurred

AESKEY_VALID (i_aeskey_valid value) 0xB0A54008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															VALID

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	VALID	R	0	Reflects i_aeskey_valid signal value.*

* The use of aeskey_valid field depends on how aeskey is provided to HDMI 1.3 Tx subsystem. If program knows whether aeskey is valid via other way, this field may not need to be used. You can use this field in such a system where HDMI 1.3 Tx Subsystem is integrated with an AESKEY module that provides the AES key along with a valid signal tied up to aeskey_valid. Then S/W can read it to check AESKEY is valid.

HPD (HPD signal) 0xB0A5400C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															VALID

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	VALID	R	0	Value of HPD signal 0: Unplugged 1: Plugged

8.4.2 HDMI Core Registers

Name	Offset	Type	Description	Default
Control Registers				
HDMI_CON_0	0x0000	R/W	HDMI system control register 0	0x00
HDMI_CON_1	0x0004	R/W	HDMI system control register 1	0x00
HDMI_CON_2	0x0008	R/W	HDMI system control register 2	0x00
STATUS	0x0010	R/W	HDMI system status register	0x00
PHY_STATUS	0x0014	R	PHY status register	0x00
STATUS_EN	0x0020	R/W	HDMI system status enable register	0x00
HPD	0x0030	R/W	HPD control register	0x00
MODE_SEL	0x0040	R/W	HDMI/DVI mode selection	0x00
ENC_EN	0x0044	R/W	HDCP encryption enable register	0x00
Video Related Registers				
BLUE_SCREEN_0	0x0050	R/W	Pixel values for blue screen	0x00
BLUE_SCREEN_1	0x0054	R/W	Pixel values for blue screen	0x00
BLUE_SCREEN_2	0x0058	R/W	Pixel values for blue screen	0x00
HDMI_YMAX	0x0060	R/W	Maximum Y (or "R,G,B)" pixel value	0x00
HDMI_YMIN	0x0064	R/W	Minimum Y (or "R,G,B)" pixel value	0x00
HDMI_CMAX	0x0068	R/W	Maximum Cb/Cr pixel value	0x00
HDMI_CMIN	0x006C	R/W	Minimum Cb/Cr pixel value	0x00
H_BLANK_0	0x00A0	R/W	Horizontal blanking setting	0x00
H_BLANK_1	0x00A4	R/W	Horizontal blanking setting	0x00
V_BLANK_0	0x00B0	R/W	Vertical blanking setting	0x00
V_BLANK_1	0x00B4	R/W	Vertical blanking setting	0x00
V_BLANK_2	0x00B8	R/W	Vertical blanking setting	0x00
H_V_LINE_0	0x00C0	R/W	Horizontal line & vertical line setting	0x00
H_V_LINE_1	0x00C4	R/W	Horizontal line & vertical line setting	0x00
H_V_LINE_2	0x00C8	R/W	Horizontal line & vertical line setting	0x00
VSYNC_POL	0x00E4	R/W	Vertical sync polarity control register	0x00
INT_PRO_MODE	0x00E8	R/W	Interlace/Progressive control register	0x00
V_BLANK_F_0	0x0110	R/W	Vertical blanking setting for bottom field	0x00
V_BLANK_F_1	0x0114	R/W	Vertical blanking setting for bottom field	0x00
V_BLANK_F_2	0x0118	R/W	Vertical blanking setting for bottom field	0x00
H_SYNC_GEN_0	0x0120	R/W	Horizontal sync generation setting	0x00
H_SYNC_GEN_1	0x0124	R/W	Horizontal sync generation setting	0x00
H_SYNC_GEN_2	0x0128	R/W	Horizontal sync generation setting	0x00
V_SYNC_GEN1_0	0x0130	R/W	Vertical sync generation for top field or frame	0x01
V_SYNC_GEN1_1	0x0134	R/W	Vertical sync generation for top field or frame	0x10
V_SYNC_GEN1_2	0x0138	R/W	Vertical sync generation for top field or frame	0x00
V_SYNC_GEN2_0	0x0140	R/W	Vertical sync generation for bottom field - vertical position	0x01
V_SYNC_GEN2_1	0x0144	R/W	Vertical sync generation for bottom field - vertical position	0x10
V_SYNC_GEN2_2	0x0148	R/W	Vertical sync generation for bottom field - vertical position	0x00
V_SYNC_GEN3_0	0x0150	R/W	Vertical sync generation for bottom field - horizontal position	0x01
V_SYNC_GEN3_1	0x0154	R/W	Vertical sync generation for bottom field - horizontal position	0x10
V_SYNC_GEN3_2	0x0158	R/W	Vertical sync generation for bottom field - horizontal position	0x00
Audio Related Registers				

Name	Offset	Type	Description	Default
ASP_CON	0x0160	R/W	ASP packet control register	0x00
ASP_SP_FLAT	0x0164	R/W	ASP packet sp_flat bit control	0x00
ASP_CHCFG0	0x0170	R/W	ASP audio channel configuration	0x04
ASP_CHCFG1	0x0174	R/W	ASP audio channel configuration	0x1a
ASP_CHCFG2	0x0178	R/W	ASP audio channel configuration	0x2c
ASP_CHCFG3	0x017C	R/W	ASP audio channel configuration	0x3e
ACR_CON	0x0180	R/W	ACR packet control register	0x00
ACR_MCTS0	0x0184	R/W	Measured CTS value	0x01
ACR_MCTS1	0x0188	R/W	Measured CTS value	0x00
ACR_MCTS2	0x018C	R/W	Measured CTS value	0x00
ACR_CTS0	0x0190	R/W	CTS value for fixed CTS transmission mode.	0xe8
ACR_CTS1	0x0194	R/W	CTS value for fixed CTS transmission mode.	0x03
ACR_CTS2	0x0198	R/W	CTS value for fixed CTS transmission mode.	0x00
ACR_N0	0x01A0	R/W	N value for ACR packet	0xe8
ACR_N1	0x01A4	R/W	N value for ACR packet	0x03
ACR_N2	0x01A8	R/W	N value for ACR packet	0x00
ACR_LSB2	0x01B0	R/W	Alternate LSB for fixed CTS transmission mode	0x00
ACR_TXCNT	0x01B4	R/W	Number of ACR packet transmission per frame	0x1f
ACR_TXINTERVAL	0x01B8	R/W	Interval for ACR packet transmission	0x63
ACR_CTS_OFFSET	0x01BC	R/W	CTS offset for measured CTS mode	0x00
Packet Related Registers				
GCP_CON	0x01C0	R/W	ACR packet control register	0x00
GCP_BYTE1	0x01D0	R/W	GCP packet body	0x00
GCP_BYTE2	0x01D4	R/W	GCP packet body	0x00
GCP_BYTE3	0x01D8	R/W	GCP packet body	0x00
ACP_CON	0x01E0	R/W	ACP packet control register	0x00
ACP_TYPE	0x01F0	R/W	ACP packet header	0x00
ACP_DATA00~16	0x0200~ 0x00240	R/W	ACP packet body	0x00
ISRC_CON	0x0250	R/W	ACR packet control register	0x00
ISRC1_HEADER1	0x0264	R/W	ISCR1 packet header	0x00
ISRC1_DATA00~15	0x0270~ 0x02AC	R/W	ISRC1 packet body	0x00
ISRC2_DATA00~15	0x02B0~ 0x02EC	R/W	ISRC2 packet body	0x00
AVI_CON	0x0300	R/W	AVI packet control register	0x00
AVI_CHECK_SUM	0x0310	R/W	AVI packet checksum	0x00
AVI_BYTE01~13	0x0320~ 0x0350	R/W	AVI packet body	0x00
AUI_CON	0x0360	R/W	AUI packet control register	0x00
AUI_CHECK_SUM	0x0370	R/W	AUI packet checksum	0x00
AUI_BYTE1~5	0x0380~ 0x0390	R/W	AUI packet body	0x00
MPG_CON	0x03A0	R/W	MPG packet control register	0x00
MPG_CHECK_SUM	0x03B0	R/W	MPG packet checksum	0x00
MPG_BYTE1~5	0x03C0~ 0x03D0	R/W	MPG packet body	0x00
SPD_CON	0x0400	R/W	SPD packet control register	0x00
SPD_HEADER0	0x0410	R/W	SPD packet header	0x00
SPD_HEADER1	0x0414	R/W	SPD packet header	0x00
SPD_HEADER2	0x0418	R/W	SPD packet header	0x00
SPD_DATA00~27	0x0420~	R/W	SPD packet body	0x00

Name	Offset	Type	Description	Default
	0x048C			
HDCP Related Registers				
HDCP_SHA1_00~19	0x0600~ 0x064C	R/W	SHA-1 value from repeater	0x00
HDCP_KSV_LIST_0~4	0x0650~ 0x0660	R/W	KSV list from repeater	0x00
HDCP_KSV_LIST_CON	0x0664	R/W	KSV list control	0x00
HDCP_SHA_RESULT	0x0670	R/W	SHA-1 checking result register	0x00
HDCP_CTRL1	0x0680	R/W	HDCP control register1	0x00
HDCP_CTRL2	0x0684	R/W	HDCP control register2	0x00
HDCP_CHECK_RESULT	0x0690	R/W	Ri and Pj value checking result	0x00
HDCP_BKSV_0~4	0x06A0~ 0x06B0	R/W	KSV of Rx	0x00
HDCP_AKSV_0~4	0x06C0~ 0x06D0	R/W	KSV of Tx	0x00
HDCP_An_0~7	0x06E0~ 0x06FC	R/W	An value	0x00
HDCP_BCAPS	0x0700	R/W	BCAPS from Rx	0x00
HDCP_BSTATUS_0	0x0710	R/W	BSTATUS from Rx	0x00
HDCP_BSTATUS_1	0x0714	R/W	BSTATUS from Rx	0x00
HDCP_Ri_0	0x0740	R/W	Ri value of Tx	0x00
HDCP_Ri_1	0x0744	R/W	Ri value of Tx	0x00
HDCP_I2C_INT	0x0780	R/W	I2C interrupt flag	0x00
HDCP_AN_INT	0x0790	R/W	An value ready interrupt flag	0x00
HDCP_WATCGDOG_INT	0x07A0	R/W	Wachdog interrupt flag	0x00
HDCP_Ri_INT	0x07B0	R/W	Ri value update interrupt flag	0x00
HDCP_Ri_Compare_0	0x07D0	R/W	HDCP Ri Interrupt Frame number index register 0	0x80
HDCP_Ri_Compare_1	0x07D4	R/W	HDCP Ri Interrupt Frame number index register 1	0x7f
HDCP_Frame_Count	0x07E0	R	Current value of the frame count index in the hardware	0x00
GAMUT_CON	0x0500	R/W	GAMUT packet control register	0x00
GAMUT_HEADER0	0x0504	R/W	GAMUT packet header	0x00
GAMUT_HEADER1	0x0508	R/W	GAMUT packet header	0x00
GAMUT_HEADER2	0x050C	R/W	GAMUT packet header	0x00
GAMUT_DATA00~27	0x0510~ 0x057C	R/W	GAMUT packet body	0x00
DC_CONTROL	0x05C0	R/W	Deep Color Control Register	0x00
VIDEO_PATTERN_GEN	0x05C4	R/W	Video Pattern Generation Register	0x00
HPD_GEN	0x05C8	R/W	HPD Duration value register	0x01

8.4.2.1 Control Registers

HDMI_CON_0 (HDMI system control register 0) **0xB0A55000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0										Blue_Scr_En	Encoding_Option	YCBCR422_Sel	Asp_E	Power_Down	System_En

Field	Name	RW	Reset	Description
31-6	-	R	0	Reserved
5	Blue_Scr_En	R/W	0	Blue screen mode control. When set, the input video pixels are discarded and BLUESCREEN register values are transmitted for all video data period. 0: Disable 1: Enable
4	Encoding_Option	R/W	0	10-bit TMDS encoding bit order option 0: bit order reverse among the 10-bit encoding (to be set to 1 when connecting to the TMDS PHY 1.3) 1: bit order as it is
3	YCBCR422_Sel	R/W	0	Video Input mode control. 0: 4:4:4 mode 1: 4:2:2 12 bit YCbCr When 8-bit mode, the 12-bit inputs are rounded up to generate 8-bit outputs.
2	Asp_E	R/W	0	Audio sample packet generation control. This bit is only valid when SYSTEM_EN is set. 0: discard audio sample 1: When the audio sample is received, the audio sample packet is generated.
1	Power_Down	R/W	0	TMDS PHY(Transition Minimized Differential Signaling PHY) power down mode. When it's set to 0, data could not be transferred to a receiver. 0: normal operation mode 1: power down
0	System_En	R/W	0	HDMI system enable. 0: No op. 1: HDMI enable

HDMI_CON_1 (HDMI system control register 1) 0xB0A55004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Pxl_Lmt_Ctrl		0			Pxl_Rep_Ratio	

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6-5	Pxl_Lmt_Ctrl	R/W	0	Pixel value limitation control 0b00 : By-pass (Do not limit the pixel value) 0b01 : RGB mode All channel’s video input pixels are limited according to YMAX and YMIN register values. 0b10: YCbCr mode The value of I_VIDEO_G is limited according to YMAX and YMIN. The values of I_VIDEO_B and I_VIDEO_R are limited according to CMAX and CMIN. 0b11 : Reserved
4-2	-	R/W	0	Reserved
1-0	Pxl_Rep_Ratio	R/W	0	Pixel repetition ratio 0b00 : no pixel repetition 0b01 : 2 times repetition 0b10 : 3 times repetition 0b11 : 4 times repetition * Use the resulting video mode setting for pixel repetition. e.g. For 720x480p (pixel repetition = 1) Use 1440x480p video mode

HDMI_CON_2 (HDMI system control register 2) 0xB0A55008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										Vid_Period_En	0			Dvi_Band_En	0

Field	Name	RW	Reset	Description
31-6	-	R	0	Reserved
5	Vid_Period_En	R/W	0	Video preamble control. 0 : Video Preamble is applied. (HDMI mode) 1 : Video Preamble is not applied (DVI mode)
4-2	-	R/W	0	Reserved
1	Dvi_ Band_En	R/W	0	In DIV mode, the leading guard band is not used. 0 : Guard band is applied (HDMI mode) 1 : Guard band is not applied (DVI mode)
0	-	R/W	0	Reserved

STATUS (HDMI system status register) **0xB0A55010**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Authen_Ack	Aud_Fifo_Ovf	0	Update_Ri_Int	0	An_Write_Int	Watchdog_Int	I2c_Init_Int

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	Authen_Ack	R/W	0	When hdcp is authenticated, it occurs. This bit keeps the authentication signal constantly. It's not cleared at all. It's just one delayed signal of authen_ack signal from hdcp block. This bit is not an interrupt source. Read Only bit 0:not authenticated 1: authenticated
6	Aud_Fifo_Ovf	R/W	0	When audio FIFO is overflowed, this bit will be set. Once it is set, it should be cleared by host. 0: not full 1: full
5	-	R/W	0	Reserved
4	Update_Ri_Int	R/W	0	Ri Interrupt status bit If it is written by 1, it is cleared. 0: not occurred 1: interrupt occurred
3	-	R/W	0	Reserved
2	An_Write_Int	R/W	0	Indicates that {An} random value is ready. If it is written by 1, it is cleared. 0: not occurred 1: interrupt occurred
1	Watchdog_Int	R/W	0	Indicates that 2nd part of HDCP authentication protocol is initiated and CPU should set a watchdog timer to check 5 sec interval. If it is written by 1, it is cleared. 0: not occurred 1: interrupt occurred
0	I2c_Init_Int	R/W	0	Indicates that 1st part of HDCP authentication protocol can start. If it is written by 1, it is cleared. 0: not occurred 1: interrupt occurred

PHY_STATUS (PHY status register) 0xB0A55014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															Phy_Ready

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	Phy_Ready	R	0	Indicates that the PHY is ready to receive the HDMI signals from link 0: Not Ready 1: Ready

STATUS_EN (HDMI system status enable register) 0xB0A55020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Aud_Fifo_Ovr_En	0	Update_Ri_Int_En	0	An_Write_Int_En	Watchdog_Int_En	I2c_Int_En

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6	Aud_Fifo_Ovf_En	R/W	0	Audio buffer overflow interrupt enable When it is set to '1', interrupt assertion is written on the STATUS registers. 0 : Disable 1 : Enable
5	-	R/W	0	Reserved
4	Update_Ri_Int_En	R/W	0	UPDATE_RI_INT interrupt enable. 0 : Disable 1 : Enable
3	-	R/W	0	Reserved
2	An_Write_Int_En	R/W	0	AN_WRITE_INT interrupt enable. 0 : Disable 1 : Enable
1	Watchdog_Int_En	R/W	0	WATCHDOG_INT interrupt enable. 0 : Disable 1 : Enable
0	I2c_Int_En	R/W	0	I2C_INT interrupt enable. 0 : Disable 1 : Enable

HPD (HPD control register) 0xB0A55030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														Sw_Hpd	Hpd_Sel

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1	Sw_Hpd	R/W	0	When HPD_SEL bit is set, this SW_HPD signal is used for HPD (HDMI/DVI cable plugging). When set to low during HDMI transmission, status machines in HDCP core is reset. Note that other HDCP register values are not influenced. 0 : Low (unplugged) 1 : High (plugged)
0	Hpd_Sel	R/W	0	When clear, the I_HPD signal from the I/O port is used for HPD. When set, the SW_HPD signal is used for HPD. 0 : HPD signal 1 : SW_HPD internal HPD signal

MODE_SEL (HDMI/DVI mode selection) 0xB0A55040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														Hdmi_Mode	Dvi_Mode

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1	Hdmi_Mode	R/W	0	Select a mode. 0 : Disable 1 : Enable
0	Dvi_Mode	R/W	0	Select a mode. 0 : Disable 1 : Enable

ENC_EN (HDCP encryption enable register) 0xB0A55044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															Hdcp_Enc_En

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	Hdcp_Enc_En	R/W	0	When set, HDCP encryption is applied. Before setting this bit, the HDCP authentication process has to be accomplished. 0 : Encryption disable 1 : Enable

8.4.2.2 Video Related Registers

BLUESCREEN_0/1/2 (Pixel values for blue screen) **0xB0A55050/54/58**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								BLUESCREEN_0/1/2							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	BLUESCREEN_0	R/W	0	Channel 0 color setting. (Cb or B)
7-0	BLUESCREEN_1	R/W	0	Channel 1 color setting. (Y or G)
7-0	BLUESCREEN_2	R/W	0	Channel 2 color setting. (Cr or R)

HDMI_YMAX (Maximum Y (or "R,G,B)" pixel value) **0xB0A55060**
HDMI_YMIN (Minimum Y (or "R,G,B)" pixel value) **0xB0A55064**
HDMI_CMAX (Maximum Cb/Cr pixel value) **0xB0A55068**
HDMI_CMIN (Minimum Cb/Cr pixel value) **0xB0A5506C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HDMI_YMAX/UMIN/CMAX/CMIN							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved

7-0
7-0
7-0
7-0
	HDMI_YMAX
	HDMI_YMIN
	HDMI_CMAX
	HDMI_CMIN
		R/W
		R/W
		R/W
		R/W
			0xEB
			0x10
			0xF0
			0x10
				These registers are used according to PX_LMT_CTRL bits in HDMI_CON_1 register.
				
				For RGB mode
				if (i_video_x > HDMI_YMAX x 16)
				output = HDMI_YMAX x 16
				else if (i_video_x < HDMI_YMIN x 16)
				output = HDMI_YMIN x 16
				else output = i_video_x
				
				For YCbCr mode, the Y input is dealt with the same as above.
				For Cb and Cr values,
				if (i_video_x > HDMI_CMAX x 16)
				output = HDMI_CMAX x 16
				else if (i_video_x < HDMI_CMIN x 16)
				output = HDMI_CMIN x 16
				else output = i_video_x
				
				Note) Value 16 in each line compensates the difference

H_BLANK_0/1 (Horizontal blanking setting) 0xB0A550A0/A4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								H_BLANK[15:8] / HBLANK[7:0]							

Field	Name	RW	Reset	Description
31-10	-	R	0	Reserved
7-0	H_BLANK[7:0] H_BLANK[15:8]	R/W	0	Clock Cycles of horizontal Blanking Size. Refer to the Reference CEA-861D

60Hz	720x480i	720x480p	1440x480p	1280x720p	1920x1080i	1920x1080p
H_BLANK	276(114h)	138(8Ah)	276(114h)	370(172h)	280(118h)	280(118h)
50Hz	**720x576i**	**720x576p**	**1440x576p**	**1280x720p**	**1920x1080i**	**1920x1080p**
H_BLANK	288(120h)	144(90h)	288(120h)	700(2bch)	720(2d0h)	720(2d0h)

Note) 1440x480p and 1440x576p are the pixel doubling format of 720x480p and 720x576p

V_BLANK_0/1/2 (Vertical blanking setting) 0xB0A550B0/B4/B8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								V_BLANK[7:0] / V_BLANK[15:8] / V_BLANK[23:16]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	V_BLANK[7:0] V_BLANK[15:8] V_BLANK[23:16]	R/W R/W R/W	0 0 0	Vertical Blanking setting V1_BLANK (V_BLANK[21:11]) Vertical Blanking Line Size. Front Part. Refer to the Reference CEA-861D V2_BLANK (V_BLANK[10:0]) V1_BLANK+Active Lines. End Part. This value is the same as V_LINE value for progressive mode. For interlace mode, use the reference value in the table. Refer to the Reference CEA-861D

60Hz	720x480i	720x480p	1440x480p	1280x720p	1920x1080i	1920x1080p
V2_BLANK	262(d)	525(d)	525(d)	750(d)	562(d)	1125(d)
V1_BLANK	22(d)	45(d)	45(d)	30(d)	22(d)	45(d)
V_BLANK	b106(h)	16a0d(h)	16a0d(h)	f2ee(h)	b232(h)	1_6c65(h)
50Hz	**720x576i**	**720x576p**	**1440x576p**	**1280x720p**	**1920x1080i**	**1920x1080p**
V2_BLANK	312(d)	625(d)	625(d)	750(d)	562(d)	1125(d)
V1_BLANK	24(d)	49(d)	49(d)	30(d)	22(d)	45(d)
V_BLANK	c138(h)	18a71(h)	18a71(h)	F2ee(h)	B232(h)	1_6c65(h)

H_V_LINE_0/1/2 (Horizontal line & vertical line setting) 0xB0A550C0/C4/C8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								H_V_LINE[7:0] / H_V_LINE[15:8] / H_V_LINE[23:16]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	H_V_LINE[7:0] H_V_LINE[15:8] H_V_LINE[23:16]	R/W R/W R/W	0 0 0	Horizontal line & Vertical line setting H_LINE (H_V_LINE[23:12]) Horizontal Line Length. Refer to the Reference CEA-861D V_LINE (H_V_LINE[11:0]) Vertical Line Length. Refer to the Reference CEA-861D

60Hz	720x480i	720x480p	1440x480p	1280x720p	1920x1080i	1920x1080p
V_LINE	525(d)	525(d)	525(d)	750(d)	1125(d)	1125(d)
H_LINE	1716(d)	858(d)	1716(d)	1650(d)	2200(d)	2200(d)
H_V_LINE	6b420d(h)	35a20d(h)	6b420d(h)	6722ee(h)	898465(h)	898465(h)
50Hz	**720x576i**	**720x576p**	**1440x576p**	**1280x720p**	**1920x1080i**	**1920x1080p**
V_LINE	625(d)	625(d)	625(d)	750(d)	1125(d)	1125(d)
H_LINE	1728(d)	864(d)	1728(d)	1980(d)	2640(d)	2640(d)
H_V_LINE	6c0271(h)	360271(h)	6C0271(h)	7bc2ee(h)	a50465(h)	a50465(h)

VSYNC_POL (Vertical sync polarity control register) 0xB0A550E4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															V_Sync_Pol_Sel

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	V_Sync_Pol_Sel	R/W	0	Start point detection polarity selection bit. 720p or 1080i's sync shapes are different from 480p,480i, and 576p's. They are inverted shapes. 0 : active high 1 : active low Refer to the Reference CEA-861D

60Hz	720x480i	720x480p	1440x480p	1280x720p	1920x1080i	1920x1080p
VSYNC_POL	1	1	1	0	0	0
50Hz	**720x576i**	**720x576p**	**1440x576p**	**1280x720p**	**1920x1080i**	**1920x1080p**
VSYNC_POL	1	1	1	0	0	0

INT_PRO_MODE (Interlace/Progressive control register) **0xB0A550E8**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															INT_PRO_MODE

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	INT_PRO_MODE	R/W	0	Interlaced or Progressive Mode Selection. 0: progressive 1: interlaced. Refer to the Reference CEA-861D

V_BLANK_F_0/1/2 (Vertical blanking setting for bottom field) **0xB0A55110/114/118**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								V_BLANK_F[7:0] / V_BLANK_F[15:8] / V_BLANK_F[23:16]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	V_BLANK_F[7:0] V_BLANK_F[15:8] V_BLANK_F[23:16]	R/W R/W R/W	0 0 0	Vertical blanking setting for bottom field V_BOT_END (V_BLANK_F[21:11]) In the interlace mode, v_blank length of even field and odd field is different. This register specifies the end position of bottom field's active region. V_BOT_ST (V_BLANK_F[10:0]) The start position of bottom field's active region. This value is the same as V_LINE value for Interlace mode. For progressive mode, This value is not used. Refer to the Reference CEA-861D

*The above register only affects the interlace mode.

60Hz	720x480i	720x480p	1440x480p	1280x720p	1920x1080i	1920x1080p
V_BOT_ST	285(d)	Don't care	Don't care	Don't care	585(d)	Don't care
V_BOT_END	525(d)				1125(d)	
V_BLANK_F	10691d(h)				232a49(h)	
50Hz	**720x576i**	**720x576p**	**1440x576p**	**1280x720p**	**1920x1080i**	**1920x1080p**
V_BOT_ST	337(d)	Don't care	Don't care	Don't care	585(d)	Don't care
V_BOT_END	625(d)				1125(d)	
V_BLANK_F	138951(h)				232a49(h)	

H_SYNC_GEN_0/1/2 (Horizontal sync generation setting) 0xB0A55120/124/128

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								H_SYNC_GEN[7:0] / H_SYNC_GEN[15:8] / H_SYNC_GEN[23:16]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	H_SYNC_GEN[7:0] H_SYNC_GEN[15:8] H_SYNC_GEN[23:16]	R/W R/W R/W	0 0 0	Hsync_Pol (H_SYNC_GEN[20]) Set this bit for inverting the generated signal to meet the modes. In 720p and 1080i modes don't need to invert the signal. Others need to be inverted. 0 : active high 1 : active low Hsync_Edn (H_SYNC_GEN[19:10]) Set the end point of H sync. Hsync_Start (H_SYNC_GEN[9:0]) Set the start point of H sync. Refer to the Reference CEA-861D

60Hz	720x480i	720x480p	1440x480p	1280x720p	1920x1080i	1920x1080p
HSYNC_START	36(d)	14(d)	30(d)	108(d)	86(d)	86(d)
HSYNC_END	160(d)	76(d)	154(d)	148(d)	130(d)	130(d)
HSYNC_POL	1	1	1	0	0	0
H_SYNC_GEN	128024(h)	11300e(h)	12681e(h)	2506c(h)	20856(h)	20856(h)
50Hz	**720x576i**	**720x576p**	**1440x576p**	**1280x720p**	**1920x1080i**	**1920x1080p**
HSYNC_START	22(d)	10(d)	22(d)	438(d)	526(d)	526(d)
HSYNC_END	148(d)	74(d)	150(d)	478(d)	570(d)	570(d)
HSYNC_POL	1	1	1	0	0	0
H_SYNC_GEN	125016(h)	11280a(h)	125816(h)	779b6(h)	8ea0e(h)	8ea0e(h)

V_SYNC_GEN1_0/1/2 (Vertical sync generation for top field or frame) 0xB0A55130/134/138

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								V_SYNC_GEN1[7:0] / V_SYNC_GEN1[15:8] / V_SYNC_GEN1[23:16]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	V_SYNC_GEN1[7:0] V_SYNC_GEN1[15:8] V_SYNC_GEN1[23:16]	R/W R/W R/W	0x01 0x10 0x00	Vsync_T_St (V_SYNC_GEN1[23:12]) Top field (or frame) V sync start line number. Vsync_T_End (V_SYNC_GEN1[11:0]) Top field (or frame) V sync end line number. Refer to the Reference CEA-861D

60Hz	720x480i	720x480p	1440x480p	1280x720p	1920x1080i	1920x1080p
VSYNC_T_END	7(d)	15(d)	15(d)	10(d)	7(d)	9(d)
VSYNC_T_ST	4(d)	9(d)	9(d)	5(d)	2(d)	4(d)
V_SYNC_GEN1	4007(h)	900f(h)	900f(h)	500a(h)	2007(h)	4009(h)
50Hz	**720x576i**	**720x576p**	**1440x576p**	**1280x720p**	**1920x1080i**	**1920x1080p**
VSYNC_T_END	5(d)	10(d)	10(d)	10(d)	7(d)	9(d)
VSYNC_T_ST	2(d)	5(d)	5(d)	5(d)	2(d)	4(d)
V_SYNC_GEN1	2005(h)	500a(h)	500a(h)	500a(h)	2007(h)	4009(h)

V_SYNC_GEN2_0/1/2 (Vertical sync generation for bottom field) 0xB0A55140/144/148

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								V_SYNC_GEN2[7:0] / V_SYNC_GEN2[15:8] / V_SYNC_GEN2[23:16]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	V_SYNC_GEN2[7:0] V_SYNC_GEN2[15:8] V_SYNC_GEN2[23:16]	R/W R/W R/W	0x01 0x10 0x00	Vsync_B_St (V_SYNC_GEN2[23:12]) Bottom field V sync start line number. Vsync_B_End (V_SYNC_GEN2[11:0]) Bottom field V sync end line number. Refer to the Reference CEA-861D

60Hz	720x480i	720x480p	1440x480p	1280x720p	1920x1080i	1920x1080p
VSYNC_B_END	269(d)	Don't Care	Don't Care	Don't Care	569(d)	Don't Care
VSYNC_B_ST	266(d)				564(d)	
V_SYNC_GEN2	10a10d(h)				234239(h)	
50Hz	**720x576i**	**720x576p**	**1440x576p**	**1280x720p**	**1920x1080i**	**1920x1080p**
VSYNC_B_END	317(d)	Don't Care	Don't Care	Don't Care	569(d)	Don't Care
VSYNC_B_ST	314(d)				564(d)	
V_SYNC_GEN2	13a13d(h)				234239(h)	

V_SYNC_GEN3_0/1/2 (Vertical sync generation for bottom field) **0xB0A55150/154/158**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								V_SYNC_GEN3[7:0] / V_SYNC_GEN3[15:8] / V_SYNC_GEN3[23:16]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	V_SYNC_GEN3[7:0] V_SYNC_GEN3[15:8] V_SYNC_GEN3[23:16]	R/W R/W R/W	0x01 0x10 0x00	Vsync_H_Pos_St (V_SYNC_GEN3[23:12]) Bottom field V sync start transition point. Vsync_H_Pos_End (V_SYNC_GEN3[11:0]) Bottom field V sync end transition point. Refer to the Reference CEA-861D

60Hz	720x480i	720x480p	1440x480p	1280x720p	1920x1080i	1920x1080p
VSYNC_H_POS_ST	896(d)	Don't Care	Don't Care	Don't Care	1188(d)	Don't Care
VSYNC_H_POS_END	896(d)				1188(d)	
V_SYNC_GEN3	380380(h)				4A44A4(h)	
50Hz	**720x576i**	**720x576p**	**1440x576p**	**1280x720p**	**1920x1080i**	**1920x1080p**
VSYNC_H_POS_ST	888(d)	Don't Care	Don't Care	Don't Care	1848(d)	Don't Care
VSYNC_H_POS_END	888(d)				1848(d)	
V_SYNC_GEN3	378378(h)				738738(h)	

8.4.2.3 Audio related Packet Registers

ASP_CON (ASP packet control register) **0xB0A55160**

7	6	5	4	3	2	1	0
DST_Double	Aud_Type		Aud_Mode	SP_Pre			

Field	Name	RW	Reset	Description
7	DST_Double	R/W	0	DST double
6-5	Aud_Type	R/W	0	Packet type instead of audio type 00 : Audio Sample Packet 01 : One-bit audio packet 10 : HBR packet 11 : DST packet
4	Aud_Mode	R/W	0	Two channel or multi-channel mode selection This bit will be also used for layout bit in ASP header. 0 : 2 channel mode 1 : Multi channel mode. Set this bit to transmit HBR packets.
3-0	SP_Pre	R/W	0	Control sub-packet usage for multi channel mode only. When two-channel mode, this register value is not used.

ASP_SP_FLAT (ASP packet sp_flat bit control) **0xB0A55164**

7	6	5	4	3	2	1	0
				SP_Flat			

Field	Name	RW	Reset	Description
7-4	-	R/W	0	Reserved
3-0	SP_Flat	R/W	0	The sp_flat/sample_invalid value in the ASP header. Refer to the HDMI specification v1.3 (5.3.4 and 5.3.9)

ASP_CHCFG0/1/2/3 (ASP audio channel configuration) **0xB0A55170/174/178/17C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ASP_CHCFG[7:0] / ASP_CHCFG[15:8] / ASP_CHCFG[23:16] / ASP_CHCFG[31:24]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ASP_CHCFG[7:0] ASP_CHCFG[15:8] ASP_CHCFG[23:16] ASP_CHCFG[31:24]	R/W R/W R/W R/W	0x04 0x1A 0x2C 0x3E	Spk3R_Sel (ASP_CHCFG[29:27]) Audio channel Selection for subpacket 3 right channel data in multi channel mode. 000 : i_pcm0L is used for sub packet 0 Left channel 001 : i_pcm0R is used for sub packet 0 Left channel 010 : i_pcm1L is used for sub packet 0 Left channel 011 : i_pcm1R is used for sub packet 0 Left channel 100 : i_pcm2L is used for sub packet 0 Left channel 101 : i_pcm2R is used for sub packet 0 Left channel 110 : i_pcm3L is used for sub packet 0 Left channel 111 : i_pcm3R is used for sub packet 0 Left channel Spk3L_Sel (ASP_CHCFG[26:24]) Audio channel Selection for subpacket 3 left channel data in multi channel mode. The meaning is the same as Spk3R_SEL. Spk2R_Sel (ASP_CHCFG[21:19]) Audio channel Selection for subpacket 2 right channel data in multi channel mode. The meaning is the same as Spk3R_SEL. Spk2L_Sel (ASP_CHCFG[18:16]) Audio channel Selection for subpacket 2 left channel data in multi channel mode. The meaning is the same as Spk3R_SEL. Spk1R_SEL (ASP_CHCFG[13:11]) Audio channel Selection for subpacket 1 right channel data in multi channel mode. The meaning is the same as Spk3R_SEL. Spk1L_Sel (ASP_CHCFG[10:8]) Audio channel Selection for subpacket 1 left channel data in multi channel mode. The meaning is the same as Spk3R_SEL. Spk0R_Sel (ASP_CHCFG[5:3]) Audio channel Selection for subpacket 0 right channel data in multi channel mode. The meaning is the same as Spk3R_SEL. Spk0L_Sel (ASP_CHCFG[2:0]) Audio channel Selection for subpacket 0 right channel data in multi channel mode. The meaning is the same as Spk3R_SEL.

ACR_CON (ACR packet control register) 0xB0A55180

7	6	5	4	3	2	1	0
			Alt_Cts_Rate		ACR_Tx_Mode		

Field	Name	R/W	Reset	Description
7-5	-	R/W	0b000	Reserved
4-3	Alt_Cts_Rate	R/W	0b00	In some audio format, the CTS value can be changed alternately. CTS value 1 = ACR_CTS[19:0] CTS value 2 = {ARC_CTS[19:8], ACR_LSB2} These two values can be transmitted alternately at the ratio of this register setting. 00 : always CTS value 1 01 : 1:1 (CTS value 1 : CTS value2) 10 : 2:1 (CTS value 1 : CTS value2) 11 : 3:1 (CTS value 1 : CTS value2) Measured CTS mode, this value is not used.
2-0	ACR_Tx_Mode	R/W	0b000	000: Do not Tx (Transfer) the ACR packet. 001: Tx once – Transmit ACR packet once when anytime available after this value is set. After transmitting, these bits are reset to all zero. 010 : Tx ACR_TXCNT times during every VBI period 011 : Tx by counting i_clk_vid for a given CTS value in the ACR_CTS0~2 registers. 100 : Measured CTS mode. Make ACR packet with CTS value by counting TMDS clock for Fs x 128 / N duration. In this case, the 7 LSBs of N value (ACR_N register) should be all zero.

ACR_MCTS0/1/2 (Measured CTS value) 0xB0A55184/188/18C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ACR_MCTS[7:0] / ACR_MCTS[15:8] / ACR_MCTS[23:16]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ACR_MCTS[7:0] ACR_MCTS[15:8] ACR_MCTS[23:16]	R/W R/W R/W	0x01 0x00 0x00	ACR_MCTS (ACR_MCTS[19:0]) This value is the TMDS clock cycles for N[19:7] number of audio sample inputs. It is only valid when measured CTS mode is set on ACR_CON register.

ACR_CTS0/1/2 (CTS value for fixed CTS transmission mode.) 0xB0A55190/194/198

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ACR_CTS[7:0] / ACR_CTS[15:8] / ACR_CTS[23:16]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ACR_CTS[7:0] ACR_CTS[15:8] ACR_CTS[23:16]	R/W R/W R/W	0xE8 0x03 0x00	ACR_CTS (ACR_CTS[19:0]) CTS value for transmission mode other than ‘measured CTS’ mode.

ACR_N0/1/2 (N value for ACR packet) 0xB0A551A0/1A4/1A8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ACR_N[7:0] / ACR_N[15:8] / ACR_N[23:16]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ACR_N[7:0] ACR_N[15:8] ACR_N[23:16]	R/W R/W R/W	0xE8 0x03 0x00	ACR_N (ACR_N[19:0] The N value in ACR packet

ACR_LSB2 (Alternate LSB for fixed CTS transmission mode) 0xB0A551B0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ACR_LSB2							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ACR_LSB2	R/W	0	Alternate CTS least significant byte See ALT_CTS_RATE in ACR_CON register.

ACR_TXCNT (Number of ACR packet transmission per frame) 0xB0A551B4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											ACR_TXCNT				

Field	Name	RW	Reset	Description
31-5	-	R	0	Reserved
4-0	ACR_TXCNT	R/W	0x1F	When ACR_TX_MODE is ‘10’, the ACR packet will be transmitted ACR_TXCNT + 1’ times per every VBI period. ALT_CTS_RATE is also applied. This register is only valid when ACR_TX_MODE is ‘10’.

ACR_TXINTERVAL (Interval for ACR packet transmission) 0xB0A551B8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ACR_TXINTERVAL							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ACR_TX_INTERVAL	R/W	0x63	When ACR_TX_MODE is ‘10’, the ACR packet will be transmitted ACR_TXCNT times during VBI. This register specifies the number of cycles between each ACR packets. This register is used to avoid continuous transmission more than 18 packets within single DI band. This register is only valid when ACR_TX_MODE is ‘10’.

ACR_CTS_OFFSET (CTS offset for measured CTS mode) 0xB0A551BC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ACR_CTS_OFFSET							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ACR_CTS_OFFSET	R/W	0x00	When 'measured CTS mode' is used, the CTS value will be measured by counting the TMDS clock for a given duration. This value is added to measured CTS value. It is 8 bit signed integer, so subtraction is possible.

GCP_CON (ACR packet control register) 0xB0A551C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												ENABLE_1st_VSYNC	ENABLE_2nd_VSYNC	GCP_CON	

Field	Name	RW	Reset	Description
31-3	-	R	0	Reserved
3	ENABLE_1st_VSYNC	R/W	0	For Interlace mode, Enable this bit to transfer the GCP packet on the 1st VSYNC in a frame 0: Do not transfer GCP packet 1: Transfer GCP packet * For Progressive mode, GCP packet is transferred regardless of this bit. i.e. GCP packet in progressive mode is transferred every vsync if GCP_CON is 0b1x
2	ENABLE_2nd_VSYNC	R/W	1	For Interlace mode, Enable this bit to transfer the GCP packet on the 2nd VSYNC in a frame 0: Do not transfer GCP packet 1: Transfer GCP packet
1-0	GCP_CON	R/W	0	00 : Do not transmit 01 : Transmit once 1x : Transmit every vsync. GCP packet will be transmitted within 384 cycles after active vsync.

GCP_BYTE1 (GCP packet body) 0xB0A551D0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								GCP_BYTE1							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	GCP_BYTE1	R/W	0	GCP packet's first data byte. It shall be either 0x10 (Clear AVMUTE) or 0x01 (Set AVMUTE).

GCP_BYTE2 (GCP packet body) 0xB0A551D4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								PP				CD			

Field	Name	RW	Reset	Description
7-4	PP	R/W	0x0	PP (Packing Phase), Read Only
3-0	CD	R/W	0x0	CD (Color Depth)

GCP_BYTE3 (GCP packet body) 0xB0A551D8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0															GCP_ BYTE3

Field	Name	RW	Reset	Description
7-1	-	-	0	Reserved
0	GCP_BYTE3	R/W	0	Default State

8.4.2.4 ACP Packet Registers

ACP_CON (ACP packet control register) **0xB0A551E0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ACP_FR_RATE					0	ACP_TX_CON	

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-3	ACP_FR_RATE	R/W	0	Transmit ACP packet once per every ACP_FR_RATE+1 frames (or fields).
2	-	-	0	Reserved
1-0	ACP_TX_CON	R/W	0	00 : Do not transmit 01 : Transmit once 1x : Transmit every vsync with ACP_FR_RATE

ACP_TYPE (ACP packet header) **0xB0A551F0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ACP_TYPE							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ACP_TYPE	R/W	0	ACP packet header. (HB1 of ACP packet header) See Table 5-18 in HDMI v1.3 specification

ACP_DATA00~16 (ACP packet body) **0xB0A55200~0xB0A55400**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ACP_DATA00 ~ 16							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ACP_DATA00~16	R/W	0	ACP packet body data. (PB0~PB16 of ACP packet body) See 9.3 in HDMI v1.3 specification

8.4.2.5 ISRC1/2 packet registers

ISRC_CON (ACR packet control register) **0xB0A55250**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ISRC_FR_RATE					ISRC2_EN	ISRC_TX_CON	

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-3	ISRC_FR_RATE	R/W	0	Transmit ISRC1 (with ISRC2 or not) packet once per every ISRC_FR_RATE+1 frames (or fields).
2	ISRC2_EN	R/W	0	Transmit ISRC2 packet with ISRC1 packet
1-0	ISRC_TX_CON	R/W	0	00 : Do not transmit 01 : Transmit once 1x : Transmit every vsync with ISRC_FR_RATE

ISRC1_HEADER1 (ISCR1 packet header) **0xB0A55264**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ISRC_Cont	ISRC_Valid	0			SRC_Status		

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	ISRC_Cont	R/W	0	See table 5-20 in HDMI v1.3specification
6	ISRC_Valid	R/W	0	See table 5-20 in HDMI v1.3 specification
5-3	-	R	0	Reserved
2-0	ISRC status	R/W	0	See table 5-20 in HDMI v1.3 specification

ISRC1_DATA 00~15 (ISRC1 packet body) **0xB0A55270~0xB0A552AC**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ISRC1_DATA 00 ~ 15							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ISRC1_DATA00~15	R/W	0	ISRC2 packet body data. (PB0~15 of ISRC2 packet body) See Table 5-21 in HDMI v1.3 specification.

SRC2_DATA 00~15 (ISRC2 packet body) **0xB0A552B0~0xB0A552EC**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								ISRC2_DATA 00 ~ 15							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	ISRC2_DATA00~15	R/W	0	ISRC2 packet body data. (PB0~15 of ISRC2 packet body) See Table 5-21 in HDMI v1.3 specification.

8.4.2.6 AVI InfoFrame registers

AVI_CON (AVI packet control register) **0xB0A55300**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														AVI_TX_CON	

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1-0	AVI_TX_CON	R/W	0	00 : Do not transmit 01 : Transmit once 1x : Transmit every vsync

AVI_CHECK_SUM (AVI packet checksum) **0xB0A55310**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								AVI_CHECK_SUM							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	AVI_CHECK_SUM	R/W	0	AVI InfoFrame checksum byte. (PB0 byte of AVI packet body)

AVI_DATA01 ~ AVI_DATA13 (AVI packet body) **0xB0A55320~0xB0A5350**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								AVI_DATA 01~13							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	AVI_DATA01~ AVI_DATA13	R/W	0	AVI Infoframe packet data registers. (PB1~PB13 bytes of AVI packet body)

8.4.2.7 Audio InfoFrame registers

AUI_CON (AUI packet control register) **0xB0A55360**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														AUI_TX_CON	

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1-0	AUI_TX_CON	R/W	0b00	00 : Do not transmit 01 : Transmit once 1x : Transmit every vsync

AUI_CHECK_SUM (AUI packet checksum) **0xB0A55370**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								AUI_CHECK_SUM							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	AUI_CHECK_SUM	R/W	0	AUI Infoframe checksum data. (PB0 byte of AUI packet body)

AUI_DATA1 ~ AUI_DATA5 (AUI packet body) **0xB0A55380~0xB0A55390**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								AUI_DATA 1~5							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	AUI_DATA1~5	R/W	0	AUI Infoframe packet body. (PB1~PB5 bytes of AUI packet body)

8.4.2.8 MPEG Source InfoFrame

MPG_CON (MPG packet control register) **0xB0A553A0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														MPG_TX_CON	

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1-0	MPG_TX_CON	R/W	0	00 : Do not transmit 01 : Transmit once 1x : Transmit every vsync

MPG_CHECK_SUM (MPG packet checksum) **0xB0A553B0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								MPG_CHECK_SUM							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	MPG_CHECK_SUM	R/W	0	MPG infoframe checksum register (PB0 byte of MPG packet body)

MPG_DATA1 ~ MPG_DATA5 (MPG packet body) **0xB0A553C0~0xB0A553D0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								MPG_DATA 1~5							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	MPGI_DATA1~5	R/W	0	MPG Infoframe packet data. (PB1~PB5 bytes of MPG packet body)

8.4.2.9 Source Product Descriptor InfoFrame (or general packet generation)

These registers can be used for Source Product Descriptor (SPD) packet transmission. Furthermore, they consist of full configurable header and packet body registers (3 bytes header register and 28 bytes packet body registers) so that they can be used for transmission of any type of packet.

SPD_CON (SPD packet control register) **0xB0A55400**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														SPD_TX_CON	

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1-0	SPD_TX_CON	R/W	0	00 : Do not transmit 01 : Transmit once 1x : Transmit every vsync

SPD_HEADER0/1/2 (SPD packet header) **0xB0A55410/414/418**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								SPD_HEADER 0/1/2							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	SPD_HEADER0	R/W	0	HB0 byte of SPD packet header
	SPD_HEADER1	R/W	0	HB1 byte of SPD packet header
	SPD_HEADER2	R/W	0	HB2 byte of SPD packet header

SPD_DATA00~27 (SPD packet body) **0xB0A55440~0xB0A5548C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								SPD_DATA 00 ~ 27							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	SPD_DATA00~ SPD_DATA27	R/W	0	SPD packet data registers. (PB0~PB27 bytes)

8.4.2.10 HDCP Register Description

HDCP_SHA1_00~19 (SHA-1 value from repeater) **0xB0A55600~0xB0A5564C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HDCP_SHA1[7:0] … HDCP_SHA1[159:152]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	HDCP_SHA1[7:0] … HDCP_SHA1[159:152]	R/W	0	HDCP_SHA1 (HDCP_SHA1[159:0] 160-bit HDCP repeater's SHA-1 value. Least significant byte first. These registers are readable but they are not modified by HDCP H/W.

HDCP_KSV_LIST_0~4 (KSV list from repeater) **0xB0A55650~0xB0A55660**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HDCP_KSV_LIST[7:0] … HDCP_KSV_LIST[39:32]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	HDCP_KSV_LIST[7:0] … HDCP_KSV_LIST[39:32]	R/W	0	HDCP_KSV_LIST (HDCP_KSV_LIST[39:0] Little endian addressing. One KSV value of the HDCP repeater's KSV list. These registers are readable.

HDCP_KSV_LIST_CON (KSV list control) 0xB0A55664

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												Hdcp_Ksv_Write_Done	Hdcp_Ksv_List_Empty	Hdcp_Ksv_End	Hdcp_Ksv_Read

Field	Name	RW	Reset	Description
31-4	-	R	0	Reserved
3	Hdcp_Ksv_Write_Done	R/W	0	After writing KSV data into HDCP_KSV_LIST_X registers and then writing the value “1” to this register, HW processes the written KSV value and clears this bit to “0” 0: Not yet written 1: Written
2	Hdcp_Ksv_List_Empty	R/W	0	If the number of KSV list is zero, set this value to make SHA-1 module to start to calculate without KSV list. 0: Not empty 1: Empty
1	Hdcp_Ksv_End	R/W	0	It is used to indicate that current KSV value in HDCP_KSV_LIST_X registers is the last one. 0: Not End 1: End
0	Hdcp_Ksv_Read	R/W	0	After writing KSV data into HDCP_KSV_LIST_X registers HDCP SHA-1 module keeps that KSV value into internal buffer and set this flag into ‘1’ to notify that it has been read. After checking that it is set to ‘1’, SW clears to ‘0’ at the same time when writing the HDCP_KSV_WRITE_DONE bit for next KSV list value. 0: Not Read 1: Read

HDCP_SHA_RESULT (SHA-1 checking result register) 0xB0A55670

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														Hdcp_Sha_Valid_Ready	Hdcp_Sha_Valid

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1	Hdcp_Sha_Valid_Ready	R/W	0	Indicates that the SHA comparison has been done by the HW. Must be cleared by SW by writing 0 0: Not ready 1: Ready
0	Hdcp_Sha_Valid	R/W	0	Indicates that the SHA-1 comparison succeeds. Must be cleared by SW by writing 0 0: Valid 1: Not valid

HDCP_CTRL1 (HDCP control register1) 0xB0A55680

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													Timeout	CP_Desired	0

Field	Name	RW	Reset	Description
31-3	-	R	0	Reserved
2	Timeout	R/W	0	Set when Rx is repeater and its KSV list is not ready until 5 sec waiting. 0: Not timeout 1: Timeout(KSV Ready bit in the HDCP_BCAPS register is not high until 5 sec) and re-start the 1st authentication. Refer to the Figure 2-6 in the HDCP 1.3 specification
1	CP_Desired	R/W	0	HDCP enable 0: Not Desired 1: Desired
0	-	R	0	Reserved

HDCP_CTRL2 (HDCP control register2) 0xB0A55684

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															Revocation_Set

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	Revocation_Set	R/W	0	KSV list is on the revocation list & Fail the 2nd authentication. 0: Revocation Not set 1: Revocation Set

HDCP_CHECK_RESULT (Ri and Pj value checking result) 0xB0A55690

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															Ri_Match_Result

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1-0	Ri_Match_Result	R/W	0	Write the result of comparison between Ri of Rx and Tx as the following values. (Ri : Tx, Ri' : Rx) Must be cleared by SW after setting 10 or 11 before next Ri Interrupt occurs. 0x : don't care 10 : Ri ≠ Ri' 11 : Ri = Ri'

HDCP_BKSV0~4 (KSV of Rx) 0xB0A556A0~0xB0A556B0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HDCP_BKSV[7:0] … HDCP_BKSV[39:32]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	HDCP_BKSV[7:0] … HDCP_BKSV[39:32]	R/W	0	HDCP_BKSV (HDCP_BKSV[39:0]) Key selection vector (KSV) value from receiver.

HDCP_AKSV0~4 (KSV of Tx) 0xB0A556C0~0xB0A556D0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HDCP_AKSV[7:0] … HDCP_AKSV[39:32]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	HDCP_AKSV[7:0] … HDCP_AKSV[39:32]	R/W	0	HDCP_AKSV (HDCP_AKSV [39:0]) KSV value of transmitter

HDCP_An_0~7 (An value) 0xB0A556E0~0xB0A556FC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HDCP_An[7:0] … HDCP_An[63:56]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	HDCP_An[7:0] … HDCP_An[63:56]	R/W	0	HDCP_An (HDCP_An[63:0]) 64-bit Random number generated by Tx (An)

HDCP_BCAPS (BCAPS from Rx) 0xB0A55700

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Repeater	Ready	Fast	0		1.1_Features	Fast_Reauthentication

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6	Repeater	R/W	0	The receiver supports downstream connections 0: Not Repeater 1: Repeater
5	Ready	R/W	0	KSV FIFO, SHA-1 calculation ready 0: Not Ready 1: Ready
4	Fast	R/W	0	The receiver devices supports 400KHz transfer 0: Not Fast 1: Fast
3-2	-	R/W	0	Must be 0’s
1	1.1_Features	R/W	0	Supports EESS, Advance cipher, Enhanced link verification 0: un-set 1: set
0	Fast_Reauthentication	R/W	0	ALL HDMI receiver should be capable of reauthentication 0: un-set 1: set

HDCP_BSTATUS_0/1 (BSTATUS from Rx) 0xB0A55710/714

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HDCP_BSTATUS[7:0] / HDCP_BSTATUS[15:8]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	HDCP_BSTATUS[7:0] HDCP_BSTATUS[15:8]	R/W	0	Hdmi_Mode (HDCP_BSTATUS[12]) HDMI mode indication. If set, HDCP is in HDMI mode. Max_Cascade_Exceeded (HDCP_BSTATUS[11]) Topology error Depth (HDCP_BSTATUS[10:8]) Cascade depth Max_Devs_Exceeded (HDCP_BSTATUS[7]) Topology error indicator 0: No Error 1: Error Device_Count (HDCP_BSTATUS[6:0]) Total number of attached downstream devices

HDCP_Ri_0/1 (Ri value of Tx) 0xB0A55740/744

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HDCP_Ri[7:0] / HDCP_Ri[15:8]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	HDCP_Ri[7:0] HDCP_Ri[15:8]	R/W R/W	0	HDCP Ri value of the transmitter.

HDCP_I2C_INT (I2C interrupt flag) 0xB0A55780

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															HDCP_I2C_INT

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	HDCP_I2C_INT	R/W	0	HDCP I2C Interrupt status. Active high. It indicates the start of I2C transaction when it is set. After active, it should be cleared by S/W by writing 0. 0: Not Occurred 1: Occurred

HDCP_AN_INT (An value ready interrupt flag) 0xB0A55790

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															HDCP_AN_INT

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	HDCP_AN_INT	R/W	0	HDCP An Interrupt status. Active high. If An value is available, it is set. After active, it should be cleared by S/W by writing 0. 0: Not Occurred 1: Occurred

HDCP_WATCHDOG_INT (Wachdog interrupt flag) 0xB0A557A0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															HDCP_WATCHDONG_INT

Field	Name	RW	Reset	Description
31-1	-	-	0	Reserved
0	HDCP_WATCHDOG_INT	R/W	0	HDCP Watchdog Interrupt status. Active high. If Repeater bit value is set after 1st authentication success, it is set. After active, it should be cleared by S/W by writing 0. 0: Not Occurred 1: Occurred

HDCP_RI_INT (Ri value update interrupt flag) 0xB0A557B0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															HDCP_RI_INT

Field	Name	RW	Reset	Description
31-1	-	-	0	Reserved
0	HDCP_RI_INT	R/W	0	If Ri value is updated internally (at every 128 video frames), It is set to high. After set, it should be cleared by S/W by writing 0. 0: Not Occurred 1: Occurred

8.4.2.11 Ri Check Registers

HDCP_Ri_Compare_0 (HDCP Ri Interrupt Frame number index register 0) **0xB0A557D0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Enable	Frame Number Index						

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	Enable	R/W	1	Enable the interrupt for this frame number index
6-0	Frame Number index	R/W	0	When the Frame count reaches this “Frame number index”, an Ri Link integrity check interrupt occurs

HDCP_Ri_Compare_1 (HDCP Ri Interrupt Frame number index register 1) **0xB0A557D4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Enable	Frame Number Index						

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	Enable	R/W	1	Enable the interrupt for this frame number index
6-0	Frame Number index	R/W	0x7F	When the Frame count reaches this “Frame number index”, an Ri Link integrity check interrupt occurs

HDCP_Frame_Count (Current value of the frame count index) **0xB0A557E0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Count						

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6-0	Frame Count	R	0	Current value of the frame count index in the hardware

8.4.2.12 Gamut Metadata Packet Registers

GAMUT_CON (GAMUT packet control register) **0xB0A55500**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														GAMUT_CON	

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1-0	GAMUT_CON	R/W	0	00 : Do not transmit 01 : Transmit once 1x : Transmit every vsync

GAMUT_HEADER0 (GAMUT packet header) **0xB0A55504**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HB0							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	HB0	R/W	0	HB0 value in the table 5-30 in HDMI 1.3 specification

GAMUT_HEADER1 (GAMUT packet header) **0xB0A55508**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Next_Field	GBD_profile			Affected_Gamut_Seq_Num			

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	Next_Field	R/W	0	Set to indicate that the GBD carried in this packet will be effective on the next video field.
6-4	GBD_profile	R/W	0	Transmission profile number (We only support profile 0)
3-0	Affected_Gamut_Seq_Num	R/W	0	Indicates which video fields are relevant for this metadata

GAMUT_HEADER2 (GAMUT packet header) 0xB0A5550C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								No_Crnt_GBD	0	Packet_Seq		Current_Camut_Seq_Num			

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	No_Crnt_GBD	R/W	0	Set to indicate that there is no gamut metadata available for the currently transmitted video
6	-	R/W	0	Reserved
5-4	Packet_Seq	R/W	0	Indicates whether this packet is the only, the first, an intermediate or the last packet in a Gamut metadata packet sequence
3-0	Current_Gamut_Seq_Num	R/W	0	Indicates the gamut number of the currently transmitted video stream

GAMUT_METADATA0~27 (GAMUT packet body) 0xB0A55510~0xB0A5557C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								GAMUT_METADATA 0 ~ 27							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	GAMUT_METADATA0~ GAMUT_METADATA27	R/W	0	Gamut Metadata Packet body for P0 transmission profile

8.4.2.13 Video Mode Registers

DC_CONTROL (Deep Color Control Register) **0xB0A555C0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														Deep Color Mode	

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1-0	Deep Color Mode	R/W	0	00: 8 bits /pixel 01:10bits /pixel 10:12 bits / pixel 11: Not Used

VIDEO_PATTERN_GEN (Video Pattern Generation Register) **0xB0A555C4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														Ext_Video_En	Video_Pattern_Enable

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1	Ext_Video_En	R/W	0	0: Ext Off 1: Ext En
0	Video Pattern Enable	R/W	0	0: Disable 1: Use Internally generated video pattern

HPD_GEN (HPD Duration value register) **0xB0A555C8**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								HPD_Duration							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	HPD_Duration	R/W	0x01	Num of cycles for determining stable HPD input Internal count = TMDS clock * HPD_Duration * 16 (cycles) Default value = 0x1 (16 TMDS clock cycles)

INTERNAL_VIDEO **0xB0A55900~0xB0A55A7C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								R_00_00_0 … R_11_11_1 / G_00_00_0 … G_11_11_1 / B_00_00_0 … B_11_11_1							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	R_00_00_0 ~ B_11_11_1	R/W	0	-Generation of Internal Video RGB Data -Can be set every 256 line and 256 pixel count values - 12bit R, G, B values are separated on 2 registers. ex) r_00_00 => R pixel data on 0~256th line & 0~256th pixel in a frame. r_00_00_0 => r_00_00[11:4], r_00_00_1 => r_00_00[3:0]

16 (line) * 2 (bytes) * 3 (colors) * 4 (word size) = 384(d) = 180(h)

8.4.3 AES Registers

Name	Offset	Type	Description	Default
AES_START	0x0000	R/W	AES_START	0x00
AES_DATA_SIZE_L	0x0020	R/W	AES_DATA_SIZE_L	0x20
AES_DATA_SIZE_H	0x0024	R/W	AES_DATA_SIZE_H	0x01
AES_DATA	0x0040	W	AES_DATA	-

AES_START **0xB0A56000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															AES_Start

Field	Name	RW	Reset	Description
31-1	Reserved	R	0	Reserved
0	AES_Start	R/W	0	AES Start signal If specified amount of data is decrypted and written in memory, then AES start signal goes to 0. 0 : AES does not decrypt data. (AES decryption completed) 1 : AES starts to decrypt data from memory.

AES_DATA_SIZE_L/H **0xB0A56020/24**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								AES_Data_Size_L / AES_Data_Size_H							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	AES_Data_Size_L AES_Data_Size_H	R/W R/W	0x20 0x01	AES data size (in bytes) to decrypt If the data size is not the multiple of 128 bits, zeros should be padded 128bit align Maximum number is 120h(internal memory size limits the maximum data size Default value : 288 bytes

AES_DATA **0xB0A56040**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								AES_Data							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	AES_Data	R/W	0	Write buffer to store AES-encrypted data in memory before starting decryption Memory address is automatically increased. Zeros should be padded for 128 bit align.

8.4.4 SPDIF Registers

Name	Offset	Type	Description	Default
SPDIFIN_CLK_CTRL	0x0000	R/W	SPDIFIN Clock Control Register	0x02
SPDIFIN_OP_CTRL	0x0004	R/W	SPDIFIN Operation Control Register 1	0x00
SPDIFIN_IRQ_MASK	0x0008	R/W	SPDIFIN Interrupt Request Mask Register	0x00
SPDIFIN_IRQ_STATUS	0x000C	R/W	SPDIFIN Interrupt Request Status Register	0x00
SPDIFIN_CONFIG_1	0x0010	R/W	SPDIFIN Configuration Register 1	0x02
SPDIFIN_CONFIG_2	0x0014	R/W	SPDIFIN Configuration Register 2	0x00
-	0x0018	-	Reserved	-
-	0x001C	-	Reserved	-
SPDIFIN_USER_VALUE_1	0x0020	R	SPDIFIN User Value Register 1	0x00
SPDIFIN_USER_VALUE_2	0x0024	R	SPDIFIN User Value Register 2	0x00
SPDIFIN_USER_VALUE_3	0x0028	R	SPDIFIN User Value Register 3	0x00
SPDIFIN_USER_VALUE_4	0x002C	R	SPDIFIN User Value Register 4	0x00
SPDIFIN_CH_STATUS_0_1	0x0030	R	SPDIFIN Channel Status Register 0-1	0x00
SPDIFIN_CH_STATUS_0_2	0x0034	R	SPDIFIN Channel Status Register 0-2	0x00
SPDIFIN_CH_STATUS_0_3	0x0038	R	SPDIFIN Channel Status Register 0-3	0x00
SPDIFIN_CH_STATUS_0_4	0x003C	R	SPDIFIN Channel Status Register 0-4	0x00
SPDIFIN_CH_STATUS_1	0x0040	R	SPDIFIN Channel Status Register 1	0x00
-	0x0044	-	Reserved	-
SPDIFIN_FRAME_PERIOD_1	0x0048	R	SPDIFIN Frame Period Register 1	0x00
SPDIFIN_FRAME_PERIOD_2	0x004C	R	SPDIFIN Frame Period Register 2	0x00
SPDIFIN_Pc_INFO_1	0x0050	R	SPDIFIN Pc Info Register 1	0x00
SPDIFIN_Pc_INFO_2	0x0054	R	SPDIFIN Pc Info Register 2	0x00
SPDIFIN_Pd_INFO_1	0x0058	R	SPDIFIN Pd Info Register 1	0x00
SPDIFIN_Pd_INFO_2	0x005C	R	SPDIFIN Pd Info Register 2	0x00
SPDIFIN_DATA_BUF_0_1	0x0060	R	SPDIFIN Data Buffer Register 0_1	0x00
SPDIFIN_DATA_BUF_0_2	0x0064	R	SPDIFIN Data Buffer Register 0_2	0x00
SPDIFIN_DATA_BUF_0_3	0x0068	R	SPDIFIN Data Buffer Register 0_3	0x00
SPDIFIN_USER_BUF_0	0x006C	R	SPDIFIN User Buffer Register 0	0x00
SPDIFIN_DATA_BUF_1_1	0x0070	R	SPDIFIN Data Buffer Register 1_1	0x00
SPDIFIN_DATA_BUF_1_2	0x0074	R	SPDIFIN Data Buffer Register 1_2	0x00
SPDIFIN_DATA_BUF_1_3	0x0078	R	SPDIFIN Data Buffer Register 1_3	0x00
SPDIFIN_USER_BUF_1	0x007C	R	SPDIFIN User Buffer Register 1	0x00

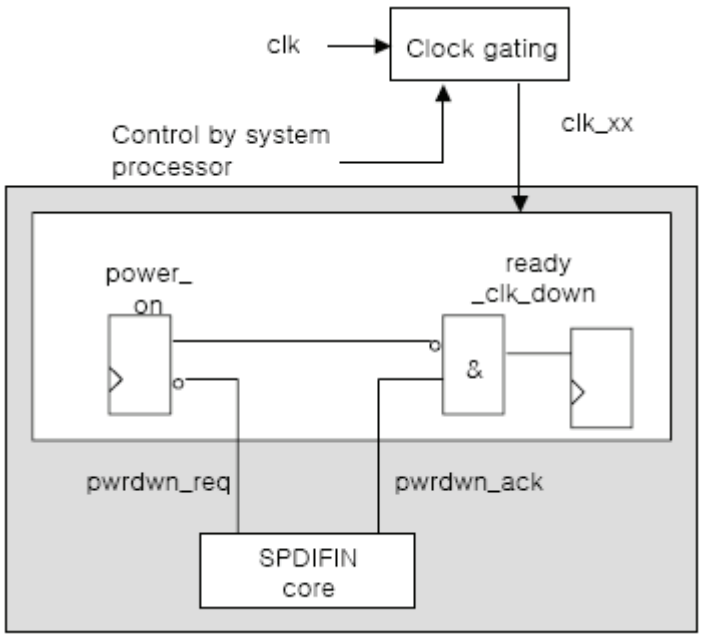
8.4.4.1 Control Registers

SPDIFIN_CLK_CTRL (SPDIFIN Clock Control Register) **0xB0A57000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														ready_clk_down	power_on

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1	ready_clk_down	R/W	1	0: clock is enabled 1: ready for disabling clock (default)
0	power_on	R/W	0	0: clock will be disabled (default) 1: clock will be activated If this bit is reset, SPDIFIN stops checking the input signal just before next 'subframe' of SPDIF signal format and wait the 'acknowledge' signal from HDMI for unresolved previous 'request' toward HDMI. Then asserts 'ready_clk_down' as HIGH. To initialize internal states, you have to assert S/W reset, i.e. SPDIFIN_OP_CTRL. op_ctrl=00b right after activating clock again.

The spdif_clk may be gated by an external clock gating module for low-power or etc. Disabling the clock must not cause stalling HDMI data transfer. Therefore the system processor requests disabling of the clock by setting the power_on register to low and the module acknowledge this request by setting the ready_clk_down register to high after a current transaction on the I2C bus and HDMI is finished. The module must not commence a new bus transaction until the system processor sets the power_on register to high again.



The system processor may only switch off clk_xx when the ready_clk_down bit is one and the power_on bit is zero.

The system processor can switch on the clock at any time. After having switched on clk_xx the system processor has to set the power_on bit to 1, which forces the ready_clk_down bit to zero.

After reset the clock of SPDIFIN is switched off, therefore the power_on bit is zero and the ready_clk_down bit is set to one.

SPDIFIN_OP_CTRL (SPDIFIN Operation Control Register 1) 0xB0A57004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														op_ctrl	

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1-0	op_ctrl	R/W	0	00b : software reset 01b : status checking mode (run) 11b : status checking + HDMI operation mode (run with HDMI) Others : undefined, do not use 00b : During a software reset, all state machines are set to the idle or init state and all internal registers are set to their initial values. Interrupt status registers are cleared, all other registers that are writable by the system processor keep their values. 01b : This command should be asserted after SPDIFIN_CLK_CTRL.power_on is set. SPDIFIN starts clock recovery. When recovery is done, SPDIFIN begins detecting preambles of SPDIF signal format and stream data header, abnormal time signal input, abnormal signal input, and also reports these status via interrupts in SPDIFIN_IRQ_STATUS. 11b : “01b” case operations + checking internal buffer overflow + write received data, which can be either audio sample word of PCM or payload of stream. Data will be transferred via HDMI. - You should assert ‘op_ctrl’=11b after SPDIFIN_IRQ_STATUS. ch_status_recovered_ir was asserted at least once for linear PCM data. Or you should assert ‘op_ctrl’=11b after SPDIFIN_IRQ_STATUS.stream_header_detected_ir was asserted at least once for non-linear PCM stream data.

SPDIFIN_IRQ_MASK (SPDIFIN Interrupt Request Mask Register) 0xB0A57008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								irq_mask							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	irq_mask[7]	R/W	0	buf_overflow_ir_en mask bit for Interrupt 7
6	irq_mask[6]	R/W	0	Reserved
5	irq_mask[5]	R/W	0	Reserved
4	irq_mask[4]	R/W	0	stream_header_detected_ir_en mask bit for Interrupt 4
3	irq_mask[3]	R/W	0	stream_header_not_detected_ir_en mask bit for Interrupt 3
2	irq_mask[2]	R/W	0	wrong_preamble_ir_en mask bit for Interrupt 2
1	irq_mask[1]	R/W	0	ch_status_recovered_ir_en mask bit for Interrupt 1
0	irq_mask[0]	R/W	0	wrong_signal_ir_en mask bit for Interrupt 0
For every bit: ‘0’: interrupt generation is disabled. ‘1’: interrupt generation is enabled.				

SPDIFIN_IRQ_STATUS (SPDIFIN Interrupt Request Status Register) **0xB0A5700C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								irq_status							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	Irq_status[7]	R/W	0	buf_overflow_ir 0: no interrupt 1: internal buffer overflow SPDIFIN internal buffer(s) (SPDIFIN_DATA_BUF_x) was overflowed because HDMI did not transfer the data in the buffer(s) to memory in time. - This interrupt will be asserted only if SPDIFIN_OP_CTRL.op_ctrl was set as “011”. - If user does not handle this interrupt, SPDIFIN will overwrite next subframe data to the internal data buffer (SPDIFIN_DATA_BUF_x) and continue data transfer via HDMI.
6	Irq_status[6]	R/W	0	Reserved
5	Irq_status[5]	R/W	0	Reserved
4	Irq_status[4]	R/W	0	stream_header_detected_ir 0: no interrupt 1: stream data header (Pa~Pd) detected - This interrupt will be asserted when SPDIFIN_OP_CTRL.op_ctrl=001b or 011b. - Cases for interrupt case1 : Initially after power_on case2 : Next stream header at right time when receiving stream data with SPDIFIN_CONFIG.data_type set as ‘stream mode’. case3: Initially detected stream header when receiving stream data with SPDIFIN_CONFIG.data_type set as ‘PCM mode’.
3	Irq_status[3]	R/W	0	stream_header_not_detected_ir 0: no interrupt 1: stream data header not detected for 4096 repetition time - This interrupt will be asserted when SPDIFIN_OP_CTRL.op_ctrl=001b or 011b. - Cases for interrupt case1 : Initially after power_on case2 : SPDIFIN was receiving stream but could not find next stream header for 4096 repetition time since previous stream header case3 : Could not find stream header for 4096 repetition time since previous reset of repetition time counter after previous interrupt of ‘stream_header_not_detected_ir’.
2	Irq_status[2]	R/W	0	wrong_preamble_ir 0: no interrupt 1: preamble was detected but there is a problem with detected time - This interrupt will be asserted when SPDIFIN_OP_CTRL.op_ctrl=001b or 011b. - Meaningless until ch_status_recovered_ir is asserted initially after SPDIFIN_OP_CTRL.op_ctrl=01b - Cases for interrupt case1: preamble was detected in the middle of a subframe audio sample word time case2: next preamble was not detected at exact time after a subframe duration

				case3: it was time for preamble B(or M or W) to be detected but other preamble was detected at that time
1	Irq_status[1]	R/W	0	ch_status_recovered_ir 0: no interrupt 1: recovered channel status Detected preamble of 2 consecutive B-preamble thus recovered 192 bit wide channel status. - Only supports consumer mode, so just 36bits will be reconstructed. If a user wants to see the channel status bits through SPDIFIN_CH_STATUS_x, you'd better read two consecutive 'ch_status_recovered_ir' and read that register each time; if these two channel status value are same, you can rely on that value.
0	Irq_status[1]	R/W	0	wrong_signal_ir 0: no interrupt 1: clock recovery fail Can not recover clock from input because of tolerable range violation(unlock) or because of no signal from outside or because of non-biphase in non-preamble duration - Meaningless until ch_status_recovered_ir is asserted initially after SPDIFIN_OP_CTRL.op_ctrl=01b
For every bit the following holds: Reading returns interrupt request status. Writing '0' has no effect. Writing '1' clears the interrupt request.				
1) Detection of stream header Wait for matching of Pa, Pb; 0xF872, 0x4E1F respectively Wait for the repetition time (From decoded Pc value or from user-set Pc in SPDIFIN_USER_VALUE.repetition_time_manual according to SPDIFIN_CONFIG. PcPd_value_mode) Check for matching of Pa, Pb on right time.				

SPDIFIN_CONFIG_1 (SPDIFIN Configuration Register 1) **0xB0A57010**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Abnormal_Pd_ir	0	PcPd_value_mode	Word_length_value_mode	U_V_CP_report	0	data_align

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6	abnormal_Pd_ir	R/W	0	0: filtering with 3 consecutive samples 1: filtering with 2 consecutive samples Noise filtering is done for over-sampled SPDIF input signal. This operation will be done as follows. If 'noise_filter_samples' is 0, 3 consecutive over-sampled signal will be regarded as a high or low only when those 3 samples are all high or low respectively. If 1 or 2 samples are low or high respectively for 3 over-sample duration, that noise filtered signal will keep previous value. If 'noise_filter_samples' is 1, 2 consecutive over-sampled signal will be regarded as a high or low only when those 2 samples are all high or low respectively. This setting can be used for reduced over-sampling ratio; recommended over-sampling ratio is 10 (see also 'clk_divisor')
5	-	R/W	0	Reserved (Must be '0')
4	PcPd_value_mode	R/W	0	0: automatically set 1: manually set If 0 for automatic setting, Pc, Pd values will be chosen by value of Pc, Pd from decoded stream header reported as in SPDIFIN_Px_INFO. If user sets this register, the receiver will use SPDIFIN_USER_VALUE[31:16],

				SPDIFIN_USER_VALUE[15:4] value as Pc, Pd respectively instead of decoded data from stream header as reported in SPDIFIN_Px_INFO. (cf) Burst payload length, whether it is automatically set or manually set, will affect the data size to be written in memory via HDMI by dumping the full sub-frame for the last bit for burst payload length. For example, if burst payload length is 257bit, i.e. (16subframe * 16bit + 1bit), then HDMI will write data in 17 consecutive sub-frames.
3	word_length_value_mode	R/W	0	0: automatically set 1: manually set If 0 for automatic setting, word length value will be chosen by value of channel status from decoded SPDIF format as reported in SPDIFIN_CH_STATUS_1. word_length. If user sets this register, the receiver will use SPDIFIN_USER_VALUE[3:0] value as word length instead of decoded data from channel status as reported in SPDIFIN_CH_STATUS_1.word_length.
2	U_V_C_P_report	R/W	0	0: neglects 'user_bit', 'validity_bit', 'channel status' 'parity_bit' of SPDIF format. 1: reports 'user_bit', 'validity_bit', 'channel status' 'parity_bit' of SPDIF format Report will be via HDMI for each subframe. Valid only if SPDIFIN_CONFIG. data_align is set for 32bit mode; see also SPDIFIN_DATA_BUF_x.
1	-	R/W	1	Reserved (Must be '1')
0	data_align	R/W	0	0: 16bit mode 1: 32bit mode 16bit: only takes 16bits from MSB in a subframe of SPDIF format then concatenates two consecutive 16bit data in one 32bit register of SPDIFIN_DATA_BUF_x. 32bit: data from one subframe with zero padding to MSB part. (ex: 0x00ffffff for 24bit data) When stream mode, you should set 'word_length_value_mode' as 1 and set SPDIFIN_USER_VALUE.word_length_manual as 0b000. - These two modes will be applied to both modes of SPDIFIN_CONFIG.data_type, i.e. PCM or stream; see also SPDIFIN_DATA_BUF_x.

