MT6797 LTE-A Smartphone Application Processor Functional Specification for Development Board

Version: 1.0
Release date: 2016-07-19

Specifications are subject to change without notice.

Document Revision History

Revision	Date	Author	Description
1.0	2016-07-19	Clavin Peng	First release

Table of Contents

List of Figures

List of Tables

Preface

Acronyms for register types

R/W	For both read and write access
RO	Read only
RC	Read only. After the register bank is read, every bit that is HIGH(1) will be cleared to LOW(0) automatically.
WO	Write only
W1S	Write only. When data bits are written to the register bank, every bit that is HIGH(1) will cause the corresponding bit to be set to 1. Data bits that are LOW(0) have no effects on the corresponding bit.
W1C	Write only. When data bits are written to the register bank, every bit that is HIGH(1) will cause the corresponding bit to be cleared to 0. Data bits that are LOW(0) have no effects on the corresponding bit.

1 System Overview

The MT6797 device (see Figure 1-1), with integrated Bluetooth, FM, WLAN and GPS modules, is a highly integrated baseband platform incorporating both modem and application processing subsystems to enable LTE smart phone applications. The chip integrates dual ARM® Cortex-A72 operating up to 2.3GHz, quad ARM® Cortex-A53 operating up to 1.85GHz, quad ARM® Cortex-A53 up to 1.4GHz, an ARM® Cortex-R4 MCU and powerful multi-standard video codec. In addition, an extensive set of interfaces and connectivity peripherals are included to interface to cameras, touch-screen displays and MMC/SD cards.

The application processor, a Tri-cluster having Multi-core ARM® Cortex-A72 and ARM® Cortex-A53 MPCore™ equipped with NEON engine offers processing power necessary to support the latest OpenOS along with its demanding applications such as web browsing, email, GPS navigation and games.
All are viewed on a high resolution touch screen display with graphics enhanced by the 2D and 3D graphics acceleration.

The multi-standard video accelerator and an advanced audio subsystem are also integrated to provide advanced multimedia applications and services such as streaming audio and video, a multitude of decoders and encoders.

An ARM® Cortex-R4, DSP, and 2G and 3G coprocessors combined provide a powerful modem subsystem capable of supporting LTE Cat 6, Category 24 HSDPA downlink and Category 7 HSUPA uplink data rates, Category 14 TD-HSDPA downlink and Category 6 TD-HSUPA uplink, as well as Class 12 GPRS, EDGE.

MT6797 also embodies wireless communication device, including WLAN, Bluetooth and GPS. With four advanced radio technologies integrated into one single chip, MT6797 provides the best and most convenient connectivity solution in the industry.

The enhanced overall quality is achieved for simultaneous voice, data and audio/video transmission on mobile phones and Media Tablets. The small footprint with low-power consumption greatly reduces the PCB layout resource.

1.1 Highlighted Features Integrated in MT6797

- Dual-core ARM® Cortex-A72 MPCore™ operating at 2.3GHz
- Quad-core ARM® Cortex-A53 MPCore™ operating at 1.85GHz
- Quad-core ARM® Cortex-A53 MPCore™ operating at 1.4GHz
- LPDDR3 up to 4GB, 2 ch, 933MHz
- LTE Cat 6 (300Mps)
- Embedded connectivity system including WLAN/BT/FM/GPS
- Resolution up to WQHD (2,560*1,440)
- OpenGL ES 3.1 3D graphic accelerator
- ISP supports 25MP@30fps.
- HEVC 2160p@30fps decoder
- VP9 2160p@30fps decoder
- H.264 2160p@30fps encoder
- Speech codec (FR, HR, EFR, AMR FR, AMR HR and Wide-Band AMR)

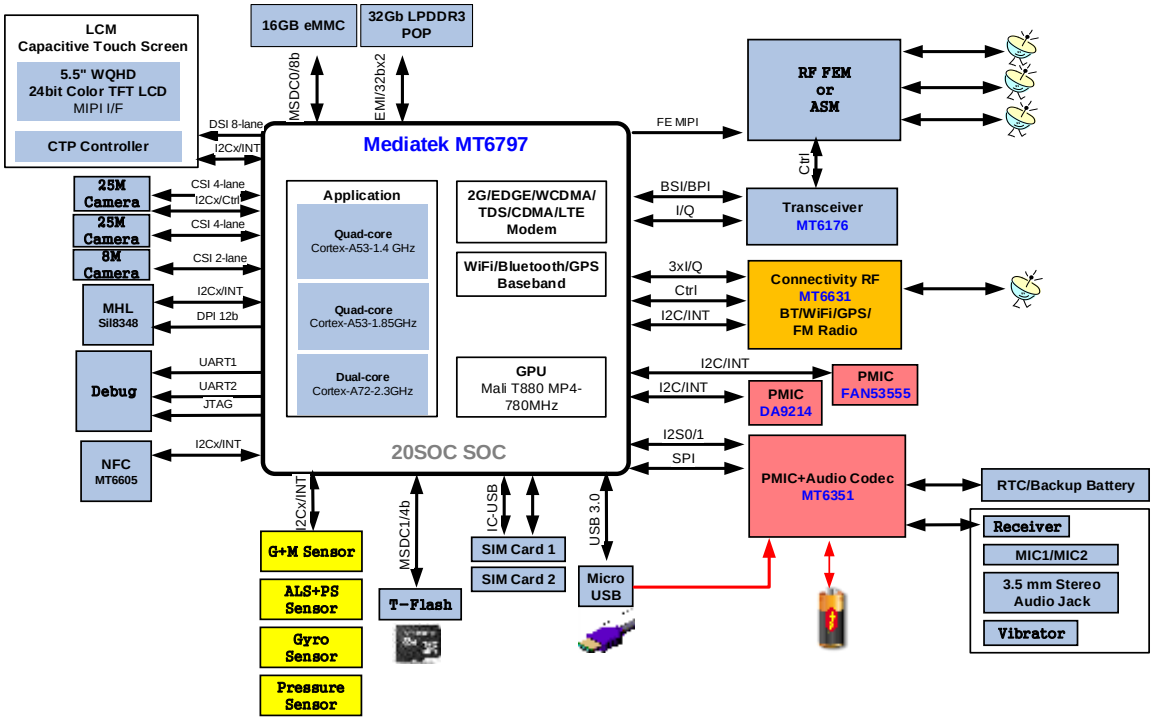


Figure 1-1. High-level MT6797 functional block diagram

1.2 Platform Features

- **General**
  - Smartphone, two MCU subsystems architecture
  - eMMC boot support
  - Supports LPDDR-3

- **AP MCU subsystem**
  - Dual-core ARM® 2.3GHz Cortex-A72 MPCore™ with 32KB L1 I-cache, 32KB L1 D-cache and 1MB unified L2 cache
  - Quad-core ARM® 1.85GHz Cortex-A53 MPCore™ with 32KB L1 I-cache, 32KB L1 D-cache and 512KB unified L2 cache
  - Quad-core ARM® 1.4GHz Cortex-A53 MPCore™ with 32KB L1 I-cache, 32KB L1 D-cache and 512KB unified L2 cache
  - NEON multimedia processing engine with SIMDv2 / VFPv4 ISA support
  - DVFS technology with adaptive operating voltage from 0.7V to 1.2V

- **MD MCU subsystem**
  - Two ARM® Cortex-R4 processors with max. 800 MHz operation frequency
  - 64KB I-cache, 64KB D-cache
  - 512KB TCM (tightly-coupled memory)
  - Coresonic DSP for running LTE modem tasks
  - FD216 DSP for running modem/voice tasks, with max. 312MHz operation frequency
  - High-performance AXI and AHB bus
  - General DMA engine and dedicated DMA channels for peripheral data transfer
  - Watchdog timer for system error recovery
  - Power management for clock gating control

- **MD external interfaces**
  - Dual SIM/USIM interface
  - Interface pins with RF and radio-related peripherals (antenna tuner, PA, etc.)

- **Security**
  - ARM® TrustZone® Security

- **External memory interface**
  - LPDDR3 up to 4GB
  - Dual channel with 32-bit data bus width
  - Memory clock up to 933 MHz
  - Self-refresh/partial self-refresh mode
  - Low-power operation
  - Programmable slew rate for memory controller's IO pads
  - Dual rank memory device
  - Advanced bandwidth arbitration control

- **Peripherals**
  - USB3.0 SS support
  - USB2.0 host mode
  - IC-USB device mode
  - eMMC5.1
  - 4 UART for debugging and applications
  - 6 SPI master for external device
  - 8 I2C to control peripheral devices, e.g. CMOS image sensor, LCM or FM receiver module
  - Max. 3 PWM channels (depending on system configuration/IO usage)
  - I2S for connection with optional external hi-end audio codec
  - GPIOs
  - 1 set of memory card controllers supporting SD/SDHC/MS/MSPRO/MMC and SDIO2.0/3.0 protocols

- **Operating conditions**
  - Core voltage: 0.9V
  - I/O voltage: 1.8V/2.8V/3.3V
  - Memory: 1.2V
  - LCM interface: 1.8V
  - Clock source: 26MHz, 32.768kHz

- **Package**
  - Type: POP
  - 14mm*14mm
  - Height: Max. 1.4mm
  - Ball count: 950 balls
  - Ball pitch: 0.4mm

1.3 Modem Features

- **LTE**
  - FDD/TDD Up to 300Mbps downlink, 50Mbps uplink
  - Downlink carrier aggregation (CA) ability; 1.4 to 20MHz RF bandwidth per component carrier (CC) and up to 2 CCs
  - 8*2 downlink SU-MIMO per component carrier
  - Downlink MU-MIMO per component carrier
  - Supports feICIC
  - Supports MBMS
  - Uplink CoMP ability

- **3G UMTS FDD supported features**
  - 3G modem supports most main features in 3GPP Release 7 and Release 8
  - CPC (DTX in CELL_DCH, UL DRX DL DRX), HS-SCCH-less, HS-DSCH
  - Dual cell operation
  - MAC-ehs
  - 2 DRX (receiver diversity) schemes in URA_PCH and CELL_PCH
  - Uplink Cat. 7 (16QAM), throughput up to 11.5Mbps
  - Downlink Cat. 24 (64QAM, dual-cell HSDPA), throughput up to 42.2Mbps
  - Fast dormancy
  - ETWS
  - Network selection enhancements

- **TD-SCDMA**
  - CDMA/HSDPA/HSUPA baseband
  - TD-SCDMA Bands 34, 39 & 40 and Quad band GSM/EDGE
  - Circuit-switched voice and data; packet-switched data
  - 384/384Kbps class in UL/DL for TD-SCDMA
  - TD-HSDPA: 2.8Mbps DL (Cat.14)
  - TD-HSUPA: 2.2Mbps UL (Cat.6)
  - F8/F9 ciphering/integrity protection

- **Radio interface and baseband front-end**
  - High dynamic range delta-sigma ADC converts the downlink analog I and Q signals to digital baseband.
  - 10-bit D/A converter for Automatic Power Control (APC)
  - Programmable radio RX filter with adaptive gain control
  - Dedicated RX filter for FB acquisition
  - Baseband Parallel Interface (BPI) with programmable driving strength
  - Supports multi-band

- **GSM modem and voice CODEC**
  - Dial tone generation
  - Noise reduction
  - Echo suppression
  - Advanced side-tone oscillation reduction
  - Digital side-tone generator with programmable gain
  - 2 programmable acoustic compensation filters
  - GSM quad vocoders for adaptive multi-rate (AMR), enhanced full rate (EFR), full rate (FR) and half rate (HR)
  - GSM channel coding, equalization and A5/1, A5/2 and A5/3 ciphering
  - GPRS GEA1, GEA2 and GEA3 ciphering
  - Programmable GSM/GPRS/EDGE modem
  - Packet switched data with CS1/CS2/CS3/CS4 coding schemes
  - GSM circuit switch data
  - GPRS/EDGE Class 12
  - Supports SAIC (Single Antenna Interference Cancellation) technology
  - VAMOS (Voice services over Adaptive Multi-user channels on One Slot) technology in R9 spec

- **CDMA2000 modem interfaces**
  - Supports CDMA2000 1xRTT (releases 0) and CDMA2000 HRPD/1xEV-DO Revision 0 and A
  - Supports maximum 1x data rates of 153.6kbps for forward and reverse links and DO data rates of 3.1Mbps for forward link and 1.8Mbps for reverse link
  - Hybrid operation between 1x and HRPD
  - Simultaneous Hybrid Dual Receiver (SHDR) support
  - Supports 1x Diversity
  - Supports SRLTE

1.4 Connectivity Features

MT6797 includes four wireless connectivity functions:

- WLAN
- Bluetooth
- GPS
- FM Receiver

The RF parts of those four blocks are placed on chip MT6631. With four advanced radio technologies integrated on one chip, MT6797/MT6631 is the best and most convenient connectivity solution in the industry, implementing advanced and sophisticated Radio Coexistence algorithms and hardware mechanisms. It supports single antenna sharing among 2.4GHz Bluetooth, 2.4GHz/5GHz WLAN and 1.575GHz for GPS. The enhanced overall quality is achieved for simultaneous voice, data and audio/video transmission on mobile phones and Media Tablets. The small footprint with low-power consumption greatly reduces PCB layout resource.

- **Supports integrated Wi-Fi/Bluetooth/GPS**
  - Single antenna for Bluetooth and WLAN/GPS/Bluetooth
  - Supports single tri-band antenna for WLAN (2.4GHz and 5GHz), Bluetooth, ANT+ and GNSS
  - Self calibration
  - Single TCXO and TMS for GPS, BT and WLAN
  - Best-in-class current consumption performance
  - Intelligent BT/WLAN coexistence scheme that goes beyond PTA signaling (e.g. transmit window and duration that take into account protocol exchange sequence, frequency, etc.)
- **Wi-Fi**
  - Dual-band (2.4/5GHz) single stream 802.11 a/b/g/n/ac MAC/BB/RF SoC, 20/40/80MHz bandwidth, MCS0~9 (256-QAM)
  - 802.11 d/e/h/i/j/k/r/v compliant
  - Security: WFA WPA/WPA2 personal, AES-CCMP, WPI-SMS4, GCMP, WPS2.0, WAPI (hardware)
  - QoS: WFA WMM, WMM PS
  - 802.11n optional features: LDPC, STBC, A-MPDU, Blk-Ack, RIFS, MCS Feedback, 20/40MHz coexistence (PCO), unscheduled PSMP
  - Supports 802.11w protected managed frames
  - Supports 802.11ac LDPC TX/RX, STBC TX/RX, 4T1R beamformee, MU-MIMO RX, WoWLAN
  - Supports MediaTek proprietary low power Green AP mode for portable hotspot operation
  - Auto rate control for optimizing the signal range and performance
  - Supports Wi-Fi Direct (WFA P-2-P standard) and Wi-Fi Miracast (Wi-Fi Display)
  - Supports Wi-Fi HotSpot 2.0
  - Integrated 2.4GHz PA with max. 23dBm CCK output power and 5GHz PA with max. 18.5dBm OFDM 54Mbps output power
  - RX sensitivity at 11n HT20 MCS7 mode and -62dBm 5GHz RX sensitivity at 11ac VHT80 MCS9 mode
  - Supports 32 multicast address filters and TCP/UDP/IP checksum offload
  - Per packet TX power control
- **Bluetooth**
  - Bluetooth specification v2.1+EDR
  - Bluetooth specification 3.0+HS compliance

- Bluetooth v4.1+HS compliant
- Supports BT5.0
- PIP RX only, public indoor position (Direction Finding) (HW+FW+SW)
- AOD RX (for product) , AOA TX(test only)
- Integrated PA with 12dBm (class 1) transmit power
- Typical RX sensitivity with companion chip modem: GFSK -94dBm, DQPSK -95dBm, 8-DPSK -89dBm, BLE -96dBm
- Best-in-class BT/Wi-Fi coexistence performance
- Up to 4 piconets simultaneously with background inquiry/page scan
- Supports BT legacy, BLE and ANT+ scatternet
- Packet Loss Concealment (PLC) function for better voice quality
- Low-power scan function to reduce power consumption in scan modes
- Supports Wideband speech (16KHz sampling rate)
- SBC encode include mono and stereo
- SBC decode only support mono
- mSBC support in controller
- Supports secure connection with AES128 and ECC256
- Supports LTE coexistence enhanced features: Clock nudge and generalized interlace scan
- Supports FM over BT A2DP

ANT/ANT+

- The wireless protocol standard for sport and fitness monitors
- Supports different profiles for various applications: sport & fitness, health & Wellness, recreational activity, transportation, and information management etc.

GPS

- Supports GPS/Glonass/Beidou/Galileo/QZSS tri-band reception concurrently
  - GPS/Galileo only (GPS only)
  - GPS/Galileo - GLONASS (G+G)
  - GPS/Beidou (G+B)
- Supports SBAS (Satellite-Based Augmentation Systems): WAAS/MSAS/EGNOS/GAGAN
- Best-in-class sensitivity performance
  - -165 dBm tracking sensitivity
  - -163 dBm hot start sensitivity
  - -148 dBm cold start sensitivity
  - -151 dBm warm start sensitivity
- AGPS sensitivity is 8dB design margin over 3GPP
- Full A-GPS capability (E911/SUPL/EPO/HotStill)
- Active interference cancellation for up to 12 in-band tones
- Supports both TCXO and TMS (Thermister Crystal) clock source
- 5Hz update rate

FM

- 65-108MHz with 50kHz step
- RDS/RBDS
- Digital stereo demodulator
- Simplified digital audio interface (I2S)
- Stereo noise reduction
- Audio sensitivity 2dBμVemf (SINAD=26dB)
- Audio SINAD 60dB
- Anti-jamming
- Integrated short antenna

WBT IPD

- Integrated matching network, balance band-pass filter, GPS-WBT diplexer
- Fully integrated in one IPD die
- Single and dual antenna operation

- **GPS IPD**
  - Integrated high-pass type matching network and 5th-order ellipse low-pass filter
  - Fully integrated in one IPD die
  - Single and dual antenna operation

1.5 Multimedia Features

- **Display**
  - Portrait panel resolution up to WQHD (2,560*1,440)
  - MIPI DSI interface (8 data lanes)
  - MiraVision™ for picture quality enhancement
  - ClearMotion™ for DTV-class video quality
  - Embedded LCD gamma correction
  - True colors
  - 12 overlay layers with per-pixel alpha channel and gamma table
  - Spatial and temporal dithering
  - Side-by-side format output to stereo 3D panel in both portrait and landscape modes
  - Color enhancement
  - Adaptive contrast enhancement
  - Image/video/graphic sharpness enhancement
  - Dynamic backlight scaling
  - Wide gamut

- **Graphics**
  - OpenGL ES 3.1 3D graphic accelerator capable of processing 390M tri/sec and 3,120M pixel/sec @ 780MHz
  - OpenVG1.1 vector graphics accelerator

- **Image**
  - Integrated image signal processor supports 25MP@30fps
  - Electronic image stabilization
  - Video stabilization
  - Preference color adjustment
  - Noise reduction
  - Multiple frame noise reduction for image capture
  - Temporal noise reduction for video recording
  - Lens shading correction
  - Auto sensor defect pixel correction
  - Supports AE/AWB/AF
  - Edge enhancement (sharpness)
  - Face detection and visual tracking
  - Video face beautification
  - Zero shutter delay image capture
  - Captures full size image when recording video (up to 25M sensors)
  - 3 MIPI CSI-2 high-speed camera serial interfaces; two are 4 data lane, and one is 2 data lane
  - PIP (picture in picture), [13MP + 13MP]@30fps
  - Hardware JPEG encoder: Baseline encoding with 200M pixel/sec. Continuous shot with 26M pixel@ 7fps
  - Supports YUV422/YUV420 color format and EXIF/JFIF format

- **Video**
  - HEVC decoder 2160p@30fps
  - VP9 decoder 2160p@30fps
  - H.264 decoder: 2160p@30fps
  - Sorenson H.263/H.263 decoder: 1080p@60fps/40Mbps
  - MPEG-4 SP/ASP decoder: 1080p@60fps/40Mbps
  - DIVX4/DIVX5/DIVX6/DIVX HD/XVID decoder: 1080p@60fps/40Mbps
  - MPEG-4 encoder: Simple profile D1@30fps
  - H.263 encoder: Simple profile D1@30fps
  - H.264 encoder: High profile 2160p@30fps
  - HEVC encoder: Main profile 2160p@30fps

- **Audio**
  - Audio content sampling rates supported: 8kHz to 192kHz
  - Audio content sample formats supported: 8-bit/16-bit/24-bit/32-bit, Mono/Stereo
  - Interfaces supported: I2S, PCM

  - External CODEC I2S interface supports 16-bit/24-bit, Mono/Stereo, 8kHz to 192kHz.
  - 4-band IIR compensation filter to enhance loudspeaker responses
  - Proprietary audio post-processing technologies: BesLoudness(MB-DRC), BesSurround, Android built-in post processing
  - Audio encoding: AMR-NB, AMR-WB, AAC, OGG, ADPCM
  - Audio decoding: WAV, MP3, MP2, AAC, AMR-NB, AMR-WB, MIDI, Vorbis, APE, AAC-plus v1, AAC-plus v2, FLAC, WMA, ADPCM
  - Voice wakeup
  - Headphone ANC

- **Speech**
  - Speech codec (FR, HR, EFR, AMR FR, AMR HR and Wide-Band AMR)
  - CTM
  - Compensation filter and digital gain for both uplink and downlink paths
  - MagiNRdual (Dual-MIC noise cancellation)
  - MagiNR (Noise suppression)
  - MagiAEC (Echo cancellation & Echo suppression)
  - MagiConference (Echo cancellation and Dual-MIC second source cancellation for speaker phone)
  - MagiLoudness (maximizes loudness while controlling the maximum receiver output power; feed-forward receiver protection)
  - MagiClarity (enhances the voice clarity based on near end environment noise)
  - MagiTDNC (remove TDD noise at GSM Link)
  - Dual-MIC stereo sound recording w/o Wind Noise Rejection
  - Dual-MIC voice tracking

1.6 General Description

MediaTek's MT6797 is a highly integrated LTE System-on-Chip (SoC) which incorporates advanced features, e.g. LTE cat.6, Deca HMP core operating up to 2.3GHz, 3D graphics (OpenGL|ES 3.1), 25M camera ISP, 2 channels LPDDR3-1866 Mbps, WQHD display and 2160p video codec. MT6797 helps phone manufacturers build high-performance LTE smart phones with PC-like browser, 3D gaming and cinema class home entertainment experiences.

The World-leading Technology!

Based on MediaTek's world-leading mobile chip SoC architecture with advanced 20SOC process, MT6797 is the brand-new generation smart phone SoC integrating MediaTek LTE modem, Multi-core ARM® Cortex-A72 and Cortex-A53 MPCore™, 3D graphics and high-definition 2160p video decoder.

Rich in Features, High-value Product!

To enrich the camera features, MT6797 equips a 25M camera ISP with advanced features, e.g. auto focus, electrical stabilization, auto sensor defect pixel correction, continuous video AF, face detection, face beautify, burst shot, optical zoom, panorama view, picture in picture, video in video and video face beautification.

Incredible Browser Experience!

The powerful CPU architecture with NEON multimedia processing engine brings PC-like browser experiences while keeping low standby power. GPU supporting OpenGL|ES 3.1 also provides you with excellent multimedia experiences.

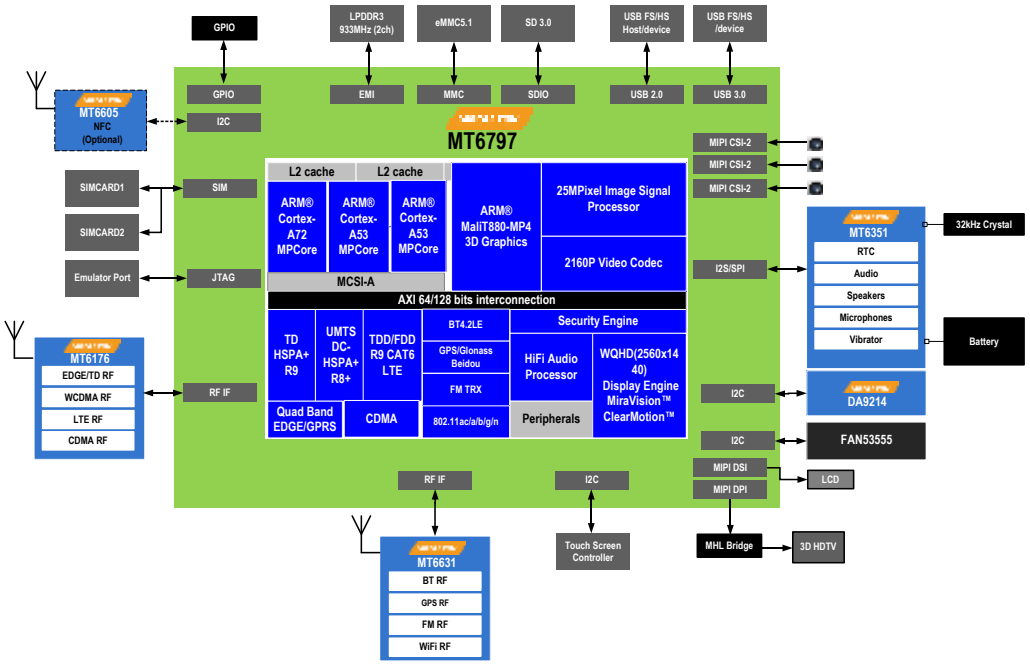


Figure 1-2. Block diagram of MT6797

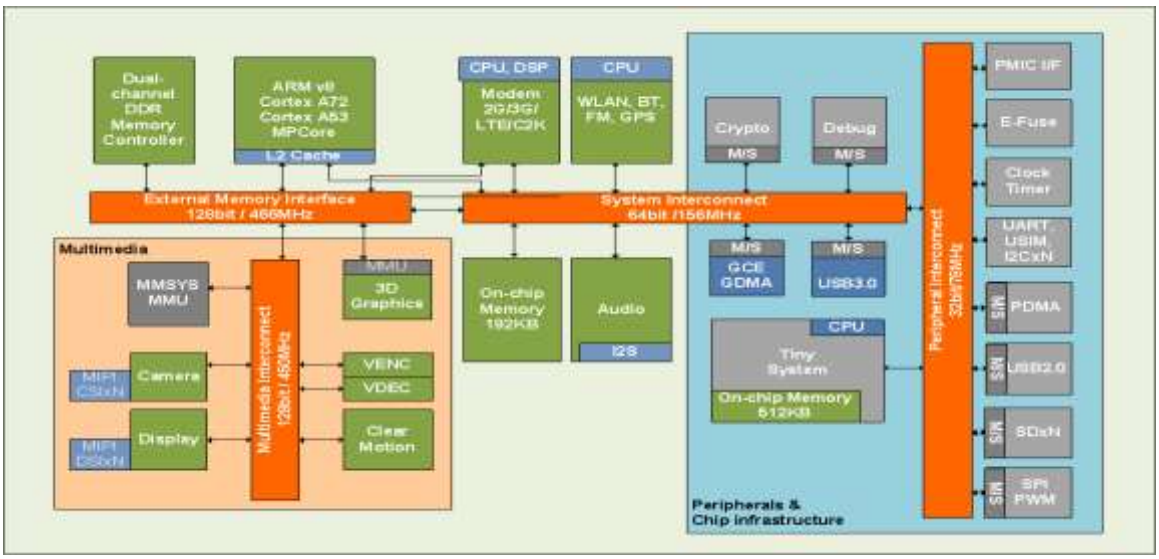


Figure 1-3. Bus structure of MT6797

2 Product Description

2.1 Pin Description

2.1.1 Ball Map View

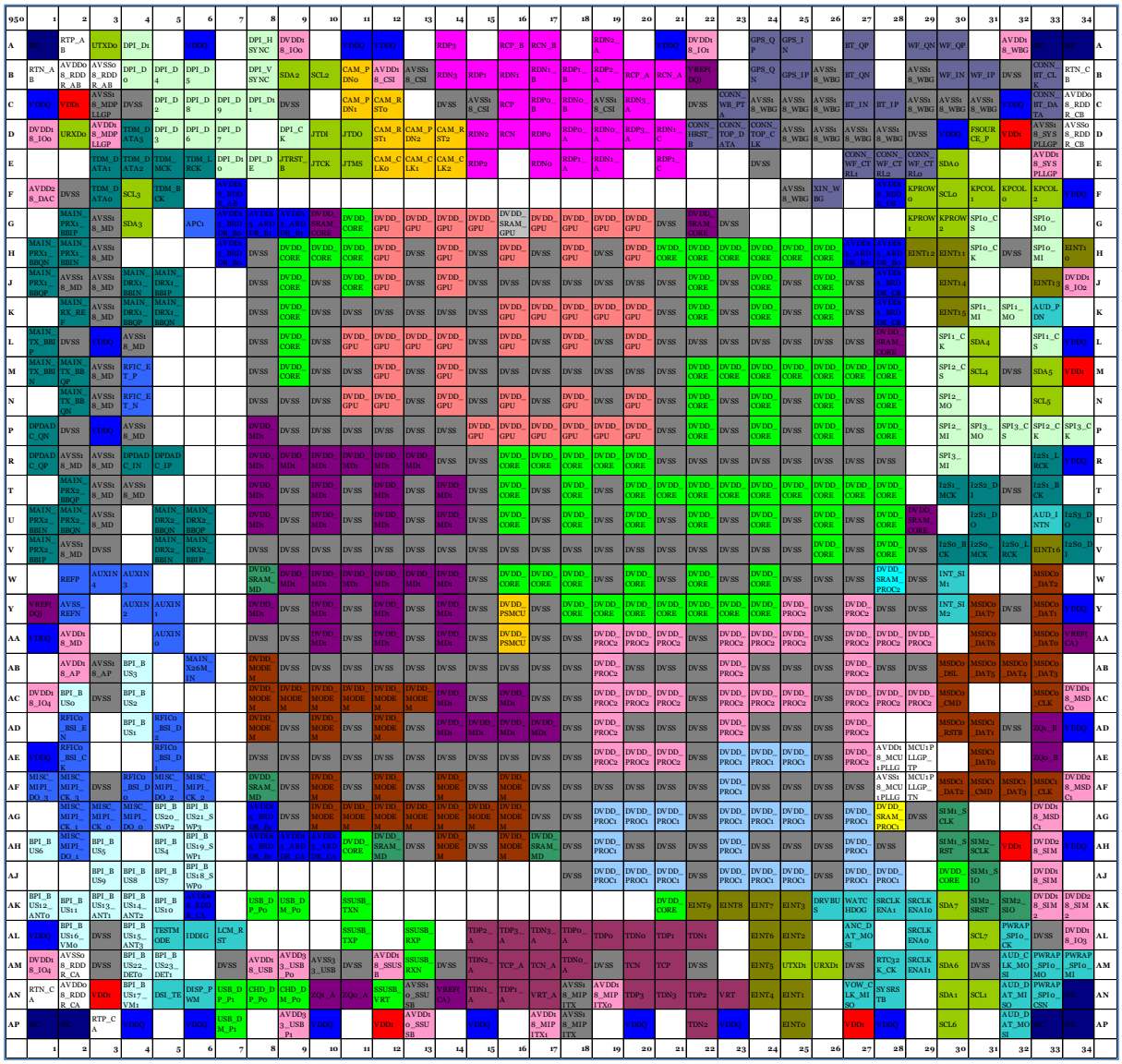


Figure 2-1. Ball map view

2.1.2 Pin Coordinate

Table 2-1. Pin coordinate

Ball Loc.	Ball name	Ball Loc.	Ball name	Ball Loc.	Ball name
A1	NC	M22	DVDD_CORE	AC18	DVSS
A2	RTP_AB	M23	DVDD_CORE	AC19	DVDD_PROC2
A3	UTXD0	M24	DVDD_CORE	AC20	DVDD_PROC2
A4	DPI_D1	M25	DVDD_CORE	AC21	DVDD_PROC2
A6	VDDQ	M26	DVDD_CORE	AC22	DVSS
A8	DPI_HSYNC	M27	DVDD_CORE	AC23	DVDD_PROC2
A9	DVDD18_IO0	M28	DVDD_CORE	AC24	DVDD_PROC2
A11	VDDQ	M30	SPI2_CS	AC25	DVDD_PROC2
A12	VDDQ	M31	SCL4	AC26	DVSS
A14	RDP3	M32	DVSS	AC27	DVDD_PROC2
A16	RCP_B	M33	SDA5	AC28	DVDD_PROC2
A17	RCN_B	M34	VDD1	AC29	DVDD_PROC2
A19	RDN2_A	N2	MAIN_TX_BBQN	AC30	MSDC0_CMD
A21	VDDQ	N3	AVSS18_MD	AC33	MSDC0_CLK
A22	DVDD18_IO1	N4	RFIC_ET_N	AC34	DVDD18_MSDC0
A24	GPS_QP	N8	DVSS	AD2	RFIC0_BSI_EN
A25	GPS_IN	N9	DVSS	AD4	BPI_BUS1
A27	BT_QP	N10	DVSS	AD5	RFIC0_BSI_D2
A29	WF_QN	N11	DVDD_GPU	AD8	DVDD_MODEM
A30	WF_QP	N12	DVDD_GPU	AD9	DVSS
A32	AVDD18_WBG	N13	DVSS	AD10	DVDD_MODEM
A33	NC	N14	DVDD_GPU	AD11	DVSS
A34	NC	N15	DVSS	AD12	DVDD_MODEM
B1	RTN_AB	N16	DVDD_GPU	AD13	DVSS
B2	AVDD08_RDDR_AB	N17	DVSS	AD14	DVDD_MD1
B3	AVSS08_RDDR_AB	N18	DVDD_GPU	AD15	DVDD_MD1
B4	DPI_D0	N19	DVSS	AD16	DVDD_MD1
B5	DPI_D4	N20	DVDD_GPU	AD17	DVDD_MD1
B6	DPI_D5	N21	DVSS	AD18	DVSS
B8	DPI_VSYNC	N22	DVDD_CORE	AD19	DVDD_PROC2
B9	SDA2	N23	DVSS	AD20	DVSS
B10	SCL2	N24	DVDD_CORE	AD21	DVSS
B11	CAM_PDN0	N25	DVSS	AD22	DVSS
B12	AVDD18_CSI	N26	DVDD_CORE	AD23	DVDD_PROC2
B13	AVSS18_CSI	N27	DVSS	AD24	DVSS
B14	RDN3	N28	DVDD_CORE	AD25	DVSS

Ball Loc.	Ball name	Ball Loc.	Ball name	Ball Loc.	Ball name
B15	RDP1	N30	SPI2_MO	AD26	DVSS
B16	RDN1	N33	SCL5	AD27	DVDD_PROC2
B17	RDN1_B	P1	DPDADC_QN	AD30	MSDC0_RSTB
B18	RDP1_B	P2	DVSS	AD31	MSDC1_DAT1
B19	RDP2_A	P3	VDDQ	AD32	DVSS
B20	RCP_A	P4	AVSS18_MD	AD33	ZQ1_B
B21	RCN_A	P8	DVDD_MD1	AD34	VDDQ
B22	VREF(DQ)	P9	DVSS	AE1	VDDQ
B24	GPS_QN	P10	DVSS	AE2	RFIC0_BSI_CK
B25	GPS_IP	P11	DVSS	AE5	RFIC0_BSI_D1
B26	AVSS18_WBG	P12	DVSS	AE8	DVSS
B27	BT_QN	P13	DVSS	AE9	DVSS
B29	AVSS18_WBG	P14	DVSS	AE10	DVSS
B30	WF_IN	P15	DVDD_GPU	AE11	DVSS
B31	WF_IP	P16	DVDD_GPU	AE12	DVSS
B32	DVSS	P17	DVDD_GPU	AE13	DVSS
B33	CONN_BT_CLK	P18	DVDD_GPU	AE14	DVSS
B34	RTN_CB	P19	DVDD_GPU	AE15	DVSS
C1	VDDQ	P20	DVDD_GPU	AE16	DVSS
C2	VDD1	P21	DVSS	AE17	DVSS
C3	AVSS18_MDPLLGP	P22	DVDD_CORE	AE18	DVSS
C4	DVSS	P23	DVSS	AE19	DVDD_PROC2
C5	DPI_D2	P24	DVDD_CORE	AE20	DVDD_PROC2
C6	DPI_D8	P25	DVSS	AE21	DVDD_PROC2
C7	DPI_D9	P26	DVDD_CORE	AE22	DVSS
C8	DPI_D11	P27	DVSS	AE23	DVDD_PROC1
C9	DVSS	P28	DVDD_CORE	AE24	DVDD_PROC1
C11	CAM_PDN1	P30	SPI2_MI	AE25	DVDD_PROC1
C12	CAM_RST0	P31	SPI3_MO	AE26	DVSS
C14	DVSS	P32	SPI3_CS	AE27	DVDD_PROC2
C15	AVSS18_CSI	P33	SPI2_CK	AE28	AVDD18_MCU1PLLGP
C16	RCP	P34	SPI3_CK	AE29	MCU1PLLGP_TP
C17	RDP0_B	R1	DPDADC_QP	AE31	MSDC1_DAT0
C18	RDN0_B	R2	AVSS18_MD	AE33	ZQ0_B
C19	AVSS18_CSI	R3	AVSS18_MD	AF1	MISC_MIPI_DO_3
C20	RDN3_A	R4	DPDADC_IN	AF2	MISC_MIPI_CK_3
C22	DVSS	R5	DPDADC_IP	AF3	DVSS
C23	CONN_WB_PTA	R8	DVDD_MD1	AF4	RFIC0_BSI_D0
C24	AVSS18_WBG	R9	DVDD_MD1	AF5	MISC_MIPI_DO_2

Ball Loc.	Ball name	Ball Loc.	Ball name	Ball Loc.	Ball name
C25	AVSS18_WBG	R10	DVDD_MD1	AF6	MISC_MIPI_CK_2
C26	AVSS18_WBG	R11	DVDD_MD1	AF8	DVDD_SRAM_MD
C27	BT_IN	R12	DVDD_MD1	AF9	DVSS
C28	BT_IP	R13	DVDD_MD1	AF10	DVDD_MODEM
C29	AVSS18_WBG	R14	DVSS	AF11	DVSS
C30	AVSS18_WBG	R15	DVSS	AF12	DVDD_MODEM
C31	AVSS18_WBG	R16	DVDD_CORE	AF13	DVSS
C32	VDDQ	R17	DVDD_CORE	AF14	DVDD_MODEM
C33	CONN_BT_DATA	R18	DVDD_CORE	AF15	DVSS
C34	AVDD08_RDDR_CB	R19	DVDD_CORE	AF16	DVDD_MODEM
D1	DVDD18_IO0	R20	DVDD_CORE	AF17	DVSS
D2	URXD0	R21	DVSS	AF18	DVSS
D3	AVDD18_MDPLLGP	R22	DVSS	AF19	DVSS
D4	TDM_DATA3	R23	DVSS	AF20	DVSS
D5	DPI_D3	R24	DVSS	AF21	DVSS
D6	DPI_D6	R25	DVSS	AF22	DVSS
D7	DPI_D7	R26	DVSS	AF23	DVDD_PROC1
D9	DPI_CK	R27	DVSS	AF24	DVSS
D10	JTDI	R28	DVSS	AF25	DVSS
D11	JTDO	R30	SPI3_MI	AF26	DVSS
D12	CAM_RST1	R33	I2S1_LRCK	AF27	DVSS
D13	CAM_PDN2	R34	VDDQ	AF28	AVSS18_MCU1PLLGP
D14	CAM_RST2	T2	MAIN_PRX2_BBQP	AF29	MCU1PLLGP_TN
D15	RDN2	T3	AVSS18_MD	AF30	MSDC1_DAT2
D16	RCN	T4	AVSS18_MD	AF31	MSDC1_CMD
D17	RDP0	T8	DVDD_MD1	AF32	MSDC1_DAT3
D18	RDP0_A	T9	DVSS	AF33	MSDC1_CLK
D19	RDN0_A	T10	DVDD_MD1	AF34	DVDD28_MSDC1
D20	RDP3_A	T11	DVSS	AG2	MISC_MIPI_CK_1
D21	RDN1_C	T12	DVDD_MD1	AG3	MISC_MIPI_CK_0
D22	CONN_HRST_B	T13	DVSS	AG4	MISC_MIPI_DO_0
D23	CONN_TOP_DATA	T14	DVDD_MD1	AG5	BPI_BUS20_SWP2
D24	CONN_TOP_CLK	T15	DVSS	AG6	BPI_BUS21_SWP3
D25	AVSS18_WBG	T16	DVDD_CORE	AG8	AVDD15_BRDDR_B13
D26	AVSS18_WBG	T17	DVSS	AG9	DVSS
D27	AVSS18_WBG	T18	DVDD_CORE	AG10	DVDD_MODEM
D28	AVSS18_WBG	T19	DVSS	AG11	DVDD_MODEM
D29	DVSS	T20	DVDD_CORE	AG12	DVDD_MODEM
D30	VDDQ	T21	DVDD_CORE	AG13	DVDD_MODEM

Ball Loc.	Ball name	Ball Loc.	Ball name	Ball Loc.	Ball name
D31	FSOURCE_P	T22	DVDD_CORE	AG14	DVDD_MODEM
D32	VDD1	T23	DVDD_CORE	AG15	DVDD_MODEM
D33	AVSS18_SYSPLLGP	T24	DVDD_CORE	AG16	DVDD_MODEM
D34	AVSS08_RDDR_CB	T25	DVDD_CORE	AG17	DVSS
E3	TDM_DATA1	T26	DVDD_CORE	AG18	DVSS
E4	TDM_DATA2	T27	DVDD_CORE	AG19	DVDD_PROC1
E5	TDM_MCK	T28	DVDD_CORE	AG20	DVDD_PROC1
E6	TDM_LRCK	T30	I2S1_MCK	AG21	DVDD_PROC1
E7	DPI_D10	T31	I2S2_DI	AG22	DVSS
E8	DPI_DE	T32	DVSS	AG23	DVDD_PROC1
E9	JTRST_B	T33	I2S1_BCK	AG24	DVDD_PROC1
E10	JTCK	U1	MAIN_PRX2_BBIN	AG25	DVDD_PROC1
E11	JTMS	U2	MAIN_PRX2_BBQN	AG26	DVSS
E12	CAM_CLK0	U3	AVSS18_MD	AG27	DVDD_PROC1
E13	CAM_CLK1	U5	MAIN_DRX2_BBQN	AG28	DVDD_SRAM_PROC1
E14	CAM_CLK2	U6	MAIN_DRX2_BBQP	AG29	DVSS
E15	RDP2	U8	DVDD_MD1	AG30	SIM1_SCLK
E17	RDN0	U9	DVSS	AG33	DVDD18_MSDC1
E18	RDP1_A	U10	DVDD_MD1	AH1	BPI_BUS6
E19	RDN1_A	U11	DVSS	AH2	MISC_MIPI_DO_1
E21	RDP1_C	U12	DVDD_MD1	AH3	BPI_BUS5
E24	DVSS	U13	DVSS	AH5	BPI_BUS4
E27	CONN_WF_CTRL1	U14	DVDD_MD1	AH6	BPI_BUS19_SWP1
E28	CONN_WF_CTRL2	U15	DVSS	AH8	AVDD15_BRDDR_B13
E29	CONN_WF_CTRL0	U16	DVDD_CORE	AH9	AVDD15_ARDDR_CA
E30	SDA0	U17	DVSS	AH10	AVDD15_ARDDR_CA
E33	AVDD18_SYSPLLGP	U18	DVDD_CORE	AH11	DVDD_CORE
F1	AVDD28_DAC	U19	DVSS	AH12	DVDD_SRAM_MD
F2	DVSS	U20	DVDD_CORE	AH13	DVSS
F3	TDM_DATA0	U21	DVSS	AH14	DVDD_MODEM
F4	SCL3	U22	DVDD_CORE	AH15	DVSS
F5	TDM_BCK	U23	DVSS	AH16	DVDD_MODEM
F7	AVDD18_RDDR_AB	U24	DVDD_CORE	AH17	DVDD_SRAM_MD
F25	AVSS18_WBG	U25	DVSS	AH18	DVSS
F26	XIN_WBG	U26	DVDD_CORE	AH19	DVDD_PROC1
F28	AVDD18_RDDR_CB	U27	DVSS	AH20	DVSS
F29	KPROW0	U28	DVDD_CORE	AH21	DVSS
F30	SCL0	U29	DVDD_SRAM_CORE	AH22	DVSS
F31	KPCOL1	U31	I2S1_DO	AH23	DVDD_PROC1

Ball Loc.	Ball name	Ball Loc.	Ball name	Ball Loc.	Ball name
F32	KPCOL0	U33	AUD_INTN	AH24	DVSS
F33	KPCOL2	U34	I2S3_DO	AH25	DVSS
F34	VDDQ	V1	MAIN_PRX2_BBIP	AH26	DVSS
G2	MAIN_PRX1_BBIP	V2	AVSS18_MD	AH27	DVDD_PROC1
G3	AVSS18_MD	V3	DVSS	AH28	DVSS
G4	SDA3	V5	MAIN_DRX2_BBIN	AH30	SIM1_SRST
G6	APC1	V6	MAIN_DRX2_BBIP	AH31	SIM2_SCLK
G7	AVDD15_BRDDR_B02	V8	DVSS	AH32	VDD1
G8	AVDD15_ARDDR_B13	V9	DVSS	AH33	DVDD28_SIM
G9	AVDD15_ARDDR_B13	V10	DVSS	AH34	VDDQ
G10	DVDD_SRAM_CORE	V11	DVSS	AJ3	BPI_BUS9
G11	DVDD_CORE	V12	DVSS	AJ4	BPI_BUS8
G12	DVDD_GPU	V13	DVSS	AJ5	BPI_BUS7
G13	DVDD_GPU	V14	DVSS	AJ6	BPI_BUS18_SWP0
G14	DVDD_GPU	V15	DVSS	AJ18	DVSS
G15	DVDD_GPU	V16	DVSS	AJ19	DVDD_PROC1
G16	DVDD_SRAM_GPU	V17	DVSS	AJ20	DVDD_PROC1
G17	DVDD_GPU	V18	DVSS	AJ21	DVDD_PROC1
G18	DVDD_GPU	V19	DVSS	AJ22	DVSS
G19	DVDD_GPU	V20	DVSS	AJ23	DVDD_PROC1
G20	DVDD_GPU	V21	DVSS	AJ24	DVDD_PROC1
G21	DVSS	V22	DVSS	AJ25	DVDD_PROC1
G22	DVDD_SRAM_CORE	V23	DVSS	AJ26	DVSS
G23	DVSS	V24	DVSS	AJ27	DVDD_PROC1
G29	KPROW1	V25	DVSS	AJ28	DVDD_PROC1
G30	KPROW2	V26	DVDD_CORE	AJ30	DVDD_CORE
G31	SPI0_CS	V27	DVSS	AJ31	SIM1_SIO
G33	SPI0_MO	V28	DVDD_CORE	AJ33	DVDD18_SIM
H1	MAIN_PRX1_BBQN	V29	DVSS	AK1	BPI_BUS12_ANT0
H2	MAIN_PRX1_BBIN	V30	I2S0_BCK	AK2	BPI_BUS11
H3	AVSS18_MD	V31	I2S0_MCK	AK3	BPI_BUS13_ANT1
H7	AVDD15_BRDDR_B02	V32	I2S0_LRCK	AK4	BPI_BUS14_ANT2
H8	DVSS	V33	EINT16	AK5	BPI_BUS10
H9	DVDD_CORE	V34	I2S0_DI	AK6	AVDD18_RDDR_CA
H10	DVDD_CORE	W2	REFP	AK8	USB_DP_P0
H11	DVDD_CORE	W3	AUXIN4	AK9	USB_DM_P0
H12	DVDD_GPU	W4	AUXIN3	AK11	SSUSB_TXN
H13	DVSS	W8	DVDD_SRAM_MD	AK21	DVDD_CORE
H14	DVDD_GPU	W9	DVDD_MD1	AK22	EINT9

Ball Loc.	Ball name	Ball Loc.	Ball name	Ball Loc.	Ball name
H15	DVSS	W10	DVDD_MD1	AK23	EINT8
H16	DVDD_GPU	W11	DVDD_MD1	AK24	EINT7
H17	DVSS	W12	DVDD_MD1	AK25	EINT3
H18	DVDD_GPU	W13	DVDD_MD1	AK26	DRVBUS
H19	DVSS	W14	DVDD_MD1	AK27	WATCHDOG
H20	DVDD_GPU	W15	DVSS	AK28	SRCLKENA1
H21	DVDD_CORE	W16	DVDD_CORE	AK29	SRCLKENAI0
H22	DVDD_CORE	W17	DVDD_CORE	AK30	SDA7
H23	DVDD_CORE	W18	DVDD_CORE	AK31	SIM2_SRST
H24	DVDD_CORE	W19	DVSS	AK32	SIM2_SIO
H25	DVDD_CORE	W20	DVDD_CORE	AK33	DVDD18_SIM2
H26	DVDD_CORE	W21	DVSS	AK34	DVDD28_SIM2
H27	AVDD15_ARDDR_B02	W22	DVDD_CORE	AL1	VDDQ
H28	AVDD15_ARDDR_B02	W23	DVSS	AL2	BPI_BUS16_VM0
H29	EINT12	W24	DVDD_CORE	AL3	DVSS
H30	EINT11	W25	DVSS	AL4	BPI_BUS15_ANT3
H31	SPI0_CK	W26	DVSS	AL5	TESTMODE
H32	DVSS	W27	DVSS	AL6	IDDIG
H33	SPI0_MI	W28	DVDD_SRAM_PROC2	AL7	LCM_RST
H34	EINT10	W29	DVSS	AL11	SSUSB_TXP
J1	MAIN_PRX1_BBQP	W30	INT_SIM1	AL13	SSUSB_RXP
J2	AVSS18_MD	W33	MSDC0_DAT2	AL15	TDP2_A
J3	AVSS18_MD	Y1	VREF(DQ)	AL16	TDP3_A
J4	MAIN_DRX1_BBIN	Y2	AVSS_REFN	AL17	TDN3_A
J5	MAIN_DRX1_BBIP	Y4	AUXIN2	AL18	TDP0_A
J8	DVSS	Y5	AUXIN1	AL19	TDP0
J9	DVDD_CORE	Y8	DVDD_MD1	AL20	TDN0
J10	DVSS	Y9	DVSS	AL21	TDP1
J11	DVDD_CORE	Y10	DVDD_MD1	AL22	TDN1
J12	DVDD_GPU	Y11	DVSS	AL24	EINT6
J13	DVSS	Y12	DVDD_MD1	AL25	EINT2
J14	DVDD_GPU	Y13	DVSS	AL27	ANC_DAT_MOSI
J15	DVSS	Y14	DVDD_MD1	AL29	SRCLKENA0
J16	DVSS	Y15	DVSS	AL31	SCL7
J17	DVSS	Y16	DVDD_PSMCU	AL32	PWRAP_SPI0_CK
J18	DVSS	Y17	DVSS	AL33	DVSS
J19	DVSS	Y18	DVDD_CORE	AL34	DVDD18_IO3
J20	DVSS	Y19	DVDD_CORE	AM1	DVDD18_IO4
J21	DVSS	Y20	DVDD_CORE	AM2	AVSS08_RDDR_CA

Ball Loc.	Ball name	Ball Loc.	Ball name	Ball Loc.	Ball name
J22	DVDD_CORE	Y21	DVDD_CORE	AM3	DVSS
J23	DVSS	Y22	DVDD_CORE	AM4	BPI_BUS22_DET0
J24	DVDD_CORE	Y23	DVDD_CORE	AM5	BPI_BUS23_DET1
J25	DVSS	Y24	DVDD_CORE	AM7	DVSS
J26	DVDD_CORE	Y25	DVDD_PROC2	AM8	AVDD18_USB
J27	DVSS	Y26	DVSS	AM9	AVDD33_USB_P0
J28	AVDD15_BRDDR_CB	Y27	DVDD_PROC2	AM10	AVSS33_USB
J30	EINT14	Y28	DVSS	AM11	DVSS
J33	EINT13	Y29	DVSS	AM12	AVDD18_SSUSB
J34	DVDD18_IO2	Y30	INT_SIM2	AM13	SSUSB_RXN
K2	MAIN_RX_REF	Y31	MSDC0_DAT7	AM14	DVSS
K3	AVSS18_MD	Y32	DVSS	AM15	TDN2_A
K4	MAIN_DRX1_BBQP	Y33	MSDC0_DAT1	AM16	TCP_A
K5	MAIN_DRX1_BBQN	Y34	VDDQ	AM17	TCN_A
K8	DVSS	AA1	VDDQ	AM18	TDN0_A
K9	DVDD_CORE	AA2	AVDD18_MD	AM19	DVSS
K10	DVSS	AA5	AUXIN0	AM20	TCN
K11	DVSS	AA8	DVSS	AM21	TCP
K12	DVSS	AA9	DVSS	AM22	DVSS
K13	DVSS	AA10	DVDD_MD1	AM24	EINT5
K14	DVSS	AA11	DVSS	AM25	UTXD1
K15	DVSS	AA12	DVDD_MD1	AM26	URXD1
K16	DVDD_GPU	AA13	DVSS	AM27	DVSS
K17	DVDD_GPU	AA14	DVDD_MD1	AM28	RTC32K_CK
K18	DVDD_GPU	AA15	DVSS	AM29	SRCLKENAI1
K19	DVDD_GPU	AA16	DVDD_PSMCU	AM30	SDA6
K20	DVDD_GPU	AA17	DVSS	AM31	DVSS
K21	DVSS	AA18	DVSS	AM32	AUD_CLK_MOSI
K22	DVDD_CORE	AA19	DVDD_PROC2	AM33	PWRAP_SPI0_MO
K23	DVSS	AA20	DVDD_PROC2	AM34	PWRAP_SPI0_MI
K24	DVDD_CORE	AA21	DVDD_PROC2	AN1	RTN_CA
K25	DVSS	AA22	DVSS	AN2	AVDD08_RDDR_CA
K26	DVDD_CORE	AA23	DVDD_PROC2	AN3	VDD1
K27	DVSS	AA24	DVDD_PROC2	AN4	BPI_BUS17_VM1
K28	AVDD15_BRDDR_CB	AA25	DVDD_PROC2	AN5	DSI_TE
K30	EINT15	AA26	DVSS	AN6	DISP_PWM
K31	SPI1_MI	AA27	DVDD_PROC2	AN7	USB_DP_P1
K32	SPI1_MO	AA28	DVDD_PROC2	AN8	CHD_DP_P0
K33	AUD_PDN	AA29	DVDD_PROC2	AN9	CHD_DM_P0

Ball Loc.	Ball name	Ball Loc.	Ball name	Ball Loc.	Ball name
L1	MAIN_TX_BBIP	AA31	MSDC0_DAT6	AN10	ZQ1_A
L2	DVSS	AA33	MSDC0_DAT0	AN11	ZQ0_A
L3	VDDQ	AA34	VREF(CA)	AN12	SSUSB_VRT
L4	AVSS18_MD	AB2	AVDD18_AP	AN13	AVSS10_SSUSB
L8	DVSS	AB3	AVSS18_AP	AN14	VREF(CA)
L9	DVDD_CORE	AB4	BPI_BUS3	AN15	TDN1_A
L10	DVSS	AB6	MAIN_X26M_IN	AN16	TDP1_A
L11	DVDD_GPU	AB8	DVDD_MODEM	AN17	VRT_A
L12	DVDD_GPU	AB9	DVSS	AN18	AVSS18_MIPITX
L13	DVDD_GPU	AB10	DVSS	AN19	AVDD18_MIPITX0
L14	DVDD_GPU	AB11	DVSS	AN20	TDP3
L15	DVSS	AB12	DVSS	AN21	TDN3
L16	DVDD_GPU	AB13	DVSS	AN22	TDP2
L17	DVSS	AB14	DVSS	AN23	VRT
L18	DVDD_GPU	AB15	DVSS	AN24	EINT4
L19	DVSS	AB16	DVSS	AN25	EINT1
L20	DVDD_GPU	AB17	DVSS	AN27	VOW_CLK_MISO
L21	DVSS	AB18	DVSS	AN28	SYSRSTB
L22	DVSS	AB19	DVDD_PROC2	AN30	SDA1
L23	DVSS	AB20	DVSS	AN31	SCL1
L24	DVSS	AB21	DVSS	AN32	AUD_DAT_MISO
L25	DVSS	AB22	DVSS	AN33	PWRAP_SPI0_CSN
L26	DVSS	AB23	DVDD_PROC2	AN34	NC
L27	DVSS	AB24	DVSS	AP1	NC
L28	DVDD_SRAM_CORE	AB25	DVSS	AP2	NC
L30	SPI1_CK	AB26	DVSS	AP3	RTP_CA
L31	SDA4	AB27	DVDD_PROC2	AP4	VDDQ
L33	SPI1_CS	AB28	DVSS	AP6	VDDQ
L34	VDDQ	AB29	DVSS	AP7	USB_DM_P1
M1	MAIN_TX_BBIN	AB30	MSDC0_DSL	AP9	AVDD33_USB_P1
M2	MAIN_TX_BBQP	AB31	MSDC0_DAT5	AP10	VDDQ
M3	AVSS18_MD	AB32	MSDC0_DAT4	AP12	VDD1
M4	RFIC_ET_P	AB33	MSDC0_DAT3	AP13	AVDD10_SSUSB
M8	DVSS	AC1	DVDD18_IO4	AP15	VDDQ
M9	DVDD_CORE	AC2	BPI_BUS0	AP17	AVDD18_MIPITX1
M10	DVSS	AC3	DVSS	AP18	AVSS18_MIPITX
M11	DVSS	AC4	BPI_BUS2	AP20	VDDQ
M12	DVDD_GPU	AC8	DVDD_MODEM	AP22	TDN2
M13	DVSS	AC9	DVDD_MODEM	AP23	VDDQ

Ball Loc.	Ball name	Ball Loc.	Ball name	Ball Loc.	Ball name
M14	DVDD_GPU	AC10	DVDD_MODEM	AP25	EINT0
M15	DVSS	AC11	DVDD_MODEM	AP27	VDD1
M16	DVSS	AC12	DVDD_MODEM	AP28	VDDQ
M17	DVSS	AC13	DVDD_MODEM	AP30	SCL6
M18	DVSS	AC14	DVDD_MD1	AP32	AUD_DAT_MOSI
M19	DVSS	AC15	DVSS	AP33	NC
M20	DVSS	AC16	DVDD_MD1	AP34	NC
M21	DVSS	AC17	DVSS		

2.1.3 Detailed Pin Description

Table 2-2. Acronym for pin type

Abbreviation	Description
AI	Analog input
AO	Analog output
AIO	Analog bi-direction
DI	Digital input
DO	Digital output
DIO	Digital bi-direction
P	Power
G	Ground

Table 2-3. Detailed pin description

Pin name	Pin No.	Type	Description	Power domain
System				
SYSRSTB	AN28	DIO	System reset input	DVDD18_IO3
WATCHDOG	AK27	DO	Watchdog reset output	DVDD18_IO3
TESTMODE	AL5	DIO	Test mode	DVDD18_IO3
RTC32K_CK	AM28	DIO	RTC 32K input	DVDD18_IO3
SRCLKENA0	AL29	DIO		DVDD18_IO3
SRCLKENA1	AK28	DIO		DVDD18_IO3
SRCLKENAI0	AK29	DIO		DVDD18_IO3
SRCLKENAI1	AM29	DIO		DVDD18_IO3
PMIC				
PWRAP_SPI0_MO	AM33	DIO	PMIC SPI control interface	DVDD18_IO3
PWRAP_SPI0_MI	AM34	DIO	PMIC SPI control interface	DVDD18_IO3
PWRAP_SPI0_CSN	AN33	DIO	PMIC SPI control interface	DVDD18_IO3
PWRAP_SPI0_CK	AL32	DIO	PMIC SPI control interface	DVDD18_IO3
VOW_CLK_MISO	AN27	DIO	PMIC SPI control interface	DVDD18_IO3
AUD_CLK_MOSI	AM32	DIO	PMIC audio input interface	DVDD18_IO3
AUD_DAT_MISO	AN32	DIO	PMIC audio input interface	DVDD18_IO3
AUD_DAT_MOSI	AP32	DIO	PMIC audio input interface	DVDD18_IO3
ANC_DAT_MOSI	AL27	DIO	PMIC audio input interface	DVDD18_IO3
AUD_INTN	U33	DIO	PMIC audio input interface	DVDD18_IO2
AUD_PDN	K33	DIO	PMIC audio input interface	DVDD18_IO2
SIM				
SIM1_SCLK	AG30	DIO	SIM1 clock, PMIC interface	DVDD28_SIM1
SIM1_SIO	AJ31	DIO	SIM1 data, PMIC interface	DVDD28_SIM1
SIM1_SRST	AH30	DIO	SIM1 data, PMIC interface	DVDD28_SIM1
SIM2_SCLK	AH31	DIO	SIM2 clock, PMIC interface	DVDD28_SIM2
SIM2_SIO	AK32	DIO	SIM2 data, PMIC interface	DVDD28_SIM2
SIM2_SRST	AK31	DIO	SIM2 data, PMIC interface	DVDD28_SIM2
INT_SIM1	W30	DIO	SIM1 interrupt	DVDD18_IO2
INT_SIM2	Y30	DIO	SIM2 interrupt	DVDD18_IO2
LCD				
DSI_TE	AN5	DIO	Parallel display interface tearing effect	DVDD18_IO4
LCM_RST	AL7	DIO	Parallel display interface reset signal	DVDD18_IO4
DPI_HSYNC	A8	DIO	Parallel display interface HSYNC	DVDD18_IO0
DPI_VSYNC	B8	DIO	Parallel display interface VSYNC	DVDD18_IO0
DPI_CK	D9	DIO	Parallel display interface CLK	DVDD18_IO0
DPI_DE	E8	DIO	Parallel display interface DE	DVDD18_IO0
DPI_D11	C8	DIO	Data pin 11 for DPI parallel LCD interface	DVDD18_IO0
DPI_D10	E7	DIO	Data pin 10 for DPI parallel LCD interface	DVDD18_IO0

Pin name	Pin No.	Type	Description	Power domain
DPI_D9	C7	DIO	Data pin 9 for DPI parallel LCD interface	DVDD18_IO0
DPI_D8	C6	DIO	Data pin 8 for DPI parallel LCD interface	DVDD18_IO0
DPI_D7	D7	DIO	Data pin 7 for DPI parallel LCD interface	DVDD18_IO0
DPI_D6	D6	DIO	Data pin 6 for DPI parallel LCD interface	DVDD18_IO0
DPI_D5	B6	DIO	Data pin 5 for DPI parallel LCD interface	DVDD18_IO0
DPI_D4	B5	DIO	Data pin 4 for DPI parallel LCD interface	DVDD18_IO0
DPI_D3	D5	DIO	Data pin 3 for DPI parallel LCD interface	DVDD18_IO0
DPI_D2	C5	DIO	Data pin 2 for DPI parallel LCD interface	DVDD18_IO0
DPI_D1	A4	DIO	Data pin 1 for DPI parallel LCD interface	DVDD18_IO0
DPI_D0	B4	DIO	Data pin 0 for DPI parallel LCD interface	DVDD18_IO0
PWM				
DISP_PWM	AN6	DIO	Display PWM	DVDD18_IO4
Keypad interface				
KPCOL0	F32	DIO	Keypad column 0	DVDD18_IO2
KPCOL1	F31	DIO	Keypad column 1	DVDD18_IO2
KPCOL2	F33	DIO	Keypad column 2	DVDD18_IO2
KPROW0	F29	DIO	Keypad row 0	DVDD18_IO2
KPROW1	G29	DIO	Keypad row 1	DVDD18_IO2
KPROW2	G30	DIO	Keypad row 2	DVDD18_IO2
I2S Interface				
TDM_BCK	F5	DIO	TDM_BCK	DVDD18_IO0
TDM_DATA0	F3	DIO	TDM_DATA0	DVDD18_IO0
TDM_DATA1	E3	DIO	TDM_DATA1	DVDD18_IO0
TDM_DATA2	E4	DIO	TDM_DATA2	DVDD18_IO0
TDM_DATA3	D4	DIO	TDM_DATA3	DVDD18_IO0
TDM_LRCK	E6	DIO	TDM_LRCK	DVDD18_IO0
TDM_MCK	E5	DIO	TDM_MCK	DVDD18_IO0
I2S0_BCK	V30	DIO	I2S0_BCK	DVDD18_IO2
I2S0_DI	V34	DIO	I2S0_DI	DVDD18_IO2
I2S0_LRCK	V32	DIO	I2S0_LRCK	DVDD18_IO2
I2S0_MCK	V31	DIO	I2S0_MCK	DVDD18_IO2
I2S3_DO	U34	DIO	I2S3_DO	DVDD18_IO2
I2S1_BCK	T33	DIO	I2S1_BCK	DVDD18_IO2
I2S1_DO	U31	DIO	I2S1_DO	DVDD18_IO2
I2S1_LRCK	R33	DIO	I2S1_LRCK	DVDD18_IO2
I2S1_MCK	T30	DIO	I2S1_MCK	DVDD18_IO2
I2S2_DI	T31	DIO	I2S2_DI	DVDD18_IO2
SPI				
SPI0_CS	G31	DIO	SPI0 chip select	DVDD18_IO2
SPI0_MI	H33	DIO	SPI0 data in	DVDD18_IO2

Pin name	Pin No.	Type	Description	Power domain
SPI0_MO	G33	DIO	SPI0 data out	DVDD18_IO2
SPI0_CK	H31	DIO	SPI0 clock	DVDD18_IO2
SPI1_CS	L33	DIO	SPI1 chip select	DVDD18_IO2
SPI1_MI	K31	DIO	SPI1 data in	DVDD18_IO2
SPI1_MO	K32	DIO	SPI1 data out	DVDD18_IO2
SPI1_CK	L30	DIO	SPI1 clock	DVDD18_IO2
SPI2_CS	M30	DIO	SPI2 chip select	DVDD18_IO2
SPI2_MI	P30	DIO	SPI2 data in	DVDD18_IO2
SPI2_MO	N30	DIO	SPI2 data out	DVDD18_IO2
SPI2_CK	P33	DIO	SPI2 clock	DVDD18_IO2
SPI3_CS	P32	DIO	SPI3 chip select	DVDD18_IO2
SPI3_MI	R30	DIO	SPI3 data in	DVDD18_IO2
SPI3_MO	P31	DIO	SPI3 data out	DVDD18_IO2
SPI3_CK	P34	DIO	SPI3 clock	DVDD18_IO2
BPI				
BPI_BUS0	AC2	DIO	BPI_BUS0	DVDD18_IO4
BPI_BUS1	AD4	DIO	BPI_BUS1	DVDD18_IO4
BPI_BUS2	AC4	DIO	BPI_BUS2	DVDD18_IO4
BPI_BUS3	AB4	DIO	BPI_BUS3	DVDD18_IO4
BPI_BUS4	AH5	DIO	BPI_BUS4	DVDD18_IO4
BPI_BUS5	AH3	DIO	BPI_BUS5	DVDD18_IO4
BPI_BUS6	AH1	DIO	BPI_BUS6	DVDD18_IO4
BPI_BUS7	AJ5	DIO	BPI_BUS7	DVDD18_IO4
BPI_BUS8	AJ4	DIO	BPI_BUS8	DVDD18_IO4
BPI_BUS9	AJ3	DIO	BPI_BUS9	DVDD18_IO4
BPI_BUS10	AK5	DIO	BPI_BUS10	DVDD18_IO4
BPI_BUS11	AK2	DIO	BPI_BUS11	DVDD18_IO4
BPI_BUS12_ANT0	AK1	DIO	BPI_BUS12	DVDD18_IO4
BPI_BUS13_ANT1	AK3	DIO	BPI_BUS13	DVDD18_IO4
BPI_BUS14_ANT2	AK4	DIO	BPI_BUS14	DVDD18_IO4
BPI_BUS15_ANT3	AL4	DIO	BPI_BUS15	DVDD18_IO4
BPI_BUS16_VM0	AL2	DIO	BPI_BUS16	DVDD18_IO4
BPI_BUS17_VM1	AN4	DIO	BPI_BUS17	DVDD18_IO4
BPI_BUS18_SWP0	AJ6	DIO	BPI_BUS18	DVDD18_IO4
BPI_BUS19_SWP1	AH6	DIO	BPI_BUS19	DVDD18_IO4
BPI_BUS20_SWP2	AG5	DIO	BPI_BUS20	DVDD18_IO4
BPI_BUS21_SWP3	AG6	DIO	BPI_BUS21	DVDD18_IO4
BPI_BUS22_DET0	AM4	DIO	BPI_BUS22	DVDD18_IO4
BPI_BUS23_DET1	AM5	DIO	BPI_BUS23	DVDD18_IO4
BSI				

Pin name	Pin No.	Type	Description	Power domain
RFIC0_BSI_CK	AE2	DIO	RFIC0 BSI CLK	DVDD18_IO4
RFIC0_BSI_D0	AF4	DIO	RFIC0 BSI DATA0	DVDD18_IO4
RFIC0_BSI_D1	AE5	DIO	RFIC0 BSI DATA1	DVDD18_IO4
RFIC0_BSI_D2	AD5	DIO	RFIC0 BSI DATA2	DVDD18_IO4
RFIC0_BSI_EN	AD2	DIO	RFIC0 BSI CS	DVDD18_IO4
MISC_MIPI_CK_0	AG3	DIO	MISC_MIPI_CK_0	DVDD18_IO4
MISC_MIPI_DO_0	AG4	DIO	MISC_MIPI_DO_0	DVDD18_IO4
MISC_MIPI_CK_1	AG2	DIO	MISC_MIPI_CK_1	DVDD18_IO4
MISC_MIPI_DO_1	AH2	DIO	MISC_MIPI_DO_1	DVDD18_IO4
MISC_MIPI_CK_2	AF6	DIO	MISC_MIPI_CK_2	DVDD18_IO4
MISC_MIPI_DO_2	AF5	DIO	MISC_MIPI_DO_2	DVDD18_IO4
MISC_MIPI_CK_3	AF2	DIO	MISC_MIPI_CK_3	DVDD18_IO4
MISC_MIPI_DO_3	AF1	DIO	MISC_MIPI_DO_3	DVDD18_IO4
MSDC1				
MSDC1_CLK	AF33	DIO	MSDC1 clock output	DVDD28_MSDC1
MSDC1_CMD	AF31	DIO	MSDC1 command pin	DVDD28_MSDC1
MSDC1_DAT0	AE31	DIO	MSDC1 data0 pin	DVDD28_MSDC1
MSDC1_DAT1	AD31	DIO	MSDC1 data1 pin	DVDD28_MSDC1
MSDC1_DAT2	AF30	DIO	MSDC1 data2 pin	DVDD28_MSDC1
MSDC1_DAT3	AF32	DIO	MSDC1 data3 pin	DVDD28_MSDC1
MSDC0				
MSDC0_CLK	AC33	DIO	MSDC0 clock output	DVDD18_MSDC0
MSDC0_CMD	AC30	DIO	MSDC0 command pin	DVDD18_MSDC0
MSDC0_DAT0	AA33	DIO	MSDC0 data0 pin	DVDD18_MSDC0
MSDC0_DAT1	Y33	DIO	MSDC0 data1 pin	DVDD18_MSDC0
MSDC0_DAT2	W33	DIO	MSDC0 data2 pin	DVDD18_MSDC0
MSDC0_DAT3	AB33	DIO	MSDC0 data3 pin	DVDD18_MSDC0
MSDC0_DAT4	AB32	DIO	MSDC0 data4 pin	DVDD18_MSDC0
MSDC0_DAT5	AB31	DIO	MSDC0 data5 pin	DVDD18_MSDC0
MSDC0_DAT6	AA31	DIO	MSDC0 data6 pin	DVDD18_MSDC0
MSDC0_DAT7	Y31	DIO	MSDC0 data7 pin	DVDD18_MSDC0
MSDC0_DSL	AB30	DIO	MSDC0 DSL pin	DVDD18_MSDC0
MSDC0_RSTB	AD30	DIO	MSDC0 Reset pin	DVDD18_MSDC0
EMI				
VREF(CA)	AA34	AIO	Reserved. DRAM interface	-
VREF(DQ)	B22	AIO	Reserved. DRAM interface	-
RTN_AB	B1	AIO	Reserved. DRAM interface	-
RTN_CA	AN1	AIO	Reserved. DRAM interface	-
RTN_CB	B34	AIO	Reserved. DRAM interface	-
RTP_AB	A2	AIO	Reserved. DRAM interface	-

Pin name	Pin No.	Type	Description	Power domain
RTP_CA	AP3	AIO	Reserved. DRAM interface	-
ZQ0_A	AN11	AIO	ZQ0 of DRAM die A	-
ZQ0_B	AE33	AIO	ZQ0 of DRAM die B	-
ZQ1_A	AN10	AIO	ZQ1 of DRAM die A	-
ZQ1_B	AD33	AIO	ZQ0 of DRAM die B	-
CAM				
CAM_CLK0	E12	DIO	Master clock to 1st sensor	DVDD18_IO0
CAM_PDN0	B11	DIO	Power down to 1st sensor	DVDD18_IO0
CAM_RST0	C12	DIO	Reset control to 1st sensor	DVDD18_IO0
CAM_CLK1	E13	DIO	Master clock to 2nd sensor	DVDD18_IO0
CAM_PDN1	C11	DIO	Power down to 2nd sensor	DVDD18_IO0
CAM_RST1	D12	DIO	Reset control to 2nd sensor	DVDD18_IO0
CAM_CLK2	E14	DIO	Master clock to 3rd sensor	DVDD18_IO0
CAM_PDN2	D13	DIO	Power down to 3rd sensor	DVDD18_IO0
CAM_RST2	D14	DIO	Reset controlto 3rd sensor	DVDD18_IO0
I2C				
SCL0	F30	DIO	I2C0 clock	DVDD18_IO2
SDA0	E30	DIO	I2C0 data	DVDD18_IO2
SCL1	AN31	DIO	I2C1 clock	DVDD18_IO3
SDA1	AN30	DIO	I2C1 data	DVDD18_IO3
SCL2	B10	DIO	I2C2 clock	DVDD18_IO0
SDA2	B9	DIO	I2C2 data	DVDD18_IO0
SCL3	F4	DIO	I2C3 clock	DVDD18_IO0
SDA3	G4	DIO	I2C3 data	DVDD18_IO0
SCL4	M31	DIO	I2C4 clock	DVDD18_IO2
SDA4	L31	DIO	I2C4 data	DVDD18_IO2
SCL5	N33	DIO	I2C5 clock	DVDD18_IO2
SDA5	M33	DIO	I2C5 data	DVDD18_IO2
SCL6	AP30	DIO	I2C6 clock	DVDD18_IO3
SDA6	AM30	DIO	I2C6 data	DVDD18_IO3
SCL7	AL31	DIO	I2C7 clock	DVDD18_IO3
SDA7	AK30	DIO	I2C7 data	DVDD18_IO3
CONN				
CONN_BT_CLK	B33	DIO	Control for CONN_RF	DVDD18_IO1
CONN_BT_DATA	C33	DIO	Control for CONN_RF	DVDD18_IO1
CONN_HRST_B	D22	DIO	Control for CONN_RF	DVDD18_IO1
CONN_TOP_CLK	D24	DIO	Control for CONN_RF	DVDD18_IO1
CONN_TOP_DATA	D23	DIO	Control for CONN_RF	DVDD18_IO1
CONN_WB_PTA	C23	DIO	Control for CONN_RF	DVDD18_IO1
CONN_WF_CTRL0	E29	DIO	Control for CONN_RF	DVDD18_IO1