SPDIFIN_CONFIG_2 (SPDIFIN Configuration Register 2) **0xB0A57014**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0												Clk_divisor			

Field	Name	RW	Reset	Description
31-4	-	R	0	Reserved
3-0	clk_divisor	R/W	2	SPDIFIN_internal_clock = system_clock / (clk_divisor + 1) (SPDIFIN_internal_clock ≤ 135 Mhz) SPDIFIN over-samples the SPDIF input signal with internally made clock which is divided from system clock. Recommended over-sampling ratio is 8~10, thus following calculation holds. Recommended SPDIFIN_internal_clock = Sampling Frequency of SPDIF Input Signal * 64 bits * 10 times-over-sampling (ex) 48 kHz * 64 bits * 10 times-over-sampling = 31 Mhz

SPDIFIN_USER_VALUE_1 (SPDIFIN User Value Register 1) 0xB0A57020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Repetition_time_manual_low				Word_length_manual			

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-4	repetition_time_manual_low	R/W	0	Repetition time[3:0] Repetition_time_manual register 12bits value. This register is low 4bits. Will be used for counting one block of stream data; valid only when SPDIFIN_CONFIG.PcPd_value_mode is set for manual mode. (Unit: frames (1 frame = 2 subframes) of SPDIF format) The value should be (actual repetition time – 1). For example, if you want to use 1536 as manually set repetition time, you should write 1535.
3-0	word_length_manual	R/W	0	Word length Will be used as size for transferring data to memory via HDMI; valid only when SPDIFIN_CONFIG.word_length_value_mode is set for manual mode; see also SPDIFIN_DATA_BUF_x. [0] is 1 [0] is 0 [3:1] 101: 24 bits 20 bits 001: 23 bits 19 bits 010: 22 bits 18 bits 011: 21 bits 17 bits 100: 20 bits 16 bits

SPDIFIN_USER_VALUE_2 (SPDIFIN User Value Register 2) 0xB0A57024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								repetition_time_manual_high							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	repetition_time_manual_high	R/W	0	Repetition time[11:4] Repetition_time_manual register 12bits value. This register is high 8bits. Will be used for counting one block of stream data; valid only when SPDIFIN_CONFIG.PcPd_value_mode is set for manual mode. (Unit: frames (1 frame = 2 subframes) of SPDIF format) The value should be (actual repetition time – 1). For example, if you want to use 1536 as manually set repetition time, you should write 1535.

SPDIFIN_USER_VALUE_3 (SPDIFIN User Value Register 3) 0xB0A57028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Burst_payload_length_manual_low							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	burst_payload_length_manual_low	R/W	0	Burst_payload_length_manual[7:0] Burst_payload_length register is 16bits value. This register is low 8bits Valid only when SPDIFIN_CONFIG.PcPd_value_mode is set for manual mode. (Unit: bits)

SPDIFIN_USER_VALUE_4 (SPDIFIN User Value Register 4) 0xB0A5702C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Burst_payload_length_manual_low							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	burst_payload_length_manual_high	R/W	0	Burst_payload_length_manual[15:8] Burst_payload_length register is 16bits value. this register is high 8bits Valid only when SPDIFIN_CONFIG.PcPd_value_mode is set for manual mode. (Unit: bits)

8.4.4.2 Channel Status Registers

SPDIFIN_CH_STATUS_0_1 (SPDIFIN Channel Status Register 0-1) **0xB0A57030**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Channel_status_mode		Emphasis			Copyright_assertion	Audio_sample_word	Channel_status_block

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-6	channel_status_mode	R	0	00: mode 0 others: reserved
5-3	emphasis	R	0	000: emphasis not indicated 100: emphasis – CD type
2	copyright_assertion	R	0	0: copyright 1: no copyright
1	audio_sample_word	R	0	0: linear PCM 1: non-linear PCM
0	channel_status_block	R	0	0: consumer format 1: professional format

This register will be updated every 192 frames(1 block) of SPDIF format.
SPDIFIN_CH_STATUS_0_1 [7:0] is matched internal register SPDIFIN_CH_STATUS_0 [7:0].

SPDIFIN_CH_STATUS_0_2 (SPDIFIN Channel Status Register 0-2) **0xB0A57034**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								category_code							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	category_code	R	0	equipment type : [8:15] CD player: 1000_0000 DAT player: 1100_000L DCC player: 1100_001L Mini disc: 1001_001L (L : information about generation status of the material)

SPDIFIN_CH_STATUS_0_3 (SPDIFIN Channel Status Register 0-3) **0xB0A57038**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								channel_nuymber				source_number			

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-4	channel_number	R	0	Channel Number (bit 20 is LSB)
3-0	source_number	R	0	Source Number (bit 16 is LSB)

SPDIFIN_CH_STATUS_0_4 (SPDIFIN Channel Status Register 0-4) 0xB0A57038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										clock_accuracy		sampling_frequency			

Field	Name	RW	Reset	Description
31-6	-	R	0	Reserved
5-4	clock_accuracy	R	0	Clock accuracy 00: level II, ±1000ppm 01: level I, ±50ppm 10: level III, variable pitch shifted
3-0	sampling_frequency	R	0	Sampling Frequency 0100: 22.05kHz 0000: 44.1 kHz 1000: 88.2kHz 1100: 176.4kHz 0110: 24kHz 0010: 48 kHz 1010: 96kHz 1110: 192kHz 0011: 32 kHz

SPDIFIN_CH_STATUS_1 (SPDIFIN Channel Status Register 1) 0xB0A57040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												word_length			field_size

Field	Name	RW	Reset	Description
31-4	-	R	0	Reserved
3-1	word_length	R	0	Word Length (field_size = 1) (field_size = 0) 000: not indicated not indicated 101: 24 bits 20 bits 100: 23 bits 19 bits 010: 22 bits 18 bits 110: 21 bits 17 bits 001: 20 bits 16 bits
0	field_size	R	0	Field Size 0: maximum length 20 bits 1: maximum length 24 bits

8.4.4.3 SPDIFIN Info Register

SPDIFIN_FRAME_PERIOD_1 (SPDIFIN Frame Period Register 1) **0xB0A57048**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								frame_cnt_low							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	frame_cnt_low	R	0	frame count value [7:0] Frame_cnt register is 16bits value. This is low 8bits. The period of a frame (2 sub-frames), thus this register will be updated every 2 sub-frames. It will be measured by ‘SPDIFIN_internal_clk’ made with SPDIFIN_CONFIG.clk_divisor. (Unit: SPDIF_internal_clk Cycles) (Recommended value for locking incoming signals : Over 0x220 (8.5timesx64bits))

SPDIFIN_FRAME_PERIOD_2 (SPDIFIN Frame Period Register 2) **0xB0A5704C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								frame_cnt_high							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	frame_cnt_high	R/W	0	frame count value [15:8] Frame_cnt register is 16bits value. This is high 8bits. The period of a frame (2 sub-frames), thus this register will be updated every 2 sub-frames. It will be measured by ‘SPDIFIN_internal_clk’ made with SPDIFIN_CONFIG.clk_divisor. (Unit: SPDIF_internal_clk Cycles) (Recommended value for locking incoming signals : Over 0x220 (8.5timesx64bits))

SPDIFIN_Pc_INFO_1 (SPDIFIN Pc Info Register 1) 0xB0A57050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								error_flag	0		compressed_data_type				

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	error_flag	R	0	0: valid burst payload 1: burst payload may contain errors
6-5	-	R	0	Reserved
4-0	compressed_data_type	R	0	0d: null data 1d: Dolby AC-3 2d: reserved 3d: pause 4d: MPEG-1 layer 1 5d: MPEG-1 layer 2 or 3 or MPEG-2 w/o extension 6d: MPEG-2 w/ extension 7d: reserved 8d: MPEG-2 layer 1 low sampling freq. 9d: MPEG-2 layer 2 or 3 low sampling freq. 10d: reserved 11d, 12d, 13d: DTS 14d~31d: reserved

SPDIFIN_Pc_INFO_2 (SPDIFIN Pc Info Register 2) 0xB0A57054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								bit_stream_number			data_type_dependent_info				

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-5	bit_stream_number	R	0	bit stream number.
4-0	data_type_dependent_info	R	0	data type dependent information.

SPDIFIN_Pd_INFO_1 (SPDIFIN Pd Info Register 1) 0xB0A57058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								burst_payload_length_low							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	burst_payload_length_low	R	0	length of burst payload [7:0] (Unit: bits)

SPDIFIN_Pd_INFO_2 (SPDIFIN Pd Info Register 2) 0xB0A5705C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								burst_payload_length_high							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	burst_payload_length_high	R	0	length of burst payload [15:8] (Unit: bits)

SPDIFIN_DATA_BUF0_1/2/3 (SPDIFIN Data Buffer Register 0_1/0_2/0_3) 0xB0A57060/64/68

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								received_data_0_1/2/3							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	received_data_0_1 received_data_0_2 received_data_0_3	R R R	0 0 0	PCM or stream data for 1st burst of HDMI SPDIFIN_DATA_BUF_0_1 = SPDIFIN_DATA_BUF_0[7:0] SPDIFIN_DATA_BUF_0_2 = SPDIFIN_DATA_BUF_0[15:8] SPDIFIN_DATA_BUF_0_3 = SPDIFIN_DATA_BUF_0[23:16] When SPDIFIN_CONFIG.data_align is 0 for 16bit received_data = {data_(N)th, data_(N+1)th} When SPDIFIN_CONFIG.data_align is 1 for 32bit received_data = {U, V, C, P, zero-padding, data[n:0]} received_data = {zero-padding, data[n:0]} (if SPDIFIN_CONFIG.U_V_P_report is 0) ('n' is dependent on SPDIFIN_CH_STATUS_1. word_length when SPDIFIN_CONFIG. data_type is 0 for PCM. Or 'n' is 15 when SPDIFIN_CONFIG.data_type is 1 for stream) If SPDIFIN_CONFIG.HDMI_burst_size is 1 burst, HDMI accesses only this data register.

SPDIFIN_USER_BUF_0 (SPDIFIN User Buffer Register 0) 0xB0A5706C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								received_data_user0				0			

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-4	received_data_user_0	R	0	User bit of 1st burst of HDMI received_data[7:4] = SPDIFIN_DATA_BUF_0[31:28]
3-0	-	R	0	Reserved

SPDIFIN_DATA_BUF1_1/2/3 (SPDIFIN Data Buffer Register 1_1/1_2/1_3) **0xB0A57070/74/78**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								received_data_1/1/2/3							

Field	Name	RW	Reset	Description
7-0	received_data_1_1 received_data_1_2 received_data_1_3	R R R	0 0 0	PCM or stream data for 2nd burst of HDMI SPDIFIN_DATA_BUF_0_1 = SPDIFIN_DATA_BUF_0[7:0] SPDIFIN_DATA_BUF_0_2 = SPDIFIN_DATA_BUF_0[15:8] SPDIFIN_DATA_BUF_0_3 = SPDIFIN_DATA_BUF_0[23:16] When SPDIFIN_CONFIG.data_align is 0 for 16bit received_data = {data_(N)th, data_(N+1)th} When SPDIFIN_CONFIG.data_align is 1 for 32bit received_data = {U, V, C, P, zero-padding, data[n:0]} received_data = {zero-padding, data[n:0]} (if SPDIFIN_CONFIG.U_V_P_report is 0) ('n' is dependent on SPDIFIN_CH_STATUS_1. word_length when SPDIFIN_CONFIG. data_type is 0 for PCM. Or 'n' is 15 when SPDIFIN_CONFIG.data_type is 1 for stream) If SPDIFIN_CONFIG.HDMI_burst_size is 1 burst, HDMI accesses only this data register.

SPDIFIN_USER_BUF_1 (SPDIFIN User Buffer Register 1) **0xB0A5707C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								received_data_user1				0			

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-4	received_data_user_1	R	0	User bit of 2nd burst of HDMI received_data[7:4] = SPDIFIN_DATA_BUF_1[31:28]
3-0	-	R	0	Reserved

8.4.5 I2S Registers

Name	Offset	Type	Description	Default
I2S_CLK_CON	0x0000	R/W	I2S Clock Enable Register	0x00
I2S_CON_1	0x0004	R/W	I2S Control Register 1	0x00
I2S_CON_2	0x0008	R/W	I2S Control Register 2	0x00
I2S_PIN_SEL_0	0x000C	R/W	I2S Input Pin Selection Register 0	0x77
I2S_PIN_SEL_1	0x0010	R/W	I2S Input Pin Selection Register 1	0x77
I2S_PIN_SEL_2	0x0014	R/W	I2S Input Pin Selection Register 2	0x77
I2S_PIN_SEL_3	0x0018	R/W	I2S Input Pin Selection Register 3	0x07
I2S_DSD_CON	0x001C	R/W	I2S DSD Control Register	0x02
I2S_MUX_CON	0x0020	R/W	I2S In/Mux Control Register	0x60
I2S_CH_ST_CON	0x0024	R/W	I2S Channel Status Control Register	0x00
I2S_CH_ST_0	0x0028	R/W	I2S Channel Status Block 0	0x00
I2S_CH_ST_1	0x002C	R/W	I2S Channel Status Block 1	0x00
I2S_CH_ST_2	0x0030	R/W	I2S Channel Status Block 2	0x00
I2S_CH_ST_3	0x0034	R/W	I2S Channel Status Block 3	0x00
I2S_CH_ST_4	0x0038	R/W	I2S Channel Status Block 4	0x00
I2S_CH_ST_SH_0	0x003C	R	I2S Channel Status Block Shadow Register 0	0x00
I2S_CH_ST_SH_1	0x0040	R	I2S Channel Status Block Shadow Register 1	0x00
I2S_CH_ST_SH_2	0x0044	R	I2S Channel Status Block Shadow Register 2	0x00
I2S_CH_ST_SH_3	0x0048	R	I2S Channel Status Block Shadow Register 3	0x00
I2S_CH_ST_SH_4	0x004C	R	I2S Channel Status Block Shadow Register 4	0x00
I2S_VD_DATA	0x0050	R/W	I2S Audio Sample Validity Register	0x00
I2S_MUX_CH	0x0054	R/W	I2S Channel Enable Register	0x03
I2S_MUX_CUV	0x0058	R/W	I2S CUV Enable Register	0x03
I2S_IRQ_MASK	0x005C	R/W	I2S Interrupt Request Mask Register	0x03
I2S_IRQ_STATUS	0x0060	R/W	I2S Interrupt Request Status Register	0x00
I2S_CH0_L_0	0x0064	R	I2S PCM Output Data Register	0x00
I2S_CH0_L_1	0x0068	R	I2S PCM Output Data Register	0x00
I2S_CH0_L_2	0x006C	R	I2S PCM Output Data Register	0x00
I2S_CH0_L_3	0x0070	R	I2S PCM Output Data Register	0x00
I2S_CH0_R_0	0x0074	R	I2S PCM Output Data Register	0x00
I2S_CH0_R_1	0x0078	R	I2S PCM Output Data Register	0x00
I2S_CH0_R_2	0x007C	R	I2S PCM Output Data Register	0x00
I2S_CH0_R_3	0x0080	R	I2S PCM Output Data Register	0x00
I2S_CH1_L_0	0x0084	R	I2S PCM Output Data Register	0x00
I2S_CH1_L_1	0x0088	R	I2S PCM Output Data Register	0x00
I2S_CH1_L_2	0x008C	R	I2S PCM Output Data Register	0x00
I2S_CH1_L_3	0x0090	R	I2S PCM Output Data Register	0x00

I2S_CH1_R_0	0x0094	R	I2S PCM Output Data Register	0x00
I2S_CH1_R_1	0x0098	R	I2S PCM Output Data Register	0x00
I2S_CH1_R_2	0x009C	R	I2S PCM Output Data Register	0x00
I2S_CH1_R_3	0x00A0	R	I2S PCM Output Data Register	0x00
I2S_CH2_L_0	0x00A4	R	I2S PCM Output Data Register	0x00
I2S_CH2_L_1	0x00A8	R	I2S PCM Output Data Register	0x00
I2S_CH2_L_2	0x00AC	R	I2S PCM Output Data Register	0x00
I2S_CH2_L_3	0x00B0	R	I2S PCM Output Data Register	0x00
I2S_CH2_R_0	0x00B4	R	I2S PCM Output Data Register	0x00
I2S_CH2_R_1	0x00B8	R	I2S PCM Output Data Register	0x00
I2S_CH2_R_2	0x00BC	R	I2S PCM Output Data Register	0x00
I2S_Ch2_R_3	0x00C0	R	I2S PCM Output Data Register	0x00
I2S_CH3_L_0	0x00C4	R	I2S PCM Output Data Register	0x00
I2S_CH3_L_1	0x00C8	R	I2S PCM Output Data Register	0x00
I2S_CH3_L_2	0x00CC	R	I2S PCM Output Data Register	0x00
I2S_CH3_R_0	0x00D0	R	I2S PCM Output Data Register	0x00
I2S_CH3_R_1	0x00D4	R	I2S PCM Output Data Register	0x00
I2S_CH3_R_2	0x00D8	R	I2S PCM Output Data Register	0x00
I2S_CUV_L_R	0x00DC	R	I2S CUV Output Data Register	0x00

8.4.5.1 Control Registers

I2S_CLK_CON (I2S Clock Enable Register) **0xB0A58000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0															i2s_en

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	i2s_en	R/W	0	I2S Clock Enable 0: i2s will be disabled (default) 1: i2s will be activated You must set i2s_en, after other registers are configured. when you want to reset the i2s, this register is 0□ 1.

I2S_CON_1 (I2S Control Register 1) **0xB0A58004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0														r_sc_pol	r_ch_pol

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1	r_sc_pol	R/W	0	SDATA is synchronous to 0 : SCLK falling edge 1 : SCLK rising edge
0	r_ch_pol	R/W	0	LRCLK polarity 0 : Left Channel for Low polarity 1 : Left Channel for High polarity

I2S_CON_2 (I2S Control Register 2) 0xB0A58008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									mlsb	bit_ch		data_num		i2s_mode	

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6	mlsb	R/W	0	0: MSB first mode 1: LSB first mode
5-4	bit_ch	R/W	0	Bit clock per Frame(Frame = left + right) 0b00: 32fs 0b01: 48fs 0b10: 64fs
3-2	data_num	R/W	0	Serial data bit per channel 0b01: 16bit 0b10: 20bit 0b11: 24bit
1-0	i2s_mode	R/W	0	0b00: I2S basic format 0b10: left justified format 0b11: right justified format

I2S_PIN_SEL_0 (I2S Input Pin Selection Register 0) 0xB0A5800C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									pin_sel_1			0	pin_sel_0		

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6-4	pin_sel_1	R/W	7	SCLK(I2S) & DSD_D0(DSD) selection 0b111 : i_i2s_in[1] 0b110 : i_i2s_in[6] 0b101 : i_i2s_in[5] 0b100 : i_i2s_in[4] 0b011 : i_i2s_in[3] 0b010 : i_i2s_in[2] 0b001 : i_i2s_in[1] 0b000 : i_i2s_in[0]
3	-	R/W	0	Reserved
2-0	pin_sel_0	R/W	7	LRCK(I2S) & DSD_CLK(DSD) selection 0b111 : i_i2s_in[0] 0b110 : i_i2s_in[6] 0b101 : i_i2s_in[5] 0b100 : i_i2s_in[4] 0b011 : i_i2s_in[3] 0b010 : i_i2s_in[2] 0b001 : i_i2s_in[1] 0b000 : i_i2s_in[0]

I2S_PIN_SEL_1 (I2S Input Pin Selection Register 1) 0xB0A58010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									pin_sel_3			0	pin_sel_2		

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6-4	pin_sel_3	R/W	7	SDATA_1(I2S) & DSD_D2(DSD) selection 0b111 : i_i2s_in[3] 0b110 : i_i2s_in[6] 0b101 : i_i2s_in[5] 0b100 : i_i2s_in[4] 0b011 : i_i2s_in[3] 0b010 : i_i2s_in[2] 0b001 : i_i2s_in[1] 0b000 : i_i2s_in[0]
3	-	R/W	0	Reserved
2-0	pin_sel_2	R/W	7	SDATA_0(I2S) & DSD_D1(DSD) selection 0b111 : i_i2s_in[2] 0b110 : i_i2s_in[6] 0b101 : i_i2s_in[5] 0b100 : i_i2s_in[4] 0b011 : i_i2s_in[3] 0b010 : i_i2s_in[2] 0b001 : i_i2s_in[1] 0b000 : i_i2s_in[0]

I2S_PIN_SEL_2 (I2S Input Pin Selection Register 2) 0xB0A58014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									pin_sel_5			0	pin_sel_4		

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6-4	pin_sel_5	R/W	7	SDATA_3(I2S) & DSD_D4(DSD) selection 0b111 : i_i2s_in[5] 0b110 : i_i2s_in[6] 0b101 : i_i2s_in[5] 0b100 : i_i2s_in[4] 0b011 : i_i2s_in[3] 0b010 : i_i2s_in[2] 0b001 : i_i2s_in[1] 0b000 : i_i2s_in[0]
3	-	R/W	0	Reserved
2-0	pin_sel_4	R/W	7	SDATA_2(I2S) & DSD_D3(DSD) selection 0b111 : i_i2s_in[4] 0b110 : i_i2s_in[6] 0b101 : i_i2s_in[5] 0b100 : i_i2s_in[4] 0b011 : i_i2s_in[3] 0b010 : i_i2s_in[2] 0b001 : i_i2s_in[1] 0b000 : i_i2s_in[0]

I2S_PIN_SEL_3 (I2S Input Pin Selection Register 3) 0xB0A58018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0													pin_sel_6		

Field	Name	RW	Reset	Description
31-3	-	R	0	Reserved
2-0	pin_sel_6	R/W	7	DSD_D5(DSD) selection 0b111 : i_i2s_in[6] 0b110 : i_i2s_in[6] 0b101 : i_i2s_in[5] 0b100 : i_i2s_in[4] 0b011 : i_i2s_in[3] 0b010 : i_i2s_in[2] 0b001 : i_i2s_in[1] 0b000 : i_i2s_in[0]

I2S_DSD_CON (I2S DSD Control Register) 0xB0A5801C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														r_dsd_pol	dsd_en

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1	r_dsd_pol	R/W	1	1 : DSD_DATA change at DSD_CLK rising edge 0 : DSD_DATA change at DSD_CLK falling edge
0	dsd_en	R/W	0	1 : DSD module enable 0 : DSD module disable

I2S_IN_MUX_CON (I2S In/Mux Control Register) **0xB0A58020**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								f_num			in_en	audio_sel		CUV_sel	mux_en

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-5	f_num	R/W	0	Number of stage of noise filter for I2S input pins 000: no filtering 001: 2 stage filter 010: 3 stage filter 011: 4 stage filter 100: 5 stage filter others : reserved
4	in_en	R/W	0	Enable i2s_in, a sub-module at the input stage. 0 : i2s_in module disable 1 : i2s_in module enable All output data is '0' if disabled.
3-2	audio_sel	R/W	1	Audio selection 0b00 : SPDIF audio data enable 0b01 : I2S audio data enable 0b10 : DSD audio data enable
1	CUV_sel	R/W	1	C.U.V. Selection 0 : SPDIF C.U.V. data enable 1 : I2S C.U.V. data enable
0	mux_en	R/W	0	Enable i2s_mux, a sub-module for audio selection. 0 : i2s_mux module disable 1 : i2s_mux module enable All output data is '0' if disabled.

8.4.5.2 Channel Status Register

I2S_CH_ST_CON (I2S Channel Status Control Register) **0xB0A58024**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															channel_status_reload

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	channel_status_reload	R/W	0	0: The shadow channel status registers are updated. 1: Set this bit to update the shadow channel status registers with the values updated in I2S_CH_ST_0 ~ I2S_CH_ST_4. When the shadow channel status registers are updated, this bit is cleared.

Channel status information needs to be applied to the audio stream at the IEC 60958 block boundary. For this synchronization, there are two register sets for channel status block. Users can set the channel status registers, I2S_CH_ST_0~I2S_CH_ST_4, while the I2S Rx module still refers to the shadow channel status registers, I2S_CH_ST_SH_0~ I2S_CH_ST_CH4. To reflect the user configuration in the channel status registers, users should set 'channel_status_reload' bit in I2S_CH_ST_CON then I2S Rx module copies the channel status registers into the shadow channel status registers at the beginning of an IEC-60958 block.

I2S_CH_ST_0, I2S_CH_ST_SH_0 (I2S Channel Status Block 0) 0xB0A58028/3C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								i2s_ch_st_0 / i2s_ch_st_sh_0							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-6	i2s_ch_st_0[7:6] i2s_ch_st_sh_0[7:6]	R/W, R	0 0	channel_status_mode 0b00 : Mode 0 others : Reserved
5-3	i2s_ch_st_0[5:3] i2s_ch_st_sh_0[5:3]	R/W, R	0 0	emphasis When bit1 = 0, 0b000 : 2 audio channels without pre-emphasis* 0b001 : 2 audio channels with 50us / 15us pre-emphasis When bit1 = 1, 0b000 : default state
2	i2s_ch_st_0[2] i2s_ch_st_sh_0[2]	R/W, R	0 0	copyright 0 : copyright 1 : no copyright
1	i2s_ch_st_0[1] i2s_ch_st_sh_0[1]	R/W, R	0 0	audio_sample_word 0 : linear PCM 1 : non-linear PCM
0	i2s_ch_st_0[0] i2s_ch_st_sh_0[0]	R/W, R	0 0	channel_status_block 0 : consumer format 1 : professional format

Note that bits listed here in Channel Status Registers look swapped from those in IEC-60958-3 Specification, as the bit order is different (LSB is right-most bit).

I2S_CH_ST_1, I2S_CH_ST_SH_1 (I2S Channel Status Block 1) 0xB0A5802C/40

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								category / category_sh							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	category category_sh	R/W, R	0 0	Equipment type CD player : 0000_0001 DAT player : L000_0011 DCC player : L100_0011 Mini disc : L100_1001 (L : information about generation status of the material)

I2S_CH_ST_2, I2S_CH_ST_SH_2 (I2S Channel Status Block 2) 0xB0A58030/44

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-4	channel_number channel_number_sh	R/W, R	0 0	Channel Number Note that bit4 is LSB.
3-0	source_number source_number_sh	R/W, R	0 0	Source Number Note that bit0 is LSB.

I2S_CH_ST_3, I2S_CH_ST_SH_3 (I2S Channel Status Block 3) 0xB0A58034/48

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										clock_accuracy / clock_accuracy_sh		sampling_frequency / sampling_frequency_sh			

Field	Name	RW	Reset	Description
31-5	-	R	0	Reserved
5-4	Clock_Accuracy Clock_Accuracy_sh	R/W, R	0 0	Clock Accuracy as specified in IEC-60958-3 0b01 : Level I, ±50 ppm 0b00 : Level II, ±1000 ppm 0b10 : Level III, variable pitch shifted
3-0	Sampling_Frequency Sampling_Frequency_sh	R/W, R	0 0	Sampling Frequency as specified in IEC-60958-3 0b0000 : 44.1 kHz 0b0010 : 48 kHz 0b0011 : 32 kHz 0b1010 : 96 kHz …

I2S_CH_ST_4, I2S_CH_ST_SH_4 (I2S Channel Status Block 4) 0xB0A58038/4C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								org_sampling_freq (shadow)				word_length (shadow)			max_word_length (shadow)

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-4	Org_Sampling_Freq Org_Sampling_Freq_sh	R/W, R	0 0	Original Sampling Frequency 0b1111 : 44.1Khz 0b0111 : 88.2Khz 0b1011 : 22.05Khz 0b0011 : 176.4Khz … For other frequencies, refer to original sampling frequency specified in IEC-60958-3
3-1	Word_Length Word_Length_sh	R/W, R	0 0	Word length Max. length 24bits 20 bits 0b000 : not defined not defined 0b001 : 20 bits 16bits 0b010 : 22 bits 18bits 0b100 : 23 bits 19bits 0b101 : 24 bits 20bits 0b110 : 21 bits 17bits
0	Max_Word_Length Max_Word_Length_sh	R/W, R	0 0	Maximum sample word length 1 : 24 bits 0 : 20 bits

I2S_VD_DATA (I2S Audio Sample Validity Register) **0xB0A58050**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															validity_flag

Field	Name	RW	Reset	Description
31-1	-	R	0	Reserved
0	validity_flag	R/W	0	Validity bit 0 : audio sample is reliable 1 : audio sample is unreliable

8.4.5.3 Mux Control Register

I2S_MUX_CH (I2S Channel Enable Register) **0xB0A58054**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								CH3_R_en	CH3_L_en	CH2_R_en	CH2_L_en	CH1_R_en	CH1_L_en	CH0_R_en	CH0_L_en

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	CH3_R_en	R/W	0	0 : Channel 3 right audio data output is disable 1 : Channel 3 right audio data output is enable
6	CH3_L_en	R/W	0	0 : Channel 3 left audio data output is disable 1 : Channel 3 left audio data output is enable
5	CH2_R_en	R/W	0	0 : Channel 2 right audio data output is disable 1 : Channel 2 right audio data output is enable
4	CH2_L_en	R/W	0	0 : Channel 2 left audio data output is disable 1 : Channel 2 left audio data output is enable
3	CH1_R_en	R/W	0	0 : Channel 1 right audio data output is disable 1 : Channel 1 right audio data output is enable
2	CH1_L_en	R/W	0	0 : Channel 1 left audio data output is disable 1 : Channel 1 left audio data output is enable
1	CH0_R_en	R/W	1	0 : Channel 0 right audio data output is disable 1 : Channel 0 right audio data output is enable
0	CH0_L_en	R/W	1	0 : Channel 0 left audio data output is disable 1 : Channel 0 left audio data output is enable

I2S_MUX_CUV (I2S CUV Enable Register) **0xB0A58058**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														CUV_R_en	CUV_L_en

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1	CUV_R_en	R/W	1	0 : Right channel CUV data is disable 1 : Right channel CUV data is enable
0	CUV_L_en	R/W	1	0 : Left channel CUV data is disable 1 : Left channel CUV data is enable

8.4.5.4 Interrupt Control Registers

I2S_IRQ_MASK (I2S Interrupt Request Mask Register) **0xB0A5805C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														int_2_ mask	int_1_ mask

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1	int_2_mask	R/W	1	Disable interrupt request by int_2 interrupt 0: int_2 interrupt is disabled 1: int_2 interrupt is enabled
0	int_1_mask	R/W	1	Disable interrupt request by int_1 interrupt 0: int_1 interrupt is disabled 1: int_1 interrupt is enabled

I2S_IRQ_STATUS (I2S Interrupt Request Status Register) **0xB0A58060**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0														int_2	int_1

Field	Name	RW	Reset	Description
31-2	-	R	0	Reserved
1	int_2	R/W	0	Interrupt status : Wrong register setting This interrupt is asserted when the I2S_CON_2.bit_ch is set to be 32fs while I2S_CON_2.data_num is set to either 20bit or 24bit. According to wrong register setting, Some audio data MSB bits removed. The audio data is not available.
0	int_1	R/W	0	Interrupt status : Bit Per Channel mismatch This interrupt is asserted when the number of SCLKs in one channel does not match with I2S_CON_2.bit_ch register value.
For all bits, the following holds; Reading returns interrupt request status. Writing '0' has no effect. Writing '1' clears the interrupt request.				

8.4.5.5 Output Buffer Registers

I2S_CHX_Y_Z (I2S PCM Output Data Register) **0xB0A58064~0xB0A580D8**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								I2S_CHX_Y_Z							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	I2S_CHX_Y_Z	R	0	PCM output data from I2S Rx module. X : Channel = 0, 1, 2 Y : Left/Right = L, R Z : Byte number I2S_CHX_Y_0 = PCMX_Y[7:0] I2S_CHX_Y_1 = PCMX_Y[15:8] I2S_CHX_Y_2 = PCMX_Y[23:16] I2S_CHX_Y_3 = PCMX_Y[27:24] Channel 3 has 24 bitwidth. I2S_CH3_Y_0 = PCM3_Y[7:0] I2S_CH3_Y_1 = PCM3_Y[15:8] I2S_CH3_Y_2 = PCM3_Y[23:16]

I2S_CUV_L_R (I2S CUV Output Data Register) **0xB0A580DC**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									CUV_R			0	CUV_L		

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6-4	CUV_R	R	0	VUCP data of Right channel CUV_R[3:0] = {valid bit, user bit, channel state bit, parity bit}
3	-	R	0	Reserved
2-0	CUV_L	R	0	VUCP data of Left channel CUV_L[3:0] = {valid bit, user bit, channel state bit, parity bit}

8.4.6 CEC Registers

Name	Offset	Type	Description	Default
CEC Configure/Status Registers				
CEC_TX_STATUS_0	0x0000	R	CEC Tx status register 0.	0x00
CEC_TX_STATUS_1	0x0004	R	CEC Tx status register 1. Number of blocks transferred.	0x00
CEC_RX_STATUS_0	0x0008	R	CEC Rx status register 0.	0x00
CEC_RX_STATUS_1	0x000C	R	CEC Rx status register 1. Number of blocks received.	0x00
CEC_INTR_MASK	0x0010	R/W	CEC interrupt mask register	0x00
CEC_INTR_CLEAR	0x0014	R/W	CEC interrupt clear register	0x00
CEC_LOGIC_ADDR	0x0020	R/W	HDMI Tx logical address register	0x0F
CEC_DIVISOR_0	0x0030	R/W	Clock divisor for 0.05ms period count ([7:0] of 32-bit)	0x00
CEC_ DIVISOR_1	0x0034	R/W	Clock divisor for 0.05ms period count ([15:8] of 32-bit)	0x00
CEC_DIVISOR_2	0x0038	R/W	Clock divisor for 0.05ms period count ([23:16] of 32-bit)	0x00
CEC_DIVISOR_3	0x003C	R/W	Clock divisor for 0.05ms period count ([31:24] of 32-bit)	0x00
CEC Tx related Registers				
CEC_TX_CTRL	0x0040	R/W	CEC Tx control register	0x10
CEC_TX_BYTE_NUM	0x0044	R/W	Number of blocks in a message to be transferred	0x00
CEC_TX_STATUS_2	0x0060	R	CEC Tx status register 2	0x00
CEC_TX_STATUS_3	0x0064	R	CEC Tx status register 3	0x00
CEC_TX_BUFFER_0 ~ CEC_TX_BUFFER_15	0x0080 ~ 0x00BC	R/W	Byte #0 ~ #15 of CEC message to be transferred. (#0 is transferred 1st)	0x00
CEC Rx related Registers				
CEC_RX_CTRL	0x00C0	R/W	CEC Rx control register	0x00
CEC_RX_STATUS_2	0x00E0	R	CEC Rx status register 2	0x00
CEC_RX_STATUS_3	0x00E4	R	CEC Rx status register 3eived 1st)	0x00
CEC_RX_BUFFER_0 ~ CEC_RX_BUFFER_15	0x0100 ~ 0x013C	R	Byte #0 ~ #15 of CEC message received (#0 is received 1st)	0x00

8.4.6.1 CEC Configure Registers

CEC_TX_STATUS_0 (CEC Tx status register 0.) **0xB0A59000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0															

Field	Name	RW	Reset	Description
31-4	-	R	0	Reserved
3	Tx_Error	R	0	CEC Tx_Error interrupt flag. This bit field also indicates the status of Tx_Error interrupt. This bit is valid only when Tx_Done bit is set. O : No error has occurred. 1 : An error has occurred during CEC Tx transfer. It will be cleared: - when set to 0 Tx_Enable bit of CEC_TX_CTRL register - when set Clear_Intr_Tx_Done or Clear_Intr_Tx_Error bit in CEC_INTR_CLEAR register
2	Tx_Done	R	0	CEC Tx_Done interrupt flag. This bit field also indicates the status of Tx_Done interrupt. O : Running or Idle 1 : CEC Tx transfer finished. It will be cleared: - when reset Tx_Enable bit of CEC_TX_CTRL_0 - when set Clear_Intr_Tx_Done or Clear_Intr_Tx_Error bit in CEC_INTR_CLEAR register
1	Tx_Transferring	R	0	If set RX-Running , this field is valid O : Tx is waiting for CEC Bus 1 : CEC Tx is transferring data via CEC Bus.
0	Tx_Running	R	0	O : Tx Idle 1 : CEC Tx is enabled and is either waiting for the CEC bus or transferring message.

CEC_TX_STATUS_1 (CEC Tx status register 1) **0xB0A59004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0								Tx_Bytes_Transferred							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	Tx_Bytes_Transferred	R	0	Number of blocks transferred (1 byte = 1 block in a CEC message). After sending CEC message, field will be updated. It will be cleared when set Clear_Intr_Tx_Done or Clear_Intr_Tx_Error bit in CEC_Intr_Clear register.

CEC_RX_STATUS_0 (CEC Rx status register 0.) 0xB0A59008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0											Rx_BCast	Rx_Error	Rx_Done	Rx_Receiving	Rx_Running

Field	Name	RW	Reset	Description
31-5	-	R	0	Reserved
4	Rx_BCast	R	0	Broadcast message flag O : Received CEC message is address to a single device. 1 : Received CEC message is a broadcast message. It will be cleared: - when reset Rx_Enable bit of CEC_RX_CTRL_0 - when set Clear_Intr_Rx_Done or Clear_Intr_Rx_Error bit in CEC_INTR_CLEAR register
3	Rx_Error	R	0	CEC Rx_Error interrupt flag. This bit field also indicates the status of Rx_Error interrupt. This bit is valid only when Rx_Done bit is set. O : No error has occurred. 1 : An error has occurred in receiving a CEC message It will be cleared: - when reset Rx_Enable bit of CEC_RX_CTRL_0 - when set Clear_Intr_Rx_Done or Clear_Intr_Rx_Error bit in CEC_INTR_CLEAR register
2	Rx_Done	R	0	CEC Rx done interrupt Flag. This bit field also indicates the status of Rx_Done interrupt. O : Running or Idle 1 : CEC Rx transfer finished It will be cleared: - when reset Rx_Enable bit of CEC_RX_CTRL_0 - when set Clear_Intr_Rx_Done or Clear_Intr_Rx_Error bit in CEC_INTR_CLEAR register
1	Rx_Receiving	R	0	O : Rx is waiting for a CEC message. 1 : Rx is currently receiving data via CEC Bus.
0	Rx_Running	R	0	O : Rx disabled 1 : CEC Rx is enabled and is either waiting for a message on the CEC bus.

CEC_RX_STATUS_1 (CEC Rx status register 1) 0xB0A5900C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Rx_Bytes_Received							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	Rx_Bytes_Received	R	0	Number of blocks received (1 byte = 1 block in a CEC message). After receiving CEC message, field will be updated. It will be cleared when set Clear_Intr_Rx_Done or Clear_Intr_Rx_Error bit in CEC_Intr_Clear register.

CEC_INTR_MASK (CEC interrupt mask register) 0xB0A59010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										Mask_Intr_Rx_Error	Mask_Intr_Rx_Done	0		Mask_Intr_Tx_Erro	Mask_Intr_Tx_Done

Field	Name	RW	Reset	Description
31-6	-	R	0	Reserved
5	Mask_Intr_Rx_Error	R/W	0	Rx_Error interrupt mask bit. 0 : Enabled 1 : Disabled.
4	Mask_Intr_Rx_Done	R/W	0	Rx_Done interrupt mask bit. 0 : Enabled 1 : Disabled.
3-2	-	R/W	0	Reserved
1	Mask_Intr_Tx_Error	R/W	0	Tx_Error interrupt mask bit. 0 : Enabled 1 : Disabled.
0	Mask_Intr_Tx_Done	R/W	0	Tx_Done interrupt mask bit. 0 : Enabled 1 : Disabled.

CEC_INTR_CLEAR (CEC interrupt clear register) 0x B0A59014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										Clear_Intr_Rx_Error	Clear_Intr_Rx_Done	0		Clear_Intr_Tx_Error	Clear_Intr_Tx_Done

Field	Name	RW	Reset	Description
31-6	-	R	0	Reserved
5	Clear_Intr_Rx_Error	R/W	0	Rx_Error interrupt clear bit. 0 : No effect 1 : Clear Rx_Error and Rx_Bytes_Received fields in CEC_RX_STATUS_0 and 1 register. It will be cleared after one clock.
4	Clear_Intr_Rx_Done	R/W	0	Rx_Done interrupt clear bit. 0 : No effect 1 : Clears Rx_Done and Rx_Bytes_Received filds in CEC_RX_STATUS_0 and 1 register. Resets to 0 after one clock.
3-2	-	R/W	0	Reserved
1	Clear_Intr_Tx_Error	R/W	0	Tx_Error interrupt clear bit. 0 : No effect 1 : Clears Tx_Error and Tx_Bytes_Received filds in CEC_TX_STATUS_0 and 1 register. Resets to 0 after one clock.
0	Clear_Intr_Tx_Done	R/W	0	Tx_Done interrupt clear bit. 0 : No effect 1 : Clears Tx_Done and Tx_Bytes_Received filds in CEC_TX_STATUS_0 and 1 register. Resets to 0 after one clock.

CEC_LOGIC_ADDR (HDMI Tx logical address register) **0xB0A59020**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												Logic_Addr			

Field	Name	RW	Reset	Description
31-4	-	R	0	Reserved
3-0	Logic_Addr	R/W	0xF	HDMI Tx logical address (0~15)

CEC_ DIVISOR_0 ~ CEC_ DIVISOR_3 (Clock divisor for 0.05ms period count) **0xB0A59030**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								CEC_Divisor[7:0] … CEC_Divisor[31:24]							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	CEC_Divisor[7:0] … CEC_Divisor[31:24]	R/W	0	A divisor used in counting 0.05ms period. This divisor should satisfy the following equation: (CEC_DIVISOR) x (clock cycle time(ns)) = 0.05ms

8.4.6.2 Tx Related Registers

CEC_TX_CTRL (CEC Tx control register) **0xB0A59040**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Reset	Tx_Retrans_Num			0		Tx_BCast	Tx_Start

Field	Name	RW	Reset	Description
31-8	-	0	R	Reserved
7	Reset	R/W	0	CEC Tx reset bit. 0 : No effect 1 : Immediately resets CEC Tx related registers and state machines to its reset value. Resets to 0 after one clock.
6-4	Tx_Retrans_Num	R/W	1	Number of retransmissions tried when situations in CEC spec. page CEC-13 occurs. According to the specification, it should be set to 5.
3-2	-	R/W	0	Reserved
1	Tx_BCast	R/W	0	CEC Tx broadcast message bit. This bit indicates whether a CEC message in CEC_TX_BUFFER_00~15 is directly-addressed (addressed to a single device) or broadcast. This bit has effect on determining whether a block transfer is acknowledged or not. (following ACK scheme in CEC Spec.(section CEC 6.1.2)) 0 : Directly-addressed message 1 : Broadcast message.
0	Tx_Start	R/W	0	CEC Tx start bit. 0 : Tx idle. 1 : Start CEC message transfer (Resets to 0 after start)

CEC_TX_BYTE_NUM (Number of blocks in a message to be transferred) **0xB0A59044**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Tx_Byte_Num							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	Tx_Byte_Num	R/W	0	Number of blocks in a message to be sent. (1 byte = 1 block in a CEC message).

CEC_TX_STATUS_2 (CEC Tx status register 2) **0xB0A59060**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Tx_Byte_Num							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	Tx_Byte_Num	R/W	0	Number of blocks in a message to be sent. (1 byte = 1 block in a CEC message).

CEC_TX_STATUS_3 (CEC Tx status register 3) 0xB0A59064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Tx_Wait_SFT_Succ	Tx_Wait_SFT_New	Tx_Wait_SFT_Retran	Tx_Retrans_Cnt			Tx_ACK_Failed

Field	Name	RW	Reset	Description
31-7	Reserved	R	0	Reserved
6	Tx_Wait_SFT_Succ	R	0	CEC Tx signal free time for successive message transfer waiting flag bit 0 : Tx is in other state 1 : Tx is waiting for a signal free time (SFT) with a precondition that Tx is the most recent initiator on the CEC bus and wants to send another frame immediately after its previous frame. (SFT ≥ 7x2.4ms)
5	Tx_Wait_SFT_New	R	0	CEC Tx signal free time for a new initiator waiting flag bit 0 : Tx is in other state 1 : Tx is waiting for a signal free time (SFT) with a precondition that Tx is a new initiator and wants to send a frame.(SFT ≥ 5x2.4ms)
4	Tx_Wait_SFT_Retran	R	0	CEC Tx signal free time for a new initiator waiting flag bit 0 : Tx is in other state 1 : Tx is waiting for a signal free time (SFT) with a precondition that Tx is attempting a retransmission of the message. (SFT ≥ 3 x2.4ms)
3-1	Tx_Retrans_Cnt	R	0	It indicates current retransmissions count. If 0, no retransmission occurred. It will be cleared when set Clear_Intr_Tx_Done or Clear_Intr_Tx_Error bit in CEC_Intr_Clear register.
0	Tx_ACK_Failed	R	0	CEC Tx acknowledge failed flag bit 0 : Tx is in other state 1 : Tx is not acknowledged. This bit is set when - ACK bit in a block is logical 1 in a directly-addressed message - ACK bit in a block is logical 0 in a broadcast message

CEC_TX_BUFFER_0 ~ CEC_ TX_BUFFER_15 0xB0A59080~0xB0A590BC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Tx_Block_0 ~ Tx_Block_15							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	Tx_Block_0 ~ Tx_Block_15	R/W	0	Byte #0 ~ #15 of CEC message. Each byte corresponds to a block in a message. Block_0 is the header block and Block_1 ~15 are data blocks. Note that initiator and destination logical address in a header block should written by S/W.

8.4.6.3 Rx Related Registers

CEC_ RX_CTRL (CEC Rx control register) **0xB0A590C0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Reset	Check_Sampling_Error	Check_Low_Time_Error	Check_Start_Bit_Error	0		Rx_Host_Busy	Rx_Enable

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	Reset	R/W	0	CEC Rx reset bit. 0 : No effect 1 : Immediately resets CEC Rx related registers and state machines to its reset value. It will be cleared after one clock.
6	Check_Sampling_Error	R/W	0	CEC Rx sampling error check enable bit 0 : Do not check sampling error. 1 : Check sampling error while receiving data bits. CEC Rx samples the CEC bus three times (at 1.00, 1.05, 1.10 ms) and checks whether three samples are identical.
5	Check_Low_Time_Error	R/W	0	CEC Rx low-time error check enable bit 0 : Do not check low-time error. 1 : Check low-time error while receiving data bits. In receiving each bit from the CEC bus, CEC Rx checks the duration of logical 0 from the starting of one bit transfer (falling edge on the CEC bus). Rx checks whether the duration is longer than the maximum time the CEC bus can be in logical 0 (max 1.7 ms).
4	Check_Start_Bit_Error	R/W	0	CEC Rx start bit error check enable bit 0 : Do not check start bit error. 1 : Check start bit error while receiving a start bit. In receiving a start bit from the CEC bus, CEC Rx checks the duration of logical 0 and 1 of a starting bit as specified in CEC spec. page CEC-8. Rx checks whether the duration meets the specification.
3-2	-	R/W	0	Reserved
1	Rx_Host_Busy	R/W	0	CEC Rx host busy bit 0 : Rx receives incoming message and send acknowledges. 1 : A host processor is unavailable to receive and process CEC messages. Rx sends not acknowledged signal to a message initiator to indicate that a host processor is unavailable to receive and process CEC messages.
0	Rx_Enable	R/W	0	CEC Rx start bit. 0 : Rx disabled. 1 : Enable CEC Rx module to receive a message. This bit is cleared after receiving a message

CEC_ RX_STATUS_2 (CEC Rx status register 2) 0xB0A590E0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Rx_Waiting	Rx_Receiving_Start_Bit	Rx_Receiving_Hdr_Blk	Rx_Receiving_Data_Blk	0			

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7	Rx_Waiting	R	0	CEC Rx waiting flag bit 0 : Rx is in other state 1 : CEC Rx is waiting for a message.
6	Rx_Receiving_Start_Bit	R	0	CEC Rx start bit receiving flag bit 0 : Rx is in other state 1 : CEC Rx is currently receiving a start bit.
5	Rx_Receiving_Hdr_Blk	R	0	CEC Rx header block receiving flag bit 0 : Rx is in other state 1 : CEC Rx is currently receiving a header block.
4	Rx_Receiving_Data_Blk	R	0	CEC Rx data block receiving flag bit 0 : Rx is in other state 1 : CEC Rx is currently receiving data blocks.
3-0	-	R	0	Reserved

CEC_ RX_STATUS_3 (CEC Rx status register 3eived 1st) **0xB0A590E4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0									Sampling_Error	Low_Time_Error	Start_Bit_Error	0			CEC_Line_Error

Field	Name	RW	Reset	Description
31-7	-	R	0	Reserved
6	Sampling_Error	R	0	CEC Rx sampling error flag bit 0 : No sampling error has occurred. 1 : A sampling error has occurred in receiving a message. CEC Rx samples the CEC bus three times (at 1.00, 1.05, 1.10 ms) and sets this bit if - Check_Sampling_Error bit in CEC_RX_CTRL_0 is set and - Three samples are not identical. It will be cleared when set to 0 when Clear_Intr_Rx_Done or Clear_Intr_Rx_Error bit in CEC_INTR_CLEAR register.
5	Low_Time_Error	R	0	CEC Rx low-time error flag bit 0 : No low-time error has occurred. 1 : A low-time error has occurred in receiving a message. In receiving each bit from the CEC bus, CEC Rx checks the duration of logical 0 from the starting of one-bit transfer falling edge on the CEC bus). If the duration of longer than the maximum time the CEC bus can be in logical 0 (max 1.7 ms), CEC RX sets this bit. This bit field will be set to 0 when Clear_Intr_Rx_Done or Clear_Intr_Rx_Error bit in CEC_INTR_CLEAR register is set.
4	Start_Bit_Error	R	0	CEC Rx start bit error flag bit 0 : No start bit error has occurred. 1 : A start bit error has occurred in receiving a message. In receiving a start bit from the CEC bus, CEC Rx checks the duration of logical 0 and 1 of a starting bit as specified in CEC spec. page CEC-8. If the duration does not meet the spec., CEC RX sets this bit. This bit field will be set to 0 when Clear_Intr_Rx_Done or Clear_Intr_Rx_Error bit in CEC_INTR_CLEAR register is set.
3-1	-	R	0	Reserved
0	CEC_Line_Error	R	0	CEC Rx line error flag bit 0 : No line error has occurred. 1 : A start bit error line error has occurred in receiving a message. In CEC spec. page CEC-13, CEC line error is defined as a situation that period between two consecutive falling edge is smaller than a minimum data bit period. Rx check for this condition and if it occurs, sends line error notification, i.e. sending logical 0 for more than 1.4~1.6 times of the nominal data bit period (2.4ms). This bit will be cleared: When set Rx_Enable bit of CEC_RX_CTRL_0 When set Clear_Intr_Rx_Done or Clear_Intr_Rx_Error bit in CEC_INTR_CLEAR register.

CEC_RX_BUFFER_0 ~ CEC_RX_BUFFER_15 **0xB0A59100~0xB0A5913C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0								Rx_Block_0 ~ Rx_Block_15							

Field	Name	RW	Reset	Description
31-8	-	R	0	Reserved
7-0	Rx_Block_0 ~ Rx_Block_15	R/W	0	Byte #0 ~ #15 of CEC message. Each byte corresponds to a block in a message. Block_0 is the header block and Block_1 ~15 are data blocks.

8.5 HDMI PHY

8.5.1 Clock Scheme of the HDMI TX PHY Core

The HDMI TX PHY has an internal video PLL and can generate a pixel clock. LCD controller uses the pixel clock to deliver its video data into HDMI LINK layer. HDMI Link layer transfers the TMDS input data (TXDn_E/O[0:9], n=0,1,2) into PHY layer with TMDS clock, which is also generated in TX PHY. For the sake of enough timing margin between LINK and PHY, the TMDS clock into PHY (TMDS_CLKHI) has a half frequency of the TMDS clock (TMDS_CLKO). 20bit interface per data channel is used to compensate the halved frequency. A timing skew between TMDS_CLKHI and TMDS_CLKO due to dividing and clock buffering is treated in de-skewing block within PHY.

One of the advantages of integrated video PLL is good pixel clock jitter.
Figure 8.5 shows the clock scheme of HDMI TX PHY core using the internally generated pixel clock.

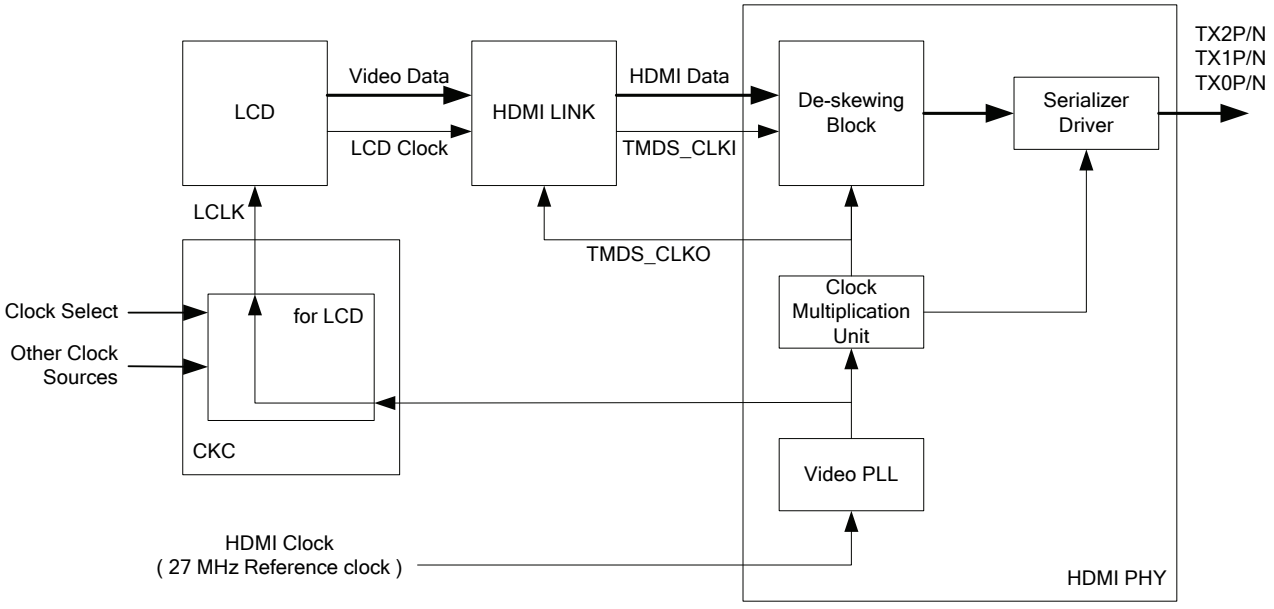


Figure 8.5 HDMI Clock Scheme Using the Integrated Video PLL for Pixel Clock Generation

8.5.2 PHY Configuration Change through I2C

HDMI PHY has many internal registers to change its configuration, like pixel clock frequency or analog characteristics. These registers can be access through I2C. For secure configuration of the PHY core, MODE_SET_DONE register is used for an indicator of I2C setting state as shown in Figure 8.6 .

(MODE_SET_DONE register is also controlled by I2C.)

To reconfigure PHY by new register setting, MODE_SET_DONE register should be set “0x00” instead of asserting overall RESET signal. Then PHY_READY signal goes to “0” state and PHY waits for new register setting. After new values are written on PHY registers, MODE_SET_DONE register should be set “0x80” again letting PHY start to configure its state with new register values. Once configuration is done, PHY_READY signal is automatically asserted.

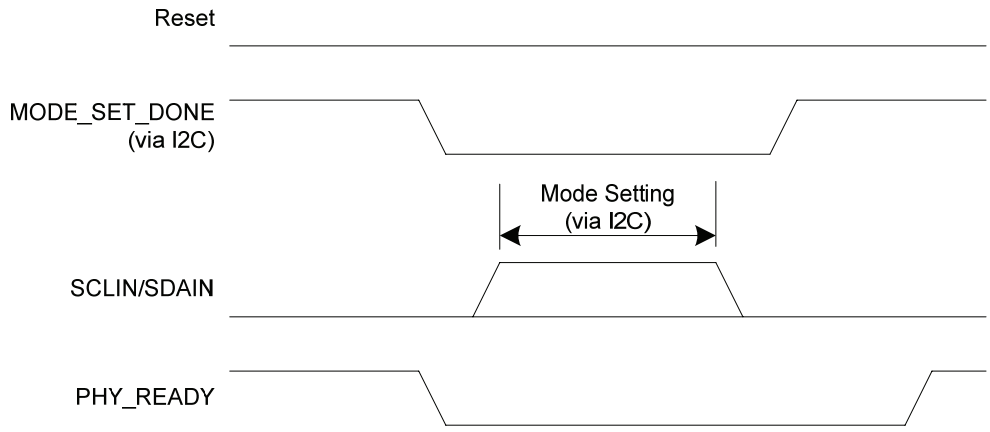


Figure 8.6 PHY Configuration through I2C with MODE_SET_DONE Register

8.5.3 PHY Reset Timing

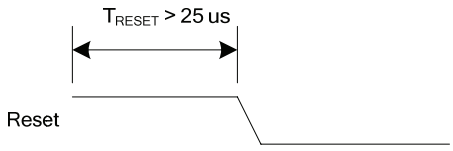


Figure 8.7 HDMI PHY Reset Timing

8.5.4 Register Map

Table 8.1 PHY Register Map (I2C Address = 0x70)

Name	Address	Description
BIAS	0x01	Bias Generation Selection
	0x02	Reserved
Video PLL	0x03	Pre-divider setting
	0x04	Pre-divider setting & Main divider setting
	0x05	Main divider setting
	0x06	Post-divider setting
	0x07	Post-divider setting
	0x08	Pixel clock divider setting
	0x09	PLL loop parameter setting
	0x0A	PLL loop parameter setting
	0x0B	PLL loop parameter setting
	0x0C	PLL loop parameter setting
	0x0D	Control code for sigma-delta modulator
	0x0E	Control code for sigma-delta modulator
	0x0F	Control code for automatic frequency calibration
	0x10	Control code for automatic frequency calibration
	0x11	Control code for automatic frequency calibration
Power Down	0x12	Power-down control
TX	0x13	TMDS data amplitude and pre-emphasis control
	0x14	TX source termination
Loop-back Test	0x15	Loop-back Test
CMU	0x16	Pre-divider & post-divider setting
	0x17	Main divider setting
	0x18	Control code for automatic frequency calibration
	0x19	Control code for automatic frequency calibration
	0x1A	Control code for automatic frequency calibration
	0x1B	Control code for automatic frequency calibration
	0x1C	PLL loop parameter setting
	0x1D	PLL loop parameter setting
	0x1E	PLL loop parameter setting
	0x1F	PLL loop parameter setting
	0x20	PLL loop parameter setting
RESET	0x21	RESET
Logic	0x22	PHY Logic
	0x23	PHY Logic
	0x24	PHY Logic
	0x25	PHY Logic
	0x26	PHY Logic
	0x27	PHY Logic
	0x28	MODE_SET_DONE

8.5.5 Recommended Register Setting Value

Using the Integrated Video PLL for Pixel Clock Generation

	25.200	25.175	27.000	27.027	54.000	54.054	74.250	74.176
0x01	05h	05h	05h	05h	05h	05h	05h	05h
0x02	00h	00h	00h	00h	00h	00h	00h	00h
0x03	D8h	D8h	D8h	D8h	D8h	D8h	D8h	D8h
0x04	10h	10h	10h	10h	10h	10h	10h	10h
0x05	1Ch	9Ch	1Ch	9Ch	1Ch	1Ch	1Ch	9Ch
0x06	30h	FCh	30h	02h	30h	30h	30h	56h
0x07	40h	48h	40h	32h	40h	40h	40h	5Bh
0x08	6Bh	6Bh	6Bh	6Bh	6Bh	6Bh	6Bh	6Bh
0x09	50h	50h	10h	10h	10h	10h	10h	10h
0x0A	11h	11h	02h	02h	01h	01h	01h	01h
0x0B	54h	54h	54h	54h	54h	54h	54h	54h
0x0C	58h	58h	38h	38h	38h	38h	98h	18h
0x0D	20h	20h	20h	20h	20h	20h	21h	25h
0x0E	5Bh	5Bh	5Bh	5Bh	5Bh	5Bh	5Bh	5Bh
0x0F	A8h	A8h	78h	78h	78h	78h	A5h	A5h
0x10	84h	84h	84h	84h	84h	84h	84h	84h
0x11	00h	00h	00h	00h	00h	00h	00h	00h
0x12	00h	00h	00h	00h	00h	00h	20h	00h
0x13	38h	38h	38h	38h	38h	38h	38h	38h
0x14	00h	00h	00h	00h	00h	00h	00h	00h
0x15	08h	08h	08h	08h	08h	08h	08h	08h
0x16	10h	10h	10h	10h	10h	10h	10h	10h
0x17	E0h	E0h	E0h	E0h	E0h	E0h	E0h	E0h
0x18	22h	22h	22h	22h	22h	22h	22h	22h
0x19	40h	40h	40h	40h	40h	40h	40h	40h
0x1A	11h	12h	FFh	FFh	FFh	FFh	B9h	B9h
0x1B	27h	27h	26h	26h	26h	26h	26h	26h
0x1C	00h	00h	00h	00h	01h	01h	01h	01h
0x1D	00h	00h	00h	00h	00h	00h	00h	00h
0x1E	00h	00h	00h	00h	00h	00h	00h	00h
0x1F	80h	80h	80h	80h	80h	80h	80h	80h

	148.500	148.352	108.108	72.000	25.000	65.000	108.000	162.000
0x01	05h	05h	05h	05h	05h	05h	05h	05h
0x02	00h	00h	00h	00h	00h	00h	00h	00h
0x03	D8h	D8h	D8h	D8h	D8h	D8h	D8h	D8h
0x04	10h	10h	10h	10h	10h	10h	10h	10h
0x05	1Ch	9Ch	9Ch	1Ch	9Ch	9Ch	1Ch	1Ch
0x06	30h	56h	02h	30h	E8h	08h	30h	30h
0x07	40h	5Bh	32h	40h	36h	36h	40h	40h
0x08	6Bh	6Bh	6Bh	6Bh	6Bh	6Bh	6Bh	6Bh
0x09	18h	18h	18h	18h	50h	10h	18h	18h
0x0A	01h	01h	01h	01h	11h	01h	01h	01h
0x0B	54h	54h	54h	54h	54h	54h	54h	54h
0x0C	98h	18h	38h	68h	58h	48h	38h	48h
0x0D	21h	25h	20h	21h	20h	20h	20h	21h
0x0E	5Bh	5Bh	5Bh	5Bh	5Bh	5Bh	5Bh	5Bh
0x0F	A5h	A5h	78h	50h	A7h	90h	78h	B4h
0x10	84h	84h	84h	84h	84h	84h	84h	84h
0x11	00h	00h	00h	00h	00h	00h	00h	00h
0x12	20h	00h	00h	21h	00h	00h	00h	00h
0x13	38h	38h	38h	38h	38h	38h	38h	38h
0x14	00h	00h	00h	00h	00h	00h	00h	00h
0x15	08h	08h	08h	08h	08h	08h	08h	08h
0x16	10h	10h	10h	10h	10h	10h	10h	10h
0x17	E0h	E0h	E0h	E0h	E0h	E0h	E0h	E0h
0x18	22h	22h	22h	22h	22h	22h	22h	22h
0x19	40h	40h	40h	40h	40h	40h	40h	40h
0x1A	B9h	B9h	FFh	BFh	13h	D4h	FFh	AAh
0x1B	26h	26h	26h	26h	27h	26h	26h	26h
0x1C	02h	02h	02h	01h	00h	01h	02h	02h
0x1D	00h	00h	00h	00h	00h	00h	00h	00h
0x1E	00h	00h	00h	00h	00h	00h	00h	00h
0x1F	80h	80h	80h	80h	80h	80h	80h	80h

8.6 Examples - HDMI Register Setting

When HDMI is used, an outputted clock from HDMI PHY is recommended to be used for LCD Clock. All the example values below take HDMI PHY PLL. The below values are recommended for LCD setting value, HDMI PHY, HDMI Link according to video output format.

8.6.1 1920x1080p

1920x1080p @ 60Hz								
Pixel Clock		148.5	MHz					
H Frequency		67.5	KHz					
V Frequency		60	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	280	LCTRL	IV	0
0x01	05h	0x11	00h	V2_BLANK	1125		IH	0
0x02	00h	0x12	20h	V1_BLANK	45		IP	0
0x03	D8h	0x13	38h	V_LINE	1125		DP	0
0x04	10h	0x14	00h	H_LINE	2200		NI	1
0x05	1Ch	0x15	08h	VSYNC_POL	0		TV	0
0x06	30h	0x16	10h	INT_PRO_MODE	0		TFT	1
0x07	40h	0x17	E0h	V_BOT_ST	-		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	-	LHTIME1	LPW	43
0x09	18h	0x19	40h	HSYNC_START	86		LPC	1919
0x0A	01h	0x1A	B9h	HSYNC_END	130	LHTIME2	LSWC	147
0x0B	54h	0x1B	26h	HSYNC_POL	0		LEWC	87
0x0C	98h	0x1C	02h	VSYNC_T_END	9	LVTIME1	VDB	0
0x0D	21h	0x1D	00h	VSYNC_T_ST	4		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	-		FPW	4
0x0F	A5h	0x1F	80h	VSYNC_B_ST	-		FLC	1079
0x10	84h			VSYNC_H_POS_ST	-	LVTIME2	FSWC	35
				VSYNC_H_POS_END	-		FEWC	3
						LVTIME3	FPW2	4
							FLC2	1079
						LVTIME4	FSWC2	35

1920x1080p @ 59.94Hz								
Pixel Clock		148.35	MHz					
H Frequency		67.432	KHz					
V Frequency		59.939	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	280	LCTRL	IV	0
0x01	05h	0x11	00h	V2_BLANK	1125		IH	0
0x02	00h	0x12	00h	V1_BLANK	45		IP	0
0x03	D8h	0x13	38h	V_LINE	1125		DP	0
0x04	10h	0x14	00h	H_LINE	2200		NI	1
0x05	9Ch	0x15	08h	VSYNC_POL	0		TV	0
0x06	56h	0x16	10h	INT_PRO_MODE	0		TFT	1
0x07	5Bh	0x17	E0h	V_BOT_ST	-		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	-	LHTIME1	LPW	43
0x09	18h	0x19	40h	HSYNC_START	86		LPC	1919
0x0A	01h	0x1A	B9h	HSYNC_END	130	LHTIME2	LSWC	147
0x0B	54h	0x1B	26h	HSYNC_POL	0		LEWC	87
0x0C	18h	0x1C	02h	VSYNC_T_END	9	LVTIME1	VDB	0
0x0D	25h	0x1D	00h	VSYNC_T_ST	4		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	-		FPW	4
0x0F	A5h	0x1F	80h	VSYNC_B_ST	-		FLC	1079
0x10	84h			VSYNC_H_POS_ST	-	LVTIME2	FSWC	35
				VSYNC_H_POS_END	-		FEWC	3
						LVTIME3	FPW2	4
							FLC2	1079
						LVTIME4	FSWC2	35
							FEWC2	3

1920x1080p @ 50Hz								
Pixel Clock		148.5	MHz					
H Frequency		56.25	KHz					
V Frequency		50	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	720	LCTRL	IV	0
0x01	05h	0x11	00h	V2_BLANK	1125		IH	0
0x02	00h	0x12	20h	V1_BLANK	45		IP	0
0x03	D8h	0x13	38h	V_LINE	1125		DP	0
0x04	10h	0x14	00h	H_LINE	2640		NI	1
0x05	1Ch	0x15	08h	VSYNC_POL	0		TV	0
0x06	30h	0x16	10h	INT_PRO_MODE	0		TFT	1
0x07	40h	0x17	E0h	V_BOT_ST	-		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	-	LHTIME1	LPW	43
0x09	18h	0x19	40h	HSYNC_START	526		LPC	1919
0x0A	01h	0x1A	B9h	HSYNC_END	570	LHTIME2	LSWC	163
0x0B	54h	0x1B	26h	HSYNC_POL	0		LEWC	511
0x0C	98h	0x1C	02h	VSYNC_T_END	9	LVTIME1	VDB	0
0x0D	21h	0x1D	00h	VSYNC_T_ST	4		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	-		FPW	4
0x0F	A5h	0x1F	80h	VSYNC_B_ST	-		FLC	1079
0x10	84h			VSYNC_H_POS_ST	-	LVTIME2	FSWC	35
				VSYNC_H_POS_END	-		FEWC	3
						LVTIME3	FPW2	4
							FLC2	1079
						LVTIME4	FSWC2	35
							FEWC2	3

8.6.2 1920x1080i

1920x1080i @ 60Hz								
Pixel Clock		74.25	MHz					
H Frequency		33.75	KHz					
V Frequency		60	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	280	LCTRL	IV	0
0x01	05h	0x11	00h	V2_BLANK	562		IH	0
0x02	00h	0x12	20h	V1_BLANK	22		IP	0
0x03	D8h	0x13	38h	V_LINE	1125		DP	0
0x04	10h	0x14	00h	H_LINE	2200		NI	0
0x05	1Ch	0x15	08h	VSYNC_POL	0		TV	1
0x06	30h	0x16	10h	INT_PRO_MODE	1		TFT	0
0x07	40h	0x17	E0h	V_BOT_ST	585		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	1125	LHTIME1	LPW	43
0x09	10h	0x19	40h	HSYNC_START	86		LPC	1919
0x0A	01h	0x1A	B9h	HSYNC_END	130	LHTIME2	LSWC	147
0x0B	54h	0x1B	26h	HSYNC_POL	0		LEWC	87
0x0C	98h	0x1C	01h	VSYNC_T_END	7	LVTIME1	VDB	0
0x0D	21h	0x1D	00h	VSYNC_T_ST	2		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	569		FPW	9
0x0F	A5h	0x1F	80h	VSYNC_B_ST	564		FLC	1079
0x10	84h			VSYNC_H_POS_ST	1188	LVTIME2	FSWC	29
				VSYNC_H_POS_END	1188		FEWC	3
						LVTIME3	FPW2	9
							FLC2	1079
						LVTIME4	FSWC2	31
							FEWC2	3

1920x1080i @ 59.94Hz								
Pixel Clock		74.175	MHz					
H Frequency		33.716	KHz					
V Frequency		59.939	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	280	LCTRL	IV	0
0x01	05h	0x11	00h	V2_BLANK	562		IH	0
0x02	00h	0x12	00h	V1_BLANK	22		IP	0
0x03	D8h	0x13	38h	V_LINE	1125		DP	0
0x04	10h	0x14	00h	H_LINE	2200		NI	0
0x05	9Ch	0x15	08h	VSYNC_POL	0		TV	1
0x06	56h	0x16	10h	INT_PRO_MODE	1		TFT	0
0x07	5Bh	0x17	E0h	V_BOT_ST	585		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	1125	LHTIME1	LPW	43
0x09	10h	0x19	40h	HSYNC_START	86		LPC	1919
0x0A	01h	0x1A	B9h	HSYNC_END	130	LHTIME2	LSWC	147
0x0B	54h	0x1B	26h	HSYNC_POL	0		LEWC	87
0x0C	18h	0x1C	01h	VSYNC_T_END	7	LVTIME1	VDB	0
0x0D	25h	0x1D	00h	VSYNC_T_ST	2		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	569		FPW	9
0x0F	A5h	0x1F	80h	VSYNC_B_ST	564		FLC	1079
0x10	84h			VSYNC_H_POS_ST	1188	LVTIME2	FSWC	29
				VSYNC_H_POS_END	1188		FEWC	3
						LVTIME3	FPW2	9
							FLC2	1079
						LVTIME4	FSWC2	31
							FEWC2	3

1920x1080i @ 50Hz								
Pixel Clock		74.25	MHz					
H Frequency		28.125	KHz					
V Frequency		50	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	720	LCTRL	IV	0
0x01	05h	0x11	00h	V2_BLANK	562		IH	0
0x02	00h	0x12	20h	V1_BLANK	22		IP	0
0x03	D8h	0x13	38h	V_LINE	1125		DP	0
0x04	10h	0x14	00h	H_LINE	2640		NI	0
0x05	1Ch	0x15	08h	VSYNC_POL	0		TV	1
0x06	30h	0x16	10h	INT_PRO_MODE	1		TFT	0
0x07	40h	0x17	E0h	V_BOT_ST	585		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	1125	LHTIME1	LPW	43
0x09	10h	0x19	40h	HSYNC_START	526		LPC	1919
0x0A	01h	0x1A	B9h	HSYNC_END	570	LHTIME2	LSWC	163
0x0B	54h	0x1B	26h	HSYNC_POL	0		LEWC	511
0x0C	98h	0x1C	01h	VSYNC_T_END	7	LVTIME1	VDB	0
0x0D	21h	0x1D	00h	VSYNC_T_ST	2		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	569		FPW	9
0x0F	A5h	0x1F	80h	VSYNC_B_ST	564		FLC	1079
0x10	84h			VSYNC_H_POS_ST	1848	LVTIME2	FSWC	29
				VSYNC_H_POS_END	1848		FEWC	3
						LVTIME3	FPW2	9
							FLC2	1079
						LVTIME4	FSWC2	31
							FEWC2	3

8.6.3 1280x720p

1280x720p @ 60Hz								
Pixel Clock		74.25	MHz					
H Frequency		45	KHz					
V Frequency		60	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	370	LCTRL	IV	0
0x01	05h	0x11	00h	V2_BLANK	750		IH	0
0x02	00h	0x12	20h	V1_BLANK	30		IP	0
0x03	D8h	0x13	38h	V_LINE	750		DP	0
0x04	10h	0x14	00h	H_LINE	1650		NI	1
0x05	1Ch	0x15	08h	VSYNC_POL	0		TV	0
0x06	30h	0x16	10h	INT_PRO_MODE	0		TFT	1
0x07	40h	0x17	E0h	V_BOT_ST	-		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	-	LHTIME1	LPW	39
0x09	10h	0x19	40h	HSYNC_START	108		LPC	1279
0x0A	01h	0x1A	B9h	HSYNC_END	148	LHTIME2	LSWC	219
0x0B	54h	0x1B	26h	HSYNC_POL	0		LEWC	109
0x0C	98h	0x1C	01h	VSYNC_T_END	10	LVTIME1	VDB	0
0x0D	21h	0x1D	00h	VSYNC_T_ST	5		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	-		FPW	4
0x0F	A5h	0x1F	80h	VSYNC_B_ST	-		FLC	719
0x10	84h			VSYNC_H_POS_ST	-	LVTIME2	FSWC	19
				VSYNC_H_POS_END	-		FEWC	4
						LVTIME3	FPW2	4
							FLC2	719
						LVTIME4	FSWC2	19
							FEWC2	4

1280x720p @ 59.94Hz								
Pixel Clock		74.175	MHz					
H Frequency		44.955	KHz					
V Frequency		59.939	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	370	LCTRL	IV	0
0x01	05h	0x11	00h	V2_BLANK	750		IH	0
0x02	00h	0x12	00h	V1_BLANK	30		IP	0
0x03	D8h	0x13	38h	V_LINE	750		DP	0
0x04	10h	0x14	00h	H_LINE	1650		NI	1
0x05	9Ch	0x15	08h	VSYNC_POL	0		TV	0
0x06	56h	0x16	10h	INT_PRO_MODE	0		TFT	1
0x07	5Bh	0x17	E0h	V_BOT_ST	-		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	-	LHTIME1	LPW	39
0x09	10h	0x19	40h	HSYNC_START	108		LPC	1279
0x0A	01h	0x1A	B9h	HSYNC_END	148	LHTIME2	LSWC	219
0x0B	54h	0x1B	26h	HSYNC_POL	0		LEWC	109
0x0C	18h	0x1C	01h	VSYNC_T_END	10	LVTIME1	VDB	0
0x0D	25h	0x1D	00h	VSYNC_T_ST	5		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	-		FPW	4
0x0F	A5h	0x1F	80h	VSYNC_B_ST	-		FLC	719
0x10	84h			VSYNC_H_POS_ST	-	LVTIME2	FSWC	19
				VSYNC_H_POS_END	-		FEWC	4
						LVTIME3	FPW2	4
							FLC2	719
						LVTIME4	FSWC2	19
							FEWC2	4

1280x720p @ 50Hz								
Pixel Clock		74.25	MHz					
H Frequency		37.5	KHz					
V Frequency		50	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	700	LCTRL	IV	0
0x01	05h	0x11	00h	V2_BLANK	750		IH	0
0x02	00h	0x12	20h	V1_BLANK	30		IP	0
0x03	D8h	0x13	38h	V_LINE	750		DP	0
0x04	10h	0x14	00h	H_LINE	1980		NI	1
0x05	1Ch	0x15	08h	VSYNC_POL	0		TV	0
0x06	30h	0x16	10h	INT_PRO_MODE	0		TFT	1
0x07	40h	0x17	E0h	V_BOT_ST	-		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	-	LHTIME1	LPW	39
0x09	10h	0x19	40h	HSYNC_START	438		LPC	1279
0x0A	01h	0x1A	B9h	HSYNC_END	478	LHTIME2	LSWC	219
0x0B	54h	0x1B	26h	HSYNC_POL	0		LEWC	439
0x0C	98h	0x1C	01h	VSYNC_T_END	10	LVTIME1	VDB	0
0x0D	21h	0x1D	00h	VSYNC_T_ST	5		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	-		FPW	4
0x0F	A5h	0x1F	80h	VSYNC_B_ST	-		FLC	719
0x10	84h			VSYNC_H_POS_ST	-	LVTIME2	FSWC	19
				VSYNC_H_POS_END	-		FEWC	4
						LVTIME3	FPW2	4
							FLC2	719
						LVTIME4	FSWC2	19
							FEWC2	4

8.6.4 720x480p

720x480p @ 60Hz								
Pixel Clock		27.027	MHz					
H Frequency		31.5	KHz					
V Frequency		60	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	138	LCTRL	IV	1
0x01	05h	0x11	00h	V2_BLANK	525		IH	1
0x02	00h	0x12	00h	V1_BLANK	45		IP	0
0x03	D8h	0x13	38h	V_LINE	525		DP	0
0x04	10h	0x14	00h	H_LINE	858		NI	1
0x05	9Ch	0x15	08h	VSYNC_POL	1		TV	0
0x06	02h	0x16	10h	INT_PRO_MODE	0		TFT	1
0x07	32h	0x17	E0h	V_BOT_ST	-		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	-	LHTIME1	LPW	61
0x09	10h	0x19	40h	HSYNC_START	14		LPC	719
0x0A	02h	0x1A	FFh	HSYNC_END	76	LHTIME2	LSWC	59
0x0B	54h	0x1B	26h	HSYNC_POL	1		LEWC	15
0x0C	38h	0x1C	00h	VSYNC_T_END	15	LVTIME1	VDB	0
0x0D	20h	0x1D	00h	VSYNC_T_ST	9		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	-		FPW	5
0x0F	78h	0x1F	80h	VSYNC_B_ST	-		FLC	479
0x10	84h			VSYNC_H_POS_ST	-	LVTIME2	FSWC	29
				VSYNC_H_POS_END	-		FEWC	8
						LVTIME3	FPW2	5
							FLC2	479
						LVTIME4	FSWC2	29
							FEWC2	8

720x480p @ 59.94Hz								
Pixel Clock		27	MHz					
H Frequency		31.469	KHz					
V Frequency		59.94	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	138	LCTRL	IV	1
0x01	05h	0x11	00h	V2_BLANK	525		IH	1
0x02	00h	0x12	00h	V1_BLANK	45		IP	0
0x03	D8h	0x13	38h	V_LINE	525		DP	0
0x04	10h	0x14	00h	H_LINE	858		NI	1
0x05	1Ch	0x15	08h	VSYNC_POL	1		TV	0
0x06	30h	0x16	10h	INT_PRO_MODE	0		TFT	1
0x07	40h	0x17	E0h	V_BOT_ST	-		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	-	LHTIME1	LPW	61
0x09	10h	0x19	40h	HSYNC_START	14		LPC	719
0x0A	02h	0x1A	FFh	HSYNC_END	76	LHTIME2	LSWC	59
0x0B	54h	0x1B	26h	HSYNC_POL	1		LEWC	15
0x0C	38h	0x1C	00h	VSYNC_T_END	15	LVTIME1	VDB	0
0x0D	20h	0x1D	00h	VSYNC_T_ST	9		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	-		FPW	5
0x0F	78h	0x1F	80h	VSYNC_B_ST	-		FLC	479
0x10	84h			VSYNC_H_POS_ST	-	LVTIME2	FSWC	29
				VSYNC_H_POS_END	-		FEWC	8
						LVTIME3	FPW2	5
							FLC2	479
						LVTIME4	FSWC2	29
							FEWC2	8

8.6.5 720x576p

720x576p @ 50Hz								
Pixel Clock		27	MHz					
H Frequency		31.25	KHz					
V Frequency		50	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	144	LCTRL	IV	1
0x01	05h	0x11	00h	V2_BLANK	625		IH	1
0x02	00h	0x12	00h	V1_BLANK	49		IP	0
0x03	D8h	0x13	38h	V_LINE	625		DP	0
0x04	10h	0x14	00h	H_LINE	864		NI	1
0x05	1Ch	0x15	08h	VSYNC_POL	1		TV	0
0x06	30h	0x16	10h	INT_PRO_MODE	0		TFT	1
0x07	40h	0x17	E0h	V_BOT_ST	-		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	-	LHTIME1	LPW	63
0x09	10h	0x19	40h	HSYNC_START	10		LPC	719
0x0A	02h	0x1A	FFh	HSYNC_END	74	LHTIME2	LSWC	67
0x0B	54h	0x1B	26h	HSYNC_POL	1		LEWC	11
0x0C	38h	0x1C	00h	VSYNC_T_END	10	LVTIME1	VDB	0
0x0D	20h	0x1D	00h	VSYNC_T_ST	5		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	-		FPW	4
0x0F	78h	0x1F	80h	VSYNC_B_ST	-		FLC	575
0x10	84h			VSYNC_H_POS_ST	-	LVTIME2	FSWC	38
				VSYNC_H_POS_END	-		FEWC	4
						LVTIME3	FPW2	4
							FLC2	575
						LVTIME4	FSWC2	38
							FEWC2	4

8.6.6 720x480i

720x480i @ 60Hz								
Pixel Clock		27.027	MHz					
H Frequency		15.75	KHz					
V Frequency		60	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	276	LCTRL	IV	1
0x01	05h	0x11	00h	V2_BLANK	262		IH	1
0x02	00h	0x12	00h	V1_BLANK	22		IP	0
0x03	D8h	0x13	38h	V_LINE	525		DP	1
0x04	10h	0x14	00h	H_LINE	1716		NI	0
0x05	9Ch	0x15	08h	VSYNC_POL	1		TV	1
0x06	02h	0x16	10h	INT_PRO_MODE	1		TFT	0
0x07	32h	0x17	E0h	V_BOT_ST	285		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	525	LHTIME1	LPW	123
0x09	10h	0x19	40h	HSYNC_START	36		LPC	1439
0x0A	02h	0x1A	FFh	HSYNC_END	160	LHTIME2	LSWC	113
0x0B	54h	0x1B	26h	HSYNC_POL	1		LEWC	37
0x0C	38h	0x1C	00h	VSYNC_T_END	7	LVTIME1	VDB	0
0x0D	20h	0x1D	00h	VSYNC_T_ST	4		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	269		FPW	5
0x0F	78h	0x1F	80h	VSYNC_B_ST	266		FLC	479
0x10	84h			VSYNC_H_POS_ST	896	LVTIME2	FSWC	29
				VSYNC_H_POS_END	896		FEWC	7
						LVTIME3	FPW2	5
							FLC2	479
						LVTIME4	FSWC2	31
							FEWC2	7

720x480i @ 59.94Hz								
Pixel Clock		27	MHz					
H Frequency		15.734	KHz					
V Frequency		59.94	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	276	LCTRL	IV	1
0x01	05h	0x11	00h	V2_BLANK	262		IH	1
0x02	00h	0x12	00h	V1_BLANK	22		IP	0
0x03	D8h	0x13	38h	V_LINE	525		DP	1
0x04	10h	0x14	00h	H_LINE	1716		NI	0
0x05	1Ch	0x15	08h	VSYNC_POL	1		TV	1
0x06	30h	0x16	10h	INT_PRO_MODE	1		TFT	0
0x07	40h	0x17	E0h	V_BOT_ST	285		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	525	LHTIME1	LPW	123
0x09	10h	0x19	40h	HSYNC_START	36		LPC	1439
0x0A	02h	0x1A	FFh	HSYNC_END	160	LHTIME2	LSWC	113
0x0B	54h	0x1B	26h	HSYNC_POL	1		LEWC	37
0x0C	38h	0x1C	00h	VSYNC_T_END	7	LVTIME1	VDB	0
0x0D	20h	0x1D	00h	VSYNC_T_ST	4		VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	269		FPW	5
0x0F	78h	0x1F	80h	VSYNC_B_ST	266		FLC	479
0x10	84h			VSYNC_H_POS_ST	896	LVTIME2	FSWC	29
				VSYNC_H_POS_END	896		FEWC	7
						LVTIME3	FPW2	5
							FLC2	479
						LVTIME4	FSWC2	31
							FEWC2	7

8.6.7 720x576i

720x576i @ 50Hz								
Pixel Clock		27	MHz					
H Frequency		15.625	KHz					
V Frequency		50	Hz					
HDMI PHY Setting				HDMI link setting		LCD Setting		
Address	Value	Address	Value	H_BLANK	288		IV	1
0x01	05h	0x11	00h	V2_BLANK	312		IH	1
0x02	00h	0x12	00h	V1_BLANK	24		IP	0
0x03	D8h	0x13	38h	V_LINE	625	LCTRL	DP	1
0x04	10h	0x14	00h	H_LINE	1728		NI	0
0x05	1Ch	0x15	08h	VSYNC_POL	1		TV	1
0x06	30h	0x16	10h	INT_PRO_MODE	1		TFT	0
0x07	40h	0x17	E0h	V_BOT_ST	337		STN	0
0x08	6Bh	0x18	22h	V_BOT_END	625	LHTIME1	LPW	125
0x09	10h	0x19	40h	HSYNC_START	22		LPC	1439
0x0A	02h	0x1A	FFh	HSYNC_END	148	LHTIME2	LSWC	137
0x0B	54h	0x1B	26h	HSYNC_POL	1		LEWC	23
0x0C	38h	0x1C	00h	VSYNC_T_END	5		VDB	0
0x0D	20h	0x1D	00h	VSYNC_T_ST	2	LVTIME1	VDF	0
0x0E	5Bh	0x1E	00h	VSYNC_B_END	317		FPW	5
0x0F	78h	0x1F	80h	VSYNC_B_ST	314		FLC	575
0x10	84h			VSYNC_H_POS_ST	888	LVTIME2	FSWC	37
				VSYNC_H_POS_END	888		FEWC	3
						LVTIME3	FPW2	5
							FLC2	575
						LVTIME4	FSWC2	39
							FEWC2	3

9 Video and Image Quality Enhancer (VIQE)

9.1 Overview

The VIQE(Video and Image Quality Enhancer) block is high-resolution video/image quality enhancing hardware that facilitates lot of video/image processing features. Aimed primarily at digital flat high-resolution display, it provides advanced and efficient video improvement solutions. In addition to the high quality, the pipeline architecture with specific and characteristic structure can provide minimized memory bandwidth requirement and dynamic module interoperations.
The main functions of the VIQE are listed below.