Pin name	Pin No.	Type	Description	Power domain
CONN_WF_CTRL1	E27	DIO	Control for CONN_RF	DVDD18_IO1
CONN_WF_CTRL2	E28	DIO	Control for CONN_RF	DVDD18_IO1
ABB				
MAIN_DRX1_BBIN	J4	AIO	MAIN_DRX1_BBIN	AVDD18_MD
MAIN_DRX1_BBIP	J5	AIO	MAIN_DRX1_BBIP	AVDD18_MD
MAIN_DRX1_BBQN	K5	AIO	MAIN_DRX1_BBQN	AVDD18_MD
MAIN_DRX1_BBQP	K4	AIO	MAIN_DRX1_BBQP	AVDD18_MD
MAIN_DRX2_BBIN	V5	AIO	MAIN_DRX2_BBIN	AVDD18_MD
MAIN_DRX2_BBIP	V6	AIO	MAIN_DRX2_BBIP	AVDD18_MD
MAIN_DRX2_BBQN	U5	AIO	MAIN_DRX2_BBQN	AVDD18_MD
MAIN_DRX2_BBQP	U6	AIO	MAIN_DRX2_BBQP	AVDD18_MD
MAIN_PRX1_BBIN	H2	AIO	MAIN_PRX1_BBIN	AVDD18_MD
MAIN_PRX1_BBIP	G2	AIO	MAIN_PRX1_BBIP	AVDD18_MD
MAIN_PRX1_BBQN	H1	AIO	MAIN_PRX1_BBQN	AVDD18_MD
MAIN_PRX1_BBQP	J1	AIO	MAIN_PRX1_BBQP	AVDD18_MD
MAIN_PRX2_BBIN	U1	AIO	MAIN_PRX2_BBIN	AVDD18_MD
MAIN_PRX2_BBIP	V1	AIO	MAIN_PRX2_BBIP	AVDD18_MD
MAIN_PRX2_BBQN	U2	AIO	MAIN_PRX2_BBQN	AVDD18_MD
MAIN_PRX2_BBQP	T2	AIO	MAIN_PRX2_BBQP	AVDD18_MD
MAIN_RX_REF	K2	AIO	MAIN_RX_REF	AVDD18_MD
MAIN_TX_BBIN	M1	AIO	MAIN_TX_BBIN	AVDD18_MD
MAIN_TX_BBIP	L1	AIO	MAIN_TX_BBIP	AVDD18_MD
MAIN_TX_BBQN	N2	AIO	MAIN_TX_BBQN	AVDD18_MD
MAIN_TX_BBQP	M2	AIO	MAIN_TX_BBQP	AVDD18_MD
MAIN_X26M_IN	AB6	AIO	MAIN_X26M_IN	AVDD18_MD
DPDADC_IN	R4	AIO	DPDADC_IN	AVDD18_MD
DPDADC_IP	R5	AIO	DPDADC_IP	AVDD18_MD
DPDADC_QN	P1	AIO	DPDADC_QN	AVDD18_MD
DPDADC_QP	R1	AIO	DPDADC_QP	AVDD18_MD
AUXIN0	AA5	AIO	AUXADC external input channel 0	AVDD18_MD
AUXIN1	Y5	AIO	AUXADC external input channel 1	AVDD18_MD
AUXIN2	Y4	AIO	AUXADC external input channel 2	AVDD18_MD
AUXIN3	W4	AIO	AUXADC external input channel 3	AVDD18_MD
AUXIN4	W3	AIO	AUXADC external input channel 4	AVDD18_MD
REFP	W2	AIO	REFP	AVDD18_MD
RFIC_ET_N	N4	AIO	RFIC_ET_N	AVDD18_MD
RFIC_ET_P	M4	AIO	RFIC_ET_P	AVDD18_MD
APC1	G6	AIO	APC1	AVDD18_MD
MIPI DSI				
TDN0	AL20	AIO	DSI0 lane0 N	AVDD18_MIPITX0

Pin name	Pin No.	Type	Description	Power domain
TDP0	AL19	AIO	DSI0 lane0 P	AVDD18_MIPITX0
TDN1	AL22	AIO	DSI0 lane1 N	AVDD18_MIPITX0
TDP1	AL21	AIO	DSI0 lane1 P	AVDD18_MIPITX0
TDN2	AP22	AIO	DSI0 lane2 N	AVDD18_MIPITX0
TDP2	AN22	AIO	DSI0 lane2 P	AVDD18_MIPITX0
TDN3	AN21	AIO	DSI0 lane3 N	AVDD18_MIPITX0
TDP3	AN20	AIO	DSI0 lane3 P	AVDD18_MIPITX0
TCN	AM20	AIO	DSI0 CK lane N	AVDD18_MIPITX0
TCP	AM21	AIO	DSI0 CK lane P	AVDD18_MIPITX0
VRT	AN23	AO	External resistor for DSI bias Connect 1.5K ohm 1% resistor to ground.	AVDD18_MIPITX0
TDN0_A	AM18	AIO	DSI1 lane0 N	AVDD18_MIPITX1
TDP0_A	AL18	AIO	DSI1 lane0 P	AVDD18_MIPITX1
TDN1_A	AN15	AIO	DSI1 lane1 N	AVDD18_MIPITX1
TDP1_A	AN16	AIO	DSI1 lane1 P	AVDD18_MIPITX1
TDN2_A	AM15	AIO	DSI1 lane2 N	AVDD18_MIPITX1
TDP2_A	AL15	AIO	DSI1 lane2 P	AVDD18_MIPITX1
TDN3_A	AL17	AIO	DSI1 lane3 N	AVDD18_MIPITX1
TDP3_A	AL16	AIO	DSI1 lane3 P	AVDD18_MIPITX1
TCN_A	AM17	AIO	DSI1 CK lane N	AVDD18_MIPITX1
TCP_A	AM16	AIO	DSI1 CK lane P	AVDD18_MIPITX1
VRT_A	AN17	AO	External resistor for DSI bias Connect 1.5K ohm 1% resistor to ground.	AVDD18_MIPITX1
MIPI CSI				
RDN2	D15	AI0	CSI0 Lane 2 N	AVDD18_CSI
RDP2	E15	AI0	CSI0 Lane 2 P	AVDD18_CSI
RDN0	E17	AI0	CSI0 Lane 0 N	AVDD18_CSI
RDP0	D17	AI0	CSI0 Lane 0 P	AVDD18_CSI
RCN	D16	AI0	CSI0 CK Lane N	AVDD18_CSI
RCP	C16	AI0	CSI0 CK Lane P	AVDD18_CSI
RDN1	B16	AI0	CSI0 Lane 1 N	AVDD18_CSI
RDP1	B15	AI0	CSI0 Lane 1 P	AVDD18_CSI
RDN3	B14	AI0	CSI0 Lane 3 N	AVDD18_CSI
RDP3	A14	AI0	CSI0 Lane 3 P	AVDD18_CSI
RDN2_A	A19	AI0	CSI1 Lane 2 N	AVDD18_CSI
RDP2_A	B19	AI0	CSI1 Lane 2 P	AVDD18_CSI
RDN0_A	D19	AI0	CSI1 Lane 0 N	AVDD18_CSI
RDP0_A	D18	AI0	CSI1 Lane 0 P	AVDD18_CSI
RCN_A	B21	AI0	CSI1 CK Lane N	AVDD18_CSI
RCP_A	B20	AI0	CSI1 CK Lane P	AVDD18_CSI

Pin name	Pin No.	Type	Description	Power domain
RDN1_A	E19	AI0	CSI1 Lane 1 N	AVDD18_CSI
RDP1_A	E18	AI0	CSI1 Lane 1 P	AVDD18_CSI
RDN3_A	C20	AI0	CSI1 Lane 3 N	AVDD18_CSI
RDP3_A	D20	AI0	CSI1 Lane 3 P	AVDD18_CSI
RDN1_C	D21	AI0	Spare pin for CSI option	AVDD18_CSI
RDP1_C	E21	AI0	Spare pin for CSI option	AVDD18_CSI
RDN0_B	C18	AI0	CSI2 Lane 0 N	AVDD18_CSI
RDP0_B	C17	AI0	CSI2 Lane 0 P	AVDD18_CSI
RCN_B	A17	AI0	CSI2 CK Lane N	AVDD18_CSI
RCP_B	A16	AI0	CSI2 CK Lane P	AVDD18_CSI
RDN1_B	B17	AI0	CSI2 Lane 1 N	AVDD18_CSI
RDP1_B	B18	AI0	CSI2 Lane 1 P	AVDD18_CSI
USB				
USB_DM_P0	AK9	AIO	USB D+ differential data line	AVDD33_USB_P0
USB_DP_P0	AK8	AIO	USB D- differential data line	AVDD33_USB_P0
USB_DM_P1	AP7	AIO	USB D+ differential data line	AVDD33_USB_P1
USB_DP_P1	AN7	AIO	USB D- differential data line	AVDD33_USB_P1
CHD_DM_P0	AN9	AIO	BC1.1 charger DP	AVDD33_USB_P0
CHD_DP_P0	AN8	AIO	BC1.1 charger DM	AVDD33_USB_P0
SSUSB_RXN	AM13	AIO	SSUSB_RXN	AVDD10_SSUSB
SSUSB_RXP	AL13	AIO	SSUSB_RXP	AVDD10_SSUSB
SSUSB_TXN	AK11	AIO	SSUSB_TXN	AVDD10_SSUSB
SSUSB_TXP	AL11	AIO	SSUSB_TXP	AVDD10_SSUSB
SSUSB_VRT	AN12	AO	USB output for bias current; connect with 5.11K 1% Ohm to GND	AVDD10_SSUSB
WBG				
WF_IN	B30	AIO	IN for WIFI	AVDD18_WBG
WF_IP	B31	AIO	IP for WIFI	AVDD18_WBG
WF_QN	A29	AIO	QN for WIFI	AVDD18_WBG
WF_QP	A30	AIO	QP for WIFI	AVDD18_WBG
BT_IN	C27	AIO	IN for BT	AVDD18_WBG
BT_IP	C28	AIO	IP for BT	AVDD18_WBG
BT_QN	B27	AIO	QN for BT	AVDD18_WBG
BT_QP	A27	AIO	QP for BT	AVDD18_WBG
GPS_IN	A25	AIO	RX_IN for GPS RX	AVDD18_WBG
GPS_IP	B25	AIO	RX_IP for GPS RX	AVDD18_WBG
GPS_QN	B24	AIO	RX_QN for GPS RX	AVDD18_WBG
GPS_QP	A24	AIO	RX_QP for GPS RX	AVDD18_WBG
XIN_WBG	F26	AIO	26MHz clock input for WBG	AVDD18_WBG
External interrup				

Pin name	Pin No.	Type	Description	Power domain
EINT0	AP25	DIO	External interrupt 0	DVDD18_IO3
EINT1	AN25	DIO	External interrupt 1	DVDD18_IO3
EINT2	AL25	DIO	External interrupt 2	DVDD18_IO3
EINT3	AK25	DIO	External interrupt 3	DVDD18_IO3
EINT4	AN24	DIO	External interrupt 4	DVDD18_IO3
EINT5	AM24	DIO	External interrupt 5	DVDD18_IO3
EINT6	AL24	DIO	External interrupt 6	DVDD18_IO3
EINT7	AK24	DIO	External interrupt 7	DVDD18_IO3
EINT8	AK23	DIO	External interrupt 8	DVDD18_IO3
EINT9	AK22	DIO	External interrupt 9	DVDD18_IO3
EINT10	H34	DIO	External interrupt 10	DVDD18_IO2
EINT11	H30	DIO	External interrupt 11	DVDD18_IO2
EINT12	H29	DIO	External interrupt 12	DVDD18_IO2
EINT13	J33	DIO	External interrupt 13	DVDD18_IO2
EINT14	J30	DIO	External interrupt 14	DVDD18_IO2
EINT15	K30	DIO	External interrupt 15	DVDD18_IO2
EINT16	V33	DIO	External interrupt 16	DVDD18_IO2
MISC				
DRVBUS	AK26	DIO	USB OTG	DVDD18_IO3
IDDIG	AL6	DIO	USB OTG	DVDD18_IO4
JTCK	E10	DIO	TCK of JATG	DVDD18_IO0
JTDI	D10	DIO	TDI of JATG	DVDD18_IO0
JTDO	D11	DIO	TDO of JATG	DVDD18_IO0
JTMS	E11	DIO	TMS of JATG	DVDD18_IO0
JTRST_B	E9	DIO	TRST_B of JATG	DVDD18_IO0
UTXD0	A3	DIO	TX of UART0	DVDD18_IO0
URXD0	D2	DIO	RX of UART0	DVDD18_IO0
UTXD1	AM25	DIO	TX of UART1	DVDD18_IO3
URXD1	AM26	DIO	RX of UART1	DVDD18_IO3
MCU1PLLGP_TN	AF29	AIO	Reserved	AVDD18_MCU1PLLGP
MCU1PLLGP_TP	AE29	AIO	Reserved	AVDD18_MCU1PLLGP
Analog power				
AVDD18_AP	AB2	P	Analog power input 1.8V	-
AVDD18_MD	AA2	P	Analog power input 1.8V for modem	-
AVDD18_MDPLLGP	D3	P	Analog power input 1.8V for PLL	-
AVDD18_MCU1PLLGP	AE28	P	Analog power input 2.8V for PLL	-
AVDD18_SYSPLLGP	E33	P	Analog power input 2.8V for PLL	-
AVDD18_CSI	B12	P	Analog power for MIPI CSI	-
AVDD18_MIPITX0	AN19	P	Analog power for MIPI DSI0	-
AVDD18_MIPITX1	AP17	P	Analog power for MIPI DSI1	-

Pin name	Pin No.	Type	Description	Power domain
AVDD10_SSUSB	AP13	P	Analog power 0.97V for SSUSB	-
AVDD18_SSUSB	AM12	P	Analog power 1.8V for SSUSB	-
AVDD18_WBG	A32	P	Analog power 1.8V for WBG	-
AVDD28_DAC	F1	P	Analog power 1.8V for DAC	-
AVDD18_USB	AM8	P	Analog power 1.8V for USB	-
AVDD33_USB_P0	AM9	P	Analog power 3.3V for USB	-
AVDD33_USB_P1	AP9	P	Analog power 3.3V for USB	-
AVDD08_RDDR_AB	B2	P	Analog power 1.0V for DDRPHY	-
AVDD08_RDDR_CA	AN2	P	Analog power 1.0V for DDRPHY	-
AVDD08_RDDR_CB	C34	P	Analog power 1.0V for DDRPHY	-
AVDD15_ARDDR_B02	H27	P	Analog power 1.2V for CH_A DDRPHY	-
AVDD15_ARDDR_B02	H27	P	Analog power 1.2V for CH_A DDRPHY	-
AVDD15_ARDDR_B13	G8	P	Analog power 1.2V for CH_A DDRPHY	-
AVDD15_ARDDR_B13	G8	P	Analog power 1.2V for CH_A DDRPHY	-
AVDD15_ARDDR_CA	AH9	P	Analog power 1.2V for CH_A DDRPHY	-
AVDD15_ARDDR_CA	AH9	P	Analog power 1.2V for CH_A DDRPHY	-
AVDD15_BRDDR_B02	G7	P	Analog power 1.2V for CH_B DDRPHY	-
AVDD15_BRDDR_B02	G7	P	Analog power 1.2V for CH_B DDRPHY	-
AVDD15_BRDDR_B13	AG8	P	Analog power 1.2V for CH_B DDRPHY	-
AVDD15_BRDDR_B13	AG8	P	Analog power 1.2V for CH_B DDRPHY	-
AVDD15_BRDDR_CB	J28	P	Analog power 1.2V for CH_B DDRPHY	-
AVDD15_BRDDR_CB	J28	P	Analog power 1.2V for CH_B DDRPHY	-
AVDD18_RDDR_AB	F7	P	Analog power 1.8V for DDRPHY	-
AVDD18_RDDR_CA	AK6	P	Analog power 1.8V for DDRPHY	-
AVDD18_RDDR_CB	F28	P	Analog power 1.8V for DDRPHY	-
Digital power				
FSOURCE_P	D31	P	E-FUSE blowing power input	-
VDD1	C2	P	VDD1 of DRAM die	-
VDDQ	A6	P	VDDQ of DRAM die	-
DVDD_CORE	G11	P	Digital power input for CORE	-
DVDD_SRAM_CORE	G10	P	Digital power input for SRAM_CORE	-
DVDD_MD1	P8	P	Digital power input for MD1	-
DVDD_MODEM	AB8	P	Digital power input for MODEM	-
DVDD_PSMCU	Y16	P	Digital power input for PSMCU	-
DVDD_SRAM_MD	W8	P	Digital power input for SRAM_MD	-
DVDD_PROC1	AE23	P	Digital power input for CPU PROC1	-
DVDD_SRAM_PROC1	AG28	P	Digital power input for LDO SRAM_PROC1	-
DVDD_PROC2	Y25	P	Digital power input for CPU PROC2	-
DVDD_SRAM_PROC2	W28	P	Digital power input for LDO SRAM_PROC1	-
DVDD_GPU	G12	P	Digital power input for GPU	-

Pin name	Pin No.	Type	Description	Power domain
DVDD_SRAM_GPU	G16	P	Digital power input for LDO SRAM_GPU	-
DVDD18_IO0	A9	P	Digital power input for IO	-
DVDD18_IO1	A22	P	Digital power input for IO	-
DVDD18_IO2	J34	P	Digital power input for IO	-
DVDD18_IO3	AL34	P	Digital power input for IO	-
DVDD18_IO4	AC1	P	Digital power input for IO	-
DVDD18_MSDC0	AC34	P	Digital power input for MSDC0	-
DVDD18_MSDC1	AG33	P	Digital power input for MSDC1	-
DVDD28_MSDC1	AF34	P	Digital power input for MSDC1	-
DVDD28_SIM	AH33	P	Digital power input for SIM1	-
DVDD18_SIM	AJ33	P	Digital power input for SIM1	-
DVDD28_SIM2	AK34	P	Digital power input for SIM2	-
DVDD18_SIM2	AK33	P	Digital power input for SIM2	-
Analog ground				
AVSS18_MD	G3	G	Analog ground input for modem	-
AVSS18_AP	AB3	G	Analog ground input for modem	-
AVSS18_MDPLLGP	C3	G	Analog ground input for PLL	-
AVSS18_MCU1PLLGP	AF28	G	Analog ground input for PLL	-
AVSS18_CSI	B13	G	Analog ground input for MIPI CSI	-
AVSS18_MIPITX	AN18	G	Analog ground input for MIPI DSI	-
AVSS18_SYSPLLGP	D33	G	Analog ground input for PLL	-
AVSS18_WBG	B26	G	Analog ground input for WBG	-
AVSS33_USB	AM10	G	Analog ground input for USB	-
AVSS10_SSUSB	AN13	G	Analog ground input for SSUSB	-
AVSS08_RDDR_AB	B3	G	Analog ground input for DDRPHY	-
AVSS08_RDDR_CA	AM2	G	Analog ground input for DDRPHY	-
AVSS08_RDDR_CB	D34	G	Analog ground input for DDRPHY	-
AVSS_REFN	Y2	G	Analog ground input for ABB REFN	-
Digital ground				
DVSS		G		-

Table 2-4. Acronym for table of state of pins

Abbreviation	Description
I	Input
LO	Low output
HO	High output
XO	Low or high output
PU	Pull-up
PD	Pull-dowm
-	No PU/PD

Abbreviation	Description
0~N	Aux. function number
X	Delicate function pin

Table 2-5. State of pins

Name	Reset			Output drivability	Termination when not used	IO type
	State[1]	Aux[2]	PU/PD[3]			
System						
SYSRSTB	I	1	PU	DIOH1, DIOL1	No need	IO Type 1
WATCHDOG	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
TESTMODE	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
RTC32K_CK	I	1	-	DIOH1, DIOL1	No need	IO Type 1
SRCLKENA0	OH	1	-	DIOH1, DIOL1	No need	IO Type 1
SRCLKENA1	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
SRCLKENAI0	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SRCLKENAI1	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
PMIC						
PWRAP_SPI0_MO	I	1	PU	DIOH1, DIOL1	No need	IO Type 1
PWRAP_SPI0_MI	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
PWRAP_SPI0_CSN	OH	1	-	DIOH1, DIOL1	No need	IO Type 1
PWRAP_SPI0_CK	OH	1	PD	DIOH1, DIOL1	No need	IO Type 1
VOW_CLK_MISO	I	1	-	DIOH1, DIOL1	Tie low	IO Type 1
AUD_CLK_MOSI	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
AUD_DAT_MISO	I	1	-	DIOH1, DIOL1	Tie low	IO Type 1
AUD_DAT_MOSI	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
ANC_DAT_MOSI	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
AUD_INTN	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
AUD_PDN	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SIM						
SIM1_SCLK	I	0	PD	DIOH7, DIOL7	No need	IO Type 7
SIM1_SIO	I	0	PD	DIOH7, DIOL7	No need	IO Type 7
SIM1_SRST	I	0	PD	DIOH7, DIOL7	No need	IO Type 7
SIM2_SCLK	I	0	PD	DIOH7, DIOL7	No need	IO Type 7
SIM2_SIO	I	0	PD	DIOH7, DIOL7	No need	IO Type 7
SIM2_SRST	I	0	PD	DIOH7, DIOL7	No need	IO Type 7
INT_SIM1	I	0	PD	DIOH1, DIOL1	No need	IO Type 1

[1] The column “State” of “Reset” shows the pin state during reset (Input, High Output, Low Output, etc).
[2] The column “Aux” for “Reset” means the default aux funtion number shown in Table “Pin Multiplexing, Capability and Settings”.
[3] The column “PU/PD” for “Reset” means if there is internal pull-up or pull-down when the pin is input in the reset state.

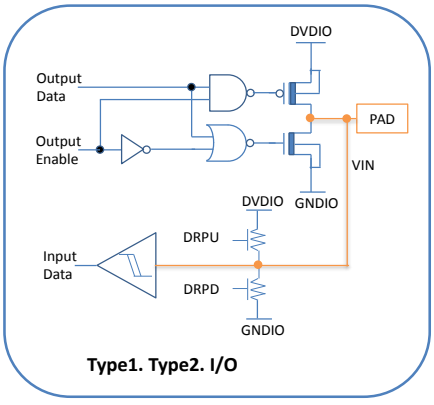
Name	Reset			Output drivability	Termination when not used	IO type
	State[1]	Aux[2]	PU/PD[3]			
INT_SIM2	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
LCD						
DSI_TE	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
LCM_RST	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
DPI_HSYNC	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_VSYNC	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_CK	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_DE	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D11	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D10	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D9	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D8	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D7	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D6	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D5	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D4	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D3	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D2	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D1	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
DPI_D0	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
PWM						
DISP_PWM	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
Keypad interface						
KPCOL0	I	1	PU	DIOH1, DIOL1	No need	IO Type 1
KPCOL1	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
KPCOL2	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
KPROW0	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
KPROW1	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
KPROW2	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2S interface						
TDM_BCK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
TDM_DATA0	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
TDM_DATA1	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
TDM_DATA2	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
TDM_DATA3	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
TDM_LRCK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
TDM_MCK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2S0_BCK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2S0_DI	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2S0_LRCK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2S0_MCK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1

Name	Reset			Output drivability	Termination when not used	IO type
	State[1]	Aux[2]	PU/PD[3]			
I2S3_DO	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2S1_BCK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2S1_DO	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2S1_LRCK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2S1_MCK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2S2_DI	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI						
SPI0_CS	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI0_MI	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI0_MO	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI0_CK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI1_CS	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI1_MI	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI1_MO	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI1_CK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI2_CS	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI2_MI	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI2_MO	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI2_CK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI3_CS	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI3_MI	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI3_MO	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
SPI3_CK	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
BPI						
BPI_BUS0	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS1	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS2	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS3	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS4	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS5	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS6	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS7	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS8	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS9	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS10	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS11	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS12_ANT0	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS13_ANT1	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS14_ANT2	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS15_ANT3	OL	1	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS16_VM0	OL	2	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS17_VM1	OL	2	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS18_SWP0	OL	2	-	DIOH1, DIOL1	No need	IO Type 1

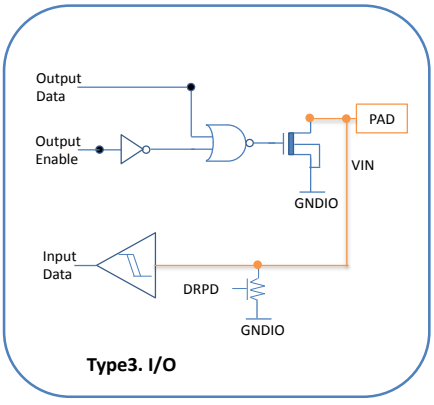
Name	Reset			Output drivability	Termination when not used	IO type
	State[1]	Aux[2]	PU/PD[3]			
BPI_BUS19_SWP1	OL	2	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS20_SWP2	OL	2	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS21_SWP3	OL	2	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS22_DET0	OL	2	-	DIOH1, DIOL1	No need	IO Type 1
BPI_BUS23_DET1	OL	2	-	DIOH1, DIOL1	No need	IO Type 1
BSI						
RFIC0_BSI_CK	OL	1	PD	DIOH1, DIOL1	No need	IO Type 1
RFIC0_BSI_D0	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
RFIC0_BSI_D1	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
RFIC0_BSI_D2	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
RFIC0_BSI_EN	OL	1	PD	DIOH1, DIOL1	No need	IO Type 1
MISC_MIPI_CK_0	OL	1	PD	DIOH1, DIOL1	No need	IO Type 1
MISC_MIPI_DO_0	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
MISC_MIPI_CK_1	OL	1	PD	DIOH1, DIOL1	No need	IO Type 1
MISC_MIPI_DO_1	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
MISC_MIPI_CK_2	OL	1	PD	DIOH1, DIOL1	No need	IO Type 1
MISC_MIPI_DO_2	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
MISC_MIPI_CK_3	OL	1	PD	DIOH1, DIOL1	No need	IO Type 1
MISC_MIPI_DO_3	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
MSDC1						
MSDC1_CLK	OL	1	PD	DIOH6, DIOL6	No need	IO Type 6
MSDC1_CMD	I	1	PU	DIOH6, DIOL6	No need	IO Type 6
MSDC1_DAT0	I	1	PU	DIOH6, DIOL6	No need	IO Type 6
MSDC1_DAT1	I	1	PU	DIOH6, DIOL6	No need	IO Type 6
MSDC1_DAT2	I	1	PU	DIOH6, DIOL6	No need	IO Type 6
MSDC1_DAT3	I	1	PU	DIOH6, DIOL6	No need	IO Type 6
MSDC0						
MSDC0_CLK	OL	1	PD	DIOH4, DIOL4	No need	IO Type 4
MSDC0_CMD	I	1	PU	DIOH4, DIOL4	No need	IO Type 4
MSDC0_DAT0	I	1	PU	DIOH4, DIOL4	No need	IO Type 4
MSDC0_DAT1	I	1	PU	DIOH4, DIOL4	No need	IO Type 4
MSDC0_DAT2	I	1	PU	DIOH4, DIOL4	No need	IO Type 4
MSDC0_DAT3	I	1	PU	DIOH4, DIOL4	No need	IO Type 4
MSDC0_DAT4	I	1	PU	DIOH4, DIOL4	No need	IO Type 4
MSDC0_DAT5	I	1	PU	DIOH4, DIOL4	No need	IO Type 4
MSDC0_DAT6	I	1	PU	DIOH4, DIOL4	No need	IO Type 4
MSDC0_DAT7	I	1	PU	DIOH4, DIOL4	No need	IO Type 4
MSDC0_DSL	I	1	PD	DIOH4, DIOL4	No need	IO Type 4
MSDC0_RSTB	OH	1	PU	DIOH4, DIOL4	No need	IO Type 4
CAM						
CAM_CLK0	I	0	PD	DIOH1, DIOL1	No need	IO Type 1

Name	Reset			Output drivability	Termination when not used	IO type
	State[1]	Aux[2]	PU/PD[3]			
CAM_PDN0	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
CAM_RST0	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
CAM_CLK1	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
CAM_PDN1	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
CAM_RST1	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
CAM_CLK2	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
CAM_PDN2	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
CAM_RST2	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
I2C						
SCL0	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SDA0	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SCL1	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SDA1	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SCL2	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SDA2	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SCL3	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SDA3	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SCL4	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SDA4	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SCL5	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SDA5	I	1	-	DIOH3, DIOL3	No need	IO Type 3
SCL6	I	1	PU	DIOH1, DIOL1	No need	IO Type 1
SDA6	I	1	PU	DIOH1, DIOL1	No need	IO Type 1
SCL7	I	1	PU	DIOH1, DIOL1	No need	IO Type 1
SDA7	I	1	PU	DIOH1, DIOL1	No need	IO Type 1
CONN						
CONN_BT_CLK	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
CONN_BT_DATA	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
CONN_HRST_B	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
CONN_TOP_CLK	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
CONN_TOP_DATA	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
CONN_WB_PTA	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
CONN_WF_CTRL0	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
CONN_WF_CTRL1	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
CONN_WF_CTRL2	I	1	PD	DIOH1, DIOL1	No need	IO Type 1
External interrup						
EINT0	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT1	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT2	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT3	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT4	I	0	PD	DIOH1, DIOL1	No need	IO Type 1

Name	Reset			Output drivability	Termination when not used	IO type
	State[1]	Aux[2]	PU/PD[3]			
EINT5	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT6	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT7	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT8	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT9	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT10	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT11	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT12	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT13	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT14	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT15	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
EINT16	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
MISC						
DRVBUS	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
IDDIG	I	0	PD	DIOH1, DIOL1	No need	IO Type 1
JTCK	I	1	PU	DIOH2, DIOL2	No need	IO Type 2
JTDI	I	1	PU	DIOH2, DIOL2	No need	IO Type 2
JTDO	I	1	PD	DIOH2, DIOL2	No need	IO Type 2
JTMS	I	1	PU	DIOH2, DIOL2	No need	IO Type 2
JTRST_B	I	0	PD	DIOH2, DIOL2	No need	IO Type 2
UTXD0	OH	1	-	DIOH2, DIOL2	No need	IO Type 2
URXD0	I	1	PU	DIOH2, DIOL2	No need	IO Type 2
UTXD1	OH	1	-	DIOH2, DIOL2	No need	IO Type 2
URXD1	I	1	PU	DIOH2, DIOL2	No need	IO Type 2



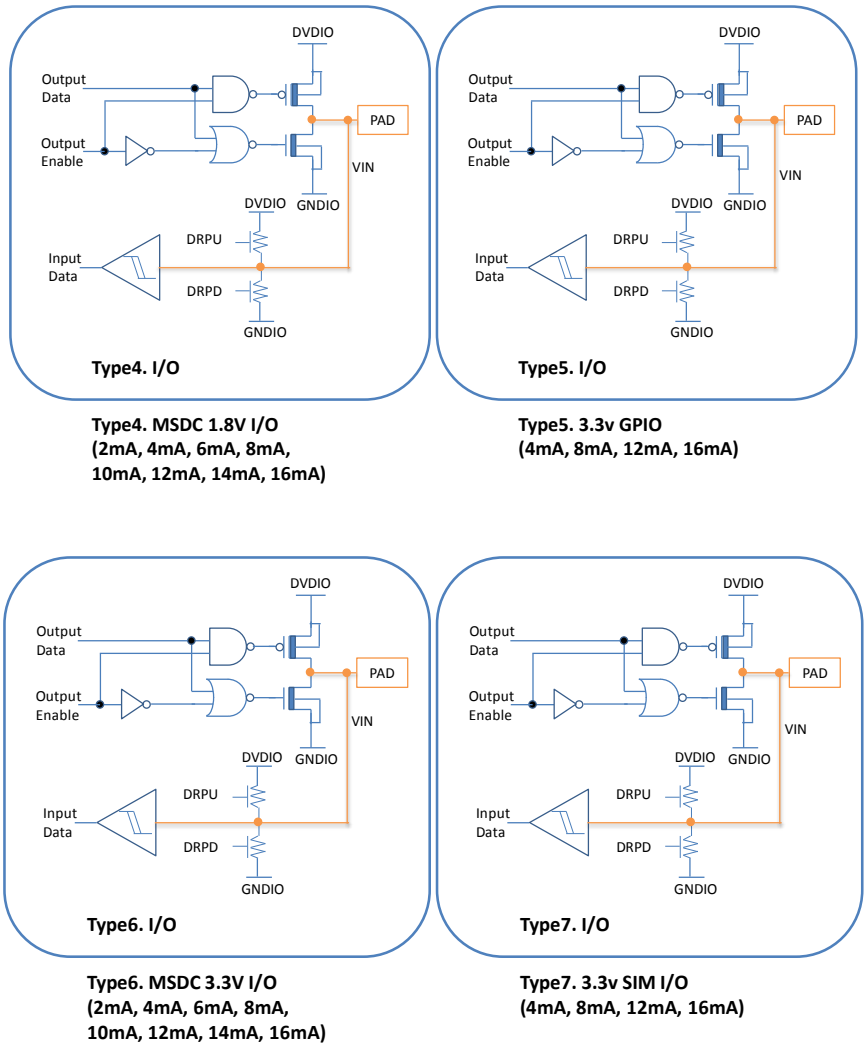
Type1. Type2. I/O



Type3. I/O

Type1. GPIO 28 type (2mA, 4mA, 6mA, 8mA)
Type2. GPIO 4G type (4mA, 8mA, 12mA, 16mA)

Type 3. Open-drain IO



Type4. MSDC 1.8V I/O (2mA, 4mA, 6mA, 8mA, 10mA, 12mA, 14mA, 16mA)

Type5. 3.3v GPIO (4mA, 8mA, 12mA, 16mA)

Type6. MSDC 3.3V I/O (2mA, 4mA, 6mA, 8mA, 10mA, 12mA, 14mA, 16mA)

Type7. 3.3v SIM I/O (4mA, 8mA, 12mA, 16mA)

Figure 2-2. IO types in state of pins

2.1.4 Pin multiplexing, Capability and Settings

Table 2-6. Acronym for pull-up and pull-down type

Abbreviation	Description
PU	Pull-up, not controllable
PD	Pull-down, not controllable
CU	Pull-up, controllable
CD	Pull-down, controllable
X	Cannot pull-up or pull-dowm

Table 2-7. Pin multiplexing, capability and settings

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
RDP2	N	0	GPI0	I	-	-	-
		1	CSI0A_L0P_T0A	AIO	-	-	-
RDN2	N	0	GPI1	I	-	-	-
		1	CSI0A_L0N_T0B	AIO	-	-	-
RDP0	N	0	GPI2	I	-	-	-
		1	CSI0A_L1P_T0C	AIO	-	-	-
RDN0	N	0	GPI3	I	-	-	-
		1	CSI0A_L1N_T1A	AIO	-	-	-
RCP	N	0	GPI4	I	-	-	-
		1	CSI0A_L2P_T1B	AIO	-	-	-
RCN	N	0	GPI5	I	-	-	-
		1	CSI0A_L2N_T1C	AIO	-	-	-
RDP1	N	0	GPI6	I	-	-	-
		1	CSI0B_L0P_T0A	AIO	-	-	-
RDN1	N	0	GPI7	I	-	-	-
		1	CSI0B_L0N_T0B	AIO	-	-	-
RDP3	N	0	GPI8	I	-	-	-
		1	CSI0B_L1P_T0C	AIO	-	-	-
RDN3	N	0	GPI9	I	-	-	-
		1	CSI0B_L1N_T1A	AIO	-	-	-
RDP2_A	N	0	GPI10	I	-	-	-
		1	CSI1A_L0P_T0A	AIO	-	-	-
RDN2_A	N	0	GPI11	I	-	-	-
		1	CSI1A_L0N_T0B	AIO	-	-	-
RDP0_A	N	0	GPI12	I	-	-	-
		1	CSI1A_L1P_T0C	AIO	-	-	-
RDN0_A	N	0	GPI13	I	-	-	-
		1	CSI1A_L1N_T1A	AIO	-	-	-
RCP_A	N	0	GPI14	I	-	-	-
		1	CSI1A_L2P_T1B	AIO	-	-	-
RCN_A	N	0	GPI15	I	-	-	-
		1	CSI1A_L2N_T1C	AIO	-	-	-
RDP1_A	N	0	GPI16	I	-	-	-
		1	CSI1B_L0P_T0A	AIO	-	-	-
RDN1_A	N	0	GPI17	I	-	-	-
		1	CSI1B_L0N_T0B	AIO	-	-	-
RDP3_A	N	0	GPI18	I	-	-	-
		1	CSI1B_L1P_T0C	AIO	-	-	-

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
RDN3_A	N	0	GPI19	I	-	-	-
		1	CSI1B_L1N_T1A	AIO	-	-	-
RDP1_C	N	0	GPI20	I	-	-	-
		1	CSI1B_L2P_T1B	AIO	-	-	-
RDN1_C	N	0	GPI21	I	-	-	-
		1	CSI1B_L2N_T1C	AIO	-	-	-
RDP0_B	N	0	GPI22	I	-	-	-
		1	CSI2_L0P_T0A	AIO	-	-	-
RDN0_B	N	0	GPI23	I	-	-	-
		1	CSI2_L0N_T0B	AIO	-	-	-
RCP_B	N	0	GPI24	I	-	-	-
		1	CSI2_L1P_T0C	AIO	-	-	-
RCN_B	N	0	GPI25	I	-	-	-
		1	CSI2_L1N_T1A	AIO	-	-	-
RDP1_B	N	0	GPI26	I	-	-	-
		1	CSI2_L2P_T1B	AIO	-	-	-
RDN1_B	N	0	GPI27	I	-	-	-
		1	CSI2_L2N_T1C	AIO	-	-	-
CAM_PDN0	Y	0	GPIO28	IO	CU, CD	2/4/6/8 mA	0
		1	SPI5_CLK_A	O	CU, CD	2/4/6/8 mA	0
		2	IRTX_OUT	O	CU, CD	2/4/6/8 mA	0
		3	UDI_TDO	O	CU, CD	2/4/6/8 mA	0
		4	SCP_JTAG_TDO	O	CU, CD	2/4/6/8 mA	0
		5	CONN_MCU_TDO	O	CU, CD	2/4/6/8 mA	0
		6	PWM_A	O	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_OTDO	O	CU, CD	2/4/6/8 mA	0
CAM_PDN1	Y	0	GPIO29	IO	CU, CD	2/4/6/8 mA	0
		1	SPI5_MI_A	I	CU, CD	2/4/6/8 mA	0
		2	DAP_SIB1_SWD	IO	CU, CD	2/4/6/8 mA	0
		3	UDI_TMS	I	CU, CD	2/4/6/8 mA	0
		4	SCP_JTAG_TMS	IO	CU, CD	2/4/6/8 mA	0
		5	CONN_MCU_TMS	I	CU, CD	2/4/6/8 mA	0
		6	CONN_MCU_AICE_TMSC	IO	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_OTMS	I	CU, CD	2/4/6/8 mA	0
CAM_CLK0	N	0	GPIO30	IO	CU, CD	2/4/6/8 mA	0
		1	CMMCLK0	O	CU, CD	2/4/6/8 mA	0
		7	MD_CLKM0	O	CU, CD	2/4/6/8 mA	0
CAM_CLK1	Y	0	GPIO31	IO	CU, CD	2/4/6/8 mA	0
		1	CMMCLK1	O	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		7	MD_CLKM1	O	CU, CD	2/4/6/8 mA	0
CAM_RST0	Y	0	GPIO32	IO	CU, CD	2/4/6/8 mA	0
		1	SPI5_CS_A	O	CU, CD	2/4/6/8 mA	0
		2	DAP_SIB1_SWCK	I	CU, CD	2/4/6/8 mA	0
		3	UDI_TCK_XI	I	CU, CD	2/4/6/8 mA	0
		4	SCP_JTAG_TCK	I	CU, CD	2/4/6/8 mA	0
		5	CONN_MCU_TCK	I	CU, CD	2/4/6/8 mA	0
		6	CONN_MCU_AICE_TCKC	I	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_OTCK	I	CU, CD	2/4/6/8 mA	0
CAM_RST1	Y	0	GPIO33	IO	CU, CD	2/4/6/8 mA	0
		1	SPI5_MO_A	O	CU, CD	2/4/6/8 mA	0
		2	CMFLASH	O	CU, CD	2/4/6/8 mA	0
		3	UDI_TDI	I	CU, CD	2/4/6/8 mA	0
		4	SCP_JTAG_TDI	I	CU, CD	2/4/6/8 mA	0
		5	CONN_MCU_TDI	I	CU, CD	2/4/6/8 mA	0
		6	MD_URXD0	I	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_OTDI	I	CU, CD	2/4/6/8 mA	0
CAM_PDN2	Y	0	GPIO34	IO	CU, CD	2/4/6/8 mA	0
		1	CMFLASH	O	CU, CD	2/4/6/8 mA	0
		2	CLKM0	O	CU, CD	2/4/6/8 mA	0
		3	UDI_NTRST	I	CU, CD	2/4/6/8 mA	0
		4	SCP_JTAG_TRSTN	I	CU, CD	2/4/6/8 mA	0
		5	CONN_MCU_TRST_B	I	CU, CD	2/4/6/8 mA	0
		6	MD_UTXD0	O	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_JTINTP	O	CU, CD	2/4/6/8 mA	0
CAM_RST2	Y	0	GPIO35	IO	CU, CD	2/4/6/8 mA	0
		1	CMMCLK3	O	CU, CD	2/4/6/8 mA	0
		2	CLKM1	O	CU, CD	2/4/6/8 mA	0
		3	MD_URXD1	I	CU, CD	2/4/6/8 mA	0
		4	PTA_RXD	I	CU, CD	2/4/6/8 mA	0
		5	CONN_MCU_DBGACK_N	O	CU, CD	2/4/6/8 mA	0
		6	PWM_B	O	CU, CD	2/4/6/8 mA	0
		7	PCC_PPC_IO	IO	CU, CD	2/4/6/8 mA	0
CAM_CLK2	Y	0	GPIO36	IO	CU, CD	2/4/6/8 mA	0
		1	CMMCLK2	O	CU, CD	2/4/6/8 mA	0
		2	CLKM2	O	CU, CD	2/4/6/8 mA	0
		3	MD_UTXD1	O	CU, CD	2/4/6/8 mA	0
		4	PTA_TXD	O	CU, CD	2/4/6/8 mA	0
		5	CONN_MCU_DBGI_N	I	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		6	PWM_C	O	CU, CD	2/4/6/8 mA	0
		7	EXT_FRAME_SYNC	I	CU, CD	2/4/6/8 mA	0
SCL0	Y	0	GPIO37	IO	CD	-	0
		1	SCL0_0	IO	CD	-	0
SDA0	Y	0	GPIO38	IO	CD	-	0
		1	SDA0_0	IO	CD	-	0
DPI_D0	Y	0	GPIO39	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D0	O	CU, CD	4/8/12/16 mA	0
		2	SPI1_CLK_A	O	CU, CD	4/8/12/16 mA	0
		3	PCM0_SYNC	O	CU, CD	4/8/12/16 mA	0
		4	I2S0_LRCK	IO	CU, CD	4/8/12/16 mA	0
		5	CONN_MCU_TRST_B	I	CU, CD	4/8/12/16 mA	0
		6	URXD3	I	CU, CD	4/8/12/16 mA	0
		7	C2K_NTRST	I	CU, CD	4/8/12/16 mA	0
DPI_D1	Y	0	GPIO40	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D1	O	CU, CD	4/8/12/16 mA	0
		2	SPI1_MI_A	I	CU, CD	4/8/12/16 mA	0
		3	PCM0_CLK	O	CU, CD	4/8/12/16 mA	0
		4	I2S0_BCK	IO	CU, CD	4/8/12/16 mA	0
		5	CONN_MCU_TDO	O	CU, CD	4/8/12/16 mA	0
		6	UTXD3	O	CU, CD	4/8/12/16 mA	0
		7	C2K_TCK	I	CU, CD	4/8/12/16 mA	0
DPI_D2	Y	0	GPIO41	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D2	O	CU, CD	4/8/12/16 mA	0
		2	SPI1_CS_A	O	CU, CD	4/8/12/16 mA	0
		3	PCM0_DO	O	CU, CD	4/8/12/16 mA	0
		4	I2S3_DO	O	CU, CD	4/8/12/16 mA	0
		5	CONN_MCU_DBGACK_N	O	CU, CD	4/8/12/16 mA	0
		6	URTS3	O	CU, CD	4/8/12/16 mA	0
		7	C2K_TDI	I	CU, CD	4/8/12/16 mA	0
DPI_D3	Y	0	GPIO42	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D3	O	CU, CD	4/8/12/16 mA	0
		2	SPI1_MO_A	O	CU, CD	4/8/12/16 mA	0
		3	PCM0_DI	I	CU, CD	4/8/12/16 mA	0
		4	I2S0_DI	I	CU, CD	4/8/12/16 mA	0
		5	CONN_MCU_TDI	I	CU, CD	4/8/12/16 mA	0
		6	UCTS3	I	CU, CD	4/8/12/16 mA	0
		7	C2K_TMS	I	CU, CD	4/8/12/16 mA	0
DPI_D4	Y	0	GPIO43	IO	CU, CD	4/8/12/16 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		1	DPI_D4	O	CU, CD	4/8/12/16 mA	0
		2	SPI2_CLK_A	O	CU, CD	4/8/12/16 mA	0
		3	PCM1_SYNC	IO	CU, CD	4/8/12/16 mA	0
		4	I2S2_LRCK	O	CU, CD	4/8/12/16 mA	0
		5	CONN_MCU_TMS	I	CU, CD	4/8/12/16 mA	0
		6	CONN_MCU_AICE_TMSC	IO	CU, CD	4/8/12/16 mA	0
		7	C2K_TDO	IO	CU, CD	4/8/12/16 mA	0
DPI_D5	Y	0	GPIO44	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D5	O	CU, CD	4/8/12/16 mA	0
		2	SPI2_MI_A	I	CU, CD	4/8/12/16 mA	0
		3	PCM1_CLK	IO	CU, CD	4/8/12/16 mA	0
		4	I2S2_BCK	O	CU, CD	4/8/12/16 mA	0
		5	CONN_MCU_TCK	I	CU, CD	4/8/12/16 mA	0
		6	CONN_MCU_AICE_TCKC	I	CU, CD	4/8/12/16 mA	0
		7	C2K_RTCK	O	CU, CD	4/8/12/16 mA	0
DPI_D6	Y	0	GPIO45	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D6	O	CU, CD	4/8/12/16 mA	0
		2	SPI2_CS_A	O	CU, CD	4/8/12/16 mA	0
		3	PCM1_DI	I	CU, CD	4/8/12/16 mA	0
		4	I2S2_DI	I	CU, CD	4/8/12/16 mA	0
		5	CONN_MCU_DBGI_N	I	CU, CD	4/8/12/16 mA	0
		6	MD_URXD0	I	CU, CD	4/8/12/16 mA	0
DPI_D7	Y	0	GPIO46	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D7	O	CU, CD	4/8/12/16 mA	0
		2	SPI2_MO_A	O	CU, CD	4/8/12/16 mA	0
		3	PCM1_DO0	O	CU, CD	4/8/12/16 mA	0
		4	I2S1_DO	O	CU, CD	4/8/12/16 mA	0
		5	ANT_SEL0	O	CU, CD	4/8/12/16 mA	0
		6	MD_UTXD0	O	CU, CD	4/8/12/16 mA	0
DPI_D8	Y	0	GPIO47	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D8	O	CU, CD	4/8/12/16 mA	0
		2	CLKM0	O	CU, CD	4/8/12/16 mA	0
		3	PCM1_DO1	O	CU, CD	4/8/12/16 mA	0
		4	I2S0_MCK	O	CU, CD	4/8/12/16 mA	0
		5	ANT_SEL1	O	CU, CD	4/8/12/16 mA	0
		6	PTA_RXD	I	CU, CD	4/8/12/16 mA	0
		7	C2K_URXD0	I	CU, CD	4/8/12/16 mA	0
DPI_D9	Y	0	GPIO48	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D9	O	CU, CD	4/8/12/16 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		2	CLKM1	O	CU, CD	4/8/12/16 mA	0
		3	CMFLASH	O	CU, CD	4/8/12/16 mA	0
		4	I2S2_MCK	O	CU, CD	4/8/12/16 mA	0
		5	ANT_SEL2	O	CU, CD	4/8/12/16 mA	0
		6	PTA_TXD	O	CU, CD	4/8/12/16 mA	0
		7	C2K_UTXD0	O	CU, CD	4/8/12/16 mA	0
DPI_D10	Y	0	GPIO49	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D10	O	CU, CD	4/8/12/16 mA	0
		2	MD_INT1_C2K_UIM1_HOT_PLUG_IN	I	CU, CD	4/8/12/16 mA	0
		3	PWM_C	O	CU, CD	4/8/12/16 mA	0
		4	IRTX_OUT	O	CU, CD	4/8/12/16 mA	0
		5	ANT_SEL3	O	CU, CD	4/8/12/16 mA	0
		6	MD_URXD1	I	CU, CD	4/8/12/16 mA	0
DPI_D11	Y	0	GPIO50	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_D11	O	CU, CD	4/8/12/16 mA	0
		2	MD_INT2	I	CU, CD	4/8/12/16 mA	0
		3	PWM_D	O	CU, CD	4/8/12/16 mA	0
		4	CLKM2	O	CU, CD	4/8/12/16 mA	0
		5	ANT_SEL4	O	CU, CD	4/8/12/16 mA	0
		6	MD_UTXD1	O	CU, CD	4/8/12/16 mA	0
DPI_DE	Y	0	GPIO51	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_DE	O	CU, CD	4/8/12/16 mA	0
		2	SPI4_CLK_A	O	CU, CD	4/8/12/16 mA	0
		3	IRTX_OUT	O	CU, CD	4/8/12/16 mA	0
		4	SCL0_1	IO	CU, CD	4/8/12/16 mA	0
		5	ANT_SEL5	O	CU, CD	4/8/12/16 mA	0
		7	C2K_UTXD1	IO	CU, CD	4/8/12/16 mA	0
DPI_CK	Y	0	GPIO52	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_CK	O	CU, CD	4/8/12/16 mA	0
		2	SPI4_MI_A	I	CU, CD	4/8/12/16 mA	0
		3	SPI4_MO_A	O	CU, CD	4/8/12/16 mA	0
		4	SDA0_1	IO	CU, CD	4/8/12/16 mA	0
		5	ANT_SEL6	O	CU, CD	4/8/12/16 mA	0
		7	C2K_URXD1	IO	CU, CD	4/8/12/16 mA	0
DPI_HSYNC	Y	0	GPIO53	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_HSYNC	O	CU, CD	4/8/12/16 mA	0
		2	SPI4_CS_A	O	CU, CD	4/8/12/16 mA	0
		3	CMFLASH	O	CU, CD	4/8/12/16 mA	0
		4	SCL1_1	IO	CU, CD	4/8/12/16 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		5	ANT_SEL7	O	CU, CD	4/8/12/16 mA	0
		6	MD_URXD2	I	CU, CD	4/8/12/16 mA	0
		7	PCC_PPC_IO	IO	CU, CD	4/8/12/16 mA	0
DPI_VSYNC	Y	0	GPIO54	IO	CU, CD	4/8/12/16 mA	0
		1	DPI_VSYNC	O	CU, CD	4/8/12/16 mA	0
		2	SPI4_MO_A	O	CU, CD	4/8/12/16 mA	0
		3	SPI4_MI_A	I	CU, CD	4/8/12/16 mA	0
		4	SDA1_1	IO	CU, CD	4/8/12/16 mA	0
		5	PWM_A	O	CU, CD	4/8/12/16 mA	0
		6	MD_UTXD2	O	CU, CD	4/8/12/16 mA	0
		7	EXT_FRAME_SYNC	I	CU, CD	4/8/12/16 mA	0
SCL1	Y	0	GPIO55	IO	CD	-	0
		1	SCL1_0	IO	CD	-	0
SDA1	Y	0	GPIO56	IO	CD	-	0
		1	SDA1_0	IO	CD	-	0
SPI0_CK	Y	0	GPIO57	IO	CU, CD	2/4/6/8 mA	0
		1	SPI0_CLK	O	CU, CD	2/4/6/8 mA	0
		2	SCL0_2	IO	CU, CD	2/4/6/8 mA	0
		3	PWM_B	O	CU, CD	2/4/6/8 mA	0
		4	UTXD3	O	CU, CD	2/4/6/8 mA	0
		5	PCM0_SYNC	O	CU, CD	2/4/6/8 mA	0
SPI0_MI	Y	0	GPIO58	IO	CU, CD	2/4/6/8 mA	0
		1	SPI0_MI	I	CU, CD	2/4/6/8 mA	0
		2	SPI0_MO	O	CU, CD	2/4/6/8 mA	0
		3	SDA1_2	IO	CU, CD	2/4/6/8 mA	0
		4	URXD3	I	CU, CD	2/4/6/8 mA	0
		5	PCM0_CLK	O	CU, CD	2/4/6/8 mA	0
SPI0_MO	Y	0	GPIO59	IO	CU, CD	2/4/6/8 mA	0
		1	SPI0_MO	O	CU, CD	2/4/6/8 mA	0
		2	SPI0_MI	I	CU, CD	2/4/6/8 mA	0
		3	PWM_C	O	CU, CD	2/4/6/8 mA	0
		4	URTS3	O	CU, CD	2/4/6/8 mA	0
		5	PCM0_DO	O	CU, CD	2/4/6/8 mA	0
SPI0_CS	Y	0	GPIO60	IO	CU, CD	2/4/6/8 mA	0
		1	SPI0_CS	O	CU, CD	2/4/6/8 mA	0
		2	SDA0_2	IO	CU, CD	2/4/6/8 mA	0
		3	SCL1_2	IO	CU, CD	2/4/6/8 mA	0
		4	UCTS3	I	CU, CD	2/4/6/8 mA	0
		5	PCM0_DI	I	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
EINT0	Y	0	GPIO61	IO	CU, CD	2/4/6/8 mA	0
		1	EINT0	I	CU, CD	2/4/6/8 mA	0
		2	IDDIG	I	CU, CD	2/4/6/8 mA	0
		3	SPI4_CLK_B	O	CU, CD	2/4/6/8 mA	0
		4	I2S0_LRCK	IO	CU, CD	2/4/6/8 mA	0
		5	PCM0_SYNC	O	CU, CD	2/4/6/8 mA	0
		7	C2K_EINT0	IO	CU, CD	2/4/6/8 mA	0
EINT1	Y	0	GPIO62	IO	CU, CD	2/4/6/8 mA	0
		1	EINT1	I	CU, CD	2/4/6/8 mA	0
		2	USB_DRVVBUS	O	CU, CD	2/4/6/8 mA	0
		3	SPI4_MI_B	I	CU, CD	2/4/6/8 mA	0
		4	I2S0_BCK	IO	CU, CD	2/4/6/8 mA	0
		5	PCM0_CLK	O	CU, CD	2/4/6/8 mA	0
		7	C2K_EINT1	IO	CU, CD	2/4/6/8 mA	0
EINT2	Y	0	GPIO63	IO	CU, CD	2/4/6/8 mA	0
		1	EINT2	I	CU, CD	2/4/6/8 mA	0
		2	IRTX_OUT	O	CU, CD	2/4/6/8 mA	0
		3	SPI4_MO_B	O	CU, CD	2/4/6/8 mA	0
		4	I2S0_MCK	O	CU, CD	2/4/6/8 mA	0
		5	PCM0_DI	I	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT0	IO	CU, CD	2/4/6/8 mA	0
EINT3	Y	0	GPIO64	IO	CU, CD	2/4/6/8 mA	0
		1	EINT3	I	CU, CD	2/4/6/8 mA	0
		2	CMFLASH	O	CU, CD	2/4/6/8 mA	0
		3	SPI4_CS_B	O	CU, CD	2/4/6/8 mA	0
		4	I2S0_DI	I	CU, CD	2/4/6/8 mA	0
		5	PCM0_DO	O	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT1	IO	CU, CD	2/4/6/8 mA	0
EINT4	Y	0	GPIO65	IO	CU, CD	2/4/6/8 mA	0
		1	EINT4	I	CU, CD	2/4/6/8 mA	0
		2	CLKM0	O	CU, CD	2/4/6/8 mA	0
		3	SPI5_CLK_B	O	CU, CD	2/4/6/8 mA	0
		4	I2S1_LRCK	O	CU, CD	2/4/6/8 mA	0
		5	PWM_A	O	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT2	IO	CU, CD	2/4/6/8 mA	0
EINT5	Y	0	GPIO66	IO	CU, CD	2/4/6/8 mA	0
		1	EINT5	I	CU, CD	2/4/6/8 mA	0
		2	CLKM1	O	CU, CD	2/4/6/8 mA	0
		3	SPI5_MI_B	I	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		4	I2S1_BCK	O	CU, CD	2/4/6/8 mA	0
		5	PWM_B	O	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT3	IO	CU, CD	2/4/6/8 mA	0
EINT6	Y	0	GPIO67	IO	CU, CD	2/4/6/8 mA	0
		1	EINT6	I	CU, CD	2/4/6/8 mA	0
		2	CLKM2	O	CU, CD	2/4/6/8 mA	0
		3	SPI5_MO_B	O	CU, CD	2/4/6/8 mA	0
		4	I2S1_MCK	O	CU, CD	2/4/6/8 mA	0
		5	PWM_C	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A0	IO	CU, CD	2/4/6/8 mA	0
EINT7	Y	0	GPIO68	IO	CU, CD	2/4/6/8 mA	0
		1	EINT7	I	CU, CD	2/4/6/8 mA	0
		2	CLKM3	O	CU, CD	2/4/6/8 mA	0
		3	SPI5_CS_B	O	CU, CD	2/4/6/8 mA	0
		4	I2S1_DO	O	CU, CD	2/4/6/8 mA	0
		5	PWM_D	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A1	IO	CU, CD	2/4/6/8 mA	0
I2S0_LRCK	Y	0	GPIO69	IO	CU, CD	2/4/6/8 mA	0
		1	I2S0_LRCK	IO	CU, CD	2/4/6/8 mA	0
		2	I2S3_LRCK	O	CU, CD	2/4/6/8 mA	0
		3	I2S1_LRCK	O	CU, CD	2/4/6/8 mA	0
		4	I2S2_LRCK	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A2	IO	CU, CD	2/4/6/8 mA	0
I2S0_BCK	Y	0	GPIO70	IO	CU, CD	2/4/6/8 mA	0
		1	I2S0_BCK	IO	CU, CD	2/4/6/8 mA	0
		2	I2S3_BCK	O	CU, CD	2/4/6/8 mA	0
		3	I2S1_BCK	O	CU, CD	2/4/6/8 mA	0
		4	I2S2_BCK	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A3	IO	CU, CD	2/4/6/8 mA	0
I2S0_MCK	Y	0	GPIO71	IO	CU, CD	2/4/6/8 mA	0
		1	I2S0_MCK	O	CU, CD	2/4/6/8 mA	0
		2	I2S3_MCK	O	CU, CD	2/4/6/8 mA	0
		3	I2S1_MCK	O	CU, CD	2/4/6/8 mA	0
		4	I2S2_MCK	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A4	IO	CU, CD	2/4/6/8 mA	0
I2S0_DI	Y	0	GPIO72	IO	CU, CD	2/4/6/8 mA	0
		1	I2S0_DI	I	CU, CD	2/4/6/8 mA	0
		2	I2S0_DI	I	CU, CD	2/4/6/8 mA	0
		3	I2S2_DI	I	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		4	I2S2_DI	I	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A5	IO	CU, CD	2/4/6/8 mA	0
I2S3_DO	Y	0	GPIO73	IO	CU, CD	2/4/6/8 mA	0
		1	I2S3_DO	O	CU, CD	2/4/6/8 mA	0
		2	I2S3_DO	O	CU, CD	2/4/6/8 mA	0
		3	I2S1_DO	O	CU, CD	2/4/6/8 mA	0
		4	I2S1_DO	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A6	IO	CU, CD	2/4/6/8 mA	0
SCL3	Y	0	GPIO74	IO	CD	-	0
		1	SCL3_0	IO	CD	-	0
		7	AUXIF_CLK1	O	CD	-	0
SDA3	Y	0	GPIO75	IO	CD	-	0
		1	SDA3_0	IO	CD	-	0
		7	AUXIF_ST1	O	CD	-	0
CONN_HRST_B	Y	0	GPIO76	IO	CU, CD	2/4/6/8 mA	0
		1	CONN_HRST_B	O	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT0	IO	CU, CD	2/4/6/8 mA	0
CONN_TOP_CLK	Y	0	GPIO77	IO	CU, CD	2/4/6/8 mA	0
		1	CONN_TOP_CLK	O	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT1	IO	CU, CD	2/4/6/8 mA	0
CONN_TOP_DATA	Y	0	GPIO78	IO	CU, CD	2/4/6/8 mA	0
		1	CONN_TOP_DATA	IO	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT2	IO	CU, CD	2/4/6/8 mA	0
CONN_WB_PTA	Y	0	GPIO79	IO	CU, CD	2/4/6/8 mA	0
		1	CONN_WB_PTA	IO	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT3	IO	CU, CD	2/4/6/8 mA	0
CONN_WF_CTRL0	Y	0	GPIO80	IO	CU, CD	2/4/6/8 mA	0
		1	CONN_WF_HB0	IO	CU, CD	2/4/6/8 mA	0
		7	C2K_EINT0	IO	CU, CD	2/4/6/8 mA	0
CONN_WF_CTRL1	Y	0	GPIO81	IO	CU, CD	2/4/6/8 mA	0
		1	CONN_WF_HB1	IO	CU, CD	2/4/6/8 mA	0
		7	C2K_EINT1	IO	CU, CD	2/4/6/8 mA	0
CONN_WF_CTRL2	Y	0	GPIO82	IO	CU, CD	2/4/6/8 mA	0
		1	CONN_WF_HB2	IO	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		7	MD_CLKM0	O	CU, CD	2/4/6/8 mA	0
CONN_BT_CLK	Y	0	GPIO83	IO	CU, CD	2/4/6/8 mA	0
		1	CONN_BT_CLK	IO	CU, CD	2/4/6/8 mA	0
		7	MD_CLKM1	O	CU, CD	2/4/6/8 mA	0
CONN_BT_DATA	Y	0	GPIO84	IO	CU, CD	2/4/6/8 mA	0
		1	CONN_BT_DATA	IO	CU, CD	2/4/6/8 mA	0
EINT8	Y	0	GPIO85	IO	CU, CD	2/4/6/8 mA	0
		1	EINT8	I	CU, CD	2/4/6/8 mA	0
		2	I2S1_LRCK	O	CU, CD	2/4/6/8 mA	0
		3	I2S2_LRCK	O	CU, CD	2/4/6/8 mA	0
		4	URXD1	I	CU, CD	2/4/6/8 mA	0
		5	MD_URXD0	I	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A7	IO	CU, CD	2/4/6/8 mA	0
EINT9	Y	0	GPIO86	IO	CU, CD	2/4/6/8 mA	0
		1	EINT9	I	CU, CD	2/4/6/8 mA	0
		2	I2S1_BCK	O	CU, CD	2/4/6/8 mA	0
		3	I2S2_BCK	O	CU, CD	2/4/6/8 mA	0
		4	UTXD1	O	CU, CD	2/4/6/8 mA	0
		5	MD_UTXD0	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A8	IO	CU, CD	2/4/6/8 mA	0
EINT10	Y	0	GPIO87	IO	CU, CD	2/4/6/8 mA	0
		1	EINT10	I	CU, CD	2/4/6/8 mA	0
		2	I2S1_MCK	O	CU, CD	2/4/6/8 mA	0
		3	I2S2_MCK	O	CU, CD	2/4/6/8 mA	0
		4	URTS1	O	CU, CD	2/4/6/8 mA	0
		5	MD_URXD1	I	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A9	IO	CU, CD	2/4/6/8 mA	0
EINT11	Y	0	GPIO88	IO	CU, CD	2/4/6/8 mA	0
		1	EINT11	I	CU, CD	2/4/6/8 mA	0
		2	I2S1_DO	O	CU, CD	2/4/6/8 mA	0
		3	I2S2_DI	I	CU, CD	2/4/6/8 mA	0
		4	UCTS1	I	CU, CD	2/4/6/8 mA	0
		5	MD_UTXD1	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A10	IO	CU, CD	2/4/6/8 mA	0
EINT12	Y	0	GPIO89	IO	CU, CD	2/4/6/8 mA	0
		1	EINT12	I	CU, CD	2/4/6/8 mA	0
		2	IRTX_OUT	O	CU, CD	2/4/6/8 mA	0
		3	CLKM0	O	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		4	PCM1_SYNC	IO	CU, CD	2/4/6/8 mA	0
		5	URTS0	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A11	IO	CU, CD	2/4/6/8 mA	0
EINT13	Y	0	GPIO90	IO	CU, CD	2/4/6/8 mA	0
		1	EINT13	I	CU, CD	2/4/6/8 mA	0
		2	CMFLASH	O	CU, CD	2/4/6/8 mA	0
		3	CLKM1	O	CU, CD	2/4/6/8 mA	0
		4	PCM1_CLK	IO	CU, CD	2/4/6/8 mA	0
		5	UCTS0	I	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT0	IO	CU, CD	2/4/6/8 mA	0
EINT14	Y	0	GPIO91	IO	CU, CD	2/4/6/8 mA	0
		1	EINT14	I	CU, CD	2/4/6/8 mA	0
		2	PWM_A	O	CU, CD	2/4/6/8 mA	0
		3	CLKM2	O	CU, CD	2/4/6/8 mA	0
		4	PCM1_DI	I	CU, CD	2/4/6/8 mA	0
		5	SDA0_3	IO	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT1	IO	CU, CD	2/4/6/8 mA	0
EINT15	Y	0	GPIO92	IO	CU, CD	2/4/6/8 mA	0
		1	EINT15	I	CU, CD	2/4/6/8 mA	0
		2	PWM_B	O	CU, CD	2/4/6/8 mA	0
		3	CLKM3	O	CU, CD	2/4/6/8 mA	0
		4	PCM1_DO0	O	CU, CD	2/4/6/8 mA	0
		5	SCL0_3	IO	CU, CD	2/4/6/8 mA	0
EINT16	Y	0	GPIO93	IO	CU, CD	2/4/6/8 mA	0
		1	EINT16	I	CU, CD	2/4/6/8 mA	0
		2	IDDIG	I	CU, CD	2/4/6/8 mA	0
		3	CLKM4	O	CU, CD	2/4/6/8 mA	0
		4	PCM1_DO1	O	CU, CD	2/4/6/8 mA	0
		5	MD_INT2	I	CU, CD	2/4/6/8 mA	0
		7	DROP_ZONE	I	CU, CD	2/4/6/8 mA	0
DRVBUS	Y	0	GPIO94	IO	CU, CD	4/8/12/16 mA	0
		1	USB_DRVVBUS	O	CU, CD	4/8/12/16 mA	0
		2	PWM_C	O	CU, CD	4/8/12/16 mA	0
		3	CLKM5	O	CU, CD	4/8/12/16 mA	0
SDA2	Y	0	GPIO95	IO	CD	-	0
		1	SDA2_0	IO	CD	-	0
		7	AUXIF_ST0	O	CD	-	0
SCL2	Y	0	GPIO96	IO	CD	-	0
		1	SCL2_0	IO	CD	-	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		7	AUXIF_CLK0	O	CD	-	0
URXD0	Y	0	GPIO97	IO	CU, CD	4/8/12/16 mA	0
		1	URXD0	I	CU, CD	4/8/12/16 mA	0
		2	UTXD0	O	CU, CD	4/8/12/16 mA	0
		3	MD_URXD0	I	CU, CD	4/8/12/16 mA	0
		4	MD_URXD1	I	CU, CD	4/8/12/16 mA	0
		5	MD_URXD2	I	CU, CD	4/8/12/16 mA	0
		6	C2K_URXD0	I	CU, CD	4/8/12/16 mA	0
		7	C2K_URXD1	IO	CU, CD	4/8/12/16 mA	0
UTXD0	Y	0	GPIO98	IO	CU, CD	4/8/12/16 mA	0
		1	UTXD0	O	CU, CD	4/8/12/16 mA	0
		2	URXD0	I	CU, CD	4/8/12/16 mA	0
		3	MD_UTXD0	O	CU, CD	4/8/12/16 mA	0
		4	MD_UTXD1	O	CU, CD	4/8/12/16 mA	0
		5	MD_UTXD2	O	CU, CD	4/8/12/16 mA	0
		6	C2K_UTXD0	O	CU, CD	4/8/12/16 mA	0
		7	C2K_UTXD1	IO	CU, CD	4/8/12/16 mA	0
RTC32K_CK	N	0	GPIO99	IO	CU, CD	2/4/6/8 mA	0
		1	RTC32K_CK	I	CU, CD	2/4/6/8 mA	0
SRCLKENAI0	N	0	GPIO100	IO	CU, CD	2/4/6/8 mA	0
		1	SRCLKENAI0	I	CU, CD	2/4/6/8 mA	0
SRCLKENAI1	N	0	GPIO101	IO	CU, CD	2/4/6/8 mA	0
		1	SRCLKENAI1	I	CU, CD	2/4/6/8 mA	0
SRCLKENA0	N	0	GPIO102	IO	CU, CD	2/4/6/8 mA	0
		1	SRCLKENA0	O	CU, CD	2/4/6/8 mA	0
SRCLKENA1	N	0	GPIO103	IO	CU, CD	2/4/6/8 mA	0
		1	SRCLKENA1	O	CU, CD	2/4/6/8 mA	0
SYSRSTB	N	0	GPI104	I	CU, CD	2/4/6/8 mA	0
		1	SYSRSTB	I	CU, CD	2/4/6/8 mA	0
WATCHDOG	N	0	GPIO105	IO	CU, CD	2/4/6/8 mA	0
		1	WATCHDOG	O	CU, CD	2/4/6/8 mA	0
KPROW0	Y	0	GPIO106	IO	CU, CD	2/4/6/8 mA	0
		1	KPROW0	IO	CU, CD	2/4/6/8 mA	0
		2	CMFLASH	O	CU, CD	2/4/6/8 mA	0
		3	CLKM4	O	CU, CD	2/4/6/8 mA	0
		4	TP_GPIO0_AO	IO	CU, CD	2/4/6/8 mA	0
		5	IRTX_OUT	O	CU, CD	2/4/6/8 mA	0
KPROW1	Y	0	GPIO107	IO	CU, CD	2/4/6/8 mA	0
		1	KPROW1	IO	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		2	IDDIG	I	CU, CD	2/4/6/8 mA	0
		3	CLKM5	O	CU, CD	2/4/6/8 mA	0
		4	TP_GPIO1_AO	IO	CU, CD	2/4/6/8 mA	0
		5	I2S1_BCK	O	CU, CD	2/4/6/8 mA	0
		7	DAP_SIB1_SWD	IO	CU, CD	2/4/6/8 mA	0
KPROW2	Y	0	GPIO108	IO	CU, CD	2/4/6/8 mA	0
		1	KPROW2	IO	CU, CD	2/4/6/8 mA	0
		2	USB_DRVVBUS	O	CU, CD	2/4/6/8 mA	0
		3	PWM_A	O	CU, CD	2/4/6/8 mA	0
		4	CMFLASH	O	CU, CD	2/4/6/8 mA	0
		5	I2S1_LRCK	O	CU, CD	2/4/6/8 mA	0
		7	DAP_SIB1_SWCK	I	CU, CD	2/4/6/8 mA	0
KPCOL0	Y	0	GPIO109	IO	CU, CD	2/4/6/8 mA	0
		1	KPCOL0	IO	CU, CD	2/4/6/8 mA	0
KPCOL1	Y	0	GPIO110	IO	CU, CD	2/4/6/8 mA	0
		1	KPCOL1	IO	CU, CD	2/4/6/8 mA	0
		2	SDA1_3	IO	CU, CD	2/4/6/8 mA	0
		3	PWM_B	O	CU, CD	2/4/6/8 mA	0
		4	CLKM0	O	CU, CD	2/4/6/8 mA	0
		5	I2S1_DO	O	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT3	IO	CU, CD	2/4/6/8 mA	0
KPCOL2	Y	0	GPIO111	IO	CU, CD	2/4/6/8 mA	0
		1	KPCOL2	IO	CU, CD	2/4/6/8 mA	0
		2	SCL1_3	IO	CU, CD	2/4/6/8 mA	0
		3	PWM_C	O	CU, CD	2/4/6/8 mA	0
		4	DISP_PWM	O	CU, CD	2/4/6/8 mA	0
		5	I2S1_MCK	O	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT2	IO	CU, CD	2/4/6/8 mA	0
INT_SIM1	Y	0	GPIO112	IO	CU, CD	2/4/6/8 mA	0
		1	MD_INT1_C2K_UIM1_HOT_PLUG_IN	I	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT1	IO	CU, CD	2/4/6/8 mA	0
INT_SIM2	Y	0	GPIO113	IO	CU, CD	2/4/6/8 mA	0
		1	MD_INT0_C2K_UIM0_HOT_PLUG_IN	I	CU, CD	2/4/6/8 mA	0
		7	C2K_DM_EINT0	IO	CU, CD	2/4/6/8 mA	0
MSDC0_DAT0	N	0	GPIO114	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_DAT0	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
MSDC0_DAT1	N	0	GPIO115	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_DAT1	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC0_DAT2	N	0	GPIO116	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_DAT2	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC0_DAT3	N	0	GPIO117	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_DAT3	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC0_DAT4	N	0	GPIO118	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_DAT4	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC0_DAT5	N	0	GPIO119	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_DAT5	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC0_DAT6	N	0	GPIO120	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_DAT6	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC0_DAT7	N	0	GPIO121	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_DAT7	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC0_CMD	N	0	GPIO122	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_CMD	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC0_CLK	N	0	GPIO123	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_CLK	O	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC0_DSL	N	0	GPIO124	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_DSL	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC0_RSTB	N	0	GPIO125	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC0_RSTB	O	CU, CD	2/4/6/8/10/12/14/16 mA	0
SIM1_SCLK	Y	0	GPIO126	IO	CU, CD	4/8/12/16 mA	0
		1	MD1_SIM1_SCLK	O	CU, CD	4/8/12/16 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		2	MD1_SIM2_SCLK	O	CU, CD	4/8/12/16 mA	0
		3	C2K_UIM0_CLK	O	CU, CD	4/8/12/16 mA	0
		4	C2K_UIM1_CLK	O	CU, CD	4/8/12/16 mA	0
SIM1_SRST	Y	0	GPIO127	IO	CU, CD	4/8/12/16 mA	0
		1	MD1_SIM1_SRST	O	CU, CD	4/8/12/16 mA	0
		2	MD1_SIM2_SRST	O	CU, CD	4/8/12/16 mA	0
		3	C2K_UIM0_RST	O	CU, CD	4/8/12/16 mA	0
		4	C2K_UIM1_RST	O	CU, CD	4/8/12/16 mA	0
SIM1_SIO	Y	0	GPIO128	IO	CU, CD	4/8/12/16 mA	0
		1	MD1_SIM1_SIO	IO	CU, CD	4/8/12/16 mA	0
		2	MD1_SIM2_SIO	IO	CU, CD	4/8/12/16 mA	0
		3	C2K_UIM0_IO	IO	CU, CD	4/8/12/16 mA	0
		4	C2K_UIM1_IO	IO	CU, CD	4/8/12/16 mA	0
MSDC1_CMD	Y	0	GPIO129	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC1_CMD	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		2	CONN_DSP_JMS	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
		3	LTE_JTAG_TMS	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		4	UDI_TMS	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
		5	C2K_TMS	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC1_DAT0	Y	0	GPIO130	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC1_DAT0	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		2	CONN_DSP_JDI	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
		3	LTE_JTAG_TDI	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
		4	UDI_TDI	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
		5	C2K_TDI	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC1_DAT1	Y	0	GPIO131	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC1_DAT1	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		2	CONN_DSP_JDO	O	CU, CD	2/4/6/8/10/12/14/16 mA	0
		3	LTE_JTAG_TDO	O	CU, CD	2/4/6/8/10/12/14/	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
						16 mA	
		4	UDI_TDO	O	CU, CD	2/4/6/8/10/12/14/16 mA	0
		5	C2K_TDO	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC1_DAT2	Y	0	GPIO132	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC1_DAT2	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		5	C2K_RTCK	O	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC1_DAT3	Y	0	GPIO133	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC1_DAT3	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		2	CONN_DSP_JINTP	O	CU, CD	2/4/6/8/10/12/14/16 mA	0
		3	LTE_JTAG_TRSTN	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
		4	UDI_NTRST	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
		5	C2K_NTRST	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
MSDC1_CLK	Y	0	GPIO134	IO	CU, CD	2/4/6/8/10/12/14/16 mA	0
		1	MSDC1_CLK	O	CU, CD	2/4/6/8/10/12/14/16 mA	0
		2	CONN_DSP_JCK	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
		3	LTE_JTAG_TCK	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
		4	UDI_TCK_XI	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
		5	C2K_TCK	I	CU, CD	2/4/6/8/10/12/14/16 mA	0
TDM_LRCK	Y	0	GPIO135	IO	CU, CD	2/4/6/8 mA	0
		1	TDM_LRCK	O	CU, CD	2/4/6/8 mA	0
		2	I2S0_LRCK	IO	CU, CD	2/4/6/8 mA	0
		3	CLKM0	O	CU, CD	2/4/6/8 mA	0
		4	PCM1_SYNC	IO	CU, CD	2/4/6/8 mA	0
		5	PWM_A	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A12	IO	CU, CD	2/4/6/8 mA	0
TDM_BCK	Y	0	GPIO136	IO	CU, CD	2/4/6/8 mA	0
		1	TDM_BCK	O	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		2	I2S0_BCK	IO	CU, CD	2/4/6/8 mA	0
		3	CLKM1	O	CU, CD	2/4/6/8 mA	0
		4	PCM1_CLK	IO	CU, CD	2/4/6/8 mA	0
		5	PWM_B	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A13	IO	CU, CD	2/4/6/8 mA	0
TDM_MCK	Y	0	GPIO137	IO	CU, CD	2/4/6/8 mA	0
		1	TDM_MCK	O	CU, CD	2/4/6/8 mA	0
		2	I2S0_MCK	O	CU, CD	2/4/6/8 mA	0
		3	CLKM2	O	CU, CD	2/4/6/8 mA	0
		4	PCM1_DI	I	CU, CD	2/4/6/8 mA	0
		5	IRTX_OUT	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A14	IO	CU, CD	2/4/6/8 mA	0
TDM_DATA0	Y	0	GPIO138	IO	CU, CD	2/4/6/8 mA	0
		1	TDM_DATA0	O	CU, CD	2/4/6/8 mA	0
		2	I2S0_DI	I	CU, CD	2/4/6/8 mA	0
		3	CLKM3	O	CU, CD	2/4/6/8 mA	0
		4	PCM1_DO0	O	CU, CD	2/4/6/8 mA	0
		5	PWM_C	O	CU, CD	2/4/6/8 mA	0
		6	SDA3_1	IO	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A15	IO	CU, CD	2/4/6/8 mA	0
TDM_DATA1	Y	0	GPIO139	IO	CU, CD	2/4/6/8 mA	0
		1	TDM_DATA1	O	CU, CD	2/4/6/8 mA	0
		2	I2S3_DO	O	CU, CD	2/4/6/8 mA	0
		3	CLKM4	O	CU, CD	2/4/6/8 mA	0
		4	PCM1_DO1	O	CU, CD	2/4/6/8 mA	0
		5	ANT_SEL2	O	CU, CD	2/4/6/8 mA	0
		6	SCL3_1	IO	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A16	IO	CU, CD	2/4/6/8 mA	0
TDM_DATA2	Y	0	GPIO140	IO	CU, CD	2/4/6/8 mA	0
		1	TDM_DATA2	O	CU, CD	2/4/6/8 mA	0
		2	DISP_PWM	O	CU, CD	2/4/6/8 mA	0
		3	CLKM5	O	CU, CD	2/4/6/8 mA	0
		4	SDA1_4	IO	CU, CD	2/4/6/8 mA	0
		5	ANT_SEL1	O	CU, CD	2/4/6/8 mA	0
		6	URXD3	I	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A17	IO	CU, CD	2/4/6/8 mA	0
TDM_DATA3	Y	0	GPIO141	IO	CU, CD	2/4/6/8 mA	0
		1	TDM_DATA3	O	CU, CD	2/4/6/8 mA	0
		2	CMFLASH	O	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		3	IRTX_OUT	O	CU, CD	2/4/6/8 mA	0
		4	SCL1_4	IO	CU, CD	2/4/6/8 mA	0
		5	ANT_SEL0	O	CU, CD	2/4/6/8 mA	0
		6	UTXD3	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A18	IO	CU, CD	2/4/6/8 mA	0
PWRAP_SPI0_MI	N	0	GPIO142	IO	CU, CD	2/4/6/8 mA	0
		1	PWRAP_SPI0_MI	IO	CU, CD	2/4/6/8 mA	0
		2	PWRAP_SPI0_MO	IO	CU, CD	2/4/6/8 mA	0
PWRAP_SPI0_MO	N	0	GPIO143	IO	CU, CD	2/4/6/8 mA	0
		1	PWRAP_SPI0_MO	IO	CU, CD	2/4/6/8 mA	0
		2	PWRAP_SPI0_MI	IO	CU, CD	2/4/6/8 mA	0
PWRAP_SPI0_CK	N	0	GPIO144	IO	CU, CD	2/4/6/8 mA	0
		1	PWRAP_SPI0_CK	O	CU, CD	2/4/6/8 mA	0
PWRAP_SPI0_CSN	N	0	GPIO145	IO	CU, CD	2/4/6/8 mA	0
		1	PWRAP_SPI0_CSN	O	CU, CD	2/4/6/8 mA	0
AUD_CLK_MOSI	N	0	GPIO146	IO	CU, CD	2/4/6/8 mA	0
		1	AUD_CLK_MOSI	O	CU, CD	2/4/6/8 mA	0
AUD_DAT_MISO	N	0	GPIO147	IO	CU, CD	2/4/6/8 mA	0
		1	AUD_DAT_MISO	I	CU, CD	2/4/6/8 mA	0
		2	AUD_DAT_MOSI	O	CU, CD	2/4/6/8 mA	0
		3	VOW_DAT_MISO	I	CU, CD	2/4/6/8 mA	0
AUD_DAT_MOSI	N	0	GPIO148	IO	CU, CD	2/4/6/8 mA	0
		1	AUD_DAT_MOSI	O	CU, CD	2/4/6/8 mA	0
		2	AUD_DAT_MISO	I	CU, CD	2/4/6/8 mA	0
VOW_CLK_MISO	N	0	GPIO149	IO	CU, CD	2/4/6/8 mA	0
		1	VOW_CLK_MISO	I	CU, CD	2/4/6/8 mA	0
ANC_DAT_MOSI	N	0	GPIO150	IO	CU, CD	2/4/6/8 mA	0
		1	ANC_DAT_MOSI	O	CU, CD	2/4/6/8 mA	0
SCL6	Y	0	GPIO151	IO	CU, CD	2/4/6/8 mA	0
		1	SCL6_0	IO	CU, CD	2/4/6/8 mA	0
SDA6	Y	0	GPIO152	IO	CU, CD	2/4/6/8 mA	0
		1	SDA6_0	IO	CU, CD	2/4/6/8 mA	0
SCL7	Y	0	GPIO153	IO	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		1	SCL7_0	IO	CU, CD	2/4/6/8 mA	0
SDA7	Y	0	GPIO154	IO	CU, CD	2/4/6/8 mA	0
		1	SDA7_0	IO	CU, CD	2/4/6/8 mA	0
SIM2_SCLK	Y	0	GPIO155	IO	CU, CD	4/8/12/16 mA	0
		1	MD1_SIM2_SCLK	O	CU, CD	4/8/12/16 mA	0
		2	MD1_SIM1_SCLK	O	CU, CD	4/8/12/16 mA	0
		3	C2K_UIM0_CLK	O	CU, CD	4/8/12/16 mA	0
		4	C2K_UIM1_CLK	O	CU, CD	4/8/12/16 mA	0
SIM2_SRST	Y	0	GPIO156	IO	CU, CD	4/8/12/16 mA	0
		1	MD1_SIM2_SRST	O	CU, CD	4/8/12/16 mA	0
		2	MD1_SIM1_SRST	O	CU, CD	4/8/12/16 mA	0
		3	C2K_UIM0_RST	O	CU, CD	4/8/12/16 mA	0
		4	C2K_UIM1_RST	O	CU, CD	4/8/12/16 mA	0
SIM2_SIO	Y	0	GPIO157	IO	CU, CD	4/8/12/16 mA	0
		1	MD1_SIM2_SIO	IO	CU, CD	4/8/12/16 mA	0
		2	MD1_SIM1_SIO	IO	CU, CD	4/8/12/16 mA	0
		3	C2K_UIM0_IO	IO	CU, CD	4/8/12/16 mA	0
		4	C2K_UIM1_IO	IO	CU, CD	4/8/12/16 mA	0
TDP0	N	0	GPI158	I	-	-	-
		1	MIPI_TDP0	AIO	-	-	-
TDN0	N	0	GPI159	I	-	-	-
		1	MIPI_TDN0	AIO	-	-	-
TDP1	N	0	GPI160	I	-	-	-
		1	MIPI_TDP1	AIO	-	-	-
TDN1	N	0	GPI161	I	-	-	-
		1	MIPI_TDN1	AIO	-	-	-
TCP	N	0	GPI162	I	-	-	-
		1	MIPI_TCP	AIO	-	-	-
TCN	N	0	GPI163	I	-	-	-
		1	MIPI_TCN	AIO	-	-	-
TDP2	N	0	GPI164	I	-	-	-
		1	MIPI_TDP2	AIO	-	-	-
TDN2	N	0	GPI165	I	-	-	-
		1	MIPI_TDN2	AIO	-	-	-
TDP3	N	0	GPI166	I	-	-	-
		1	MIPI_TDP3	AIO	-	-	-
TDN3	N	0	GPI167	I	-	-	-
		1	MIPI_TDN3	AIO	-	-	-
TDP0_A	N	0	GPI168	I	-	-	-