- De-interlacing
- Edge-based/Motion-adaptive
- File-mode processing
- Judder Cancellation
- Remove Noise
- Temporal/Spatial Engines
- Edge Enhancing
- Noise Measurement
- Histogram Measurement
- Contrast Enhancing
- Color Gamut Mapping

The interlaced video sources are spreading very widely, mainly for broadcasting, film sources and DVD video markets. With the need of interlaced contents to be different, nowadays the main display interface is flat panel display, so the conversion into the progressive content is indispensible. The de-interlacing hardware of VIQE has advanced and efficient features, all of which support pixel-wise processing and are composed of several algorithms corresponded to very various situations making the artifacts. The film mode detection becomes an essential feature in interlaced-to-progressive conversion as a usual de-interlacing for file mode contents tends to make the output to be contaminated and to have reduced resolution spatially and temporally. The film mode detector in the VIQE supports reverse 3:2 pull-down and also compensates pixel-by-pixel the bad-edit problem and the additional text problem having not film frame rate.

To reduce the noise over video sequence effectively, the procedure of discriminating between noise and normal information in using temporal tracing as well as spatial data referencing is necessary. The de-noising hardware in the VIQE uses a novel algorithm with neighboring data in spatial and temporal domain, and controls the strength with recursive path. Also, the VIQE has the specific hardware for reducing the memory bandwidth, and that is important feature because the recursive path for de-noiser may need additional memory bandwidth.

Some of the hardware engine utilized for de-noising procedure can be shared for the edge enhancing, which improves the contrast of the image and can be expected to make the image more clear and vivid through harmonizing the de-noising. Besides, the VIQE provides the programmability of utilizing the hardware filter structure through arbitrary filter coefficients.
Histogram measurement hardware gathers automatically histogram characteristics at each frame or for user-defined duration. This information can be used for image contrast improvements. In addition, the VIQE provides some optional functions for utilizing measured histogram information.

Lastly, color gamut mapping hardware can be used for calibration of color characteristics to be different as display panel devices. Nowadays, the memorial color concept such as skin color, blue sky, and green grass becomes more important. The color gamut mapping hardware provides the flexibility to apply various gamut mapping algorithms with LUT-based programmability. Easily accessible software tool is provided.

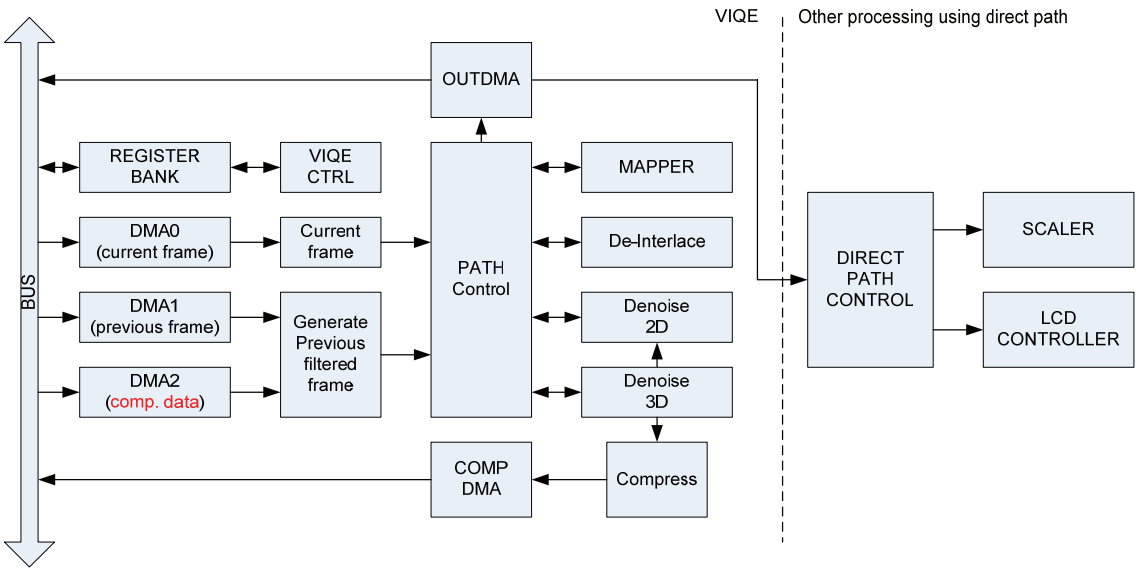


Figure 9.1 VIQE Block Diagram

The VIQE is composed to the major five modules, or DMA ports, de-Interlacer, de-noiser, spatial universal filter, and pixel-mapper. Each module can share the hardware resources and operates independently with some limitations. Any combination of enhancing procedure in the VIQE

Each module is designed to share the hardware resources with other modules to the highest degree, and to have the flexibility to select between performance and quality if the resource sharing is not possible in that combination.
The four DMA channels have specific services for transferring data. Two DMA channels for fetching input frame data, one DMA channel for writing output frame data, and one DMA channel for the recursive data have independent internal FIFO and can operate simultaneously. The output DMA channel can be used as direct path to LCDC as well as external or internal frame buffer interface. The DMAs automatically performs burst word (64-bit) transfers, filling or emptying of the FIFO.

The particular hardware for compressing and de-compressing data is attached to recursive path DMA port. This hardware is used only to reduce memory bandwidth, and the algorithm of compressing is designed to be very suitable for the recursive algorithm in de-noising hardware. This compressor/de-compressor has the flexibility of controlling the strength and performance of compressing.

9.2 De-interlacing module

The objective of de-interlacing is to convert interlaced scanning into progressive scanning, overcoming the intrinsic nature of the interlaced video sampling structure. In interlaced scanning method, if a fast moving part exists between two fields resulting annoying jitters between lines, and an edges especially along horizontal direction exists, that performance degradation of de-interlacing becomes more noticeable. In other words, there is a theoretical limitation in high frequency components temporally or spatially.
The de-interlacing module in the VIQE has the following features.

- Pixel-wise processing
- Edge directional interpolation
- Motion adaptive processing
- 3:2 pull-down film mode detection
- Bad edition/judder detection

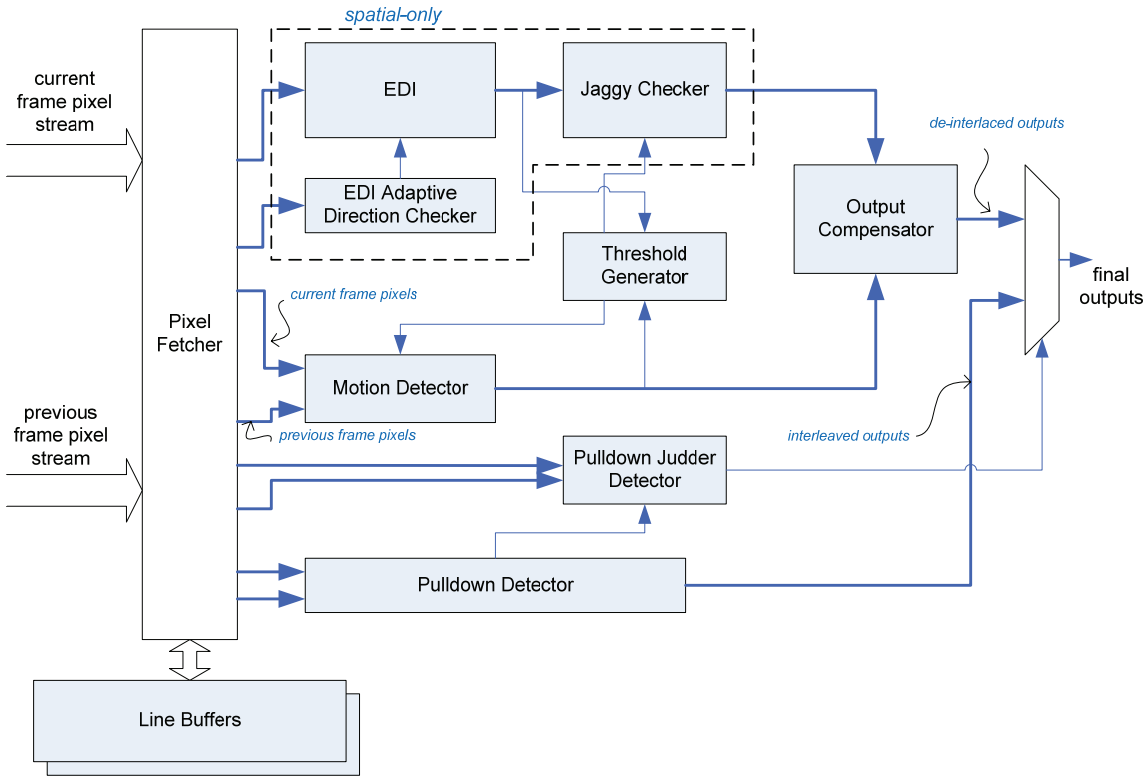


Figure 9.2 De-Interlacer Block Diagram

De-interlacing hardware is composed of several sub modules to implement various algorithms for compensating and coping with the serious negative effects from conversion scanning. Figure below describes the internal block diagram of de-interlacing module. All modules including de-interlacing module in the VIQE are pipelined in inter-module as well as inner-module, and input/output interfaces consist of actual pixel data stream and timing control signals.
The EDI, EDI adaptive direction checker, and jaggy checker have functions for spatial de-interlacing, and are default streaming path if the motion detector determines solid existence of motion or if the spatial-only mode in the control register is selected. The threshold generator block can provide the thresholds or parameter values which can be changed dynamically adapting the neighboring pixels instead of fixing as user-defined values. Motion detector decides whether the current pixel is moving or not through examination of neighboring pixels spatially or temporally. The failure of detecting motion causes serious visual artifacts, so the output compensator would check the reliability of the result of motion adaptive processing as well as spatial adaptive processing.
The pulldown detector and pulldown judder detector are designed to operate reverse 3:2 pull-down procedure, which consists of three steps in the VIQE. The first is to detect whether this stream is film source or not, and the next step is to check the existence of judder caused by a bad edition or addition of texture at normal display rate and the last step is to interleaving accurate field line or to operate de-interlacing at the detected judder pixels. The detection of film mode takes advantage of statistical information acquired during some frames.

9.3 De-noising module

The objective of de-noising is to remove the noise or sharpening of edge from a single image or video stream. De-noiser is composed of the temporal de-noiser and spatial one that are supported the sharpening and smoothing. Two blocks of de-noiser can support several data-path connection methods as follows.

- Usage of only temporal path (blue arrow)
- Usage of only spatial path (red arrow)
- Usage of both temporal and spatial path (magenta arrow)
  - 1. In first, smoothing operation, next sharpening.
  - 2. All operation is smoothing. (But it is not recommended.)

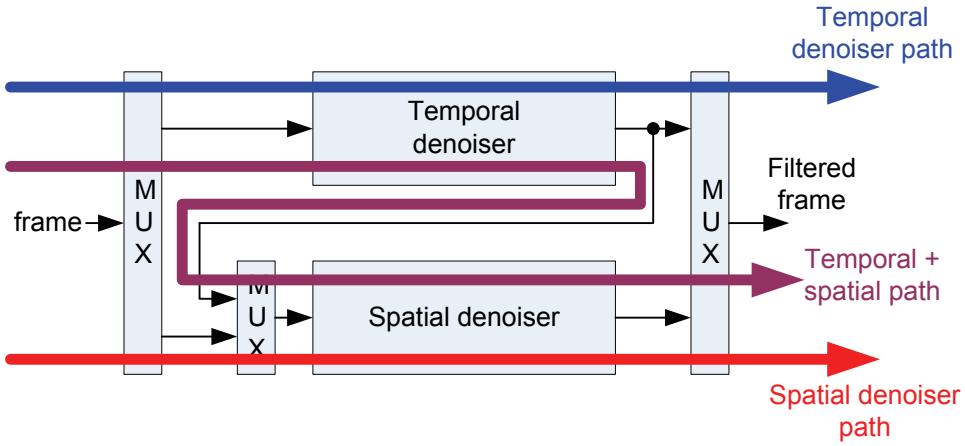


Figure 9.3 De-noiser Block Diagram

9.3.1.1 Temporal de-noiser

Temporal de-noiser detects and reduces noises through referencing the adjacent pixels in horizontal and vertical spatial domain and tracing the pixels of previous frame in temporal domain. The filtering method of each domain is the method of a kind of recursive filter which is fed on the current input and the immediate filtered data. Figure 9.4 describes the block diagram of the temporal de-noiser.

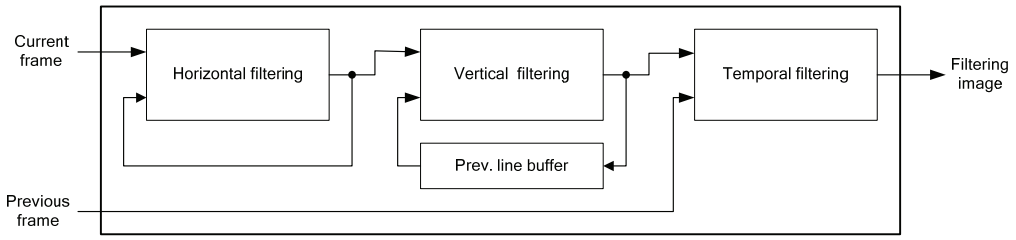


Figure 9.4 Temporal de-noiser Block Diagram

This module needs to store the filtered data for feeding at the following processing because all filter structures have the recursive path. In the temporal domain processing, because the size of frame buffer for storing the previous filtered immediate data is very huge, the special means for reducing memory bandwidth should be considered.
In advance, the differential data between the filtered frame data and original frame data are calculated. In the statistics, because the average of this differential data tends to converge into zero, the size of data to be stored is reduced effectively. Afterward, the differential data are compressed using the specific algorithm suitable for the statistical characteristics of the differential data. These procedures are implemented in fully hardware, and the controllability of adjusting compression strength through register setting is provided.

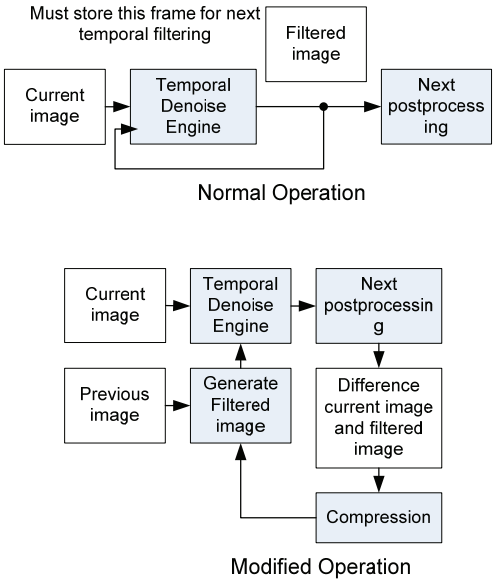


Figure 9.5 Recursive operation

9.3.1.2 Spatial de-noiser

The spatial de-noiser without referencing the previous frame data in temporal domain uses only the adjacent pixels of the current pixel to be processed. Figure 9.6 shows the internal architecture in which low-pass filter and high-pass filter share the main data path. The main procedure of this block is to take consideration of the edge directional properties about the current processing window block.

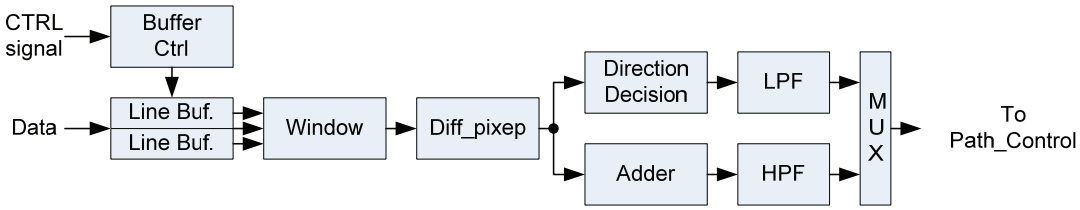


Figure 9.6 Spatial de-noising block

In LPF mode, the smoothing filter with preserving the edge property in the current window can reduce the noise along the same direction like that of the detected edge. On the contrary, the HPF mode emphasizes the edge effectively.

9.4 RDMA (Read DMA)

The RDMA block is the first beginning block in the VIQE, and should be enabled before any operation in the VIQE starts. The RDMA has three channels as Figure 9.7 depicts, and two channels of three are used as normal transferring the current frame and the previous frame which is for the temporal mode in either de-interlacer or de-noiser. The other channel is designed especially for reading the compressed reference data, which is explained in the de-noiser temporal mode. The read data are stored in temporary buffer and are transferred as streaming format similar to CCIR format. To synchronize three kinds of input data, the control signals from each RDMA channel are communicated with the PATH_CONTROL block as the VIQE central block.

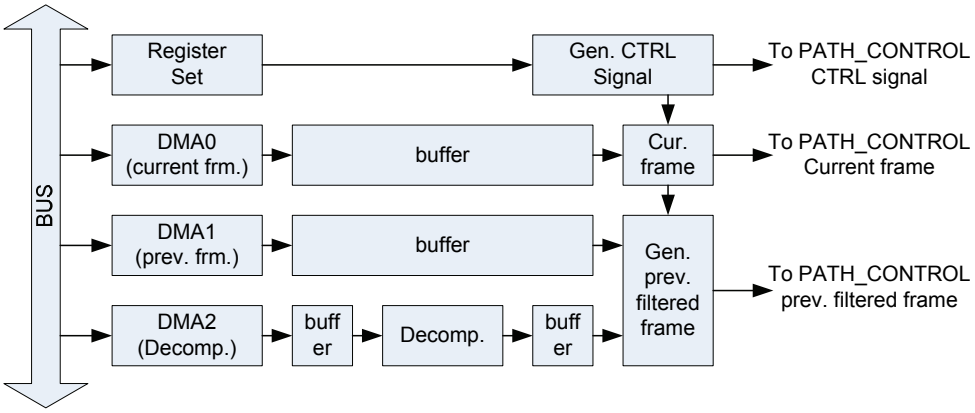


Figure 9.7 RDMA block diagram

9.5 ODMA (OUTPUT DMA)

The ODMA is for storing the final output pixel into external memory, or to transfer those results directly into display device or other processing blocks such as image scaler. The merit of the direct path is to prevent the additional memory read/write accesses just for external display. Such the on-the-fly operations as this make it possible to process all features of the VIQE without repetitive or unnecessary memory accesses from staring at the RDMA to finishing at the ODMA. Figure 9.8 shows the block diagram of the ODMA connected to external scaler and LCD controller through the direct path controller.

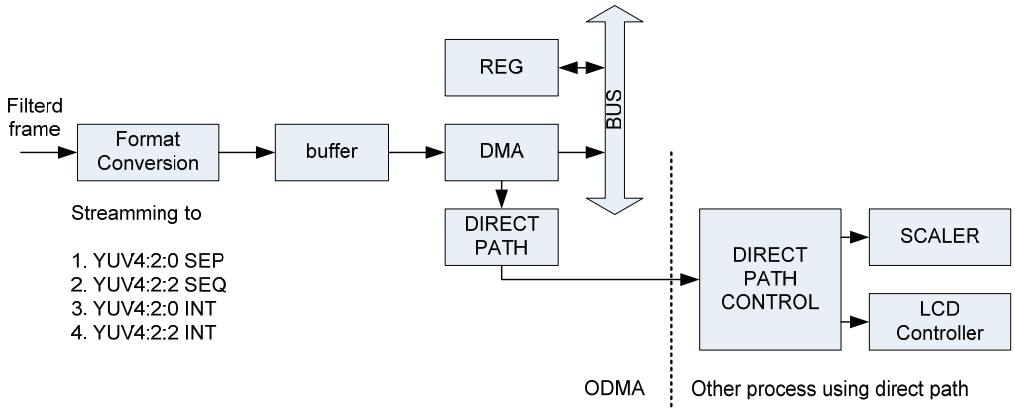


Figure 9.8 ODMA block diagram

9.6 Gamut mapper module

The gamut mapper module in the VIQE is designed to map the arbitrary three color component data into specified three component data. The mapping table is based on segmented LUT, and the missing value between adjacent segments is interpolated using the pre-specified value of vertex in segmented cubic. This method do not provide a mapping characteristics changed dynamically pixel-wise, but provide a color converting space which maps any arbitrary pixel into another arbitrary pixel. Also, unlike a hardware-fixed algorithm implementation, the table-based implementation can be modified according as a specific algorithm.
The programming method can be provided as software package.

9.7 Histogram Generator module

The histogram generator is used to measure histogram of video image. It supports fast histogram processing. It enhances image contrast using automatic equalization of measured histogram.
The features of the histogram generator are as follows.

- Supports up to 1920 * 1080 (HDTV) image size
- Supports segmented histogram measure of raw video image
- Supports fast CDF generation and interpolation of measured histogram
- Supports programmable histogram equalization of video image
- Supports statistical histogram processing of multi-frame image
-

The following key parameters can be programmed.

- Input image width / height
- Horizontal / vertical measuring frequency
- Size of measured frame
- 16 histogram segment range
- Equalization strength of each segment

The abstract behavior of histogram generator is as follows.
Raw image data is synchronously presented with hsyn signal. vsyn signal generated at hsyn interval time. Horizontal / vertical offset counters are incremented by hsyn and vsyn signal. If offset counters are matched to each offset size, matched offset counter generates the match signal and cleared at next sync signal. If vertical and horizontal match signal is generated, the pixel data is captured and one of sixteen pixel counter is incremented. Pixel counters represent segmented histogram data. Captured pixel data is compared to each segment data. If pixel data is in the range of one segment, pixel counter of that segment is incremented. Histogram measurement can be processed in multi-frame size. Multi_frame_size is defined by the MFRM field in HI_CTRL register.
Generated pixel counter values are divided by Multi_frame_size to average multi-frame histogram data. Averaged pixel counter values are accumulated to make normalized CDF of measured histogram. Normalized CDF data is generated by equations (1)-(4). Sample size is given by external control machine.

$$avg\_pixel\_count[i] = \frac{pixel\_count[i]}{Multi\_frame\_size} \quad (1)$$

$$acc\_data[i] = \sum_{k=0}^{i} avg\_pixel\_count[i] \quad (2)$$

$$sample\_size = (\frac{image\_width}{hoffset\_size}) \times (\frac{image\_height}{voffset\_size} + 1) \quad (3)$$

$$normalized\_CDF[i] = \frac{acc\_data[i]}{sample\_size} \times 256 \quad (4)$$

Normalized CDF data can be used to make LUT. To make image preserving equalization, we make little variation to original image. Following equation shows the generation process of segmented LUT. The eq_scale values represent equalization strength. This value can be different in each segment, and its range is 0 to 128. Larger eq_scale makes stronger equalized CDF data.

$$LUT[i] = (normalized\_CDF[i] - org[i]) \times eq\_scale[i]) >> 7 + org[i] \quad (5)$$

Segmented LUT data is just composed of 16 values which can be set to be arbitrary. Histogram generator interpolates 16 LUT data to 256 LUT data. The missing value between adjacent segments is interpolated linearly. The interpolated data can be used in histogram processing or LUT mapping.
If only the histogram measurement is used, the programming of the hardware is simple. When the automatic mode is selected, the user only read the result registers at the end of one frame processing or at whenever the user wants to read. The result registers are categorized as three groups, one of which is just raw counter value at the specified segments, and the second group is the normalized CDF format, the range of which is 0 ~ 255 and which can be used as the transformation

function like the histogram equalization transformation. Both of groups are not fully ranged mapping table, but user-defined segmented mapping table. The last group is the interpolated 256 LUT values, which are calculated from the normalized CDF format and the missing values inside each segment are interpolated linearly. These values can be read in the HI_LUT registers.
Otherwise, if the histogram is configured as the equalization mapping, the programming method is little complex. If the user tends to read the LUT output, the user should read that results only when the frame processing is done. Otherwise the result values would be corrupted. This checking can be through monitoring the status register or controlling the re-start control register.

9.8 Register Description

Table 9.1 VIQE Register Map (Base Address = 0xB0A52000)

Name	Address	Type	Reset	Description
CTRL	0x000	R/W	0x00000000	VIQE General Control Register
SIZE	0x004	R/W	0x00000000	VIQE SIZE Register
TIMEGEN	0x008	R/W	0x00000000	VIQE Time Generator Register
LUMADLY	0x00C	R/W	0x00000000	VIQE Luma Delay Register
IMGCONF	0x010	R/W	0x00000000	VIQE Image Configuration Register
IMGFMT	0x014	R/W	0x00000000	VIQE Image Format Register
MISCC	0x018	R/W	0x00000000	VIQE Misc, Control Register
FRMC	0x01C	R/W	0x00000000	VIQE Frame Control Register
INT	0x020	R/W	0x00000000	VIQE Interrupt Register
INTMASK	0x024	R/W	0x00000000	VIQE Interrupt Mask Register
VERSION	0x03c	R	0x4d2b3401	VIQE Version Register

Name	Address	Type	Reset	Description
DI_CTRL	0x080	R/W	0x00000000	De-interlacer Control Register
DI_ENGINE0	0x084	R/W	0x00000000	De-interlacer Engine 0 Register
DI_ENGINE1	0x088	R/W	0x00000000	De-interlacer Engine 1 Register
DI_ENGINE2	0x08C	R/W	0x00000000	De-interlacer Engine 2 Register
DI_ENGINE3	0x090	R/W	0x00000000	De-interlacer Engine 3 Register
DI_ENGINE4	0x094	R/W	0x00000000	De-interlacer Engine 4 Register
PD_THRES0	0x098	R/W	0x00000000	De-interlacer Pulldown Threshold 0 Register
PD_THRES1	0x09C	R/W	0x00000000	De-interlacer Pulldown Threshold 1 Register
PD_JUDDER	0x0A0	R/W	0x00000000	De-interlacer Pulldown Judder Register
PD_JUDDER_M	0x0A4	R/W	0x00000000	De-interlacer Pulldown Judder Misc. Register
DI_MISCC	0x0A8	R/W	0x00000000	De-interlacer Misc. Control Register
DI_STATUS	0x0AC	R		De-interlacer Status Register
PD_STATUS	0x0B0	R		De-interlacer Pulldown Status Register
DI_REGION0	0x0B4	R/W	0x00000000	De-interlacer Region 0 Register
DI_REGION1	0x0B8	R/W	0x00000000	De-interlacer Region 1 Register
DI_INT	0x0BC	R/W	0x00000000	De-interlacer Interrupt Register

Name	Address	Type	Reset	Description
DN_C_H_Y0	0x100	R/W	0xbfffffa4	Denoise-3D horizontal coefficient #0 in luminance
DN_C_H_Y1	0x104	R/W	0x15556aaa	Denoise-3D horizontal coefficient #1 in luminance
DN_C_V_Y0	0x108	R/W	0xaaaaaaa4	Denoise-3D vertical coefficient #0 in luminance
DN_C_V_Y1	0x10C	R/W	0x15556aaa	Denoise-3D vertical coefficient #1 in luminance
DN_C_T_Y0	0x110	R/W	0xaaaaaaa4	Denoise-3D temporal coefficient #0 in luminance
DN_C_T_Y1	0x114	R/W	0x15556aaa	Denoise-3D temporal coefficient #1 in luminance
DN_C_H_C0	0x118	R/W	0xbfffffa4	Denoise-3D horizontal coefficient #0 in chrominance
DN_C_H_C1	0x11C	R/W	0x15556aaa	Denoise-3D horizontal coefficient #1 in chrominance
DN_C_V_C0	0x120	R/W	0xaaaaaaa4	Denoise-3D vertical coefficient #0 in chrominance
DN_C_V_C1	0x124	R/W	0x15556aaa	Denoise-3D vertical coefficient #1 in chrominance
DN_C_T_C0	0x128	R/W	0xaaaaaaa4	Denoise-3D temporal coefficient #0 in chrominance
DN_C_T_C1	0x12C	R/W	0x15556aaa	Denoise-3D temporal coefficient #1 in chrominance
DN_STATE0_3D	0x130	R/W	0x00000000	Denoise-3D count states and int. mask
DN_STATE1_3D	0x134	R	-	Denoise-3D count states
DN_DIV_IMG_3D	0x138	R/W	0x00000168	Denoise-3D image divide
DN_C_2D_Y0	0x140	R/W	0x12320e0a	Denoise-2D coefficient #0 in luminance
DN_C_2D_Y1	0x144	R/W	0x373c051d	Denoise-2D coefficient #1 in luminance
DN_C_2D_Y2	0x148	R/W	0x4a0640ff	Denoise-2D coefficient #2 in luminance
DN_C_2D_Y3	0x14C	R/W	0x003100fb	Denoise-2D coefficient #3 in luminance

Name	Address	Type	Reset	Description
DN_C_2D_C0	0x150	R/W	0x12190805	Denoise-2D coefficient #0 in chrominance
DN_C_2D_C1	0x154	R/W	0x373c0507	Denoise-2D coefficient #1 in chrominance
DN_C_2D_C2	0x158	R/W	0x4a0640ff	Denoise-2D coefficient #2 in chrominance
DN_C_2D_C3	0x15C	R/W	0x003100fb	Denoise-2D coefficient #3 in chrominance
DN_FIFOSTATE	0x160	R/W	0x00000000	FIFO states
DN_STATE0_2D	0x164	R/W	0x00000000	Denoise-2D count states and int.mask
DN_STATE1_2D	0x168	R	-	Denoise-2D count states
DN_CTRL	0x16C	R/W	0x00000000	Denoise FIFO and coefficient ctrl
DN_DIV_IMG_2D	0x170	R/W	0x00000168	Denoise-2D image divide

Name	Address	Type	Reset	Description
RD_IMG0_BASE0	0x180	R/W	0x00000000	RDMA image #0 base address in Y channel
RD_IMG0_BASE1	0x184	R/W	0x00000000	RDMA image #0 base address in U channel
RD_IMG0_BASE2	0x188	R/W	0x00000000	RDMA image #0 base address in V channel
RD_IMG0_OFS	0x18C	R/W	0x00000000	RDMA image #0 address offset
RD_IMG1_BASE0	0x190	R/W	0x00000000	RDMA image #1 base address in Y channel
RD_IMG1_BASE1	0x194	R/W	0x00000000	RDMA image #1 base address in U channel
RD_IMG1_BASE2	0x198	R/W	0x00000000	RDMA image #1 base address in V channel
RD_IMG1_OFS	0x19C	R/W	0x00000000	RDMA image #1 address offset
RD_IMG2_BASE0_0	0x1A0	R/W	0x00000000	RDMA Comp. data #0 base address in Y channel
RD_IMG2_BASE1_0	0x1A4	R/W	0x00000000	RDMA Comp. data #0 base address in U channel
RD_IMG2_BASE2_0	0x1A8	R/W	0x00000000	RDMA Comp. data #0 base address in V channel
RD_IMG2_BASE0_1	0x1AC	R/W	0x00000000	RDMA Comp. data #1 base address in Y channel
RD_IMG2_BASE1_1	0x1B0	R/W	0x00000000	RDMA Comp. data #1 base address in U channel
RD_IMG2_BASE2_1	0x1B4	R/W	0x00000000	RDMA Comp. data #1 base address in V channel
RD_CUR_ADDR0	0x1B8	R	-	RDMA image #0 current address
RD_CUR_ADDR1	0x1BC	R	-	RDMA image #1 current address
RD_CUR_ADDR2	0x1C0	R	-	RDMA Huffman data current address
RD_FIFOSTATE	0x1C4	R/W	0x00000000	RDMA FIFO States
RD_LINE_STATE0	0x1C8	R	-	RDMA count states #0
RD_LINE_STATE1	0x1CC	R/W	0x00000000	RDMA count states #1
RD_CTRL	0x1D0	R/W	0x00000000	RDMA control register
RD_HUFF_PL0	0x1D4	R/W	0x00000000	RDMA Comp. data number in Y channel
RD_HUFF_PL1	0x1D8	R/W	0x00000000	RDMA Comp. data number in C channel
-	0x1DC	-	-	-
RD_IMG2_ENDADR0	0x1E0	R/W	0x00000000	RDMA Comp. data end address in Y channel
RD_IMG2_ENDADR1	0x1E4	R/W	0x00000000	RDMA Comp. data end address in U channel
RD_IMG2_ENDADR2	0x1E8	R/W	0x00000000	RDMA Comp. data end address in V channel
RD_ROLL_CTRL	0x1EC	R/W	0x00000000	RDMA Comp. DMA rolling-mode control register
RD_IMG2_CURADR0	0x1E0	R/W	0x00000000	RDMA Comp. current address in Y channel
RD_IMG2_CURADR1	0x1E4	R/W	0x00000000	RDMA Comp. current address in U channel
RD_IMG2_CURADR2	0x1E8	R/W	0x00000000	RDMA Comp. current address in V channel

Name	Address	Type	Reset	Description
WD_BASE0_0	0x200	R/W	0x00000000	Comp. DMA #0 base address in Y channel
WD_BASE1_0	0x204	R/W	0x00000000	Comp. DMA #0 base address in U channel
WD_BASE2_0	0x208	R/W	0x00000000	Comp. DMA #0 base address in V channel
WD_BASE0_1	0x20C	R/W	0x00000000	Comp. DMA #1 base address in Y channel
WD_BASE1_1	0x210	R/W	0x00000000	Comp. DMA #1 base address in U channel
WD_BASE2_1	0x214	R/W	0x00000000	Comp. DMA #1 base address in V channel
WD_CUR_ADDR	0x218	R	-	Comp. DMA current address
WD_STATE	0x21C	R/W	0x00000000	Comp. DMA states
WD_CTRL	0x220	R/W	0x00000000	Comp. DMA control register
WD_HUFF_CNT0	0x230	R	-	Comp. DMA encoded data count in Y channel
WD_HUFF_CNT1	0x234	R	-	Comp. DMA encoded data count in U channel
WD_HUFF_CNT2	0x238	R		Comp. DMA encoded data count in V channel
-	0x23C	-	-	-
WD_ENDADR0	0x240	R/W	0x00000000	Comp. DMA end-address in Y channel
WD_ENDADR1	0x244	R/W	0x00000000	Comp. DMA end-address in U channel
WD_ENDADR2	0x248	R/W	0x00000000	Comp. DMA end-address in V channel
WD_ROLL_CTRL	0x24C	R/W	0x00000000	Comp. DMA rolling-mode control register
WD_CUR_ADR0	0x240	R/W	0x00000000	Comp. DMA current address in Y channel
WD_CUR_ADR1	0x244	R/W	0x00000000	Comp. DMA current address in U channel
WD_CUR_ADR2	0x248	R/W	0x00000000	Comp. DMA current address in V channel

Name	Address	Type	Reset	Description
OD_BASE0	0x280	R/W	0x00000000	ODMA base address in Y channel
OD_BASE1	0x284	R/W	0x00000000	ODMA base address in U channel
OD_BASE2	0x288	R/W	0x00000000	ODMA base address in V channel
OD_SIZE	0x28C	R/W	0x00000000	ODMA image size
OD_OFS	0x290	R/W	0x00000000	ODMA address offset
OD_CFG	0x294	R/W	0x00000000	ODMA image type
OD_CTRL	0x2A4	R/W	0x00000000	ODMA control register
OD_STATE	0x2A8	R/W	0x00000000	ODMA States

Name	Address	Type	Reset	Description
GM_CTRL	0x400	R/W	0x00000000	Gamut-mapper Control Register
GM_STATUS	0x404	R/W	0x00000000	Gamut-mapper Status Register
GM_REGION0	0x408	R/W	0x00000000	Gamut-mapper Region 0 Register
GM_REGION1	0x40C	R/W	0x00000000	Gamut-mapper Region 1 Register
GM_INT	0x41C	R/W	0x00000000	Gamut-mapper Interrupt Register

Name	Address	Type	Reset	Description
HI_CTRL	0x600	R/W	0x00000000	Histogram Control Register
HI_STATUS	0x604	R/W	0x00000000	Histogram Status Register
HI_CONFIG	0x608	R/W	0x00000000	Histogram Configuration Register
HI_REGION0	0x60C	R/W	0x00000000	Histogram Region 0 Register
HI_REGION1	0x610	R/W	0x00000000	Histogram Region 1 Register
HI_INT	0x61C	R/W	0x00000000	Histogram Interrupt Register
HI_SEGS	0x620 ~ 0x62C	R/W	0x00000000	Histogram Segments Register
HI_CDFS	0x630 ~ 0x63C	R	0x00000000	Histogram CDF Register
HI_CNTS	0x640 ~ 0x65C	R	0x00000000	Histogram CNT Register
HI_SCALE	0x660 ~ 0x66C	R/W	0x00000000	Histogram Scale Register
HI_LUTS	0x700 ~ 0x7FC	R/W	0x00000000	Histogram LUT Table Register

CTRL (Control Register) 0xB0A52000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
											FMTDIS	HILUT	HIEN	GMEN	OEN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
M2AD	M2	FDI	DISP		HPFMODE	DEMODE		RCMEN	DN2EN	DN3EN	DIEN	REN2	REN1	REN0	RGEN

Field	Name	RW	Reset	Description
20	FMTDIS	R/W	0	- **Image Format Automatic Conversion Disable** 1'b0 = Auto Conversion is Enable by default. If the LCDC direct path is used, the output image format should be 4:2:2 format. This bit by default enables to make the output image format 4:2:2 format. 1'b1 = disable
19	HILUT	R/W	0	- **Histogram CDF or LUT Enable** 1'b0 = disable 1'b1 = This bit enables the main PATH Controller to check the status of Histogram Generator in order to dynamically insert the Histogram Generator into pipeline path. When this bit is only enabled, the interpolation engine or LUT in Histogram Generator can be used.
18	HIEN	R/W	0	- **Histogram Generator Enable** 1'b0 = disable 1'b1 = enable
17	GMEM	R/W	0	- **Gamut Mapper Enable** 1'b0 = disable 1'b1 = enable
16	OEN	R/W	0	- **OutDMA Enable** 1'b0 = disable 1'b1 = enable
15	M2AD	R/W	0	- **De-interlacer Two buffers Auto detection Disable** 1'b0 = Automatic Buffer usage 1'b1 = Forcedly, only two buffers are used in the de-interlacer
14	M2	R/W	0	- **De-interlacer Two buffers Mode** 1'b0 = Forcedly, two buffers mode is not used. It is meaningful only when M2AD is high. 1'b1 = Forcedly, two buffers mode is used. It is meaningful only when M2AD is high.
13	FDI	R/W	0	- **De-interlacer Order Priority** 1'b0 = De-interlacer processing shall be done after De-noising processing. 1'b1 = De-interlacer operates using only current frame information
12	DISP	R/W	0	- **Spatial De-interlacer Only Mode** 1'b0 = De-interlacer operates using both current frame and previous frame information 1'b1 = De-interlacer operates using only current frame information
10	HPFMODE	R/W	0	- **HighPass Filter Mode** 1'b0 = Lowpass Filter Mode 1'b1 = Highpass Filter Mode Spatial de-noiser supports high-pass filter and low-pass filter. If the use want to use the spatial de-noiser, this field should be selected.
9-8	DEMODE	R/W	0	- **Denoise Mode** 2'b00 = Use only temporal de-noiser 2'b01 = Use only spatial de-noiser 2'b1x = Use both of temporal and spatial de-noiser. In this mode, temporal de-noiser is low-pass filter, and spatial de-noiser is high-pass filter.

Field	Name	RW	Reset	Description
7	RCMEM	R/W	0	- **Recursive Pixel Compressor Enable** 1'b0 = disable 1'b1 = This bit turns compressor/decompressor on for recursive pixel data. This feature is meaningful only when RDMA3 port is enabled and the temporal de-noiser is enabled.
6	DN2EN	R/W	0	- **Spatial De-noiser Enable** 1'b0 = disable 1'b1 = enable
5	DN3EN	R/W	0	- **Temporal De-noiser Enable** 1'b0 = disable 1'b1 = enable
4	DIEN	R/W	0	- **De-interlacer Enable** 1'b0 = disable 1'b1 = enable
3	REN2	R/W	0	- **RDMA 2 Enable** 1'b0 = disable 1'b1 = This bit starts the RDMA2 to transfer the frame data. This bit is meaningless unless RGEN bit is enabled. This channel 2 of RDMA transfers the compressed differential data of previous filtered frame and previous frame. This bit should be set when the temporal de-noiser mode is on.
2	REN1	R/W	0	- **RDMA 1 Enable** 1'b0 = disable 1'b1 = This bit starts the RDMA1 to transfer the frame data. This bit is meaningless unless RGEN bit is enabled. This channel 1 of RDMA transfers the previous frame. This field should be set when the temporal mode of either de-noiser or de-interlacer is on.
1	REN0	R/W	0	- **RDMA 0 Enable** 1'b0 = disable 1'b1 = This bit starts the RDMA0 to transfer the frame data. This bit is meaningless unless RGEN bit is enabled. This channel 0 of RDMA should be enabled when the VIQE is operated
0	RGEN	R/W	0	- **RDMAGlobal Enable** 1'b0 = disable 1'b1 = enable

SIZE (Image Size Register) **0xB0A52004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					HEIGHT										
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					WIDTH										

Field	Name	RW	Reset	Description
26-16	HEIGHT	R/W	0	- **Image Height** Input image height by pixel
10-0	WIDTH	R/W	0	- **Image Width** Input image width by pixel

TIMEGEN (Time Generator Register) **0xB0A52008**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								H2H							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				H2V				V2H				P2P			

Field	Name	RW	Reset	Description
23-16	H2H	R/W	0x16	-**Pixel Count of Line-end to Line-start** This field is the pixel count between the ending pixel of the line and the first pixel of the next line. Refer to Figure 9.9.
11-8	H2V	R/W	2	- **Line Count of Line-end to Frame-start** This field is the line count between the ending of the last line and the starting of the next frame. Refer to Figure 9.9.
7-4	V2H	R/W	1	- **Line Count of Frame-start to Line-start** This field is the line count between the starting of the frame and the starting of the first line. Refer to Figure 9.9.
3-0	P2P	R/W	0	- **Clock Count of Current to Next-pixel** This field is the cycle count between the valid pixels. Refer to Figure 9.9.

The RDMA transfers the frame data to the processing modules such as de-noiser, de-interlacer, and histogram using the interface similar to CCIR streaming format for on-the-fly operation. This register is used as the configuration of interface timing.

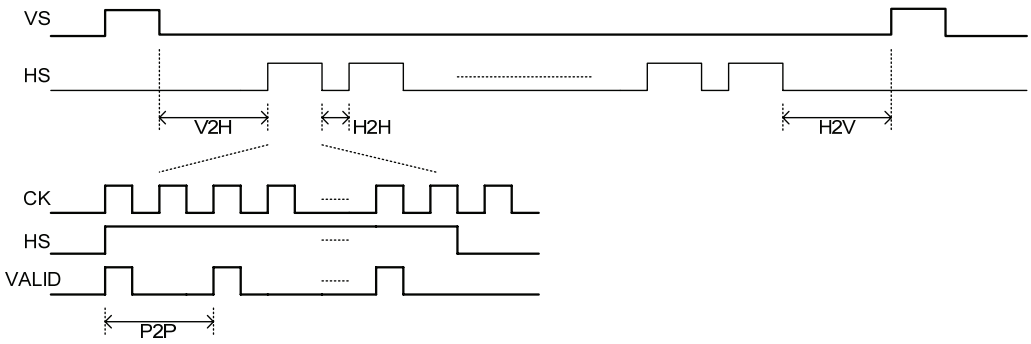


Figure 9.9 Timing Diagram in internal operating

LUMADLY (Luminance Delay Register) **0xB0A5200C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												RDDLY		DNDLY	

Field	Name	RW	Reset	Description
3-2	RDDLY	R/W	0	- **Luma Delay Value for RDMA** Luminance data delay value from output chrominance data.
1-0	WIDTH	R/W	0	- **Luma Delay Value for Denoiser** Luminance data delay value from output chrominance data.

Each module has the delay factor for filtering method. For example, spatial de-noiser needs one line delay, de-interlacer needs two line delays. But if the image format is 4:2:0, the output data delay of luminance and chrominance are different because chrominance data is processed once in either of odd or even line. So, the VIQE could not be operated in the on-the-fly mode because of this limitation. For solving this problem, the module is fed on the chrominance data in one to three lines beforehand. Refer to Figure 9.10.

- RDDLY value is 1 : only use the spatial de-noiser
- RDDLY value is 2 : use the de-interlacer or the composed de-interlaces and temporal de-noiser
- RDDLY value is 3 : user the composed de-interlacer and spatial de-noiser.
- DNDLY : DNDLY is the same value in RDDLY, but DNDLY is '1' when the spatial de-noiser is inputted from output data of de-intelacer.

If the image format is not 4:2:0, this register does not need to be set to '1'.
If FMTDIS field of CTRL register is set HIGH, this delay values are set automatically.

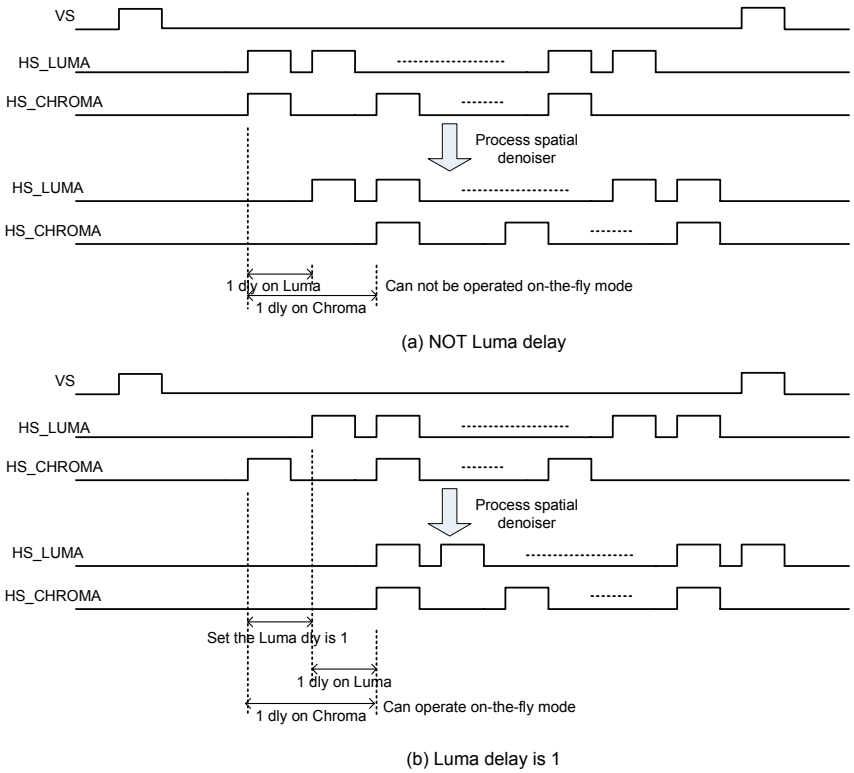


Figure 9.10 Timing Diagram in Luminance Delay Register

IMGCONF (Image Configuration Register) 0xB0A52010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ODDF

Field	Name	RW	Reset	Description
0	ODDF	R/W	0	- **Odd Field First Indicator for De-interlacer** 1'b0 = Even field first line in interlaced frame. 1'b1 = odd field first line in interlaced frame.

This register should be set correctly. Otherwise, de-interlacing output should be deteriorated seriously. This register have an influence on only de-interlacing.

IMGFMT (Image Format Register) 0xB0A52014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FPF	CRT	CNF		IMGFMT		

Field	Name	RW	Reset	Description
6	FPF	R/W	0	- **First Previous Frame** 1'b0 = The first previous reference frame is read from the memory buffer described in RDMA1 address register. 1'b1 = The first previous reference frame is copied from the first current frame.
5	CRT	R/W	0	- **Chroma Line Read Twice** 1'b0 = Chroma Lines are read just once. 1'b1 = Chroma Lines are read twice in 4:2:0 format. This bit makes the input image in 4:2:0 format to be manipulated as 4:2:2 input format.
4	CNF	R/W	0	- **Chroma Not First Line** 1'b0 = Chroma Valid Line is in first luminance line in 4:2:0 format. 1'b1 = Chroma Valid Line is in second luminance line in 4:2:0 format.
3-0	IMGFMT	R/W	0	- **Image Format** 4'b0000 = YUV4:2:0 separate mode 4'b0001 = YUV4:2:2 separate mode 4'b0010 = YUV4:2:2 sequence mode. This stored image format if V-Y1-U-Y0(MSB-LSB sequence) in word unit 4'b0011 = NOT USED 4'b0100 = YUV4:2:0 interleave 0 mode 4'b0101 = YUV4:2:0 interleave 1 mode 4'b0110 = YUV4:2:2 interleave 0 mode 4'b0111 = YUV4:2:2 interleave 1 mode Others = NOT USED

In interleave mode, the stored image is separated into two channels – luminance(Y) and chrominance(C).
Luminance format is Y3-Y2-Y1-Y0 (MSB-LSB sequence) in a word unit.
Chrominance format is V2-U2-V0-U0 (MSB-LSB sequence) in interleave0 mode, but in the other mode, U2-V2-U0-V0 (MSB-LSB sequence).

MISCC (Miscellaneous Control Register) 0xB0A52018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIFOF			RESERVED					OVLDIS	RESERVED						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															FRMUPM

Field	Name	RW	Reset	Description
31-29	FIFOF	R/W	0	- **Internal FIFOs Flush Control** Bit[31] = This bit is the de-interlacer Input FIFO Flush control. When it is set, the FIFO shall be flushed. Bit[30] = This bit is the Luma Delayer Input FIFO Flush control. When it is set, the FIFO shall be flushed. Bit[29] = This bit is the Format Converter FIFO Flush Control. When it is set, the FIFO shall be flushed.
23	OVLDIS	R/W	0	- **Overlapped Block Processing Disable** 1'b0 = All Blocks can be processed overlappedly each other. 1'b1 = All Blocks cannot be processed overlappedly each other or, each block processing shall wait until other block processing is done If this bit is enabled, the processing performance may be declined slightly.
0	FRMUPM	R/W	0	- **Frame Restart Register Manual Update** 1'b0 = Frame Restart Registers is updated automatically. In other words, the next frame processing starts immediately after the current frame processing is done. 1'b1 = Frame Registers are updated manually. Only FRMRST bit in FRMCTRL register can make restart the VIQE processing. This bit can be set even during the current frame processing, and after the current frame is done, the next processing shall starts immediately. The VIQE has the temporal register groups for restoring the configuration changes not to be effected during the current frame processing.

FRMCTRL (Frame Control Register) 0xB0A5201C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															FRMRST

Field	Name	RW	Reset	Description
0	FRMRST	R/W	0	- **Frame Restart Register** 1'b0 = Writing '0' is meaningless. 1'b1 = VIQE starts to process the next frame data.

INTMASK (Interrupt Mask Register) 0xB0A52020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRREG				RESERVED						PM	ODMA	WDMA	DI	DN	RDMA

Field	Name	RW	Reset	Description
15-12	ERRREG	R/W	0	- **Register Setting Error Interrupt Mask** Bit[15] = The interrupt mask of the reference frame HPF_DI_FIRST setting error. This happens when the HPF mode and DI_FIRST mode are simultaneously enabled. Bit[14] = The interrupt mask of the reference frame DI_DNS setting error. This happens when the de-interlacer and spatial-only de-noiser mode are simultaneously enabled. Bit[13] = The interrupt mask of the reference frame RDMA3 setting error. This happens when the de-noiser is enabled but either the RDMA3 or the WDMA is not simultaneously enabled. Bit[12] = The interrupt mask of the reference frame RDMA2 setting error. This happens when the reference frame data is needed – either temporal de-interlacer or temporal de-noiser mode is enabled, the RDMA2 is not enabled.
5	PM	R/W	0	- **Histogram/Gamut Mapper Interrupt Mask** 1'b0 = disable. 1'b1 = enable.
4	ODMA	R/W	0	- **Output DMA Interrupt Mask** 1'b0 = disable 1'b1 = enable
3	WDMA	R/W	0	- **Write DMA Interrupt Mask** 1'b0 = disable 1'b1 = enable
2	DI	R/W	0	**- De-interlacer Interrupt Mask** 1'b0 = disable 1'b1 = enable
1	DN	R/W	0	**- De-noiser Interrupt Mask** 1'b0 = disable 1'b1 = enable
0	RDMA	R/W	0	**- Read DMA Interrupt Mask** 1'b0 = disable 1'b1 = enable

The interrupt of each module is described in the register fields of each module.
Actually, this register is checked or used only for where the interrupt comes from. The interrupt processing such as clearing interrupts should be checked in each module register.

INT (Interrupt Register) **0xB0A52024**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERRREG				RESERVED						PM	ODMA	WDMA	DI	DN	RDMA

Field	Name	RW	Reset	Description
15-12	ERRREG	R/W	0	- **Register Setting Error Interrupt** Bit[15] = The interrupt mask of the reference frame HPF_DI_FIRST setting error. This happens when the HPF mode and DI_FIRST mode are simultaneously enabled. Bit[14] = The interrupt mask of the reference frame DI_DNS setting error. This happens when the de-interlacer and spatial-only de-noiser mode are simultaneously enabled. Bit[13] = The interrupt mask of the reference frame RDMA3 setting error. This happens when the de-noiser is enabled but either the RDMA3 or the WDMA is not simultaneously enabled. Bit[12] = The interrupt mask of the reference frame RDMA2 setting error. This happens when the reference frame data is needed – either temporal de-interlacer or temporal de-noiser mode is enabled, the RDMA2 is not enabled.
5	PM	R/W	0	- **Histogram/Gamut Mapper Interrupt** 1'b0 = disable. 1'b1 = An interrupt from Histogram Generator /Gamut Mapper.
4	ODMA	R/W	0	- **Output DMA Interrupt** 1'b0 = disable. 1'b1 = An interrupt from Output DMA
3	WDMA	R/W	0	- **Write DMA Interrupt** 1'b0 = disable. 1'b1 = An interrupt from Write DMA
2	DI	R/W	0	**- De-interlacer Interrupt** 1'b0 = disable. 1'b1 = An interrupt from De-interlacer.
1	DN	R/W	0	**- De-noiser Interrupt** 1'b0 = disable 1'b1 = An interrupt from De-noiser
0	RDMA	R/W	0	**- Read DMA Interrupt** 1'b0 = disable 1'b1 = An interrupt from Read DMA.

In above all interrupt register fields, reading '1' means the occurrence of the according interrupt, and writing '1' means the clearing bit of the according interrupt

DI_CTRL (De-interlacer Control Register) **0xB0A52080**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLRIR	FLSFF	BYPASS	RESERVED												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDCLFI				PDPF		PDJUD	PDEN	YDM		MRSP	MRTM		JT	JSP	JS

Field	Name	RW	Reset	Description
31	CLRIR	R/W	0	- **Clear Internal Registers** 1'b0 = No meaning 1'b1 = Clear Internal Registers in De-interlacer
30	FLSFF	R/W	0	- **Flush Synchronizer FIFO** 1'b0 = No meaning 1'b1 = Flush Synchronizer FIFO
29	BYPASS	R/W	0	- **Bypass Register** 1'b0 = Normal process 1'b1 De-interlacer Bypass Enable. If this bit is set,

Field	Name	RW	Reset	Description
				the input data to the de-interlacer will be intact through hardware block.
15	PDCLFI	R/W	0	- **Clear Internal Frame Index Register** 1'b0 = Normal process 1'b1 = Clear Internal Frame Index Register in Pulldown Detector
11	PDPF	R/W	0	**- Use Prevention-Flag in Pulldown Detector** 1'b0 = Use Prevention-Flag algorithm in Pulldown Detector. The Prevention-Flag algorithm can reduce the possibility of miss-detection in film mode checking. 1'b1 = Don't use Prevention-Flag Algorithm.
9	PDJUD	R/W	0	**- User Judder Detection in Pulldown Detector** 1'b0 = Don't use judder detection in Pulldown detector. 1'b1 = Use judder detection in Pulldown detector. The judder in the reverse pull-down processing means the artifacts from bad-edit or flowing text that be edited additionally at the normal frame rate. These artifacts can be removed using the judder detection algorithm in the VIQE.
8	PDEN	R/W	0	**- Pulldown Detector Enable** 1'b0 = Pulldown detector is disable. 1'b1 = Pulldown detector is enable. Pulldown detector can be used for detecting the film source, which is transferred into digital format using 3:2 pull-down method.
7	YDM	R/W	0	**- YD Mode** 1'b0 = disable 1'b1 = enable
6	GTHSJ	R/W	0	**- Generate Jaggy Checker Threshold** 1'b0 = Use only pre-defined threshold for jaggy checker. 1'b1 = Use generated threshold for jaggy checker. This is optional feature that the thresholds for jaggy checker can be generated adaptively by the specific hardware. This feature is recommended strongly.
5	MRSP	R/W	0	**- Pixel Output Range Enable in Spatial Processing** 1'b0 = Don't use output range limitation algorithm. 1'b1 = Use output range limitation algorithm. This option can be used only when the spatial de-interlacer only mode(DISP bit in CTRL register) is off.
4	MRTM	R/W	0	**- Pixel Output Range Enable in Temporal Processing** 1'b0 = Don't us output range limitation algorithm. 1'b1 = Use output range limitation algorithm. This option can be used only when the spatial de-interlacer only mode(DISP bit in CTRL register) is off.
2	JT	R/W	0	**- Temporal Jaggycheck** 1'b0 = disable 1'b1 = enable
1	JSP	R/W	0	**- Temporal Jaggycheck** 1'b0 = disable 1'b1 = enable
0	JS	R/W	0	**- Spatio Jaggycheck** 1'b0 = disable 1'b1 = enable

DI_ENGINE0 (De-interlacer Engine 0 Register) 0xB0A52084

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMTPXLC								DMTSADC							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DMTPXL								DMTSAD							

Field	Name	RW	Reset	Description
31-24	DMTSADC	R/W	0	- **SAD Threshold in Motion Detection for Chroma** Threshold for SAD value in Motion Detection for Chrominance
23-16	DMTPXLC	R/W	0	- **Pixel Threshold in Motion Detection for Chroma** Threshold for Pixel value in Motion Detection for Chrominance
15-8	DMTPXL	R/W	0	- **Pixel Threshold in Motion Detection** Threshold for Pixel value in Motion Detection for Luminance
7-0	DMTSAD	R/W	0	- **SAD Threshold in Motion Detection** Threshold for SAD value in Motion Detection for Luminance

DI_ENGINE1 (De-interlacer Engine 1 Register) 0xB0A52088

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
THSJMAX								THSJMIN							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LRT		LRL					LRD	DIRLT				GLD	JDH		

Field	Name	RW	Reset	Description
31-24	THSJMAX	R/W	0	- **Maximum Threshold in Jaggy Threshold Generator** This value means the maximum boundary for the jaggy threshold generator.
23-16	THSJMIN	R/W	0	- **Minimum Threshold in Jaggy Threshold Generator** This value means the minimum boundary for the jaggy threshold generator.
15-14	LRT	R/W	0	- **Horizontal Reference Type** Horizontal Reference Type Normally, it should be zero.
13-9	LRL	R/W	2	- **Horizontal Reference Length** Horizontal Reference Length Normally, it should be 2
8	LRD	R/W	0	- **Horizontal Reference Disable** 1'b0 = enable 1'b1 = disable
7-4	DIRLT	R/W	0	- **Spatial Edge Direction Detector Length Type** Spatial Edge Direction Detector Length Type Normally, it should be zero.
3	GLD	R/W	0	- **Small Angle Detection Disable** 1'b0 = enable 1'b1 = disable
2	JDH	R/W	0	- **Jaggy Detection Half Divider** 1'b0 = Jaggy Detection Half Divider Off 1'b1 = Jaggy Detection Half Divider On

DI_ENGINE2 (De-interlacer Engine 2 Register) 0xB0A5208C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
A_THS												E_THS			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
E_THS								EP2	EP1				OPPD	EARLYD	ADAPD

Field	Name	RW	Reset	Description
31-20	A_THS	R/W	0	- **Threshold of Early determination** This value indicates threshold value which determines whether current edge directional result is ignored compared as vertical direction or not.
19-8	E-THS	R/W	0	- **Threshold of Adaptive Edge determination** This value indicates threshold value which determines whether current edge directional result is acceptable compared as other directional cost values.
7	EP2	R/W	0	- **Early determination Suppress Algorithm 2 Disable** 1'b0 = suppress algorithm 2 enable 1'b1 = suppress algorithm 2 disable
6	EP1	R/W	2	- **Early determination Suppress Algorithm 1 Disable** 1'b0 = suppress algorithm 1 enable 1'b1 = suppress algorithm 1 disable
2	OPPD	R/W	0	- **Edge adaptive direction determination Disable** 1'b0 = enable 1'b1 = disable
1	EARLYD	R/W	0	- **Early determination Disable** 1'b0 = enable 1'b1 = disable
0	ADAPD	R/W	0	- **Adaptive Edge determination Disable** 1'b0 = enable 1'b1 = disable

DI_ENGINE3 (De-interlacer Engine 3 Register) 0xB0A52090

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
STTHW												STTHM			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
STTHM															STTHD

Field	Name	RW	Reset	Description
31-20	STTHW	R/W	0	- **Threshold Weight Parameter of Stationary Detection** Threshold Weight Parameter of Stationary Detection Increasing one bit means 0.5 weight addition.
19-8	STTHM	R/W	0	- **Threshold Multiplier Parameter of Stationary Detection** Threshold Multiplier Parameter of Stationary Detection Increasing one bit means 0.5 multiplier addition
0	STTHD	R/W	0	- **Adaptive Stationary Checker Disable** 1'b0 = Use adaptive stationary checker algorithm. The stationary checker in the VIQE does not determine only whether this pixel is stationary or not, but also analyzes the degree of stationarity quantitatively. This value can be used in the compensating the outputs. These procedures shall process in the adaptive stationary checker hardware. 1'b1 = Don't use adaptive stationary checker algorithm

DI_ENGINE4 (De-interlacer Engine 4 Register) **0xB0A52094**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VARLG				VARR1				VARR0				DYNVAR			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HPFVAR				HPFFLD				HPFFRM						DMDC	DMDL

Field	Name	RW	Reset	Description
31-28	VARLG	R/W	0	- **Variance small stationary range** Variance small stationary range.
27-24	VARR1	R/W	0	- **Variance dynamic threshold ratio 1** Variance dynamic threshold ratio 1
23-20	VARR0	R/W	0	- **Variance dynamic threshold ratio 0** Variance dynamic threshold ratio 0
19-16	DYNVAR	R/W	0	- **Derivative Variance Minimum Threshold** Derivative Variance Minimum Threshold
15-12	HPFVAR	R/W	0	- **Derivative Variance Threshold for HPF output** Derivative Variance Threshold for HPF output
11-8	HPFFLD	R/W	0	- **HPF Threshold of Field Difference** HPF Threshold of Field Difference
7-4	HPFFRM	R/W	0	- **HPF Threshold of Frame Difference** HPF Threshold of Frame Difference
1	DMDC	R/W	0	- **Stationary Checker Mode in Chrominance** 1'b0 = Use advanced stationary checker algorithm in chrominance processing. 1'b1 = Don't use advanced stationary checker algorithm in chrominance processing
0	DMDL	R/W	0	- **Stationary Checker Mode in Luminance** 1'b0 = Use advanced stationary checker algorithm in luminance processing. 1'b1 = Don't use advanced stationary checker algorithm in luminance processing.

*) All bits in DI_ENGINE4 register are meaningful only when either DMDC or DMDL is zero.

PD_THRES0 (Pulldown Detector Threshold 0 Register) 0xB0A52098

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
							ZP	CNTSCO						CO	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UVALDIS	WEIGHT					CNTS									

Field	Name	RW	Reset	Description
24	ZP	R/W	0	- **Avoidance of zero difference** 1'b0 = Do not use the avoidance of zero frame difference. 1'b1 = Use the avoidance of zero frame difference. It is recommended to use.
23-20	CNTSCO	R/W	0	- **Threshold for counter value of Pulldown Prevention-Flag** Threshold for counter value of pulldown Prevention-Flag algorithm
17-16	CO	R/W	0	**- Value of Pulldown Detection type** 2'b00 = Not interleaving, but normal de-interlacing process 2'b01 = Reserved 2'b10 = Interleaving fields from the previous fields 2'b11 = Interleaving fields from the next fields Through this field, the user can control manually the method of interleaving in the reverse pull-down process.
15	UVALDIS	R/W	0	- **Not Use User-defined Threshold Value in Prevention-Flag** 1'b0 = Use an user-defined threshold value 1'b1 = Use hardware-generated threshold value Its range can be controlled by WEIGHT field in PD_THRES0 register.
14-10	WEIGHT	R/W	0	- **Weight Value for Threshold of Pulldown Prevention-Flag** This means the ratio of relative amount between minimum detection value and the second smaller detection value. This bit is meaningless when UVALDIS is '0'.
9-0	CNTS	R/W	0	- **Threshold for counter value of Pulldown checker** Threshold for counter value of pulldown checker If the number of continuous success of pulldown detection is over that threshold, field interleaving operates instead of de-interlacing.

PD_THRES1 (Pulldown Detector Threshold 1 Register) 0xB0A5209C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								THRES2							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
THRES1								THRES0							

Field	Name	RW	Reset	Description
23-16	THRES2	R/W	0	- **Threshold for pixel difference value of third level** This means the threshold value for pixel difference between the current frame and the previous frame.
15-8	THRES1	R/W	0	- **Threshold for pixel difference value of second level** This means the threshold value for pixel difference between the current frame and the previous frame.
7-0	THRES0	R/W	0	**- Threshold for pixel difference value of first level** This means the threshold value for pixel difference between the current frame and the previous frame.

In normal configuration, threshold value of a level should be larger than that of lower level. Or, threshold value of the first level is smallest of all, and that of the third level is largest.

PD_JUDDER (Pulldown Detector Judder Register) 0xB0A520A0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
THSJDMAX								THSJDMIN							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HORLINE				DNLINE				CNTS							

Field	Name	RW	Reset	Description
31-24	THSJDMAX	R/W	0	- **Threshold maximum boundary for Pulldown Judder Detector** This means the maximum boundary of hardware-generated threshold for Pulldown Judder Detector
23-16	THSJDMIN	R/W	0	- **Threshold minimum boundary for Pulldown Judder Detector** This means the minimum boundary of hardware-generated threshold for Pulldown Judder Detector
15-12	HORLINE	R/W	0	**- Horizontal Margin for Judder Elimination Processing** This means the left/right margin of the lines detected as having some judder pixels. If its value is larger, the negative effect for judder miss-detection is less.
11-8	DNLINE	R/W	0	**- Downward Margin for Judder Elimination Processor** This means the downward margin of the lines detected as having some judder pixels. If its value is larger, the negative effect for judder miss-detection is less.
7-0	CNTS	R/W	0	**- Threshold for counter of judder pixels** This value determines whether the current line is judder line or not. If the pixels detected as judder pixels are more of that value, the current line is determined as the judder line.

PD_JUDDER_M (Pulldown Detector Judder Misc. Register) 0xB0A520A4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								MULDMT							NOJDS
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
THSJD2B															JDH

Field	Name	RW	Reset	Description
23-17	MULDMT	R/W	0	- **Multiplier Stationary Checker Threshold** Multiplier Stationary Checker Threshold for stationary pixels during Film-mode processing
16	NOJDS	R/W	0	- **Prevent to count stationary pixels** 1'b0 = Count judder pixels to be even stationary 1'b1 = Not count judder pixels to be stationary
15-8	THSJDS2	R/W	0	**- Threshold Pulldown judder for Two buffers mode** Threshold Pulldown judder for Two buffers mode.
0	JDH	R/W	0	**- Judder Detection Half Divider** 1'b0 = Judder Detection Half Divider Off 1'b1 = Judder Detection Half Divider On

DI_MISCC (Deinterlace Misc. Register) 0xB0A000A8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				INIT_L2L				Y_MST_JMP		X_MST_END					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IDX_Y_MST								IDX_X_MST							

*** This register is TEST mode. DO NOT access this register.**

DI_STATUS (De-interlacer Status Register) 0xB0A520AC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		POS_Y											POS_X		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POS_X															BUSY

Field	Name	RW	Reset	Description
29-19	POS_Y	R	0	- **Current Vertical Position** This indicates the vertical position in image to be processed currently. This position value is based on luminance resolution.
18-8	POS_X	R	0	- **Current Horizontal Position** This indicates the horizontal position in image to be processed currently. This position value is based on luminance resolution.
0	BUSY	R	0	**- De-interlacer Busy** This indicates that the De-interlacer hardware is processing currently.

PD_STATUS (Pulldown Detector Status Register) 0xB0A520B0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					MPOFF					CO		STATE			

Field	Name	RW	Reset	Description
10-8	MPOFF	R	0	- **Minimum Position Offset** This indicates the information to know the minimum frame position in current operation. This field is used for hardware debugging.
5-4	CO	R	0	- **Value of Pulldown Detection type** 2'b00 = Not interleaving, but normal de-interlacing process 2'b01 = Reserved 2'b10 = Interleaving fields from the previous fields 2'b11 = Interleaving field from the next fields
3-0	STATE	R	0	**- Current FSM State** This indicates the current state of the hardware FSM. This field is used for hardware debugging.

DI_REGION0 (De-interlacer Region 0 Register) 0xB0A520B4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN						XEND									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						XSTART									

Field	Name	RW	Reset	Description
31	EN	R/W	0	- **Region Selection Enable** 1'b0 = Normal Process 1'b1 = Only specific region of the whole image will be processed in de-interlacing
25-16	XEND	R/W	0	- **Last Horizontal Position of Region Selection** This means the last horizontal position of region selection.
9-0	XSTART	R/W	0	**- First Horizontal Position of Region Selection** This means the first horizontal position of region selection.

DI_REGION1 (De-interlacer Region 1 Register) 0xB0A520B8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
						YEND									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						YSTART									

Field	Name	RW	Reset	Description
25-16	YEND	R/W	0	- **Last Vertical Position of Region Selection** This means the last vertical position of region selection.
9-0	YSTART	R/W	0	- **First Vertical Position of Region Selection** This means the first vertical position of region selection.

DN_C_H_Y0 (Denoise-3D Coefficient Horizontal Luminance #0) 0xB0A52100

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HY15		HY14		HY13		HY12		HY11		HY10		HY9		HY8	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HY7		HY6		HY5		HY4		HY3		HY2		HY1		HY0	

Field	Name	RW	Reset	Description
31-0	HY#	R/W	0xbfffffa4	- **Horizontal filter coefficient in Luminance** Filter coefficient in denoise-3D # : coefficient number (0~15) DO NOT set this register. This register is set automatically firmwire code.

DN_C_H_Y1 (Denoise-3D Coefficient Horizontal Luminance #1) 0xB0A52104

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HY31		HY30		HY29		HY28		HY27		HY26		HY25		HY24	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HY23		HY22		HY21		HY20		HY19		HY18		HY17		HY16	

Field	Name	RW	Reset	Description
31-0	HY#	R/W	0x15556aaa	- **Horizontal filter coefficient in Luminance** Filter coefficient in denoise-3D # : coefficient number (16~31) DO NOT set this register. This register is set automatically firmwire code.

DN_C_V_Y0 (Denoise-3D Coefficient Vertical Luminance #0) 0xB0A52108

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VY15		VY14		VY13		VY12		VY11		VY10		VY9		VY8	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VY7		VY6		VY5		VY4		VY3		HY2		VY1		VY0	

Field	Name	RW	Reset	Description
31-0	VY#	R/W	0xaaaaaaa4	- **Vertical filter coefficient in Luminance** Filter coefficient in denoise-3D # : coefficient number (0~15) DO NOT set this register. This register is set automatically firmwire code.

DN_C_V_Y1 (Denoise-3D Coefficient Vertical Luminance #1) 0xB0A5210C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VY31		VY30		VY29		VY28		VY27		VY26		VY25		VY24	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VY23		VY22		VY21		VY20		VY19		HY18		VY17		VY16	

Field	Name	RW	Reset	Description
31-0	VY#	R/W	0x15556aaa	- **Vertical filter coefficient in Luminance** Filter coefficient in denoise-3D # : coefficient number (16~31) DO NOT set this register. This register is set automatically firmwire code.

DN_C_T_Y0 (Denoise-3D Coefficient Temporal Luminance #0) 0xB0A52110

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TY15		TY14		TY13		TY12		TY11		TY10		TY9		TY8	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TY7		TY6		TY5		TY4		TY3		TY2		TY1		TY0	

Field	Name	RW	Reset	Description
31-0	TY#	R/W	0xaaaaaaa4	- **Temporal filter coefficient in Luminance** Filter coefficient in denoise-3D # : coefficient number (0~15) DO NOT set this register. This register is set automatically firmwire code.

DN_C_T_Y1 (Denoise-3D Coefficient Temporal Luminance #1) 0xB0A52114

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TY31		TY30		TY29		TY28		TY27		TY26		TY25		TY24	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TY23		TY22		TY21		TY20		TY19		TY18		TY17		TY16	

Field	Name	RW	Reset	Description
31-0	TY#	R/W	0x0x15556aaa	- **Temporal filter coefficient in Luminance** Filter coefficient in denoise-3D # : coefficient number (16~31) DO NOT set this register. This register is set automatically firmwire code.

DN_C_H_C0 (Denoise-3D Coefficient Horizontal Chrominance #0) 0xB0A52118

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HY15		HY14		HY13		HY12		HY11		HY10		HY9		HY8	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HY7		HY6		HY5		HY4		HY3		HY2		HY1		HY0	

Field	Name	RW	Reset	Description
31-0	HC#	R/W	0xbfffffa4	- **Horizontal filter coefficient in Chrominance** Filter coefficient in denoise-3D # : coefficient number (0~15) DO NOT set this register. This register is set automatically firmwire code.

DN_C_H_Y1 (Denoise-3D Coefficient Horizontal Chrominance #1) 0xB0A5211C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HY31		HY30		HY29		HY28		HY27		HY26		HY25		HY24	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HY23		HY22		HY21		HY20		HY19		HY18		HY17		HY16	

Field	Name	RW	Reset	Description
31-0	HC#	R/W	0xbfffffa4	- **Horizontal filter coefficient in Chrominance** Filter coefficient in denoise-3D # : coefficient number (16~31) DO NOT set this register. This register is set automatically firmwire code.

DN_C_V_C0 (Denoise-3D Coefficient Vertical Chrominance #0) 0xB0A52120

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VY15		VY14		VY13		VY12		VY11		VY10		VY9		VY8	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VY7		VY6		VY5		VY4		VY3		HY2		VY1		VY0	

Field	Name	RW	Reset	Description
31-0	VC#	R/W	0xaaaaaaa4	- **Vertical filter coefficient in Chrominance** Filter coefficient in denoise-3D # : coefficient number (0~15) DO NOT set this register. This register is set automatically firmwire code.

DN_C_V_C1 (Denoise-3D Coefficient Vertical Chrominance #1) 0xB0A52124

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VY31		VY30		VY29		VY28		VY27		VY26		VY25		VY24	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VY23		VY22		VY21		VY20		VY19		HY18		VY17		VY16	

Field	Name	RW	Reset	Description
31-0	VC#	R/W	0x15556aaa	- **Vertical filter coefficient in Chrominance** Filter coefficient in denoise-3D # : coefficient number (16~31) DO NOT set this register. This register is set automatically firmwire code.

DN_C_T_C0 (Denoise-3D Coefficient Temporal Chrominance #0) 0xB0A52128

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TY15		TY14		TY13		TY12		TY11		TY10		TY9		TY8	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TY7		TY6		TY5		TY4		TY3		TY2		TY1		TY0	

Field	Name	RW	Reset	Description
31-0	TC#	R/W	0xaaaaaaa4	- **Temporal filter coefficient in Chrominance** Filter coefficient in denoise-3D # : coefficient number (0~15) DO NOT set this register. This register is set automatically firmwire code.

DN_C_T_C1 (Denoise-3D Coefficient Temporal Chrominance #1) 0xB0A5212C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TY31		TY30		TY29		TY28		TY27		TY26		TY25		TY24	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TY23		TY22		TY21		TY20		TY19		TY18		TY17		TY16	

Field	Name	RW	Reset	Description
31-0	TC#	R/W	0x15556aaa	- **Temporal filter coefficient in Chrominance** Filter coefficient in denoise-3D # : coefficient number (16~31) DO NOT set this register. This register is set automatically firmwire code.

DN_STATES0_3D (Denoise-3D Count States #0) 0xB0A52130

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BY	DN	IM	0		LCNT_Y_3D										
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0					PCNT_Y_3D										

Field	Name	RW	Reset	Description
31	BY	R	0	- **Busy operation** 1'b0 = idle 1'b1 = busy
30	DN	R/W	0	- **Operation Done** 1'b0 = Normal 1'b1 = Operation Done. (If write '1', this field is clear.)
29	IM	R/W	0	- **Done Interrupt mask** 1'b0 = Masked 1'b1 = Unmasked
26-16	LCNT_Y_3D	R	-	- **Line Counter in Luminance** Line Counter in Luminance
10-0	PCN_Y_3D	R	-	**- Pixel Counter in Luminance** Pixel Counter in Luminance

DN_STATES1_3D (Denoise-3D Count States #1) 0xB0A52134

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					LCNT_C_3D										
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0					PCNT_C_3D										

Field	Name	RW	Reset	Description
26-16	LCNT_C	R	-	- **Line Counter in Chrominance** Line Counter in Chrominance
10-0	PCNT_C	R	-	- **Pixel Counter in Chrominance** Pixel Counter in Chrominance

DN_DIV_IMG_3D (Denoise-3D Divide image) **0xB0A52138**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DEN	DTOG	0			POS										

Field	Name	RW	Reset	Description
15	DEN	R/W	0	- **Image divide enable** 1'b0 = Normal 1'b1 = Divide image (filtered image – original image)
14	DTOG	R/W	0	- **Divided image swap** 1'b0 = Left image : filtered image Right image : original image 1'b1 = Left image : original image Right image : filtered image
10-0	POS	R/W	0	- **Divide Position** Divide Position

If DEN=1, Image is divided original and filtered image in Figure 9.11.

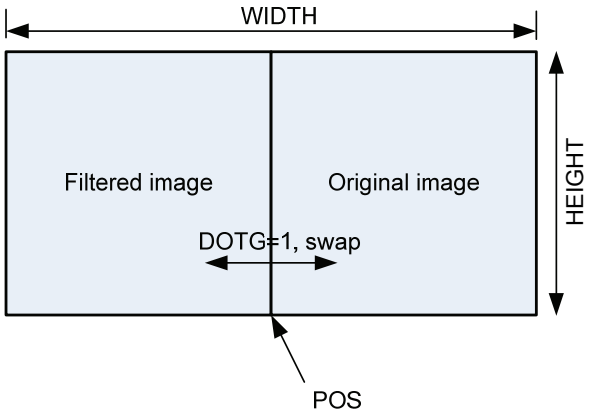


Figure 9.11 Image Divide to original and filtered image

DN_C_2D_Y0 (Denoise-2D Coefficient in Luminance #0) **0xB0A52140**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLOPE_Y								CA_Y							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA_Y								A_Y							

Field	Name	RW	Reset	Description
31-24	SLOPE_Y	R/W	0x12	- **Parameter SLOPE in Luminance data (LPF)** This value is set the converted sign of original value. The original value is always negative. (bit[7:6] is integer, the others is float) (default : **18** (-0.2857))
23-16	CA_Y	R/W	0x32	- **Multiplied parameter A and C in Luminance data (LPF)** Multiplied parameter A and C in Luminance data (LPF) (positive integer) (default : **50**)
15-8	BA_Y	R/W	0xf	- **Multiplied parameter A and B in Luminance data (LPF)** Multiplied parameter A and B in Luminance data (LPF) (positive integer) (default : **15**)
7-0	A_Y	R/W	0xa	- **Parameter A in Luminance data (LPF)** Parameter A in Luminance data (LPF) (positive integer) (default : **10**)

DO NOT set these registers. These registers are set automatically firmwire code.

DN_C_2D_Y1 (Denoise-2D Coefficient in Luminance #1) **0xB0A52144**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F_Y								E_Y							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D_Y								Y_PLANE_Y							

Field	Name	RW	Reset	Description
31-24	F_Y	R/W	0x37	- **Parameter F in Luminance data (HPF)** Parameter F in Luminance data (HPF) (positive integer value) (default : **55**)
23-16	E_Y	R/W	0x3C	- **Parameter E in Luminance data (HPF)** Parameter E in Luminance data (HPF) (positive integer value) (default : **60**)
15-8	D_Y	R/W	0x5	- **Parameter D in Luminance data (HPF)** (positive integer value) (default : **5**)
7-0	Y_PLANE_Y	R/W	0x12	**- Parameter Y-plane in Luminance data (LPF)** Parameter Y-plane in Luminance data (LPF) (positive integer) (default : **18**)

DO NOT set these registers. These registers are set automatically firmwire code.

DN_C_2D_Y2 (Denoise-2D Coefficient in Luminance #2) 0xB0A52148

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L_Y								M2_Y							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
M1_Y								G_Y							

Field	Name	RW	Reset	Description
31-24	L_Y	R/W	0x4a	- **Parameter L in Luminance data (HPF)** Parameter L in Luminance data (HPF) (positive integer value) (default : **74**)
23-16	M2_Y	R/W	0x6	- **Parameter M2 in Luminance channel (HPF)** Parameter M2 in Luminance channel (HPF) (bit[7:6] is integer, the others is float) (positive integer value) (default : **6** (0.1))
15-8	M1_Y	R/W	0x40	- **Parameter M2 in Luminance channel (HPF)** Parameter M2 in Luminance channel (HPF (bit[7:6] is integer, the others is float) (positive integer value) (default : **64** (1.0))
7-0	G_Y	R/W	0xff	**- Parameter G in Luminance data (HPF)** Parameter G in Luminance data (HPF) (positive integer value) (default : **255**)

DO NOT set these registers. These registers are set automatically firmwire code.

DN_C_2D_Y3 (Denoise-2D Coefficient in Luminance #3) 0xB0A5214C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							Y_PLANE_H2_Y								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0							Y_PLANE_H1_Y								

Field	Name	RW	Reset	Description
24-16	Y_PLANE_H2_Y	R/W	0x31	- **Parameter Y-plane in Luminance data (HPF)** Parameter Y-plane in Luminance data (HPF) This value is set the converted sign of original value. (positive/negative integer) (default : **49** (-49))
8-0	Y_PLANE_H1_Y	R/W	0x5	- **Parameter Y-plane in Luminance data (HPF)** Parameter Y-plane in Luminance data (HPF) This value is set the converted sign of original value. The original value is always positive. (positive integer) (default : **-5** (5))

DO NOT set these registers. These registers are set automatically firmwire code.

DN_C_2D_C0 (Denoise-2D Coefficient in Chrominance #0) **0xB0A52150**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SLOPE_C								CA_C							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BA_C								A_C							

Field	Name	RW	Reset	Description
31-24	SLOPE_C	R/W	0x12	- **Parameter SLOPE in Chrominance data (LPF)** This value is set the converted sign of original value. The original value is always negative. (bit[7:6] is integer, the others is float) (default : **18** (-0.2857))
23-16	CA_C	R/W	0x19	- **Multiplied parameter A and C in Chrominance data (LPF)** Multiplied parameter A and C in Chrominance data (LPF) (positive integer) (default : **25**)
15-8	BA_C	R/W	0x8	- **Multiplied parameter A and B in Chrominance data (LPF)** Multiplied parameter A and B in Chrominance data (LPF) (positive integer) (default : **8**)
7-0	A_C	R/W	0x5	- **Parameter A in Chrominance data (LPF)** Parameter A in Chrominance data (LPF) (positive integer) (default : **5**)

DO NOT set these registers. These registers are set automatically firmwire code.

DN_C_2D_C1 (Denoise-2D Coefficient in Chrominance #1) **0xB0A52154**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
F_C								E_C							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D_C								Y_PLANE_C							

Field	Name	RW	Reset	Description
31-24	F_C	R/W	0x37	- **Parameter F in Chrominance data (HPF)** Parameter F in Chrominance data (HPF) (positive integer value) (default : **55**)
23-16	E_C	R/W	0x3C	- **Parameter E in Chrominance data (HPF)** Parameter E in Chrominance data (HPF) (positive integer value) (default : **60**)
15-8	D_C	R/W	0x5	- **Parameter D in Chrominance data (HPF)** (positive integer value) (default : **5**)
7-0	Y_PLANE_C	R/W	0x7	**- Parameter Y-plane in Chrominance data (LPF)** Parameter Y-plane in Chrominance data (LPF) (positive integer) (default : **7**)

DO NOT set these registers. These registers are set automatically firmwire code.

DN_C_2D_C2 (Denoise-2D Coefficient in Chrominance #2) 0xB0A52158

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
L_C								M2_C							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
M1_C								G_C							

Field	Name	RW	Reset	Description
31-24	L_C	R/W	0x4a	- **Parameter L in Chrominance data (HPF)** Parameter L in Chrominance data (HPF) (positive integer value) (default : **74**)
23-16	M2_C	R/W	0x6	- **Parameter M2 in Chrominance channel (HPF)** Parameter M2 in Chrominance channel (HPF) (bit[7:6] is integer, the others is float) (positive integer value) (default : **6** (0.1))
15-8	M1_C	R/W	0x40	- **Parameter M2 in Chrominance channel (HPF)** Parameter M2 in Chrominance channel (HPF (bit[7:6] is integer, the others is float) (positive integer value) (default : **64** (1.0))
7-0	G_C	R/W	0xff	**- Parameter G in Chrominance data (HPF)** Parameter G in Chrominance data (HPF) (positive integer value) (default : **255**)

DO NOT set these registers. These registers are set automatically firmwire code.

DN_C_2D_C3 (Denoise-2D Coefficient in Chrominance #3) 0xB0A5215C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0							Y_PLANE_H2_C								
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0							Y_PLANE_H1_C								

Field	Name	RW	Reset	Description
24-16	Y_PLANE_H2_C	R/W	0x31	- **Parameter Y-plane in Chrominance data (HPF)** Parameter Y-plane in Chrominance data (HPF) This value is set the converted sign of original value. (positive/negative integer) (default : **49** (-49))
8-0	Y_PLANE_H1_C	R/W	0x5	- **Parameter Y-plane in Chrominance data (HPF)** Parameter Y-plane in Chrominance data (HPF) This value is set the converted sign of original value. The original value is always positive. (positive integer) (default : **-5** (5))

DO NOT set these registers. These registers are set automatically firmwire code.

DN_FIFOSTATE (Denoise FIFO States) **0xB0A52160**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				FIFOSTATE[23:12]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0				FIFOSTATE[11:00]											

Field	Name	RW	Reset	Description
27	FIFOSTATE[27]	R/W	0x0	- **FIFO overrun in denoise-3D** 1'b0 = normal 1'b1 = overrun
26-24	FIFOSTATE[26:24]	R	0x0	- **FIFO full states in denoise-3D** (each FIFOs are Y,U,V) 1'b0 = normal 1'b1 = full
23	FIFOSTATE[23]	R/W	0x0	- **FIFO overrun in denoise-2D (1st line buffer)** 1'b0 = normal 1'b1 = overrun
22-20	FIFOSTATE[22:20]	R	0x0	- **FIFO full states in denoise-2D (1st line buffers are Y,U,V)** 1'b0 = normal 1'b1 = full
19	FIFOSTATE[19]	R/W	0x0	- **FIFO overrun in denoise-2D (2nd line buffer)** 1'b0 = normal 1'b1 = overrun
18-16	FIFOSTATE[18:16]	R	0x0	- **FIFO full states in denoise-2D (2nd line buffers are Y,U,V)** 1'b0 = normal 1'b1 = full
11	FIFOSTATE[11]	R/W	0x0	- **FIFO underrun in denoise-3D** 1'b0 = normal 1'b1 = underrun
10-8	FIFOSTATE[10:8]	R	0x0	- **FIFO empty states in denoise-3D (rach FIFOs are Y,U,V)** 1'b0 = normal 1'b1 = empty
7	FIFOSTATE[7]	R/W	0x0	- **FIFO underrun in denoise-2D (1st line buffer)** 1'b0 = normal 1'b1 = underrun
6-4	FIFOSTATE[6:4]	R	0x0	- **FIFO empty states in denoise-2D (1st line buffers are Y,U,V)** 1'b0 = normal 1'b1 = empty
3	FIFOSTATE[3]	R/W	0x0	- **FIFO underrun in denoise-2D (2nd line buffer)** 1'b0 = normal 1'b1 = underrun
2-0	FIFOSTATE[2:0]	R	0x0	- **FIFO empty states in denoise-2d (2nd line buffers are Y,U,V)** 1'b0 = normal 1'b1 = empty

DN_STATES0_2D (Denoise-2D Count States #0) 0xB0A52164

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BY	DN	IM	0		LCNT_Y_2D										
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0					PCNT_Y_2D										

Field	Name	RW	Reset	Description
31	BY	R	-	- **Busy operation** 1'b0 = idle 1'b1 = busy
30	DN	R/W	0x0	- **Operation Done** 1'b0 = normal 1'b1 = Operation Done. (If write '1', this field is clear.)
29	IM	R/W	0x0	- **Done Interrupt mask** 1'b0 = Masked 1'b1 = Unmasked
26-16	LCNT_Y_2D	R	-	- **Line Counter in Luminance** Line Counter in Luminance
10-0	PCNT_Y_2D	R	-	- **Pixel Counter in Luminance** Pixel Counter in Luminance

DN_STATES1_2D (Denoise-2D Count States #1) 0xB0A52168

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0					LCNT_C_2D										
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0					PCNT_C_2D										

Field	Name	RW	Reset	Description
26-16	LCNT_Y_2D	R	-	- **Line Counter in Chrominance** Line Counter in Chrominance
10-0	PCNT_Y_2D	R	-	- **Pixel Counter in Chrominance** Pixel Counter in Chrominance

DN_CTRL (Denoise Control) 0xB0A5216C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLUSH		0													
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0														BY_COEFF	

Field	Name	RW	Reset	Description
31-30	FLUSH	R/W	-	- **FIFO Flush** Bit[31] = FIFO flish in denoise-3D Bit[30] = FIFO flish in denoise-2D
1-0	BY_COEFF	R/W	-	- **Bypass Coefficient** Bit[1] = Bypass coefficient in denoise-3D All coefficient of denoise-3D is set to '0' Bit[0] = Bypass coefficient in denoise-2D All coefficient of denoise-2D is set to '0'

DN_DIV_IMG_2D (Denoise-2D Divide image) 0xB0A52170

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEN	DTOG	0			POS										

Field	Name	RW	Reset	Description
15	DEN	R/W	0x0	- **Image divide enable** 1'b0 = normal 1'b1 = divide image (filtered image – original image)
14	DTOG	R/W	0x0	**- Divided image swap** 1'b0 = Left image : filtered image Right image : original image 1'b1 = Left image : original image Right image : filtered image
10-0	POS	R/W	0x0	- **Divide Position** Divide Position

Refer to Figure 9.11.

RD_IMG0_BASE0 (RDMA image #0 Base Address in Y Channel) 0xB0A52180

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG0_BASE0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG0_BASE0[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE0	R/W	0x0	- **RDMA image #0 base address in Y channel** RDMA image #0 base address in Y channel

RD_IMG0_BASE1 (RDMA image #0 Base Address in U Channel) 0xB0A52184

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG0_BASE1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG0_BASE1[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE1	R/W	0x0	- **RDMA image #0 base address in U channel** RDMA image #0 base address in U channel

RD_IMG0_BASE2 (RDMA image #0 Base Address in V Channel) 0xB0A52188

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG0_BASE2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG0_BASE2[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE2	R/W	0x0	- **RDMA image #0 base address in V channel** RDMA image #0 base address in V channel

RD_IMG0_OFS (RDMA image #0 Offset Address) 0xB0A5218C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG0_OFS1[15:0]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RD_IMG0_OFS0[15:0]															

Field	Name	RW	Reset	Description
31-16	OFS1	R/W	0x0	- **RDMA image #0 offset address in C channel** This offset means the line start point to the next line start point.
15-0	OFS0	R/W	0x0	- **RDMA image #0 offset address in Y channel** This offset means the line start point to the next line start point.

RD_IMG1_BASE0 (RDMA image #1 Base Address in Y Channel) 0xB0A52190

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG1_BASE0[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RD_IMG1_BASE0[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE0	R/W	0x0	- **RDMA image #1 base address in Y channel** RDMA image #1 base address in Y channel

RD_IMG1_BASE1 (RDMA image #1 Base Address in U Channel) 0xB0A52194

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG1_BASE1[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RD_IMG1_BASE1[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE1	R/W	0x0	- **RDMA image #1 base address in U channel** RDMA image #1 base address in U channel

RD_IMG1_BASE2 (RDMA image #1 Base Address in V Channel) 0xB0A52198

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG1_BASE2[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RD_IMG1_BASE2[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE2	R/W	0x0	- **RDMA image #1 base address in V channel** RDMA image #1 base address in V channel

RD_IMG1_OFS (RDMA image #1 Offset Address) 0xB0A5219C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG1_OFS1[15:0]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG1_OFS0[15:0]															

Field	Name	RW	Reset	Description
31-16	OFS1	R/W	0x0	- **RDMA image #1 offset address in C channel** This offset means the line start point to the next line start point.
15-0	OFS0	R/W	0x0	- **RDMA image #1 offset address in Y channel** This offset means the line start point to the next line start point.

RD_IMG2_BASE0_0 (RDMA Comp. data #0 Base Address in Y Channel) 0xB0A521A0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_BASE0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG2_BASE0[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE0	R/W	0x0	- **RDMA Comp. data #0 base address in Y channel** RDMA Comp. data #0 base address in Y channel

RD_IMG2_BASE1_0 (RDMA Comp. data #0 Base Address in U Channel) 0xB0A521A4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_BASE1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG2_BASE1[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE1	R/W	0x0	- **RDMA Comp. data #0 base address in U channel** RDMA Comp. data #0 base address in U channel

RD_IMG2_BASE2_0 (RDMA Comp. data #0 Base Address in V Channel) 0xB0A521A8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_BASE2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG2_BASE2[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE2	R/W	0x0	- **RDMA Comp. data #0 base address in V channel** RDMA Comp. data #0 base address in V channel

RD_IMG2_BASE0_1 (RDMA Comp. data #1 Base Address in Y Channel) 0xB0A521AC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_BASE0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG2_BASE0[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE0	R/W	0x0	- **RDMA Comp. data #1 base address in Y channel** RDMA Comp. data #1 base address in Y channel

RD_IMG2_BASE1_1 (RDMA Comp. data #1 Base Address in U Channel) 0xB0A521B0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_BASE1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG2_BASE1[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE1	R/W	0x0	- **RDMA Comp. data #1 base address in U channel** RDMA Comp. data #1 base address in U channel

RD_IMG2_BASE2_1 (RDMA Comp. data #1 Base Address in V Channel) 0xB0A521B4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_BASE2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG2_BASE2[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE2	R/W	0x0	- **RDMA Comp. data #1 base address in V channel** RDMA Comp. data #1 base address in V channel

RDMA2 is assigned 2 addresses for each channel. This mode is for ping-pong operation that uses two memory areas for auto-swap address. Normally, RD_IMG2_BASE#0 is set to odd frame address and the other is set to even frame address. This register is related to compression DMA. Therefore, RD_IMG2_BASE#_0 and WD_BASE#_1 are the same. (Therefore, RD_IMG2_BASE#_1 and WD_BASE#_2 are the same).

RD_CUR_ADDR0 (RDMA Current Address in Y Channel) 0xB0A521B8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CUR_ADDR0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CUR_ADDR0[15:2]														0	

Field	Name	RW	Reset	Description
31-0	CUR_ADDR0	R	-	- **RDMA image #0 Current Address** RDMA image #0 Current Address.

RD_CUR_ADDR1 (RDMA Current Address in U Channel) 0xB0A521BC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CUR_ADDR1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CUR_ADDR1[15:2]														0	

Field	Name	RW	Reset	Description
31-0	CUR_ADDR1	R	-	- **RDMA image #1 Current Address** RDMA image #1 Current Address.

RD_CUR_ADDR2 (RDMA Current Address in V Channel) 0xB0A521C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CUR_ADDR2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CUR_ADDR2[15:2]														0	

Field	Name	RW	Reset	Description
31-0	CUR_ADDR2	R	-	- **RDMA image #2 Current Address** RDMA image #2 Current Address.

RD_FIFOSTATE (RDMA FIFO States) 0xB0A521C4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FIFOSTATE[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FIFOSTATE[15:0]															

Field	Name	RW	Reset	Description
31	FIFOSTATE[31]	R/W	0x0	- **FIFO underrun in the 4th FIFO (Comp. OUT FIFO)** 1'b0 = normal 1'b1 = underrun
30-28	FIFOSTATE[30:28]	R	-	- **FIFO full states in the 4th FIFO (Comp.OUT Y,U,V)** 1'b0 = normal 1'b1 = full
27	FIFOSTATE[27]	R/W	0x0	- **FIFO underrun in the 3rd FIFO (Comp. DMA FIFO)** 1'b0 = normal 1'b1 = underrun
26-24	FIFOSTATE[26:24]	R	-	- - **FIFO full states in the 3rd FIFO (Comp. DMA Y,U,V)** 1'b0 = normal 1'b1 = full
23	FIFOSTATE[23]	R/W	0x0	- **FIFO underrun in the 2nd FIFO (RDMA #1 FIFO)** 1'b0 = normal 1'b1 = underrun
22-20	FIFOSTATE[22:20]	R	-	- **FIFO full states in the 2nd FIFO (RDMA #1 FIFO Y,U,V)** 1'b0 = normal 1'b1 = full
19	FIFOSTATE[19]	R/W	0x0	- **FIFO underrun in the 1st FIFO (RDMA #0 FIFO)** 1'b0 = normal 1'b1 = underrun
18-16	FIFOSTATE[18:16]	R	-	- **FIFO full states in the 1st FIFO (RDMA #0 FIFO)** 1'b0 = normal 1'b1 = full
15	FIFOSTATE[15]	R/W	0x0	- **FIFO overrun in the 4th FIFO (Comp. OUT FIFO)** 1'b0 = normal 1'b1 = overrrun
14-12	FIFOSTATE[14:12]	R	-	- **FIFO empty states in the 4th FIFO (Comp. OUT FIFO Y,U,V)** 1'b0 = normal 1'b1 = empty
11	FIFOSTATE[11]	R/W	0x0	- **FIFO overrun in the 3rd FIFO (Comp. DMA FIFO)** 1'b0 = normal 1'b1 = overrun
10-8	FIFOSTATE[10:8]	R	-	- **FIFO empty states in the 3rd FIFO (Comp. DMA FIFO Y,U,V)** 1'b0 = normal

Field	Name	RW	Reset	Description
				1'b1 = empty
7	FIFOSTATE[7]	R/W	0x0	- **FIFO overrun in the 2nd FIFO (RDMA #1 FIFO)** 1'b0 = normal 1'b1 = overrun
6-4	FIFOSTATE[6:4]	R	-	- **FIFO empty states in the 2nd FIFO (RDMA #1 FIFO Y,U,V)** 1'b0 = normal 1'b1 = empty
3	FIFOSTATE[3]	R/W	0x0	- **FIFO underrun in the 1st FIFO (RDMA #0 FIFO)** 1'b0 = normal 1'b1 = overrun
2-0	FIFOSTATE[2:0]	R	-	- **FIFO empty states in the 1st FIFO (RDMA #0 FIFO)** 1'b0 = normal 1'b1 = empty

RD_LINE_STATE0 (RDMA Count States #0) 0xB0A521C8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BUSY			0		LCNT_Y										
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0					PCNT_Y										

Field	Name	RW	Reset	Description
31-29	BUSY	R	-	- **RDMA BUSY** Bit[31] = RDMA Comp.DMA busy Bit[30] = RDMA image #1 busy Bit[29] = RDMA image #2 busy
26-16	LCNT_Y	R	-	- **Line Count in Y channel** Current line counter that is processing.
10-0	PCNT_Y	R	-	- **Pixel Count in Y channel** Current pixel counter that is processing.

RD_LINE_STATE1 (RDMA Count States #1) 0xB0A521CC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RUD	DN	MASK		0	LCNT_C										
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0					PCNT_C										

Field	Name	RW	Reset	Description
31	RUD	R/W	-	- **Register Update Done** 1'b0 = normal 1'b1= register update
30	DN	R/W	-	- **Frame Done** 1'b0 = normal 1'b1 = frame processing done
29-28	MASK	R/W	-	- **Interrupt mask** Bit[29] = Register updare interrupt mask Bit[28] = Frame done interrupt mask
26-16	LCNT_C	R		- **Line Count in C channel** Current line counter that is processing
10-0	PCNT_C	R		- **Pixel Count in C channel** Current pixel counter that is processing.

RD_CTRL (RDMA Control Register) 0xB0A521D0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															HC
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HM				0		HNS	CONT	BYH	TOG	0	OPT				

Field	Name	RW	Reset	Description
16	HC	R/W	0x0	- **Comp.DMA clear** 1'b0 = normal 1'b1= Comp. DMA buffer clear.
15-12	HM	R/W	0x0	- **Comp.DMA mode** 4'b0000 = Normal Mode (Quantizer level = 0) {-7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7} 4'b0001 = {-6, -6, -4, -4, -2, -2, 0, 0, 0, 2, 2, 4, 4, 6, 6} 4'b0010 = {-6, -6, -4, -4, -2, -2, -1, -0, 1, 2, 2, 4, 4, 6, 6} 4'b0011 = {-6, -6, -3, -3, -3, 0, 0, 0, 0, 0, 3, 3, 3, 6, 6} 4'b0100 = {-6, -6, -3, -3, -3, -1, -1, 0, 1, 1, 3, 3, 3, 6, 6} 4'b0101 = {-6, -6, -3, -3, -3, -2, -1, 0, 1, 2, 3, 3, 3, 6, 6} 4'b0110 = {-6, -6, -4, -4, -3, -2, -1, 0, 1, 2, 3, 4, 4, 6, 6} 4'b0111= {-6, -6,- 6, -3, -3, -2, -1, 0, 1, 2, 3, 3, 6, 6, 6} Others = NOT USED.
9	HNS	R/W	0x0	- **Comp.DMA operation Not Skip in first frame** 1'b0 = Skip in first frame 1'b1 = Not skip in first frame.
8	CONT	R/W	0x0	- **RDMA Continuous Mode** 1'b0 = frame-by-frame mode 1'b1 = continuous mode
7	BYH	R/W	0x0	- **Bypass Comp.DMA decode** 1'b0 = Decoding bitstream. Input bitstream is the encoded data. 1'b1 = Not decoding. Input bitstream is the non-encoded data.
6	TOG	R/W	0x0	- **RDMA Huffman base address auto toggle** 1'b0 = Auto toggle base address as changing frame If Odd frame, base address is RDMA_IMG2_BASE0_0, BASE1_0, BASE2_0 If Even frame, base address is RDMA_IMG2_BASE0_1, BASE1_1, BASE2_1 1'b1 = Not toggle base address. RDMA is used only RDMA_IMG2_BASE0_0, BASE0_1, BASE0_2.
4	OPT	R/W	0x0	- **RDMA transfer Optimization** 1'b0 = Burst 8 1'b1 = Burst 16

RD_COMP_PL0 (RDMA Comp. data size in Y channel) 0xB0A521D0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_HUFF_PL0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_HUFF_PL0[15:0]															

Field	Name	RW	Reset	Description
31-0	PL0	R	-	- **RDMA Comp. data size in Y channel.** In normally, this value is equal to image hsize * vsize

RD_COMP_PL1 (RDMA Comp. data size in C channel) 0xB0A521D4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_HUFF_PL1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_HUFF_PL1[15:0]															

Field	Name	RW	Reset	Description
31-0	PL1	R	-	- **RDMA Comp. data size in C channel.** In normally, this value is equal to image hsize * vsize / 4 in 420 Mode. If 422mode, this value is image hsize*vsize/2.

RD_IMG2_ENDADR0 (RDMA Comp. DMA end address in Y channel) 0xB0A521E0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_ENDADR0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG2_ENDADR0[15:0]															

Field	Name	RW	Reset	Description
31-0	ENDADR0	R/W	0x0	- **RDMA comp. data end-address in Y channel.** RDMA comp. data end-address in Y channel. This mode is only operated in rolling mode & auto toggle disable mode.

RD_IMG2_ENDADR1 (RDMA Comp. DMA end address in U channel) 0xB0A521E4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_ENDADR1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG2_ENDADR1[15:0]															

Field	Name	RW	Reset	Description
31-0	ENDADR1	R/W	0x0	- **RDMA comp. data end-address in U channel.** RDMA comp. data end-address in U channel. This mode is only operated in rolling mode & auto toggle disable mode.

RD_IMG2_ENDADR2 (RDMA Comp. DMA end address in V channel) **0xB0A521E8**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_ENDADR2[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RD_IMG2_ENDADR2[15:0]															

Field	Name	RW	Reset	Description
31-0	ENDADR2	R/W	0x0	- **RDMA comp. data end-address in V channel.** RDMA comp. data end-address in V channel. This mode is only operated in rolling mode & auto toggle disable mode.

RD_ROLL_CTRL (RDMA Comp. Data Rolling-mode Control Register) **0xB0A521EC**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
															EN

Field	Name	RW	Reset	Description
0	EN	R/W	0x0	- **RDMA comp.DMA Rolling Mode** 1'b0 = Rolling mode disable. 1'b1 = Rolling mode enable.

In RD_ROLL_EN field, the current address is initialized when the end_address is bigger than the current address while data is stored to the memory. When this mode is used, the auto toggle mode should be disabled and start address should be reset. This mode is always set with Comp.DMA rolling mode (WD_ROLL_CTRL (0xB0A5224c)).

Figure 1.11 describes two modes for storing comp.frame data. (a) describes normal mode where memory is allocated to 2 frames when storing data. In this mode, the 1st frame data is read from the memory while the 2nd frame data is stored. (ping-pong operation). But (b) is rolling mode that memory is allocated to some of 2 frames when storing data. The write operation is occurred about 2 lines earlier than read operation. Threfore, for VIQE there is no overlapping operation. In this method, memory size can be properly reduced for this kind of operation.

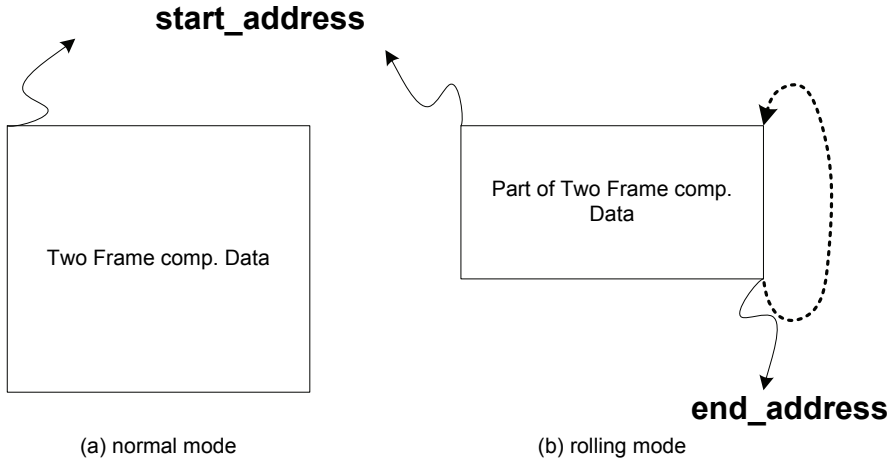


Figure 9.12 Two Modes to Store the comp. Data

Figure 9.13 below describes the above rolling mode.

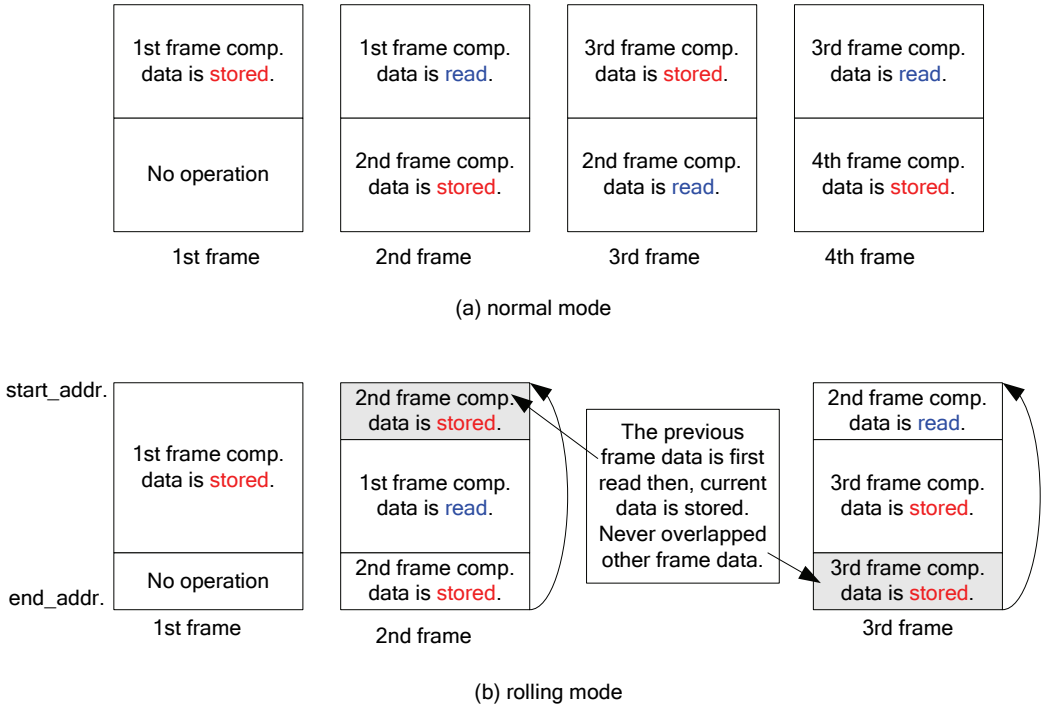


Figure 9.13 description of normal and rolling mode

- If you want to use rolling mode, refer to this sequence.
1. Set – RD_IMG1_BASE0_0, RD_IMG2_BASE1_0, RD_IMG2_BASE2_0, RD_IMG2_ENDADR0, RD_IMG2_ENDADR1, RD_IMG2_ENDADR2, WD_BASE0_0, WD_BASE1_0, WD_BASE2_0, WD_ENDADR0, WD_ENDADR1, WD_ENDADR2.
2. Set the TOG field of RD_CTRL and WD_CTRL register to 'HIGH'. Set the EN field of RD_ROLL_CTRL and WD_ROLL_CTRL register to 'HIGH'.
3.When the operation for the first frame is completed, read WD_CURADR0, WD_CURADR1 and WD_CURADR2 and save them to RD_IMG2_BASE1_0, RD_IMG2_BASE2_0. Start the operation for the next frame.
4. Repeat No.3.

RD_IMG2_CURADR0 (RDMA Comp. DMA current address in Y channel) **0xB0A521F0**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_CURADR0[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RD_IMG2_CURADR0[15:0]															

Field	Name	RW	Reset	Description
31-0	CURADR0	R	-	- **RDMA comp. current address in Y channel.** RDMA comp. current address in Y channel.

RD_IMG2_CURADR1 (RDMA Comp. DMA current address in U channel) **0xB0A521F4**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_CURADR1[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RD_IMG2_CURADR1[15:0]															

Field	Name	RW	Reset	Description
31-0	CURADR1	R	-	- **RDMA comp. current address in U channel.**

RD_IMG2_CURADR2 (RDMA Comp. DMA current address in V channel) 0xB0A521F8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RD_IMG2_CURADR2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD_IMG2_CURADR2[15:0]															

Field	Name	RW	Reset	Description
31-0	CURADR2	R	-	- **RDMA comp. current address in V channel.** RDMA comp. current address in V channel.

WD_BASE0_0 (Comp.DMA #0 base address in Y channel) 0xB0A52200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_BASE0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_BASE0[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE0	R/W	0x0	- **Comp. DMA #0 base address in Y channel** Comp.DMA #0 base address in Y channel

WD_BASE1_0 (Comp. DMA #0 base address in U channel) 0xB0A52204

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_BASE1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_BASE1[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE1	R/W	0x0	- **Comp. DMA #0 base address in U channel** Comp.DMA #0 base address in U channel

WD_BASE2_0 (Comp. DMA #0 base address in V channel) 0xB0A52208

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_BASE2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_BASE2[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE2	R/W	0x0	- **Comp. DMA #0 base address in V channel** Comp.DMA #0 base address in V channel

WD_BASE0_1 (Comp. DMA #1 base address in Y channel) 0xB0A5220C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_BASE0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_BASE0[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE0	R/W	0x0	- **Comp. DMA #1 base address in Y channel** Comp.DMA #1 base address in V channel

WD_BASE1_1 (Comp. DMA #1 base address in U channel) 0xB0A52210

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_BASE1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_BASE1[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE1	R/W	0x0	- **Comp. DMA #1 base address in U channel** Comp.DMA #1 base address in U channel

WD_BASE2_1 (Comp. DMA #1 base address in V channel) 0xB0A52214

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_BASE2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_BASE2[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE2	R/W	0x0	- **Comp. DMA #1 base address in V channel** Comp.DMA #1 base address in V channel

WD_CUR_ADDR (Comp. DMA current address) 0xB0A52218

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_CUR_ADDR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_CUR_ADDR[15:2]														0	

Field	Name	RW	Reset	Description
31-0	CUR_ADDR	R	-	- **Comp. DMA current address** Comp. DMA current address

WD_STATE (Comp. DMA State) **0xB0A5221C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BY	0											FIFOSTATE[7:4]			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DN	IM	0										FIFOSTATE[3:0]			

Field	Name	RW	Reset	Description
31	BY	R	-	- **Comp. DMA BUSY** 1'b0 = idle 1'b1 = busy
15	DN	R/W	0	- **Comp. DMA Frame Done** 1'b0 = normal 1'b1 = Frame done. If this field is write '1', this field is clear.
14	IM	R/W	0	- **Comp. DMA Frame Done interrupt mask** 1'b0 = mask 1'b1 = unmask
7	FIFOSTATE[7]	R/W	0	- **FIFO overrun in the FIFO**. 1'b0 = normal 1'b1 = overrun
6-4	FIFOSTATE[6-4]	R	-	**- FIFO full states in the FIFO (HuffDMA FIFO Y,U,V)** 1'b0 = normal 1'b1 = full
3	FIFOSTATE[3]	R/W	0	- **FIFO underrun in the FIFO**. 1'b0 = normal 1'b1 underrun
2-0	FIFOSTATE[2-0]	R	-	**- FIFO empty states in the FIFO (HuffDMA FIFO Y,U,V)** 1'b0 = normal 1'b1 = empty

WD_CTRL (Comp. DMA Control register) **0xB0A52220**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLR	0			HM				0	BY	TOG	OPT	0			

Field	Name	RW	Reset	Description
15	CLR	R/W	0	- **Comp. DMA FIFO clear** 1'b0 = normal 1'b1 = clear
11-8	HM	R/W	0	- **Comp. DMA mode** 4'b0000 = Normal Mode (Quantizer level = 0) {-7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7} 4'b0001 = {-6, -6, -4, -4, -2, -2, 0, 0, 0, 2, 2, 4, 4, 6, 6} 4'b0010 = {-6, -6, -4, -4, -2, -2, -1, -0, 1, 2, 2, 4, 4, 6, 6} 4'b0011 = {-6, -6, -3, -3, -3, 0, 0, 0, 0, 0, 3, 3, 3, 6, 6} 4'b0100 = {-6, -6, -3, -3, -3, -1, -1, 0, 1, 1, 3, 3, 3, 6, 6} 4'b0101 = {-6, -6, -3, -3, -3, -2, -1, 0, 1, 2, 3, 3, 3, 6, 6} 4'b0110 = {-6, -6, -4, -4, -3, -2, -1, 0, 1, 2, 3, 4, 4, 6, 6} 4'b0111 = {-6, -6,- 6, -3, -3, -2, -1, 0, 1, 2, 3, 3, 6, 6, 6} Others = NOT USED
6	BY	R/W	0	- **Comp. DMA Encoded Bypass** 1'b0 = Encoding data Output bitstream is the encoded data. 1'b1 Not encoding Output bitstream is non-encoded data.
5	TOG	R/W	0	- **Comp. DMA base address auto toggle** 1'b0 = Auto toggle base address as changing frame If Odd frame, base address is WD _BASE0_0, BASE1_0, BASE2_0 If Even frame, base address is WD_BASE0_1, BASE1_1, BASE2_1 1'b1 = Not toggle base address. HuffDMA is used only WD_ BASE0_0, BASE1_0, BASE2_0.
4	OPT	R/W	0	**- Comp.DMA transfer Optimization** 1'b0 = burst 8 1'b1 = burst 16

WD_COMP_CNT0 (Comp. DMA encoded data count in Y channel) **0xB0A52230**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HUFF_CNT0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HUFF_CNT0[15:2]														0	

Field	Name	RW	Reset	Description
31-0	CNT0	R/W	0x0	- **Comp. DMA encoded data count in Y channel** In normally, this field value is equal to image hsize * vsize

WD_COMP_CNT1(Comp. DMA encoded data count in U channel) 0xB0A52234

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HUFF_CNT131:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HUFF_CNT1[15:2]														0	

Field	Name	RW	Reset	Description
31-0	CNT1	R/W	0x0	- **Comp. DMA encoded data count in U channel** In normally, this field value is equal to image hsize * vsize /4 in 420 mode.

WD_COMP_CNT2 (Comp. DMA encoded data count in V channel) 0xB0A52238

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HUFF_CNT2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HUFF_CNT2[15:2]														0	

Field	Name	RW	Reset	Description
31-0	CNT2	R/W	0x0	- **Comp. DMA encoded data count in V channel** In normally, this field value is equal to image hsize * vsize /4 in 420 mode.

WD_ ENDADR0 (Comp. DMA end address in Y channel) 0xB0A52240

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_ ENDADR0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_ ENDADR0[15:0]															

Field	Name	RW	Reset	Description
31-0	ENDADR0	R/W	0x0	- **Comp. DMA end-address in Y channel.** This mode is only operated in rolling mode & auto toggle disable mode..

WD_ ENDADR1 (Comp. DMA end address in U channel) 0xB0A52244

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_ ENDADR1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_ENDADR1[15:0]															

Field	Name	RW	Reset	Description
31-0	ENDADR1	R/W	0x0	- **Comp. DMA end-address in U channel.** This mode is only operated in rolling mode & auto toggle disable mode..

WD_ ENDADR2 (Comp. DMA end address in V channel) 0xB0A52248

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_ ENDADR2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_ ENDADR2[15:0]															

Field	Name	RW	Reset	Description
31-0	ENDADR2	R/W	0x0	- **Comp. DMA end-address in V channel.** This mode is only operated in rolling mode & auto toggle disable mode..

WD_ROLL_CTRL (Comp. Data Rolling-mode Control Register) **0xB0A5224C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															EN

Field	Name	RW	Reset	Description
0	EN	R/W	0x0	- **Comp. DMA Rolling Mode.** 1'b0 = rolling mode disable 1'b1 = rolling mode enable

In WD_ROLL_EN field, the current address is initialized when the end_address is bigger than the current address while data is stored to the memory. When this mode is used, the auto toggle mode should be disabled and this mode should be set with RDMA Comp.DMA rolling mode (RD_ROLL_CTRL (0xB0A521EC)).
Refer to Figure 9.12 and Figure 9.13.

WD_ CURADR0 (Comp. DMA current address in Y channel) **0xB0A52250**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_ CURADR0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_ CURADR0[15:0]															

Field	Name	RW	Reset	Description
31-0	CURADR0	R	-	- **Comp. DMA current address in Y channel..** Comp. DMA current address in Y channel.

WD_ CURADR1 (Comp. DMA current address in U channel) **0xB0A52254**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_CURADR1[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_CURADR1[15:0]															

Field	Name	RW	Reset	Description
31-0	CURADR1	R	-	- **Comp. DMA current address in U channel..** Comp. DMA current address in U channel.

WD_ CURADR2 (Comp. DMA current address in V channel) **0xB0A52258**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WD_ CURADR2[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_ CURADR2[15:0]															

Field	Name	RW	Reset	Description
31-0	CURADR2	R	-	- **Comp. DMA current address in V channel..** Comp. DMA current address in V channel.

OD_BASE0 (ODMA base address in Y channel) **0xB0A52280**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OD_BASE0[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OD_BASE0[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE0	R/W	0x0	- **ODMA base address in Y channel** ODMA base address in Y channel

OD_BASE1 (ODMA base address in U channel) 0xB0A52284

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OD_BASE1[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OD_BASE1[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE1	R/W	0x0	- **ODMA base address in U channel** ODMA base address in U channel

OD_BASE2 (ODMA base address in V channel) 0xB0A52288

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OD_BASE2[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OD_BASE2[15:2]														0	

Field	Name	RW	Reset	Description
31-0	BASE2	R/W	0x0	- **ODMA base address in V channel** ODMA base address in V channel

OD_SIZE (ODMA image size) 0xB0A5228C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				HEIGHT											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				WIDTH											

Field	Name	RW	Reset	Description
27-16	HEIGHT	R/W	0x0	- **ODMA image height size** ODMA image height size
11-0	WIDTH	R/W	0x0	- **ODMA image widtht size** ODMA image width size

OD_OFS (ODMA address offset) 0xB0A52290

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				OFS1											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				OFS0											

Field	Name	RW	Reset	Description
27-16	OFS1	R/W	0x0	- **ODMA address offset in C channel** ODMA address offset in C channel
11-0	OFS0	R/W	0x0	- **ODMA address offset in Y channel** ODMA address offset in Y channel

OD_CFG (ODMA Config) 0xB0A52294

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0				CP	WAITCNT			0	RDY	-	DIR	TYPE			

Field	Name	RW	Reset	Description
11	CP	R/W	0x0	- **Chrominance writing mode in 420sep mode** 1'b0 = Y0->U0->V0->Y1->Y2->U1->… 1'b1 = Y0->U0->Y1->V0->Y2->U1->…
10-8	WAITCNT	R/W	0x0	- **Wait cycle** Wait cycle count for RDY being '1
6	RDY	R/W	0x0	- **Access wait control register** Access wait control register. Valid for DIR being '1' 1'b0 = wait for "WAITCNT+1" cycle for direct path. 1'b1 = wait until output FIFO is not empty
4	DIR	R/W	0x0	- **DIRECT PATH** 1'b0 = normal path 1'b1 = direct path
3-0	TYPE	R/W	0x0	- **Image Type** 4'b0000 = 4:2:0 separate mode 4'b0001 = 4:2:2 separate mode 4'b0010 = 4:2:2 sequence mode 0 4'b0011 = 4:2:2 sequence mode 1 4'b1010 = 4:2:2 interleave 4'b1011 = 4:2:0 interleave Others = NOT USED

OD_CTRL (ODMA control register) 0xB0A522A4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0							RST	0				IDONE	IBUSY	IRDY	0

Field	Name	RW	Reset	Description
8	RST	R/W	0x0	- **States machine reset** 1'b0 = normal 1'b1 = Reset
3	IDONE	R/W	0x0	- **Done Interrupt mask** 1'b0 = Mask 1'b1 = unmask
2	IBUSY	R/W	0x0	- **Busy Interrupt mask** 1'b0 = Mask 1'b1 = unmask
1	IRDY	R/W	0x0	- **RDY Interrupt mask** 1'b0 = Mask 1'b1 = unmask

OD_STATE (ODMA State) 0xB0A522A8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										IBY	IRY	IDN	FD	BA	RA

Field	Name	RW	Reset	Description
5	IBY	R/W	0x0	- **Busy interrupt flag** 1'b0 = normal 1'b1 = Generate busy interrupt
4	IRY	R/W	0x0	- **Ready Interrupt flag** 1'b0 = normal 1'b1 = Generate ready interrupt
3	IDN	R/W	0x0	- **Frame done Interrupt flag** 1'b0 = normal 1'b1 = Generate ready interrupt
2	FD	R/W	0x0	- **Frame done states** 1'b0 = normal 1'b1 = Frame done
1	BA	R/W		- **Busy all states** 1'b0 = normal 1'b1 = busy all
0	RA	R/W		- **Ready all states** 1'b0 = normal 1'b1 = ready all

GM_CTRL (Gamut Mapper Control Register) 0xB0A52400

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BYPASS															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R2YMD		Y2RMD				R2Y	Y2R							RND	INIT

Field	Name	RW	Reset	Description
31	BYPASS	R/W	0x0	- **Bypass Enable** 1'b0 = Normal Process 1'b1 = This enable bit can do bypassing between input ports and output ports in Gamut Mapper hardware. This bit can be used as another Gamut Mapper Enable bit.
15-14	R2YMD	R/W	0x0	- **R2Y Mode** 2'b00 = **** Y = 0.299*R + 0.587G + 0.114B*** **** Cb = -0.172*R – 0.339*G + 0.511*B + 128*** **** Cr = 0.511*R – 0.428*G – 0.083*B + 128*** The range for ***"RGB"*** is 16 ~ 235, ***"Studio Color".*** The result is ***"Studio Color"*** – Normally SDTV 2'b01 = **** Y = 0.257*R + 0.504*G + 0.098*B + 16*** **** Cb = -0.148*R – 0.291*G + 0.439*B + 128*** **** Cr = 0.439*R – 0.368*G – 0.071*B + 128*** The range for ***"RGB"*** is 0 ~ 255, ***"Computer System Color"*** The result is ***"Studio Color"*** – Normally SDTV***.*** **2'b10=** **** Y = 0.213*R + 0.715*G + 0.072*B*** **** Cb = -0.117*R – 0.394*G + 0.511*B + 128*** **** Cr = 0.511*R – 0.464*G – 0.047*B + 128*** The range for ***"RGB"*** is 16 ~ 235, ***"Studio Color"..*** ***The result is "Studio Color"*** – Normally

Field	Name	RW	Reset	Description
				HDTV. **2'b11=** **** Y = 0.183*R + 0.614*G + 0.062*B + 16*** **** Cb = -0.101*R – 0.338*G + 0.439*B + 128*** **** Cr = 0.439*R – 0.399*G – 0.040*B + 128*** The range for ***"RGB"*** is 0 ~ 255, ***"Computer System Color".*** The result is ***"Studio Color"*** – Normally HDTV.
13-12	Y2RMD	R/W	0x0	- **Y2R Mode** 2'b00 = **** R = Y + 1.371 * (Cr – 128)*** **** G = Y + 0.336 * (Cb – 128) – 0.698 * (Cr – 128)*** **** B = Y + 1.732 * (Cb – 128)*** The range for "***YCbCr***" is 16 ~ 235, "***Studio Color***". The result is "***Studio Color***" – Normally SDTV. 2'b01 = **** R = 1.164 * (Y – 16) + 1.596 * (Cr – 128)*** **** G = 1.164 * (Y – 16) – 0.391 * (Cb – 128) – 0.813 * (Cr – 128)*** **** B = 1.164 * (Y – 16) + 2.018 * (Cb – 128)*** The range for ***"YCbCr"*** is 16 ~ 235, ***"Studio Color".*** The result is ***"Computer System Color"*** – Normally SDTV. 2'b10 = **** R = Y + 1.540 * (Cr – 128)*** **** G = Y – 0.183 * (Cb – 128) – 0.459 * (Cr – 128)*** **** B = Y + 1.816 * (Cb – 128)*** The range for ***"YCbCr"*** is 16 ~ 235, ***"Studio Color".*** The result is ***"Studio Color"*** – Normally HDTV. 2'b11 = **** R = 1.164 * (Y – 16) + 1.793 * (Cr – 128)*** **** G = 1.164 * (Y – 16) – 0.213 * (Cb – 128) – 0.534 * (Cr – 128)*** **** B = 1.164 * (Y – 16) + 2.115 * (Cb – 128)*** The range for ***"YCbCr"*** is 16 ~ 235, ***"Studio Color".*** The result is ***"Computer System Color"*** – Normally HDTV.
9	R2Y	R/W	0x0	- **RGB-to-YCbCr Converter Enable** 1'b0 = R2Y Converter Disable 1'b1 = R2Y Converter Enable
8	Y2R	R/W	0x0	- **YCbCr-to-RGB Converter Enable** 1'b0 = Y2R Converter Disable 1'b1 = Y2R Converter Enable
1	RND	R/W	0x0	- **Round** 1'b0 = Normal Process 1'b1 = All the internal arithmetic logic use the round bit as '1'
0	INIT	R/W	0x0	- **Initialization Start** 1'b0 = No meaning 1'b1 = This initializes all LUTs in Gamut Mapper hardware. These procedures spend some processing time. The current status of initialization procedures can be checked by the GM_STATUS register.

GM_STATUS (Gamut Mapper Status Register) 0xB0A52404

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														BUSYINIT	DNINIT

Field	Name	RW	Reset	Description
1	BUSYINIT	R/W	0x0	**Busy state in Initialization** 1'b0 = Not being initialized 1'b1 = This indicates the initialization procedure is not done yet.
0	DNINIT	R/W	0x0	- **Done of initialization procedure** 1'b0 = No meaning 1'b1 = Done of initialization procedure

GM_REGION0 (Gamut Mapper Region 0 Register) 0xB0A52408

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN					XEND										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					XSTART										

Field	Name	RW	Reset	Description
31	EN	R/W	0x0	**- Region Selection Enable** 1'b0 = Normal Process 1'b1 = Only specific region of the whole image will be processed, pixels in other region will be intact
26-16	XEND	R/W	0x0	- **Last Horizontal Position of Region Selection** This means the last horizontal position of region selection
10-0	XSTART	R/W	0x0	- **First Horizontal Position of Region Selection** This means the first horizontal position of region selection

GM_REGION1 (Gamut Mapper Region 1 Register) 0xB0A5240C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					YEND										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					YSTART										

Field	Name	RW	Reset	Description
26-16	YEND	R/W	0x0	- **Last Vertical Position of Region Selection** This means the last vertical position of region selection.
10-0	YSTART	R/W	0x0	- **First Vertical Position of Region Selection** This means the first vertical position of region selection

HI_CTRL (Histogram Generator Control Register) 0xB0A52600

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLR	BCTRDIS											MFRM			AUTO
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIRFINI						LUTDIRINI	LUTDIRUP	LUTUSE	LUTINIT	LUTUUP	CDFEN	INTP		FINI	START

Field	Name	RW	Reset	Description
31	CLR	R/W	0x0	**- Clear Internal Variables** 1'b0 = No meaning 1'b1 = Clear All Internal Variables in Histogram Monitor
30	BCTRDIS	R/W	0x0	- **Busy Control Disable** 1'b0 = Normal processing 1'b1 = If this is set, the hardware ignores the BUSY status in Histogram Generator hardware. This bit may be not useful for normal processing.
19-17	MFRM	R/W	0x0	- **Multiple Frame Duration** This value means the number of frames to be processed at once. If the manual mode is on, after the user's triggering which is START bit, the hardware will operate during the number of frames defined by this value. Otherwise, the hardware will operate repeatedly the number of frames as a unit. If this is set as multiple frames, the hardware can make the average value from pixel data of corresponding frames. This register field is mapped to the followings. 3'b000 = 1 frame 3'b001 = 2 frames 3'b010 = 4 frames 3'b011 = 8 frames 3'b100 = 16 frames 3'b101 = 32 frames
16	AUTO	R/W	0x0	- **Automatic Mode** 1'b0 = Manual Mode. The measurement or LUT update can be controlled by the user. 1'b1 = If this bit is set, the hardware automatically processes the corresponding frame as the value of MFRM.
15	DIRFINI	R/W	0x0	- **Direct Finish Signal** 1'b0 = No meaning 1'b1 = This bit controls directly the hardware to be finished processing the current frame.
9	LUTDIRINI	R/W	0x0	- **Direct LUT initialization** 1'b0 = No meaning 1'b1 = This bit makes the hardware to start LUT initialization directly.
8	LUTDIRUP	R/W	0x0	- **Direct LUT updates** 1'b0 = No meaning 1'b1 This bit makes the hardware to update LUT directly..
7	LUTUSE	R/W	0x0	- **LUT Enable** 1'b0 = Do not use LUT 1'b1 = Use LUT in Histogram Generator. It affects only the luminance signals.
6	LUTINIT	R/W	0x0	- **LUT Initialization** 1'b0 = No meaning 1'b1 = This bit makes the hardware to start LUT initialization procedure after finishing the current frame processing
5	LUTUUP	R/W	0x0	- **LUT User Update** 1'b0 = No meaning

Field	Name	RW	Reset	Description
				1'b1 = This bit makes the hardware to update LUT after finishing the current frame processing.
4	CDFEN	R/W	0x0	- **CDF Enable** 1'b0 = CDF disable 1'b1 = This bit enables the feature of calculating fully histogram equalization LUT table. If LUTUSE bit is set, the luminance signals are mapped into the new values in generated LUT table. Otherwise, the result of LUT table can be monitored by the user.
3	INTP	R/W	0x0	- **Interpolation Start** 1'b0 = No meaning 1'b1 = This bit makes the hardware to start the interpolation for the missing pixels between the neighbor segments. This procedure starts after the current frame processing.
1	FINI	R/W	0x0	- **Measure Finish Signal** 1'b0 = No meaning 1'b1 = This bit controls the hardware to be finished processing the current frame and measuring the information about histogram in current frame.
0	START	R/W	0x0	- **Measure Start Signal** 1'b0 = No meaning 1'b1 = This bit controls the hardware to start processing the current frame and measuring the information about histogram in current frame.

HI_STATUS (Histogram Monitor Status Register) 0xB0A52604

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
													INTPCSTAT		MCSTAT
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
											LUTUUPB			INTPRDY	FBUSY

Field	Name	RW	Reset	Description
18-17	INTPCSTAT	R	-	**- Interpolation Controller State** This indicates the internal hardware state in interpolation logic. This is used for hardware debugging.
16	MCSTAT	R	-	- **Measure Controller State** 1'b0 = Not processing 1'b1 = This indicates that the measurement hardware logic is busy.
4	LUTUUPB	R	-	- **LUT User Update BUSY** 1'b0 = Not processing 1'b1 = This indicates that the LUT is being updated as the table written by the user.
1	INTPRDY	R	-	- **Interpolation BUSY** 1'b0 = Not processing 1'b1 = This indicates that the interpolation for missing pixels is processing.
0	FBUSY	R	-	- **Direct Finish Signal** 1'b0 = Not processing 1'b1 = This indicates that the current frame requires the Histogram Generator block to manipulate the output pixels, not just to monitor or get the information of pixels.

HI_CONFIG (Histogram Generator Configuration Register) 0xB0A52608

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	SIZE														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RND			VOFF								HOFF				

Field	Name	RW	Reset	Description
30-16	SIZE	R/W	0x0	**- Sample Size** Sample Size
15	RND	R/W	0x0	- **Round Control Bit** 1'b0 = Disable rounding 1'b1 = Enable rounding in Interpolation operation
12-8	VOFF	R/W	0x0	- **Vertical Offset of Sample Pixels** This means the vertical offset by which the pixels are sampled vertically. This value is larger, the sample size is smaller.
4-0	HOFF	R/W	0x0	- **Horizontal Offset of Sample Pixels** This means the horizontal offset by which the pixels are sampled horizontally. This value is larger, the sample size is smaller.

HI_REGION0 (Histogram Generator Region 0 Register) 0xB0A5260C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
EN					XEND										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					XSTART										

Field	Name	RW	Reset	Description
31	EN	R/W	0x0	**- Region Selection Enable** 1'b0 = Normal Process 1'b1 = Only specific region of the whole image will be processed in transformation using the LUT, but pixels in other region will be intact.
26-16	XEND	R/W	0x0	- **Last Horizontal Position of Region Selection** This means the last horizontal position of region selection.
10-0	XSTART	R/W	0x0	- **First Horizontal Position of Region Selection** This means the first horizontal position of region selection.

HI_REGION1 (Histogram Generator Region 1 Register) 0xB0A52610

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					YEND										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					YSTART										

Field	Name	RW	Reset	Description
26-16	YEND	R/W	0x0	- **Last Vertical Position of Region Selection** This means the last vertical position of region selection.
10-0	YSTART	R/W	0x0	- **First Vertical Position of Region Selection** This means the first vertical position of region selection.

HI_SEGS0 (Histogram Generator Segments 0 Register) 0xB0A52620

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEG03								SEG02							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SEG01								SEG00							

Field	Name	RW	Reset	Description
31-24	SEG03	R/W	0x0	- **Segment Positions** SEG**x** (**x** = 0, 1, 2, … 14) is the starting position of segment **x**. Total 16 segments compose 8bit pixel range(0 ~ 255). The SEG**x** is increasingly as **x** increases. The starting position of the first segment is fixed to 0, and the ending position of the last segment(SEG14, or 15th segment) is fixed to 255.
23-16	SEG02	R/W	0x0	
15-8	SEG01	R/W	0x0	
7-0	SEG00	R/W	0x0	

To program these registers should be done before starting the hardware, otherwise any output relative to the Histogram Generator is unreliable.

HI_SEGS1 (Histogram Generator Segments 1 Register) 0xB0A52624

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEG07								SEG06							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SEG05								SEG04							

Field	Name	RW	Reset	Description
31-24	SEG07	R/W	0x0	- **Segment Positions** SEG**x** (**x** = 0, 1, 2, … 14) is the starting position of segment **x**. Total 16 segments compose 8bit pixel range(0 ~ 255). The SEG**x** is increasingly as **x** increases. The starting position of the first segment is fixed to 0, and the ending position of the last segment(SEG14, or 15th segment) is fixed to 255.
23-16	SEG06	R/W	0x0	
15-8	SEG05	R/W	0x0	
7-0	SEG04	R/W	0x0	

To program these registers should be done before starting the hardware, otherwise any output relative to the Histogram Generator is unreliable.

HI_SEGS2 (Histogram Generator Segments 2 Register) 0xB0A52628

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEG11								SEG10							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SEG09								SEG08							

Field	Name	RW	Reset	Description
31-24	SEG11	R/W	0x0	- **Segment Positions** SEG**x** (**x** = 0, 1, 2, … 14) is the starting position of segment **x**. Total 16 segments compose 8bit pixel range(0 ~ 255). The SEG**x** is increasingly as **x** increases. The starting position of the first segment is fixed to 0, and the ending position of the last segment(SEG14, or 15th segment) is fixed to 255.
23-16	SEG10	R/W	0x0	
15-8	SEG09	R/W	0x0	
7-0	SEG08	R/W	0x0	

To program these registers should be done before starting the hardware, otherwise any output relative to the Histogram Generator is unreliable.

HI_SEGS3 (Histogram Generator Segments 3 Register) **0xB0A5262C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEG14															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEG13								SEG12							

Field	Name	RW	Reset	Description
23-16	SEG14	R/W	0x0	- **Segment Positions** SEG**x** (**x** = 0, 1, 2, … 14) is the starting position of segment **x**. Total 16 segments compose 8bit pixel range(0 ~ 255). The SEG**x** is increasingly as **x** increases. The starting position of the first segment is fixed to 0, and the ending position of the last segment(SEG14, or 15th segment) is fixed to 255.
15-8	SEG13	R/W	0x0	
7-0	SEG12	R/W	0x0	

To program these registers should be done before starting the hardware, otherwise any output relative to the Histogram Generator is unreliable.

HI_CDFS0 (Histogram Generator CDF output 0 Register) **0xB0A52630**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CDF03								CDF02							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDF01								CDF00							

Field	Name	RW	Reset	Description
31-24	CDF03	R	-	- **CDF outputs** CDF**x** (**x** = 0, 1, 2, … 15) is the normalized output for segment **x**. The CDF value can be described below. $$CDF_x = \frac{\sum_{k=0}^{x} PDF_x}{TotalSampleSize} \times 256$$ PDF**x** is the number of pixels which are in the corresponding segment(segment **x**). The CDF**x** is increasingly as **x** increases.
23-16	CDF02	R	-	
15-8	CDF01	R	-	
7-0	CDF00	R	-	

HI_CDFS1 (Histogram Generator CDF output 1 Register) **0xB0A52634**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CDF07								CDF06							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDF05								CDF04							

Field	Name	RW	Reset	Description
31-24	CDF07	R	-	- **CDF outputs** CDF**x** (**x** = 0, 1, 2, … 15) is the normalized output for segment **x**. The CDF value can be described below. $$CDF_x = \frac{\sum_{k=0}^{x} PDF_x}{TotalSampleSize} \times 256$$ PDF**x** is the number of pixels which are in the corresponding segment(segment **x**). The CDF**x** is increasingly as **x** increases.
23-16	CDF06	R	-	
15-8	CDF05	R	-	
7-0	CDF04	R	-	

HI_CDFS2 (Histogram Generator CDF output 2 Register) 0xB0A52638

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CDF11								CDF10							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CDF09								CDF08							

Field	Name	RW	Reset	Description
31-24	CDF11	R	-	- **CDF outputs** CDF**x** (**x** = 0, 1, 2, … 15) is the normalized output for segment **x**. The CDF value can be described below. $CDF_x = \frac{\sum_{k=0}^{x} PDF_x}{TotalSampleSize} \times 256$ PDF**x** is the number of pixels which are in the corresponding segment(segment **x**). The CDF**x** is increasingly as **x** increases.
23-16	CDF10	R	-	
15-8	CDF09	R	-	
7-0	CDF08	R	-	

HI_CDFS3 (Histogram Generator CDF output 3 Register) 0xB0A5263C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CDF15								CDF14							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CDF13								CDF12							

Field	Name	RW	Reset	Description
31-24	CDF15	R	-	- **CDF outputs** CDF**x** (**x** = 0, 1, 2, … 15) is the normalized output for segment **x**. The CDF value can be described below. $CDF_x = \frac{\sum_{k=0}^{x} PDF_x}{TotalSampleSize} \times 256$ PDF**x** is the number of pixels which are in the corresponding segment(segment **x**). The CDF**x** is increasingly as **x** increases.
23-16	CDF14	R	-	
15-8	CDF13	R	-	
7-0	CDF12	R	-	

HI_CNTS0 (Histogram Generator CNT 0 Register) 0xB0A52640

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT01															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT00															

Field	Name	RW	Reset	Description
31-16	CNT01	R	-	- **CNT output** CNT**x** (**x** = 0, 1, 2, … 15) is the number of pixels which are in the range of segment **x** in the current frame. These are raw values of hardware counters. These values are used to generate the CDF outputs, which are normalized to 8 bit pixel range.
15-0	CNT00	R	-	

HI_CNTS1 (Histogram Generator CNT 1 Register) 0xB0A52644

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT03															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT02															

Field	Name	RW	Reset	Description
31-16	CNT03	R	-	- **CNT output** CNT**x** (**x** = 0, 1, 2, … 15) is the number of pixels which are in the range of segment **x** in the current frame. These are raw values of hardware counters. These values are used to generate the CDF outputs, which are normalized to 8 bit pixel range.
15-0	CNT02	R	-	

HI_CNTS2 (Histogram Generator CNT 2 Register) 0xB0A52648

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT05															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT04															

Field	Name	RW	Reset	Description
31-16	CNT05	R	-	- **CNT output** CNT**x** (**x** = 0, 1, 2, … 15) is the number of pixels which are in the range of segment **x** in the current frame. These are raw values of hardware counters. These values are used to generate the CDF outputs, which are normalized to 8 bit pixel range.
15-0	CNT04	R	-	

HI_CNTS3 (Histogram Generator CNT 3 Register) 0xB0A5264C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT07															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT06															

Field	Name	RW	Reset	Description
31-16	CNT07	R	-	- **CNT output** CNT**x** (**x** = 0, 1, 2, … 15) is the number of pixels which are in the range of segment **x** in the current frame. These are raw values of hardware counters. These values are used to generate the CDF outputs, which are normalized to 8 bit pixel range.
15-0	CNT06	R	-	

HI_CNTS4 (Histogram Generator CNT 4 Register) 0xB0A52650

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT09															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT08															

Field	Name	RW	Reset	Description
31-16	CNT09	R	-	- **CNT output** CNT**x** (**x** = 0, 1, 2, … 15) is the number of pixels which are in the range of segment **x** in the current frame. These are raw values of hardware counters. These values are used to generate the CDF outputs, which are normalized to 8 bit pixel range.
15-0	CNT08	R	-	

HI_CNTS5 (Histogram Generator CNT 5 Register) 0xB0A52654

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT11															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT10															

Field	Name	RW	Reset	Description
31-16	CNT11	R	-	- **CNT output** CNT**x** (**x** = 0, 1, 2, … 15) is the number of pixels which are in the range of segment **x** in the current frame. These are raw values of hardware counters. These values are used to generate the CDF outputs, which are normalized to 8 bit pixel range.
15-0	CNT10	R	-	

HI_CNTS6 (Histogram Generator CNT 6 Register) 0xB0A52658

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT13															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT12															

Field	Name	RW	Reset	Description
31-16	CNT13	R	-	- **CNT output** CNT**x** (**x** = 0, 1, 2, … 15) is the number of pixels which are in the range of segment **x** in the current frame. These are raw values of hardware counters. These values are used to generate the CDF outputs, which are normalized to 8 bit pixel range.
15-0	CNT12	R	-	

HI_CNTS7 (Histogram Generator CNT 7 Register) 0xB0A5265C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT15															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT14															

Field	Name	RW	Reset	Description
31-16	CNT15	R	-	- **CNT output** CNT**x** (**x** = 0, 1, 2, … 15) is the number of pixels which are in the range of segment **x** in the current frame. These are raw values of hardware counters. These values are used to generate the CDF outputs, which are normalized to 8 bit pixel range.
15-0	CNT14	R	-	

HI_SCALE0 (Histogram Generator ALPHA 0 Register) 0xB0A52660

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ALPHA03								ALPHA02							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALPHA01								ALPHA00							

Field	Name	RW	Reset	Description
31-24	ALPHA03	R/W	0x0	- **ALPHA parameter** ALPHA**x** (**x** = 0, 1, 2, … 15) is the strength parameter for segment **x**. The ALPHA value is used to control the strength of each segment in the histogram equalization. The ALPHA value is larger, the effect of equalization increases. The maximum value of 255 means that the equalization result be transferred as it is.
23-16	ALPHA02	R/W	0x0	
15-8	ALPHA01	R/W	0x0	
7-0	ALPHA00	R/W	0x0	

HI_SCALE1 (Histogram Generator ALPHA 1 Register) 0xB0A52664

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ALPHA07								ALPHA06							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALPHA05								ALPHA04							

Field	Name	RW	Reset	Description
31-24	ALPHA07	R/W	0x0	- **ALPHA parameter** ALPHA**x** (**x** = 0, 1, 2, … 15) is the strength parameter for segment **x**. The ALPHA value is used to control the strength of each segment in the histogram equalization. The ALPHA value is larger, the effect of equalization increases. The maximum value of 255 means that the equalization result be transferred as it is.
23-16	ALPHA06	R/W	0x0	
15-8	ALPHA05	R/W	0x0	
7-0	ALPHA04	R/W	0x0	

HI_SCALE2 (Histogram Generator ALPHA 2 Register) 0xB0A52668

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ALPHA11								ALPHA10							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ALPHA09								ALPHA08							

Field	Name	RW	Reset	Description
31-24	ALPHA11	R/W	0x0	- **ALPHA parameter** ALPHA**x** (**x** = 0, 1, 2, … 15) is the strength parameter for segment **x**. The ALPHA value is used to control the strength of each segment in the histogram equalization. The ALPHA value is larger, the effect of equalization increases. The maximum value of 255 means that the equalization result be transferred as it is.
23-16	ALPHA10	R/W	0x0	
15-8	ALPHA09	R/W	0x0	
7-0	ALPHA08	R/W	0x0	

HI_SCALE3 (Histogram Generator ALPHA 3 Register) 0xB0A5266C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ALPHA15								ALPHA14							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ALPHA13								ALPHA12							

Field	Name	RW	Reset	Description
31-24	ALPHA15	R/W	0x0	- **ALPHA parameter** ALPHA**x** (**x** = 0, 1, 2, … 15) is the strength parameter for segment **x**. The ALPHA value is used to control the strength of each segment in the histogram equalization. The ALPHA value is larger, the effect of equalization increases. The maximum value of 255 means that the equalization result be transferred as it is.
23-16	ALPHA14	R/W	0x0	
15-8	ALPHA13	R/W	0x0	
7-0	ALPHA12	R/W	0x0	

HI_LUTS00 (Histogram Generator LUT 0 Register) 0xB0A52700

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LUT003								LUT002							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LUT001								LUT000							

Field	Name	RW	Reset	Description
31-24	LUT003	R/W	0x0	- **LUT value** LUT**x** (**x** = 0, 1, 2, … 255) is the mapping value for pixel **x**. If LUTUSE field in HI_CTRL register is set, the final output of Histogram Generator hardware is the output of LUT table. These registers can be read or written. Writing this table can be possible at any time, and can be updated and effected into actual outputs when the LUTUUP field or the LUTDIRUP in HI_CTRL register is set. But, Reading this table can be possible when the LUTUSE field is off or the FBUSY field in HI_STATUS register is not active.
23-16	LUT002	R/W	0x0	
15-8	LUT001	R/W	0x0	
7-0	LUT000	R/W	0x0	

HI_LUTS01 (Histogram Generator LUT 1 Register) 0xB0A52704

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LUT007								LUT006							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LUT005								LUT004							

Field	Name	RW	Reset	Description
31-24	LUT007	R/W	0x0	- **LUT value** LUT**x** (**x** = 0, 1, 2, … 255) is the mapping value for pixel **x**. If LUTUSE field in HI_CTRL register is set, the final output of Histogram Generator hardware is the output of LUT table. These registers can be read or written. Writing this table can be possible at any time, and can be updated and effected into actual outputs when the LUTUUP field or the LUTDIRUP in HI_CTRL register is set. But, Reading this table can be possible when the LUTUSE field is off or the FBUSY field in HI_STATUS register is not active.
23-16	LUT006	R/W	0x0	
15-8	LUT005	R/W	0x0	
7-0	LUT004	R/W	0x0	

HI_LUTS63 (Histogram Generator LUT 63 Register) 0xB0A527FC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LUT255								LUT254							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LUT253								LUT252							

Field	Name	RW	Reset	Description
31-24	LUT255	R/W	0x0	- **LUT value** LUT**x** (**x** = 0, 1, 2, … 255) is the mapping value for pixel **x**. If LUTUSE field in HI_CTRL register is set, the final output of Histogram Generator hardware is the output of LUT table. These registers can be read or written. Writing this table can be possible at any time, and can be updated and effected into actual outputs when the LUTUUP field or the LUTDIRUP in HI_CTRL register is set. But, Reading this table can be possible when the LUTUSE field is off or the FBUSY field in HI_STATUS register is not active.
23-16	LUT254	R/W	0x0	
15-8	LUT253	R/W	0x0	
7-0	LUT252	R/W	0x0	

10 DDI_CONFIG

10.1 Overview

The DDI_CONFIG controls over DDI bus device configurations, like LCD port muxing, on-the-fly mode port connection, HDMI AES KEY setting, and so on

10.1.1 DDI_CONFIG Specific Features

- Port connection of Scaler, LCDC, and VIQE for on-the-fly-mode
- Control the Connection path of each display devices (LCDC, HDMI, TV-Encoder)
- HDMI on-off control
- Set HDMI AESKEY for encryption

10.2 Block Diagram of DDI_CONFIG

Below figure describes simple block diagram of DDI_CONFIG block. The port mux for on-the-fly mode lies between LCD Controllers and VIQE/memory to memory scaler0/1, to eliminate the duplicate memory access, which is inevitable if they are not directly connected. For example, “VIQE-MSCL-LCDC” combination enables one-stop processing of image enhancement, scaling and ouput to external LCD and therefore it can reduce memory access time.