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		1	MIPI_TDP0_A	AIO	-	-	-
TDN0_A	N	0	GPI169	I	-	-	-
		1	MIPI_TDN0_A	AIO	-	-	-
TDP1_A	N	0	GPI170	I	-	-	-
		1	MIPI_TDP1_A	AIO	-	-	-
TDN1_A	N	0	GPI171	I	-	-	-
		1	MIPI_TDN1_A	AIO	-	-	-
TCP_A	N	0	GPI172	I	-	-	-
		1	MIPI_TCP_A	AIO	-	-	-
TCN_A	N	0	GPI173	I	-	-	-
		1	MIPI_TCN_A	AIO	-	-	-
TDP2_A	N	0	GPI174	I	-	-	-
		1	MIPI_TDP2_A	AIO	-	-	-
TDN2_A	N	0	GPI175	I	-	-	-
		1	MIPI_TDN2_A	AIO	-	-	-
TDP3_A	N	0	GPI176	I	-	-	-
		1	MIPI_TDP3_A	AIO	-	-	-
TDN3_A	N	0	GPI177	I	-	-	-
		1	MIPI_TDN3_A	AIO	-	-	-
DISP_PWM	N	0	GPIO178	IO	CU, CD	2/4/6/8 mA	0
		1	DISP_PWM	O	CU, CD	2/4/6/8 mA	0
		2	PWM_D	O	CU, CD	2/4/6/8 mA	0
		3	CLKM5	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A19	IO	CU, CD	2/4/6/8 mA	0
DSI_TE	Y	0	GPIO179	IO	CU, CD	2/4/6/8 mA	0
		1	DSI_TE0	I	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A20	IO	CU, CD	2/4/6/8 mA	0
LCM_RST	Y	0	GPIO180	IO	CU, CD	2/4/6/8 mA	0
		1	LCM_RST	O	CU, CD	2/4/6/8 mA	0
		2	DSI_TE1	I	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A21	IO	CU, CD	2/4/6/8 mA	0
IDDIG	Y	0	GPIO181	IO	CU, CD	2/4/6/8 mA	0
		1	IDDIG	I	CU, CD	2/4/6/8 mA	0
		2	DSI_TE1	I	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A22	IO	CU, CD	2/4/6/8 mA	0
TESTMODE	N	0	GPI182	I	CU, CD	2/4/6/8 mA	0
		1	TESTMODE	I	CU, CD	2/4/6/8 mA	0
RFIC0_BSI_CK	Y	0	GPIO183	IO	CU, CD	2/4/6/8 mA	0
		1	RFIC0_BSI_CK	O	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		2	SPM_BSI_CK	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B27	IO	CU, CD	2/4/6/8 mA	0
RFIC0_BSI_EN	Y	0	GPIO184	IO	CU, CD	2/4/6/8 mA	0
		1	RFIC0_BSI_EN	O	CU, CD	2/4/6/8 mA	0
		2	SPM_BSI_EN	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B28	IO	CU, CD	2/4/6/8 mA	0
RFIC0_BSI_D0	Y	0	GPIO185	IO	CU, CD	2/4/6/8 mA	0
		1	RFIC0_BSI_D0	IO	CU, CD	2/4/6/8 mA	0
		2	SPM_BSI_D0	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B29	IO	CU, CD	2/4/6/8 mA	0
RFIC0_BSI_D1	Y	0	GPIO186	IO	CU, CD	2/4/6/8 mA	0
		1	RFIC0_BSI_D1	IO	CU, CD	2/4/6/8 mA	0
		2	SPM_BSI_D1	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B30	IO	CU, CD	2/4/6/8 mA	0
RFIC0_BSI_D2	Y	0	GPIO187	IO	CU, CD	2/4/6/8 mA	0
		1	RFIC0_BSI_D2	IO	CU, CD	2/4/6/8 mA	0
		2	SPM_BSI_D2	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B31	IO	CU, CD	2/4/6/8 mA	0
MISC_MIPI_CK_0	Y	0	GPIO188	IO	CU, CD	2/4/6/8 mA	0
		1	MIPI0_SCLK	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B32	IO	CU, CD	2/4/6/8 mA	0
MISC_MIPI_DO_0	Y	0	GPIO189	IO	CU, CD	2/4/6/8 mA	0
		1	MIPI0_SDATA	IO	CU, CD	2/4/6/8 mA	0
MISC_MIPI_CK_1	Y	0	GPIO190	IO	CU, CD	2/4/6/8 mA	0
		1	MIPI1_SCLK	O	CU, CD	2/4/6/8 mA	0
MISC_MIPI_DO_1	Y	0	GPIO191	IO	CU, CD	2/4/6/8 mA	0
		1	MIPI1_SDATA	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS4	Y	0	GPIO192	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS4	O	CU, CD	2/4/6/8 mA	0
BPI_BUS5	Y	0	GPIO193	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS5	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B0	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS6	Y	0	GPIO194	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS6	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B1	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS7	Y	0	GPIO195	IO	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		1	BPI_BUS7	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B2	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS8	Y	0	GPIO196	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS8	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B3	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS9	Y	0	GPIO197	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS9	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B4	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS10	Y	0	GPIO198	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS10	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B5	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS11	Y	0	GPIO199	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS11	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B6	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS12_ANT0	Y	0	GPIO200	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS12	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B7	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS13_ANT1	Y	0	GPIO201	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS13	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B8	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS14_ANT2	Y	0	GPIO202	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS14	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B9	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS15_ANT3	Y	0	GPIO203	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS15	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B10	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS16_VM0	Y	0	GPIO204	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS16	O	CU, CD	2/4/6/8 mA	0
		2	PA_VM0	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B11	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS17_VM1	Y	0	GPIO205	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS17	O	CU, CD	2/4/6/8 mA	0
		2	PA_VM1	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B12	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS18_S	Y	0	GPIO206	IO	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
WP0							
		1	BPI_BUS18	O	CU, CD	2/4/6/8 mA	0
		2	TX_SWAP0	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B13	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS19_SWP1	Y	0	GPIO207	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS19	O	CU, CD	2/4/6/8 mA	0
		2	TX_SWAP1	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B14	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS20_SWP2	Y	0	GPIO208	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS20	O	CU, CD	2/4/6/8 mA	0
		2	TX_SWAP2	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B15	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS21_SWP3	Y	0	GPIO209	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS21	O	CU, CD	2/4/6/8 mA	0
		2	TX_SWAP3	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B16	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS22_DET0	Y	0	GPIO210	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS22	O	CU, CD	2/4/6/8 mA	0
		2	DET_BPI0	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B17	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS23_DET1	Y	0	GPIO211	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS23	O	CU, CD	2/4/6/8 mA	0
		2	DET_BPI1	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B18	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS0	Y	0	GPIO212	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS0	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B19	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS1	Y	0	GPIO213	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS1	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B20	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS2	Y	0	GPIO214	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS2	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B21	IO	CU, CD	2/4/6/8 mA	0
BPI_BUS3	Y	0	GPIO215	IO	CU, CD	2/4/6/8 mA	0
		1	BPI_BUS3	O	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		7	DBG_MON_B22	IO	CU, CD	2/4/6/8 mA	0
MISC_MIPI_CK_2	Y	0	GPIO216	IO	CU, CD	2/4/6/8 mA	0
		1	MIPI2_SCLK	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B23	IO	CU, CD	2/4/6/8 mA	0
MISC_MIPI_DO_2	Y	0	GPIO217	IO	CU, CD	2/4/6/8 mA	0
		1	MIPI2_SDATA	IO	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B24	IO	CU, CD	2/4/6/8 mA	0
MISC_MIPI_CK_3	Y	0	GPIO218	IO	CU, CD	2/4/6/8 mA	0
		1	MIPI3_SCLK	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B25	IO	CU, CD	2/4/6/8 mA	0
MISC_MIPI_DO_3	Y	0	GPIO219	IO	CU, CD	2/4/6/8 mA	0
		1	MIPI3_SDATA	IO	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_B26	IO	CU, CD	2/4/6/8 mA	0
WF_IP	N	1	CONN_WF_IP	AIO	-	-	-
WF_IN	N	1	CONN_WF_IN	AIO	-	-	-
WF_QP	N	1	CONN_WF_QP	AIO	-	-	-
WF_QN	N	1	CONN_WF_QN	AIO	-	-	-
BT_IP	N	1	CONN_BT_IP	AIO	-	-	-
BT_IN	N	1	CONN_BT_IN	AIO	-	-	-
BT_QP	N	1	CONN_BT_QP	AIO	-	-	-
BT_QN	N	1	CONN_BT_QN	AIO	-	-	-
GPS_IP	N	1	CONN_GPS_IP	AIO	-	-	-
GPS_IN	N	1	CONN_GPS_IN	AIO	-	-	-
GPS_QP	N	1	CONN_GPS_QP	AIO	-	-	-
GPS_QN	N	1	CONN_GPS_QN	AIO	-	-	-
URXD1	Y	0	GPIO232	IO	CU, CD	4/8/12/16 mA	0
		1	URXD1	I	CU, CD	4/8/12/16 mA	0
		2	UTXD1	O	CU, CD	4/8/12/16 mA	0
		3	MD_URXD0	I	CU, CD	4/8/12/16 mA	0
		4	MD_URXD1	I	CU, CD	4/8/12/16 mA	0
		5	MD_URXD2	I	CU, CD	4/8/12/16 mA	0
		6	C2K_URXD0	I	CU, CD	4/8/12/16 mA	0
		7	C2K_URXD1	IO	CU, CD	4/8/12/16 mA	0
UTXD1	Y	0	GPIO233	IO	CU, CD	4/8/12/16 mA	0
		1	UTXD1	O	CU, CD	4/8/12/16 mA	0
		2	URXD1	I	CU, CD	4/8/12/16 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		3	MD_UTXD0	O	CU, CD	4/8/12/16 mA	0
		4	MD_UTXD1	O	CU, CD	4/8/12/16 mA	0
		5	MD_UTXD2	O	CU, CD	4/8/12/16 mA	0
		6	C2K_UTXD0	O	CU, CD	4/8/12/16 mA	0
		7	C2K_UTXD1	IO	CU, CD	4/8/12/16 mA	0
SPI1_CK	Y	0	GPIO234	IO	CU, CD	2/4/6/8 mA	0
		1	SPI1_CLK_B	O	CU, CD	2/4/6/8 mA	0
		2	TP_UTXD1_AO	O	CU, CD	2/4/6/8 mA	0
		3	SCL4_1	IO	CU, CD	2/4/6/8 mA	0
		4	UTXD0	O	CU, CD	2/4/6/8 mA	0
		6	PWM_A	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A23	IO	CU, CD	2/4/6/8 mA	0
SPI1_MI	Y	0	GPIO235	IO	CU, CD	2/4/6/8 mA	0
		1	SPI1_MI_B	I	CU, CD	2/4/6/8 mA	0
		2	SPI1_MO_B	O	CU, CD	2/4/6/8 mA	0
		3	SDA4_1	IO	CU, CD	2/4/6/8 mA	0
		4	URXD0	I	CU, CD	2/4/6/8 mA	0
		6	CLKM0	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A24	IO	CU, CD	2/4/6/8 mA	0
SPI1_MO	Y	0	GPIO236	IO	CU, CD	2/4/6/8 mA	0
		1	SPI1_MO_B	O	CU, CD	2/4/6/8 mA	0
		2	SPI1_MI_B	I	CU, CD	2/4/6/8 mA	0
		3	SCL5_1	IO	CU, CD	2/4/6/8 mA	0
		4	URTS0	O	CU, CD	2/4/6/8 mA	0
		6	PWM_B	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A25	IO	CU, CD	2/4/6/8 mA	0
SPI1_CS	Y	0	GPIO237	IO	CU, CD	2/4/6/8 mA	0
		1	SPI1_CS_B	O	CU, CD	2/4/6/8 mA	0
		2	TP_URXD1_AO	I	CU, CD	2/4/6/8 mA	0
		3	SDA5_1	IO	CU, CD	2/4/6/8 mA	0
		4	UCTS0	I	CU, CD	2/4/6/8 mA	0
		6	CLKM1	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A26	IO	CU, CD	2/4/6/8 mA	0
SDA4	Y	0	GPIO238	IO	CD	-	0
		1	SDA4_0	IO	CD	-	0
SCL4	Y	0	GPIO239	IO	CD	-	0
		1	SCL4_0	IO	CD	-	0
SDA5	Y	0	GPIO240	IO	CD	-	0
		1	SDA5_0	IO	CD	-	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
SCL5	Y	0	GPIO241	IO	CD	-	0
		1	SCL5_0	IO	CD	-	0
SPI2_CK	Y	0	GPIO242	IO	CU, CD	2/4/6/8 mA	0
		1	SPI2_CLK_B	O	CU, CD	2/4/6/8 mA	0
		2	TP_UTXD2_AO	O	CU, CD	2/4/6/8 mA	0
		3	SCL4_2	IO	CU, CD	2/4/6/8 mA	0
		4	UTXD1	O	CU, CD	2/4/6/8 mA	0
		5	URTS3	O	CU, CD	2/4/6/8 mA	0
		6	PWM_C	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A27	IO	CU, CD	2/4/6/8 mA	0
SPI2_MI	Y	0	GPIO243	IO	CU, CD	2/4/6/8 mA	0
		1	SPI2_MI_B	I	CU, CD	2/4/6/8 mA	0
		2	SPI2_MO_B	O	CU, CD	2/4/6/8 mA	0
		3	SDA4_2	IO	CU, CD	2/4/6/8 mA	0
		4	URXD1	I	CU, CD	2/4/6/8 mA	0
		5	UCTS3	I	CU, CD	2/4/6/8 mA	0
		6	CLKM2	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A28	IO	CU, CD	2/4/6/8 mA	0
SPI2_MO	Y	0	GPIO244	IO	CU, CD	2/4/6/8 mA	0
		1	SPI2_MO_B	O	CU, CD	2/4/6/8 mA	0
		2	SPI2_MI_B	I	CU, CD	2/4/6/8 mA	0
		3	SCL5_2	IO	CU, CD	2/4/6/8 mA	0
		4	URTS1	O	CU, CD	2/4/6/8 mA	0
		5	UTXD3	O	CU, CD	2/4/6/8 mA	0
		6	PWM_D	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A29	IO	CU, CD	2/4/6/8 mA	0
SPI2_CS	Y	0	GPIO245	IO	CU, CD	2/4/6/8 mA	0
		1	SPI2_CS_B	O	CU, CD	2/4/6/8 mA	0
		2	TP_URXD2_AO	I	CU, CD	2/4/6/8 mA	0
		3	SDA5_2	IO	CU, CD	2/4/6/8 mA	0
		4	UCTS1	I	CU, CD	2/4/6/8 mA	0
		5	URXD3	I	CU, CD	2/4/6/8 mA	0
		6	CLKM3	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A30	IO	CU, CD	2/4/6/8 mA	0
I2S1_LRCK	Y	0	GPIO246	IO	CU, CD	2/4/6/8 mA	0
		1	I2S1_LRCK	O	CU, CD	2/4/6/8 mA	0
		2	I2S2_LRCK	O	CU, CD	2/4/6/8 mA	0
		3	I2S0_LRCK	IO	CU, CD	2/4/6/8 mA	0
		4	I2S3_LRCK	O	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		5	PCM0_SYNC	O	CU, CD	2/4/6/8 mA	0
		6	SPI5_CLK_C	O	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A31	IO	CU, CD	2/4/6/8 mA	0
I2S1_BCK	Y	0	GPIO247	IO	CU, CD	2/4/6/8 mA	0
		1	I2S1_BCK	O	CU, CD	2/4/6/8 mA	0
		2	I2S2_BCK	O	CU, CD	2/4/6/8 mA	0
		3	I2S0_BCK	IO	CU, CD	2/4/6/8 mA	0
		4	I2S3_BCK	O	CU, CD	2/4/6/8 mA	0
		5	PCM0_CLK	O	CU, CD	2/4/6/8 mA	0
		6	SPI5_MI_C	I	CU, CD	2/4/6/8 mA	0
		7	DBG_MON_A32	IO	CU, CD	2/4/6/8 mA	0
I2S2_DI	Y	0	GPIO248	IO	CU, CD	2/4/6/8 mA	0
		1	I2S2_DI	I	CU, CD	2/4/6/8 mA	0
		2	I2S2_DI	I	CU, CD	2/4/6/8 mA	0
		3	I2S0_DI	I	CU, CD	2/4/6/8 mA	0
		4	I2S0_DI	I	CU, CD	2/4/6/8 mA	0
		5	PCM0_DI	I	CU, CD	2/4/6/8 mA	0
		6	SPI5_CS_C	O	CU, CD	2/4/6/8 mA	0
I2S1_DO	Y	0	GPIO249	IO	CU, CD	2/4/6/8 mA	0
		1	I2S1_DO	O	CU, CD	2/4/6/8 mA	0
		2	I2S1_DO	O	CU, CD	2/4/6/8 mA	0
		3	I2S3_DO	O	CU, CD	2/4/6/8 mA	0
		4	I2S3_DO	O	CU, CD	2/4/6/8 mA	0
		5	PCM0_DO	O	CU, CD	2/4/6/8 mA	0
		6	SPI5_MO_C	O	CU, CD	2/4/6/8 mA	0
		7	TRAP_SRAM_PWR_BYPASS	I	CU, CD	2/4/6/8 mA	0
SPI3_MI	Y	0	GPIO250	IO	CU, CD	2/4/6/8 mA	0
		1	SPI3_MI	I	CU, CD	2/4/6/8 mA	0
		2	SPI3_MO	O	CU, CD	2/4/6/8 mA	0
		3	IRTX_OUT	O	CU, CD	2/4/6/8 mA	0
		6	TP_URXD1_AO	I	CU, CD	2/4/6/8 mA	0
		7	DROP_ZONE	I	CU, CD	2/4/6/8 mA	0
SPI3_MO	Y	0	GPIO251	IO	CU, CD	2/4/6/8 mA	0
		1	SPI3_MO	O	CU, CD	2/4/6/8 mA	0
		2	SPI3_MI	I	CU, CD	2/4/6/8 mA	0
		3	CMFLASH	O	CU, CD	2/4/6/8 mA	0
		6	TP_UTXD1_AO	O	CU, CD	2/4/6/8 mA	0
		7	C2K_RTCK	O	CU, CD	2/4/6/8 mA	0
SPI3_CK	Y	0	GPIO252	IO	CU, CD	2/4/6/8 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		1	SPI3_CLK	O	CU, CD	2/4/6/8 mA	0
		2	SCL0_4	IO	CU, CD	2/4/6/8 mA	0
		3	PWM_D	O	CU, CD	2/4/6/8 mA	0
		7	C2K_TMS	I	CU, CD	2/4/6/8 mA	0
SPI3_CS	Y	0	GPIO253	IO	CU, CD	2/4/6/8 mA	0
		1	SPI3_CS	O	CU, CD	2/4/6/8 mA	0
		2	SDA0_4	IO	CU, CD	2/4/6/8 mA	0
		3	PWM_A	O	CU, CD	2/4/6/8 mA	0
		7	C2K_TCK	I	CU, CD	2/4/6/8 mA	0
I2S1_MCK	Y	0	GPIO254	IO	CU, CD	2/4/6/8 mA	0
		1	I2S1_MCK	O	CU, CD	2/4/6/8 mA	0
		2	I2S2_MCK	O	CU, CD	2/4/6/8 mA	0
		3	I2S0_MCK	O	CU, CD	2/4/6/8 mA	0
		4	I2S3_MCK	O	CU, CD	2/4/6/8 mA	0
		5	CLKM0	O	CU, CD	2/4/6/8 mA	0
		7	C2K_TDI	I	CU, CD	2/4/6/8 mA	0
AUD_INTN	Y	0	GPIO255	IO	CU, CD	2/4/6/8 mA	0
		1	CLKM1	O	CU, CD	2/4/6/8 mA	0
		2	DISP_PWM	O	CU, CD	2/4/6/8 mA	0
		3	PWM_B	O	CU, CD	2/4/6/8 mA	0
		6	TP_GPIO1_AO	IO	CU, CD	2/4/6/8 mA	0
		7	C2K_TDO	IO	CU, CD	2/4/6/8 mA	0
AUD_PDN	Y	0	GPIO256	IO	CU, CD	2/4/6/8 mA	0
		1	CLKM2	O	CU, CD	2/4/6/8 mA	0
		2	IRTX_OUT	O	CU, CD	2/4/6/8 mA	0
		3	PWM_C	O	CU, CD	2/4/6/8 mA	0
		6	TP_GPIO0_AO	IO	CU, CD	2/4/6/8 mA	0
		7	C2K_NTRST	I	CU, CD	2/4/6/8 mA	0
JTMS	Y	0	GPIO257	IO	CU, CD	4/8/12/16 mA	0
		1	IO_JTAG_TMS	IO	CU, CD	4/8/12/16 mA	0
		2	LTE_JTAG_TMS	IO	CU, CD	4/8/12/16 mA	0
		3	DFD_TMS	I	CU, CD	4/8/12/16 mA	0
		4	DAP_SIB1_SWD	IO	CU, CD	4/8/12/16 mA	0
		5	ANC_JTAG_TMS	I	CU, CD	4/8/12/16 mA	0
		6	SCP_JTAG_TMS	IO	CU, CD	4/8/12/16 mA	0
		7	C2K_DM_OTMS	I	CU, CD	4/8/12/16 mA	0
JTCK	Y	0	GPIO258	IO	CU, CD	4/8/12/16 mA	0
		1	IO_JTAG_TCK	I	CU, CD	4/8/12/16 mA	0
		2	LTE_JTAG_TCK	I	CU, CD	4/8/12/16 mA	0

Name	EINT SRC	Aux. function	Aux. name	Aux. type	PU/PD/CU/CD	Driving	SMT
		3	DFD_TCK_XI	I	CU, CD	4/8/12/16 mA	0
		4	DAP_SIB1_SWCK	I	CU, CD	4/8/12/16 mA	0
		5	ANC_JTAG_TCK	I	CU, CD	4/8/12/16 mA	0
		6	SCP_JTAG_TCK	I	CU, CD	4/8/12/16 mA	0
		7	C2K_DM_OTCK	I	CU, CD	4/8/12/16 mA	0
JTDI	Y	0	GPIO259	IO	CU, CD	4/8/12/16 mA	0
		1	IO_JTAG_TDI	I	CU, CD	4/8/12/16 mA	0
		2	LTE_JTAG_TDI	I	CU, CD	4/8/12/16 mA	0
		3	DFD_TDI	I	CU, CD	4/8/12/16 mA	0
		5	ANC_JTAG_TDI	I	CU, CD	4/8/12/16 mA	0
		6	SCP_JTAG_TDI	I	CU, CD	4/8/12/16 mA	0
		7	C2K_DM_OTDI	I	CU, CD	4/8/12/16 mA	0
JTDO	Y	0	GPIO260	IO	CU, CD	4/8/12/16 mA	0
		1	IO_JTAG_TDO	O	CU, CD	4/8/12/16 mA	0
		2	LTE_JTAG_TDO	O	CU, CD	4/8/12/16 mA	0
		3	DFD_TDO	O	CU, CD	4/8/12/16 mA	0
		5	ANC_JTAG_TDO	IO	CU, CD	4/8/12/16 mA	0
		6	SCP_JTAG_TDO	O	CU, CD	4/8/12/16 mA	0
		7	C2K_DM_OTDO	O	CU, CD	4/8/12/16 mA	0
JTRST_B	Y	0	GPIO261	IO	CU, CD	4/8/12/16 mA	0
		2	LTE_JTAG_TRSTN	I	CU, CD	4/8/12/16 mA	0
		3	DFD_NTRST	I	CU, CD	4/8/12/16 mA	0
		5	ANC_JTAG_TRSTN	I	CU, CD	4/8/12/16 mA	0
		6	SCP_JTAG_TRSTN	I	CU, CD	4/8/12/16 mA	0
		7	C2K_DM_JTINTP	O	CU, CD	4/8/12/16 mA	0

2.2 Electrical Characteristic

2.2.1 Absolute Maximum Ratings

Table 2-8. Absolute maximum ratings for power supply

Symbol or pin name	Description	Min.	Max.	Unit
DVDD_CORE	Digital power input for CORE	0.594	1.155	V
DVDD_SRAM_CORE	Digital power input for CORE SRAM	0.810	1.320	V
DVDD_PSMCU	Digital power input for PSMCU	0.765	1.210	V
DVDD_MODEM	Digital power input for MODEM	0.850	1.100	V
DVDD_MD1	Digital power input for MD1	0.810	0.990	V
DVDD_SRAM_MD	Digital power input for MD SRAM	0.810	1.210	V
DVDD_GPU	Digital power input for GPU	0.8	1.125	V
DVDD_SRAM_GPU	Digital power input for GPU SRAM	1.620	1.980	V
DVDD_PROC1	Digital power input for CPU PROC1	0.720	1.198	V
DVDD_SRAM_PROC1	Digital power input for CPU PROC1 SRAM	1.620	1.980	V
DVDD_PROC2	Digital power input for CPU PROC2	0.600	1.310	V
DVDD_SRAM_PROC2	Digital power input for CPU PROC2 SRAM	1.620	1.980	V
VDDQ	VDD1 of DRAM die	1.140	1.300	V
VDD1	VDDQ of DRAM die	1.700	1.900	V
AVDD18_AP	Analog power input 1.8V for AuxADC, TSENSE	1.700	1.900	V
AVDD18_MD	Analog power input 1.8V for BBTX, BBRX	1.700	1.900	V
AVDD28_DAC	Analog power input 2.8V for APC	2.660	2.940	V
AVDD18_MCU1PLLGP	Analog power input 1.8V for PLL	1.700	1.900	V
AVDD18_MDPLLGP				
AVDD18_SYSPLLGP				
AVDD18_CSI	Analog power for MIPI CSI	1.700	1.900	V
AVDD18_MIPITX0	Analog power for MIPI DSI	1.700	1.900	V
AVDD18_MIPITX1				
AVDD18_SSUSB	Analog power 1.8V for SSUSB	1.700	1.900	V
AVDD10_SSUSB	Analog power 1.0V for SSUSB	0.900	1.000	V
AVDD18_USB	Analog power 1.8V for USB	1.700	1.900	V
AVDD33_USB_P0	Analog power 3.3V for USB	3.135	3.465	V
AVDD33_USB_P1				
AVDD18_WBG	Analog power 1.8V for connectivity ABB	1.700	1.900	V
DVDD18_IO0	Digital power input for 1.8V IO	1.620	1.980	V
DVDD18_IO1				
DVDD18_IO2				
DVDD18_IO3				
DVDD18_IO4				

Symbol or pin name	Description	Min.	Max.	Unit
DVDD18_MSDC0	Digital power 1.8V input for MSDC0	1.700	1.900	V
DVDD18_MSDC1	Digital power 1.8V input for MSDC1	1.700	1.900	V
DVDD28_MSDC1	Digital power 1.8/2.8V input for MSDC1	1.700	3.600	V
DVDD18_SIM	Digital power 1.8V input for SIM1	1.700	1.900	V
DVDD28_SIM	Digital power 1.8/2.8V input for SIM1	1.700	3.600	V
DVDD18_SIM2	Digital power 1.8V input for SIM2	1.700	1.900	V
DVDD28_SIM2	Digital power 1.8/2.8V input for SIM2	1.700	3.600	V
AVDD08_RDDR_AB	Analog power 1.0V for DDRPHY (Note1)	0.810	1.155	V
AVDD08_RDDR_CA				
AVDD08_RDDR_CB				
AVDD15_ARDDR_B02	Analog power 1.2V for CH_A DDRPHY	1.140	1.3	V
AVDD15_ARDDR_B02				
AVDD15_ARDDR_B13				
AVDD15_ARDDR_B13				
AVDD15_ARDDR_CA				
AVDD15_ARDDR_CA				
AVDD15_BRDDR_B02	Analog power 1.2V for CH_B DDRPHY	1.140	1.3	V
AVDD15_BRDDR_B02				
AVDD15_BRDDR_B13				
AVDD15_BRDDR_B13				
AVDD15_BRDDR_CB				
AVDD15_BRDDR_CB				
AVDD18_RDDR_AB	Analog power 1.8V for DDRPHY	1.700	1.900	V
AVDD18_RDDR_CA				
AVDD18_RDDR_CB				

***Warning**: Stressing the device beyond the absolute maximum ratings may cause permanent damage. These are stress ratings only.*
Note1: The power supply AVDD08_RDDR_AB, AVDD08_RDDR_CA, AVDD08_RDDR_CB should same as DVDD_CORE.