The LCD Controller supports various output interfaces such as HDMI, TV-OUT, LCDC. The LCDC interface can be provided upto 2 channels.

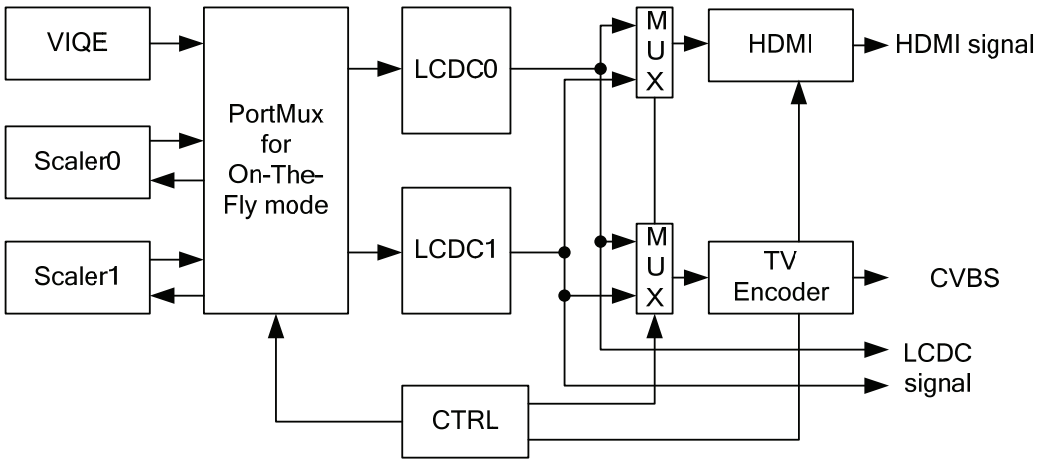


Figure 10.1 Block Diagram of DDI_CONFIG in NVS2310

10.3 Register Description

Table 10.1 DDI_CONFIG Register Map (Base Address = 0xB0A51000)

Name	Address	Type	Reset	Description
NTSCPAL_SEL	0x000	R/W	0x00000001	NTSCPAL_Encoder select
Reserved	0x004~0x028	R/W	-	-
HDMI_CTRL	0x02C	R/W	0x00008002	HDMI Control register
PWDN	0x030	R/W	0x00000000	Power Down
SWRESET	0x034	R/W	0x00000000	Soft Reset
ON_THE_FLY	0x038	R/W	0x00000000	On-The-Fly mode
HDMI_AES	0x044	R/W	0x00000000	HDMI AES
HDMI_AES_DATA0	0x048	RW	0x00000000	HDMI AES DATA #0
HDMI_AES_DATA1	0x04C	R/W	0x00000000	HDMI AES DATA #1
HDMI_AES_HW0	0x050	R/W	0x00000000	HDMI AES HW #0
HDMI_AES_HW1	0x054	R/W	0x00000000	HDMI AES HW #1
HDMI_AES_HW2	0x058	R/W	0x00000000	HDMI AES HW #2

NTSCPAL_SEL (NTSCPAL Select) 0xB0A51000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															SEL

Field	Name	RW	Reset	Discription
0	SEL	R/W	0	NTSCPAL Select 0 : LCDC0 connect to TV-Encoder 1 : LCDC1 connect to TV-Encoder

HDMI_CTRL (HDMI Control register) 0xB0A5102C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL	EN	0										RESET			

Field	Name	RW	Reset	Discription
15	SEL	R/W	0	Select for connection path of LCDC 0 : Connection path is LCDC0 and HDMI 1 : Connection path is LCDC1 and HDMI
14	EN	R/W	0	HDMI Enable 0 : Disable 1 : Enable
3-0	RESET	R/W	0	Reset for HDMI Bit 0 : HDMI Reset Bit 1 : SPDIF Reset Bit 2 : TMDS Reset Bit 3 : NOT USED

PWDN (Power down for DDIBUS) **0xB0A51030**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						PWDN									

Field	Name	RW	Reset	Discription
9	PWDN[9]	R/W	0	Power Down for LCDSI1 interface 0 : Power On 1 : Power Down
8	PWDN[8]	R/W	0	Power Down for HDMI interface 0 : Power On 1 : Power Down
7	PWDN[7]	R/W	0	Power Down for DDIBUS Cache 0 : Power On 1 : Power Down
6	PWDN[6]	R/W	0	Power Down for Memory Scaler #1 0 : Power On 1 : Power Down
5	PWDN[5]	R/W	0	Power Down for Memory Scaler #0 0 : Power On 1 : Power Down
4	PWDN[4]	R/W	0	Power Down for LCDSI0 interface 0 : Power On 1 : Power Down
3	PWDN[3]	R/W	0	Power Down for LCDC #1 0 : Power On 1 : Power Down
2	PWDN[2]	R/W	0	Power Down for LCDC #0 0 : Power On 1 : Power Down
1	PWDN[1]	R/W	0	Power Down for Video Image Quality Enhancer (VIQE) 0 : Power On 1 : Power Down
0	PWDN[0]	R/W	0	Reserved

SWRESET (SWReset for DDIBUS) 0xB0A51034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0						SWRESET									

Field	Name	RW	Reset	Discription
9	SWRESET[9]	R/W	0	Software Reset for LCDSI1 interface 0 : Normal 1 : Software Reset
8	SWRESET[8]	R/W	0	Software Reset for HDMI interface 0 : Normal 1 : Software Reset
7	SWRESET[7]	R/W	0	Software Reset for DDIBUS Cache 0 : Normal 1 : Software Reset
6	SWRESET[6]	R/W	0	Software Reset for Memory Scaler #1 0 : Normal 1 : Software Reset
5	SWRESET[5]	R/W	0	Software Reset for Memory Scaler #0 0 : Normal 1 : Software Reset
4	SWRESET[4]	R/W	0	Software Reset for LCDSI0 interface 0 : Normal 1 : Software Reset
3	SWRESET[3]	R/W	0	Software Reset for LCDC #1 0 : Normal 1 : Software Reset
2	SWRESET[2]	R/W	0	Software Reset for LCDC #0 0 : Normal 1 : Software Reset
1	SWRESET[1]	R/W	0	Software Reset for Video Image Quality Enhancer (VIQE) 0 : Normal 1 : Software Reset
0	SWRESET[0]	R/W	0	Reserved

ON_THE_FLY (On The Fly mode) 0xB0A51038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0										SC1		SC0		VIQE	

Field	Name	RW	Reset	Discription
5-4	SC1	R/W	0	Port Connection for the OnTheFly mode in MSCL1 0 : Connection path is from MSCL1 to LCDC0 1 : Connection path is from MSCL1 to LCDC1 2 : Connection path Is from MSCL1 to MSCL0 **** In this case, the VIQE should not be "2"*** 3 : NOT USED
3-2	SC0	R/W	0	Port Connection for the OnTheFly mode in MSCL0 0 : Connection path is from MSCL0 to LCDC0 **** In this case, the VIQE and SC1 should not be "0"*** 1 : Connection path is from MSCL0 to LCDC1 **** In this case, the VIQE and SC1 should not be "1"*** 2 : NOT USED 3 : Connection path Is from MSCL0 to MSCL1 **** In this case, the VIQE should not be "3"***
1-0	VIQE	R/W	0	Port Connection for the OnTheFly mode in VIQE 0 : Connection path is from VIQE to LCDC0 **** In this case, the SC0 and SC1 should not be "0"*** 1 : Connection path is from VIQE to LCDC1 **** In this case, the SC0 and SC1 should not be "1"*** 2 : Connection path is from VIQE to MSCL0 **** In this case, the SC1 should not be "2"*** 3 : Connection path Is from VIQE to MSCL1 **** In this case, the SC0 should not be "3"***

To use the on-the-fly mode, this registers should be set appropriately to get required path connections and formerly each module must be enabled.

On-the-fly connections are applicable from the combination of above register.

The related fields are "SRC" fields in the LI0C, LI1C, LI2C of the LCD controllers, "INPATH" of the MSCCTR register and "PATH" of the DSTCFG register of the memory-to-memory scaler, and "DIR" field of the OD_CFG of the Video Image Enhancement Block. Only the one of the "SRC" fields of the LI0C, LI1C, LI2C should be "1" to connect On-the-Fly path.

HDMI_AES (HDMI AESKEY Valid) 0xB0A51044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															VLD

Field	Name	RW	Reset	Discription
0	VLD	R/W	0	HDMI AES KEY valid 0 : Normal 1 : Valid

HDMI_AES_DATA0 (HDMI AESKEY DATA0) 0xB0A51048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AESKEY_DATA0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AESKEY_DATA0															

Field	Name	RW	Reset	Discription
31-0	AESKEY_DATA0	R/W	0	AESKEY DATA [31:0]

HDMI_AES_DATA1 (HDMI AESKEY DATA1) 0xB0A5104C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0															DATA1

Field	Name	RW	Reset	Discription
0	AESKEY_DATA1	R/W	0	AESKEY DATA [32]

HDMI_AES_HW_0 (HDMI AESKEY HW_0) 0xB0A51050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AESKEY_HW_0															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AESKEY_HW_0															

Field	Name	RW	Reset	Discription
31-0	AESKEY_HW_x	R/W	0	AESKEY DATA [64:33]

HDMI_AES_HW_1 (HDMI AESKEY HW_1) 0xB0A51054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AESKEY_HW_1															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AESKEY_HW_1															

Field	Name	RW	Reset	Discription
31-0	AESKEY_HW_1	R/W	0	AESKEY DATA [96:65]

HDMI_AES_HW_2 (HDMI AESKEY HW_2) 0xB0A51058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	AESKEY_HW_2														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AESKEY_HW_2															

Field	Name	RW	Reset	Discription
30-0	AESKEY_HW_2	R/W	0	AESKEY DATA [127:97]

11 DDI_Cache

11.1 Overview

Multiple memory access from each DDI sub-block and late response time degrades overall performance. Therefore the internal DDI Cache (DDIC) is required to make up for these excessive memory access time. It can fill internal buffer (64 depth * 64bit) at once when a read request has received from sub-blocks.

11.1.1 DDI_CACHE Specific Features

- 64 depths * 64 bit buffers * 6 channels * 2 DDICs
- Support the connection path of (11+14) channel DMAs
  DDIC0 (11 channels)
    LCD0 : 5 channels
    MSCL0 : 3 channels
    VIQE0 : 3 channels
  DDIC1 (14 channels)
    LCD1 : 5 channels
    MSCL1 : 3 channels
    VIQE1 : 3 channels
    VIQE2 : 3 channels
- Maximum connection paths are (6+6) for operation at the same time

11.2 Block Diagram of DDI_CACHE

The block diagram of DDI_CACHE is shown in next figure. It is composed of six buffers(64depth*64bit) and is connected with read DMA path of each sub-block such as VIQE, LCDC, MSCL.

PORT_MUX can assign only (6+6) request channels out of 25 request channels and 12 DMA slaves are access from these channels. Other channels, not assigned to DMA blocks, will read data directly from memory.

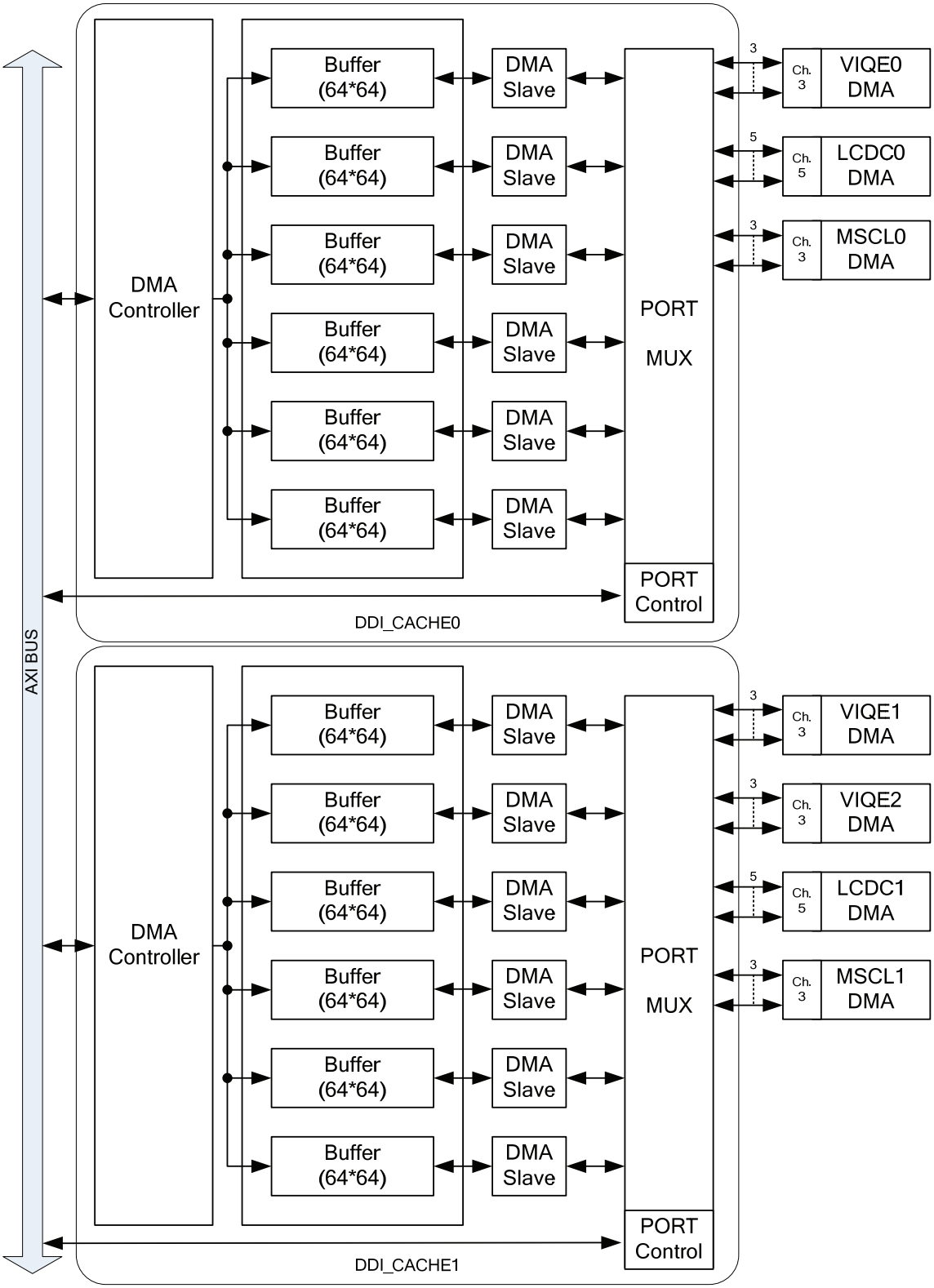


Figure 11.1 Block diagram of DDI_CACHE in NVS2310

11.3 Register Description

Table 11.1 DDI_CACHE Register Map (Base Address = 0xB0A50000)

Name	Address	Type	Reset	Description
DDIC_CTRL	0x000	R/W	0x00000000	DDI_CACHE Control
DDIC0_CFG0	0x004	R/W	0x00000000	DDI_CACHE0 Configuration #0
DDIC0_CFG1	0x008	R/W	0x00000000	DDI_CACHE0 Configuration #1
DDIC1_CFG0	0x00C	R/W	0x00000000	DDI_CACHE1 Configuration #0
DDIC1_CFG1	0x010	R/W	0x00000000	DDI_CACHE1 Configuration #1
DDIC0_CH0_AMIN	0x020	R/W	0x00000000	DDI_CACHE0 Ch0 Cacheable Address Range (MIN)
DDIC0_CH0_AMAX	0x024	R/W	0x00000000	DDI_CACHE0 Ch0 Cacheable Address Range (MAX)
DDIC0_CH1_AMIN	0x028	R/W	0x00000000	DDI_CACHE0 Ch1 Cacheable Address Range (MIN)
DDIC0_CH1_AMAX	0x02C	R/W	0x00000000	DDI_CACHE0 Ch1 Cacheable Address Range (MAX)
DDIC0_CH2_AMIN	0x030	R/W	0x00000000	DDI_CACHE0 Ch2 Cacheable Address Range (MIN)
DDIC0_CH2_AMAX	0x034	R/W	0x00000000	DDI_CACHE0 Ch2 Cacheable Address Range (MAX)
DDIC0_CH3_AMIN	0x038	R/W	0x00000000	DDI_CACHE0 Ch3 Cacheable Address Range (MIN)
DDIC0_CH3_AMAX	0x03C	R/W	0x00000000	DDI_CACHE0 Ch3 Cacheable Address Range (MAX)
DDIC0_CH4_AMIN	0x040	R/W	0x00000000	DDI_CACHE0 Ch4 Cacheable Address Range (MIN)
DDIC0_CH4_AMAX	0x044	R/W	0x00000000	DDI_CACHE0 Ch4 Cacheable Address Range (MAX)
DDIC0_CH5_AMIN	0x048	R/W	0x00000000	DDI_CACHE0 Ch5 Cacheable Address Range (MIN)
DDIC0_CH5_AMAX	0x04C	R/W	0x00000000	DDI_CACHE0 Ch5 Cacheable Address Range (MAX)
DDIC1_CH0_AMIN	0x050	R/W	0x00000000	DDI_CACHE1 Ch0 Cacheable Address Range (MIN)
DDIC1_CH0_AMAX	0x054	R/W	0x00000000	DDI_CACHE1 Ch0 Cacheable Address Range (MAX)
DDIC1_CH1_AMIN	0x058	R/W	0x00000000	DDI_CACHE1 Ch1 Cacheable Address Range (MIN)
DDIC1_CH1_AMAX	0x05C	R/W	0x00000000	DDI_CACHE1 Ch1 Cacheable Address Range (MAX)
DDIC1_CH2_AMIN	0x060	R/W	0x00000000	DDI_CACHE1 Ch2 Cacheable Address Range (MIN)
DDIC1_CH2_AMAX	0x064	R/W	0x00000000	DDI_CACHE1 Ch2 Cacheable Address Range (MAX)
DDIC1_CH3_AMIN	0x068	R/W	0x00000000	DDI_CACHE1 Ch3 Cacheable Address Range (MIN)
DDIC1_CH3_AMAX	0x06C	R/W	0x00000000	DDI_CACHE1 Ch3 Cacheable Address Range (MAX)
DDIC1_CH4_AMIN	0x070	R/W	0x00000000	DDI_CACHE1 Ch4 Cacheable Address Range (MIN)
DDIC1_CH4_AMAX	0x074	R/W	0x00000000	DDI_CACHE1 Ch4 Cacheable Address Range (MAX)
DDIC1_CH5_AMIN	0x078	R/W	0x00000000	DDI_CACHE1 Ch5 Cacheable Address Range (MIN)
DDIC1_CH5_AMAX	0x07C	R/W	0x00000000	DDI_CACHE1 Ch5 Cacheable Address Range (MAX)

DDIC_CTRL (DDI_CACHE Control) 0xB0A50000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0										DDIC1_EN					
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0										DDIC0_EN					

Field	Name	RW	Reset	Description
21	DDIC1_EN[5]	R/W	0	DDIC1 Ch5 Cache Enable 0 : Cache Disable 1 : Cache Enable
20	DDIC1_EN[4]	R/W	0	DDIC1 Ch4 Cache Enable 0 : Cache Disable 1 : Cache Enable
19	DDIC1_EN[3]	R/W	0	DDIC1 Ch3 Cache Enable 0 : Cache Disable 1 : Cache Enable
18	DDIC1_EN[2]	R/W	0	DDIC1 Ch2 Cache Enable 0 : Cache Disable 1 : Cache Enable
17	DDIC1_EN[1]	R/W	0	DDIC1 Ch1 Cache Enable 0 : Cache Disable 1 : Cache Enable
16	DDIC1_EN[0]	R/W	0	DDIC1 Ch0 Cache Enable 0 : Cache Disable 1 : Cache Enable
5	DDIC0_EN[5]	R/W	0	DDIC0 Ch5 Cache Enable 0 : Cache Disable 1 : Cache Enable
4	DDIC0_EN[4]	R/W	0	DDIC0 Ch4 Cache Enable 0 : Cache Disable 1 : Cache Enable
3	DDIC0_EN[3]	R/W	0	DDIC0 Ch3 Cache Enable 0 : Cache Disable 1 : Cache Enable
2	DDIC0_EN[2]	R/W	0	DDIC0 Ch2 Cache Enable 0 : Cache Disable 1 : Cache Enable
1	DDIC0_EN[1]	R/W	0	DDIC0 Ch1 Cache Enable 0 : Cache Disable 1 : Cache Enable
0	DDIC0_EN[0]	R/W	0	DDIC0 Ch0 Cache Enable 0 : Cache Disable 1 : Cache Enable

DDIC0_CFG0 (DDI_CACHE0 Configuration #0) 0xB0A50004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								DDIC0_CH5_SEL				DDIC0_CH4_SEL			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DDIC0_CH3_SEL				DDIC0_CH2_SEL				DDIC0_CH1_SEL				DDIC0_CH0_SEL			

Field	Name	RW	Reset	Description
23-20	DDIC0_CH5_SEL	R/W	0	DDIC0 Channel 5 Select (Cacheable)
19-16	DDIC0_CH4_SEL	R/W	0	DDIC0 Channel 4 Select (Cacheable)
15-12	DDIC0_CH3_SEL	R/W	0	DDIC0 Channel 3 Select (Cacheable)
11-8	DDIC0_CH2_SEL	R/W	0	DDIC0 Channel 2 Select (Cacheable)
7-4	DDIC0_CH1_SEL	R/W	0	DDIC0 Channel 1 Select (Cacheable)
3-0	DDIC0_CH0_SEL	R/W	0	DDIC0 Channel 0 Select (Cacheable)

DDIC0_CFG1 (DDI_CACHE0 Configuration #1) 0xB0A50008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0												DDIC0_CH10_SEL			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DDIC0_CH9_SEL				DDIC0_CH8_SEL				DDIC0_CH7_SEL				DDIC0_CH6_SEL			

Field	Name	RW	Reset	Description
19-16	DDIC0_CH10_SEL	R/W	0	DDIC0 Channel 10 Select (Non-Cacheable)
15-12	DDIC0_CH9_SEL	R/W	0	DDIC0 Channel 9 Select (Non-Cacheable)
11-8	DDIC0_CH8_SEL	R/W	0	DDIC0 Channel 8 Select (Non-Cacheable)
7-4	DDIC0_CH7_SEL	R/W	0	DDIC0 Channel 7 Select (Non-Cacheable)
3-0	DDIC0_CH6_SEL	R/W	0	DDIC0 Channel 6 Select (Non-Cacheable)

DDIC0_CHn_SEL	CACHE Selection
0	DDIC_LCD0_DMA_0_0
1	DDIC_LCD0_DMA_0_1
2	DDIC_LCD0_DMA_0_2
3	DDIC_LCD0_DMA_1
4	DDIC_LCD0_DMA_2
5	DDIC_MSCL0_DMA_0
6	DDIC_MSCL0_DMA_1
7	DDIC_MSCL0_DMA_2
8	DDIC_VIQE_DMA_0_0
9	DDIC_VIQE_DMA_0_1
10	DDIC_VIQE_DMA_0_2

The values from DDIC0_CH0_SEL to DDIC0_CH10_SEL should be all different. In the case that some of them are the same, an error occurs.

DDIC1_CFG0 (DDI_CACHE1 Configuration #0) 0xB0A5000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								DDIC1_CH5_SEL				DDIC1_CH4_SEL			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DDIC1_CH3_SEL				DDIC1_CH2_SEL				DDIC1_CH1_SEL				DDIC1_CH0_SEL			

Field	Name	RW	Reset	Description
23-20	DDIC1_CH5_SEL	R/W	0	DDIC1 Channel 5 Select (Cacheable)
19-16	DDIC1_CH4_SEL	R/W	0	DDIC1 Channel 4 Select (Cacheable)
15-12	DDIC1_CH3_SEL	R/W	0	DDIC1 Channel 3 Select (Cacheable)
11-8	DDIC1_CH2_SEL	R/W	0	DDIC1 Channel 2 Select (Cacheable)
7-4	DDIC1_CH1_SEL	R/W	0	DDIC1 Channel 1 Select (Cacheable)
3-0	DDIC1_CH0_SEL	R/W	0	DDIC1 Channel 0 Select (Cacheable)

DDIC1_CFG1 (DDI_CACHE1 Configuration #1) 0xB0A50010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DDIC1_CH13_SEL				DDIC1_CH12_SEL				DDIC1_CH11_SEL				DDIC0_CH10_SEL			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DDIC0_CH9_SEL				DDIC0_CH8_SEL				DDIC0_CH7_SEL				DDIC0_CH6_SEL			

Field	Name	RW	Reset	Description
31-28	DDIC1_CH13_SEL	R/W	0	DDIC1 Channel 13 Select (Non-Cacheable)
27-24	DDIC1_CH12_SEL	R/W	0	DDIC1 Channel 12 Select (Non-Cacheable)
23-20	DDIC1_CH11_SEL	R/W	0	DDIC1 Channel 11 Select (Non-Cacheable)
19-16	DDIC1_CH10_SEL	R/W	0	DDIC1 Channel 10 Select (Non-Cacheable)
15-12	DDIC1_CH9_SEL	R/W	0	DDIC1 Channel 9 Select (Non-Cacheable)
11-8	DDIC1_CH8_SEL	R/W	0	DDIC1 Channel 8 Select (Non-Cacheable)
7-4	DDIC1_CH7_SEL	R/W	0	DDIC1 Channel 7 Select (Non-Cacheable)
3-0	DDIC1_CH6_SEL	R/W	0	DDIC1 Channel 6 Select (Non-Cacheable)

DDIC0_CHn_SEL	CACHE Selection
0	DDIC_LCD1_DMA_0_0
1	DDIC_LCD1_DMA_0_1
2	DDIC_LCD1_DMA_0_2
3	DDIC_LCD1_DMA_1
4	DDIC_LCD1_DMA_2
5	DDIC_MSCL1_DMA_0
6	DDIC_MSCL1_DMA_1
7	DDIC_MSCL1_DMA_2
8	DDIC_VIQE_DMA_1_0
9	DDIC_VIQE_DMA_1_1
10	DDIC_VIQE_DMA_1_2
11	DDIC_VIQE_DMA_2_0
12	DDIC_VIQE_DMA_2_1
13	DDIC_VIQE_DMA_2_2

The values from DDIC1_CH0_SEL to DDIC1_CH13_SEL should be all different. In the case that some of them are the same, an error occurs.

DDIC0_CH0_AMIN (DDI_CACHE0 Ch0 Cacheable Address Range (MIN)) 0xB0A50020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH0_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH0_AMIN															

Field	Name	RW	Reset	Description
31-0	CH0_AMIN	R/W	0	DDI_CACHE0 Ch0 Cacheable Address Range (MIN)

DDIC0_CH1_AMIN (DDI_CACHE0 Ch1 Cacheable Address Range (MIN)) 0xB0A50028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH1_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1_AMIN															

Field	Name	RW	Reset	Description
31-0	CH1_AMIN	R/W	0	DDI_CACHE0 Ch1 Cacheable Address Range (MIN)

DDIC0_CH2_AMIN (DDI_CACHE0 Ch2 Cacheable Address Range (MIN)) 0xB0A50030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH2_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2_AMIN															

Field	Name	RW	Reset	Description
31-0	CH2_AMIN	R/W	0	DDI_CACHE0 Ch2 Cacheable Address Range (MIN)

DDIC0_CH3_AMIN (DDI_CACHE0 Ch3 Cacheable Address Range (MIN)) 0xB0A50038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3_AMIN															

Field	Name	RW	Reset	Description
31-0	CH3_AMIN	R/W	0	DDI_CACHE0 Ch3 Cacheable Address Range (MIN)

DDIC0_CH4_AMIN (DDI_CACHE0 Ch4 Cacheable Address Range (MIN)) 0xB0A50040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH4_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH4_AMIN															

Field	Name	RW	Reset	Description
31-0	CH4_AMIN	R/W	0	DDI_CACHE0 Ch4 Cacheable Address Range (MIN)

DDIC0_CH5_AMIN (DDI_CACHE0 Ch5 Cacheable Address Range (MIN)) 0xB0A50048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH5_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH5_AMIN															

Field	Name	RW	Reset	Description
31-0	CH0_AMIN	R/W	0	DDIC0 Channel 0 Cacheable Address Range (MIN)
31-0	CH1_AMIN	R/W	0	DDIC0 Channel 1 Cacheable Address Range (MIN)
31-0	CH2_AMIN	R/W	0	DDIC0 Channel 2 Cacheable Address Range (MIN)
31-0	CH3_AMIN	R/W	0	DDIC0 Channel 3 Cacheable Address Range (MIN)
31-0	CH4_AMIN	R/W	0	DDIC0 Channel 4 Cacheable Address Range (MIN)
31-0	CH5_AMIN	R/W	0	DDIC0 Channel 5 Cacheable Address Range (MIN)

DDIC0_CH0_AMAX (DDI_CACHE0 Ch0 Cacheable Address Range (MAX)) 0xB0A50024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH0_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH0_AMAX															

Field	Name	RW	Reset	Description
31-0	CH0_AMAX	R/W	0	DDI_CACHE0 Ch0 Cacheable Address Range (MAX)

DDIC0_CH1_AMAX (DDI_CACHE0 Ch1 Cacheable Address Range (MAX)) 0xB0A5002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH1_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1_AMAX															

Field	Name	RW	Reset	Description
31-0	CH1_AMAX	R/W	0	DDI_CACHE0 Ch1 Cacheable Address Range (MAX)

DDIC0_CH2_AMAX (DDI_CACHE0 Ch2 Cacheable Address Range (MAX)) 0xB0A50034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH2_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2_AMAX															

Field	Name	RW	Reset	Description
31-0	CH2_AMAX	R/W	0	DDI_CACHE0 Ch2 Cacheable Address Range (MAX)

DDIC0_CH3_AMAX (DDI_CACHE0 Ch3 Cacheable Address Range (MAX)) 0xB0A5003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3_AMAX															

Field	Name	RW	Reset	Description
31-0	CH3_AMAX	R/W	0	DDI_CACHE0 Ch3 Cacheable Address Range (MAX)

DDIC0_CH4_AMAX (DDI_CACHE0 Ch4 Cacheable Address Range (MAX)) 0xB0A50044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH4_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH4_AMAX															

Field	Name	RW	Reset	Description
31-0	CH4_AMAX	R/W	0	DDI_CACHE0 Ch4 Cacheable Address Range (MAX)

DDIC0_CH5_AMAX (DDI_CACHE0 Ch5 Cacheable Address Range (MAX)) 0xB0A5004C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH5_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH5_AMAX															

Field	Name	RW	Reset	Description
31-0	CH0_AMAX	R/W	0	DDI_CACHE0 Ch0 Cacheable Address Range (MAX)
31-0	CH1_AMAX	R/W	0	DDI_CACHE0 Ch1 Cacheable Address Range (MAX)
31-0	CH2_AMAX	R/W	0	DDI_CACHE0 Ch2 Cacheable Address Range (MAX)
31-0	CH3_AMAX	R/W	0	DDI_CACHE0 Ch3 Cacheable Address Range (MAX)
31-0	CH4_AMAX	R/W	0	DDI_CACHE0 Ch4 Cacheable Address Range (MAX)
31-0	CH5_AMAX	R/W	0	DDI_CACHE0 Ch5 Cacheable Address Range (MAX)

DDIC1_CH0_AMIN (DDI_CACHE1 Ch0 Cacheable Address Range (MIN)) 0xB0A50050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH0_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH0_AMIN															

Field	Name	RW	Reset	Description
31-0	CH0_AMIN	R/W	0	DDI_CACHE1 Ch0 Cacheable Address Range (MIN)

DDIC1_CH1_AMIN (DDI_CACHE1 Ch1 Cacheable Address Range (MIN)) 0xB0A50058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH1_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1_AMIN															

Field	Name	RW	Reset	Description
31-0	CH1_AMIN	R/W	0	DDI_CACHE1 Ch1 Cacheable Address Range (MIN)

DDIC1_CH2_AMIN (DDI_CACHE1 Ch2 Cacheable Address Range (MIN)) 0xB0A50060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH2_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2_AMIN															

Field	Name	RW	Reset	Description
31-0	CH2_AMIN	R/W	0	DDI_CACHE1 Ch2 Cacheable Address Range (MIN)

DDIC1_CH3_AMIN (DDI_CACHE1 Ch3 Cacheable Address Range (MIN)) 0xB0A50068

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3_AMIN															

Field	Name	RW	Reset	Description
31-0	CH3_AMIN	R/W	0	DDI_CACHE1 Ch3 Cacheable Address Range (MIN)

DDIC1_CH4_AMIN (DDI_CACHE1 Ch4 Cacheable Address Range (MIN)) 0xB0A50070

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH4_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH4_AMIN															

Field	Name	RW	Reset	Description
31-0	CH4_AMIN	RW	0	DDI_CACHE1 Ch4 Cacheable Address Range (MIN)

DDIC1_CH5_AMIN (DDI_CACHE1 Ch5 Cacheable Address Range (MIN)) 0xB0A50078

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH5_AMIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH5_AMIN															

Field	Name	RW	Reset	Description
31-0	CH0_AMIN	R/W	0	DDIC1 Channel 0 Cacheable Address Range (MIN)
31-0	CH1_AMIN	R/W	0	DDIC1 Channel 1 Cacheable Address Range (MIN)
31-0	CH2_AMIN	R/W	0	DDIC1 Channel 2 Cacheable Address Range (MIN)
31-0	CH3_AMIN	R/W	0	DDIC1 Channel 3 Cacheable Address Range (MIN)
31-0	CH4_AMIN	R/W	0	DDIC1 Channel 4 Cacheable Address Range (MIN)
31-0	CH5_AMIN	R/W	0	DDIC1 Channel 5 Cacheable Address Range (MIN)

DDIC1_CH0_AMAX (DDI_CACHE1 Ch0 Cacheable Address Range (MAX)) 0xB0A50054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH0_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH0_AMAX															

Field	Name	RW	Reset	Description
31-0	CH0_AMAX	R/W	0	DDI_CACHE1 Ch0 Cacheable Address Range (MAX)

DDIC1_CH1_AMAX (DDI_CACHE1 Ch1 Cacheable Address Range (MAX)) 0xB0A5005C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH1_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH1_AMAX															

Field	Name	RW	Reset	Description
31-0	CH1_AMAX	R/W	0	DDI_CACHE1 Ch1 Cacheable Address Range (MAX)

DDIC1_CH2_AMAX (DDI_CACHE1 Ch2 Cacheable Address Range (MAX)) 0xB0A50064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH2_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH2_AMAX															

Field	Name	RW	Reset	Description
31-0	CH2_AMAX	R/W	0	DDI_CACHE1 Ch2 Cacheable Address Range (MAX)

DDIC1_CH3_AMAX (DDI_CACHE1 Ch3 Cacheable Address Range (MAX)) 0xB0A5006C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH3_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3_AMAX															

Field	Name	RW	Reset	Description
31-0	CH3_AMAX	R/W	0	DDI_CACHE1 Ch3 Cacheable Address Range (MAX)

DDIC1_CH4_AMAX (DDI_CACHE1 Ch4 Cacheable Address Range (MAX)) 0xB0A50074

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH4_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH4_AMAX															

Field	Name	RW	Reset	Description
31-0	CH4_AMAX	R/W	0	DDI_CACHE1 Ch4 Cacheable Address Range (MAX)

DDIC1_CH5_AMAX (DDI_CACHE1 Ch5 Cacheable Address Range (MAX)) 0xB0A5007C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CH5_AMAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH5_AMAX															

Field	Name	RW	Reset	Description
31-0	CH0_AMAX	R/W	0	DDI_CACHE1 Ch0 Cacheable Address Range (MAX)
31-0	CH1_AMAX	R/W	0	DDI_CACHE1 Ch1 Cacheable Address Range (MAX)
31-0	CH2_AMAX	R/W	0	DDI_CACHE1 Ch2 Cacheable Address Range (MAX)
31-0	CH3_AMAX	R/W	0	DDI_CACHE1 Ch3 Cacheable Address Range (MAX)
31-0	CH4_AMAX	R/W	0	DDI_CACHE1 Ch4 Cacheable Address Range (MAX)
31-0	CH5_AMAX	R/W	0	DDI_CACHE1 Ch5 Cacheable Address Range (MAX)

PART8 – VIDEO BUS
NVS2310

Rev. 1.01

Apr 22, 2011

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial Release
2011-04-22	1.01	∗ Correct the register table and document format.

TABLE OF CONTENTS

Contents

Figures

Tables

1 Introduction

NVS2310 VIDEOBUS includes multi-standard Video Codec and JPEG Codec.

[Features]

- VIDEO INTERFACES
  - Block Diagram
  - Pipelines
  - Bus Architecture
  - Clock and Reset
  - Frame Buffer
  - The Bit Processor Program and Data Memory
  - Bit Processor
  - Video Codec
  - Register Descriptions

- JPEG INTERFACES
  - Encoder

2 Bus Architecture

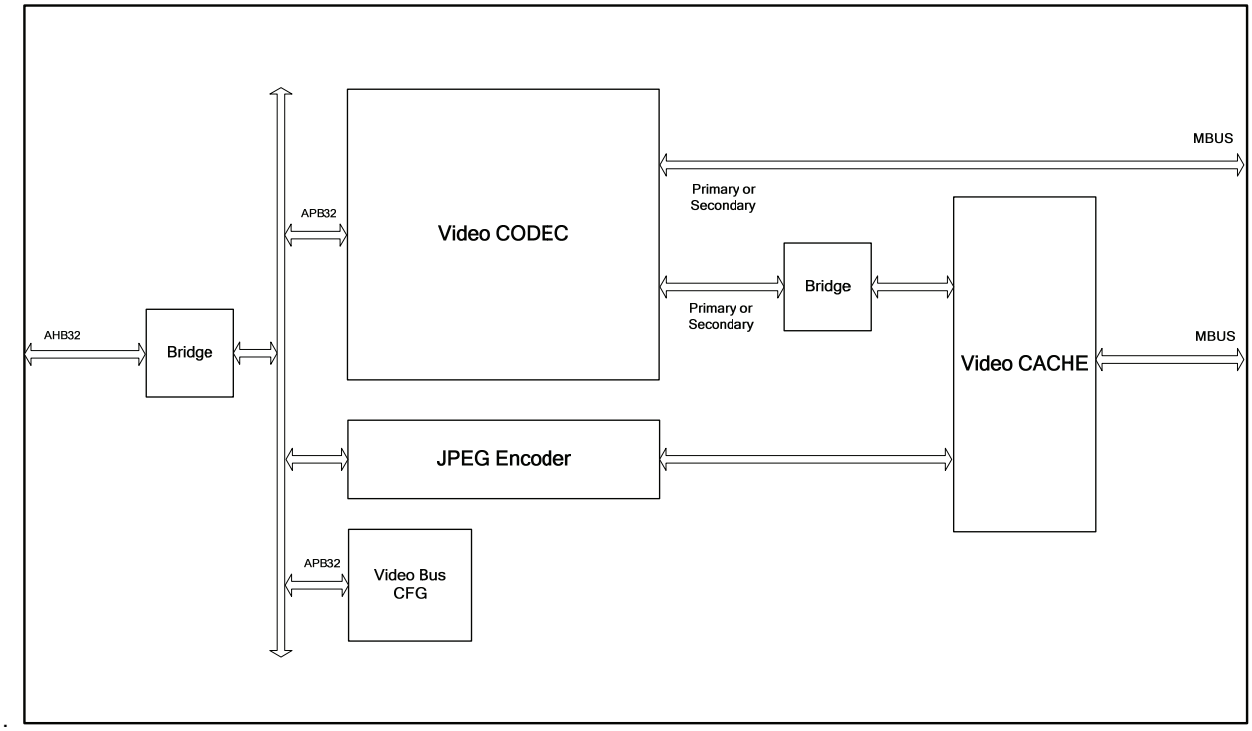


Figure 2.1 The Video Bus Architecture

3 Address and Register Map

Base Address		Peripherals
0xB0900000		VIDEO CODEC
0xB0910000		VIDEO CACHE
0xB0920000		VIDEO BUS Configuration
0xB0980000		JPEG Encoder

4 JPEG Encoder

4.1 Overview

The JPEG block of NVS2310 is a high performance solution for image and video compression applications. It can *compress* 24Mbyte/sec data stream in real-time and compliance with the baseline ISO/IEC 10918-1 JPEG standard.

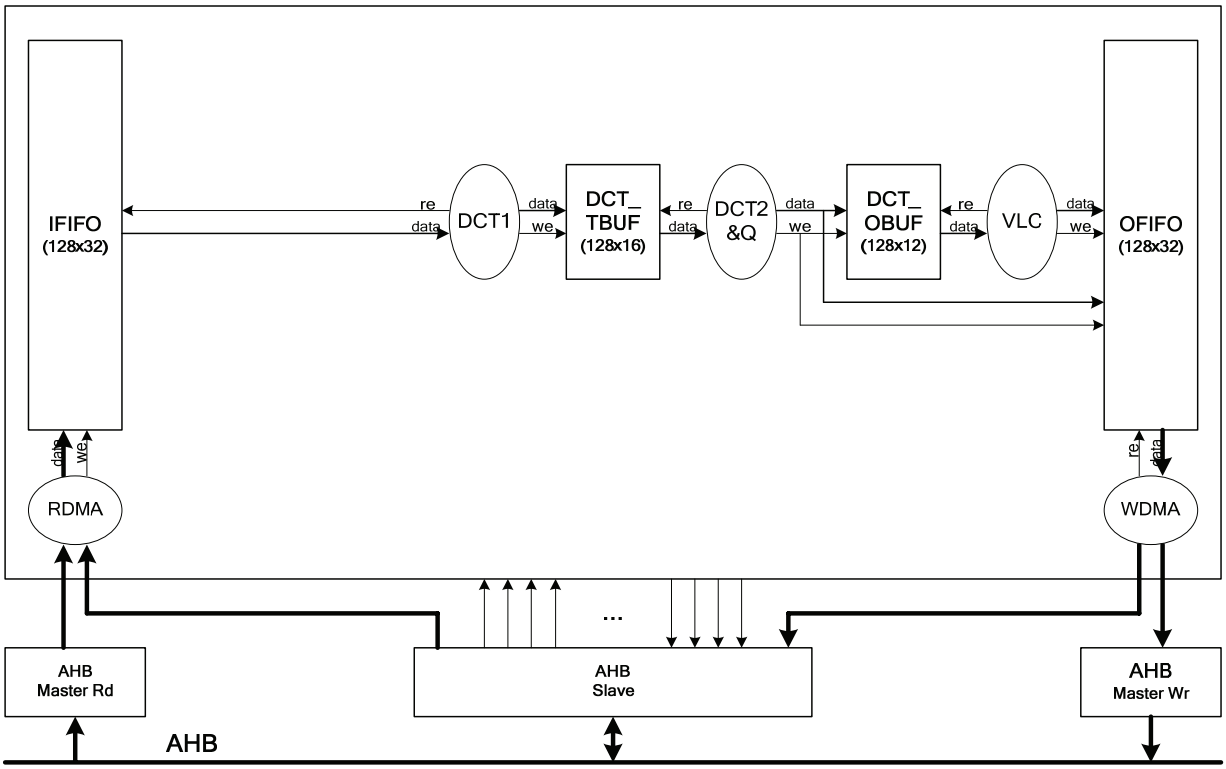


Figure 4.1 JPEC Block Diagram

The JPEG block compress any image size up to 4096x4096 and the image samples according to the user-defined quantization and Huffman tables, and it produces an ISO/IEC 10918-1 compatible data stream. Each 8x8 image block is frequency transformed by a forward DCT into the domain of 2-dimensional DCT(2D-DCT) basis images. Image samples are frequency transformed from the DCT unit using 15-bit internal accuracy. The 11-bit output DCT coefficients are then scanned int the Zig-Zag order and quantized according to programmed Quantization tables.

The quantization prepares the blocks for efficient coding. Each DCT coefficient block is divided by an 8x8 quantization matrix. The quantized DCT coefficients are then fed to the differential coding and run-length coding unit that produces the Run-Amplitude pairs for the Huffman coder. The Huffman encoder accepts symbols from RLE unit, which are further compressed by Huffman encoder, which encodes them according to one of the two possible programmed Huffman tables. Finally, Huffman compressed data stream is written to the output FIFO. The AHB bus master stores data in output FIFO to the predefined address pointer. The software header generator makes the JPEG file header and merges with the body, generated by the JPEG block.

4.2 JPEG Encoder Register

Table 4.1 JPEG Encoder Registers (Base Address = 0xB0980000)

Name	Addr	Type	Mode	Reset	Description
JP_MOD	0x004	R/W	ALL	0x00000000	JPEG codec mode register
JP_INT_MASK	0x008	R/W	ALL	0x0000001f	Interrupt mask register
JP_INT_LEVEL	0x00c	R/W	SLV	0x000000ff	FIFO interrupt level register
JP_TRG_MOD	0x010	R/W	ALL	0x00000000	Polling or Interrupt mode selection register
R_YBUF_ADDR	0x020	R/W	JP	0x00000000	Raw data buffer Y address register
R_UBUF_ADDR	0x024	R/W	JP	0x00000000	Raw data buffer U address register
R_VBUF_ADDR	0x028	R/W	JP	0x00000000	Raw data V address register
R_BUF_INFO	0x02c	R/W	JP	0x00000000	Raw data buffer information register
JP_SIZE	0x030	R/W	JP	0x00000000	Image size information register
JP_CHROMA	0x034	R/W	JP	0x00000000	Image format information register
JP_CBUF_ADDR	0x38	R/W	JP	0x00000000	Coded data buffer address register
JP_CBUF_SIZE	0x03c	R/W	JP	0x00000fff	Coded data buffer size register
JP_START	0x070	W	ALL	0x00000000	Codec start command register
JP_SBUF_WCNT	0x080	R/W	MST	0x00000000	Source buffer write count register
JP_SBUF_RCNT	0x084	R	MST	0x00000000	Source buffer read count register
JP_DBUF_WCNT	0x088	R	MST	0x00000000	Destination buffer write count register
JP_DBUF_RCNT	0x08c	R/W	MST	0x00000000	Destination buffer read count register
JP_IFIFO_ST	0x090	R	SLV	0x00000000	Input FIFO status register
JP_OFIFO_ST	0x094	R	SLV	0x00000000	Output FIFO status register
JP_INT_FLAG	0x0a0	R	ALL	0x00000000	Interrupt flag register
JP_INT_ACK	0x0a4	R	ALL	0x00000000	Interrupt ack register
JP_IFIFO_WD	0x0c0	W	SLV	0x00000000	Input FIFO write data register
JP_OFIFO_RD	0x0e0	R	SLV	0x00000000	Output FIFO read data register
JPC_QTAB0	0x100 -	W	JPC	0x00000000	Encoder Q table 0 (64 entries)
JPC_QTAB1	0x200 -	W	JPC	0x00000000	Encoder Q table 1 (64 entries)

JPEG Codec Mode Register(JP_MODE) 0xB0980004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
														OPM	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
															IPM

Field	Name	RW	Reset	Description
17-16	OPM	R/W	0	Operation Mode 0 : Encoding Mode
0	IPM	R/W	0	Input Mode 0 : Master Input Mode 1 : Slave Input Mode

Interrupt Mask(JP_INT_MASK) 0xB0980008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
										IMSK					

In the interrupt mode, those bits can mask the interrupt.

Field	Name	RW	Reset	Description
5-0	IMSK	R/W	0x0000001f	IMSK[5] : Raw Data Buffer IMSK[4] : Coded Buffer IMSK[3] : Input FIFO IMSK[2] : Output FIFO IMSK[1] : Error IMSK[0] : Operation End.

Input FIFO Interrupt Level Register(JP_INT_LEVEL) 0xB098000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								IFIFO							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								OFIFO							

Field	Name	RW	Reset	Description
23-16	IFIFO	R/W	0x000000ff	Input FIFO interrupt level. (0 ~ 127)
7-0	OFIFO	R/W	0x000000ff	Output FIFO interrupt level (0 ~ 127)

Trigger Mode Register(JP_TRG_MOD) 0xB0980010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
															TMD

Field	Name	RW	Reset	Description
0	TMD	R/W	0x00000000	0 : Polling mode. 1 : Interrupt mode.

Raw Data Buffer Address Register (R_Y/U/VBUF_A) 0xB09800(20/24/28)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R_BUF_ADDR[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R_BUF_ADDR[15:2]															

Field	Name	RW	Reset	Description
31-2	YBUF_ADDR	R/W	0x00000000	The raw buffer Y start address.
	UBUF_ADDR			The raw buffer U start address.
	VBUF_ADDR			The raw buffer V start address.

Raw Buffer Information Register (R_BUF_INFO) 0xB098002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				IFRM_HSIZE											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				IFRM_VSIZE											

Field	Name	RW	Reset	Description
27-19	IFRM_HSIZE	R/W	0x00000000	Input buffer horizontal size (unit == pixel)
11-0	IFRM_VSIZE	R/W	0x00000000	Input buffer vertical size (unit == line)

Image Size Register (IMG_SIZE) 0xB0980030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
				IMG_HSIZE[11:0]											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				IMG_VSIZE[11:0]											

Field	Name	RW	Reset	Description
27-16	IMG_HSIZE	R/W	0x00000000	Image horizontal size (unit == pixel)
11-0	IMG_VSIZE	R/W	0x00000000	Image vertical size (unit == line)

Image Format Register(CHROMA) 0xB0980034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
													CHROMA		

Field	Name	RW	Reset	Description
2-0	CHROMA	R/W	0x00000000	1 : YUV420 (Y,U,V separated mode) 2 : YUV422 (Y,U,V separated mode) 0,3,4,5,6,7 : Not in use

Coded Buffer Address Register(CBUF_ADDR) 0xB0980038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CBUF_ADDR[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CBUF_ADDR[15:2]														0	0

Field	Name	RW	Reset	Description
11-0	CBUF_ADDR	R/W	0x00000000	The coded data buffer start address.. In encoding mode : encoding data output pointer

Coded Buffer Size Register (CBUF_SIZE) 0xB098003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				CBUF_SIZE											

Field	Name	RW	Reset	Description
11-0	CBUF_SIZE	R/W	0x00000fff	Coded data buffer size (unit = 1024bytes)

Block Start Register (JP_START) 0xB0980070

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
															ST

Field	Name	RW	Reset	Description
0	ST	W	0x00000000	1 : Encoding start command.

Source Buffer Write Counter (SBUF_WC) 0xB0980080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				WC											

Field	Name	RW	Reset	Description
11-0	WC	R/W	0x00000000	Source buffer write count (unit = 1line)

Source Buffer Read Counter (SBUF_RC) 0xB0980084

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				RC											

Field	Name	RW	Reset	Description
11-0	RC	R	0x00000000	Source buffer read count (unit = 1line)

Destination Buffer Write Counter (DBUF_WC) 0xB0980088

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
														WC[17:16]	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WC[15:0]															

Field	Name	RW	Reset	Description
17-0	WC	R	0x00000000	Destination buffer write count (unit = 16bytes)

Destination Buffer Read Counter (DBUF_RC) 0xB098008C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
				RC											

Field	Name	RW	Reset	Description
11-0	RC	R/W	0x00000000	Destination buffer read count (unit = 1024bytes)

Input FIFO Status (IFIFO_ST) 0xB0980090

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								FILL							

Field	Name	RW	Reset	Description
7-0	FILL	R	0x00000000	Input FIFO fill count

Output FIFO Status (OFIFO_ST) 0xB0980094

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
								FILL							

Field	Name	RW	Reset	Description
7-0	FILL	R	0x00000000	Output FIFO fill count

Interrupt Flags (INT_FLAG) 0xB09800A0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
										IFLAG[5:0]					

Field	Name	RW	Reset	Description
5-0	IFLAG	R	0x00000000	0 : Job finished 1 : Decoding error 2 : Output FIFO status 3 : Input FIFO status 4 : Coded buffer status 5 : Raw data buffer status

Interrupt ACK Register (INT_ACK) 0xB09800A4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INT_CLEAR[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
INT_CLEAR[15:0]															

Field	Name	RW	Reset	Description
31-0	INT_CLEAR	R	0x00000000	Interrupt flags are cleared

When read this register, Interrupt Flags are cleared.

Input FIFO Write Data Register (IFIFO_WD) 0xB09800C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WDATA[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WDATA[15:0]															

Field	Name	RW	Reset	Description
31-0	WDATA	W	0x00000000	The write data register in slave mode.

In slave mode (the JPEG decoding/encoding or DCT/IDCT mode), host feds data into this register. (maximum burst size == 8)

Output FIFO Read Data Register (OFIFO_RD) 0xB09800E0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDATA[31:16]															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RDATA[15:0]															

Field	Name	RW	Reset	Description
31-0	RDATA	R	0x00000000	The read data register in slave mode.

In slave mode (the JPEG decoding/encoding), host gets result data from this register. (maximum burst size == 8)

Encoder Q Table 0 (EN_QTAB0) 0xB0980100-0xB09801FC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								Qdata1[11:4]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Qdata1[3:0]				Qdata0[11:0]											

Field	Name	RW	Reset	Description
11-0	Qdata0	W	0x00000000	Encoder quantization data (64 entries)

Encoder Q Table 1 (EN_QTAB1) 0xB0980200-0xB09802FC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								Qdata1[11:4]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Qdata1[3:0]				Qdata0[11:0]											

EN_QTAB0(64 entries) is the JPEG encoder quantization table for Y component. (4096/qstep) and raster scan ordered.

EN_QTAB1(64 entries) is the JPEG encoder quantization table for C component. (4096/qstep) and raster scan ordered.

Field	Name	RW	Reset	Description
11-0	Qdata1	W	0x00000000	Encoder quantization data (64 entries)

- 24bit Q Table Setting Guide

```
For(i=0; i<32; i=i+1) {
   For(j=0; j<64; j++) {
            If( (2*i+1) == hZZ[j] )
                     Wdata = qmat[0][j] << 12;
   }
   For(j=0; j<64; j++) {
            If( (2*i) == hZZ[j] )
                     Wdata = Wdata + qmat[0][j];
   }
   Regw(QtableBase+i*4, Wdata);
}
```

5 Video Codec

5.1 Overview

The NVS2310 VIDEO CODEC is a high performance multi-standard video codec IP that can perform the H.264 BP/MP/HP, VC-1 SP/MP/AP, MPEG-4 SP/ASP, Divx, MPEG-1/2. It can encode or decode multiple video clips with multiple standards simultaneously.

It connects with the system via the 32-bit AMBA 3 APB bus for system control and 64-bit AMBA3 AXI for data throughput, and takes advantage of on-chip memories to achieve high performance.

The VIDEO CODEC has a 16-bit DSP, called as BIT processor. The BIT processor controls the internal video codec sub blocks and communicates with a host processor through the host interface. The required resource for the host CPU to control the VIDEO CODEC is very low, under 1 MIPS, because all functions such as rate control, FMO, ASO, video codec control and error resilience are implemented in the BIT processor. The most part of sub-blocks in the VIDEO CODEC is optimally shared, which enables to achieve the ultra low power and low gate count.

5.2 Features

The main features of VIDEO CODEC are fully compliant with H.264 BP/MP/HP, VC-1 SP/MP/AP, MPEG-4 SP/ASP(except GMC), Divx (Xvid), MPEG-1/2. The VIDEO CODEC can support various error resilience tools and also support multiple decoding and full duplex multi-party-call simultaneously.

The VIDEO CODEC is easy to integrate into the system because its interface is composed of general and simple AXI/APB. And the VIDEO CODEC provides programmability, flexibility and ease of upgrade in decoding/encoding or host interface because all the controls in decoding/encoding process and host interface are implemented as firmware in a programmable BIT processor.

The detailed features of the VIDEO CODEC are as follows.

Table 5.1 Detailed Features of the CODEC

	Standard	Profile	Level
Encoder	H.264	Baseline	3.0
	MPEG-4	SP	
	H.263	Profile 3	
Decoder	H.264	BP/MP/HP	5.1
	MPEG-4	ASP	
	H.263	Profile 3	
	VC-1	SP/MP/AP	
	MPEG-2	Main	High
	Divx(Xvid)	Home theater	

- Encoding Tools
  - [±16, ±16] 1/2 and 1/4-pel accuracy motion estimation
  - 16x16, 16x8, 8x16 and 8x8 block sizes are supported.
  - Available block sizes can be configurable.
  - The encoder uses only one reference frame for the motion estimation.
  - Unrestricted motion vector
  - Prediction
    - MPEG-4 AC/DC prediction
    - H.264/AVC intra-prediction
  - H.263 Annex J, K (RS=0 and ASO=0), and T are supported.
  - Error resilience tools
    - MPEG-4 resync marker & data-partitioning with RVLC (Fixed number of bits/macroblocks between macroblocks)
    - CIR (Cyclic Intra Refresh)/AIR (Adaptive Intra Refresh)
    - Bit-rate control (CBR & VBR)
  - Minimum encoding image size is 16 pixels in horizontal and16 pixels in vertical.
- Decoding Tools
  - H.264
    - Fully compatible with the ITU-T Recommendation H.264 specification in BP/MP and HP.

- Supports CABAC/CAVLC
- Variable block size (16x16, 16x8, 8x16, 8x8, 8x4, 4x8 and 4x4)
- Error detection, concealment and error resilience tools

- VC1
  - Supports all VC-1 profile features – SMPTE “Proposed SMPTE Standard for Television: VC-1 Compressed Video Bitstream format and Decoding Process”
  - Supports Simple/Main/Advanced Profile
  - Multi-resolution (Dynamic resolution) is not processed inside of video decoder
- MPEG-4
  - Supports Simple/Advanced Simple profile except GMC.
  - Supports H.263 Baseline Profile
  - Support Divx from ver3.x to ver6.x
  - Supports Xvid
- MPEG-2
  - Fully compatible with ISO/IEC 13182-2 MPEG2 specification in Main Profile.
  - Support I,P and B frame
  - Support field coded picture (interlaced) and fame coded picture

- Value added features
  - MPEG-2 partial acceleration
  - Pre/Post rotator/mirror
  - Built-in de-blocking filter for MPEG-2/MPEG-4 and Divx
- Programmability
  - The VIDEO CODEC embeds 16-bit DSP processor dedicated to processing bitstream and controlling the codec hardware.
  - General purpose registers and interrupt for communication between a host processor and the video IP
- Interrupt
  - Interrupt from/to external host processor or interrupt controller
- Power related signal
  - Supports VPU_IDLE
  - Supports VPU_UNDERRUN

5.3 Clock and Reset

5.3.1 Clock

The VIDEO CODEC VPU uses 3 clock sources for the AXI bus (ACLK), the APB bus (PCLK), and video decoder module (CCLK), and 1 extra clock for system clock controller (RCLK). Based on the GALS (Globally Asynchronous Locally Synchronous) architecture of the IP, each clock source can be asynchronous to another. The duty cycles of all clocks are “don’t care” because the VIDEO CODEC VPU uses only rising edge of them..

5.3.1.1 AXI Bus Clock, aclk

The VIDEO CODEC VPU uses the AMBA AXI bus architecture as the system bus to interface with an external memory. All signals of the interface are output from an internal AXI register slice. The required frequency of the aclk is highly depends on the bus-loading, bandwidth, of user’s system.

5.3.1.2 Video Decoder Clock, cclk

The CCLK is for the video decoder module of the VIDEO CODEC VPU. VIDEO CODEC VPU decodes D1 30fps @ 30~50MHz. Below table shows the required frequencies of the cclk for a few examples.

Table 5.2 The Example Operating Frequency for Each Standard and Bitrate in D1

Standard	Operating Frequency	Bit Rate
MPEG-2 MP	40MHz	12Mbps
	35MHz	8Mbps
MPEG-4 ASP(Divx)	40MHz	5Mbps
	35MHz	4Mbps
H.264 HP	40MHz	4Mbps
VC-1 AP	40MHz	4Mbps

Note: This value is estimated by proprietary simulation environment.

5.3.1.3 APB Bus Clock, pclk

The PCLK is used for the APB bus. A host processor can communicate with the VIDEO CODEC VPU through the bus. The clock frequency does not have any relationship with other clock sources.

5.4 Frame Buffer

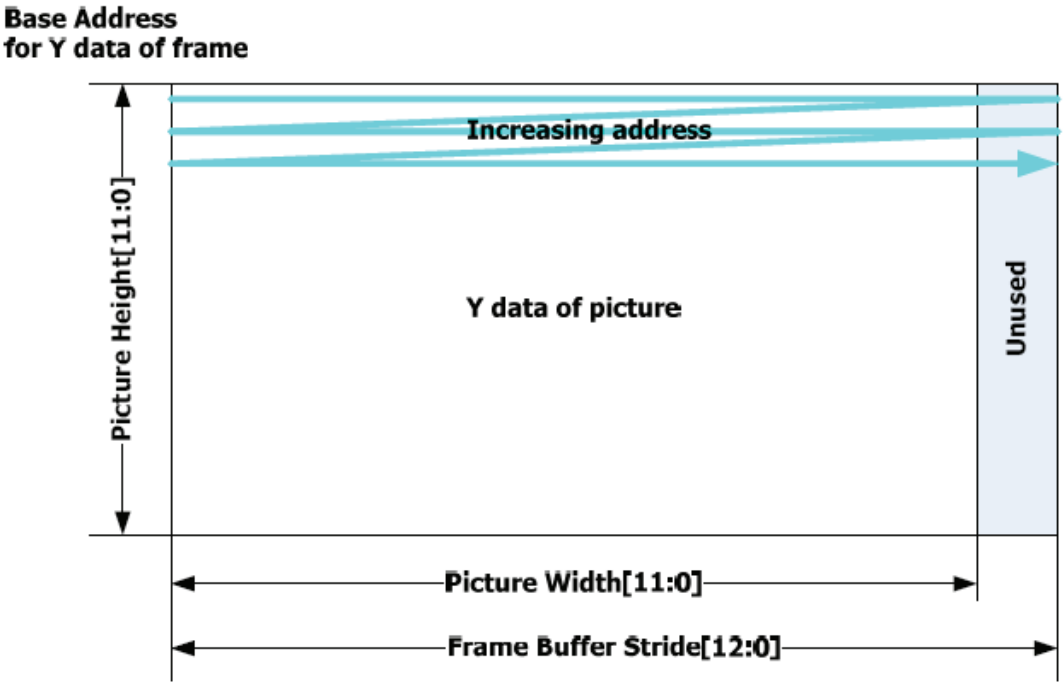
5.4.1 Memory Map

The source and decoded pictures (frames or fields) are each comprised of three sample arrays, one luma and two chroma sample arrays. In this document, the frame buffer means the sample array in an external memory. Following are the features of the frame buffer:

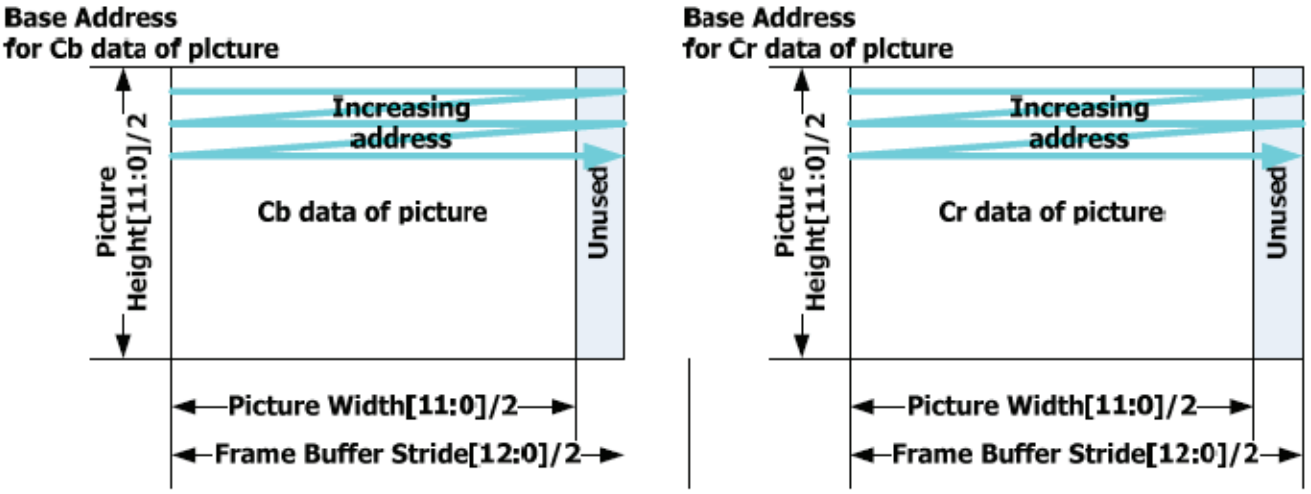
- A frame buffer is configured with following parameters:
  - Luminance frame buffer base address in 8-bye alignment
  - Cb frame buffer base address in 8-byte alignment
  - Cr fame buffer base address in 8-byte alignment: In case of the interleaved Cb/Cr map, a base address for the Cr is meaningless because a base address for the Cb is used to store or load the interleaved Cb/Cr samples.
  - Stride for the width of the luminance frame buffer: The stride should be equal or greater than the width of picture and multiple of 8-It means the least significant 3-bit of the 13-bit stride should be always 0. Stride of the chrominance frame buffers is the half of the stride for the luminance.
  - Endianess of the frame buffers
- The internal registers for specifying a frame buffer and a picture size are 12-bit width.
  - The stride for specifying the width of frame buffer in pixel unit
  - The picture size in pixel unit for both horizontal and vertical direction
- An endianess of the frame buffers is configurable as the little or the big endian.
- Both the interleaved Cb/Cr map and the separate Cb/Cr map are supported.
- A field pair is stored in a single frame buffer.

5.4.2 Mapping Picture to Frame Buffers

The Figure 5.1 shows how the frame buffer is configured. There are two kinds of configuration. The first configuration is const of (a) and (b). In the configuration, each chroma component has its own buffer. Therefore, there are two buffers for Cb and Cr components. The second configuration is consist of (a) and (c). This configuration is for reduction of bus-load using longer burst-length than the 1st configuration. As shown in the (c), the Cb and Cr samples are stored in the interleaved way.



(a) Frame buffer for luminance component



(b) Frame buffer for non-interleaved Cb & Cr components

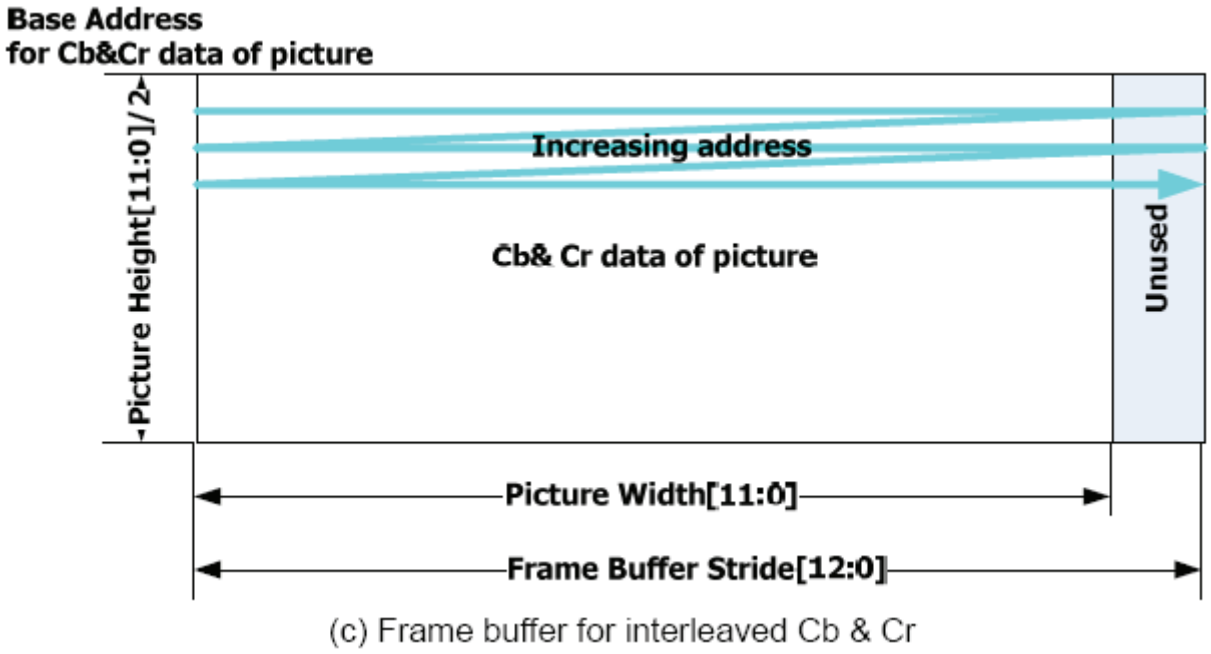


Figure 5.1 Frame Buffer Configuration

The following chapters give a detailed description for these configurations considering endianess by giving examples for QCIF(176x144) image.

5.4.3 Frames Buffers for Luminance Component

The Figure 5.2 shows how luminance samples are mapped to external memory. The base address in the figure is for luminance frame buffer.

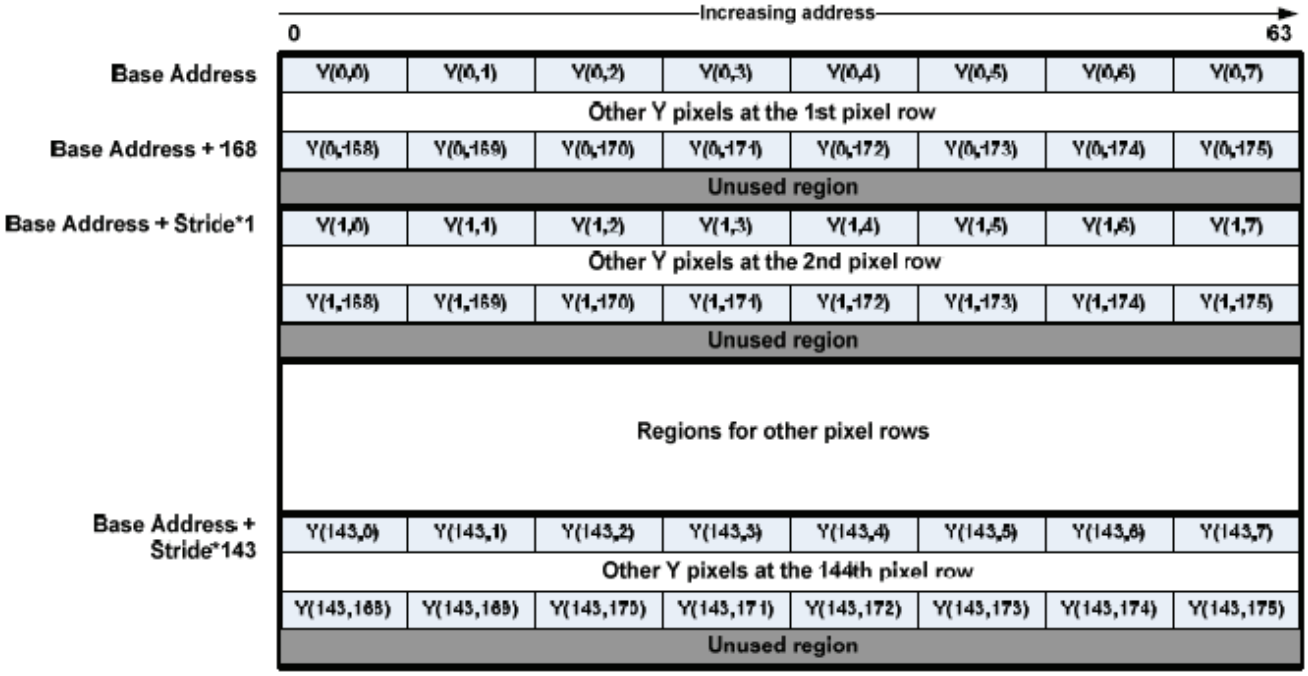


Figure 5.2 Luminance Pixel Arrangement in Frame Buffer

5.4.4 Frame Buffers for Chrominance Component

The Figure 5.3 shows how chrominance samples are mapped to separate Cb/Cr buffer of external memory. Each chrominance component has its own buffer.