2.2.2 Recommended Operating Conditions

Table 2-9. Recommended operating conditions for power supply

Symbol or pin name	Description	Min.	Typ.	Max.	Unit
DVDD_CORE	Digital power input for CORE	0.594	1.000	1.155	V
DVDD_SRAM_CORE	Digital power input for CORE SRAM	0.810	1.200	1.320	V
DVDD_PSMCU	Digital power input for PSMCU	0.765	1.100	1.210	V
DVDD_MODEM	Digital power input for MODEM	0.72	0.9	1.1	V
DVDD_MD1	Digital power input for MD1	0.72	0.9	0.99	V
DVDD_SRAM_MD	Digital power input for MD SRAM	0.765	1.000	1.21	V

Symbol or pin name	Description	Min.	Typ.	Max.	Unit
DVDD_GPU	Digital power input for GPU	0.80	1.1	1.125	V
DVDD_SRAM_GPU	Digital power input for GPU SRAM	1.620	1.800	1.980	V
DVDD_PROC1	Digital power input for CPU PROC1	0.720	1.180	1.198	V
DVDD_SRAM_PROC1	Digital power input for CPU PROC1 SRAM	1.620	1.800	1.980	V
DVDD_PROC2	Digital power input for CPU PROC2	0.600	1.200	1.310	V
DVDD_SRAM_PROC2	Digital power input for CPU PROC2 SRAM	1.620	1.800	1.980	V
VDDQ	VDD1 of DRAM die	1.140	1.200	1.300	V
VDD1	VDDQ of DRAM die	1.700	1.800	1.900	V
AVDD18_AP	Analog power input 1.8V for AUXADC, TSENSE	1.700	1.800	1.900	V
AVDD18_MD	Analog power input 1.8V for BBTX, BBRX	1.700	1.800	1.900	V
AVDD28_DAC	Analog power input 2.8V for APC	2.660	2.800	2.940	V
AVDD18_MCU1PLLGP	Analog power input 1.8V for PLL	1.700	1.800	1.900	V
AVDD18_MDPLLGP					
AVDD18_SYSPLLGP					
AVDD18_CSI	Analog power for MIPI CSI	1.700	1.800	1.900	V
AVDD18_MIPITX0	Analog power for MIPI DSI	1.700	1.800	1.900	V
AVDD18_MIPITX1					
AVDD18_SSUSB	Analog power 1.8V for SSUSB	1.700	1.800	1.900	V
AVDD10_SSUSB	Analog power 1.0V for SSUSB	0.900	0.950	1.000	V
AVDD18_USB	Analog power 1.8V for USB	1.700	1.800	1.900	V
AVDD33_USB_P0	Analog power 3.3V for USB	3.135	3.300	3.465	V
AVDD33_USB_P1					
AVDD18_WBG	Analog power 1.8V for connectivity ABB	1.700	1.800	1.900	V
DVDD18_IO0	Digital power input for 1.8V IO	1.620	1.800	1.980	V
DVDD18_IO1					
DVDD18_IO2					
DVDD18_IO3					
DVDD18_IO4					
DVDD18_MSDC0	Digital power 1.8V input for MSDC0	1.700	1.800	1.900	V
DVDD18_MSDC1	Digital power 1.8V input for MSDC1	1.700	1.800	1.900	V
DVDD28_MSDC1	Digital power 1.8/2.8V input for MSDC1	1.700	3.000	3.600	V
DVDD18_SIM	Digital power 1.8V input for SIM1	1.700	1.800	1.900	V
DVDD28_SIM	Digital power 1.8/2.8V input for SIM1	1.700	3.000	3.600	V
DVDD18_SIM2	Digital power 1.8V input for SIM2	1.700	1.800	1.900	V
DVDD28_SIM2	Digital power 1.8/2.8V input for SIM2	1.700	3.000	3.600	V
AVDD08_RDDR_AB	Analog power 1.0V for DDRPHY(Note1)	0.810	1.000	1.155	V
AVDD08_RDDR_CA					
AVDD08_RDDR_CB					
AVDD15_ARDDR_B02	Analog power 1.2V for CH_A DDRPHY	1.140	1.200	1.300	V

Symbol or pin name	Description	Min.	Typ.	Max.	Unit
AVDD15_ARDDR_B02					
AVDD15_ARDDR_B13					
AVDD15_ARDDR_B13					
AVDD15_ARDDR_CA					
AVDD15_ARDDR_CA					
AVDD15_BRDDR_B02	Analog power 1.2V for CH_B DDRPHY	1.140	1.200	1.300	V
AVDD15_BRDDR_B02					
AVDD15_BRDDR_B13					
AVDD15_BRDDR_B13					
AVDD15_BRDDR_CB					
AVDD15_BRDDR_CB					
AVDD18_RDDR_AB	Analog power 1.8V for DDRPHY	1.700	1.800	1.900	V
AVDD18_RDDR_CA					
AVDD18_RDDR_CB					

Note 1: The power supply AVDD08_RDDR_AB, AVDD08_RDDR_CA and AVDD08_RDDR_CB should be the same as DVDD_CORE.

2.2.3 Storage Condition

1. Shelf life in sealed bag: 12 months at < 40°C and < 90% relative humidity (RH).
2. After the bag is opened, devices subjected to infrared reflow, vapor-phase reflow or equivalent processing must be:
   - Mounted within 168 hours in factory condition of 30°C/60% RH, or
   - Stored at 20% RH
3. Devices require baking before being mounted, if they are placed
   - For 192 hours at 40°C +5°C/-0°C and < 5% RH in low temperature device containers, or
   - For 24 hours at 125°C +5°C/-0°C in high temperature device containers.

2.2.4 RTC, I2C1, I2C6, I2C7 DC Electrical Characteristics

Table 2-10. RTC, I2C1, I2C6, I2C7 DC electrical characteristics (DVDD18_IO3 =1.8V)

Parameters	Descriptions	Min.	Typ.	Max.	Unit
VIH	Input logic low voltage	0.65*DVDD18_IO3		DVDD18_IO3 + 0.3	V
VIL	Input logic high voltage	-0.3		0.35*DVDD18_IO3	V
VOH	DC output logic low voltage	0.75*DVDD18_IO3			V
VOL	DC output logic high voltage			0.25*DVDD18_IO3	V
F_{RTC}	Input clock frequency		32		KHz
DC_{RTC}	Input signal duty cycle	45	50	55	%

2.2.4.1 SPI, I2S, I2C0 DC Electrical Characteristics

Table 2-11. SPI, I2S, I2C0 electrical characteristics (DVDD18_IO2 =1.8V)

Parameters	Descriptions	Min.	Typ.	Max.	Unit
VIH	Input logic low voltage	0.65*DVDD18_IO2		DVDD18_IO2 + 0.3	V
VIL	Input logic high voltage	-0.3		0.35*DVDD18_IO2	V
VOH	DC output logic low voltage	0.75*DVDD18_IO2			V
VOL	DC Output logic high voltage			0.25*DVDD18_IO2	V

2.2.4.2 I2C2, I2C3 DC Electrical Characteristics

Table 2-12. I2C2, I2C3 DC electrical characteristics (DVDD18_IO0 =1.8V)

Parameters	Descriptions	Min.	Typ.	Max.	Unit
VIH	Input logic low voltage	0.65*DVDD18_IO0		DVDD18_IO0 + 0.3	V
VIL	Input logic high voltage	-0.3		0.35*DVDD18_IO0	V
VOL	DC output logic high voltage			0.2*DVDD18_IO0	V

2.2.4.3 GPIO DC Electrical Characteristics

Table 2-13. GPIO electrical characteristics (DVDD18_IO =1.8V)

Parameters	Descriptions	Min.	Typ.	Max.	Unit
VIH	Input logic low voltage	0.65*DVDD18_IO		DVDD18_IO + 0.3	V
VIL	Input logic high voltage	-0.3		0.35*DVDD18_IO	V
VOH	DC output logic low voltage	0.75*DVDD18_IO			V
VOL	DC Output logic high voltage			0.25*DVDD18_IO	V

2.2.4.4 MSDC0 DC Electrical Characteristics

Table 2-14. MSDC0 DC electrical characteristics (DVDD28_MSDC0=1.8V)

Parameters	Descriptions	Min.	Typ.	Max.	Unit
VIH	Input logic low voltage	1.3		2.0	V
VIL	Input logic high voltage	-0.3		0.58	V
VOH	DC output logic low voltage	1.4			V
VOL	DC output logic high voltage			0.45	V

2.2.4.5 MSDC1 DC Electrical Characteristics

Table 2-15. MSDC1 DC electrical characteristics (DVDD28_MSDC1=2.8V/3.3V)

Parameters	Descriptions	Min.	Typ.	Max.	Unit
VIH	Input logic low voltage	0.625* DVDD28_MSDC1		DVDD28_MSDC1 + 0.3	V
VIL	Input logic high voltage	-0.3		0.25* DVDD28_MSDC1	V
VOH	DC output logic low voltage	0.75* DVDD28_MSDC1		DVDD28_MSDC1 + 0.3	V
VOL	DC output logic high voltage	-0.3		0.125* DVDD28_MSDC1	V

Table 2-16. MSDC1 DC electrical characteristics (DVDD28_MSDC1=1.8V)

Parameters	Descriptions	Min	Typ	Max	Unit
VIH	Input logic low voltage	1.27		DVDD28_MSDC1 + 0.3	V
VIL	Input logic high voltage	-0.3		0.58	V
VOH	DC output logic low voltage	1.4		DVDD28_MSDC1 + 0.3	V
VOL	DC output logic high voltage	-0.3		0.45	V

2.2.4.6 SIM DC Electrical Characteristics

Table 2-17. SIM DC electrical characteristics

Parameter	Conditions	Symbol	Min.	Typ.	Max.	Unit
SIM1_SIO						
Input high voltage	DVDD28_SIM1 = 1.8V	V_{ih}	1.4	1.8	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.27	V
Output high voltage		V_{oh}	1.4	1.8	1.9	V
Output low voltage		V_{ol}	N/A	0.0	0.27	V
Input high voltage	DVDD28_SIM1 = 3.0V	V_{ih}	2.6	3.0	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.4	V
Output high voltage		V_{oh}	2.6	3.0	3.1	V
Output low voltage		V_{ol}	N/A	0.0	0.4	V
SIM1_SCLK						
Input high voltage	DVDD28_SIM1 = 1.8V	V_{ih}	1.4	1.8	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.27	V
Output high voltage		V_{oh}	1.62	1.8	1.9	V
Output low voltage		V_{ol}	N/A	0.0	0.22	V
Input high voltage	DVDD28_SIM1 = 3.0V	V_{ih}	2.6	3.0	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.4	V
Output high voltage		V_{oh}	2.7	3.0	3.1	V

Parameter	Conditions	Symbol	Min.	Typ.	Max.	Unit
Output low voltage		V_{ol}	N/A	0.0	0.4	V
SIM1_SRST						
Input high voltage	DVDD28_SIM1 = 1.8V	V_{ih}	1.4	1.8	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.27	V
Output high voltage		V_{oh}	1.62	1.8	1.9	V
Output low voltage		V_{ol}	N/A	0.0	0.36	V
Input high voltage	DVDD28_SIM1 = 3.0V	V_{ih}	2.6	3.0	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.4	V
Output high voltage		V_{oh}	2.7	3.0	3.1	V
Output low voltage		V_{ol}	N/A	0.0	0.36	V
SIM2_SIO						
Input high voltage	DVDD28_SIM2 = 1.8V	V_{ih}	1.4	1.8	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.27	V
Output high voltage		V_{oh}	1.4	1.8	1.9	V
Output low voltage		V_{ol}	N/A	0.0	0.27	V
Input high voltage	DVDD28_SIM2= 3.0V	V_{ih}	2.6	3.0	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.4	V
Output high voltage		V_{oh}	2.6	3.0	3.1	V
Output low voltage		V_{ol}	N/A	0.0	0.4	V
SIM2_SCLK						
Input high voltage	DVDD28_SIM2 = 1.8V	V_{ih}	1.4	1.8	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.27	V
Output high voltage		V_{oh}	1.62	1.8	1.9	V
Output low voltage		V_{ol}	N/A	0.0	0.22	V
Input high voltage	DVDD28_SIM2 = 3.0V	V_{ih}	2.6	3.0	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.4	V
Output high voltage		V_{oh}	2.7	3.0	3.1	V
Output low voltage		V_{ol}	N/A	0.0	0.4	V
SIM2_SRST						
Input high voltage	DVDD28_SIM2 = 1.8V	V_{ih}	1.4	1.8	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.27	V
Output high voltage		V_{oh}	1.62	1.8	1.9	V
Output low voltage		V_{ol}	N/A	0.0	0.36	V
Input high voltage	DVDD28_SIM2 = 3.0V	V_{ih}	2.6	3.0	N/A	V
Input low voltage		V_{il}	N/A	0.0	0.4	V
Output high voltage		V_{oh}	2.7	3.0	3.1	V
Output low voltage		V_{ol}	N/A	0.0	0.36	V

2.2.5 AC Electrical Characteristics and Timing Diagram

2.2.5.1 External Memory Interface for LPDDR3

The external memory interface, shown in Figure 2-3, Figure 2-4 and Figure 2-5, is used to connect LPDDR3 device for MT6755M. It includes pins CLK_T, CLK_C, CKE[1:0], CS[1:0], DQS[3:0], DQS#[3:0], CA[9:0] and DQ[31:0]. Table 2-18 summarizes the symbol definition and the related timing specifications.

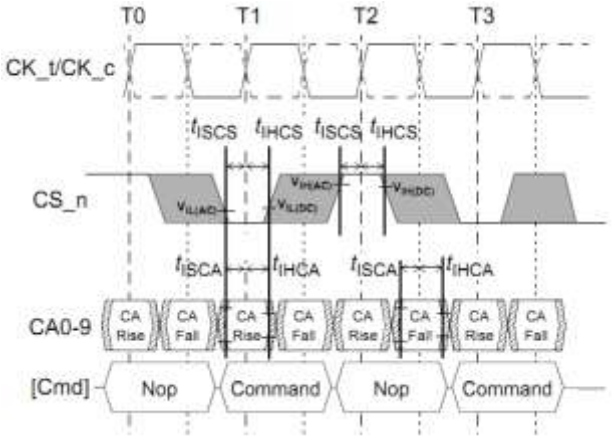


Figure 2-3. Basic timing parameter for LPDDR3 commands

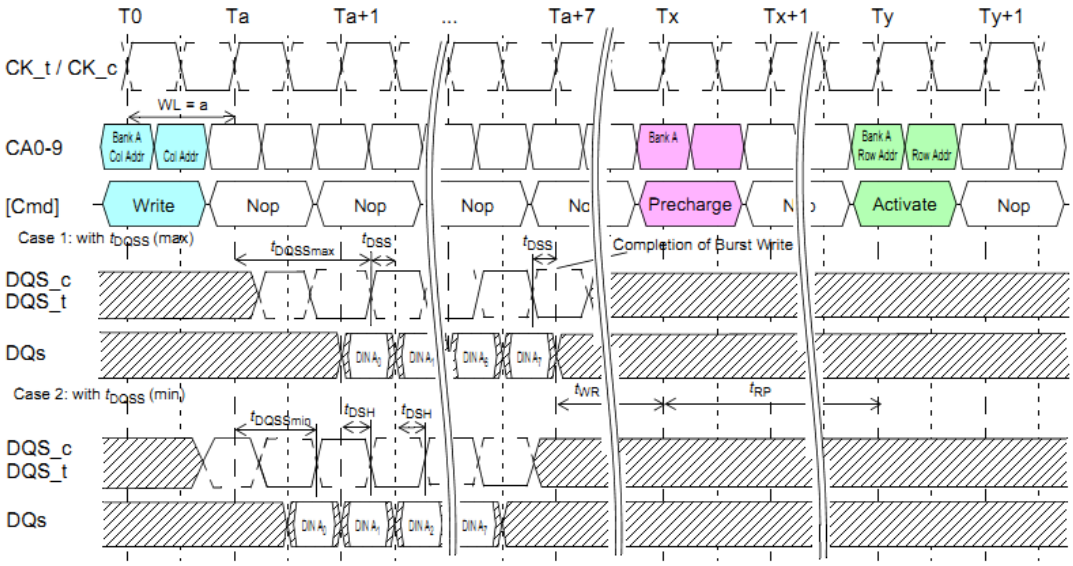


Figure 2-4. Basic timing parameter for LPDDR3 write

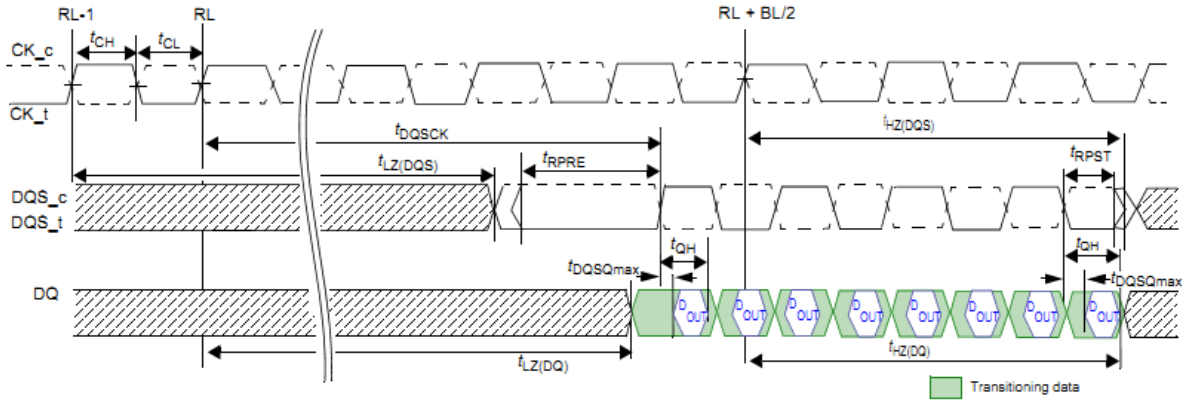


Figure 2-5. Basic LPDDR3 read timing parameter

Table 2-18. LPDDR3 AC timing parameter table of external memory interface

Symbol	Description	Min.	Typ.	Max.	Unit
tCK	Clock cycle time	1.071		100	ns
tDQSCK	DQS output access time from CK/CK'	2.5		5.5	ns
tCH	Clock high level width	0.45		0.55	tCK
tCL	Clock low level width	0.45		0.55	tCK
tDS	DQ & DM input setup time	0.13			ns
tDH	DQ & DM input hold time	0.13			ns
tDIPW	DQ and DM input pulse width	0.35			tCK
tDQSS	Write command to 1st DQS latching transition	0.75		1.25	tCK
tDSS	DQS falling edge to CK setup time	0.2			tCK
tDSH	DQS falling edge hold time from CK	0.2			tCK
tWPST	Write postamble	0.4			tCK
tWPRE	Write preamble	0.8			tCK
tISCA	Address & control input setup time	0.13			ns
tIHCA	Address & control input hold time	0.13			ns
tISCS	CS_ input setup time	0.23			ns
tIHCS	CS_ input hold time	0.23			ns
tIPWCA	Address and control input pulse width	0.35			tCK
tIPWCS	CS_ input pulse width	0.7			tCK
tCKE	CKE minimum pulse width (HIGH and LOW pulse width)	Max. (7.5ns, 3tCK)			ns
tISCKE	CKE input setup time	0.25			tCK
tIHCKE	CKE input hold time	0.25			tCK
tCPDED	Command path disable delay	2			tCK
tLZ(DQS)	DQS low-impedance time from CK/CK'	tDQSCK (MIN) - 0.3			ns

Symbol	Description	Min.	Typ.	Max.	Unit
tHZ(DQS)	DQS high-impedance time from CK/CK'			tDQSCK (MAX) – 0.1	ns
tLZ(DQ)	DQ low-impedance time from CK/CK'	tDQSCK (MIN) - 0.3			ns
tHZ(DQ)	DQ high-impedance time from CK/CK'			tDQSCK (MAX) + [1.4*tDQSQ (MAX)]	ns
tDQSQ	DQS-DQ skew			0.115	ns
tDQSH	DQS input high-level width	0.4			tCK
tDQSL	DQS input low-level width	0.4			tCK
tQSH	DQS output high pulse width	tCH - 0.05			tCK
tQSL	DQS output low pulse width	tCL - 0.05			tCK
tQH	DQ/DQS output hold time from DQS	Min. (tQSH, tQSL)			ns
tMRW	MODE register Write command period	Max. (10tCK, 15)			ns
tMRR	MODE register Read command period	4			tCK
tMRD	Mode register set command delay	Max. (10tCK, 14)			ns
tRPRE	Read preamble	0.9			tCK
tRPST	Read postamble	0.3			tCK
tRAS	ACTIVE to PRECHARGE command period	Max. (42ns, 3tCK)		70000	ns
tRC	ACTIVE to ACTIVE command period	tRAS + tRPab (with all-bank pre-charge) tRAS + tRPpb (with per-bank pre-charge)			ns
tRFC	AUTO REFRESH to ACTIVE/AUTO REFRESH command period	56			ns
tRCD	ACTIVE to READ or WRITE delay	Max. (18ns, 3tCK)			ns
tRPpb	Row PRECHARGE Time (single bank)	Max. (18ns, 3tCK)			ns
tRPab	Row PRECHARGE Time (all banks)	Max. (21ns, 3tCK)			ns
tRRD	ACTIVE bank A to ACTIVE bank B delay	Max. (10ns, 2tCK)			ns
tWR	WRITE recovery time	Max. (15ns, 4tCK)			ns
tWTR	Internal write to READ command time	Max. (7.5ns, 4tCK)			ns

Symbol	Description	Min.	Typ.	Max.	Unit
TXSR	SELF REFRESH exit to next valid command	Max. (tRFCab + 10ns, 2tCK)			ns
TXP	EXIT power down to next valid command delay	Max. (7.5ns, 3tCK)			ns
tREFW	Refresh period			32	ms
tRFCab	Refresh cycle time	130			ns
tRFCpb	Per bank refresh cycle time	60			ns
tRTP	Internal READ to PRECHARGE command delay	Max. (7.5ns, 4tCK)			ns
tCCD	CAS-to-CAS delay	4			tCK

2.2.5.2 SPI AC Timing Characteristics

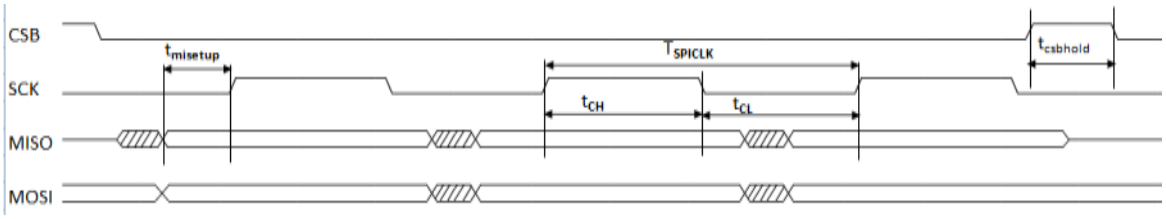


Figure 2-6. SPI timing diagram

Table 2-19. SPI AC timing parameters

Parameter	Symbol	Min.	Typ.	Max.	Unit
SPI clock period	T_{SPICLK}	18.2	-	-	ns
SPI clock low time	t_{CL}	9.1	-	-	ns
SPI clock high time	t_{CH}	9.1	-	-	ns
SPI CSB hold time	$t_{csbhold}$	9.1	-	-	ns
SPI MISO setup time (MISO 80%, SCK 20%)	$t_{misetup}$	28.5	-	-	ns

2.2.5.3 I2S AC Timing Characteristics

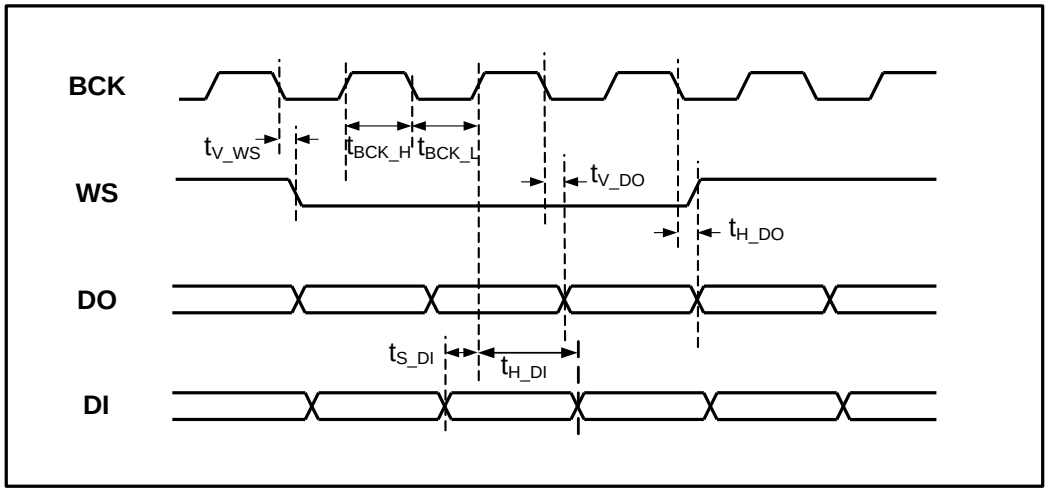


Figure 2-7. I2S master mode timing diagram

Table 2-20. I2S AC timing parameters

Parameter	Description	Min.	Typ.	Max.	Unit
f_S	Sampling frequency	8	-	192	kHz
t_{WS}	Word select period	32	-	64	$1/f_{BCK}$
f_{MCK}	Master clock frequency	-	-	24.576	MHz
f_{BCK}	Serial clock frequency	32 * f_S	-	64 * f_S	MHz
t_{BCK_H}	BCK high-level time	-	0.5	-	$1/f_{BCK}$
t_{BCK_L}	BCK low-level time	-	0.5	-	$1/f_{BCK}$
t_{V_WS}	WS valid time	-	-	0.2	$1/f_{BCK}$
t_{H_WS}	WS hold time	0	-	-	$1/f_{BCK}$
t_{V_DO}	DO valid time	-	-	0.2	$1/f_{BCK}$
t_{H_DO}	DO hold time	0	-	-	$1/f_{BCK}$
t_{S_DI}	DI setup time	0.2	-	-	$1/f_{BCK}$
t_{H_DI}	DI hold time	0.2	-	-	$1/f_{BCK}$

2.2.5.4 I2C AC Timing Characteristics

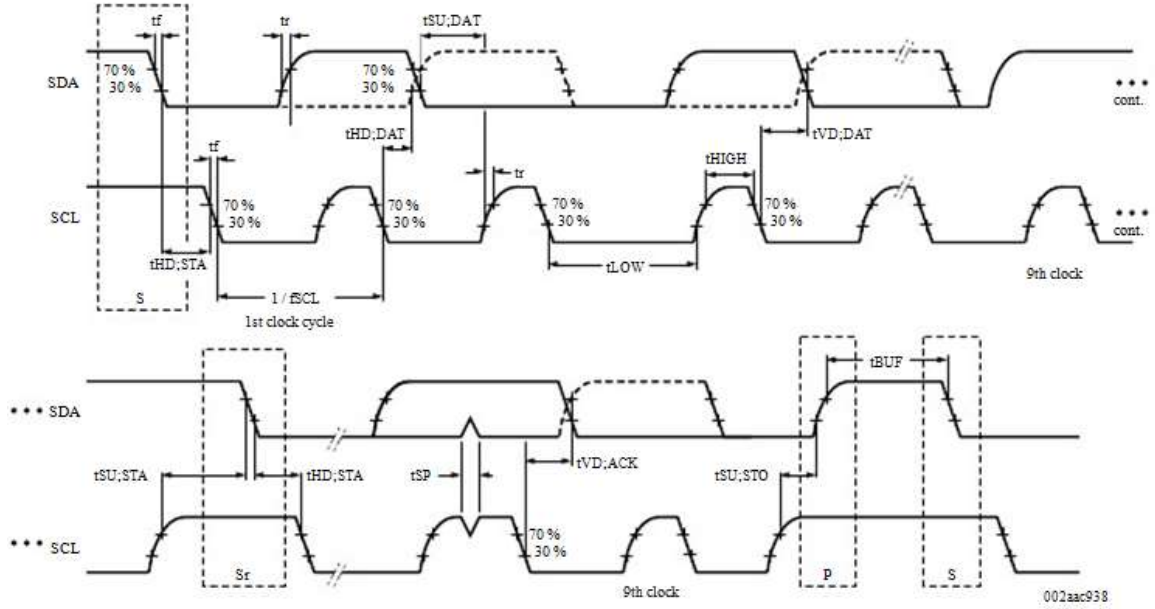


Figure 2-8. I2C timing diagram of standard mode (100kHz) and fast mode (400kHz)

Table 2-21. I2C AC timing parameters

Symbol	Standard mode	Fast mode	Unit	Note
$t_{HD;STA}$	2.5	0.625	µs	Can be extended by 0x28, extension configuration register.
t_{LOW}	5	1.25	µs	
t_{HIGH}	5	1.25	µs	
$t_{SU;STA}$	2.5	0.625	µs	
$t_{HD;DAT}$	2.5	0.625	µs	
$t_{SU;DAT}$	2.5	0.625	µs	
$t_{SU;STO}$	2.5	0.625	µs	Can be extended by 0x28, extension configuration register.

2.2.5.5 MSDC AC Timing Characteristics

2.2.5.5.1 Default Speed Timing

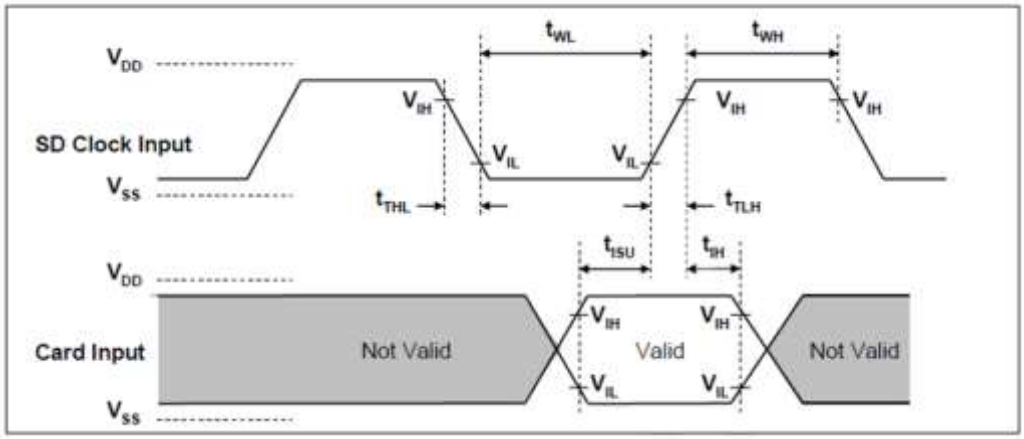


Figure 2-9. MSDC input timing diagram of default speed

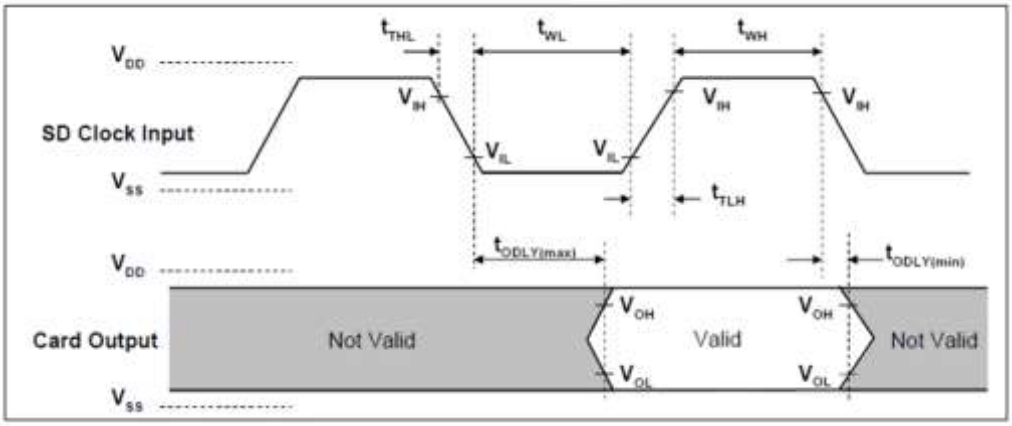


Figure 2-10. MSDC output timing diagram of default speed

Table 2-22. MSDC AC timing parameters of default speed

Parameter	Symbol	Min.	Max.	Unit	Remark
Clock CLK (All values are referred to min (V_{IH}) and max (V_{IL}))					
Clock frequency data transfer mode	f_{PP}	0	25	MHz	$C_{CARD} \leqq 10pF$ (1 card)
Clock frequency identification mode	f_{OD}	0[1] / 100	400	kHz	$C_{CARD} \leqq 10pF$ (1 card)
Clock low time	t_{WL}	10		ns	$C_{CARD} \leqq 10pF$ (1 card)
Clock high time	t_{WH}	10		ns	$C_{CARD} \leqq 10pF$ (1 card)
Clock rise time	t_{TLH}		10	ns	$C_{CARD} \leqq 10pF$ (1 card)
Clock fall time	t_{THL}		10	ns	$C_{CARD} \leqq 10pF$ (1 card)
Input CMD, DAT (referenced to CLK)					
Input setup time	T_{ISU}	5		ns	$C_{CARD} \leqq 10pF$ (1 card)
Input hold time	T_{IH}	5		ns	$C_{CARD} \leqq 10pF$ (1 card)
Output CMD, DAT (referenced to CLK)					
Output delay time during data transfer mode	T_{ODLY}	0	14	ns	$C_L \leqq 40pF$ (1 card)
Output delay time during identification mode	T_{ODLY}	0	50	ns	$C_L \leqq 40pF$ (1 card)

2.2.5.5.2 High Speed Timing

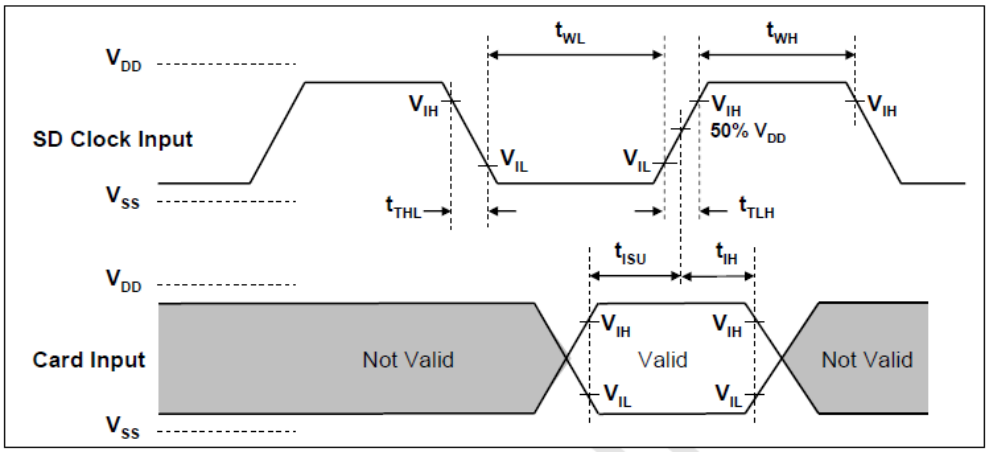


Figure 2-11. MSDC input timing diagram of high speed

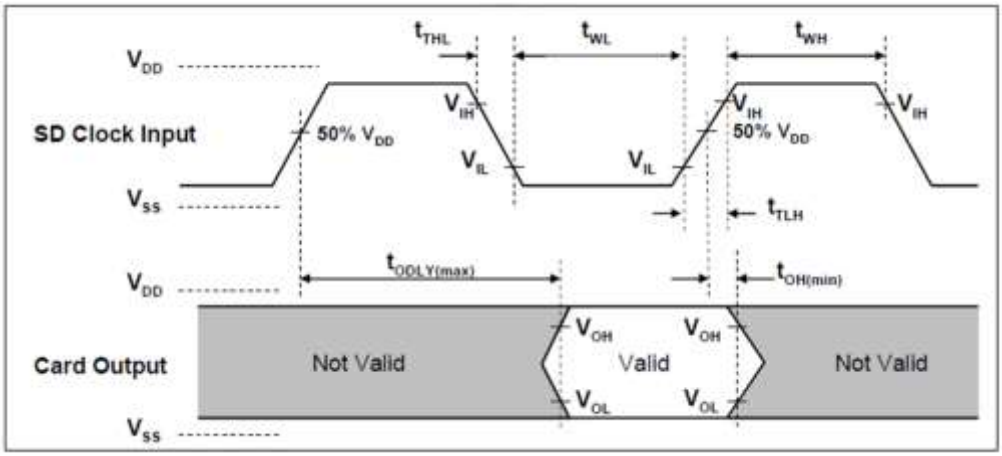


Figure 2-12. MSDC output timing diagram of high speed

Table 2-23. MSDC AC timing parameters of high speed

Parameter	Symbol	Min.	Max.	Unit	Remark
Clock CLK (All values are referred to min (V_{IH}) and max (V_{IL}))					
Clock frequency data transfer mode	f_{PP}	0	50	MHz	$C_{CARD} \leqq 10pF$ (1 card)
Clock low time	t_{WL}	7		ns	$C_{CARD} \leqq 10pF$ (1 card)
Clock high time	t_{WH}	7		ns	$C_{CARD} \leqq 10pF$ (1 card)
Clock rise time	t_{TLH}		3	ns	$C_{CARD} \leqq 10pF$ (1 card)
Clock fall time	t_{THL}		3	ns	$C_{CARD} \leqq 10pF$ (1 card)
Input CMD, DAT (referenced to CLK)					
Input setup time	t_{ISU}	6		ns	$C_{CARD} \leqq 10pF$ (1 card)
Input hold time	t_{IH}	2		ns	$C_{CARD} \leqq 10pF$ (1 card)
Output CMD, DAT (referenced to CLK)					
Output delay time during data transfer mode	t_{ODLY}		14	ns	$C_L \leqq 40pF$ (1 card)
Output hold time	t_{OH}	2.5		ns	$C_L \geqq 15pF$ (1 card)
Total system capacitance for each line	C_L		40	pF	1 card

2.2.5.5.3 SDR12/SDR25/SDR50/SDR104 Mode Timing

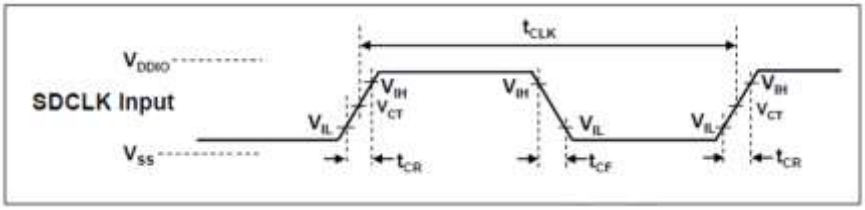


Figure 2-13. MSDC clock timing diagram of SDR12/SDR25/SDR50/SDR104 mode

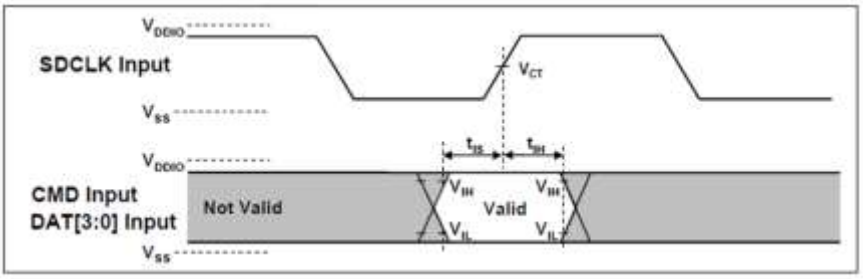


Figure 2-14. MSDC input timing diagram of SDR50/SDR104 mode

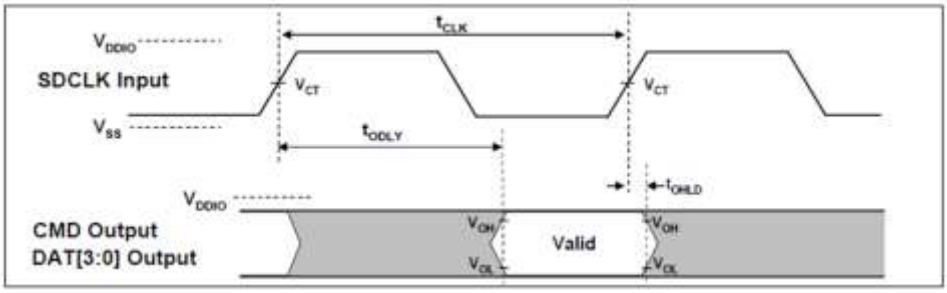


Figure 2-15. MSDC output timing diagram of fixed data window (SDR12/SDR25/SDR50)

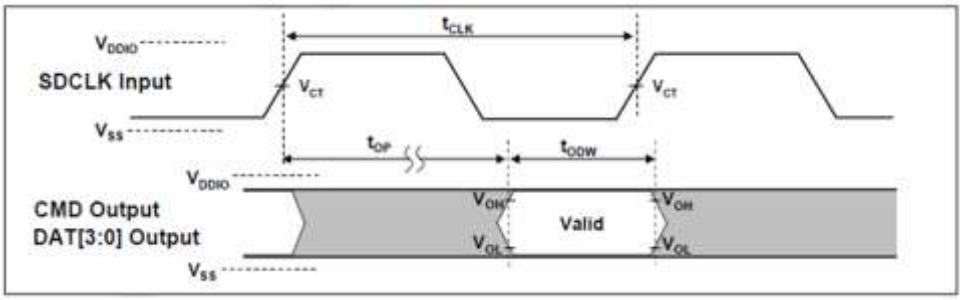


Figure 2-16. MSDC output timing diagram of variable window (SDR104)

Table 2-24. MSDC AC timing parameters of SDR12/SDR25/SDR50/SDR104 mode

Symbol	Min.	Max.	Unit	Remark
Clock CLK				
t_{CLK}	4.8	-	ns	208MHz (Max), Between rising edge, V_{CT}=0.975V
t_{CR}, t_{CF}	-	0.2*t_{CLK}	ns	t_{CR}, t_{CF} < 0.96ns (max) at 208MHz, C_{CARD}=10pF t_{CR}, t_{CF} < 2.00ns (max) at 100MHz, C_{CARD}=10pF The absolute maximum value of t_{CR}, t_{CF} is 10ns regardless of clock frequency.
Clock Duty	30	70	%	
Input CMD, DAT (SDR104)				
t_{IS}	1.40	-	ns	C_{CARD}=10pF, V_{CT}=0.975V
t_{IH}	0.80		ns	C_{CARD}=5pF, V_{CT}=0.975V
Input CMD, DAT (SDR50)				
t_{IS}	3.00	-	ns	C_{CARD}=10pF, V_{CT}=0.975V
t_{IH}	0.80	-	ns	C_{CARD}=5pF, V_{CT}=0.975V
Output CMD, DAT (SDR12/SDR25/SDR50)				
t_{ODLY}	-	7.5	ns	t_{CLK}≧10.0ns, C_L=30pF, using driver type B, for SDR50
t_{ODLY}		14	Ns	t_{CLK}≧20.0ns, C_L=40pF, using driver type B, for SDR25 and SDR12
T_{OH}	1.5	-	ns	Hold time at the t_{ODLY} (min), C_L=15pF
Output CMD, DAT (SDR104)				
t_{OP}	0	2	UI	Card output phase
Δt_{OP}	-350	+1550	ps	Delay varioation due to temperature change after tunung.
t_{ODW}	0.6	-	UI	t_{ODW}=2.88ns at 208MHz

2.2.5.5.4 DDR50 Speed Mode Timing

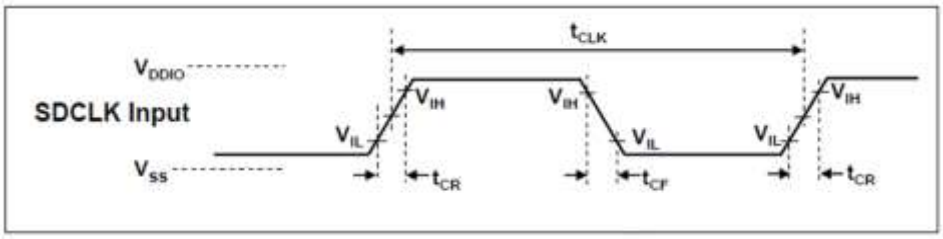


Figure 2-17. MSDC clock timing diagram of DDR50 speed mode.

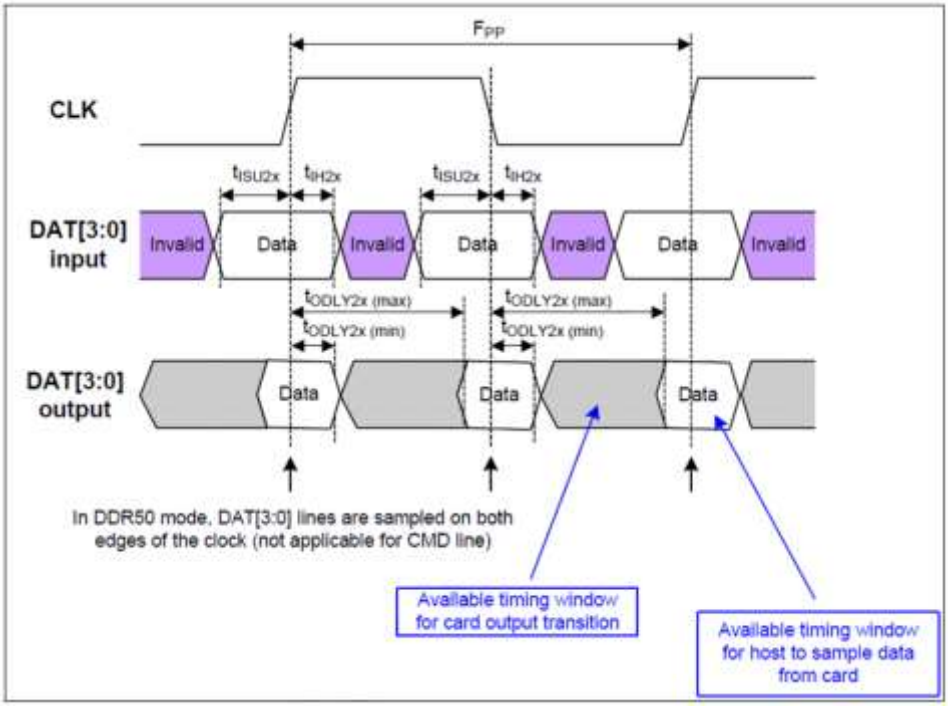


Figure 2-18. MSDC input/output timing diagram of DDR50 speed mode

Table 2-25. MSDC AC timing parameters of DDR50 speed mode

Parameter	Symbol	Min.	Max.	Unit	Remark
Input CMD (referenced to CLK rising edge)					
Input setup time	t_{ISU}	6	-	ns	$C_{CARD} \leqq 10pF$ (1 card)
Input hold time	t_{IH}	0.8	-	ns	$C_{CARD} \leqq 10pF$ (1 card)
Output CMD (referenced to CLK rising edge)					
Output delay time during data transfer mode	t_{ODLY}	-	13.7	ns	$C_L \leqq 30pF$ (1 card)
Output hold time	t_{OH}	1.5	-	ns	$C_L \geqq 15pF$ (1 card)
Input DAT (referenced to CLK rising and falling edge)					
Input setup time	t_{ISU}	3	-	ns	$C_{CARD} \leqq 10pF$ (1 card)
Input hold time	t_{IH}	0.8	-	ns	$C_{CARD} \leqq 10pF$ (1 card)
Output DAT (referenced to CLK rising and falling edge)					
Output delay time during data transfer mode	t_{ODLY}	-	7.0	ns	$C_L \leqq 25pF$ (1 card)
Output hold time	t_{OH}	1.5	-	Ns	$C_L \geqq 15pF$ (1 card)

2.2.5.5.5 HS200 Speed Timing

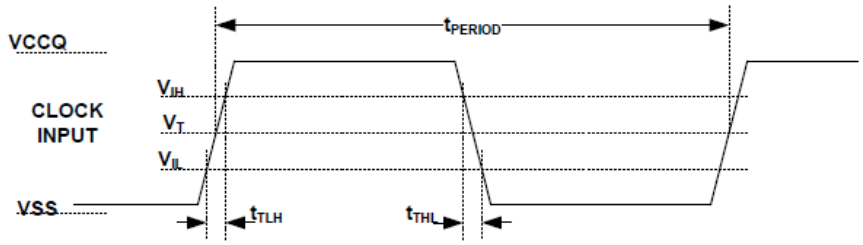
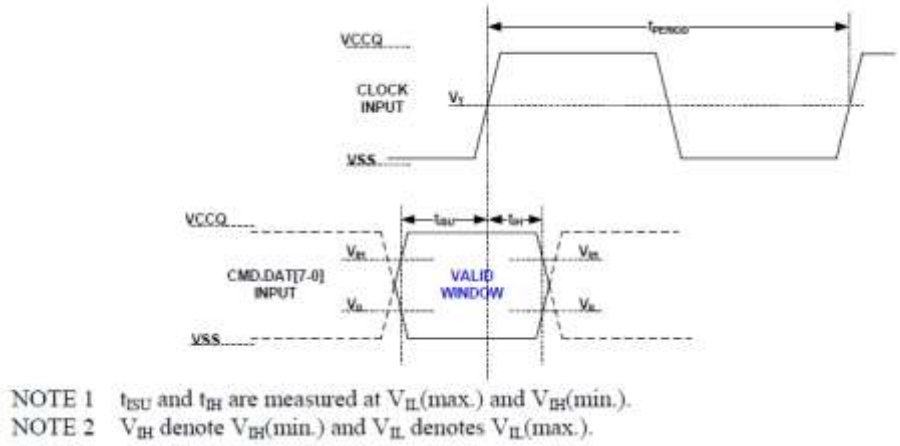


Figure 2-19. MSDC clock timing diagram of HS200



NOTE 1 t_{ISU} and t_{IH} are measured at V_{IL}(max.) and V_{IH}(min.).
NOTE 2 V_{IH} denote V_{IH}(min.) and V_{IL} denotes V_{IL}(max.).

Figure 2-20. MSDC input timing diagram of HS200

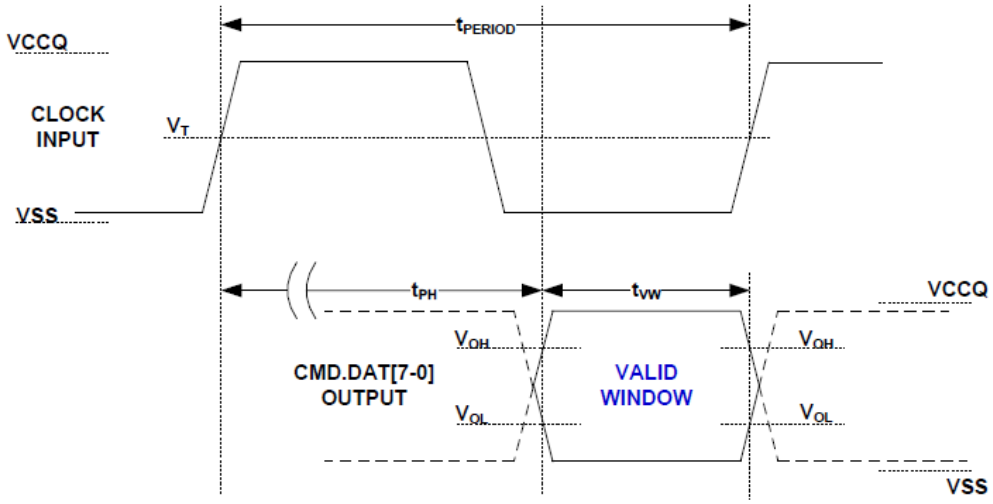


Figure 2-21. MSDC output timing diagram of HS200

Table 2-26. MSDC AC timing parameters of HS200

Symbol	Min.	Max.	Unit	Remark
Clock CLK				
t_{PERIOD}	5	-	ns	200MHz (Max), between rising edge
t_{TLH}, t_{THL}	-	0.2 * t_{PERIOD}	ns	t_{TLH}, t_{THL} < 1ns (max) at 200MHz, C_{DEVICE}=6pF The absolute maximum value of t_{TLH}, t_{THL} is 10ns regardless of clock frequency.
Clock Duty	30	70	%	
Input CMD, DAT				
t_{ISU}	1.40	-	ns	C_{DEVICE}≦6pF
t_{IH}	0.80		ns	C_{DEVICE}≦6pF
Output CMD, DAT				
t_{PH}	0	2	UI	Device output momentary phase from CLK input to CMD or DAT lines output. Does not include a long term temperature drift.
ΔT_{PH}	-350 (ΔT=-20℃)	+1550 (ΔT=90℃)	ps	Delay varioation due to temperature change after tunung. Total allowable shift of output valid window (T_{VW}) from last system Tuning procedure ΔT_{PH} is 2600ps for ΔT from -25°C to 125°C during operation.
t_{VW}	0.575	-	UI	t_{VW}=2.88ns at 208MHz
Note: Unit Interval (UI) is one bit nominal time. For example, UI=5ns at 200MHz.				

2.2.5.5.6 HS400 Speed Timing

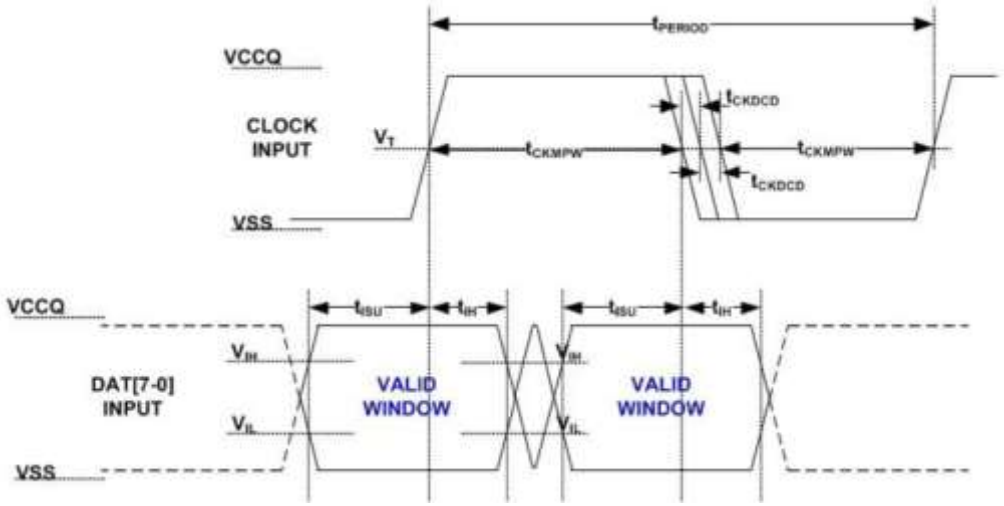


Figure 2-22. MSDC input timing diagram of HS400

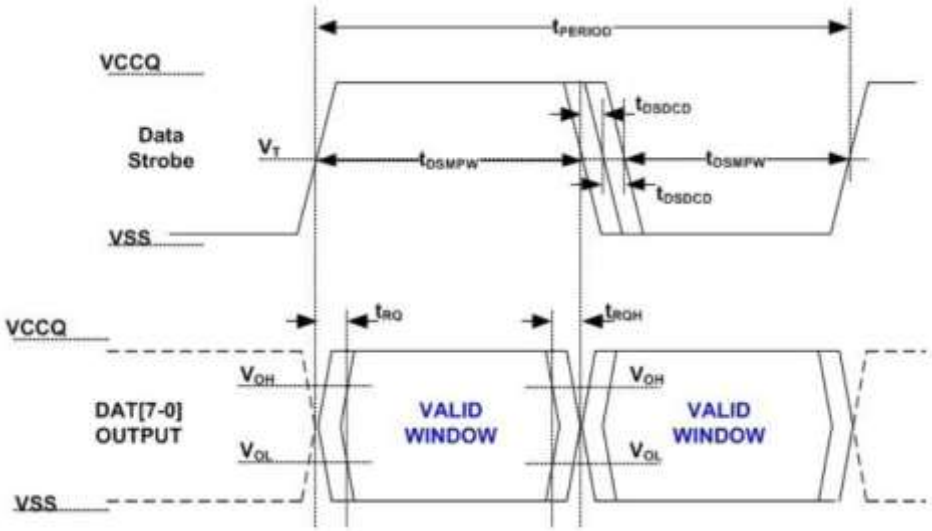


Figure 2-23. MSDC output timing diagram of HS400

Table 2-27. MSDC AC timing parameters of HS400

Parameter	Symbol	Min.	Max.	Unit	Remark
Input CLK					
Cycle time data transfer mode	t_{PERIOD}	5		ns	200MHz (max), between rising edges. With respect to V_T
Slew rate	SR	1.125		V/ns	With respect to V_{IH}/V_{IL}
Duty cycle distortion	t_{CKDCD}	0.0	0.3	ns	Allowable deviation from an ideal 50% duty cycle. With respect to V_T. Includes jitter, phase noise
Minimum pulse width	t_{CKMPW}	2.2		ns	With respect to V_T.
Input DAT (referenced to CLK)					
Input setup time	t_{ISUddr}	0.4		ns	$C_{Device} \leqq 6pF$. With respect to V_{IH}/V_{IL}
Input hold time	t_{IHddr}	0.4		ns	$C_{Device} \leqq 6pF$. With respect to V_{IH}/V_{IL}
Slew rate	SR	1.125		V/ns	With respect to V_{IH}/V_{IL}
Data Strobe					
Cycle time data transfer mode	t_{PERIOD}	5		ns	200MHz (max), between rising edges. With respect to V_T
Slew rate	SR	1.125		V/ns	With respect to V_{OH}/V_{OL} and HS400 reference load
Duty cycle distortion	t_{DSDCD}	0.0	0.2	ns	Allowable deviation from the input CLK duty cycle distortion (t_{CKDCD}). With

Parameter	Symbol	Min.	Max.	Unit	Remark
					respect to V_T. Includes jitter, phase noise
Minimum pulse width	t_{DSMPW}	2.0		ns	With respect to V_T.
Read pre-amble	t_{RPRE}	0.4		t_{PERIOD}	Max value is specified by manufacturer. Value up to infinite is valid.
Read post-amble	t_{RPST}	0.4		t_{PERIOD}	Max value is specified by manufacturer. Value up to infinite is valid.
Input DAT (referenced to Data Strobe)					
Output skew	t_{RQ}		0.4	ns	With respect to V_{OH}/V_{OL} and HS400 reference load
Output hold skew	t_{RQH}		0.4	ns	With respect to V_{OH}/V_{OL} and HS400 reference load
Slew rate	SR	1.125		V/ns	With respect to V_{OH}/V_{OL} and HS400 reference load

2.2.5.6 SIM AC Timing Characteristics

Table 2-28. SIM AC timing parameters

Parameter	Conditions	Symbol	Min.	Typ.	Max.	Unit
SIM1_SCLK						
Rise and fall time	DVDD28_SIM1 = 1.8V	T_{rise_fall}	N/A	50	50	Ns
Clock duty		Duty	47	50	53	%
Rise and fall time	DVDD28_SIM1 = 3.0V	T_{rise_fall}	N/A	18	18	ns
Clock duty		Duty	47	50	53	%
SIM1_SIO						
Rise and fall time	DVDD28_SIM1 = 1.8V	T_{rise_fall}	N/A	50	1000	ns
Rise and fall time	DVDD28_SIM1 = 3.0V	T_{rise_fall}	N/A	50	1000	ns
SIM1_SRST						
Rise and fall time	DVDD28_SIM1 = 1.8V	T_{rise_fall}	N/A	18	1000	ns
Rise and fall time	DVDD28_SIM1 = 3.0V	T_{rise_fall}	N/A	18	1000	ns
SIM2_SCLK						
Rise and fall time	DVDD28_SIM2 = 1.8V	T_{rise_fall}	N/A	50	50	ns
Clock duty		Duty	47	50	53	%
Rise and fall time	DVDD28_SIM2 = 3.0V	T_{rise_fall}	N/A	18	18	ns
Clock duty		Duty	47	50	53	%
SIM2_SIO						
Rise and fall time	DVDD28_SIM2 = 1.8V	T_{rise_fall}	N/A	50	1000	ns
Rise and fall time	DVDD28_SIM2 = 3.0V	T_{rise_fall}	N/A	50	1000	ns
SIM2_SRST						

Parameter	Conditions	Symbol	Min.	Typ.	Max.	Unit
Rise and fall time	DVDD28_SIM2 = 1.8V	T_{rise_fall}	N/A	18	1000	ns
Rise and fall time	DVDD28_SIM2 = 3.0V	T_{rise_fall}	N/A	18	1000	ns

2.3 System Configuration

2.3.1 Mode Selection

Table 2-29. Mode selection

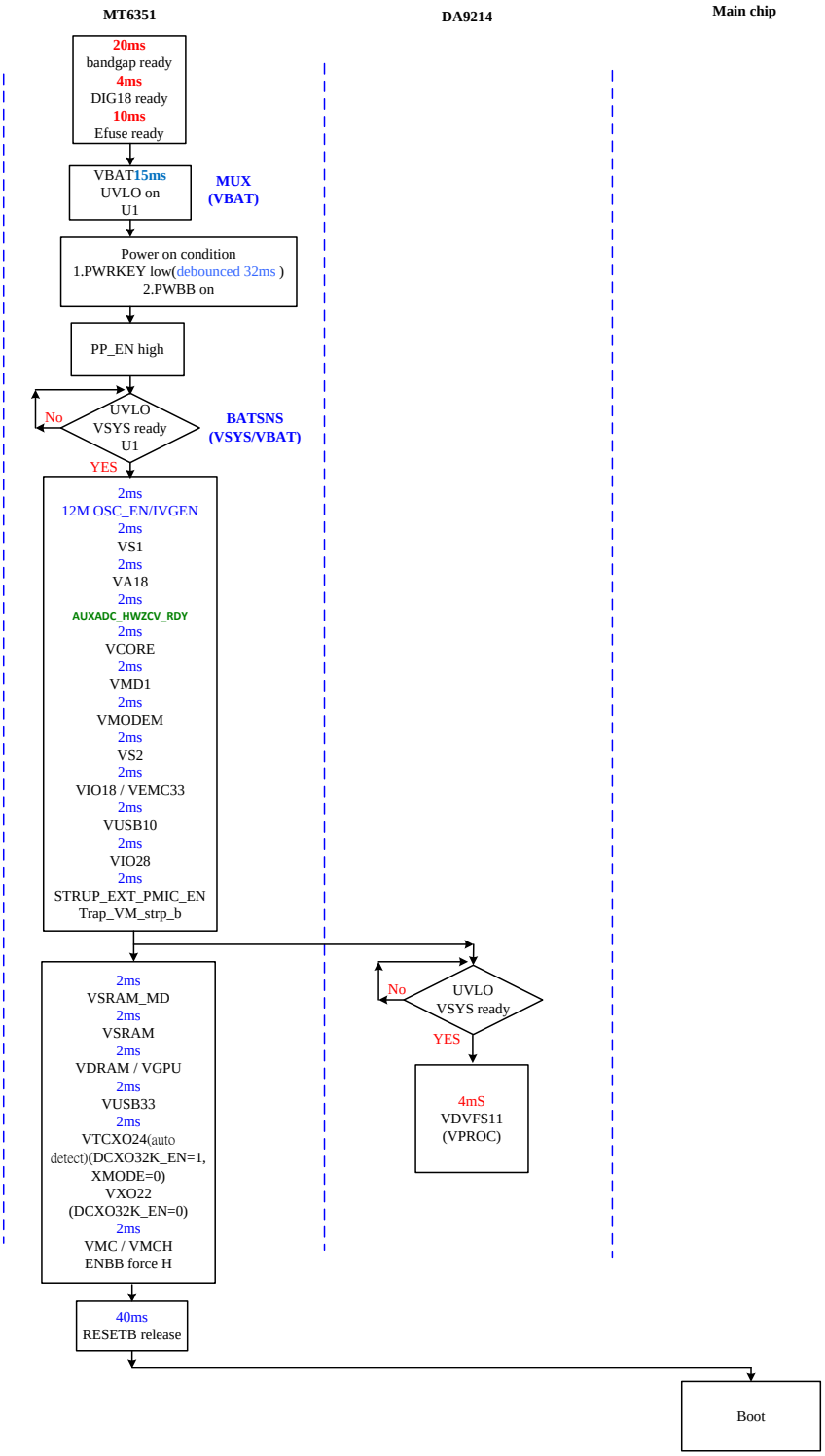
Pin name	Description
KCOL0	0: Force USB download mode in bootrom 1: NA (default)

2.3.2 Constant Tie Pins

Table 2-30. Constant tied pins

Pin name	Description
TESTMODE	Test mode (tied to GND)

2.4 Power-on Sequence



2.5 Analog Baseband

2.5.1 Introduction

To communicate with analog blocks, a common control interface for all analog blocks is implemented. In addition, there are some dedicated interfaces for data transfer. The common control interface translates the APB bus write and read cycle for specific addresses related to analog front-end control. In the write or read of any of these control registers, there is a latency associated with the transfer of data to or from the analog front-end. Dedicated data interface of each analog block is implemented in the corresponding digital block. An analog block includes the following analog functions for the complete GSM/GPRS/WCDMA/LTE/C2K base-band signal processing:

- Base-band Rx: For I/Q channels base-band A/D conversion
- Base-band Tx: For I/Q channels base-band D/A conversion and smoothing filtering
- ETDAC: DAC output to control buck-converter for envelop tracking technique.
- DETADC: ADC that detects calibration, thermal data from RF chip.
- RF control: DAC for automatic power control (APC) is included. The outputs are provided to external RF power amplifiers.
- Auxiliary ADC: Provides an ADC for auxiliary analog functions monitoring.
- Clock generation: One clock-squarer for shaping the input sinwave clock and PLLs providing clock signals to base-band TRx, DSP, MCU, USB, MSDC units.

2.5.2 Features

The analog blocks include the following analog functions for complete GSM/GPRS/WCDMA/LTE/C2K base-band signal processing:

- BBRX
- BBTX
- ETDAC
- DETADC
- APC-DAC
- AUXADC
- Phase locked loop
- Temperature sensor

2.5.3 Block Diagram

2.5.3.1 BBRX

2.5.3.1.1 Block Descriptions

The receiver (Rx) performs baseband I/Q channels downlink analog-to-digital conversion:

1. Analog input multiplexer: For each channel, a 2-input multiplexer is included.
2. A/D converter: 8 high performance sigma-delta ADCs perform I/Q digitization for further digital signal processing.

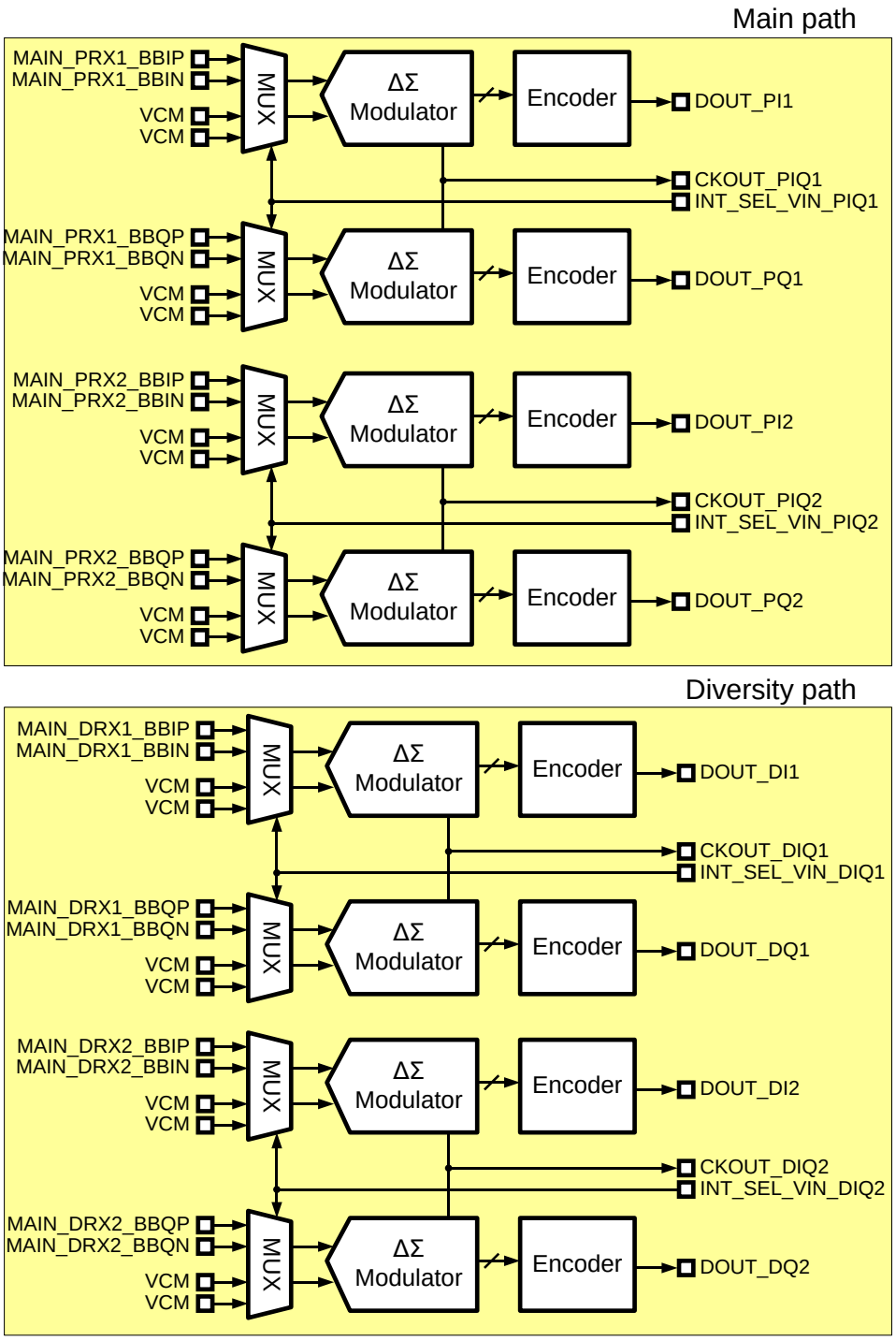


Figure 2-24. Block diagram of BBRX-ADC

2.5.3.1.1 Functional Specifications

See the table below for the functional specifications of the base-band downlink receiver.

Table 2-31. Baseband downlink specifications

Symbol	Parameter	Min.	Typ.	Max.	Unit
VIN	Differential analog input voltage (peak-to-peak)			2.4	V
VCM	Common mode input voltage	0.65	0.7	0.75	V
FC	Input clock frequency	208		832	MHz
	Input clock duty cycle	49.5	50	50.5	%
RIN	Differential input resistance	2.8		20.8	kΩ
FS	Output sampling rate	208		832	MSPS
VOS	Differential input referred offset			10	mV
SIN	Signal to in-band noise	70		89	dB
AVDD	Analog power supply	1.7	1.8	1.9	V
T	Operating temperature	–20		80	°C
	Current consumption (per channel) – Power-up – Power-down		 16 10		 mA uA

2.5.3.2 BBTX

2.5.3.2.1 Block Descriptions

BBTX includes two channels current DACs with two 1st order low pass filter.

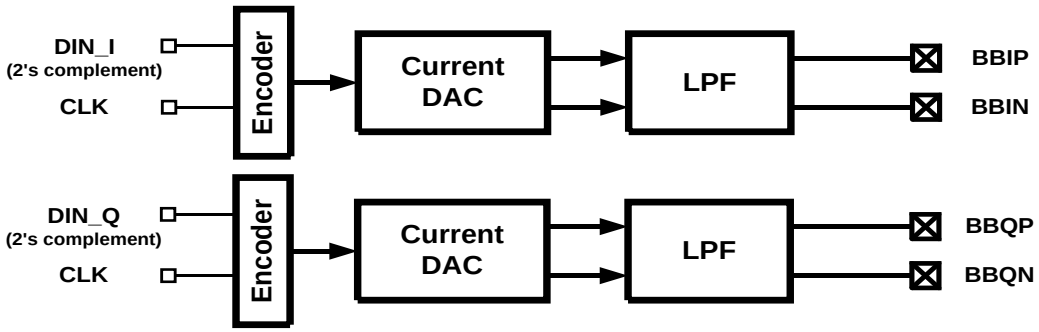


Figure 2-25. Block diagram of BBTX

2.5.3.2.2 Functional Specifications

Table 2-32. BBTX specifications

Symbol	Parameter	Min.	Typ.	Max.	Unit
V_{ocm}	DC output common mode voltage	0.615	0.65	0.685	V
V_{fs}	DAC output swing		2100		mV
N	DAC resolution		11.0		bit
F_s	Sampling clock		416		MHz
G_{mis}	3-sigma I/Q gain mismatch	-0.2		0.2	dB
V_{os}	3-sigma output differential DC offset			20	mV
F_{3dB}	3dB corner freq.		20/40		MHz
DNL			1		LSB
INL			2		LSB
IM3	In-band two-tone test swing V1=V2=290/sqrt(2) mV		-58	-55	dBc
AVDD	Analog power supply	1.7	1.8	1.9	V
T	Operating temperature	-20		80	°C
	Current consumption – Power-up – Power-down		 6 10		 mA uA

2.5.3.3 ETDAC

2.5.3.3.1 Block Descriptions

The ETDAC (Envelope Tracking DAC) provides analog envelope signal to external ET modulator. It includes one current DAC with a 1st order low pass filter.