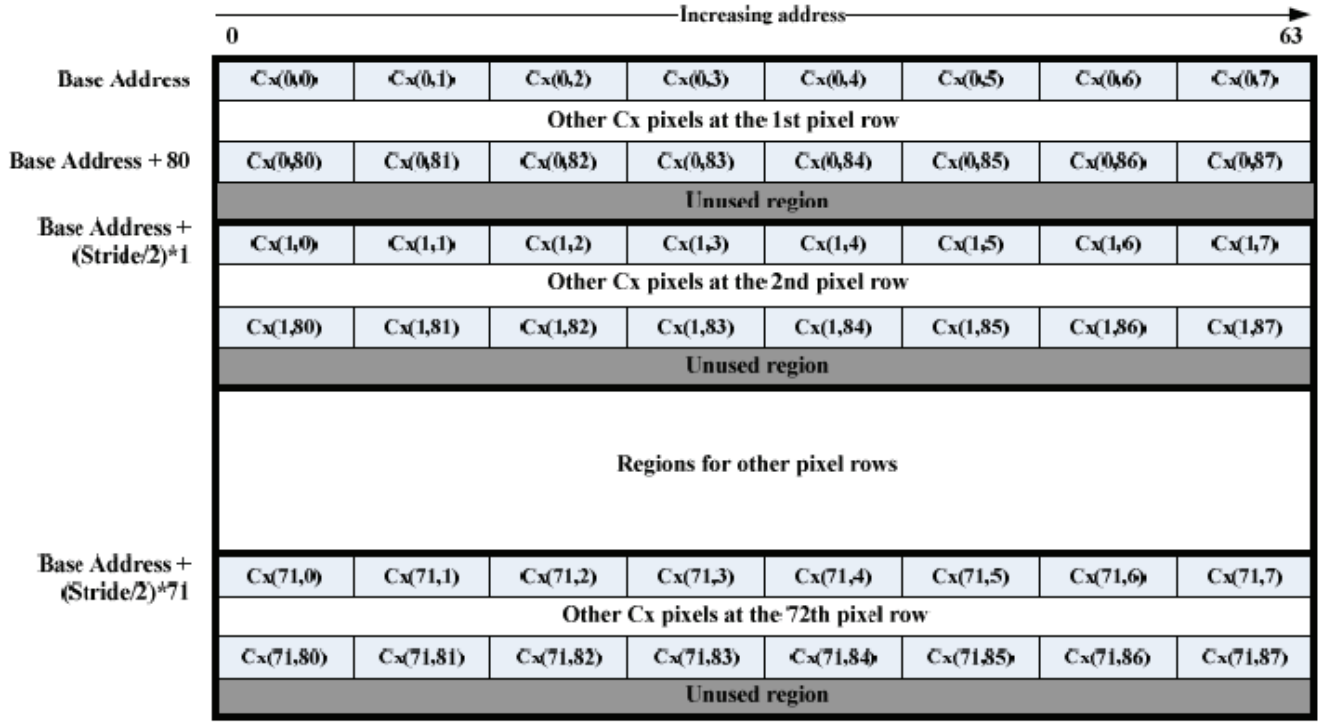


Figure 5.3 Chrominance Pixel Arrangement in Separate Cb/Cr Buffer

The Figure 5.4shows how chrominance samples are mapped to interleaved Cb/Cr buffer of external memory. In the interleaved Cb/Cr map, only one buffer is assigned for Cb and Cr of picture. Its stride is the same as the luminance's.

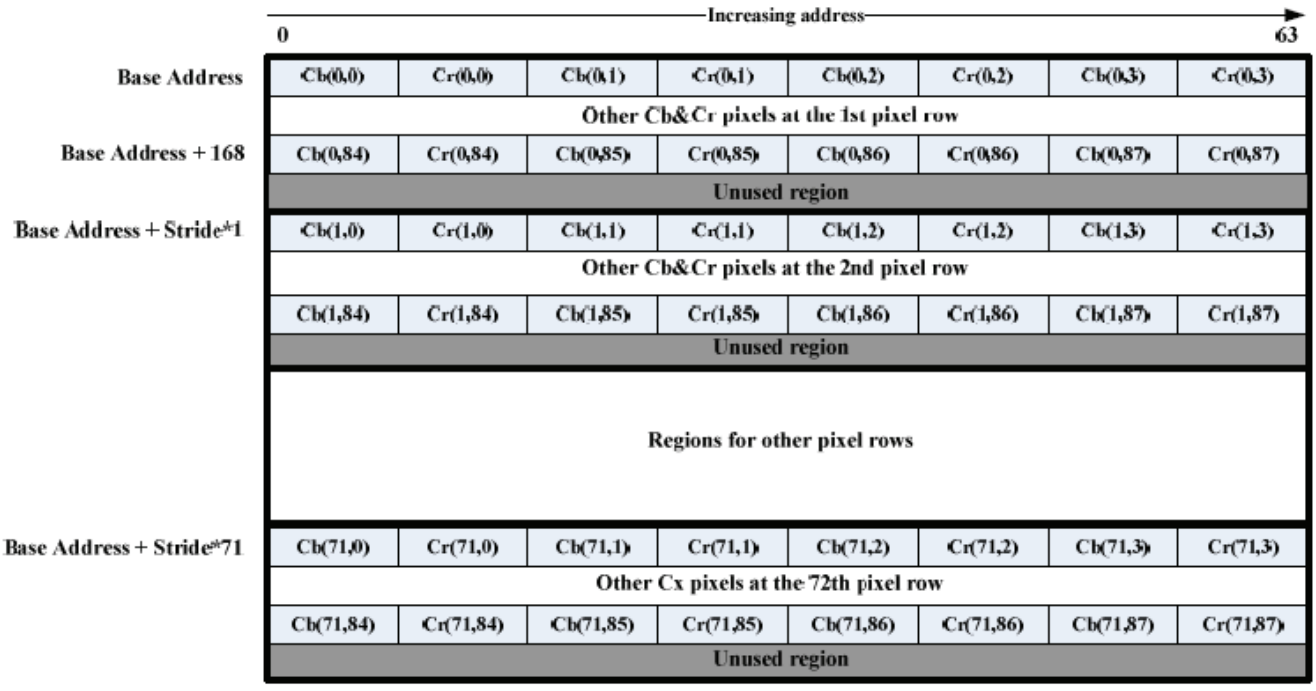


Figure 5.4 Chrominance Pixel Arrangement in Interleaved Cb/Cr Buffer

5.4.5 Frame Buffer Size in SDRAM

The required frame buffer size in SDRAM is different for each standard. Below table shows the required memory (SDRAM) size for each standard to support D1 (720x576). These values are the worst case which is defined by corresponding specification.

#		H.264	VC-1	MPEG-4	MPEG-2
1	Frame Buffer	7 frame (4,252.5 Kbyte)	7 frame (4,252.5 Kbyte)	7 frame (4,252.5 Kbyte)	7 frame (4,252.5 Kbyte)
2	Direct motion vector	708.75 Kbyte	25.4 Kbyte	25.4 Kbyte	
3	Overlap filter		7.1 Kbyte		
4	De-blocking filter	11.25 Kbyte	22.5 Kbyte		
5	Intra prediction(ACDC)	4.22 Kbyte	5.625 Kbyte	5.625 Kbyte	
6	MVP/MB information	5.625 Kbyte	2.11 Kbyte	2.11 Kbyte	
7	Slice information	131 Kbyte			
8	Bit Plane		3 Kbyte		
9	Data-partitioning			96 Kbyte	
	Total	5 Mbyte	3.6 Mbyte	2.48 Mbyte	2.38 Mbyte

The above table shows the SDRAM requirement for each standard. The SDRAM requirement for number 3, 4, 5, and 6 can be removed if user uses secondary AXI for internal SRAM.

Note : picX=720, picY=576
mbX (Number of macroblock in row) =45.
mbY (Number of macroblock in column) =36.
1 frame for D1 is 720 x 576 x 1.5 = 622,080 byte.

6 Video Cache

6.1 Resister Map

Table 6.1 Video Cache Registers (Base Address = 0xB0910000)

Name	Addr	Type	Mode	Reset	Description
VCCTRL	0x000	R/W	-	0x00000000	Video cache control register
VCREG	0x004	R/W	-	0x00000000	Video cache read/write register
VWBCTRL	0x008	R/W	-	0x00000000	Video cache write valid register
R0MIN	0x024	R/W	-	-	Video cache read reason 0 minimum register
R0MAX	0x028	R/W	-	-	Video cache read reason 0 maximum register
R1MIN	0x02c	R/W	-	-	Video cache read reason 1 minimum register
R1MAX	0x030	R/W	-	-	Video cache read reason 1 maximum register
R2MIN	0x034	R/W	-	-	Video cache read reason 2 minimum register
R2MAX	0x038	R/W	-	-	Video cache read reason 2 maximum register
R3MIN	0x03c	R/W	-	-	Video cache read reason 3 minimum register
R3MAX	0x040	R/W	-	–	Video cache read reason 3 maximum register

VCCTRL **0xB0910000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PWRASVE	-														
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
-														INVAL	ON

Field	Name	RW	Reset	Description
0	ON	R/W	0x00000000	Cache On
1	INVAL	R/W	0x0	Prefetch Cache Invalidate All
31	PWRASVE	R/W	0x00000000	Not-used

VCREG **0xB0910004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
		RD3	RD3			WR2	RD2			WR1	RD1			WR0	RD0

Field	Name	RW	Reset	Description
13	WR3	R/W		Write region 3 enable.
12	RD3	R/W		Read region 3 enable.
9	WR2	R/W		Write region 2 enable.
8	RD2	R/W		Read region 2 enable.
5	WR1	R/W	0x00000000	Write region 1 enable.
4	RD1	R/W		Read region 1 enable.
1	WR0	R/W		Write region 0 enable.
0	RD0	R/W		Read region 0 enable.

VWBCTRL **0xB0910008**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TVALUE															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										WCBV		WCRST	DRAINST	TEN	DRAIN

Field	Name	RW	Reset	Description
31-16	TVALUE	R/W	0x00000000	Timeout value.
5	WCBV	R/W		Write Access Wait Control Option Register. 0 : Wait until the data stored to the external memory such as DRAM. 1 : Wait until the data stored to the write buffer. Normally, the write time can faster than case 0.
3	WCRST	R/W		Write Cache Reset. 0 : Not-Reset 1 : Reset
2	DRAINST	R		Read only drain status.
1	TEN	R/W		Timeout enable. 0 : Write buffer is not drained when timeout. 1 : Write buffer will be drained when timeout.
0	DRAIN	R/W		Write Buffer Drain enable 0 : not-drain 1 : forced drain

R0MIN **0xB0910024**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0 MIN															

Field	Name	RW	Reset	Description
31-0	R0MIN	R/W	-	Lower bound address for region 0

R0MAX **0xB0910028**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0 MAX															

Field	Name	RW	Reset	Description
31-0	R0MAX	R/W	-	Upper bound address for region 0

R1MIN **0xB091002C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R1 MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R1 MIN															

Field	Name	RW	Reset	Description
31-0	R1MIN	R/W	-	Lower bound address for region 1

R1MAX **0xB0910030**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R1 MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R1 MAX															

Field	Name	RW	Reset	Description
31-0	R1MAX	R/W	-	Upper bound address for region 1

R2MIN **0xB0910034**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R2 MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R2 MIN															

Field	Name	RW	Reset	Description
31-0	R2MIN	R/W	-	Lower bound address for region 2

R2MAX **0xB0910038**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R2 MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R2 MAX															

Field	Name	RW	Reset	Description
31-0	R2MAX	R/W	-	Upper bound address for region 2

R3MIN **0xB091003C**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R3 MIN															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R3 MIN															

Field	Name	RW	Reset	Description
31-0	R3MIN	R/W	-	Lower bound address for region 3

R3MAX **0xB0910040**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R3 MAX															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R3 MAX															

Field	Name	RW	Reset	Description
31-0	R3MAX	R/W	-	Upper bound address for region 3

7 Video Bus Configuration

Power Down Mode Value **0xB0920000**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-												PDN3	PDN2	-	PDN0

Field	Name	RW	Reset	Description
31-4	-	-	-	Reserved
3	PDN3	W	0x0	VIDEO CACHE power down 0 : Bus Clock for the Video Cache Clock is Enabled 1 : Bus Clock for the Video Cache Clock is Disabled
2	PDN2	W	0x0	VIDEO codec power down 0 : Bus Clock for the Video Codec Clock is Enabled 1 : Bus Clock for the Video Codec Clock is Disabled ***The core clock for the video codec can be disabled in the CKC register.***
1	-	-	-	Reserved
0	PDN1	W	0x0	JPEG Encoder power down 0 : Bus Clock for the JPEG Encoder is Enabled 1 : Bus Clock for the JPEG Encoder is Disabled

Soft Reset Register **0xB0920004**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-												SWR3	SWR2	-	SWR0

Field	Name	RW	Reset	Description
31-4	-	-	-	Reserved
3	SWR3	W	0x0	VIDEO CACHE soft reset 0 : Not-Reset, 1 : Reset
2	SWR2	W	0x0	VIDEO codec soft reset 0 : Not-Reset, 1 : Reset
1	-	-	-	Reserved
0	SWR1	W	0x0	JPEG Encoder soft reset 0 : Not-Reset, 1 : Reset

PART9 – CAMERA BUS
NVS2310

Rev. 1.02

Jun 01, 2011

Revision History

Date	Revision	Description
2010-11-22	1.00	∗ Initial Release
2011-04-22	1.01	∗ Correct the register table and document format.
2011-06-01	1.02	∗ Correct the mis-spelled chip name.

TABLE OF CONTENTS

Contents

Figures

Tables

1 Introduction

NVS2310 CAMERA BUS includes Camera Interface and ISP (Image Signal Processor).

[FEATURES]

- CAMERA INTERFACES
  - CCIR 601/656, YUV4:2:2 (8bits), RGB555, RGB565
  - Support Interlace mode
  - Image Effector
  - Hardware Scaler
  - 4-Level Image Overlay, Chroma-Keying & Scaler for Preview
  - Various Input Format : YCbCr 4:2:2/4:2:0, RGB 4:2:2, 4:2:0, Bayer RGB
  - Color Space Dispatcher
  - Up-to-5M Pixels with Scaling
- ISP
  - Post-processing
    - Color Processing
    - Image Effects
    - Super Impose
    - Resize (Scaler)
    - Memory Interface
    - JPEG Encoder

2 Bus Architecture

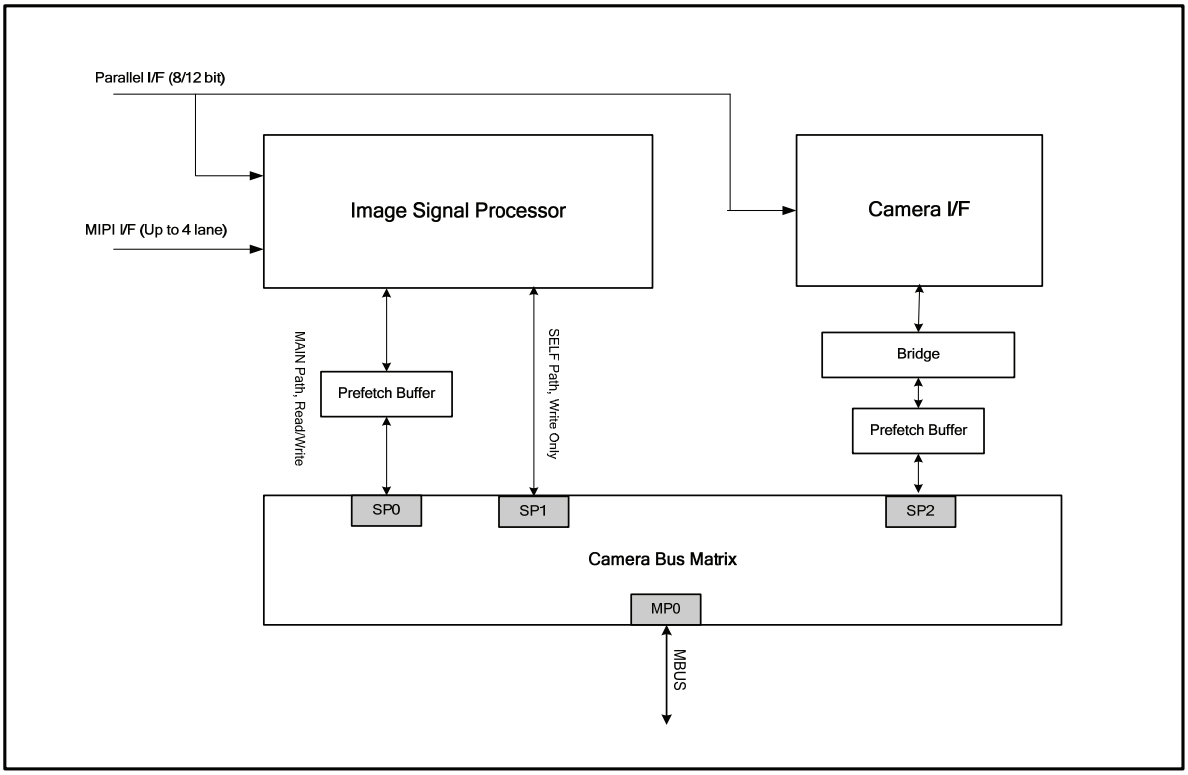


Figure 2.1 The Camera Bus Architecture

3 Address and Register Map

Refer to corresponding sections for detail information of each peripheral.

Base Address		Peripherals
0xB0200000		CIF
0xB0210000		ISP
0xB0220000		Camera Bus Matrix
0xB0230000		CAMBUS CFG

4 Camera Interface

4.1 Overview

The NVS2310 has the camera interface (CIF). Its features are as follows.

INPUT FORMATS
- CCIR 601/656, YUV4:2:2 (8bits)
- Support Interlace mode
- RGB555, RGB565 (8 bits)

OUTPUT FORMATS
- YUV4:2:2, YUV4:2:0
- Convert YUV422 to YUV420 mode

Skip Frame Mode

4-level alpha blending/Chroma-Keying (1 overlay image)

Image effecter
- Bias
- Inversion of Y value
- Strong C mode
- Y clipping
- Color filter
- Sketch mode
- Embossing
- Gray
- Sepia

Image scaler (ratio : original * 256/target)
- Upscale: 1 : 4
- Downscale: 64 : 1
- Each step size is 256 step

4.2 Operation

The block diagram of CIF is shown in Figure 4.1.

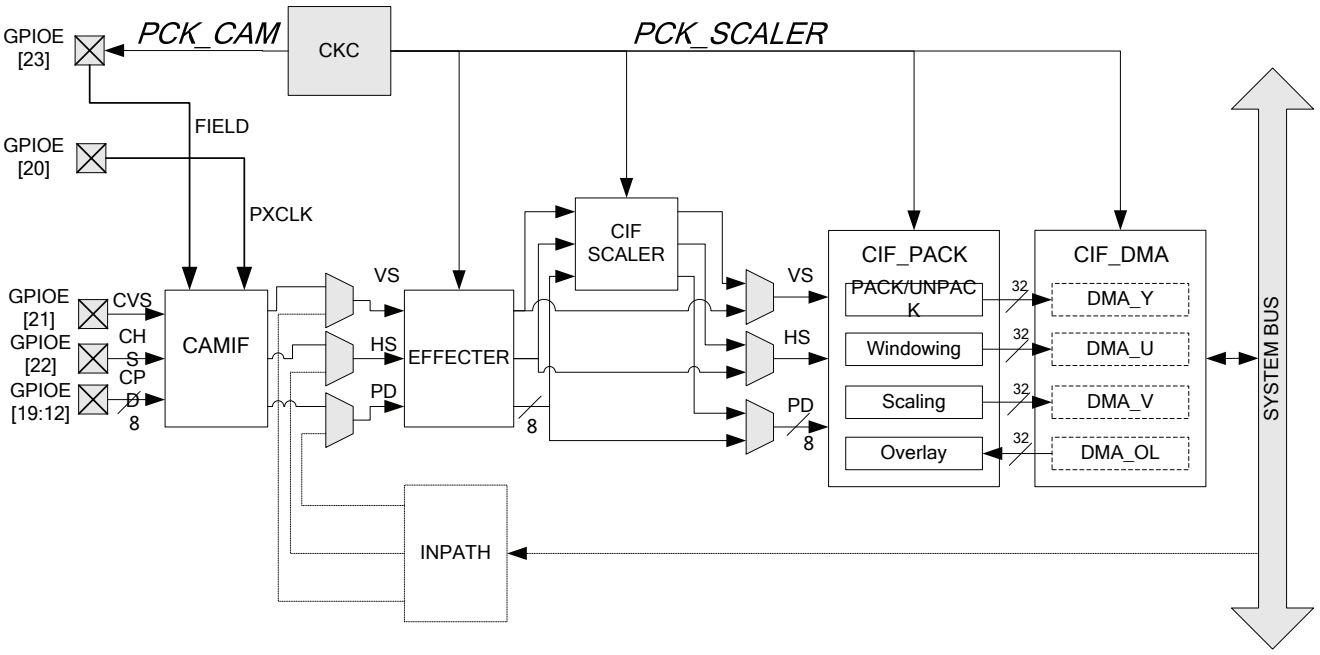


Figure 4.1 CIF Block Diagram

4.2.1 External Camera Module Interface (CAMIF)

The CIF uses CCIR-601/656-like protocol to interface with an external camera module. These signals are shown in the Figure 4.2.

The C656 bit of ICPCR1 register determines whether CPD[7:0] includes the embedded sync information or not. If the sync information is embedded in the pixel data stream, the additional sync signals, which are CVS and CHS, are not used. In this case, 656FCR1 and 656FCR2 register need to be configured. Otherwise, CVS and CHS are used for the vertical synchronous signal and the horizontal synchronous signal respectively. If the sync information is in the CVS and CHS, TV and VI bit of ICPCR1 register can be used to adjust the misaligned vertical (CVS) and horizontal sync (CHS).

When the format of input data is RGB565 or RGB555, it should be converted to YUV. Refer to CR2Y register.

PXCLK in Figure 4.1 is from a camera module. If a camera module needs the external clock input, CCKO can be used as input to one. Refer to CKC_CTRL..

The POL bit of ICPCR1 register determines whether pixel data are latched at the rising edge or the falling edge of PXCLK. And CVS and CHS polarity are determined by the VSP and the HSP bit of ICPCR1 register respectively.

The geometric property of input image is determined by CEIS register (Figure 4.3).

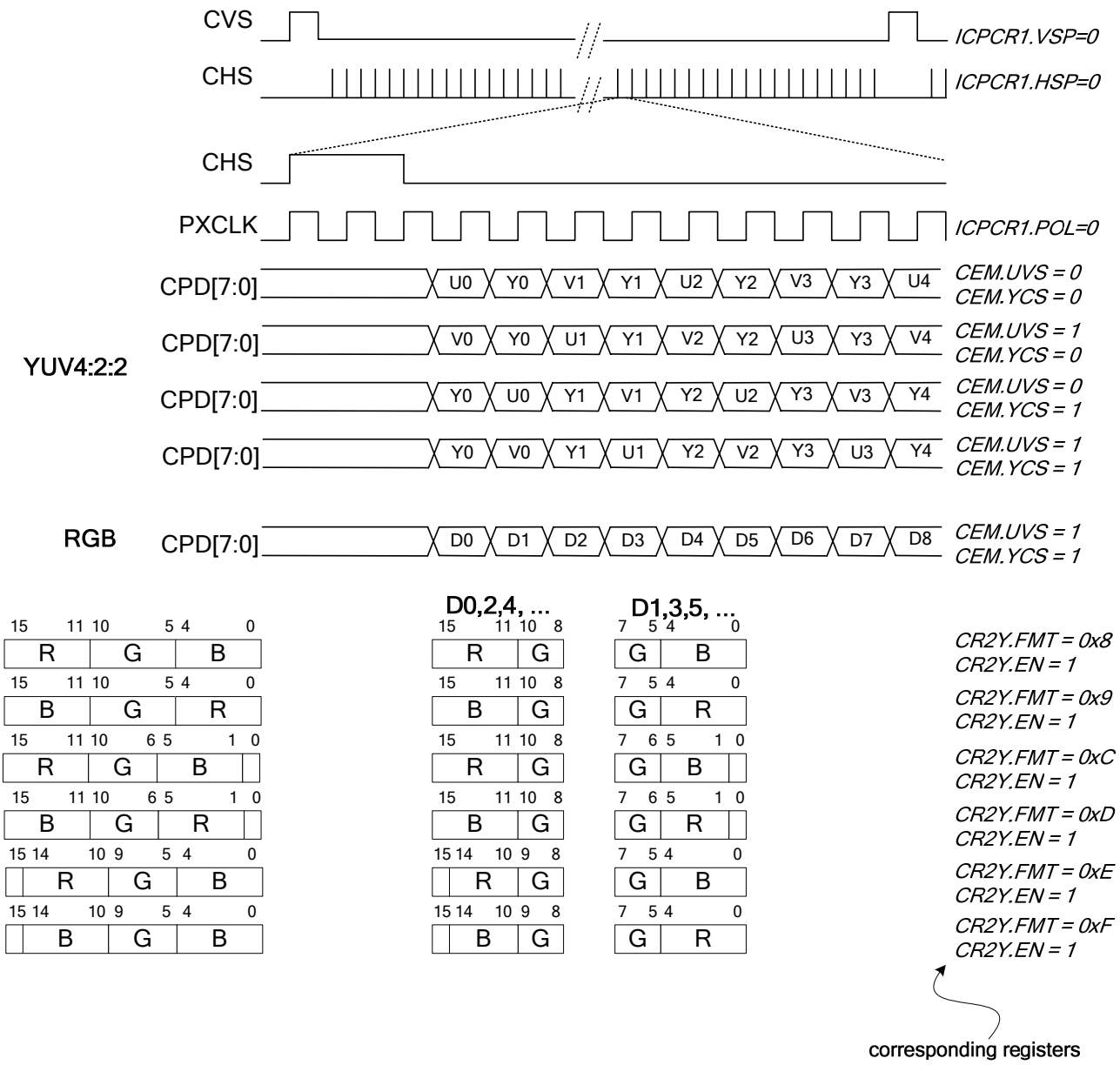


Figure 4.2 Input Data Format From the external camera module

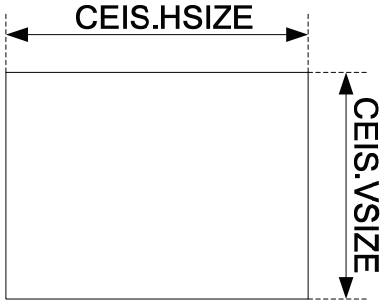


Figure 4.3 Input Image Size From the External Camera Module

In interlace mode, field signal is generated from input port signals, FIELD, HS(not same as data enable signal). PCK_CAM and FIELD signal shares the same port of NVS2310.(Figure 4.4)

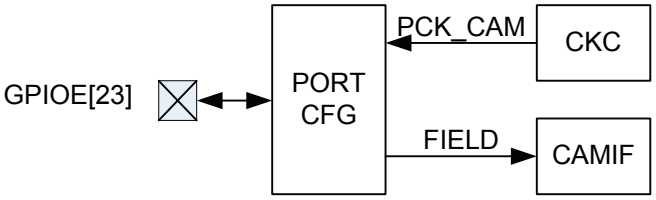
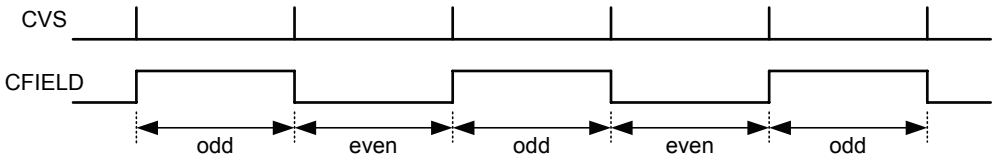
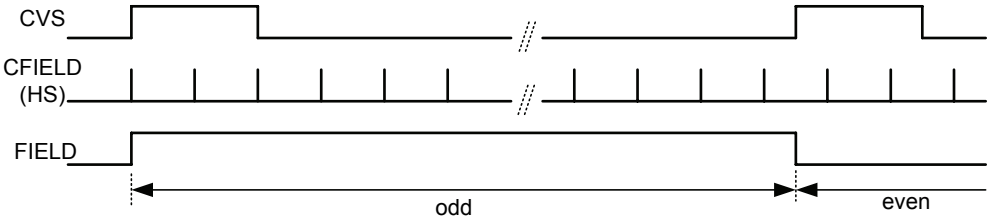


Figure 4.4 Port Control in PCK_CAM and FIELD

Field signal generation method is shown as Figure 4.5. In case Figure 4.5(a), CFIELD port input is directly used as field signal. In case (b), according to the level of CFIEID port input, HS signal(not same as CHS which indicates data enable time, and preserves it's value during vertical blanking), the field signal is generated at the positive edge of VS.



(a) CFIELD port is field signal



(b) Generate field signal when CFIELD port is HS signal
- HS is not data enable signal

Figure 4.5 Generate Field Signal

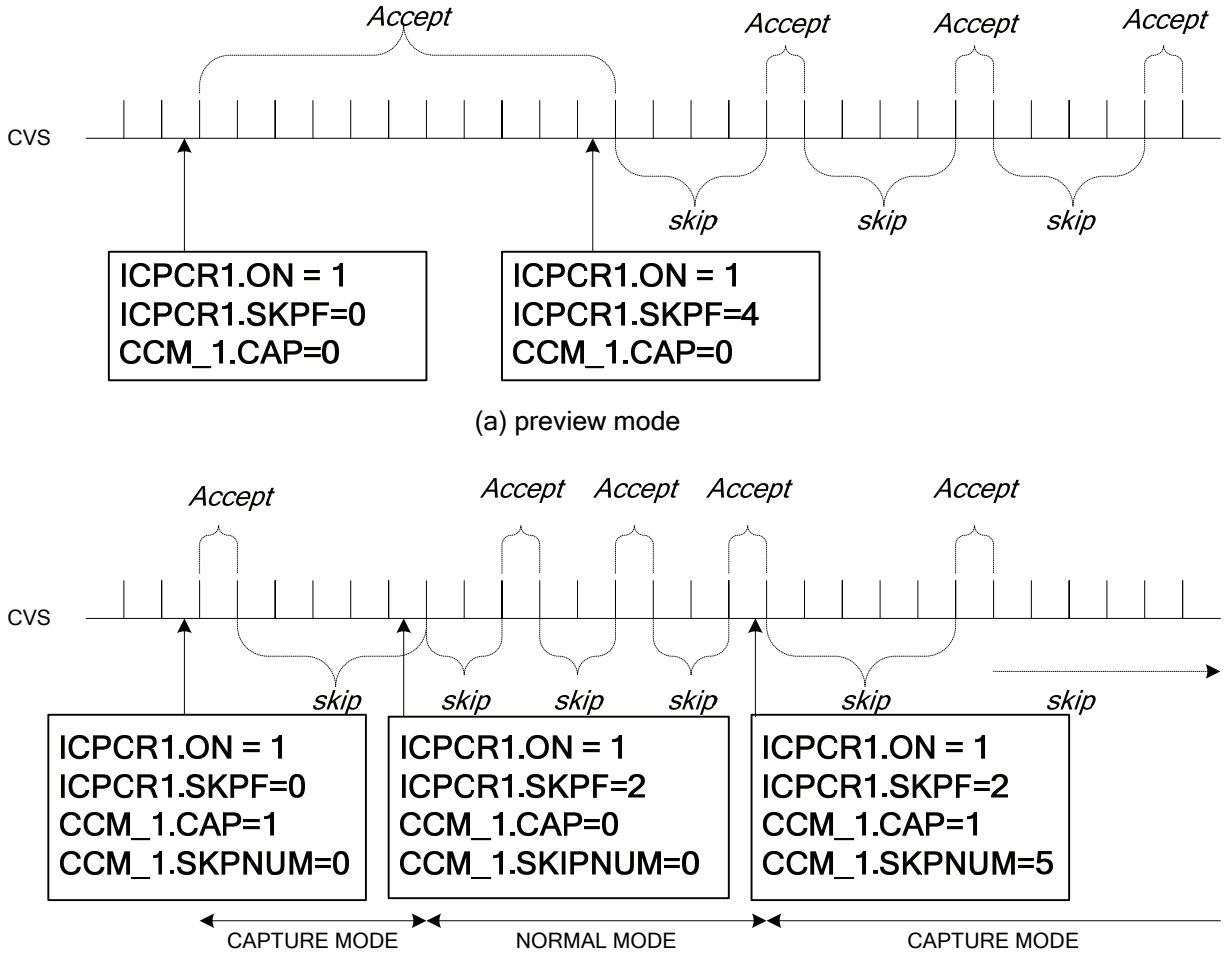


Figure 4.6 Accepting Frame Data from the External Camera Module

The CIF has two modes to accept one frame data from the external camera module. One is the preview mode. The other is the capture mode. The preview mode accepts the successive frame data from the external camera module. But, the capture mode accepts only one frame data.

Although the camera module sends the successive frame data to the CIF, the CIF can discard several frames before accepting a frame. They are called the skip frames. When the CIF is in the preview mode, the skip frames are determined by SKPF bits of ICPCR1register. When the CIF is in the capture mode, the skip frames are determined by SKIPNUM bits of CCM_1 register. Refer to ICPCR1 register on page 4-15 and CCM_1 register on page 4-28. Figure 4.6 shows the difference between the preview mode and the capture mode.

The accepted frames are sent to the effecter and their pixel format is YUV 4:2:2.

4.2.2 Effecter

The effecter supports YUV bias (YUV offset value), Inversion of Y value, strong C mode (x2 C value), Y clipping, color filter, sketch mode, embossing (positive and negative), gray, and sepia. Refer to the effecter registers on page 4-31.

4.2.3 CIF Scaler

The NVS2310 provides the CIF scaler for scaling the image from the effecter. The supported scaling ratio is from 1 : 4 (zoom up to 4 times) to 64 : 1 (zoom down to 64 times), and scaling step is 256.

The output image of the effecter becomes the CIF scaler input. Figure 4.7 shows relationship between them and how to specify them. Refer to CSSO and CSSS register on page 4-34. When the BPS bit of ICPCR1 register is set to 1, the effecter output is not scaled by the CIF scaler.

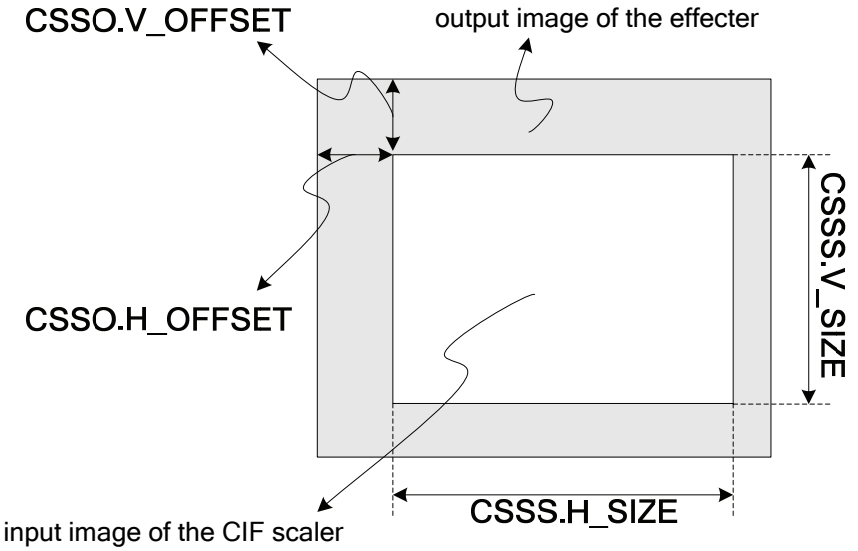


Figure 4.7 Output Image of the Effecter and Input Image of the CIF Scaler

4.2.4 Overlay

The CIF scaler output image (If the CIF scaler is disabled, it is equal to the effecter output image) can be mixed with another image in the memory. This image in the memory is called the overlay image and the CIF scaler output image is called the background image. For this operation, the CIF has the alpha-blending and the chroma-keying function.

Figure 4.8 shows how to specify the geometric property of overlay image. Refer to OIS, OIW1, and OIW2 register on page 4-26. If the size of overlay image (not active overlay image) is equal to the size of background image, the OM bit of OCTRL1 should be set to 0 (full image overlay). Otherwise, it should be set to 1 (block image overlay).

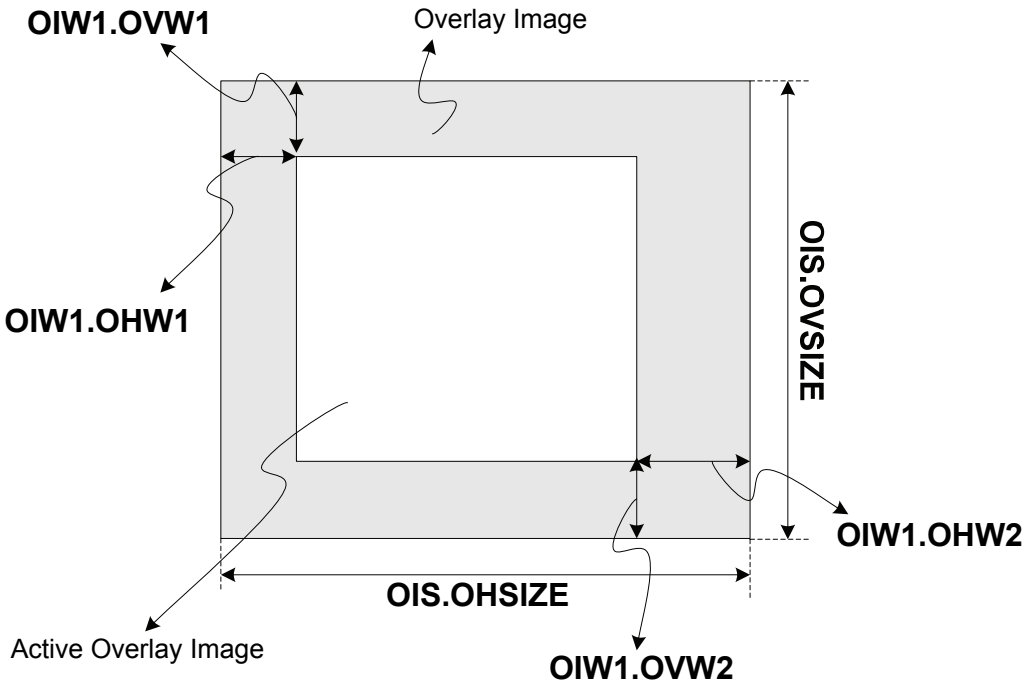


Figure 4.8 The Geometric Property of Overlay Image

4.2.5 Store to the Memory

The image data which is from the camera module through effecter and CIF scaler to overlay logic need to be stored to the memory. Figure 4.9 shows how to specify the image data to be stored. Refer to IIS, IIW1, and IIW2 register on page 4-18.

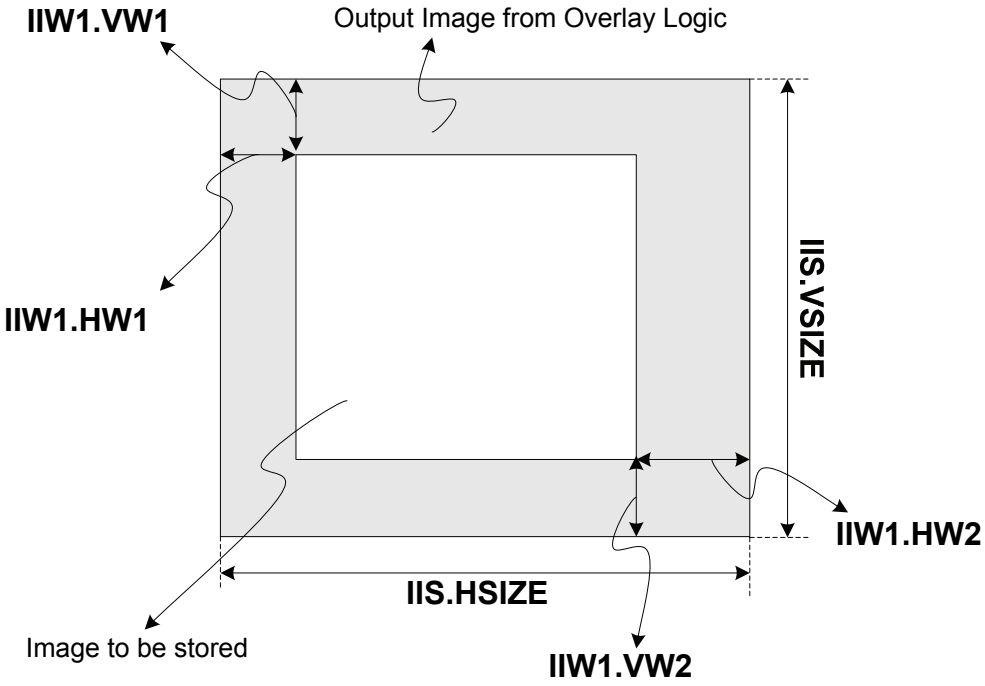


Figure 4.9 Image Data to be Stored to the Memory

The image data from the overlay logic is YUV4:2:2. The image format to be stored can be YUV4:2:2SEQ0, YUV4:2:2, or YUV4:2:0. It is determined by the BP and the M420 bit of ICPCR1 register. Refer to the ICPCR1 register on page 4-15.

The CIF has two modes to store the image data. One is the frame mode and the other is the rolling mode. The frame mode needs the memory space to store a whole frame data. But, the rolling mode needs the memory space to store the part of a frame data. It is determined by the RLY, RLU, and RLV bits of CCM_1. Therefore, in the rolling mode, before the frame data are overwritten, they should be moved to another memory space.

CDCR2, CDCR3, and CDCR4 registers are the base addresses which the image data is stored to. They are for Y, U, and V image data respectively. Additionally, the end addresses are required in the rolling mode. They are determined by CDCR5, CDCR6, and CDCR7 register. Refer to these registers on page 4-19.

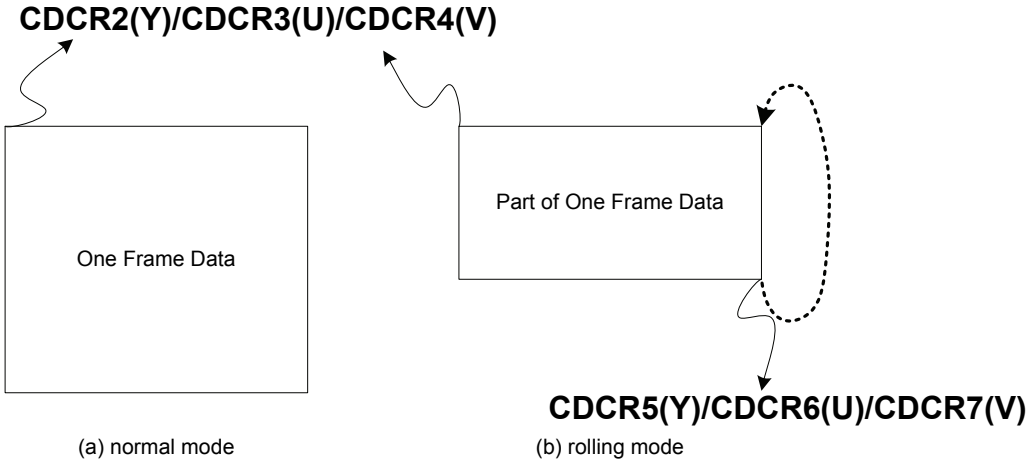


Figure 4.10 Two Modes to Store the Image Data

The CIF can inform the on-chip CPU its status during storing image data. When the CIF has stored a whole frame data to

the memory, SOF of CIRQ register is set to 1. SCF and SOF of CIRQ register is the same function, but SCF is only available in the capture mode. In the rolling mode, ROLY, ROLU, and ROLV of CIRQ register represent whether the current address which is used to store the image data reaches the corresponding “end address” or not. ENS of CIRQ register is similar to ROLY, U, V but the current address is compared with CESA register. VIT of CIRQ register is set to 1 whenever the number of lines which the CIF receives data from the camera module and stores them to specified memory is equal to the multiple of 16*VCNT lines. If VCNT of CCM_2 register is set to 0, VIT bit is never set to 1. All these status bits can be the interrupt request source. Refer to CIRQ on page 4-22.

4.3 Camera Register Descriptions

Table 4.1 CIF Register Map (Base Address = 0xB0200000)

Name	Address	Type	Reset	Description
ICPCR1	0x00	W/R	0x00000000	Input Image Color/Pattern Configuration Register 1
656FCR1	0x04	W/R	0x06ff0000	CCIR656 Format Configuration Register 1
656FCR2	0x08	W/R	0x010b	CCIR656 Format Configuration Register 2
IIS	0x0C	W/R	0x00000000	Input Image Size
IIW1	0x10	W/R	0x00000000	Input Image Windowing 1
IIW2	0x14	W/R	0x00000000	Input Image Windowing 2
CDCR1	0x18	W/R	0x0003	DMA Configuration Register 1
CDCR2	0x1C	W/R	0x00000000	DMA Configuration Register 2
CDCR3	0x20	W/R	0x00000000	DMA Configuration Register 3
CDCR4	0x24	W/R	0x00000000	DMA Configuration Register 4
CDCR5	0x28	W/R	0x00000000	DMA Configuration Register 5
CDCR6	0x2C	W/R	0x00000000	DMA Configuration Register 6
CDCR7	0x30	W/R	0x00000000	DMA Configuration Register 7
CDCR8	0x34	W/R	0x00000000	DMA Configuration Register 8
FIFOSTATE	0x38	R	0x00000000	FIFO Status Register
CIRQ	0x3C	W/R	0x00000000	Interrupt & Status register
OCTRL1	0x40	W/R	0x37000000	Overlay Control 1
OCTRL2	0x44	W/R	0x00000000	Overlay Control 2
OCTRL3	0x48	W/R	0x00000000	Overlay Control 3
OCTRL4	0x4C	W/R	0x00000000	Overlay Control 4
OIS	0x50	W/R	0x00000000	Overlay Image Size
OIW1	0x54	W/R	0x00000000	Overlay Image Windowing 1
OIW2	0x58	W/R	0x00000000	Overlay Image Windowing 2
COBA	0x5C	W/R	0x00000000	Overlay Base Address
COBO	0x60	W/R	0x00000000	Overlay Base Address Offset
CDS	0x64	W/R	0x00000000	Camera Down Scaler
CCM1	0x68	W/R	0x00000000	Capture Mode Configuration 1
CCM2	0x6C	W/R	0x00000000	Capture Mode Configuration 2
CESA	0x70	W/R	0x00000000	Point Encoding Start Address
CR2Y	0x74	W/R	0x00000000	RGB2YUV Format converter Configuration
CCYA	0x78	R	-	Current Y Address
CCYU	0x7C	R	-	Current U Address
CCYV	0x80	R	-	Current V Address
CCLC	0x84	R		Current Line count

Input Image Color/Pattern Configuration Register 1 (ICPCR1) 0xB0200000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ON	-	TV	VI	UF	ITEN	FS	FP	BPS	0	POL	SKPF			M420	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BP	-	C656	CP	PF<1:0>		RGBM<1:0>		RGBBM<1:0>		CS<1:0>		0	BO	HSP	VSP

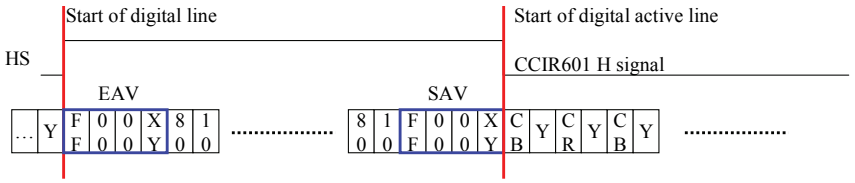
Field	Name	RW	Reset	Description
31	ON	R/W	0x0	0 : Disable CIF 1 : Enable CIF
29	TV	R/W	0x0	0 : CIF sync signal 1 : TV sync signal
28	VI	R/W	0x0	0 : Do not inset vertical blank in even field 1 : Insert vertical blank in 1 line in even field
27	UF	R/W	0x0	In interlace mode, it is decided whether field signal is used or not. The field is determined from direct field input signal or relationship between VS and HS. Also the field input port must be connected with external field input signal(field or HS) obviously. 0 : Don't use 1 : Use field signal
26	ITEN	R/W	0x0	This field must be set in case of interlace input mode 0 : Disable 1 : Enable
25	FS	R/W	0x0	Select field port input between field signal and HS, when the field signal is used in interlace mode. 0 : Field signal 1 : Horizontal sync
24	FP	R/W	0x0	0 : Active low (odd field – high, HS - active low) 1 : Active high (odd field – low, HS – active high)
23	BPS	R/W	0x0	0 : CIF scaler is used. 1 : Bypass. CIF scaler is not used.
21	POL	R/W	0x0	0 : Rising edge 1 : Falling edge.
20-18	SKPF	R/W	0x0	0 : Not-skipped 1-7 : Number of frames to be skipped.
17-16	M420	R/W	0x0	00 : Not-Convert 10 : Odd lines of U and V image are skipped. 11 : Even lines of U and V image are skipped.
15	BP	R/W	0x0	0 : Data format to be stored to memory If M420 bit is set to 0, the data format is YUV 4:2:2. Otherwise, the data format is YUV4:2:0. 1 : YUV4:2:2SEQ0
13	C656	R/W	0x0	0 : Disable 1 : Enable
12	CP	R/W	0x0	0 : It should be set to 0
11-10	PF	R/W	0x0	01 : "01b" SHOULD BE WRITTEN TO THIS BIT.
9-8	RGBM	R/W	0x0	00 : It should be set to 0.
7-6	RGBBM	R/W	0x0	00 : It should be set to 0.
5-4	CS	R/W	0x0	00 : It should be set to 0.
2	BO	R/W	0x0	1 : Switch the MSB/LSB 8bit bus. (Don't' change)
1	HSP	R/W	0x0	0 : Active low 1 : Active high (default)
0	VSP	R/W	0x0	0 : Active low (default) 1 : Active high

CCIR656 Format Configuration Register 1 (656FCR1) 0xB0200004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					PSL<1:0>		0	FPV<7:0>							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SPV<7:0>								TPV<7:0>							

This register and next register (656FCR1 and 656FCR2) define the configuration of CCIR656.

Field	Name	RW	Reset	Description
26-25	PSL	R/W	0x3	We must find the location of preamble and status for getting sync information. The total size of preamble and status is 4 bytes composed of 3 bytes of preamble and 1 byte of status. This register is used to find the location of status word. 00 : The status word is located in the first byte of EAV & SAV 01 : The status word is located in the second byte of EAV & SAV 10 : The status word is located in the third byte of EAV & SAV 11 : The status word is located in the forth byte of EAV & SAV
23-16	FPV	R/W	0x0	First preamble value Define the first preamble value. Default value is 0x00.
15-8	SPV	R/W	0x0	Second preamble value Define the second preamble value. Default value is 0x00.
7-0	TPV	R/W	0x0	Third preamble value Define the third preamble value. Default value is 0x00.



	8-bit data							
	D7 (MSB)	D6	D5	D4	D3	D2	D1	D0
preamble	1	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0
status	1	F	V	H	P3	P2	P1	P0

•Status word define
- •F='0' for field 1, '1' for field 2 (interlace mode. If progressive, this value is '0')
- •V='1' during vertical blanking
- •H='0' at SAV, '1' at EAV

•Protection bits
- •P3=V xor H
- •P2=F xor H
- •P1=F xor V
- •P0=F xor V xor H

Figure 4.11 CCIR-656 Format Diagram

CCIR656 Format Configuration Register 2 (656FCR2) 0xB0200008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		FB<3:0>					HB<3:0>				0	VB<3:0>			

Field	Name	RW	Reset	Description
13-10	FB	R/W	0x0	field information In status word, it refer to the location of ‘F’ and F valie at field imformation. The MSB 3 bit means the location of ‘F’, the other bit means value at field imformation. Default value is 0x00. but, CCIR656, this value is 0x0c.
8-5	HB	R/W	0x8	Horizontal blank In status word, it refers to the location of ‘H’ and H value at blanking. The MSB 3 bit means the location of ‘H’, the other bit means value at blanking. Default value is 0x09.
3-0	VB	R/W	0xb	Vertical blank In status word, it refers to the location of ‘V’ and V value at blanking. The MSB 3 bit means the location of ‘V’, the other bit means value at blanking. Default value is 0x0B.

Input Image Size (IIS) 0xB020000C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HSIZE <15:0>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VSIZE <15:0>															

Field	Name	RW	Reset	Description
31-16	HSIZE	R/W	0x0	Horizontal size of input image
15-0	VSIZE	R/W	0x0	Vertical size of input image

If the CIF scaler is used, this should be the same as CSDS register. Otherwise, this should be the same as CEIS register

Input Image Windowing1 (IIW1) 0xB0200010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
HW1<15:0>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HW2<15:0>															

Field	Name	RW	Reset	Description
31-16	HW1	R/W	0x0	-
15-0	HW2	R/W	0x0	-

Input Image Windowing2 (IIW2) 0xB0200014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VW1<15:0>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VW2 <15:0>															

Field	Name	RW	Reset	Description
31-16	VW1	R/W	0x0	-
15-0	VW2	R/W	0x0	-

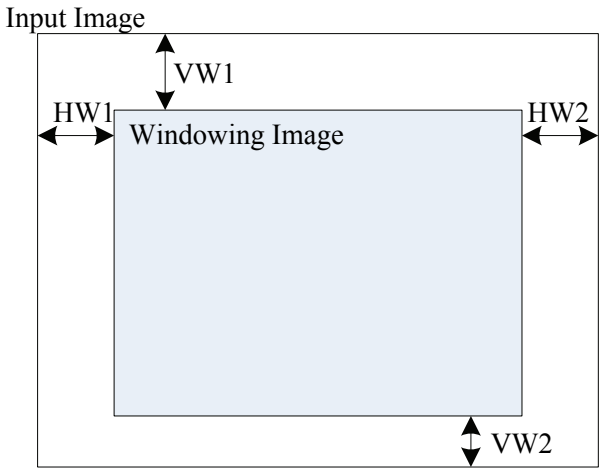


Figure 4.12 Input Image Windowing

Default value of HW1, HW2, VW1, and VW2 is 0.

CIF DMA Configuration Register 1 (CDCR1) 0xB0200018

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
												TM	Lock	BS <1:0>	

Field	Name	RW	Reset	Description
3	TM	R/W	0x0	0 : Burst Transfer(default) 1 : INC Transfer
2	LOCK	R/W	0x0	0 : Non-Lock (default) 1 : Lock Transfer
1-0	BS	R/W	0x3	00 : The DMA transfers the image data as 1 word to memory. 01 : The DMA transfers the image data as 2 words to memory. 10 : The DMA transfers the image data as 4 words to memory. 11 : The DMA transfers the image data as 8 words to memory. (default)

CIF DMA Configuration Register 2 (CDCR2) 0xB020001C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Input Image /Y(G) Channel Base Address<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Input Image / Y(G) Channel Base Address<15:0>															

Field	Name	RW	Reset	Description
31-0	CDCR2	R/W	0x0	Input Image Base Address. / Y(G) channel base address

CIF DMA Configuration Register 3 (CDCR3) 0xB0200020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
U(R) Channel Base Address<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
U(R) Channel Base Address<15:0>															

Field	Name	RW	Reset	Description
31-0	CDCR3	R/W	0x0	U(R) Channel Base Address.

CIF DMA Configuration Register 4 (CDCR4) 0xB0200024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
V(B) Channel Base Address<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
V(B) Channel Base Address<15:0>															

Field	Name	RW	Reset	Description
31-0	CDCR4	R/W	0x0	V(B) Channel Base Address

CIF DMA Configuration Register 5 (CDCR5) 0xB0200028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OFS1<15:0>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OFS0<15:0>															

Field	Name	RW	Reset	Description
31-16	OFS1	R/W	0x0	U,V channel base address offset. This offset means the line start point to the next line start point.
15-0	OFS0	R/W	0x0	Y channel base address offset. This Offset means the line start point to the next line start point.

CIF DMA Configuration Register 6 (CDCR6) 0xB020002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Input Image /Y(G) Channel End Address<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Input Image / Y(G) Channel End Address<15:0>															

Field	Name	RW	Reset	Description
31-0	CDCR5	R/W	0x0	Input Image End Address. / Y(G) channel end address This mode is operated, when rolling address Y is enabled.

CIF DMA Configuration Register 7 (CDCR7) 0xB0200030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
U(R) Channel End Address<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
U(R) Channel End Address<15:0>															

Field	Name	RW	Reset	Description
31-0	CDCR6	R/W	0x0	U(R) Channel End Address This mode is operated, when rolling address U is enabled.

CIF DMA Configuration Register 8 (CDCR8) 0xB0200034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
V(B) Channel Base Address<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
V(B) Channel Base Address<15:0>															

Field	Name	RW	Reset	Description
31-0	CDCR7	R/W	0x0	V(B) Channel End Address This mode is operated, when rolling address U is enabled.

FIFO States (FIFOSTATE) 0xB0200038

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										CLR	0	REO	REV	REU	REY
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0		WEO	WEV	WEU	WEY	0	EO	EV	EU	EY	0	FO	FV	FU	FY

Field	Name	RW	Reset	Description
21	CLR	R/W	0x0	0 : Normal 1 : When writing 1 to this bit, all status bits are cleared.
19	REO	R	0x0	0 : Normal operation. 1 : It represents that FIFO underrun is occurred.
18	REV	R	0x0	0 : Normal 1 : It represents that FIFO underrun is occurred.
17	REU	R	0x0	0 : Normal 1 : It represents that FIFO underrun is occurred.
16	REY	R	0x0	0 : Normal 1 : It represents that FIFO underrun is occurred.
13	WEO	R	0x0	0 : Normal 1 : It represents that FIFO overrun is occurred.
12	WEV	R	0x0	0 : Normal 1 : It represents that FIFO overrun is occurred.
11	WEU	R	0x0	0 : Normal 1 : It represents that FIFO overrun is occurred.
10	WEY	R	0x0	0 : Normal 1 : It represents that FIFO overrun is occurred.
8	EO	R	0x0	0 : It represents that FIFO is not empty. 1 : It represents that FIFO is empty.
7	EV	R	0x0	0 : It represents that FIFO is not empty. 1 : It represents that FIFO is empty.
6	EU	R	0x0	0 : It represents that FIFO is not empty. 1 : It represents that FIFO is empty.
5	EY	R	0x0	0 : It represents that FIFO is not empty. 1 : It represents that FIFO is empty.
3	FO	R	0x0	0 : It represents that FIFO is not full. 1 : It represents that FIFO is full.
2	FV	R	0x0	0 : It represents that FIFO is not full. 1 : It represents that FIFO is full.
1	FU	R	0x0	0 : It represents that FIFO is not full. 1 : It represents that FIFO is full.
0	FY	R	0x0	0 : It represents that FIFO is not full. 1 : It represents that FIFO is full.

CIF Interrupt Register (CIRQ) 0xB020003C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IEN	URV	ITY	ICR	0	MVN	MVP	MVIT	MSE	MSF	MENS	MRLV	MRLU	MRLY	MSCF	MSOF
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SFEN	0	0	VSS	FL	VN	VP	VIT	SE	SF	ENS	ROLV	ROLU	ROLY	0	SOF

Field	Name	RW	Reset	Description
31	IEN	R/W	0x0	0 : Interrupt disable 1 : Interrupt enable
30	URV	R/W	0x0	0 : All CIF registers are applied to CIF operation regardless of VSYNC. 1 : All CIF registers are applied to CIF operation at rising edge of CVS signal.
29	ITY	R/W	0x0	1 : “1” SHOULD BE WRITTEN TO THIS BIT.
28	ICR	R/W	0x0	0 : Normal 1 : When writing 1 to this bit, the CIF interrupt is cleared.
26	MVN	R/W	0x0	0 : Enable When VN bit is set to 1, the CIF interrupt request is issued. 1 : Disable
25	MVP	R/W	0x0	0 : Enable When VP bit is set to 1, the CIF interrupt request is issued. 1 : Disable
24	MVIT	R/W	0x0	0 : Enable When VIT bit is set to 1, the CIF interrupt request is issued. 1 : Disable
23	MSE	R/W	0x0	0 : Enable When SE bit is set to 1, the CIF interrupt request is issued. 1 : Disable
22	MSF	R/W	0x0	0 : Enable When SF bit is set to 1, the CIF interrupt request is issued. 1 : Disable
21	MENS	R/W	0x0	0 : Enable When ENS bit is set to 1, the CIF interrupt request is issued. 1 : Disable
20	MRLV	R/W	0x0	0 : Enable When RLV bit is set to 1, the CIF interrupt request is issued. 1 : Disable
19	MRLU	R/W	0x0	0 : Enable When RLU bit is set to 1, the CIF interrupt request is issued. 1 : Disable
18	MRLY	R/W	0x0	0 : Enable When RLY bit is set to 1, the CIF interrupt request is issued. 1 : Disable
17	MSCF	R/W	0x0	0 : Enable When SCF bit is set to 1, the CIF interrupt request is issued.

				1 : Disable
16	MSOF	R/W	0x0	0 : Enable When SOF bit is set to 1, the interrupt request is issued. 1 : Disable
15	SFEN	R/W	0x0	0 : Generate SOF when 2-field images are stored memory. 1 : Without field number, generate SOF when 1-field image is stored memory.
12	VSS	R/W	0x0	0 : Non- vertical sync blank area. 1 : Vertical sync blank area.
11	FL	R/W	0x0	0 : Even field 1 : Odd field
10	VN	R/W	0x0	0 : - 1 : It represents that CVS falling-edge is detected.
9	VP	R/W	0x0	0 : - 1 : It represents that CVS rising-edge is detected.
8	VIT	R/W	0x0	0 : - 1 : Refer to CCM_2 register on page 4-29. Current line counter is bigger than CCM_2 setting value, this interrupt is generated. When writing 1 to this bit, it is cleared.
7	SE	R/W	0x0	0 : - 1 : It represents that CIF scaler has an error during scaling operation. When writing 1 to this bit, it is cleared.
6	SF	R/W	0x0	0 : - 1 : It represents that CIF scaler operation is completed. When writing 1 to this bit, it is cleared.
5	ENS	R/W	0x0	0 : - 1 : Refer to CESA register on page 4-29 Current line counter is bigger than CCM_2 setting value, this interrupt is generated When writing 1 to this bit, it is cleared.
4	ROLV	R/W	0x0	0 : - 1 : It represents that the current DMA V address reaches the DMA V rolling end address. When writing 1 to this bit, it is cleared.
3	ROLU	R/W	0x0	0 : - 1 : It represents that the current DMA U address reaches the DMA U rolling end address. When writing 1 to this bit, it is cleared.
2	ROLY	R/W	0x0	0 : - 1 : It represents that the current DMA Y address reaches the DMA Y rolling end address. When writing 1 to this bit, it is cleared.
0	SOF	R/W	0x0	0 : - 1 : It represents that the CIF has received one frame data from the camera module and stored them to specified memory completely. It does not consider CAP bit.

Overlay Control 1 (OCTRL1) 0xB0200040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			OCNT<4:0>					0							OM
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			OE	0	XR1	XR0	0	AP1<1:0>		AP0<1:0>		0	AEN	0	CEN

Field	Name	RW	Reset	Description
28-24	OCNT	R/W	0x37	55 (the value is fixed)
16	OM	R/W	0x0	Full image overlay mode, overlay image size is equal to the input image size. 0 : Full image overlay 1 : Block image overlay
12	OE	R/W	0x0	0 : Disable 1 : Enable
10	XR1	R/W	0x0	When AP1 is 3 and CEN & AEN is 1, we select the 100% alpha value or XOR operation. 0 : XOR operation 1 : 100 %
9	XR0	R/W	0x0	When AP0 is 3 and CEN & AEN is 1, we select the 100% alpha value or XOR operation 0 : XOR operation 1 : 100 %
7-6	AP1	R/W	0x0	0 : 25 %in overlay image 1 : 50 % in overlay image 2 : 75 % in overlay image 3 : 100 % in overlay image or XOR operation (for XR value) in overlay and input image
5-4	AP0	R/W	0x0	When RGB565 and AEN are set, alpha value depends on AP0 value. 0 : 25 % in overlay image 1 : 50 % in overlay image 2 : 75 % in overlay image 3 : 100 % in overlay image or XOR operation (for XR value) in overlay and input image
2	AEN	R/W	0x0	0 : Disable 1 : Enable
0	CEN	R/W	0x0	0 : Disable 1 : Enable

Overlay Control 2 (OCTRL2) 0xB0200044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved												CONV	RGB		MD

Field	Name	RW	Reset	Description
3	CONV	R/W	0x0	0 : Disable 1 : Enable
2-1	RGB	R/W	0x0	0 : 565RGB 1 : 555RGB 2 : 444RGB 3 : 332RGB
0	MD	R/W	0x0	0 : YUV color 1 : RGB color

Overlay Control 3 – key value (OCTRL3) 0xB0200048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								KEYR<7:0>							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEYG<7:0>								KEYB<7:0>							

Field	Name	RW	Reset	Description
23-16	KEYR	R/W	0x0	n : Chrome-key value in R(U) channel Default value is 0x00
15-8	KEYG	R/W	0x0	n : Chrome-key value in G(Y) channel. Default value is 0x00
7-0	KEYB	R/W	0x0	n : Chrome-key value in B(V) channel. Default value is 0x00

Overlay Control 4 – mask key value (OCTRL4) 0xB020304C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								MKEYR<7:0>							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MKEYG<7:0>								MKEYB<7:0>							

Field	Name	RW	Reset	Description
23-16	MKEYR	R/W	0x0	n : Chrome-key value in R(U) channel Default value is 0x00
15-8	MKEYG	R/W	0x0	n : Chrome-key value in G(Y) channel. Default value is 0x00
7-0	MKEYB	R/W	0x0	n : Chrome-key value in B(V) channel. Default value is 0x00

Overlay Image Size (OIS) 0xB0200050

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OHSIZE<15:0>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVSIZE<15:0>															

Field	Name	RW	Reset	Description
31-16	OHSIZE	R/W	0x0	Vertical size of overlay image Default value is 0x01E0 (decimal is 480)
15-0	OVSIZE	R/W	0x0	Horizontal size of overlay image Default value is 0x0280. (decimal is 640)

Overlay Image Windowing 1 (OIW1) 0xB0200054

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OHW1<15:0>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OHW2<15:0>															

Field	Name	RW	Reset	Description
31-16	OHW1	R/W	0x0	Refer to Figure 4.13
15-0	OHW2	R/W	0x0	Refer to Figure 4.13

Overlay Image Windowing 2 (OIW2) 0xB0200058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OVW1<15:0>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OVW2 <15:0>															

Field	Name	RW	Reset	Description
31-16	OVW1	R/W	0x0	Refer to Figure 4.13
15-0	OVW2	R/W	0x0	Refer to Figure 4.13

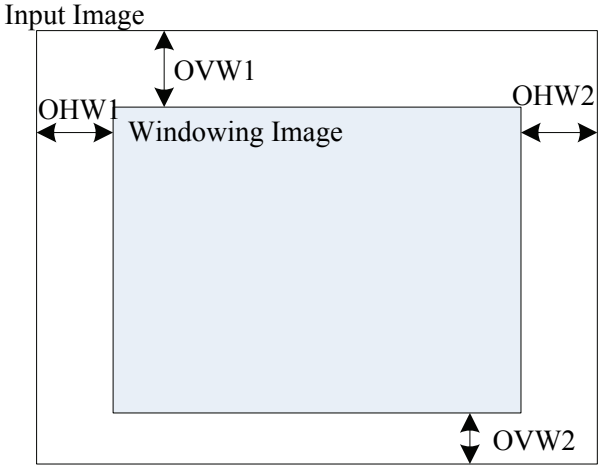


Figure 4.13 Overlay Image Windowing

CIF Overlay Base Address (COBA) 0xB020005C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Overlay Image Base Address<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Overlay Image Base Address<15:0>															

Field	Name	RW	Reset	Description
31-0	COBA	R/W	0x0	Overlay Image Base Address.

CIF Overlay Base Offset Address (COBO) 0xB0200060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Overlay Image Offset Address #1 <15:00>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Overlay Image Offset Address #0 <15:0>															

Field	Name	RW	Reset	Description
31-16	OIOA1	R/W	0x0	Overlay Image Offset Address in Chrominance.
15-0	OIOA0	R/W	0x0	Overlay image Offset Address in Luminance

CIF Down Scaler (CDS) 0xB0200064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										SFH<1:0>		SFV<1:0>		0	SEN

Field	Name	RW	Reset	Description
5-4	SFH	R/W	0x0	0 : 1/1 down scale 1 : 1/2 down scale 2 : 1/4 down scale 3 : 1/8 down scale
3-2	SFV	R/W	0x0	0 : 1/1 down scale 1 : 1/2 down scale 2 : 1/4 down scale 3 : 1/8 down scale
0	SEN	R/W	0x0	0 : Disable 1 : Enable

CIF Capture mode_1 (CCM_1) 0xB0200068

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ENCNUM				ROLNUMV				ROLNUMU				ROLNUMY			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					CB	EIT	UES	SKIPNUM				RLV	RLU	RLY	CAP

Field	Name	RW	Reset	Description
31-28	ENCNUM	R/W	0x0	0~15 : Refer to EIT bit.
27-24	ROLNUMV	R/W	0x0	0~15 : The number of times which V address is rolled while capture mode and rolling mode are enabled and one frame data is stored to the memory.
23-20	ROLNUMU	R/W	0x0	0~15 : The number of times which U address is rolled while capture mode and rolling mode are enabled and one frame data is stored to the memory.
19-16	ROLNUMY	R/W	0x0	0~15 : The number of times which Y address is rolled while capture mode and rolling mode are enabled and one frame data is stored to the memory.
10	CB	R/W	0x0	0 : Nomal 1 : It represents that frame data is storing to the memory during the capture mode.
9	EIT	R/W	0x0	0 : The ENS bit of CIRQ register is set to 1 when the current DMA Y address reaches the address to be specified in the CESA register. But, it can be occurred the only one time during storing one frame data. 1 : The ENS bit of CIRQ register is set to 1 whenever the current DMA Y address reaches the address to be specified in the CESA register. ENCNUM represents the number of times which ENS bit is set to 1.
8	UES	R/W	0x0	0 : The use of CESA register is disabled. 1 : The use of CESA register is enabled. Refer to CESA register on page 4-29
7-4	SKIPNUM	R/W	0x0	0~15 : It specifies the number of skip frames during capture mode.
3	RLV	R/W	0x0	0 : Disable (frame mode) 1 : Enable
2	RLU	R/W	0x0	0 : Disable (frame mode) 1 : Enable
1	RLY	R/W	0x0	0 : Disable (frame mode) 1 : Enable
0	CAP	R/W	0x0	0 : Preview mode 1 : Capture mode

CIF CAPTURE MODE_2 (CCM_2) 0xB020006C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								VCNT							VEN

Field	Name	RW	Reset	Description
7-4	VCNT	R/W	0x0	VIT bit is set to 1 whenever the number of lines which the CIF receives data from the camera module and stores them to specified memory is equal to the multiple of 16*VCNT lines. If VCNT is set to 0, VIT bit is never set to 1. VCNT can be 0 ~ 7. For example, when VCNT = 3, the VIT bit is set to 1 whenever a number of line to be stored is equal to a multiple of 48 line, which is 48, 96, 144, etc.
0	VEN	R/W	0x0	0 : Disable 1 : Enable(VCNT is used for setting VIT bit)

CIF Encoding Start Address (CESA) 0xB0200070

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Encoding Start Address [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Encoding Start Address [15:0]															

Field	Name	RW	Reset	Description
31-0	CESA	R/W	0x0	Default value is 0x20100000. This is compared with the current DMA Y address. When the current DMA Y address reaches the address to be specified in the CESA register, the ENS bit of CIRQ is set to 1. When the EIT bit of CCM_1 register is equal to 0, it can be occurred the only one time during storing one frame data. This operation is enabled when UES bit of CCM_1 register is set to 1. Refer to CCM_1 register on page 4-28.

CIF R2Y Configuration (CR2Y) 0xB0200074

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
											FMT				EN

Field	Name	RW	Reset	Description
4-1	FMT	R/W	0x0	FMT [4:1] / Input format 0 0 0 0 : 16bit 565RGB (RGB sequence) 0 0 0 1 : 16bit 565RGB (BGR sequence) 0 1 0 0 : 16bit 555RGB (RGB-garbage) 0 1 0 1 : 16bit 555RGB (BGR-garbage) 0 1 1 0 : 16bit 555RGB (garbage-RGB) 0 1 1 1 : 16bit 555RGB (garbage-BGR) 1 0 0 0 : 8bit 565RGB (RGB sequence) 1 0 0 1 : 8bit 565RGB (BGR sequence) 1 1 0 0 : 8bit 555RGB (RGB-garbage) 1 1 0 1 : 8bit 555RGB (BGR-garbage) 1 1 1 0 : 8bit 555RGB (garbage-RGB) 1 1 1 1 : 8bit 555RGB (garbage-BGR)
0	EN	R/W	0x0	0 : Disable 1 : Enable

CIF Current Y Address (CCYA) 0xB0200078

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Current Y address [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Current U address [15:0]															

Field	Name	RW	Reset	Description
31-0	CCYA	R	-	Current Y Address.

CIF Current U Address (CCUA) 0xB020007C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Current U address [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Current U address [15:0]															

Field	Name	RW	Reset	Description
31-0	CCUA	R	-	Current U Address

CIF Current V Address (CCVA) 0xB0200080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Current V address [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Current V address [15:0]															

Field	Name	RW	Reset	Description
31-0	CCVA	R	-	Current V Address

CIF Current Line Count (CCLC) 0xB0200084

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Line count [15:0]															

Field	Name	RW	Reset	Description
15-0	LCNT	R	-	Current Line Count

4.4 Effecter Register Descriptions

Table 4.2 Effect Register Map (Base Address = 0xB0200100)

Name	Address	Type	Reset	Description
CEM	0x00	W/R	0x00000000	Effect mode register
CSUV	0x04	W/R	0x00000000	Sepia UV setting
CCS	0x08	W/R	0x00000000	Color selection register
CHFC	0x0C	W/R	0x00000000	H-filter coefficent0
CST	0x10	W/R	0x00000000	Sketch threshold register
CCT	0x14	W/R	0x00000000	Clamp threshold register
CBR	0x18	W/R	0x00000000	BIAS register
CEIS	0x1C	W/R	0x00000000	Image size register
-	0x40	W/R	0x00000000	Reserved
CISA1	0x44	W/R	0x00000000	Source address in Y channel
CISA2	0x48	W/R	0x00000000	Source address in U channel
CISA3	0x4C	W/R	0x00000000	Source address in V channel
CISS	0x50	W/R	0x00000000	Source image size
CISO	0x54	W/R	0x00000000	Source image offset
CIDS	0x58	W/R	0x00000000	Destination image size
CIS	0x5C	W/R	0x00000000	Target scale

CIF Effect Mode (CEM) **0xB0200100**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
UVS	VB	UB	YB	YCS	IVY	STC	YCL	CS	SKT	EMM	EMB	NEGA	GRAY	SEPI	NOR

Field	Name	RW	Reset	Description
15	UVS	R/W	0x0	0 : U-V-U-V sequence 1 : V-U-V-U sequence
14	VB	R/W	0x0	0 : Disable 1 : Enable
13	UB	R/W	0x0	0 : Disable 1 : Enable
12	YB	R/W	0x0	0 : Disable 1 : Enable
11	YCS	R/W	0x0	0 : U(or V)-Y-V(or U)-Y sequence UV sequence is determined by UVS bit. 1 : Y-U(or V)-Y-V(or U) sequence UV sequence is determined by UVS bit.
10	IVY	R/W	0x0	0 : Disable 1 : Enable
9	STC	R/W	0x0	0 : Disable 1 : Enable
8	YCL	R/W	0x0	0 : Disable 1 : Enable
7	CS	R/W	0x0	0 : Disable 1 : Enable(color filter)
6	SKT	R/W	0x0	0 : Disable 1 : Enable
5	EMM	R/W	0x0	0 : Positive emboss 1 : Negative emboss
4	EMB	R/W	0x0	0 : Disable 1 : Enable
3	NEGA	R/W	0x0	0 : Disable 1 : Enable
2	GRAY	R/W	0x0	0 : Disable 1 : Enable
1	SEPI	R/W	0x0	0 : Disable 1 : Enable
0	NOR	R/W	0x0	0 : Effect mode (effecter is enabled) 1 : Normal mode (effecter is disabled)

CIF Sepia UV (CSUV) 0xB0200104

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEPIA_U								SEPIA_V							

Field	Name	RW	Reset	Description
15-8	SEPIA_U	R/W	0x0	U channel threshold value for sepia
7-0	SEPIA_V	R/W	0x0	V channel threshold value for sepia

CIF Color Selection (CCS) 0xB0200108

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
U start								U end							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
V start								V end							

Field	Name	RW	Reset	Description
31-24	U_Start	R/W	0x0	Color filter range start point of U channel
23-16	U_End	R/W	0x0	Color filter range end point of U channel
15-8	V_Start	R/W	0x0	Color filter range start point of V channel
7-0	V_End	R/W	0x0	Color filter range end point of V channel

CIF H Filter Coeff. (CHFC) 0xB020010C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0								Coeff 0							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Coeff 1								Coeff 2							

Field	Name	RW	Reset	Description
23-16	Coeff0	R/W	0x0	Horizontal filter coefficient0 for emboss or sketch.
15-8	Coeff1	R/W	0x0	Horizontal filter coefficient1 for emboss or sketch.
7-0	Coeff2	R/W	0x0	Horizontal filter coefficient2 for emboss or sketch

CIF Sketch Threshold. (CST) 0xB0200110

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								Sketch Threshold							

Field	Name	RW	Reset	Description
7-0	Sketch	R/W	0x0	Sketch threshold

CIF Clamp Threshold. (CCT) 0xB0200114

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								Clamp Threshold							

Field	Name	RW	Reset	Description
7-0	Clamp	R/W	0x0	Clamp threshold

CIF Bias Register (CBR) 0xB0200118

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
								Y bias							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
U bias								V bias							

Field	Name	RW	Reset	Description
23-16	Y_BIAS	R/W	0x0	Y value offset
15-8	U_BIAS	R/W	0x0	U value offset
7-0	V_BIAS	R/W	0x0	V value offset

CIF Effect Image Size (CEIS) 0xB020011C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
					HSIZE										
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					VSIZE										

Field	Name	RW	Reset	Description
26-16	HSiZE	R/W	0x0	Horizontal size of input image
10-0	VSIZE	R/W	0x0	Vertical size of input image

4.5 Scaler Register Descriptions

Table 4.3 Scaler Register Map (Base Address = 0xB0200200)

Name	Address	Type	Reset	Description
CSC	0x00	W/R	0x00000000	Scaler configuration
CSSF	0x04	W/R	0x00000000	Scale factor
CSSO	0x08	W/R	0x00000000	Image offset
CSSS	0x0C	W/R	0x00000000	Source image size
CSDS	0x10	W/R	0x00000000	Destination image size

CIF Scaler CTRL (CSC) 0xB0200200

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															EN

Field	Name	RW	Reset	Description
0	EN	R/W	0x0	0 : Disable 1 : Enable

CIF Scaler SCALE Factor(CSSF) 0xB0200204

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0		HSCALE													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0		VSCALE													

Field	Name	RW	Reset	Description
29-16	HSCALE	R/W	0x0	Horizontal scale factor
13-0	VSCALE	R/W	0x0	Vertical scale factor

HSCALE = SRC_HSIZE * 256 / DST_HSIZE
VSCALE = SRC_VSIZE * 256 / DST_VSIZE

CIF Scaler Source Offset (CSSO) 0xB0200208

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				H_OFFSET											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0				V_OFFSET											

Field	Name	RW	Reset	Description
27-16	H_OFFSET	R/W	0x0	Horizontal offset
11-0	V_OFFSET	R/W	0x0	Vertical offset

CIF Scaler Source Size (CSSS) 0xB020020C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				H_SIZE											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0				V_SIZE											

Field	Name	RW	Reset	Description
27-16	H_SIZE	R/W	0x0	Horizontal size in source image
11-0	V_SIZE	R/W	0x0	Vertical size in source image

CIF Scaler DST_Size (CSDS) 0xB0200210

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0				H_SIZE											
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0				V_SIZE											

Field	Name	RW	Reset	Description
27-16	H_SIZE	R/W	0x0	Horizontal size in destination image
11-0	V_SIZE	R/W	0x0	Vertical size in destination image

4.6 User guide for CAMIF Setting

4.6.1 External Camera Module Interface (CAMIF)

Figure 4.14 explains the interconnection diagram between NVS2310 and the camera sensor. In a normal camera sensor, a pixel clock is sent out by using an oscillator or external clock input as a clock source. According to the camera sensor, an input clock itself can be used or a divided input clock can be used.

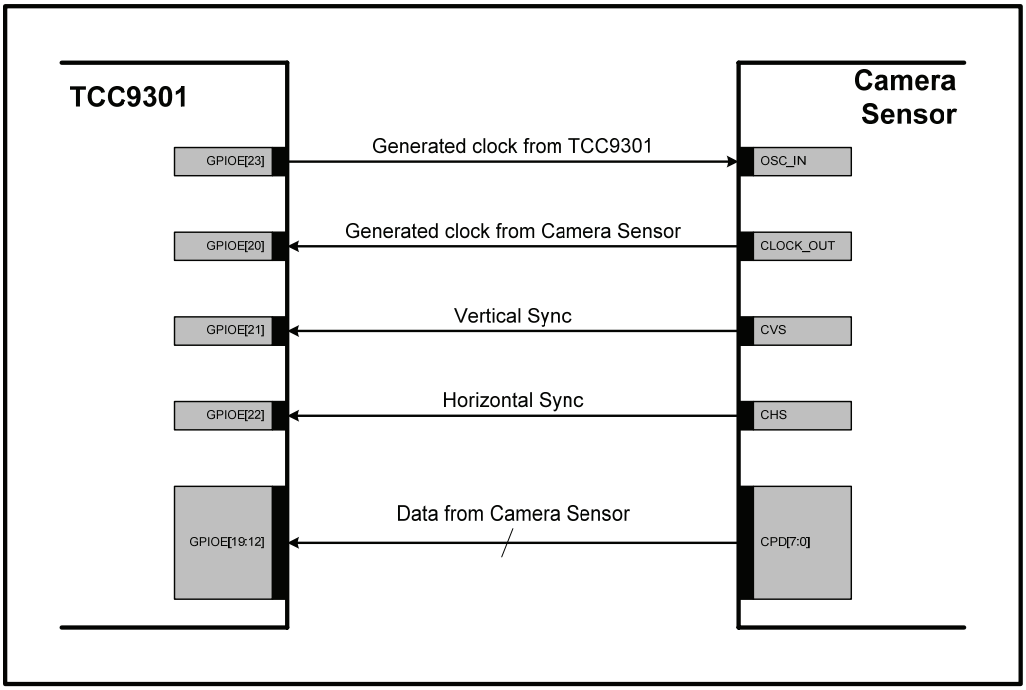


Figure 4.14 Interconnection between NVS2310 and Camera Sensor

Figure 4.15 shows the relation between each clock. The clock outputted from the Sensor is diagramed under the assumption that it is double of the input clock of the Sensor (master clock of the Sensor). In terms of the relation between each Clock, F_{SCALER} should be more than twice of FCCKI (base: inputted F_{CCKI}) as marked in the figure (precisely, twice of FCCKI is recommended). F_{IOBUS} should be bigger than at least F_{SCALER}. However, It can be changeable according to the Overlay of the Camera Block and other hardware operations. For this reason, F_{IOBUS} and F_{SCALER} should be additionally adjusted if the Overlay of the Camera Block and other hardwares are used.

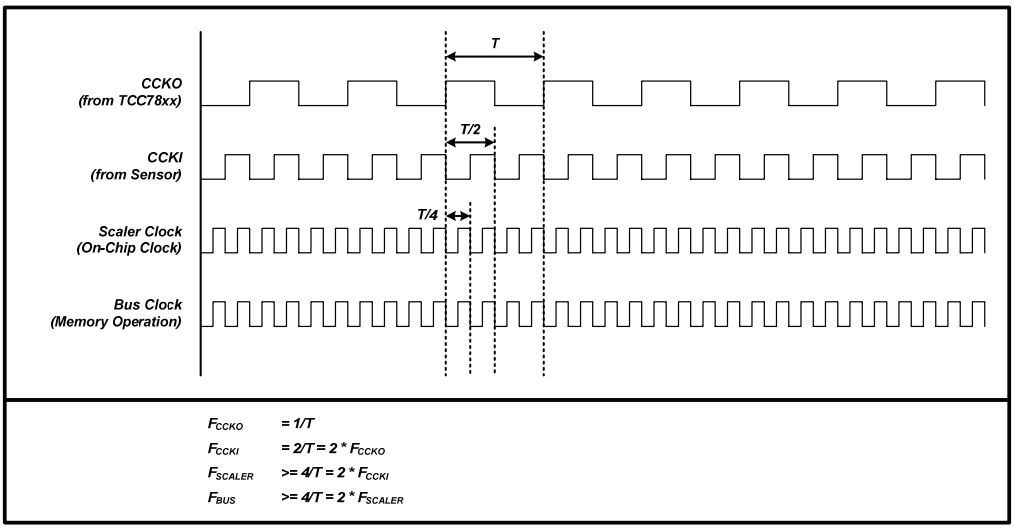


Figure 4.15 Frequency Relations for All the Clocks

4.6.2 Camera Hardware ON/OFF Sequence

Figure 4.16 shows the timing diagram for the output signal of the sensor. CVS means a vertical sync signal or frame valid signal outputted from the camera sensor. CHS means a horizontal sync or line valid signal.

At the moment when the Camera Sensor is Turned-on, a signal is continuously outputted at the timing as Figure 4.16.

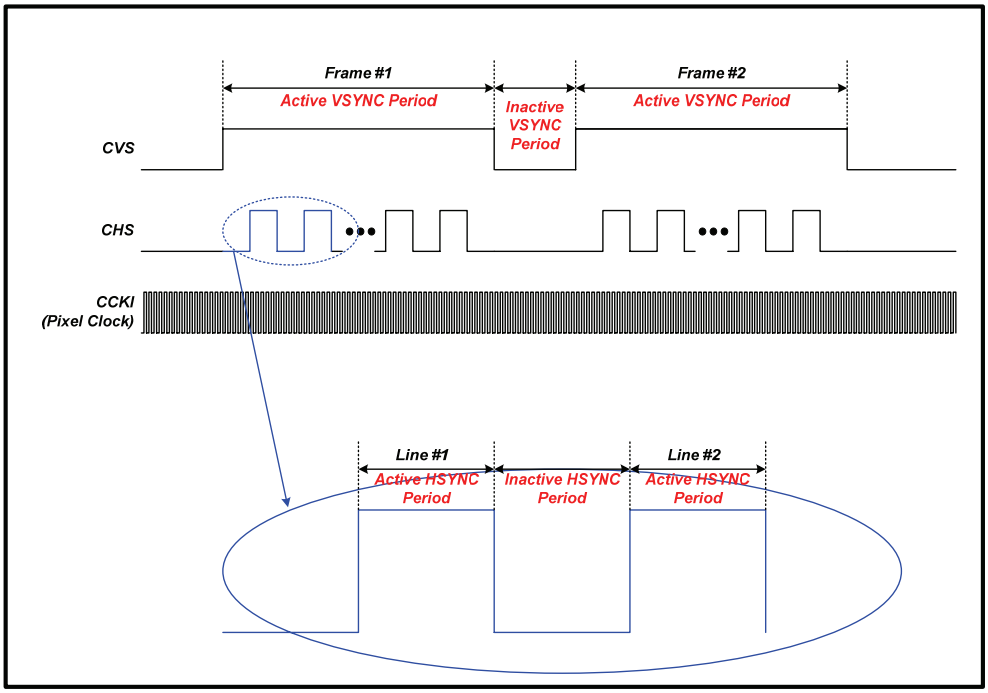


Figure 4.16 Frequency Relations for All the Clocks

Figure 4.17 explains Turn-On Sequence of Camera Interface Hardware. Before Turning on, the software of Camera Interface Hardware Block should be reset. When resetting is carried out, after the software reset register inside CKC block enters into the reset state, a camera clock and scaler clock should be set and software reset register inside CKC block should be released.

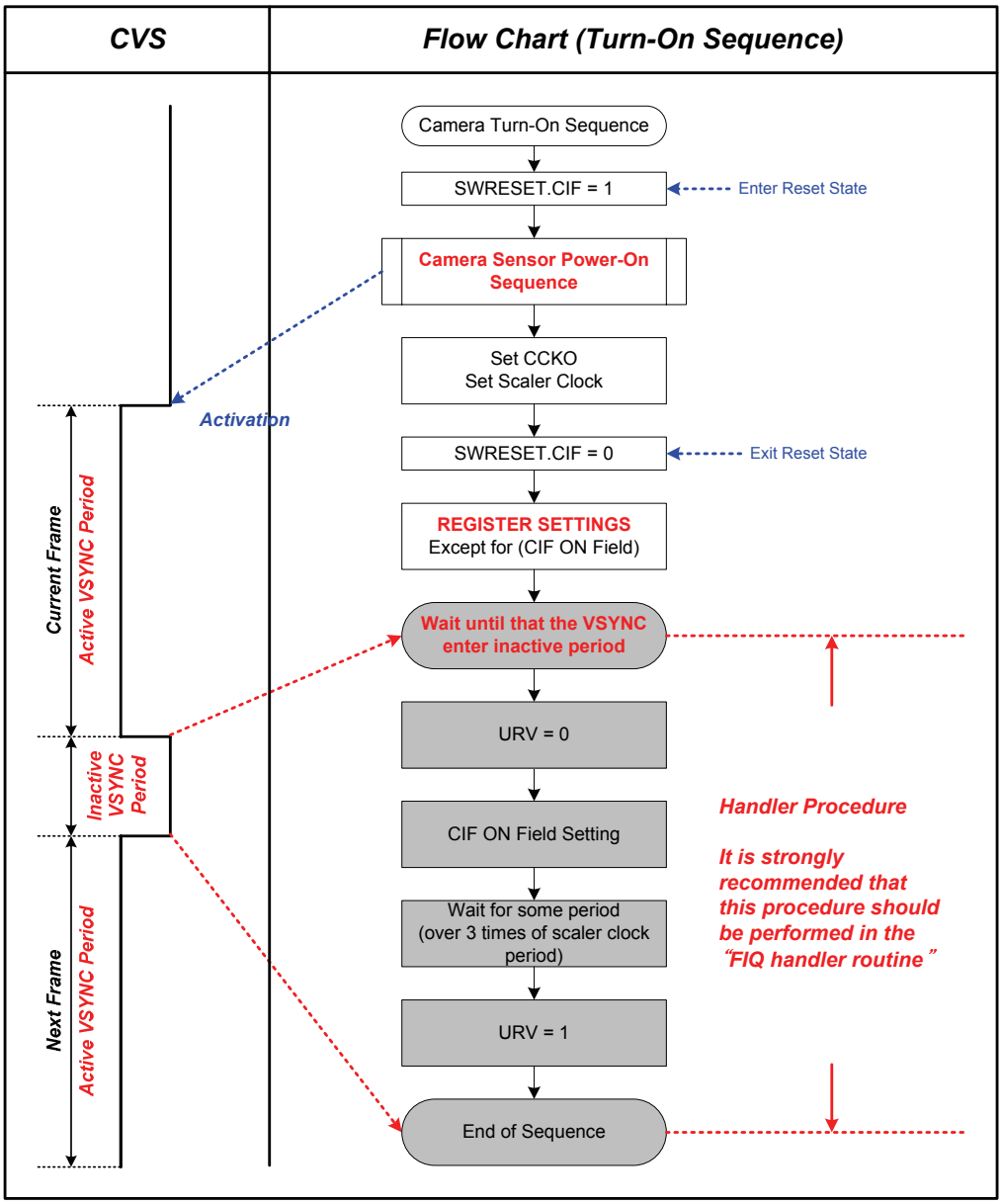


Figure 4.17 Camera Interface Hardware Turn-On Sequence

Figure 4.18 shows Turn-Off Sequence of the Camera Interface Hardware. Turning-off the camera block in NVS2310 should be done when VSYNC is in an Inactive period. If VSYNC is Turned-Off when it is Active, the remained dummy data in the memory buffer could be saved to the unintended memory.

In Turn-On Sequence and Turn-Off Sequence, a signal for an external VS should not be referred to VP or VN interrupt source inside of CAMIF hardware block. Make sure to use External Interrupt Source. – Refer to Port Multiplexor Chapter.

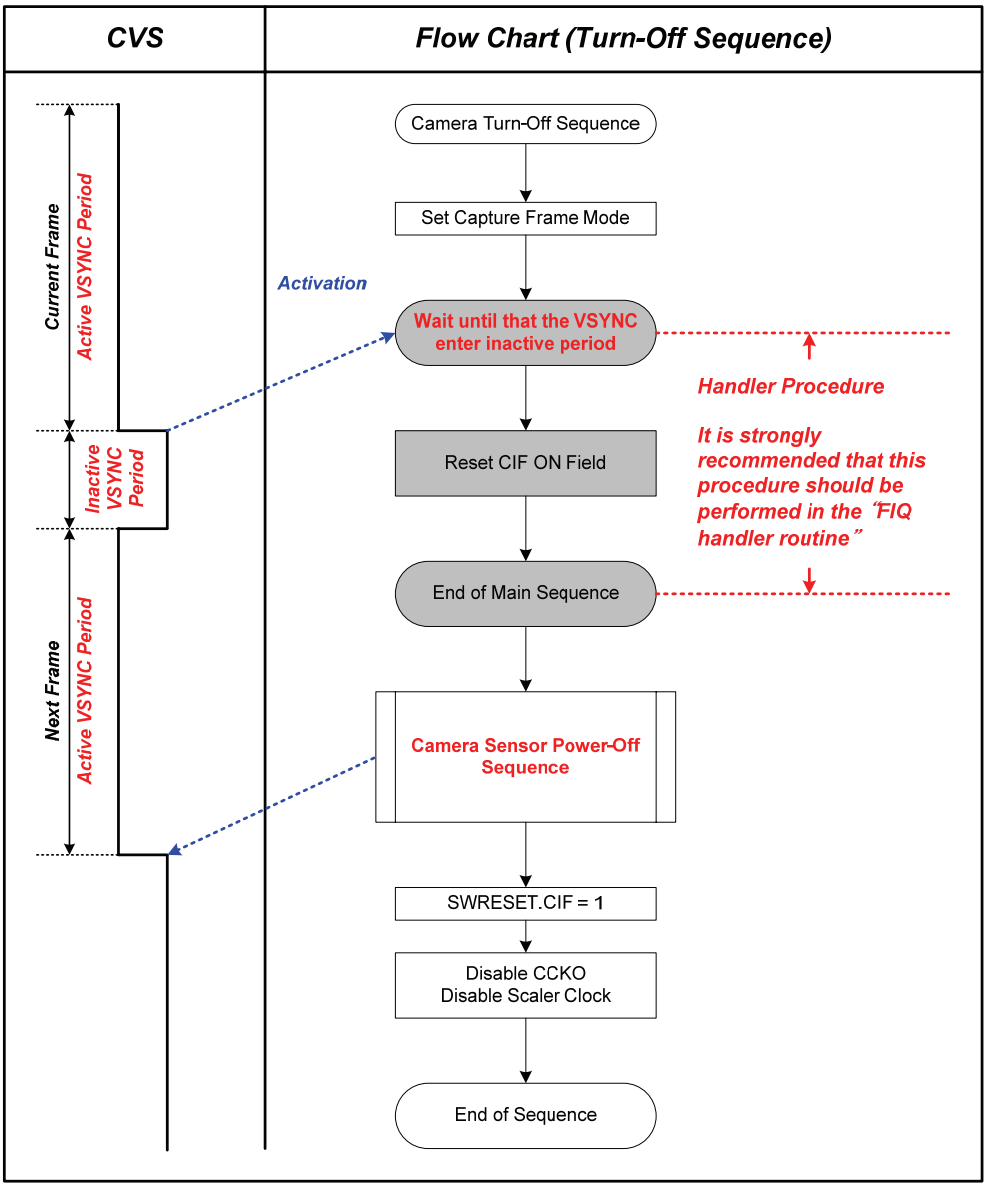


Figure 4.18 Camera Interface Hardware Turn-Off Sequence

Depending on the occasions, when Preview Mode is converted to Capture Mode in a Camera Sensor where Preview Mode and Capture Mode are separate.

4.6.3 Timing Tuning for Camera Sensor

A Camera Sensor varies by companies and pixels. Basically, it operates like Figure 4.14. However, in order to be connected to NVS2310, the below instructions should be followed for the normal operation of the CIF in NVS2310.

Figure 4.19 shows the inside scaler operation for the waveform from the Camera Sensor. Since the Scaler inside NVS2310 uses a 2 line vertical buffer for line filtering, HSYNC(CHS) signal inputted from outside occurs maximum 2 lines later. Therefore, timing should be set with the 2 line delayed operation considered.

In terms of Falling edge (a signal entering into an Inactive state) and Rising edge (a signal entering into an Active state) in CVS signal, it is designed that the preparation for the next Frame is done at the falling edge. The following is the reason for this. Overlay function is included in the Camera Interface Hardware. In order for Overlay DMA to operate normally for the first Pixel of the first Line, as soon as CVS enters into an Inactive state, the operation to process the next frame starts.
For this reason, T_{H2V} time should be obtained in order that Stored-One-Frame (SOF) Interrupt happens normally. In terms of T_{H2V} time, due to its Scaler operation structure, at least more than 2 Line time margin should be obtained for the normal operation of the CIF block in NVS2310.

In addition, T_{H2V} is relevant to an Effector block, the below chapter “Effector User Guide” explains this in details.

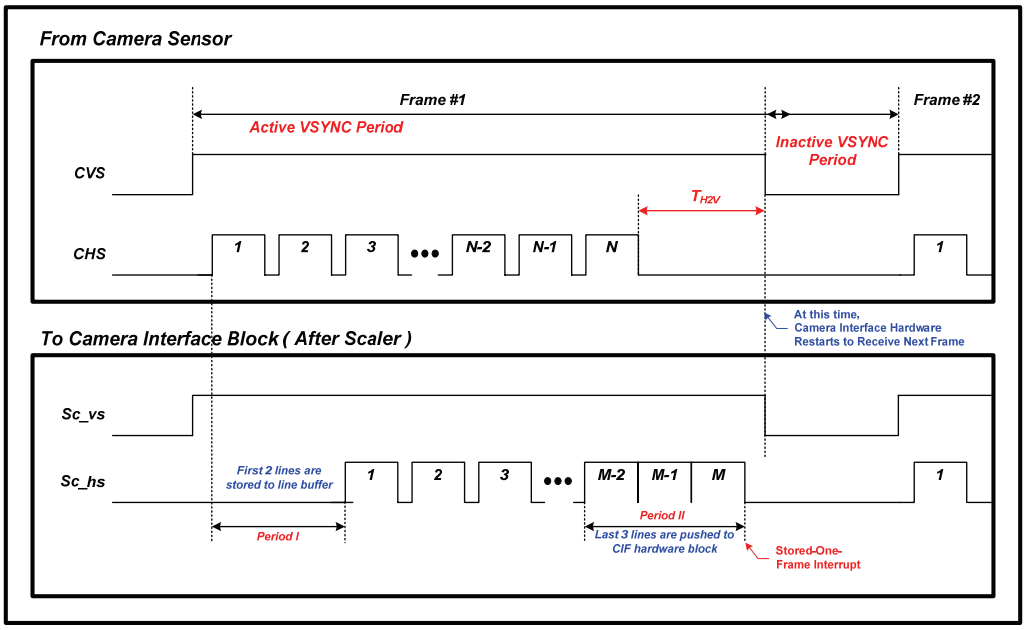


Figure 4.19 Operating Timing Diagram (Correct)

Figure 4.20 illustrates the timing diagram when an abnormal operation occurs. As can be seen in the figure, since T_{H2V} is too short, if Restart Event occurs at T_{fall} in the middle of processing (M-1) and (M) lines, this leads to that the Scaler and CIF hardware operation enters into an abnormal state., Therefore, Stored-One-Frame Interrupt does not happen.

Processing Camera Sensor Timing like Figure 4.20 is explained in the Zoom function.

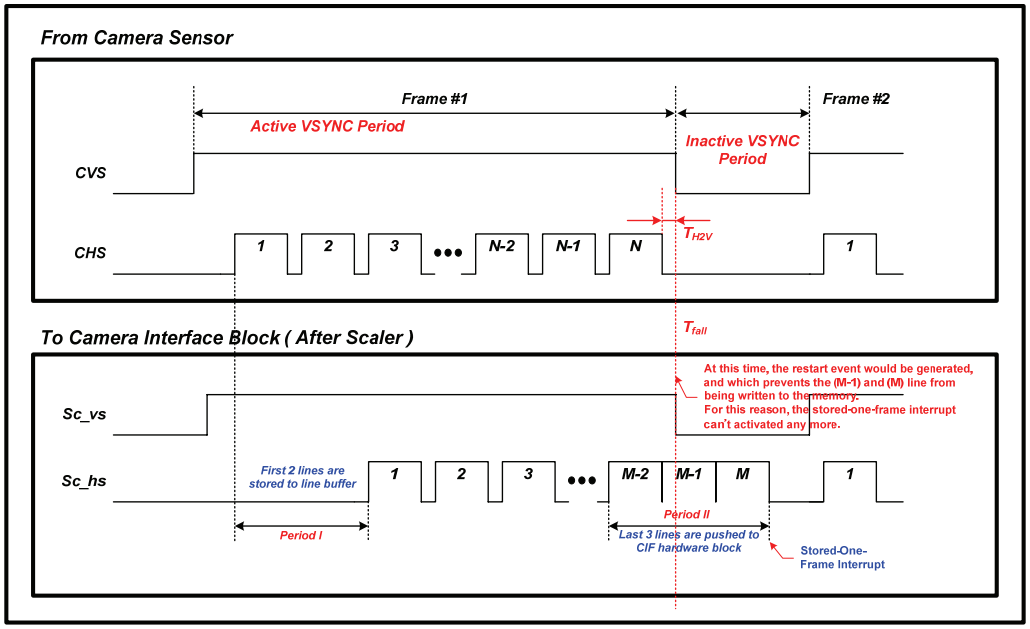


Figure 4.20 Operating Timing Diagram (Incorrect)

4.6.4 Zooming Control

Since there is a built-in Scaler in the Camera Interface Hardware of NVS2310, the inputted data from the Sensor can be Zoomed-In and Zoomed-Out.

In Figure 4.21, there is a basic image frame provided in the Camera. If the horizontal and vertical line refer to HSIZE and VSIZE., the red colored part represents Zoom-Out(Down Scale, it might not be actual Zoom-Out). The blue colored part means Zoom-In(Up-Scale). Generally, Zoom-In Region can be located anywhere within the Camera Frame since the location does not affect the operation of the scaler. However, according to the feature of a camera, Zoom-In and Zoom-out are done at the center.

If image data can be inputted from the Camera as Figure 4.19, this is absolutely fine to process the operation like Figure 4.21.

However, something is inputted as Figure 4.20, this could be fine with Zoom-In. However, it could be problematic with Zoom-Out. In the above case, if Zoom-In and Zoom-Out are done after the input data of the Camera Frame is partially windowed as Figure 4.21.

As explained in Figure 4.22, data which is inputted like in Figure 4.20 is fine in Zoom-In (Up-Scale). However, in Zoom-Out(Down-Scale), a problem occurs. Therefore, an imaginary Window should be set and a user should scale the window.
The Start Position of the Window can be the same with that of the Camera Frame. However, if symmetry is considered, 2 pixels on each side of the horizontal line and 2 lines from each ending point of the vertical line are disregarded. Due to 2 lines at the bottom, It is considered there are no data of the last 2 lines at the bottom in Figure 4.20. Therefore, the Scaler can normally operate.

Likewise, 1:1 Scaling is processed in the same way since the Window Frame is Scaled-Up to the Camera Input Frame size if there is no scaling related operation underway.

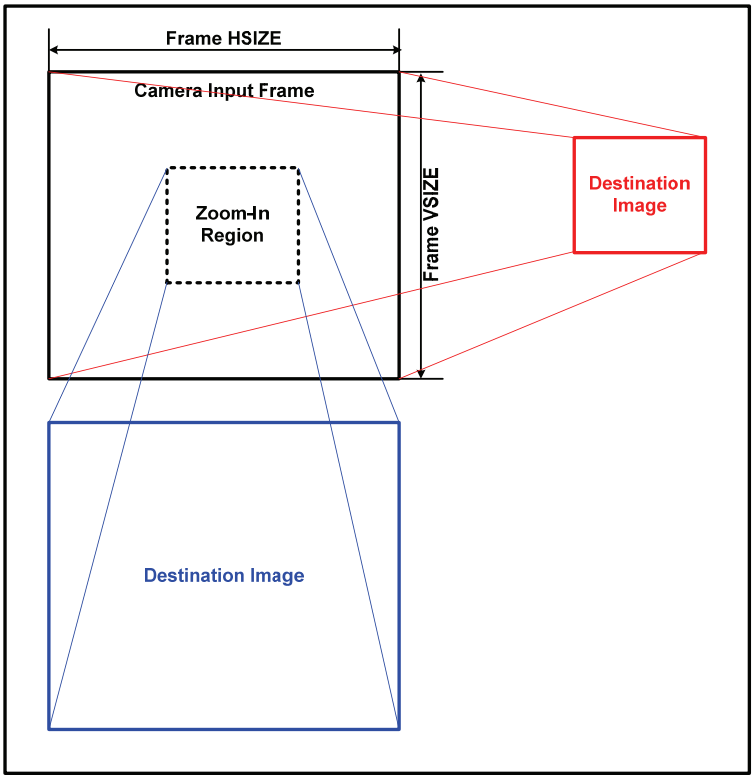


Figure 4.21 Camera Zoom-In(Up-Scale) and Zoom-Out(Down-Scale) - Correct

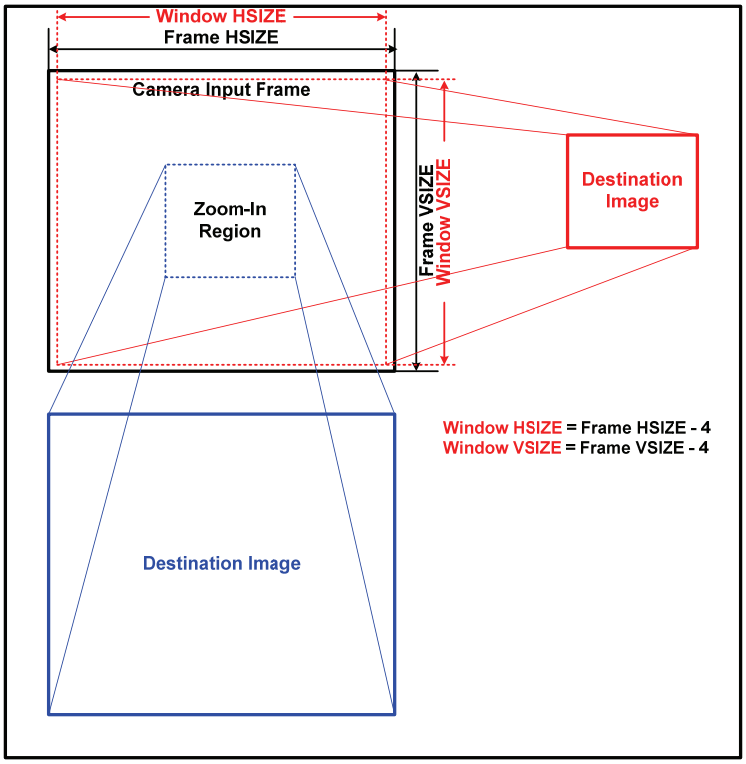


Figure 4.22 Camera Zoom-In(Up-Scale) and Zoom-Out(Down-Scale) - Incorrect

4.6.5 Using One Frame Capture Signal

Whenever one frame is received from the external Camera Sensor, a status signal is generated. The created signals are SOF, VP, VN and VSS field of CIRQ(0x3C) register of the control registers in Camera IF. Each signal creation timings are as below Figure 4.23.

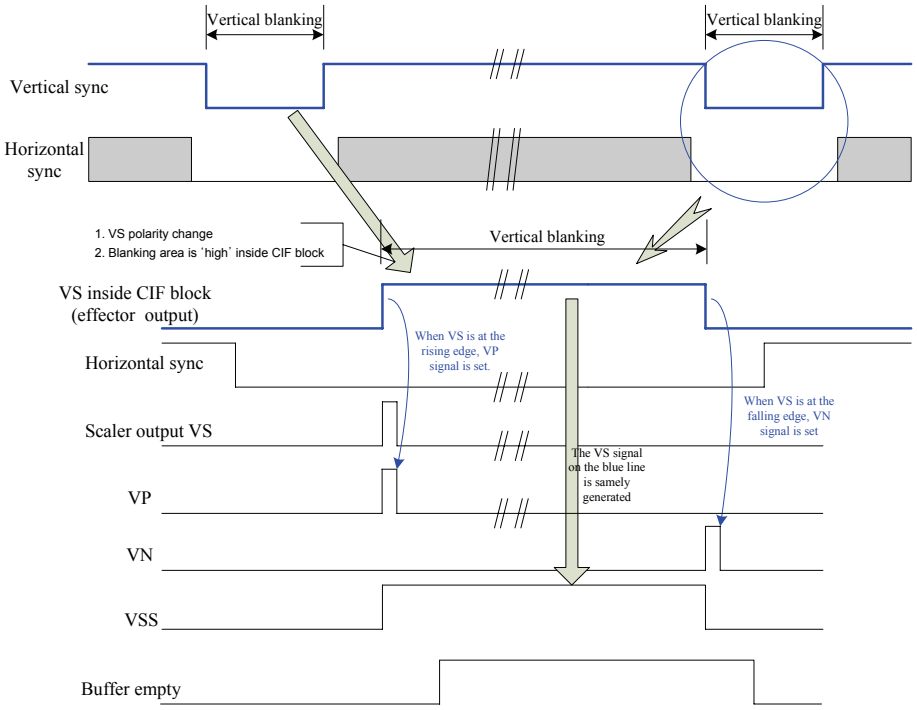


Figure 4.23 The Timing of Status Signals

The CIF operates when VS is 'low' and HS is 'high' in an active region. If VS as the above is 'high' in the active region, it

operates by changing polarity. VP signal generates when a vertical sync is at a rising edge. VN signal generates when a inside vertical sync is at an falling edge. VSS signal precisely represents the current VS state. (Since the current VS state is shown, the VS state is ‘low’ in the active region.)

How to check that one frame is saved in both Preview Mode and Capture Mode is to use SOF signal. SOF signal is generated at the similar location to VP location (refer to Figure 4.26) For Capture Mode usage, refer to “Capture” section of the below chapter.

4.6.6 Register Update

To apply the saved register as a setting value in Camera Interface Hardware to an actual operation is called register update. There are two in register update timing. One is update is done as soon as data is written to the register. The other is update is done when the next VS is rising after data is written to the register. (Refer to CIF_Effector timing in Figure 4.24)

According to URV field (bit 30) of CIRQ register (0x38), either the first or the latter is selected.

When URV field is ‘1’, if VS is rising, register data is updated. When URV field is ‘0’, update is done as soon as register data is written.

A special attention should be given When URV field is used with Camera IF On/Off.
Before Camera IF is On, URV field should not be advancely set. This is because it is concerned with sensor operation timing. (Refer to Figure 4.17, Figure 4.18)

a. When Camera IF is on

After the register values of Camera IF are all set, the next sensor should be on and ON field of CIF should be set. Then, set URV field to ‘1’.

b. When Camera IF is off

After ON field is cleared, Clear URV field to ‘0’.

Likewise, 1:1 Scaling is processed in the same way since the Window Frame is Scaled-Up to the Camera Input Frame size if there is no scaling related operation underway.

4.6.7 Capture

When a user captures the data inputted the Camera sensor, it works based on timing diagram as Figure 4.24. As Figure 4.24, if a capture signal is inputted, from the data of the next frame is saved. In the same way, if 1 frame is saved, the next frame keeps being skipped until the capture signal is ‘low’ and VP or SOF signal notifying that saving 1 frame is completed, is generated. (However, inputted VS can be continuously monitored in the register.)

There is time lapse until writing operation is transferred to the memory. It varies depending on where or not a scaler is used. When the scaler is used, 2 line + 120*PCLK exists in “first data to memory write time”. 2 lines are occurred by the delay of the line buffer of the scaler. 120*PCLK is the cycle of input-to-memory.

“Last data to memory write time” is around maximum 50*PCLK(base: PCLK). However, this is affected by the frequency of SCLK and HCLK. Therefore, the transfer can be completed even though Last data to memory write time is under 50*PCLK since.

(Note : refer to upper chapter “zooming control”)

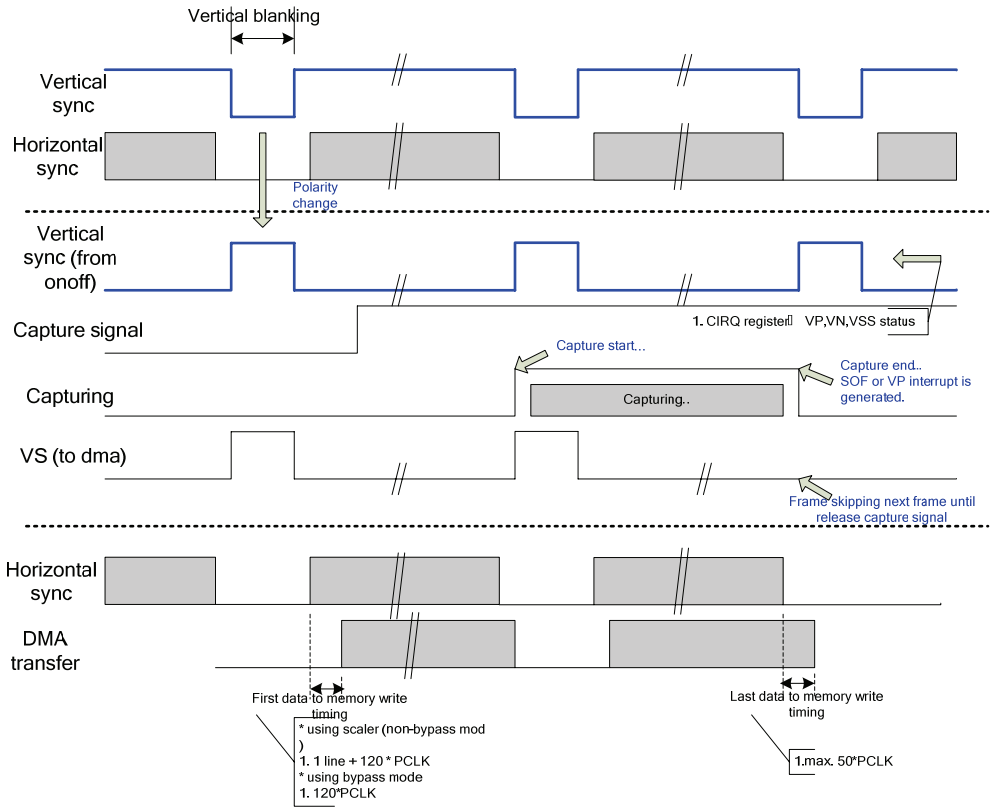


Figure 4.24 Capture Timing Diagram

Figure 4.25 represents Capture Sequence. As can be seen in the figure, check the first frame is properly inputted after turning on CIF while CVS is monitored. If SOF INT is set while SOF is used as an Interrupt source, set CAP field (bit 0 of 0x0060) when VP signal is 'high' in the handler routine. At this time, VP and SOF should be set almost at the same time. As soon as CAP field is set, 1 frame is captured.

If 1 frame capturing is completed, SOF INT is set likewise. (the second SOF waitingbox in the figure). If the SOF INT is set, the handler routine turns off CIF and processes the captured images (post processing box). JPEG Encoding, MPEG Encoding, Effect..ect are the examples about this.

(For more detailed information, refer to Figure 4.25 Flow chart)

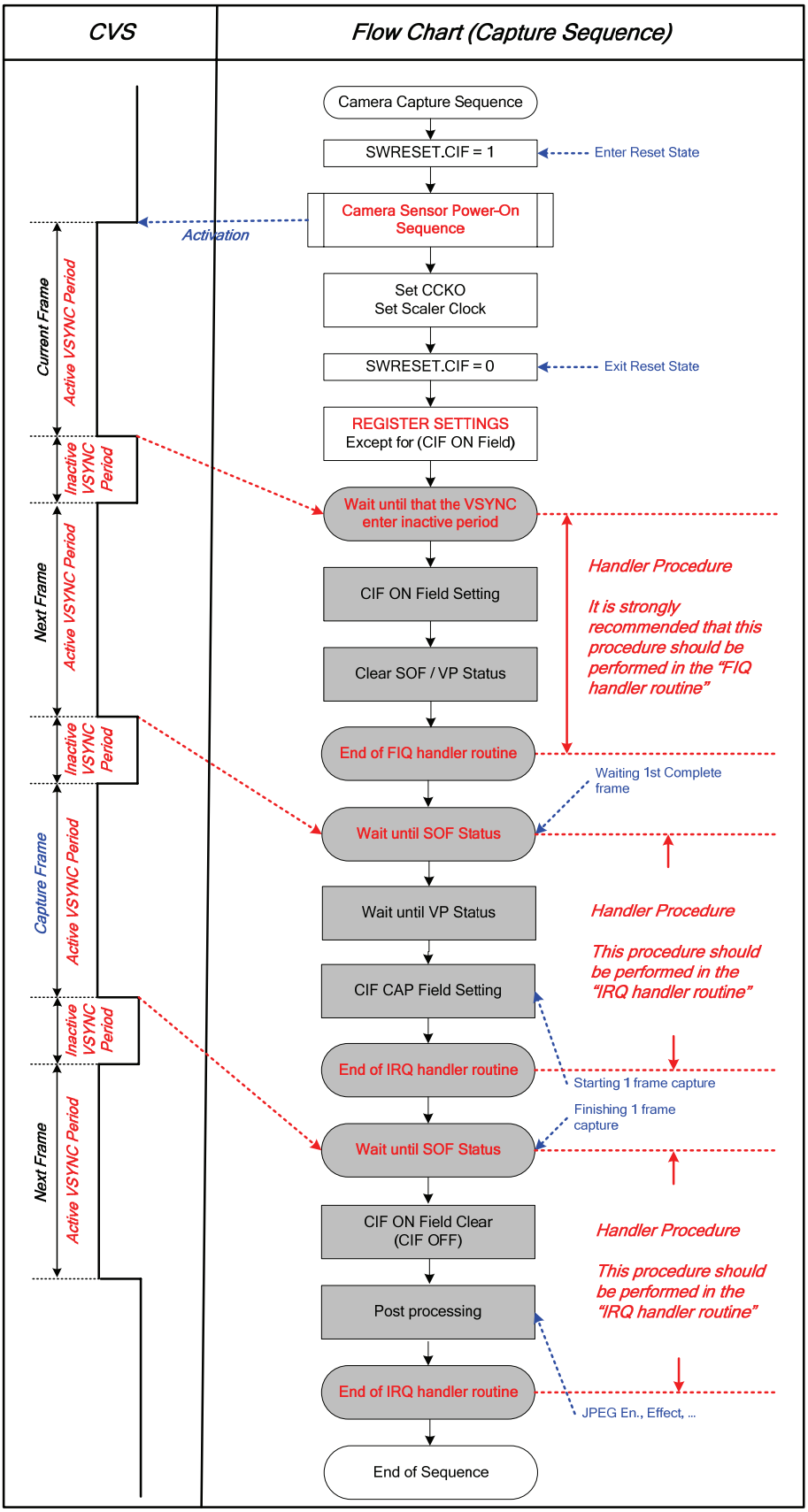


Figure 4.25 Camera Interface Hardware Capture Sequence

4.6.8 Timing and Status Diagram by CAMIF HS/VS Block

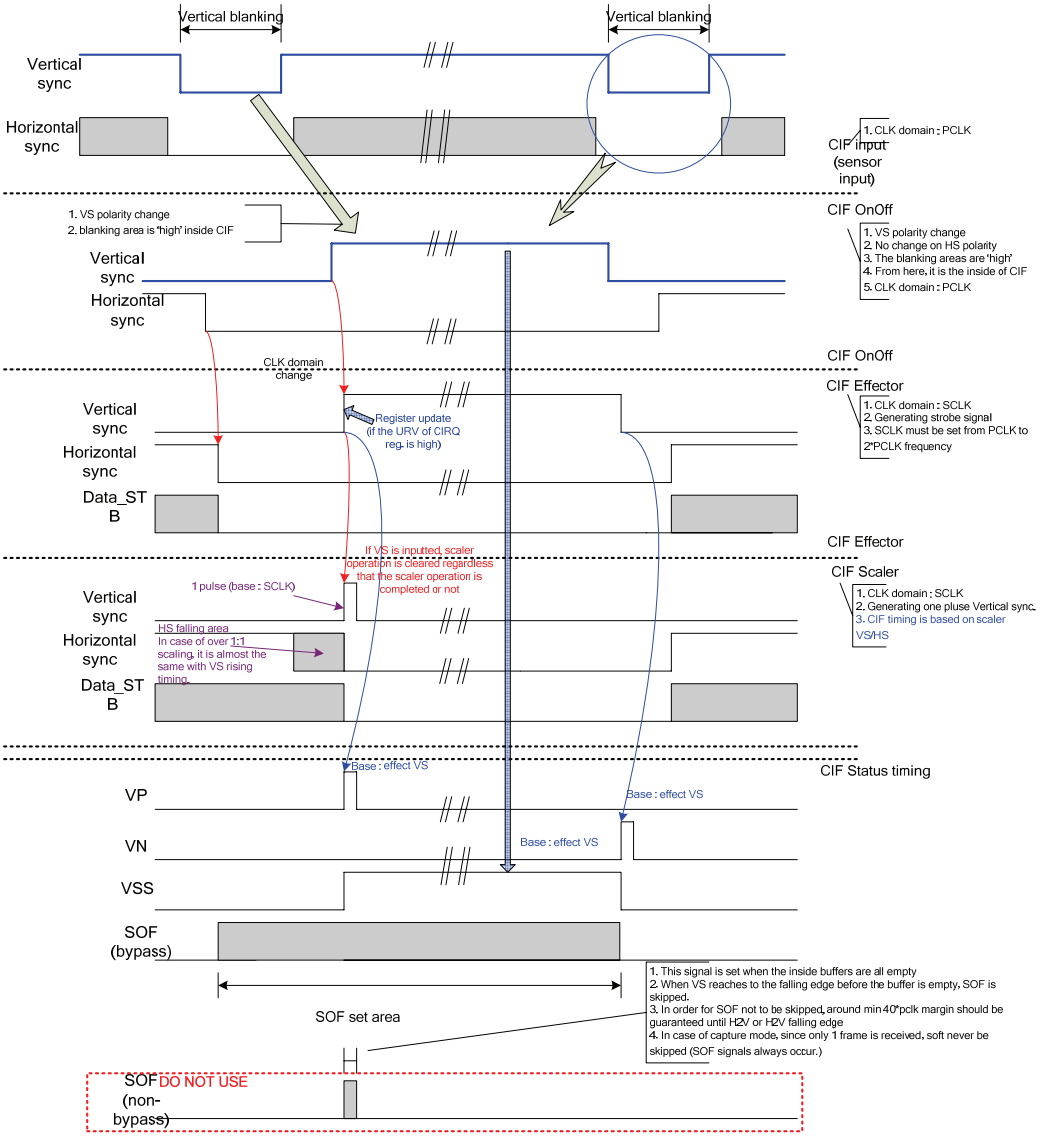


Figure 4.26 CAMIF HS/VS Timing Diagram

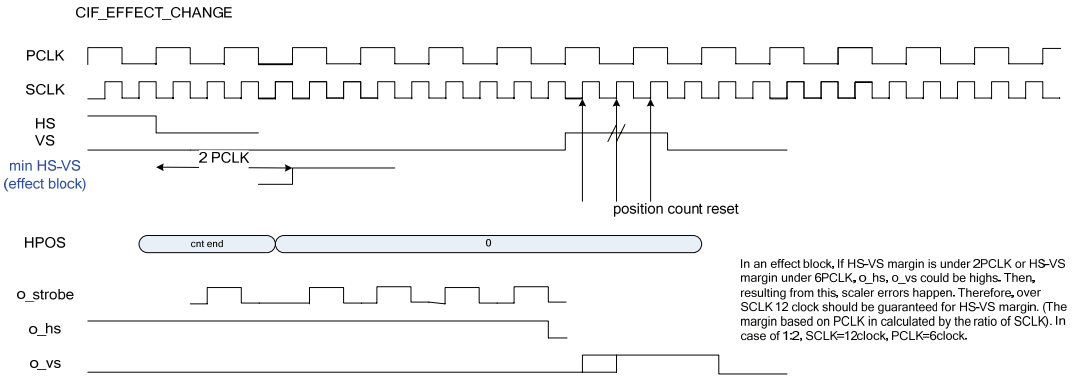
4.6.9 Effecter User Guide

Guide 1. Embossing/Sketch effector

When Embossing/Sketch effect is used, filter coefficient should be normalized. The best result could come out with coeff0=0xff(-1), coeff1=0x00 and coeff2=0x02. When 0x80 is set as sketch Th value, the best drawing effect occurred. After Sketch effect, an image has only 0 and 255 as Y component. This leads to increasing of a file size in JPEG encoding since compression is less efficient in Sketch effect compared to other effects.

Guide 2. H2V margin for the normal operation of Effector module

In order for Effector module to normally operate, H2V margin is required. As can be seen in Figure 1.27, 6*PCLK margin is needed. Effector blocks has delay aspects since they internally use a 3-tap filter. In addition, since the effector blocks initialize position registers when VS becomes active, H2V margin should be secured to prevent data at the end of the frames from being lost.



In an effect block, If HS-VS margin is under 2PCLK or HS-VS margin under 6PCLK, o_hs, o_vs could be highs. Then, resulting from this, scaler errors happen. Therefore, over SCLK 12 clock should be guaranteed for HS-VS margin. (The margin based on PCLK in calculated by the ratio of SCLK). In case of 1:2, SCLK=12clock, PCLK=6clock.

5 ISP (Image Signal Processor)

5.1 Overview

The NVS2310 has the ISP. Its features are as follows.

- ITU-R BT 601 compliant video interface supporting YCbCr and RGB Bayer data
- ITU-R BT 656 compliant video interface supporting YCbCr data
- Flash light control
- Mechanical shutter support
- 12 bit camera interface
- 12 bit resolution per color component internally
- YCbCr 4:2:2 processing
- Hardware JPEG encoding
- Maximum input resolution of 12 Mpixels (4096x3072 pixels)
- Main scaler and Self scaler are included.
- Color processing (contrast, saturation, brightness, hue)
- Digital image effects (Emboss, Sketch, Sepia, B/W (Grayscale), Color Selection, Negative image)
- Rotation unit in self-picture path (90°, 180°, 270°) for RGB output
- Bad pixel detection and correction
- Black level compensation
- Lens shade correction
- Auto focus measurement
- Auto exposure support by brightness measurement
- Enhanced color interpolation
- Noise filter
- Sharpening / Blurring filter
- Chromatic aberration correction
- Histogram calculation
- Color correction matrix
- Solarize effect through gamma correction
- De-noising pre filter
- Defect pixel cluster correction unit
- wide dynamic range unit
- image effects sharpening feature

5.2 Register Descriptions

Table 5.1 ISP Register Map (Base Address = 0xB0210000)

Name	Address	Type	Reset	Description
BASE	0x0000	RW	0x00000000	TCISP Main Control Registers
IMGEFF_BASE	0x0200	RW	0x00000000	Image Effects
SI_BASE	0x0300	RW	0x00000000	Superimpose
ISP_BASE	0x0400	RW	0x00000000	ISP main registers
FLASH_BASE	0x0660	RW	0x00000000	FLASH_LIGHT registers
SHUT_BASE	0x0680	RW	0x00000000	SHUTTER registers
CPROC_BASE	0x0800	RW	0x00000000	COLOR PROCESSING registers
MRSZ_BASE	0x0C00	RW	0x00000000	MAIN RESIZE registers
SRSZ_BASE	0x1000	RW	0x00000000	SELF RESIZE registers
MI_BASE	0x1400	RW	0x00000000	MEMORY Interface registers
JPE_BASE	0x1800	RW	0x00000000	JPEG ENCODER registers
-	0x1A00	-	-	Reserved

5.3 Post-processing

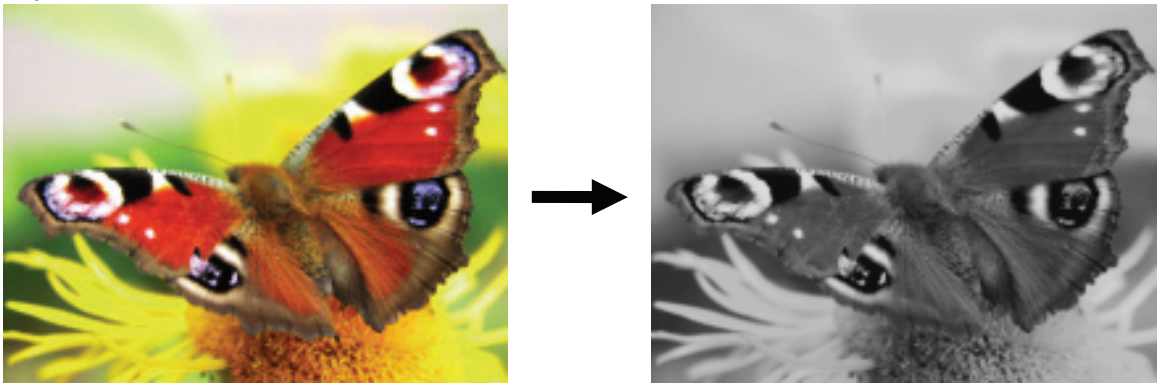
5.3.1 Color Processing Programming

Color processing is done in terms of luminance and chrominance adjustment.
Contrast processing is done by multiplying the luminance value by the contrast adjustment value defined in CPROC_CONTRAST which is in the range of 0.0 (0x00) and 1.992 (0xFF).
Brightness is adjusted by adding a value defined in CPROC_BRIGHTNESS to the luminance value of a pixel. This 8 bit value is a signed number in two's-complement.
Saturation manipulation in chrominance processing in terms of multiplying C_r and C_b with a fixed-point number between 0.0 (0x00) and 1.992 (0xFF). This value is to be programmed in CPROC_SATURATION.
Hue processing is a phase shift of the chrominance values between -90 (0x80) degree and +87.188 (0x7F) degree to be defined in CPROC_HUE.
The color processing module can be configured to produce pixel values according to the BT.601 standard or in full value range. Full value range should be used for JPEG encoding. See register COLOR_PROC_CTRL for details of the possible configurations. The input pixel range is also configurable, because pixels coming from the YCbCr path of the ISP may have both ranges depending on the camera sensor used.

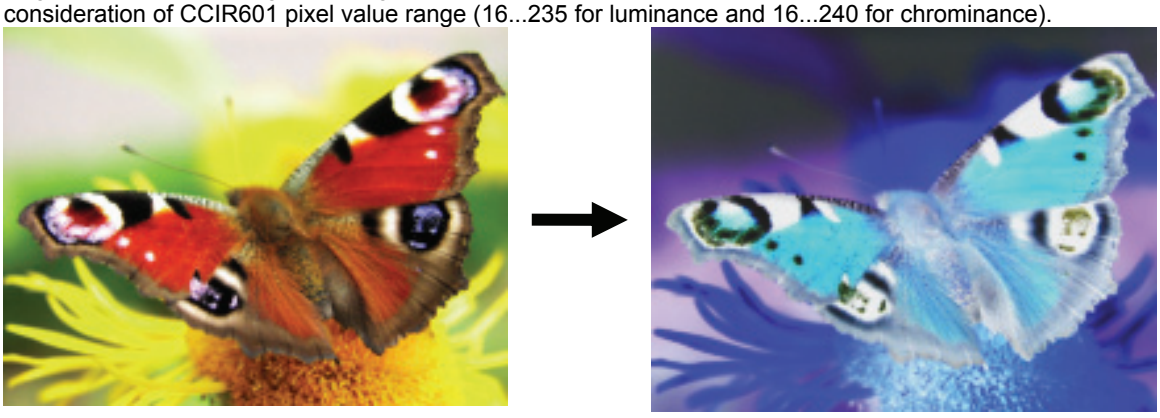
5.3.2 Image Effects Programming

The Image Effects block modifies an image by pixel modifications. The image can be modified by using one out of seven available effect modes: grayscale-, sepia-, color selection-, negative-, emboss-, sketch- and sharpen effect. The effect selection and module bypassing is done by writing to the IMG_EFF_CTRL register. The image effects block can be configured for full range or limited range (BT.601) pixel values.

- Grayscale: In this mode only the luminance component of the image is processed. The chrominance part is set to 128.

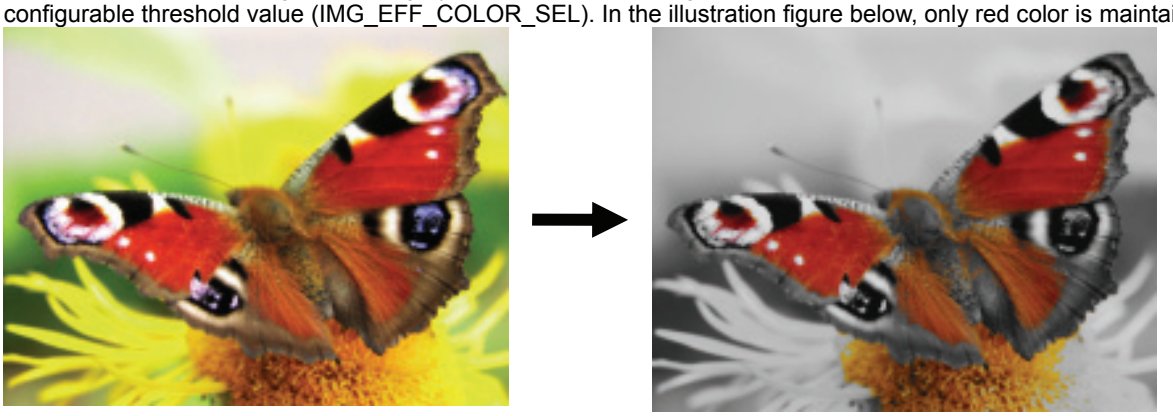


- Negative: An effect through inverting the luminance and chrominance part of a picture. The values are inverted in consideration of CCIR601 pixel value range (16...235 for luminance and 16...240 for chrominance).



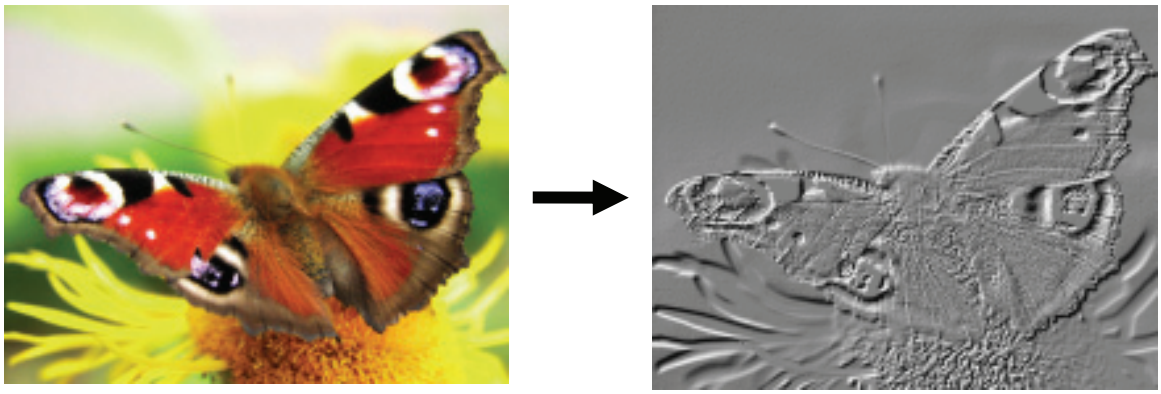
- Sepia: In sepia mode the two chrominance components are processed. The chrominance is modified in the appropriate brown hue to create a historical like image color. The new chrominance parts of resulting pixel are calculated regarding the programmed tint color value (IMG_EFF_TINT) and the luminance part of the original pixel. The Y component is passed through.

- Color selection: Converting picture to grayscale while maintaining selectable color components above a configurable threshold value (IMG_EFF_COLOR_SEL). In the illustration figure below, only red color is maintained.



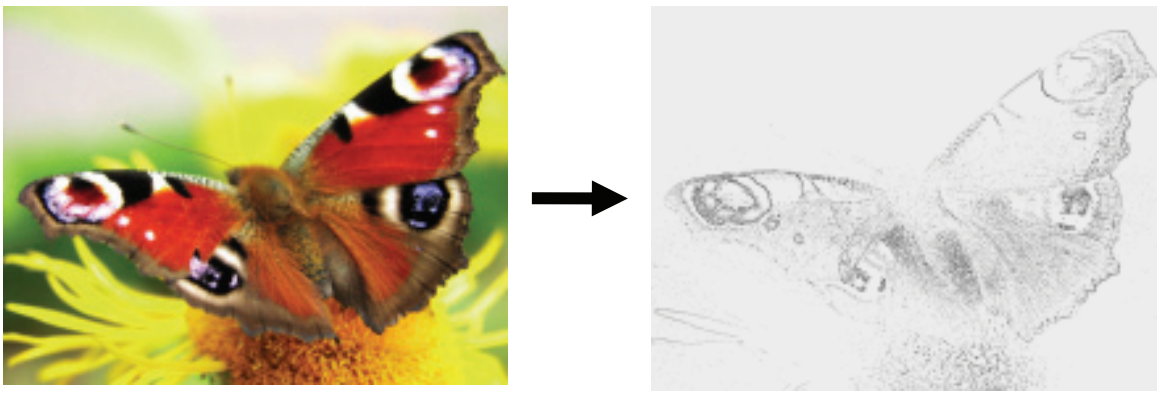
- Emboss: The emboss effect is created by the result of a 3x3 Laplace Filter with the luminance components. For emboss picture appearing (see Figure below) an offset 128 is added to the result. The chrominance components are not processed and are set to 128. The effect can be further configured by changing the matrix coefficients (IMG_EFF_MAT_1 to IMG_EFF_MAT_3). The default matrix values for emboss are:

1.1 **2**	1.2 **1**	1.3 **0**
2.1 **1**	2.2 **0**	2.3 **-1**
3.1 **0**	3.2 **-1**	3.3 **-2**



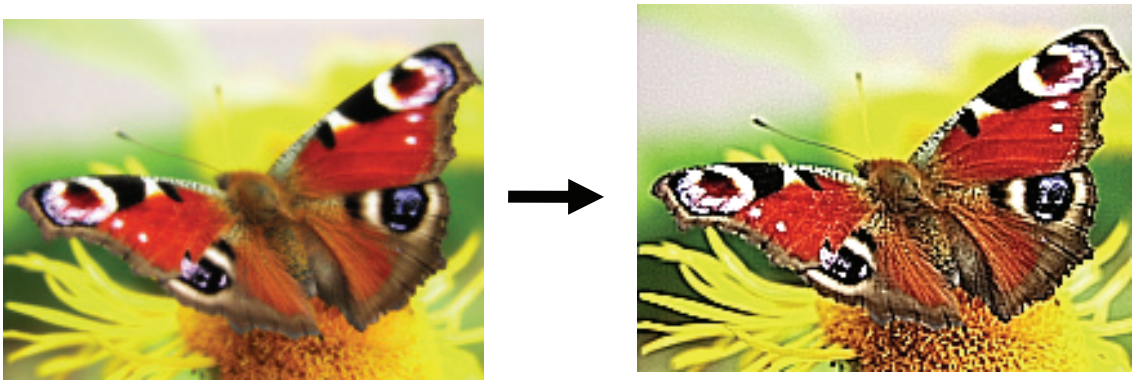
- Sketch: The sketch mode is also a result of the 3x3 Laplace Filter with an extracting of the edges in whitely background (see Figure below). A fixed threshold value of 31 is defined. All luminance values smaller 31 are set to 235 to visualize the whitely background. Other luminance values are subtracted from 235 to visualize the edges in picture. The chrominance components are not processed and are set to 128. The effect can be further configured by changing the matrix coefficients (IMG_EFF_MAT_3 to IMG_EFF_MAT_5). The default matrix values for sketch are:

1.1 **-1**	1.2 **-1**	1.3 **-1**
2.1 **-1**	2.2 **8**	2.3 **-1**
3.1 **-1**	3.2 **-1**	3.3 **-1**



- Sharpen: The sharpen effect is a result of the 3x3 Laplace Filter. The luminance signal is modified by adding a highpass signal with selectable factor (see Figure below). The factor is selectable in the range 0 to 1.875. A coring function is implemented which allows to define a threshold for the highpass signal. This may avoid amplifying noise too much. The chrominance signals are not modified. The matrix values for sharpen are:

1.1 **-1**	1.2 **-1**	1.3 **-1**
2.1 **-1**	2.2 **8**	2.3 **-1**
3.1 **-1**	3.2 **-1**	3.3 **-1**



Please note that the image effects block also utilizes the concept of shadow registers. This makes it possible to switch between the effects in a frame synchronized way without any visible artifacts even if the IMG_EFF_CTRL register was modified while image data is being processed. But this also implies that one have to set the *gen_cfg_upd* bit in the ISP main control register ISP_CTRL in order to get a configuration register update to take effect.

5.3.3 Superimpose Programming

The Superimpose module is able to combine the live image coming from the image sensor with a second bitmap image read from the system memory. The combined picture is then passed on through the rest of TcISP's processing chain. As shown in Figure 5.1 , the module is located after the ISP, color processing and image effects modules, but before the data path splits off into main and self path. This means that the bitmap image from memory won't get modified by ISP, color processing and image effects, and the combined image is passed to both main and self path. Thus, the Superimpose module can be used for both adding informative displays in the view finder picture (self path) and modifying the image to be encoded (main path).

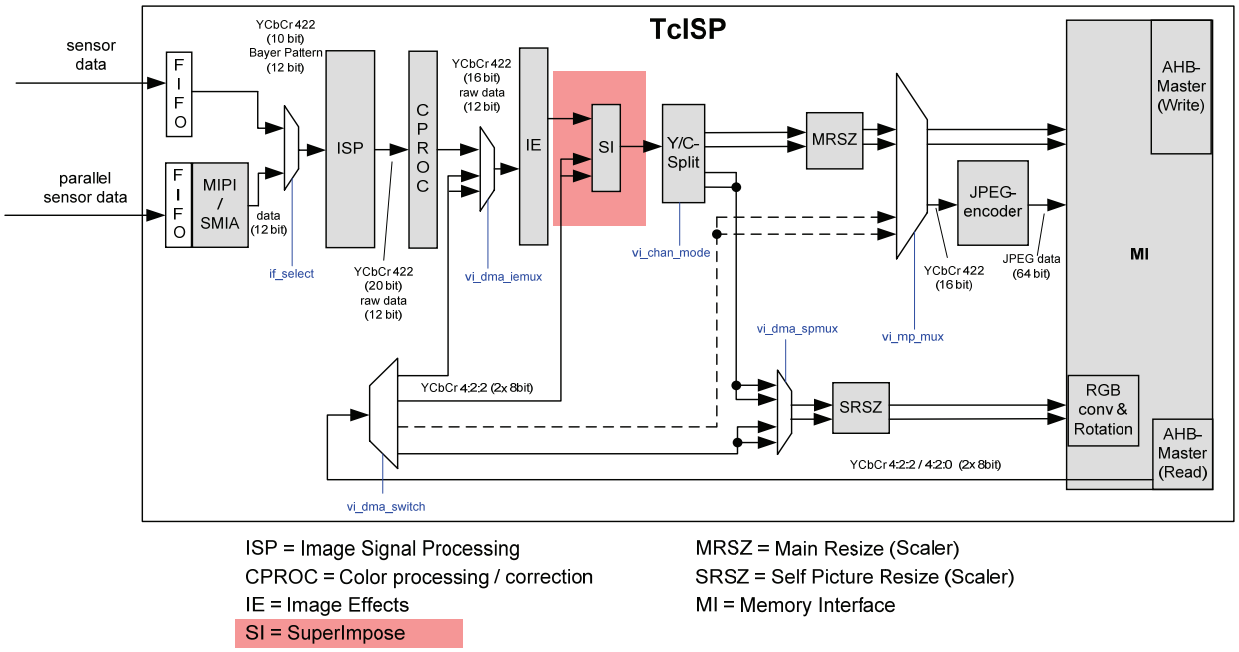


Figure 5.1 Location of the Superimpose module in the processing chain

In order to combine the sensor image with a bitmap, the memory interface must be configured to read the bitmap from the system memory by using its DMA-read feature (see chapter 5.3.5.4 , DMA-read feature programming). Further, the DMA-read multiplexer must be configured to pass the bitmap data to the Superimpose module. This can be done with the bits vi_dma_switch of the VI_DPCL register.
Note that the Superimpose module is only able to process image data in YCbCr 4:2:2 color format. Thus, the bitmap in system memory must also be available in this format, and the Superimpose module must be set to bypass mode if any other format (e.g. RAW data) is to be captured from the image sensor.
Note also that the DMA read port of the memory module must be set to 4-beat burst via MI_DMA_CTRL register to enable processing of the bitmap from memory without gaps. If 16-beat bursts are supported (M5_v4 only), then the Superimpose mode may be used with 8-beat bursts as well. An additional bit of this register is useful for Superimpose, which enables continuous operation of the DMA read port (dma_continuous_en). If this bit is not set, then the control software needs to start the DMA engine for each frame again. If set, this is not necessary.

5.3.3.1 Trivial case: Bypass mode

In case no image merging is to be done, the Superimpose module can be set to bypass mode. This can be done by setting bit super_imp_en in the SUPER_IMP_CTRL register to 0, which is also the default state after hardware reset.
In bypass mode, all other register settings of the Superimpose module are ignored. The data stream from the image sensor is passed on without any modification. The Superimpose module does not expect any data to be delivered from the DMA-read feature of the MI.

5.3.3.2 Use case: Overlay mode

The overlay mode can be used to insert a small picture read from system memory at a certain position inside the live picture coming from the image sensor.

Figure 5.2 illustrates that use case.

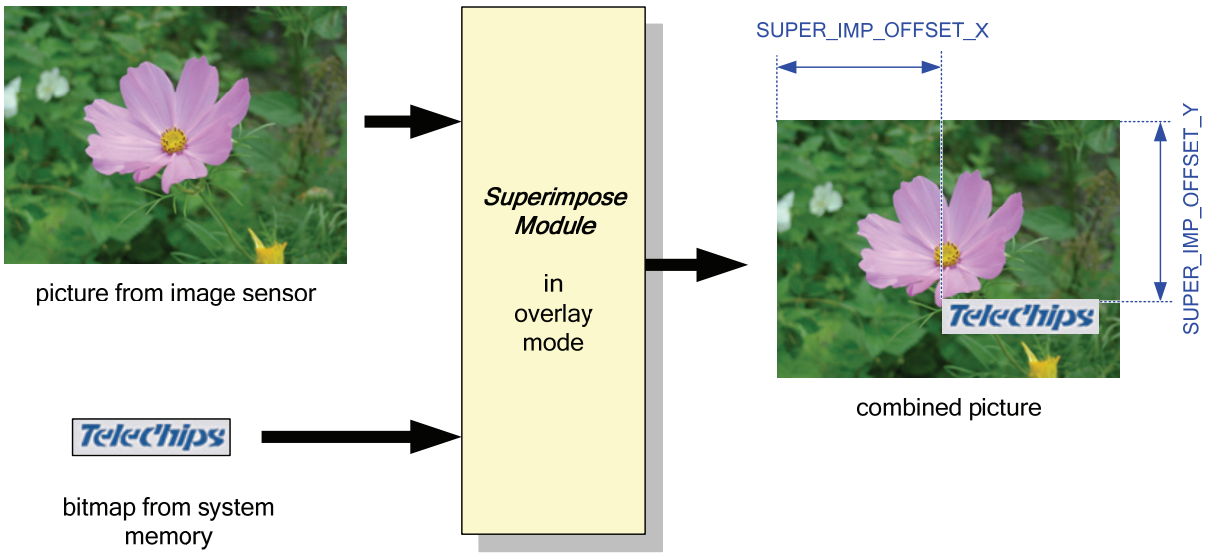


Figure 5.2 Superimpose module use case: overlay mode

For the overlay mode, following conditions must be met:

- The bitmap from system memory needs to be smaller than the picture coming from the image sensor.
- The sum of the value of register SUPER_IMP_OFFSET_X and the width of the bitmap from system memory must not exceed the width of the image sensor picture.
- The sum of the value of register SUPER_IMP_OFFSET_Y and the height of the bitmap from system memory must not exceed the height of the image sensor picture.

To enable the overlay mode, the registers of the Superimpose module must be configured as follows:

- SUPER_IMP_CTRL:
  super_imp_enable = 1, normal operation, no bypass.
  ref_image = 0, use image sensor picture as reference.
  transparency_mode = 1, transparency disabled
- SUPER_IMP_OFFSET_X
  Desired x-coordinate (in pixel) of the upper left edge of the bitmap from memory inside the combined output picture.
- SUPER_IMP_OFFSET_Y
  Desired y-coordinate (in pixel) of the upper left edge of the bitmap from memory inside the combined output picture.
- SUPER_IMP_COLOR_Y
  Value is ignored, because no color keying is performed in this mode.
- SUPER_IMP_COLOR_CB
  Value is ignored, because no color keying is performed in this mode.
- SUPER_IMP_COLOR_CR
  Value is ignored, because no color keying is performed in this mode.

5.3.3.3 Use case: Color keying mode

The color keying mode can be used to make the live picture from the image sensor be visible in certain, multiple and arbitrary shaped areas of the picture read from system memory. This is done by exchanging all pixels from the memory bitmap having the key color with the pixels coming from the image sensor. Figure 5.3 illustrates that use case. The key color is defined by the contents of the registers SUPER_IMP_COLOR_Y, SUPER_IMP_COLOR_CB and SUPER_IMP_COLOR_CR.
For the color keying mode, following conditions must be met:
The bitmap from system memory and the picture coming from the image sensor should have the same size[1].

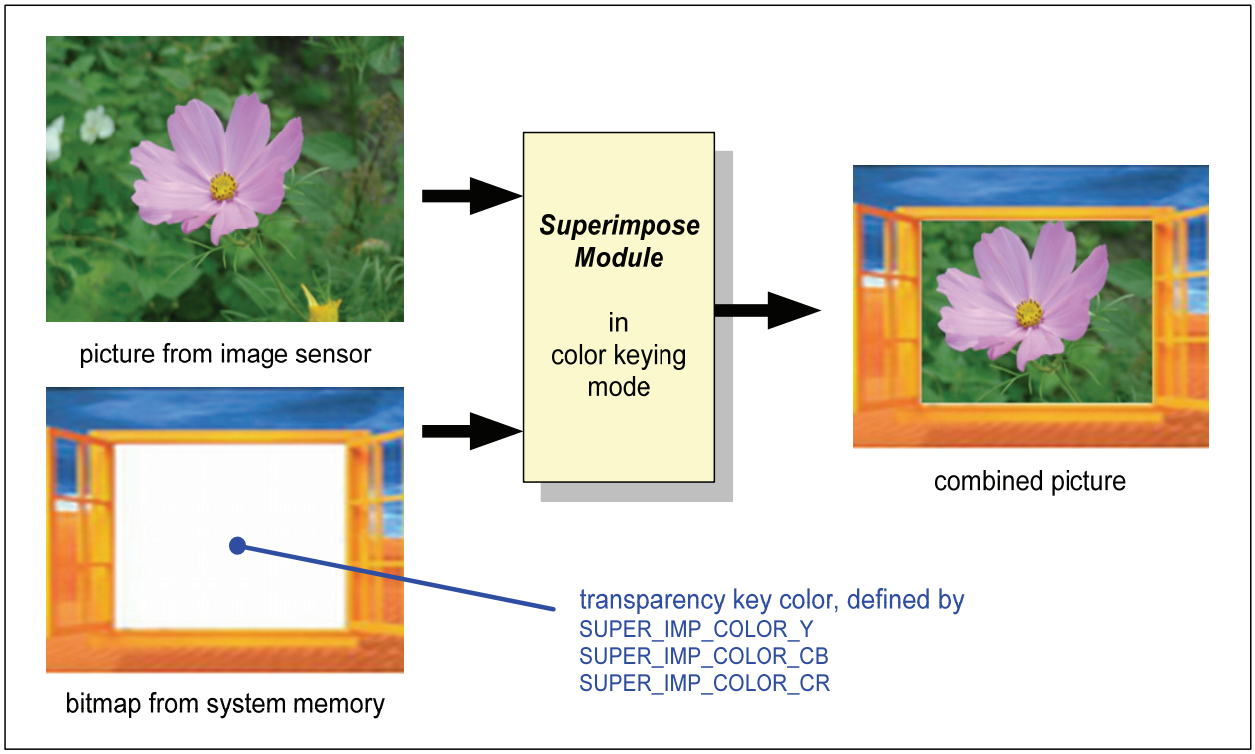


Figure 5.3 Superimpose module use case: color keying mode

To enable the color keying mode, the registers of the Superimpose module must be configured as follows:

- SUPER_IMP_CTRL:
  super_imp_enable = 1, normal operation, no bypass.
  ref_image = 1, use the bitmap from memory as reference.
  transparency_mode = 0, transparency enabled
- SUPER_IMP_OFFSET_X
  Must be set to 0 in this mode.
- SUPER_IMP_OFFSET_Y
  Must be set to 0 in this mode.
- SUPER_IMP_COLOR_Y
  Y-component of the key color in the bitmap from system memory.
- SUPER_IMP_COLOR_CB
  Cb-component of the key color in the bitmap from system memory.
- SUPER_IMP_COLOR_CR
  Cr-component of the key color in the bitmap from system memory.

[1] It is possible to use a picture coming from memory with smaller size than the picture coming from the image sensor, but normally this is not very useful. The output picture will have the size of the reference picture which is the picture coming from memory in this case.

5.3.4 Resize Programming

The scaler is implemented as a phase correct up- or downscaler with pixel accumulation using fixed point scale factors with a resolution of 14 bits for maximum flexibility. The following Figure 5.4 gives an idea of how the conversion from input image to a smaller output image is done. For more information about scaler functionality please refer to the application note.

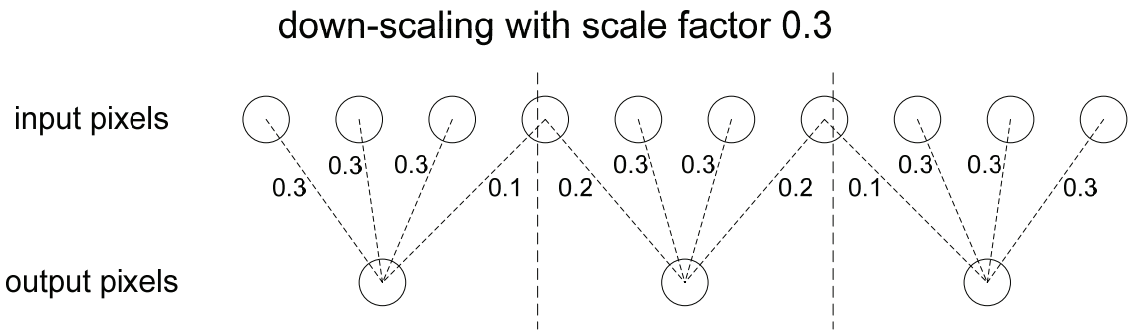


Figure 5.4 Scaling Algorithm for downscaling

The implemented scaler consists of separate horizontal and vertical scaling engine. The only difference is that the vertical scale engine needs line buffers used for pixel accumulation in the vertical direction. So, the maximum output image line length is 4096 pixels for the scaler in processing mode. In vertical bypass mode no vertical scaling is done.
For the horizontal scaling engines no output line length limitation exists. The following table gives an overview about the output image limitations coming from the needed line buffers.

Table 5.2 Scaling Limitations

	Bypass mode	Processing mode	
		maximum	minimum
horizontal	no limitation	4096 pixels	16 pixels
vertical	no limitation	3072 pixels (max. resolution of TCISP)	16 pixels

Vertical and horizontal scalers are further separated into independent luminance and chrominance scalers. These are used for color format conversions. This is done by programming different scale factors for luminance and chrominance scalers. If images with odd horizontal size shall be processed, then the chrominance scaling factors need to be programmed separately, because the number of Cb and Cr components are not the same (YCbCr 4:2:2 format). Therefore separate registers exist for horizontal scaling: SCALE_HCB and SCALE_HCR.
To enable scaling the corresponding horizontal and/or vertical scaling engines must be enabled in MRSZ_CTRL.

The scaling factors are 14 bit values and they must be calculated using the following formulas:

For downscaling $$scale = \mathrm{int}\left(\frac{size\_out - 1}{size\_in - 1} \times 2^{16} \right) + 1$$

For up-scaling $$scale = \mathrm{int}\left(\frac{size\_in - 1}{size\_out - 1} \times 2^{16} \right)$$

Additionally the start phase of the output pixel can be programmed (phase offset). This is to be used if co-sited to non co-sited conversion has to be performed. For MPEG2 compression vertical chrominance phase shift, for MPEG1 and MPEG4 both vertical and horizontal chrominance data shifts are required. For horizontal conversion the chrominance phase must be shifted by half the distance between two luminance values resulting in the shift of 1/4 of the distance between two chrominance values of the same color component. For vertical conversion also a phase shift of 1/2 between two luminance lines is needed and thus an offset of 1/4 for chrominance lines has to be programmed because of the vertical sub-sampling of 0.5.
Note: Phase shifting is not available during bypass of a scaling engine. If phase shifting is required then the corresponding scaling part must be enabled and the maximum downscale factor of 0xfff should be programmed. For VGA resolution this results in an accumulated pixel failure of 1/3, which is about 0.0005%.

5.3.4.1 Downscaling

For downscaling, the input picture pixels are considered as areas. Pixels of the downscaled picture are calculated as an average of input picture pixels, weighted according to their contributing area part.

Definitions:
y : input data
Y : pixelbuffer
Δ : scaling factor
Σ : accumulator
first : 1st-pel-of-line flag
in_h_end: last-pel-flag
out_cnt: pixelcounter
size_h: configured
ouput line width

preload: Σ := phase
Δ := scale
Y := 0
first := 1
out_cnt := 0

Remarks:
Σ is limited to $-1<\Sigma<1$,
if $\Sigma+\Delta\geq 1$
then $\Sigma:=\Sigma+\Delta$ means
$\Sigma:=\Sigma+\Delta-1$

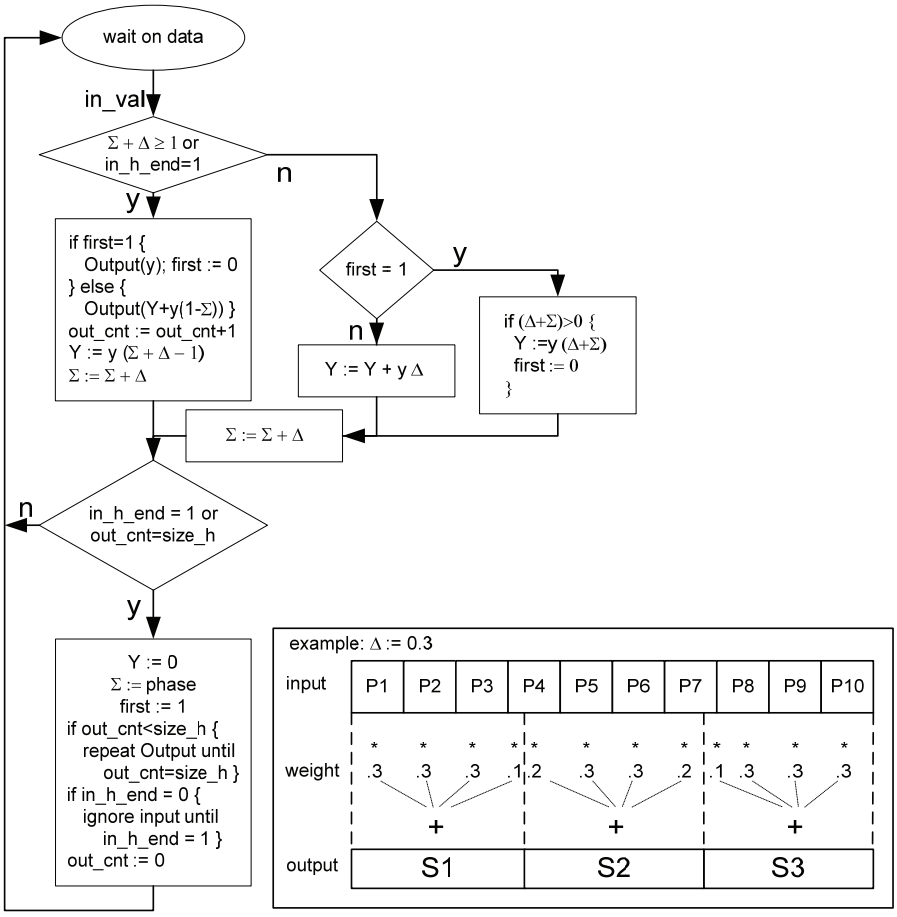


Figure 5.5 Horizontal downscaling functionality, vertical scaler works accordingly

Table 5.3 Example Scaling Calculations for downscaling

Register	Value	Comment/Calculation
UXGA 4:2:2 to QCIF 4:2:2 (co-sited, for display)		
SRSZ_CTRL	0x0F	enable all scalers for downscaling
SRSZ_SCALE_HY	0x702	int((176-1)/(1600-1)*2^16)+1
SRSZ_SCALE_VY	0x7A3	int((144-1)/(1200-1)*2^16)+1
SRSZ_SCALE_HCB	0x6F8	int((88-1)/(800-1)*2^16)+1
SRSZ_SCALE_HCR	0x6F8	same as scale_hcb
SRSZ_SCALE_VC	0x7A3	same as scale_vy
SRSZ_PHASE_HY	0x000	no phase offset
SRSZ_PHASE_VY	0x000	no phase offset
SRSZ_PHASE_HC	0x000	no phase offset
SRSZ_PHASE_VC	0x000	no phase offset
UXGA 4:2:2 to SXGA 4:2:2 (co-sited, for JPEG encoding)		
MRSZ_CTRL	0x0F	enable all scalers for downscaling
MRSZ_SCALE_HY	0x3332	int((1280-1)/(1600-1)*2^16)+1
MRSZ_SCALE_VY	0x369C	int((1024-1)/(1200-1)*2^16)+1
MRSZ_SCALE_HCB	0x3330	int((640-1)/(800-1)*2^16)+1
MRSZ_SCALE_HCR	0x3330	same as scale_hcb
MRSZ_SCALE_VC	0x369C	same as scale_vy
MRSZ_PHASE_HY	0x000	no phase offset
MRSZ_PHASE_VY	0x000	no phase offset
MRSZ_PHASE_HC	0x000	no phase offset
MRSZ_PHASE_VC	0x000	no phase offset

The resize sub-module also supports dynamic configuration updates. So do not forget to set *cfg_upd* in xRSZ_CTRL register to “1” for immediate configuration update.

5.3.4.2 Up-scaling

The Resize Module is configurable for horizontal and vertical up- or down-sampling. It supports output frames of up to 4096x3072.

The base for calculation of the up-scaled picture is the interpolation of two adjacent pixels. The interpolation factors are taken from a programmable lookup table. The distance between two input pixels is divided into 64 intervals with one LUT entry for each interval. For calculation of the output sample, the LUT entry of the interval containing the position of the output sample is used.

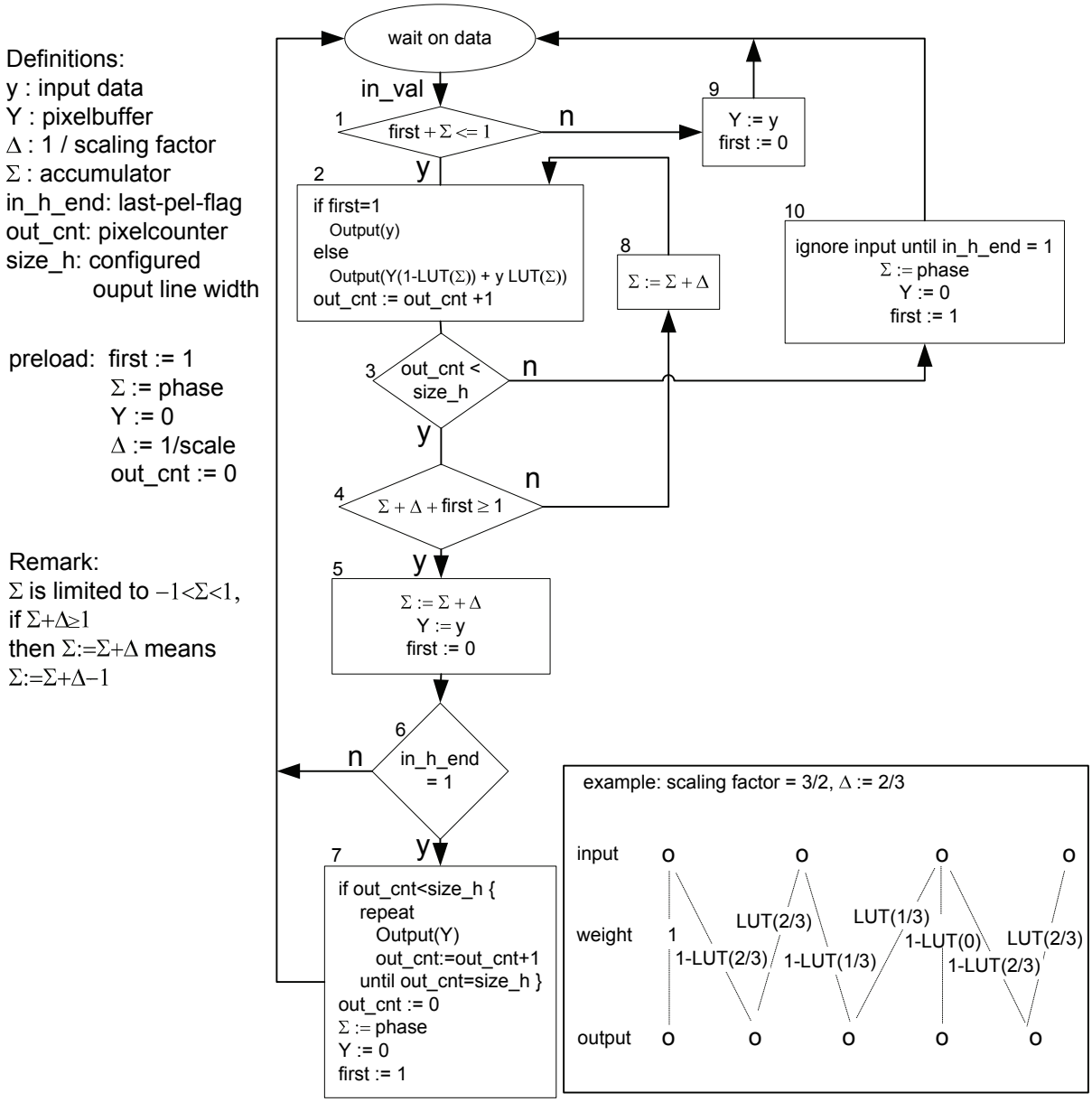


Figure 5.6 Horizontal up-scaling functionality, vertical scaler works accordingly

Separated values for the scaling factors of the luminance and the chrominance components allow for conversion from YUV4:2:0 to YUV4:2:2 color format.
Phase shift registers are provided to shift the output pixel positions with respect to the input pixel positions. This allows for shifting of the chrominance pixel positions relatively to the luminance pixel positions. Negative phase shift values are represented using 2's complement.

Lookup Table

The characteristic of the up-scaling algorithm is programmable using a 64-entry 6-bit lookup table. In the up-scaling algorithm, an output pixel is always calculated by interpolation of two neighbored input pixels. These two input pixels are weighted according to the position of the output pixel between the two input pixels, which is given by the accumulator value Σ. For each position between two input Pixels, the desired weighting factor is programmable using the lookup table, where LUT(0)/64 is used as the weighting factor for positions $0 \le \Sigma < 1/64$, LUT(1)/64 is used for positions $1/64 \le \Sigma < 2/64$, and so on, until LUT(63), which is used for positions $63/64 \le \Sigma < 1$.
In the following, we will write LUT(Σ), with $0 \le$ LUT(Σ) < 1, instead of LUT(int(64 Σ))/64.

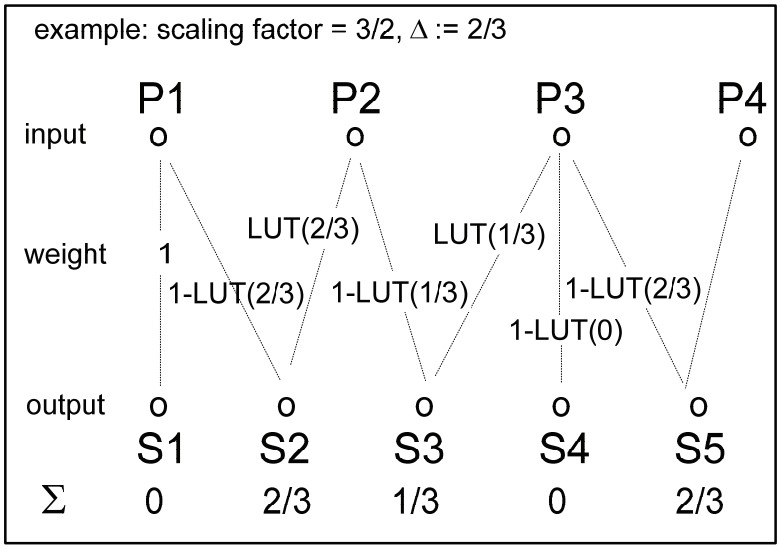


Figure 5.7 Use of Lookup Table

E.g., in Figure 5.7 , output pixel S3 is calculated using input pixels P2 and P3. As S3 is nearer to P2 than it is to P3, P2 should contribute to the value of S3 more than P3 does. In the simplest case, the contribution of the input pixel may be proportional to their distance from the output pixel. For S3, which is at position Σ =1/3 between P2 and P3, this means that

S3 := 2/3 P2 + 1/3 P3,

or, expressed in terms of Σ,

S3 := (1-Σ)P2 + Σ P3.

To achieve this behavior, the 64 entries of the LUT have to be programmed with LUT(i)=i, where i goes from 0..63.

For large scaling factors (size_out/size_in > 2), a linear interpolation leads to blurred edges in the output filter. In this case, a non-linear characteristic might be more appropriate, where the contribution of the nearer input pixel is higher and the contribution of the distant input pixel is lower than for linear characteristic.

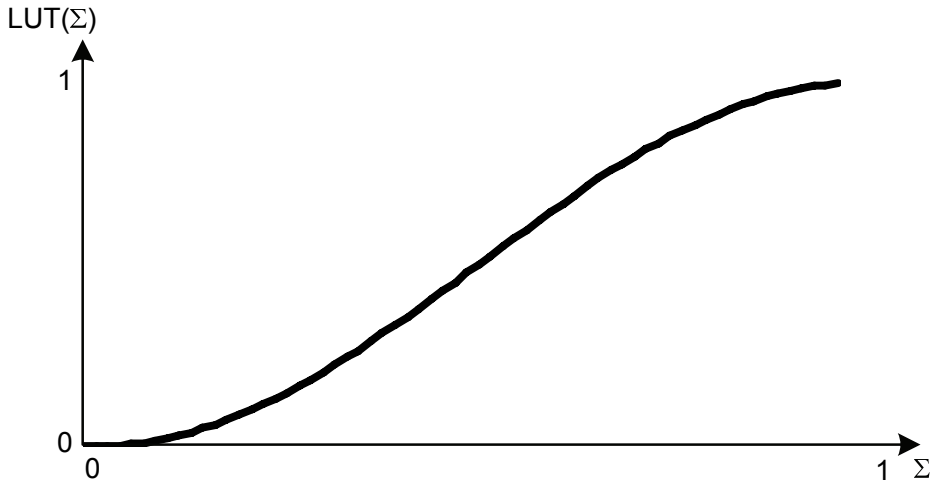


Figure 5.8 Example for a non-linear Lookup Table

If the Lookup Table is programmed in this way, this will lead to sharper edges at the cost of a more "block-like" output picture. In any case, the LUT should be programmed with a function where

- LUT(0) = 0, so that an output pixel at the same position as an input pixel is generated by copying the input pixel
- LUT(63/64) = 63/64, to ensure that an output pixel just left of an input pixel gets a similar value as an output pixel just right to the same input pixel.
- LUT(½) = ½, so an output pixel in the middle between two input pixel will get the same contribution from both input pixels.

Table 5.4 Example Scaling Calculations for up-scaling with linear lookup table

Register	Value	Comment/Calculation
QCIF 4:2:2 to QVGA 4:2:2 (co-sited, for display)		
SRSZ_CTRL	0x0FF	enable all scalers for up-scaling
SRSZ_SCALE_HY	0x231C	int((176-1)/(320-1)*2^16)
SRSZ_SCALE_VY	0x264A	int((144-1)/(240-1)*2^16)
SRSZ_SCALE_HCB	0x2304	int((88-1)/(160-1)*2^16)
SRSZ_SCALE_HCR	0x2304	same as scale_hcb
SRSZ_SCALE_VC	0x264A	same as scale_vy
SRSZ_PHASE_HY	0x000	no phase offset
SRSZ_PHASE_VY	0x000	no phase offset
SRSZ_PHASE_HC	0x000	no phase offset
SRSZ_PHASE_VC	0x000	no phase offset
SRSZ_SCALE_LUT_ADDR	0x00	first address for lookup table
SRSZ_SCALE_LUT	0x00	first entry for lookup table
SRSZ_SCALE_LUT_ADDR	0x01	second address for lookup table
SRSZ_SCALE_LUT	0x01	second entry for lookup table
SRSZ_SCALE_LUT_ADDR	0x02	third address for lookup table
SRSZ_SCALE_LUT	0x02	third entry for lookup table
…	…	…
SRSZ_SCALE_LUT_ADDR	0x3F	last address for lookup table
SRSZ_SCALE_LUT	0x3F	last entry for lookup table

The resize sub-module also supports dynamic configuration updates. So do not forget to set *cfg_upd* in xRSZ_CTRL register to “1” for immediate configuration update.

5.3.5 Memory Interface Programming

The TCISP memory interface unit (MI) is responsible for reading/writing image data from/to the system memory. As shown in Figure 5.9 , the MI has three main tasks, which can operate independently:

- Take image data from the main path, either in YCbCr or RAW format, or already JFIF-encoded from the JPEG encoder, and write it into certain data buffers located in the system memory.
- Take image data from the self path in YCbCr format, optionally change its orientation and/or convert it to RGB data, and write it into certain data buffers located in the system memory.
- Take image data in YCbCr format from the system memory, and feed it into various other parts of the TcISP processing chain. This feature is referred to as "DMA-read".

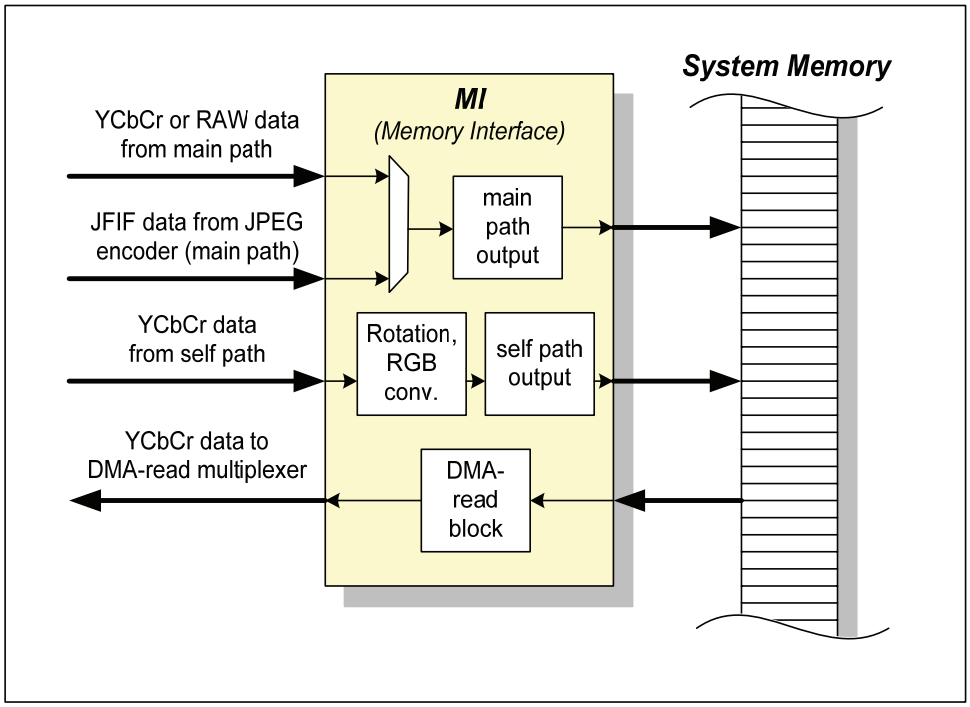


Figure 5.9 Memory Interface Unit overview

5.3.5.1 External Output Buffers

Depending on which pathes are enabled and which output format is selected, the MI unit can simultaneously output image data to up to 6 independent buffers. The buffers are named according to their relation to the main or self data path and their usage in the (most common) YCbCr planar output format: main-Y-buffer, main-Cb-buffer, main-Cr-buffer, self-Y-buffer, self-Cb-buffer and self-Cr-buffer.
The buffers itself are not part of the TcISP IP; they are system memory areas accessible from the bus that is also connected to the TcISP. One can think of the MI as a kind of DMA controller, which writes to the configured memory locations without any CPU assistance. To configure the memory locations, each of the 6 buffers mentioned earlier has a set of registers assigned to it.

- MI_xx_xx_BASE_AD_INIT
  This is the base address of the memory area to use for that buffer. (e.g. the lowest address of that buffer) Note that buffer base addresses must be 4-byte-aligned. In case the target system uses some kind of PMMU, the physical address must be specified, because virtual-to-physical translation is not possible for this kind of DMA-like transfer.

- MI_xx_xx_SIZE_INIT
  This is the size of the memory area in bytes. Note that the size is also required to be a multiple of 4. It is ensured that for write operations in this particular buffer, the MI will only write to memory locations from the base address to the base address + the size - 1. If the MI reaches the end of the memory area while continuously writing data, the buffer "wraps around" and it proceeds to write starting from the base address again. This wrap around event can also be used to trigger an interrupt. See the MI_RIS register for details.

- MI_xx_xx_OFFS_CNT_INIT
  This is the initial offset (in bytes) for writing to the buffer after e.g. a soft reset. Usually, to write to the buffer from the very beginning, an offset of zero is to be programmed into this register.

- MI_xx_xx_OFFS_CNT_START
  This is a read-only register, which holds the offset at which the MI had written the last processed frame. This register is updated at frame end, so at any given time, it contains always the starting offset of the last completely processed frame.

One of the buffers, the main-Y-buffer, is additionally able to trigger an interrupt as soon as a particular filling level is reached. This level can be configured in terms of an offset count (in bytes) from the base address using the MI_MP_Y_IRQ_OFFS_INIT register.
Further, the MI supports updating the buffer configuration seamlessly between two consecutive frames without dropping a frame by using the shadow register. Two bits in the MI_CTRL register (init_offset_en and init_base_en) can be used to select whether the base address and size, or the offset counters, or both are being updated at the next configuration update pulse.

It should be noted that it is only possible to update the selected parts of all the buffers at once. It is not possible to re-configure only specific buffers and leave the others untouched.
As mentioned earlier, there are some restrictions permitting only 4-byte-alignment buffer base addresses, and both sizes and offsets also need to be a multiple of 4. This is because the MI writes the data to the memory areas in 4-byte-accesses. It is possible to change the endianess of these accesses with the byte_swap bit in the MI_CTRL register. Figure 5.10 illustrates this with the example use case of writing 8-bit Y-samples from a frame with a width of 5 pixels. The setting of the byte_swap bit affects the write accesses to all buffers regardless of whether YCbCr, RGB, RAW or JFIF data is to be written. Note that "addr" is a 4-byte aligned address in this example. For all other output schemes illustrated and discussed in this chapter, a setting of byte_swap = 0 is assumed. In case of 64-bit output interface the same applies accordingly for 8 bytes.

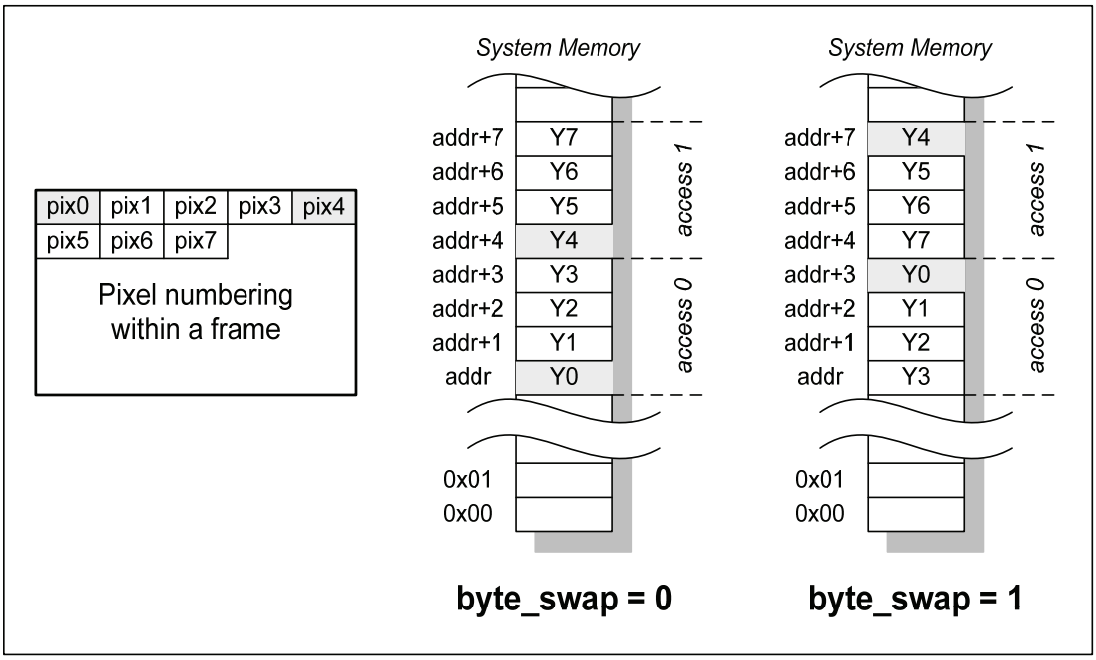


Figure 5.10 Effect of the byte_swap bit in MI_CTRL

5.3.5.2 Main Path output programming

The main path output sub-module of the MI is intended to be used for video stream generation (YCbCr for further compressing via software encoders or motion JPEG), or high resolution still image snapshot (JPEG encoded or RAW image sensor data). To achieve this, it can be configured to output one out of three data sources:

- YCbCr data from the main path scaler.
  To enable this, bit mp_enable in the MI_CTRL register is to be set. The data can be output in planar, semi-planar or interleaved format, as shown in Figure 5.11 . The selection is done with the bits mp_write_format in the MI_CTRL register. In semi-planar mode, the main-Cb-buffer is used for the multiplexed Cb/Cr values and the main-Cr-buffer remains unused. In interleaved mode, only the main-Y-buffer is used for all the multiplexed values.

- RAW data from the main path scaler.
  This mode is enabled by setting the bit raw_enable in the MI_CTRL register. RAW data is assumed to be received via the same way as YCbCr data, but is written to the main-Y-buffer only. In RAW mode, the mp_write_format bits of the MI_CTRL register are used to select between 8-bit mode (which will write only the 8 most significant bits of every sample) and 12-bit mode (which will write the full 12 bit into 2 bytes of the output buffer. This implies that for the 12 bit mode, a buffer size twice as much is necessary as for the 8-bit mode, and the 4 least significant bits of every 16-bit value are unused and remain zero.

- JFIF data from the JPEG encoder.
  This mode is enabled by setting the bit jpeg_enable in the MI_CTRL register. The data from the JPEG encoder is written to the main-Y-buffer. Both main-Cr-buffer and main-Cb-buffer remain unused.

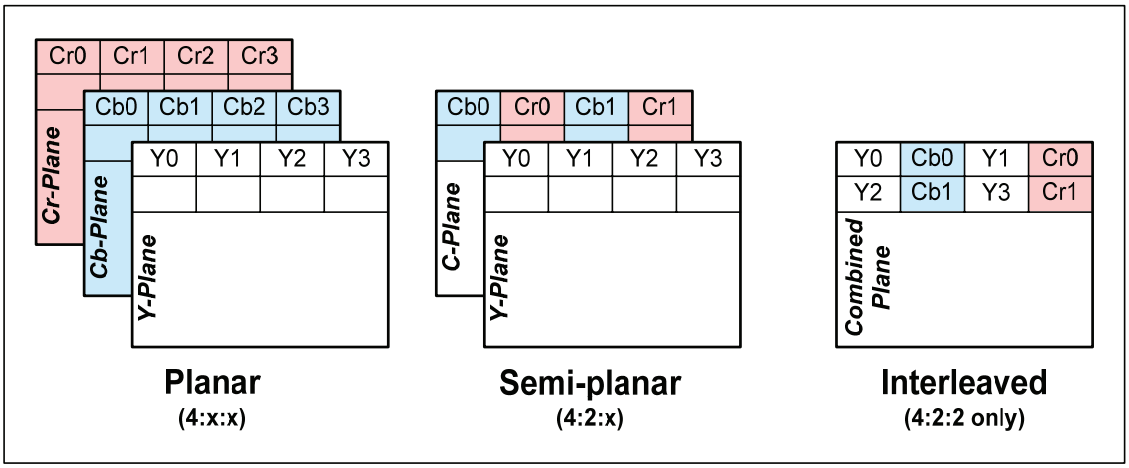


Figure 5.11 YCbCr Data organization

In some RAW data modes and in case the frame gets JPEG-encoded, the actual number of bytes to write per frame depends on the input signal and/or the compression ratio, and is not known in advance. Thus, this number is given in the read-only MI_BYTE_CNT register, which is updated at frame end only. So this register always shows the number of bytes written to the main-Y-buffer for last completely processed frame.
It is assumed that all the sub-modules within TcISP's processing chain located in front of the MI are configured in a way that the input data to the MI main path corresponds to what it is configured for. If this is not the case, the behavior of the MI is not predictable.
If output via the main path is not needed (e.g. if only a viewfinder is to be displayed using the self path), the main path can be disabled by setting all three bits mp_enable, raw_enable and jpeg_enable in the MI_CTRL register to zero.

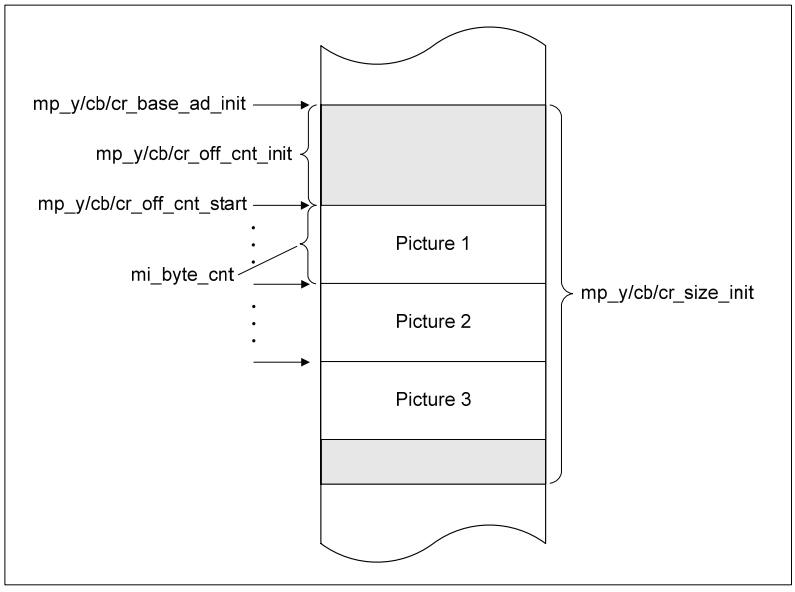


Figure 5.12 Buffer address scheme, valid for main picture buffers

5.3.5.3 Self Path output programming

The self path output sub-module of the MI is intended to be used for viewfinding purposes. Thus, it can only be fed with YCbCr data from the self path scaler, but is far more flexible in terms of output modes compared to the main path output sub-module.
To enable self path output in general, the bit sp_enable in the MI_CTRL register is to be set.

5.3.5.3.1 Additional configuration of the self path output buffer

Because of the extended capabilities like on the fly image rotation, mirroring and flipping during output, and to enable writing the frame directly into a bigger frame buffer, the MI needs some more information about the frame matrix to be written. Thus for the self path buffer, there are some registers additional to those shown in chapter 5.3.5.1 :

- MI_SP_Y_PIC_WIDTH
  Image width of the self picture Y component (or RGB picture) in pixels. This value always refers to the width of the output image. This is particularly important if the picture is to be rotated on the fly during output.

- MI_SP_Y_PIC_HEIGHT
  Image height of the self picture Y component (or RGB picture) in pixels. This value always refers to the height of the output image. This is particularly important if the picture is to be rotated on the fly during output.

- MI_SP_Y_LLENGTH
  Line length of the output buffer in pixel, also known as "line stride". This value becomes important if the MI shall write directly into a frame buffer, whose width is bigger than the width of the self picture. In this case, the MI is not allowed to output consecutive lines back-to-back, but must increment the output address by a certain amount before the start of every line. MI_SP_Y_LLENGTH must be set to the width of the whole frame buffer then. If the feature of inserting the self picture in a bigger frame buffer is not to be used, MI_SP_Y_LLENGTH must be set to the same value as MI_SP_Y_PIC_WIDTH.

- MI_SP_Y_PIC_SIZE
  Image size of the Y component (or RGB picture) in pixel. Set this value to the product of MI_SP_Y_PIC_HEIGHT and MI_SP_Y_LLENGTH. Regardless of the fact that this can be calculated from the other values, it has been decided to add a separate configuration register, because it is more economical to perform a single multiplication at initialization time in software than to add a hardware multiplier.

The registers SP_Y_LLENGTH, SP_Y_PIC_WIDTH, SP_Y_PIC_HEIGHT and SP_Y_PIC_SIZE are used, if rotation, flipping or line stride features are used. The line stride feature becomes active, if the registers SP_Y_PIC_WIDTH and SP_Y_LLENGTH are not equal. So, if both registers are set to zero (reset values) the line stride feature is switched off and the mentioned 4 registers are not used at all. The rotation and flipping feature is switched on by using the corresponding enable bits "rot, v_flip, h_flip" in MI_CTRL register. If the self path shall be used without rotation, flipping or line stride feature, it is not necessary to set the mentioned registers, but setting them to appropriate values does not hurt. Please note that there are restrictions for the picture width in rotation, flipping or line stride mode. For details please have a look at the register description.
The purpose of these four additional registers is illustrated in Figure 5.13 for the typical case of writing the self picture directly into a larger frame buffer. Additionally in this figure, some suggestions were made about how to set the registers for base address and size of the buffer. If these suggestions were followed, no adjustments of the address or the offset is necessary during operation, and all incoming self-pictures were written automatically in the same area inside the frame buffer.

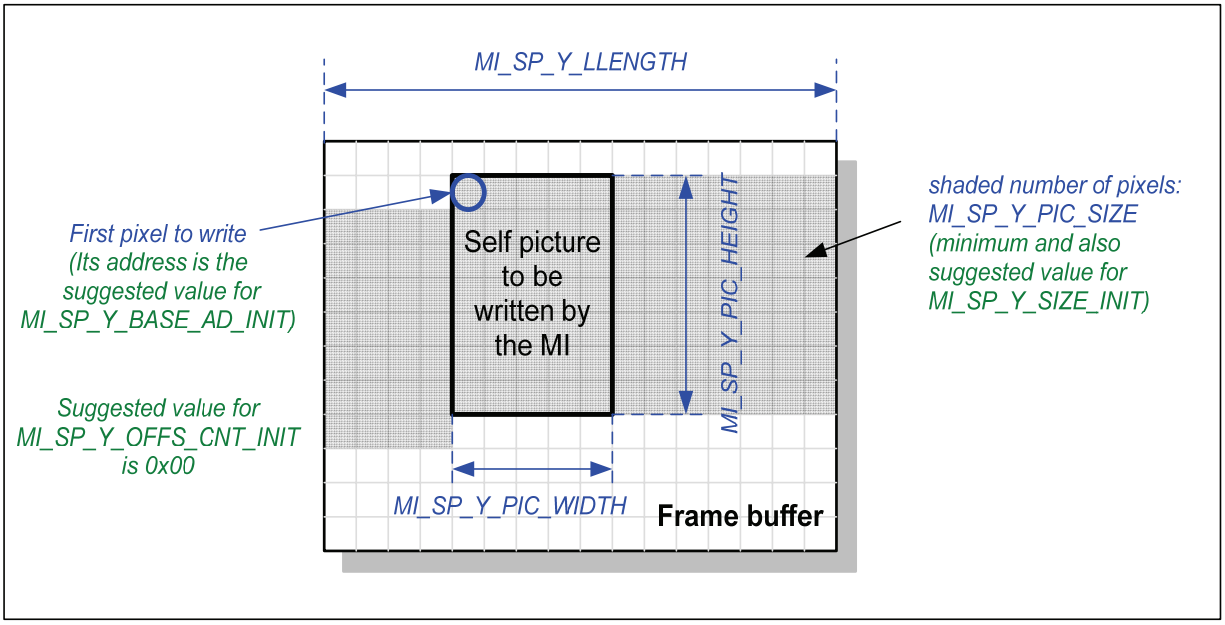


Figure 5.13 Typical case of writing the self picture directly into a frame buffer

Figure 5.14 shows a more general address scheme for self picture buffer, where base address, size and offset are not aligned with dimensions of self picture to be written.
Note that unlike the other buffer configuration registers, the additional registers MI_SP_Y_PIC_WIDTH, MI_SP_Y_PIC_HEIGHT, MI_SP_Y_PIC_SIZE and MI_SP_Y_LLENGTH do not use the shadow register concept, so changes of the values become effective immediately. Care should be taken to change the values only at times when no picture data is processed in the self path.

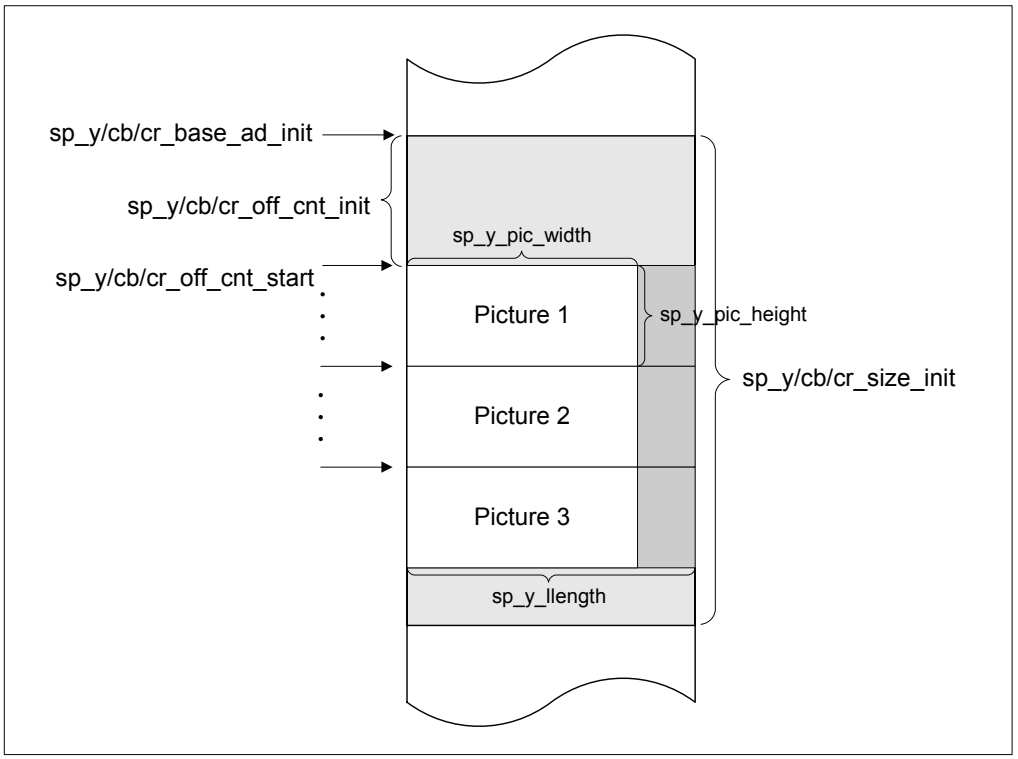


Figure 5.14 More general buffer address scheme, valid for self picture buffers

5.3.5.3.2 Input and Output formats

One may have noticed that the additional buffer configuration values are only needed to be specified for the self-Y-buffer and no corresponding registers seem to exist for the both self-picture chrominance buffers. The reason is that the MI calculates the corresponding values for the Cb and Cr buffer internally. In order to do that correctly, it needs to know the chroma sub sampling format of the incoming YCbCr image data. This information must be provided by programming the bits sp_input_format of the MI_CTRL register. Note that the self path can be fed with YCbCr data only.
For the self path, the MI can perform on-the-fly format conversion while the data is being written to the output buffers. Thus, the desired target format must be selected by programming the bits sp_output_format of the MI_CTRL register. Not all combinations of input and output format are possible. See Table 5.6 for the supported ones. Combinations that are not mentioned in this table are not supported.
If any of the YCbCr modes is selected, the data can be output in planar, semi-planar or interleaved format, as shown in Figure 5.11 . The selection is done with the bits sp_write_format in the MI_CTRL register. In semi-planar mode, the self-Cb-buffer is used for the multiplexed Cb/Cr values and the self-Cr-buffer remains unused. In interleaved mode, only the self-Y-buffer is used for all the multiplexed values.
If an RGB-mode is selected, refer to
Figure 5.15 for the expected bit and byte layout of the output data.

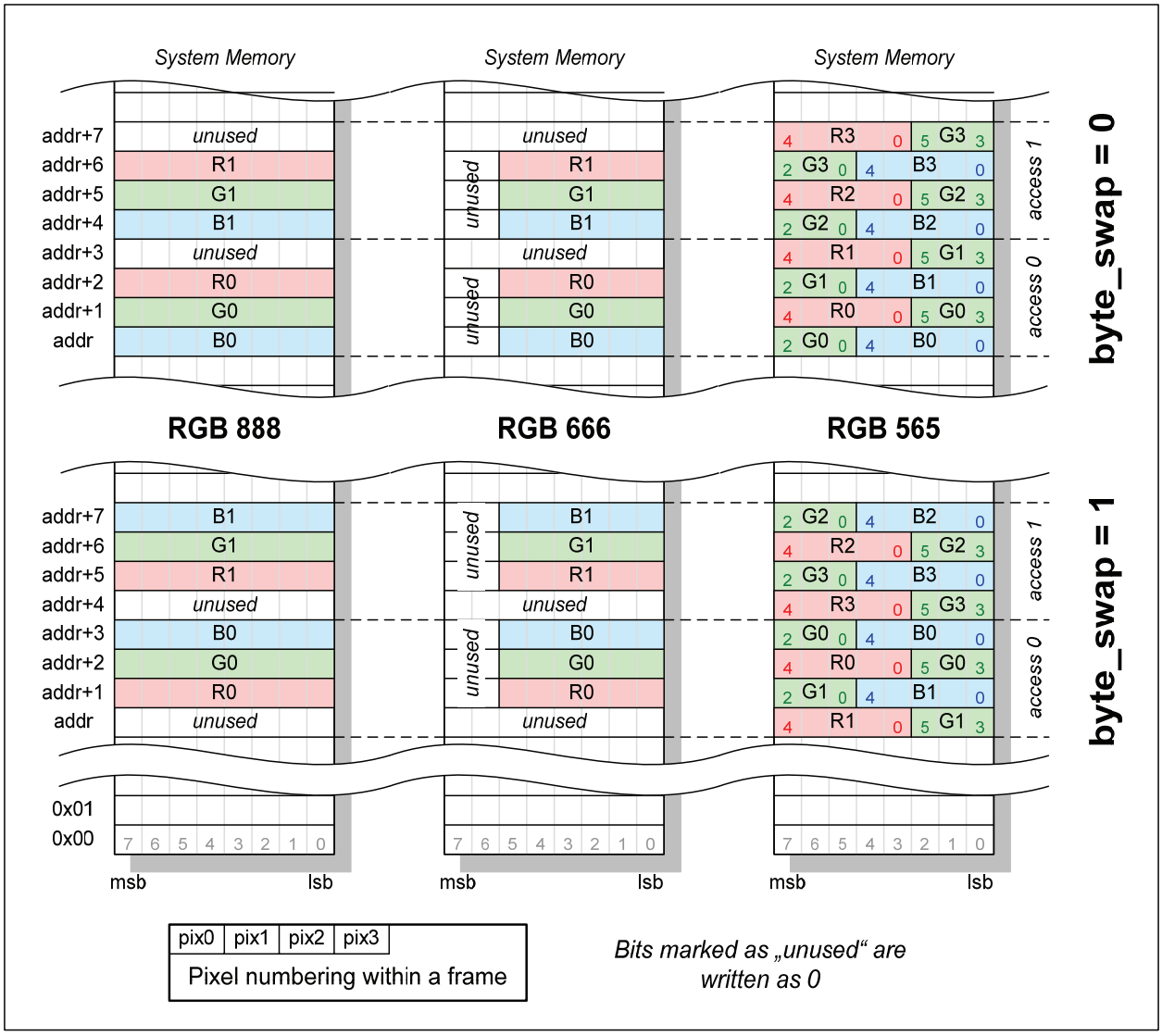


Figure 5.15 RGB output mode overview

5.3.5.3.3 Picture orientation

In addition to the format conversion, the orientation of the self picture can also be modified by flipping, mirroring and/or rotating on the fly during output. This is achieved by using parts of the JPEG encoder buffer as a rotation buffer and by generating a different address sequence pattern while writing to the system memory. Not all combinations of input/output formats and orientations are possible. See Table 5.6 for the supported ones. Combinations that are not mentioned in this table are not supported.
Note that the rotation feature requires the self-path to be operated with 4-beat bursts only. 8- or 16- beat bursts may not be configured if rotation is used, because this would require more memory for the rotation buffer.
The orientation of the output is controlled with three bits called "h_flip", "v_flip" and "rot" in the MI_CTRL register. See Table 5.5 for the resulting total of 8 possible orientations.

Table 5.5 Self picture orientations

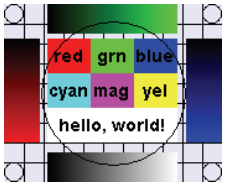
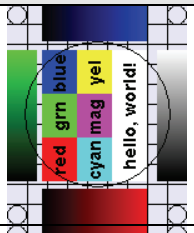
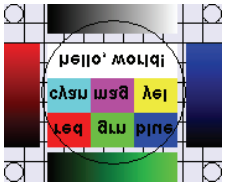
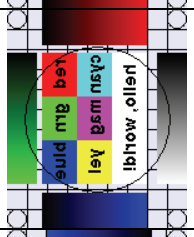
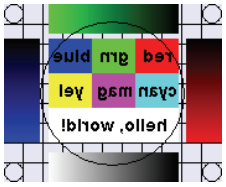
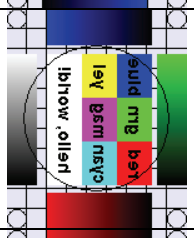
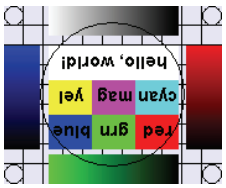
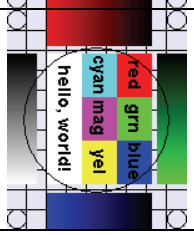
Example	Picture orientation	rot	h_flip	v_flip		Example	Picture orientation	rot	h_flip	v_flip
	Original	0	0	0			rotation 90˚ left	1	0	0
	vertical flipping	0	0	1			rotation 90˚ left and vertical flipping	1	0	1
	Horizontal flipping	0	1	0			rotation 90˚ left and vertical flipping	1	1	0
	rotation 180°	0	1	1			rotation 90˚ right	1	1	1

Table 5.6 Valid self picture format combinations and orientations

input format of MI	output format of MI	rotation	horizontal flipping	vertical flipping
YCbCr 4:4:4	YCbCr 4:4:4 planar	no	no	no
	YCbCr 4:4:4 semi planar	no	no	no
YCbCr 4:2:2	YCbCr 4:4:4 planar	yes1	yes	yes
	YCbCr 4:2:2 planar	no	yes2	yes2
	YCbCr 4:4:4 semi planar	no	no	no
	YCbCr 4:2:2 semi planar	no	no	no
	YCbCr 4:2:2 interleaved	no	no	no
	RGB 888	yes	yes	yes
	RGB 666	yes	yes	yes
	RGB 565	yes	yes	yes
YCbCr 4:2:0	YCbCr 4:2:0 planar	no	no	no
	YCbCr 4:2:0 semi planar	no	no	no
YCbCr 4:0:0	YCbCr 4:0:0 planar	no	no	no

[1] Only singles accesses on the AHB bus are generated.
[2] mi_sp_y_pic_width must be a multiple of 8 pixel.

5.3.5.4 DMA-read feature programming

The DMA-read sub-module of the MI is used to feed YCbCr image data located in the system memory back to various modules of the TcISP processing chain. This feature is intended to be used for several different use cases:

- Mixing live video from the image sensor with frames read from system memory. (see Superimpose module)
- Substituting the image sensor input with frames read from system memory for post-processing. This includes use-cases like digital zoom (with main and/or self scaler configured for up-scaling), applying image effects interactively, or encoding an already taken picture with the JPEG encoder.
- Using the self path for display purposes like zooming and/or panning in an already taken picture.

5.3.5.4.1 DMA-read input buffer programming

Regardless for what use case the DMA-read feature is planned to be used, it always needs to know where the source YCbCr can be found in the system memory, and how this data is organized. The latter can be specified by programming the bits dma_input_format and dma_read_format in the MI_DMA_CTRL register. The MI can read YCbCr data in the chroma sub sampling formats 4:4:4, 4:2:2, 4:2:0 and 4:0:0, and in the data organizations shown in Figure 5.11 .

- MI_DMA_Y_PIC_START_AD
  The base address of the memory area to use for reading the Y image component, or the only image memory area to read in interleaved mode. In case the target system uses some kind of PMMU, the physical address must be specified, because virtual-to-physical translation is not possible for this kind of DMA-like transfer.
- MI_DMA_CB_PIC_START_AD
  The base address of the memory area to use for reading the Cb image component, or the combined chroma image components in semi planar mode.
- MI_DMA_CR_PIC_START_AD
  The base address of the memory area to use for reading the Cr image component. If semi-planar or interleaved mode is used, this value is ignored.
- MI_DMA_Y_PIC_WIDTH
  The image width of the Y component in pixel.
- MI_DMA_Y_LLENGTH
  Line length of the input buffer in pixel, also known as "line stride". This value becomes important if the MI shall read only a certain section of a much larger picture. In this case, the MI is not allowed to read consecutive lines back-to-back, but must increment the read address by a certain amount before the start of every line.
  MI_DMA_Y_LLENGTH must be set to the width of the whole image then. If the complete picture located in memory is to be read, MI_DMA_Y_LLENGTH should be set to the same value as MI_DMA_Y_PIC_WIDTH.

- MI_DMA_Y_PIC_SIZE
  Image size of the Y component in pixel. Set this value to the product of MI_DMA_Y_LLENGTH and the height of the image to be read.

One may have noticed that the DMA-read buffer configuration values regarding the image matrix are only needed to be specified for the Y-component, and no corresponding registers seem to exist for the chrominance buffers. The reason is that the MI calculates the corresponding values for the Cb and Cr buffer internally with help of the DMA-read input chroma sub sampling format given in the dma_input_format bits of the MI_DMA_CTRL register. The purpose of the DMA-read buffer configuration values are illustrated in Figure 5.16 .

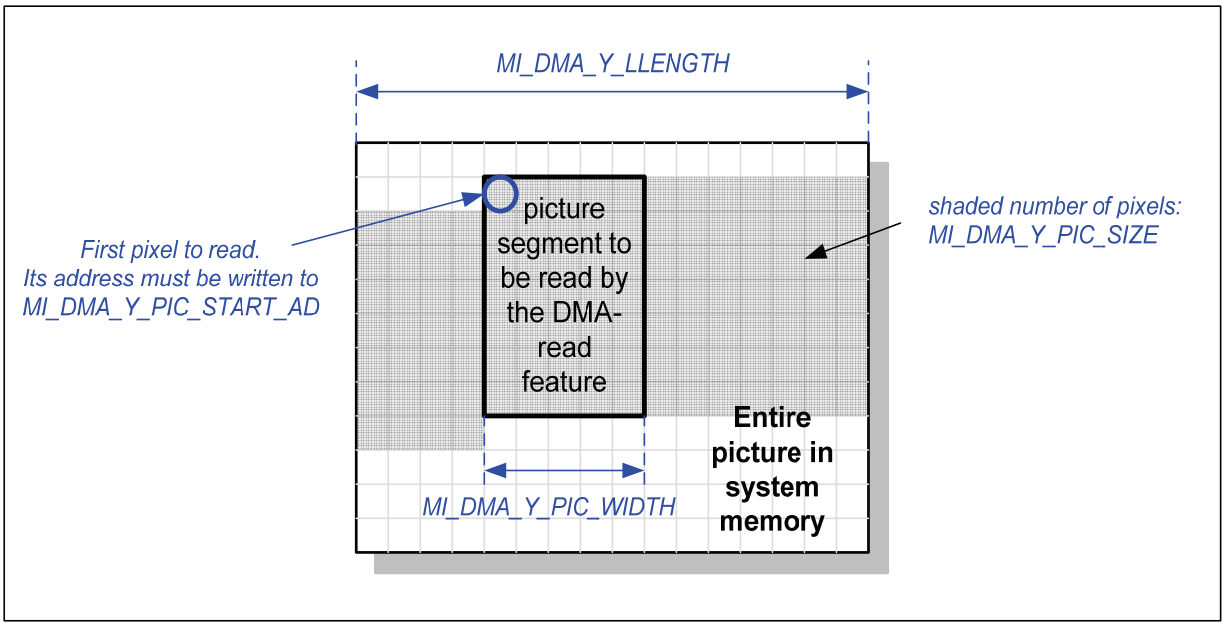


Figure 5.16 reading a picture segment from system memory

Similar to the configurations used for writing into buffers discussed in chapter 5.3.5.1 , there's also a possibility to swap the byte order for every 32-bit read access of the DMA-read feature. This can be activated by setting the dma_byte_swap bit in the MI_DMA_CTRL register. the effect of this bit is identical to that of the byte_swap bit in the MI_CTRL register shown in Figure 5.10 .

5.3.5.4.2 Triggering the actual read operation

5.3.5.4.2.1 One frame per DMA-read cycle

After the DMA-read buffer is setup correctly, and the destination of the read data is configured using the data path multiplexers (see register VI_DPCL for details), the actual reading can be triggered by setting the bit dma_start in the MI_DMA_START register. The status of the DMA-read sequence can be retrieved at any time by observing the bit dma_active of the MI_DMA_STATUS register, which is set while DMA is active.
Upon completion of reading the configured picture, the dma_ready interrupt is generated (see register MI_RIS for details). Note that the occurrence of this interrupt does only mean that the last byte of the frame has been read. Depending on the destination configured for the DMA-read-originated data, processing may still be in progress.

5.3.5.4.2.2 Mixing DMA-read data with live video

For the use-case of mixing live video data from the image sensor with data from the DMA-read block with help of the Superimpose module, it would be necessary to trigger the DMA-read operation for every frame separately. To simplify this, the bit dma_continous_en in the MI_DMA_CTRL register can be set, in which case the same frame is read back over and over again without software interaction. Setting of the dma_start bit is only necessary for the first frame to be read. To stop continuous mode, reset the dma_continous_en bit (takes effect after the next frame end) or execute a soft reset.

5.3.5.4.2.3 Frame slice processing with DMA-read

In some cases it may be convenient to perform DMA-read action without having the whole input frame in system memory. Usually, re-using a small memory block which is being filled in a round-robin fashion and process the image on a slice-by-slice base is done when there is only limited system memory space available. In general, this use-case does not seem to be

a big problem for the DMA-read part of the MI, because the starting address and the length of the data to be read can be re-configured as desired by the software before triggering the actual read action.
But the problem with this scenario is introduced by the following modules of TcISP's processing chain, because they usually react on the "frame end" condition by resetting their internal states. When DMA-read-originated data is being processed, the MI generates a "frame end" condition along with the last pixel read in every single read action, so an attempt to split off a frame into several DMA-read cycles will end up in multiple "frame end" conditions sent through the processing chain, which will in turn confuse all the other following modules.
To enable splitting up the processing of a single frame into several DMA-read cycles, the bit dma_frame_end_disable in the MI_DMA_CTRL register can be set for all but the last DMA-read cycle. This prevents the MI from sending the "frame end" condition down the processing chain. The dma_ready interrupt is raised as usual, but the dma_frame_end interrupt will not be generated until dma_frame_end_disable has been reset again. The dma_active bit in the MI_DMA_STATUS register can be used in addition to monitor the completion of the intermediate DMA-read cycles.
For the last DMA-read cycle to terminate the frame, the dma_frame_end_disable bit should be cleared again. This way, only a single correct "frame end" condition is sent down the processing chain and all the interrupts will be triggered as usual.
So, the procedure is:
- Load slice data into defined memory region
- start DMA transfer
- wait for dma_ready IRQ
- load next slice data into the same memory region
- start again the DMA transfer

For the last slice the dma_frame_end_disable bit has to be reset before the DMA transfer is started.

5.3.6 JPEG Encoder Programming

The JPEG encoder is responsible for still image compression. It operates widely independent from the system controller. This one only has to configure the encoder and to start the encoding process. So the basic flow is:

- Configuring
- JPEG header generation
- Image data encoding

To inform the system controller about finished header generation or encoding, events are triggered that can throw an interrupt allowing fully interrupt controlled operation of the software.

5.3.6.1 JPEG Encoder Reset and Configuration

The software reset of the JPEG encoder is identical to the system hardware reset and clears all configuration registers and the Huffman tables stored in encoder internal register files. So after releasing the JPEG encoder software reset about 400 clock cycles must be waited until the Huffman tables are cleared by the encoder hardware before re-programming these tables. Additionally all configuration registers have to be re-written.

5.3.6.2 Encoder Setup

Prior to any encoding the JPEG encoder has to be setup:

- Programming quantization tables
- Programming DC and AC symbol tables for Huffman variable length coding
- Setup image size
- Define YCbCr to YUV scaling
- Setup JPEG encode and header mode
- Enable status interrupt

The JPEG encoder has to do quantization of the DCT-transformed Y, U, and V color components. Four quantization tables with 64 entries of 8 bit exist. For each color component one table out of this four can be selected for processing. In general it is sufficient to use two tables, one for Y and the other for U/V quantization. So a second table-set for Y and U/V is available for switching between quality settings. The association between tables and the color components is defined in JPE_TQ_Y_SELECT, JPE_TQ_U_SELECT, and JPE_TQ_V_SELECT. Per default table 0 is used for luminance and table 1 for chrominance data quantization.
Huffman code tables are split into separate tables for coding the DC and AC values resulting from the DCT-transformation. Two sets of AC and DC tables exist, so per default table set 0 (DC0, AC0) is used for Y and set 1 (DC1, AC1) for U/V symbol generation. To change this association JPE_DC_TABLE_SELECT and JPE_AC_TABLE_SELECT have to be changed.

Switching to another symbol table must be done by fully re-programming the tables.
As a common programming interface is used to access all internal tables (quantization and Huffman tables) the table to be programmed must be set using the JPE_TABLE_ID configuration register. Quantization tables always have a fixed length of 64 bytes. Huffman tables may have different sizes, so their length in bytes must be defined using JPE_TDC0/1_LEN registers for DC0 and DC1 and JPE_TAC0/1_LEN registers for AC0 and AC1 tables.
Programming is done by simply writing the table contents to JPE_TABLE_DATA. Data are expected in 16 bit words in little endian format.
The following code example shows the programming of the tables:

```
#define TABLE_DATA_ADR   ((int*)(MARVIB_BASE + 0x1838))
#define TABLE_ID_ADR     ((int*)(TCISP_BASE + 0x1840))
#define TAC0_LEN_ADR     ((int*)(TCISP_BASE + 0x1844))
#define TAC1_LEN_ADR     ((int*)(TCISP_BASE + 0x1848))
#define TDC0_LEN_ADR     ((int*)(TCISP_BASE + 0x184C))
#define TDC1_LEN_ADR     ((int*)(TCISP_BASE + 0x1850))

// luma quantization table 75% quality setting
const int YQTable75PerCent[] = {
   0x06, 0x08, 0x07, 0x06, 0x05, 0x06, 0x07, 0x08, 0x07, 0x07, 0x09, 0x09,
   0x0a, 0x08, 0x14, 0x0c, 0x0c, 0x0d, 0x0b, 0x0b, 0x19, 0x0c, 0x13, 0x12,
   0x14, 0x0f, 0x1a, 0x1d, 0x1e, 0x1f, 0x1a, 0x1d, 0x1c, 0x1c, 0x24, 0x20,
   0x27, 0x2e, 0x22, 0x20, 0x23, 0x2c, 0x1c, 0x1c, 0x37, 0x28, 0x2c, 0x29,
   0x31, 0x30, 0x34, 0x34, 0x1f, 0x34, 0x39, 0x27, 0x38, 0x3d, 0x3c, 0x32,
   0x33, 0x2e, 0x32, 0x34
};

// chroma quantization table 75% quality setting
const int UVQTable75PerCent[] = {
   0x09, 0x09, 0x0c, 0x09, 0x0c, 0x0b, 0x0d, 0x18, 0x18, 0x0d, 0x21, 0x32,
   0x21, 0x1c, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32,
   0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32,
   0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32,
   0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32,
   0x32, 0x32, 0x32, 0x32
};

// DC luma table according to ISO/IEC 10918-1 annex K
const int DCLumaTableAnnexK[] = {
   0x01, 0x00, 0x01, 0x05, 0x01, 0x01, 0x01, 0x01, 0x00, 0x01, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x03, 0x02, 0x05, 0x04, 0x07, 0x06,
   0x09, 0x08, 0x0b, 0x0a
};

// DC chroma table according to ISO/IEC 10918-1 annex K
const int DCChromaTableAnnexK[] = {
   0x03, 0x00, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x01, 0x00, 0x01,
   0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x03, 0x02, 0x05, 0x04, 0x07, 0x06,
   0x09, 0x08, 0x0b, 0x0a
};

// AC luma table according to ISO/IEC 10918-1 annex K
const int ACLumaTableAnnexK[] = {
   0x02, 0x00, 0x03, 0x01, 0x02, 0x03, 0x03, 0x04, 0x05, 0x05, 0x04, 0x04,
   0x00, 0x00, 0x7d, 0x01, 0x02, 0x01, 0x00, 0x03, 0x11, 0x04, 0x12, 0x05,
   0x31, 0x21, 0x06, 0x41, 0x51, 0x13, 0x07, 0x61, 0x71, 0x22, 0x32, 0x14,
   0x91, 0x81, 0x08, 0xa1, 0x42, 0x23, 0xc1, 0xb1, 0x52, 0x15, 0xf0, 0xd1,
   0x33, 0x24, 0x72, 0x62, 0x09, 0x82, 0x16, 0x0a, 0x18, 0x17, 0x1a, 0x19,
   0x26, 0x25, 0x28, 0x27, 0x2a, 0x29, 0x35, 0x34, 0x37, 0x36, 0x39, 0x38,
   0x43, 0x3a, 0x45, 0x44, 0x47, 0x46, 0x49, 0x48, 0x53, 0x4a, 0x55, 0x54,
   0x57, 0x56, 0x59, 0x58, 0x63, 0x5a, 0x65, 0x64, 0x67, 0x66, 0x69, 0x68,
   0x73, 0x6a, 0x75, 0x74, 0x77, 0x76, 0x79, 0x78, 0x83, 0x7a, 0x85, 0x84,
   0x87, 0x86, 0x89, 0x88, 0x92, 0x8a, 0x94, 0x93, 0x96, 0x95, 0x98, 0x97,
   0x9a, 0x99, 0xa3, 0xa2, 0xa5, 0xa4, 0xa7, 0xa6, 0xa9, 0xa8, 0xb2, 0xaa,
   0xb4, 0xb3, 0xb6, 0xb5, 0xb8, 0xb7, 0xba, 0xb9, 0xc3, 0xc2, 0xc5, 0xc4,
   0xc7, 0xc6, 0xc9, 0xc8, 0xd2, 0xca, 0xd4, 0xd3, 0xd6, 0xd5, 0xd8, 0xd7,
   0xda, 0xd9, 0xe2, 0xe1, 0xe4, 0xe3, 0xe6, 0xe5, 0xe8, 0xe7, 0xea, 0xe9,
   0xf2, 0xf1, 0xf4, 0xf3, 0xf6, 0xf5, 0xf8, 0xf7, 0xfa, 0xf9
```

```
};

// AC Chroma table according to ISO/IEC 10918-1 annex K
const int ACChromaTableAnnexK[] = {
   0x02, 0x00, 0x02, 0x01, 0x04, 0x04, 0x04, 0x03, 0x05, 0x07, 0x04, 0x04,
   0x01, 0x00, 0x77, 0x02, 0x01, 0x00, 0x03, 0x02, 0x04, 0x11, 0x21, 0x05,
   0x06, 0x31, 0x41, 0x12, 0x07, 0x51, 0x71, 0x61, 0x22, 0x13, 0x81, 0x32,
   0x14, 0x08, 0x91, 0x42, 0xb1, 0xa1, 0x09, 0xc1, 0x33, 0x23, 0xf0, 0x52,
   0x62, 0x15, 0xd1, 0x72, 0x16, 0x0a, 0x34, 0x24, 0x25, 0xe1, 0x17, 0xf1,
   0x19, 0x18, 0x26, 0x1a, 0x28, 0x27, 0x2a, 0x29, 0x36, 0x35, 0x38, 0x37,
   0x3a, 0x39, 0x44, 0x43, 0x46, 0x45, 0x48, 0x47, 0x4a, 0x49, 0x54, 0x53,
   0x56, 0x55, 0x58, 0x57, 0x5a, 0x59, 0x64, 0x63, 0x66, 0x65, 0x68, 0x67,
   0x6a, 0x69, 0x74, 0x73, 0x76, 0x75, 0x78, 0x77, 0x7a, 0x79, 0x83, 0x82,
   0x85, 0x84, 0x87, 0x86, 0x89, 0x88, 0x92, 0x8a, 0x94, 0x93, 0x96, 0x95,
   0x98, 0x97, 0x9a, 0x99, 0xa3, 0xa2, 0xa5, 0xa4, 0xa7, 0xa6, 0xa9, 0xa8,
   0xb2, 0xaa, 0xb4, 0xb3, 0xb6, 0xb5, 0xb8, 0xb7, 0xba, 0xb9, 0xc3, 0xc2,
   0xc5, 0xc4, 0xc7, 0xc6, 0xc9, 0xc8, 0xd2, 0xca, 0xd4, 0xd3, 0xd6, 0xd5,
   0xd8, 0xd7, 0xda, 0xd9, 0xe3, 0xe2, 0xe5, 0xe4, 0xe7, 0xe6, 0xe9, 0xe8,
   0xf2, 0xea, 0xf4, 0xf3, 0xf6, 0xf5, 0xf8, 0xf7, 0xfa, 0xf9
};

int i;

// configuration register pointer declaration
int *TableDataP = TABLE_DATA_ADR;
int *TableIdP     = TABLE_ID_ADR;
int *TAC0LenP     = TAC0_LEN_ADR;
int *TAC1LenP     = TAC1_LEN_ADR;
int *TDC0LenP     = TDC0_LEN_ADR;
int *TDC1LenP     = TDC1_LEN_ADR;

// Y q-table 0 programming
TableIdP* = 0;
for (i = 0; i < 64/2; i++) {
   TableDataP* = (YQTable75PerCent[i*2+1] << 8) +
                 YQTable75PerCent[i*2];
}

// U/V q-table 0 programming
TableIdP* = 1;
for (i = 0; i < 64/2; i++) {
   TableDataP* = (UVQTable75PerCent[i*2+1] << 8) +
                 UVQTable75PerCent[i*2];
}

// Y DC-table 0 programming
TableIdP* = 4;
TDC0LenP* = 28;
for (i = 0; i < (28/2); i++) {
   TableDataP* = (DCLumaTableAnnexK[i*2+1] << 8) +
                 DCLumaTableAnnexK[i*2];
}

// U/V DC-table 1 programming
TableIdP* = 6;
TDC0LenP* = 28;
for (i = 0; i < (28/2); i++) {
   TableDataP* = (DCChromaTableAnnexK[i*2+1] << 8) +
                 DCChromaTableAnnexK[i*2];
}

// Y AC-table 0 programming
TableIdP* = 5;
TDC0LenP* = 178;
for (i = 0; i < (178/2); i++) {
   TableDataP* = (ACLumaTableAnnexK[i*2+1] << 8) +
                 ACLumaTableAnnexK[i*2];
}
```

```
// U/V AC-table 1 programming
TableIdP* = 7;
TDC0LenP* = 178;
for (i = 0; i < (178/2); i++) {
   TableDataP* = (ACChromaTableAnnexK[i*2+1] << 8) +
                 ACChromaTableAnnexK[i*2];
}
```

Note: Do not forget to re-program all AC and DC tables after system reset as well as after module software reset because after any reset the internal RAM is filled with 0xFF which is an illegal symbol. This filling takes approximately 400 clock cycles. So do not start any table programming during the first 400 clock cycles after reset is de-asserted.
In general there is no need to program other AC/DC tables than the ones proposed in the ISO/IEC 10918-1 annex K. Quantization defines the compression factor, so re-programming has to be done if more than two sets of quantization values for several compression levels exist.
Before JPEG encoding the size of the image to be encoded must be programmed using JPE_ENC_HSIZE and JPE_ENC_VSIZE. This setting must exactly match the incoming image, otherwise the error event r2b_img_size_err is thrown.
TCISP does all internal image processing in ITU_R BT.601 YC_bC_r color space defining a luminance range of 16..235 and a chrominance range of 16..240. JPEG compression is defined for YUV color space using the full range of 0..255. So the JPEG encoder supports scaling of incoming image data to full range, but this can be enabled or disabled in JPE_Y_SCALE_EN and JPE_CBCR_SCALE_EN.
Furthermore the image format for encoding has to be defined. As the JPEG encoder only supports YC_bC_r/YUV 4:2:2 the content of JPE_PIC_FORMAT must be kept at its default mode “1”.
Now the header mode has to be defined in JPE_HEADER_MODE. The stream generator is able to either write no header into system memory, or a full JFIF 1.02 compliant header is generated and written into system memory using the TCISP memory interface.

Table 5.7 JPEG Header Modes

header_mode	Description
3	reserved value
2	JFIF 1.02 header
1	reserved value
0	no header

Status interrupts must be enabled to get notified that processing steps (header generation and image encoding) have been finished.

5.3.6.3 JPEG Header Generation

Prior to the encoding process the JPEG encoder has to generate a JFIF 1.02 compliant header. To this header the actually encoded data are appended later on.
Header generation is started by setting *gen_header* in JPE_GEN_HEADER to “1”. Reading from this registers always returns “0”. For signaling that the header has been sent out the status event “header generation finished” (*gen_header_done*) is asserted.

5.3.6.4 JPEG Encoding

After image size and format as well as header generation the encoding process itself can be started. This is done by setting *encode* in JPE_ENCODE to “1”. This register is write only, so reading will give back a “0”.
If started the JPEG encoder waits for the cfg_upd signal coming from the ISP to start encoding the next image. If immediate start of the encoder is desired, then the bit *jp_init* in JPE_INIT register needs to be set. After the encoding is done, it disables its input ports. During encode the bit *busy* in JPE_ENCODER_BUSY is asserted.
The end of the encoding process is signaled by setting the “JPEG encoding done” event *encode_done* in the JPEG interrupt status register.
If consecutive frames shall be encoded (Motion JPEG), then for each new frame the JPEG encoder needs to be started again by setting

- the header generation bit gen_header in JPE_GEN_HEADER
- the encode bit in JPE_ENCODE
- the jp_init bit in JPE_INIT

All this must be done during the vertical blanking period.
As an alternative for motion JPEG encoding there are two control bits in register JPE_ENCODE which define, if the JPEG encoder shall be started in continuous mode or single shot mode. By setting these bits to ‘01’ or ‘11’ the encoder continues to encode without the need of setting the encode bit again.

The JPEG data is written to memory at the location defined by the Y FIFO address register. If the JPEG data ends at frame end with a byte count which does not correspond to a 32 bit boundary, then the MI fills the last 32 bit word and starts the next write command with a 32 bit aligned write position. The MI_BYTE_CNT register counts the real JPEG data only without any fill bytes.

6 CAMBUS Configuration

CAMBUS HCLK MASK Register (HCLKMASK) 0xB0230000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														HCLKMASK	

Field	Name	RW	Reset	Description
31-2	-	-	-	Reserved
1-0	HCLKMASK	R/W	0	Enable signals of HCLK clock gating logic AHB HCLKs, supplied to each block, are controlled by clock enable signal. 0: enable clock 1: disable clock The bit position indicates each sub-block which is controlled by Bit 1: ISP Bit 0: CIF

CAMBUS SW RESET0(SWRESET) 0xB0230004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														SWRESET	

Field	Name	RW	Reset	Description
31-2	-	-	-	Reserved
1-0	SWRESET	R/W	0	Enable signals of SWRESET logic AHB SWRESETs, supplied to each block, are controlled by SWRESET enable signal. 0 : disable SWRESET 1 : enable SWRESET The bit position indicates each sub-block which is controlled by Bit 1: ISP Bit 0: CIF

CAMBUS CLKO SEL 0xB0230008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															SEL

Field	Name	RW	Reset	Description
31-1	-	-	-	Reserved
0	SEL	R/W	0	Camera Module Clock Selection 0: CIF_ICLK 1: ISP CCLK

ISP RC CTRL 0xB0230020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															vc0_en

Field	Name	RW	Reset	Description
31-1	-	-	-	Reserved
0	vc0_en	R/W	0	ISP prefetch buffer enable

ISP RC A0 MIN 0xB0230024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
isp_vc0_addr_min [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
isp_vc0_addr_min [15:00]															

Field	Name	RW	Reset	Description
31-0	isp_vc0_addr_min	R/W	0	Start address of ISP prefetch address

ISP RC A0 MAX 0xB0230028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
isp_vc0_addr_max [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
isp_vc0_addr_max [15:00]															

Field	Name	RW	Reset	Description
31-0	isp_vc0_addr_max	R/W	0	End address of ISP prefetch address

CIF RC CTRL 0xB023002C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-															vc0_en

Field	Name	RW	Reset	Description
31-1	-	-	-	Reserved
0	vc0_en	R/W	0	CIF prefetch buffer enable

CIF RC A0 MIN 0xB0230030

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cif_ vc0_addr_min [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cif_ vc0_addr_min [15:00]															

Field	Name	RW	Reset	Description
31-0	cif_ vc0_addr_min	R/W	0	Start address of CIF prefetch address

CIF RC A0 MAX 0xB0230034

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cif_ vc0_addr_max [31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
cif_ vc0_addr_max [15:00]															

Field	Name	RW	Reset	Description
31-0	cif_ vc0_addr_max	R/W	0	End address of CIF prefetch address