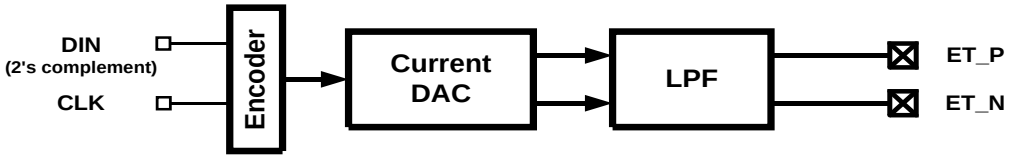


Figure 2-26. Block diagram of ETDAC

2.5.3.3.2 Functional Specifications

Table 2-33. ETDAC specifications

Symbol	Parameter	Min.	Typ.	Max.	Unit
N	Resolution		11		Bit
FS	Sampling rate		416		MSPS
IM3	3rd order Intermodulation distortion			-50	dB
	Output swing (full swing)		2		Vppd
VOCM	Output CM voltage	0.6		0.9	V
	Output loading capacitance (single-ended)			10	PF
	Output loading resistance (differential)	100			KΩ
DNL	Differential nonlinearity	-1		+1	LSB
INL	Integral nonlinearity	-2		+2	LSB
FCUT	Filter -3dB cutoff frequency (calibrated)		20/40		MHz
AVDD	Analog power supply	1.7	1.8	1.9	V
T	Operating temperature	-20		80	°C
	Current consumption – Power-up – Power-down		 4 10		 mA uA

2.5.3.4 DETADC

2.5.3.4.1 Block Descriptions

The DETADC (DETection ADC) performs I/Q-channel path detections from RF chip:
1. Input buffer: For each channel, isolates RF signals from high-speed ADCs.
2. 12-bit A/D converters: Converts the detected signals to 12-bit digital data sampled at 104MHz.

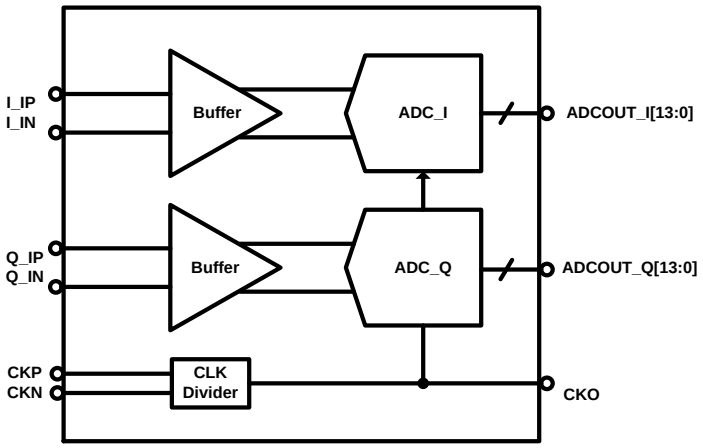


Figure 2-27. Block diagram of DETADC

2.5.3.4.2 Functional Specifications

See the table below for the functional specifications of DETADC

Table 2-34. DETADC specifications

Symbol	Parameter	Min.	Typ.	Max.	Unit
VIN	Analog input voltage (differential peak-to-peak)		1.2		V
VCM	Common mode input voltage		0.55		V
RIN	Input resistance	1.6	2	2.4	kΩ
CIN	Input capacitance		2	3	pF
FS	Sampling rate		104		MSPS
VOS	Differential input referred offset		30		mV
SNDR	Signal-to-noise-and-distortion ratio		60		dB
DR	Dynamic range		63		dB
THD	Total harmonic distortion		-66		dBc
AVDD	Analog power supply	1.7	1.8	1.9	V
T	Operating temperature	–20		80	°C
	Current consumption (per channel) – Power-up – Power-down		 6 1		 mA uA

2.5.3.5 APC-DAC

2.5.3.5.1 Block Descriptions

See the figure below. APC-DAC is designed to produce a single-ended output signal at APC pin.

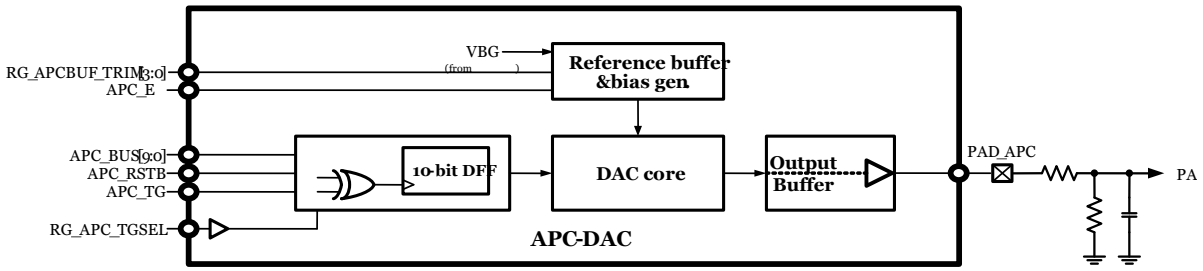


Figure 2-28. Block diagram of APC-DAC

2.5.3.5.2 Functional Specifications

See the table below for the functional specifications of the APC-DAC.

Table 2-35. APC-DAC specifications

Symbol	Parameter	Min.	Typ.	Max.	Unit
N	Resolution		10		Bit
Fs	Clock rate	1.0833		2.1666	MS/s
SNDR	Signal-to-noise-and-distortion ratio (10kHz sine wave with 1.0V swing)		50		dB
Ts	Settling time (99% full-swing settling)			5	us
$V_{O,max}$	Maximum output			AVDD – 0.2	V
C_L	Output loading capacitance		220	2200	pF
DNL	Differential nonlinearity (code 30 ~ 970)		±1.0		LSB
INL	Integral nonlinearity (code 30 ~ 970)		±2.0		LSB
AVDD	Analog power supply	2.6	2.8	3.0	V
T	Operating temperature	–20		85	°C
	Current consumption – Power-up – Power-down		 450 20		 uA uA

2.5.3.6 AUXADC

2.5.3.6.1 Block Descriptions

The auxiliary ADC includes the following functional blocks:

1. Analog multiplexer: Selects signal from one of the auxiliary input channels. There are 16 input channels of AUXADC. Some are for internal voltage measurement and some for external voltage measurement. Environmental messages to be monitored, e.g. temperature, should be transferred to the voltage domain.
2. 12-bit A/D converter: Converts the multiplexed input signal to 12-bit digital data.

Table 2-36. Definitions of AUXADC channels

AUXADC channel ID	Description
Channel 0	External use (AUX_IN0)
Channel 1	External use (AUX_IN1)
Channel 2	External use (AUX_IN2)
Channel 3	External use (AUX_IN3)
Channel 4	External use (AUX_IN4)
Channel 5	NA

AUXADC channel ID	Description
Channel 6	NA
Channel 7	NA
Channel 8	NA
Channel 9	NA
Channel 10	Internal use
Channel 11	Internal use
Channel 12	NA
Channel 13	NA
Channel 14	NA
Channel 15	NA

2.5.3.6.2 Functional Specifications

See the table below for the functional specifications of auxiliary ADC.

Table 2-37. AUXADC specifications

Symbol	Parameter	Min.	Typ.	Max.	Unit
N	Resolution		12		Bit
FC	Clock rate		3.25		MHz
FS	Sampling rate @ N-Bit		3.25/(N+8)		MSPS
	Input swing	0.05		1.45	V
CIN	Input capacitance Unselected channel Selected channel		 50 4		 fF pF
RIN	Input r esistance Unselected channel	 20			MΩ
	Clock latency		N+8		1/FC
DNL	Differential nonlinearity		+1.0/-1.0		LSB
INL	Integral nonlinearity		+2.0/-2.0		LSB
AVDD	Analog power supply	1.7	1.8	1.9	V
T	Operating temperature	-20		80	°C
	Accuracy			+-10	mV

2.5.3.7 Clock Squarer

2.5.3.7.1 Block Descriptions

For most VCXO, the output clock waveform is sinusoidal with too small amplitude (about several hundred mV) to make digital circuits function well. The clock squarer is designed to convert such a small signal to a rail-to-rail clock signal with excellent duty-cycle.

2.5.3.7.2 Functional Specifications

See the table below for the functional specifications of clock squarer.

Table 2-38. Clock squarer specifications

Symbol	Parameter	Min.	Typ.	Max.	Unit
Fin	Input clock frequency	13	26		MHz
Fout	Output clock frequency	13	26		MHz
Vin	Input signal amplitude	350	500	1,000	mVpp
DcycIN	Input signal duty cycle		50		%
DcycOUT	Output signal duty cycle	DcycIN-5		DcycIN+5	%
TR	Rise time on pin CLKSQOUT			5	ns/pF
TF	Fall time on pin CLKSQOUT			5	ns/pF
AVDD	Analog power supply	1.7	1.8	1.9	V
T	Operating temperature	-20		80	°C
	Current consumption		500		uA

2.6 Package Information

2.6.1 Package Outlines

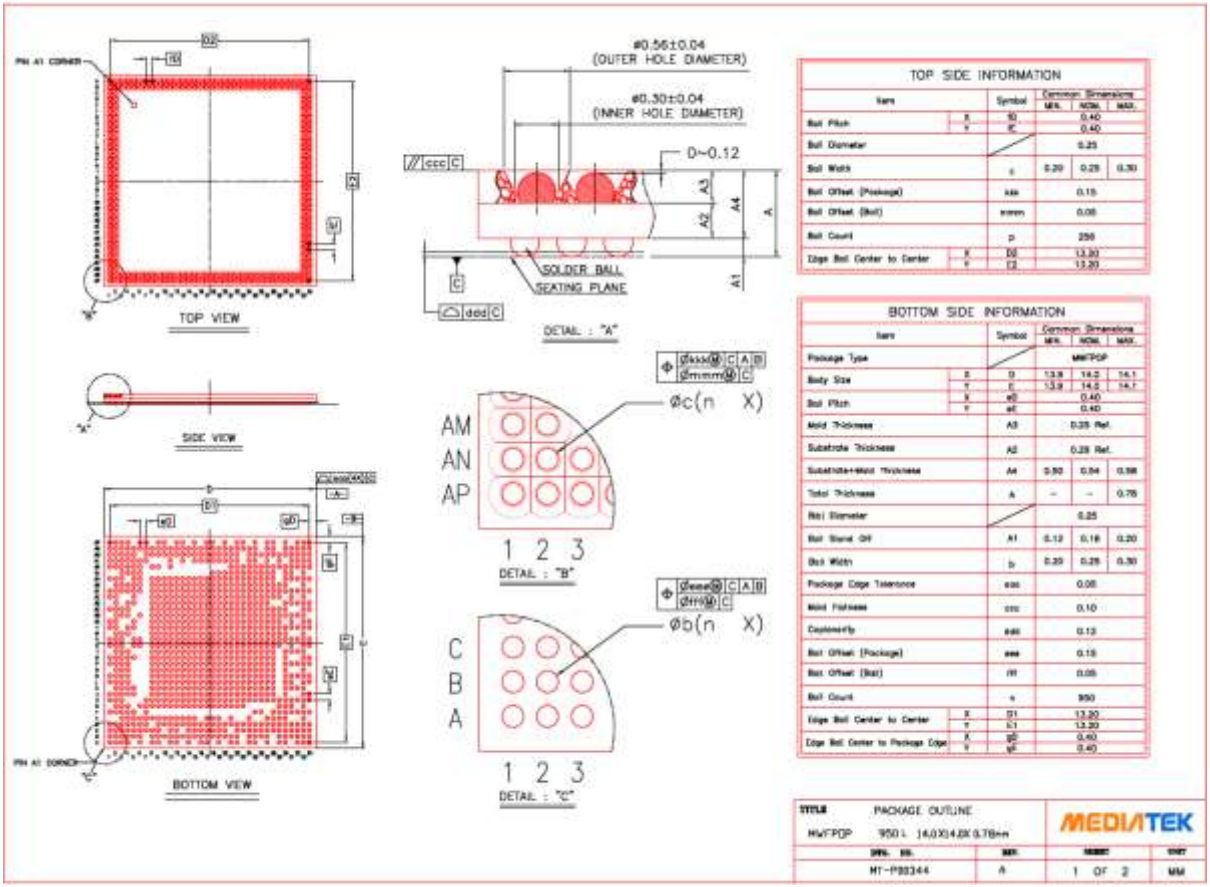


Figure 2-29. Outlines and dimensions of MWDPOP 14.0mm*14.0mm, 950-ball, 0.4mm pitch package

2.6.2 Thermal Operating Specifications

Table 2-39. Thermal operating specifications

Symbol	Description	Value	Unit	Note
	Max. operating junction temperature	125	°C	
	Package thermal resistances in nature convection	22.5	°C/Watt	

2.6.3 Lead-free Packaging

The chip is provided in a lead-free package and meets RoHS requirements.

2.7 Power Delivery Network

Table 2-40. PDN specifications

Power rail	ZAC				Remote	RDC
	Lpcb (H)	Rpcb (ohm)	Cpcb (F)	Range		
Vproc1	0.15n	6.5m	65u	40K~20M	0.55m	11m
Vproc2	0.15n	7.5m	65u	40K~20M	0.52m	10.8m
Vgpu	0.2n	18m	30u	40K~90M	2.60m	51.9m
Vcore	0.3n	75m	40u	30K~70M	1.09m	23.7m
VMD	0.4n	30m	13.5u	60K~50M	2.46m	49.2m
VMD1	0.45n	45m	13.5u	60K~60M	4.03m	80.6m
VDRAM	0.75n	110m	37u	20K~220M	41m	85.2m
	1.1n	120m	37u	20K~220M	47.4m	98.7m
	0.08	20m	37u	20K~220M	2.5m	27m
DVDD_SRAM_MD	3.9n	40m	17u	50K~280M	15.00m	300m
DVDD_SRAM_PROC1	2.3n	340m	0u	0~80M		
DVDD_SRAM_PROC2	1.8n	340m	0u	0~80M		
DVDD_SRAM_GPU	2.15n	340m	0u	0~80M		

Note: Refer to document "MT6797_PDN_Checking_Notice_V0_2.pdf" on DCC for more design concepts.

2.8 Ordering Information

2.8.1 Top Marking Definition

- MT6797W: Part No.
- DDDD: Date code
- %: Application code
- #####: Subcontractor code
- LLLLLLL: Lot number

Figure 2-30. Top mark of MT6797

2.8.2 Function Code Table

Table 2-41. Function code

Part number	MT6797 (2.3GHz)
6M	MT6797W/C
5M (w/o CDMA2000)	MT6797W/W

3 MCU and Bus Fabric

3.1 On-chip Memory Controller

3.1.1 Introduction

The on-chip memory controller provides the boot ROM and SRAM resources.

3.1.2 Features

The memory controller has the following features

- 80KB on-chip ROM, with memory access protection and detection
- 192KB on-chip SRAM, with memory access protection and detection
- 1 MB L2 share SRAM
- Chip ID

The following table is the memory map of on-chip ROM and SRAM.

Table 3-1. Memory map of on-chip memory controller

Bank	Start address	End address	Size	Device
0	0x0000_0000	0x0001_3FFF	80KB	ROM
	0x0010_0000	0x0012_FFFF	192KB	SRAM
	0x0040_0000	0x004F_FFFF	1MB	L2 Share SRAM

3.1.3 Block Diagram

The on-chip memory controller consists of a SRAM controller, a ROM controller, an AXI-FPC bus bridge, a bus interface unit, setting register and chip ID unit (see Figure 3-1). Detailed functionality is described in the following sections.

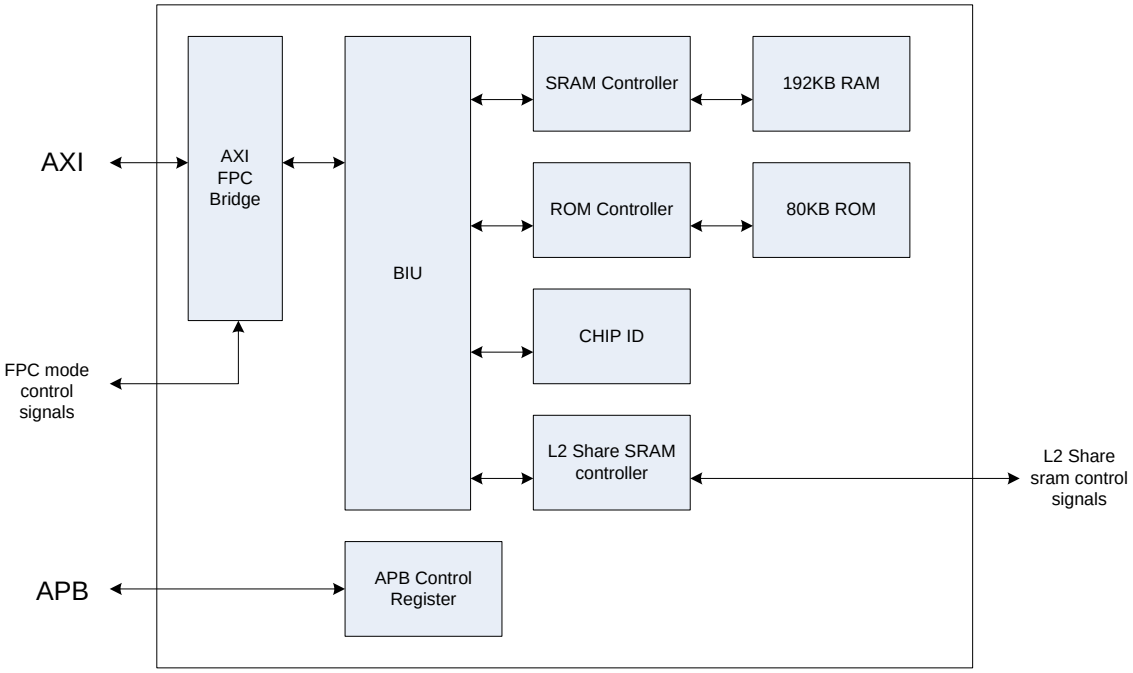


Figure 3-1. Block diagram of on-chip memory controller

3.1.3.1 BOOT ROM Power Down Mode

Boot ROM power down mode is used in the following scenarios:

1. After system boot, boot ROM will be powered down and prevented from any probe of ROM content
2. In MCDI (multi-core-deep-idle), it is the bootstrap for suspend/resume CPU.

The power down mode can be entered by setting *SRAMROM_SEC_CTRL.sramrom_sw_rom_pd* = 1. The bus interface unit will return a far jump instruction when receiving the read transactions. The jump address can be configurable by the *SRAMROM_BOOT_ADDR* register.

3.1.3.2 BOOT ROM FPC Mode

Boot ROM FPC mode is mainly used in Function Pattern mode. When the chip is trapped into FPC mode, the AXI-FPC bridge will block all the transactions to ROM address by returning a far jump instruction, with jump address specified in SRAMROM_FPC_BOOT_ADDR. The default value of SRAMROM_FPC_BOOT_ADDR is 0. The AXI-FPC bridge will automatically unblock the transaction when the FPC program is downloaded to SRAM memory address space.

3.1.3.3 On-Chip SRAM Security Protection

The on-chip SRAM can be partitioned into 2 regions with different security protection configurations. The bus interface unit performs permission check based on the settings, records the first violated address in the *SEC_VIO_ADDR* register and issues interrupts to the host processor. The violation interrupt can be cleared by a write to *SEC_VIO_ACK*.

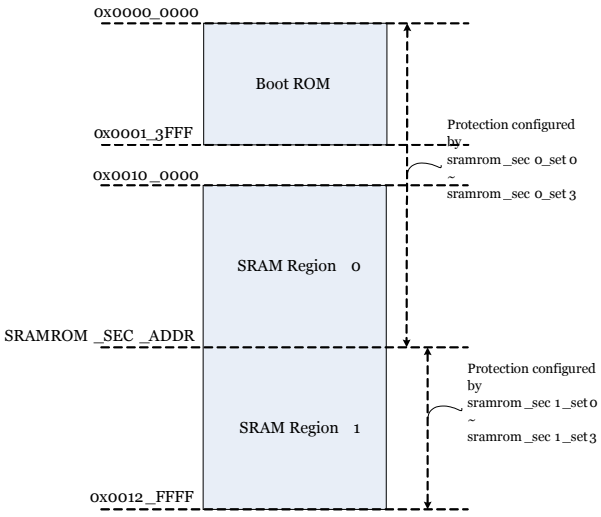


Figure 3-2. Security memory protection scheme

3.1.4 Register Definition

See chapter 1.1 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

3.2 External Interrupt Controller

3.2.1 Introduction

The external interrupt controller (EINTC) processes all off-chip interrupt sources and forwards interrupt request signals to AP MCU.

3.2.2 Features

EINTC supports up to 192 external interrupt signals and performs the following processes to the interrupt signals coming from external sources:

- Polarity inversion
- Edge/level trigger selection
- De-bounce with a configurable 32kHz clock (optional)

According to the register configuration, the external interrupt source will be forwarded to the Cortex-A7 built-in interrupt controller with different IRQ signals, eint_irq or eint_direct_irq. EINTC generates wakeup events to AP MCU.

3.2.3 Block Diagram

Figure 3-3 is the block diagram of the external interrupt controller in Denali. Every functional block is controlled by the corresponding control registers defined in next section.

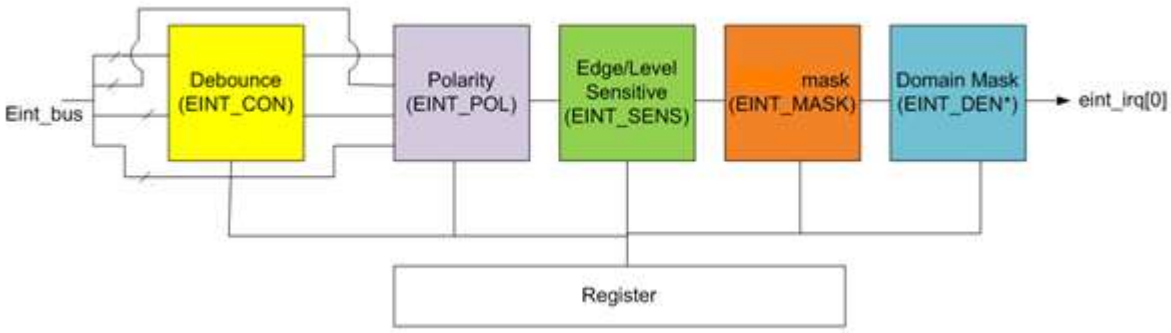


Figure 3-3. Block diagram of external interrupt controller

Figure 3-4 shows the secure view of eint module. It has several properties

- can secure wirite, global read.

- use secure_en to control security
- only GPIO0 ~ GPIO15 support security.
- if one is in secure world, the interrupt cannot be appeared at non-secure world.

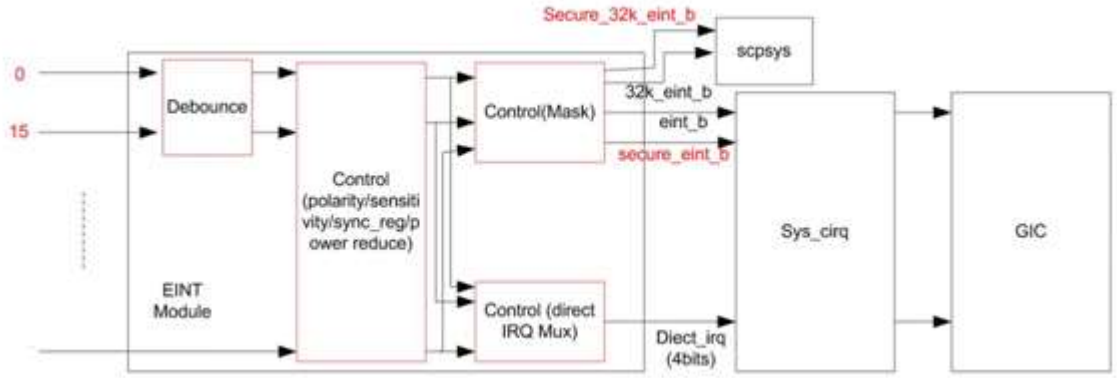


Figure 3-4. Secure view of external interrupt controller

Normally the external interrupt source goes through the de-bounce unit which is driven by 32kHz clock and triggers the corresponding CPU with eint_irq. Therefore, the minimum latency from eint_bus to eint_irq is 30.52μs.

The following tables list the signal connections to the interrupt controller of CPU.

Table 3-2. External interrupt request signal connection

IRQ name	AP MCU INTC
eint_irq	IRQ[202]
eint_irq_secure	IRQ[203]
eint_event_b	IRQ[204]
eint_event_secure_b	IRQ[205]
deint_irq[0]	IRQ[206]
deint_irq[1]	IRQ[207]
deint_irq[2]	IRQ[208]
deint_irq[3]	IRQ[209]

Table 3-3. Definitions of domains

Domain number	Target CPU/DSP
0	Application CPU

Table 3-4. EINT table

num	EINT	DEBOUNCE	Cannot use	Security
0	PAD_EINT0	v		v
1	PAD_EINT1	v		v
2	PAD_EINT2	v		v
3	PAD_EINT3	v		v
4	PAD_EINT4	v		v
5	PAD_EINT5	v		v
6	PAD_EINT6	v		v
7	PAD_EINT7	v		v
8	PAD_EINT8	v		v
9	PAD_EINT9	v		v
10	PAD_EINT10	v		v
11	PAD_EINT11	v		v
12	PAD_EINT12	v		v
13	PAD_EINT13	v		v
14	PAD_EINT14	v		v
15	PAD_EINT15	v		v
16	PAD_EINT16	v		
17	GPIO28			
18	GPIO29			
19	GPIO31			
20	GPIO32			
21	GPIO33			
22	GPIO34			
23	GPIO35			
24	GPIO36			
25	GPIO37			
26	GPIO38			
27	GPIO39			
28	GPIO40			
29	GPIO41			
30	GPIO42			
31	GPIO43			
32	GPIO44			
33	GPIO45			
34	GPIO46			
35	GPIO47			
36	GPIO48			
37	GPIO49			
38	GPIO50			

num	EINT	DEBOUNCE	Cannot use	Security
39	GPIO51			
40	GPIO52			
41	GPIO53			
42	GPIO54			
43	GPIO55			
44	GPIO56			
45	GPIO57			
46	GPIO58			
47	GPIO59			
48	GPIO60			
49	GPIO69			
50	GPIO70			
51	GPIO71			
52	GPIO72			
53	GPIO73			
54	GPIO74			
55	GPIO75			
56	GPIO76			
57	GPIO77			
58	GPIO78			
59	GPIO79			
60	GPIO80			
61	GPIO81			
62	GPIO82			
63	GPIO83			
64	GPIO84			
65	GPIO94			
66	GPIO95			
67	GPIO96			
68	GPIO97			
69	GPIO98			
70	GPIO100		x	
71	GPIO101		x	
72	GPIO104		x	
73	GPIO106			
74	GPIO107			
75	GPIO108			
76	GPIO109			
77	GPIO110			

num	EINT	DEBOUNCE	Cannot use	Security
78	GPIO111			
79	GPIO112			
80	GPIO113			
81	GPIO126			
82	GPIO127			
83	GPIO128			
84	GPIO129			
85	GPIO130			
86	GPIO131			
87	GPIO132			
88	GPIO133			
89	GPIO134			
90	GPIO135			
91	GPIO136			
92	GPIO137			
93	GPIO138			
94	GPIO139			
95	GPIO140			
96	GPIO141			
97	GPIO151			
98	GPIO152			
99	GPIO153			
100	GPIO154			
101	GPIO155			
102	GPIO156			
103	GPIO157			
104	GPIO179			
105	GPIO180			
106	GPIO181			
107	GPIO183			
108	GPIO184			
109	GPIO185			
110	GPIO186			
111	GPIO187			
112	GPIO188			
113	GPIO189			
114	GPIO190			
115	GPIO191			
116	GPIO192			
117	GPIO193			

num	EINT	DEBOUNCE	Cannot use	Security
118	GPIO194			
119	GPIO195			
120	GPIO196			
121	GPIO197			
122	GPIO198			
123	GPIO199			
124	GPIO200			
125	GPIO201			
126	GPIO202			
127	GPIO203			
128	GPIO204			
129	GPIO205			
130	GPIO206			
131	GPIO207			
132	GPIO208			
133	GPIO209			
134	GPIO210			
135	GPIO211			
136	GPIO212			
137	GPIO213			
138	GPIO214			
139	GPIO215			
140	GPIO216			
141	GPIO217			
142	GPIO218			
143	GPIO219			
144	GPIO232			
145	GPIO233			
146	GPIO234			
147	GPIO235			
148	GPIO236			
149	GPIO237			
150	GPIO238			
151	GPIO239			
152	GPIO240			
153	GPIO241			
154	GPIO242			
155	GPIO243			
156	GPIO244			
157	GPIO245			

num	EINT	DEBOUNCE	Cannot use	Security
158	GPIO246			
159	GPIO247			
160	GPIO248			
161	GPIO249			
162	GPIO250			
163	GPIO251			
164	GPIO252			
165	GPIO253			
166	GPIO254			
167	GPIO255			
168	GPIO256			
169	GPIO257			
170	GPIO258			
171	GPIO259			
172	GPIO260			
173	GPIO261			
174	RESERVED			
175	RESERVED			
176	PMIC_EINT_OUT[0]	v		
177	PMIC_EINT_OUT[1]	v		
178	RESERVED			
179	RESERVED			
180	C2K_WAKEUP_AP	v		
181	C2K_TO_AP_READY	v		
182	C2K_RST_IND	v		
183	C2K_SDIO_ACK_N	v		
184	C2K_SDIO_FCTRL_N	v		
185	RESERVED			
186	SSUSB_EINT_OUT[0]	v		
187	SSUSB_EINT_OUT[1]	v		
188	SSUSB_EINT_OUT[2]	v		
189	SSUSB_EINT_OUT[3]	v		
190	SSUSB_EINT_OUT[4]	v		
191	RESERVED			

3.2.4 Register Definition

See chapter 1.2 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

3.3 System Interrupt Controller

3.3.1 Introduction

For processors which have embedded interrupt controllers (GIC) in it, the part of MCUSYS should keep feeding clock and power to make the interrupt functional. However, due to power/leakage overhead introduced by higher clock ratio and deep submicron processes, reserving an always on (or frequently turned on) domain in MCUSYS has become power ineffective. The system interrupt controller (SYS_CIRQ) is a low power interrupt controller designed to work outside MCUSYS as a second level interrupt controller. With SYS_CIRQ, MCUSYS can be completely turned off to improve system power consumption without losing interrupts.

3.3.2 Features

SYS_CIRQ supports up to 218 interrupts which can configure following attributes individually.

- Polarity inversion
- Edge/level trigger selection

The 218 interrupts feed through SYS_CIRQ and connect to GIC in MCUSYS. When SYS_CIRQ is enabled, it will record the edge-sensitive interrupts and generate a pulse signal to CPU GIC when the flush command is executed.

3.3.3 Block Diagram

Figure 3-5 is the system level block diagram of the system interrupt controller.

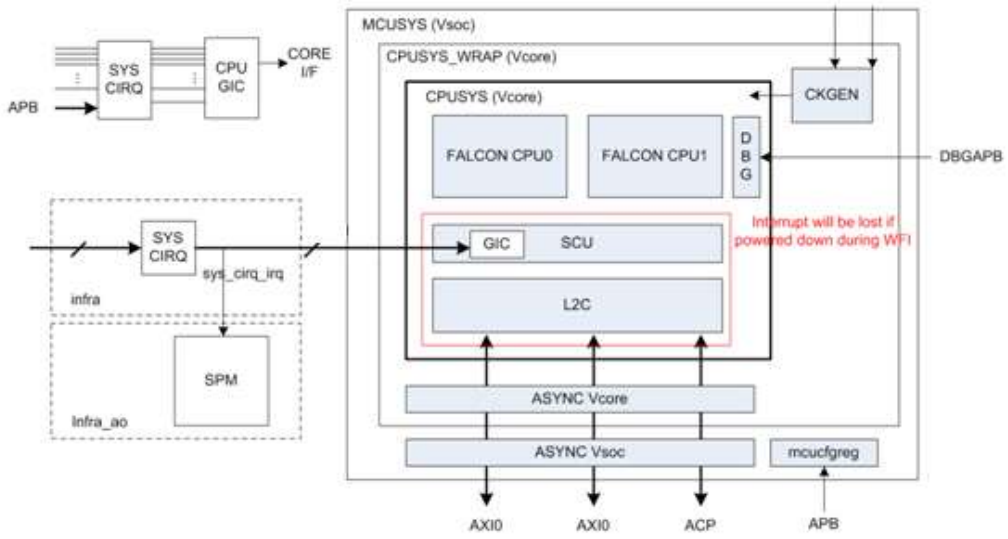


Figure 3-5. System level block diagram of system interrupt controller

The SYS_CIRQ controller is integrated in between MCUSYS and other interrupt sources as the second level interrupt controller. All interrupts are fed through SYS_CIRQ controller then bypassed to MCUSYS. In normal mode (where MCUSYS GIC is active), SYS_CIRQ is disabled and interrupts will be directly issued to MCUSYS. When MCUSYS enters the sleep mode, where GIC is power downed, the SYS_CIRQ controller will be enabled and monitor all edge-trigger interrupts (only edge-triggered interrupt will be lost in this scenario). When an edge-trigger interrupt is triggered, it will be recorded in SYS_CIRQ_STA register and can be restored to GIC by SW context restore or the SYS_CIRQ flush function.

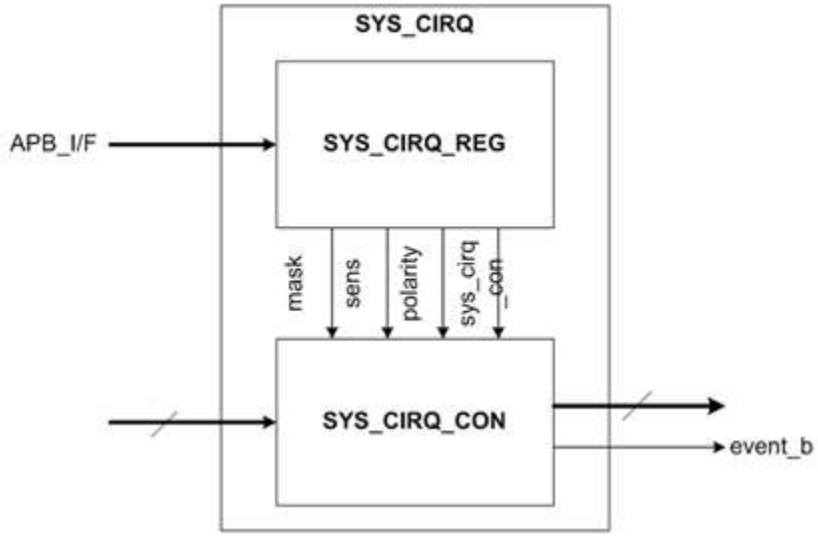


Figure 3-6. Block diagram of system interrupt controller

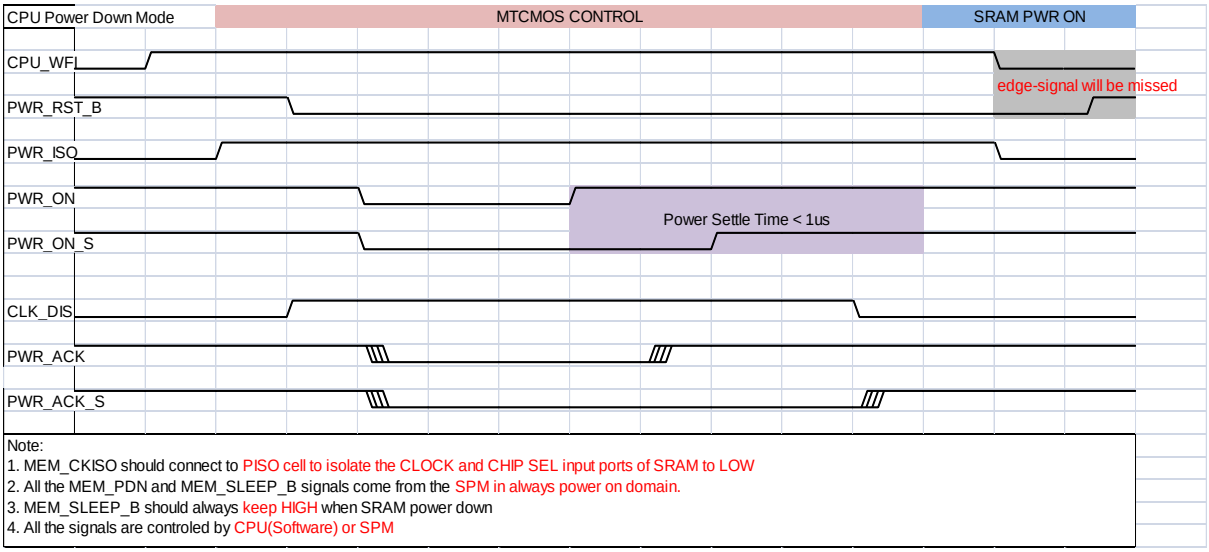
Figure 3-6 is the architecture of SYS_CIRQ. SYS_CIRQ_REG stores the mask/sensitivity/polarity attributes of each interrupt signal, and SYS_CIRQ_CON is used to mask and detect edge-triggered interrupts.

3.3.4 Register Definition

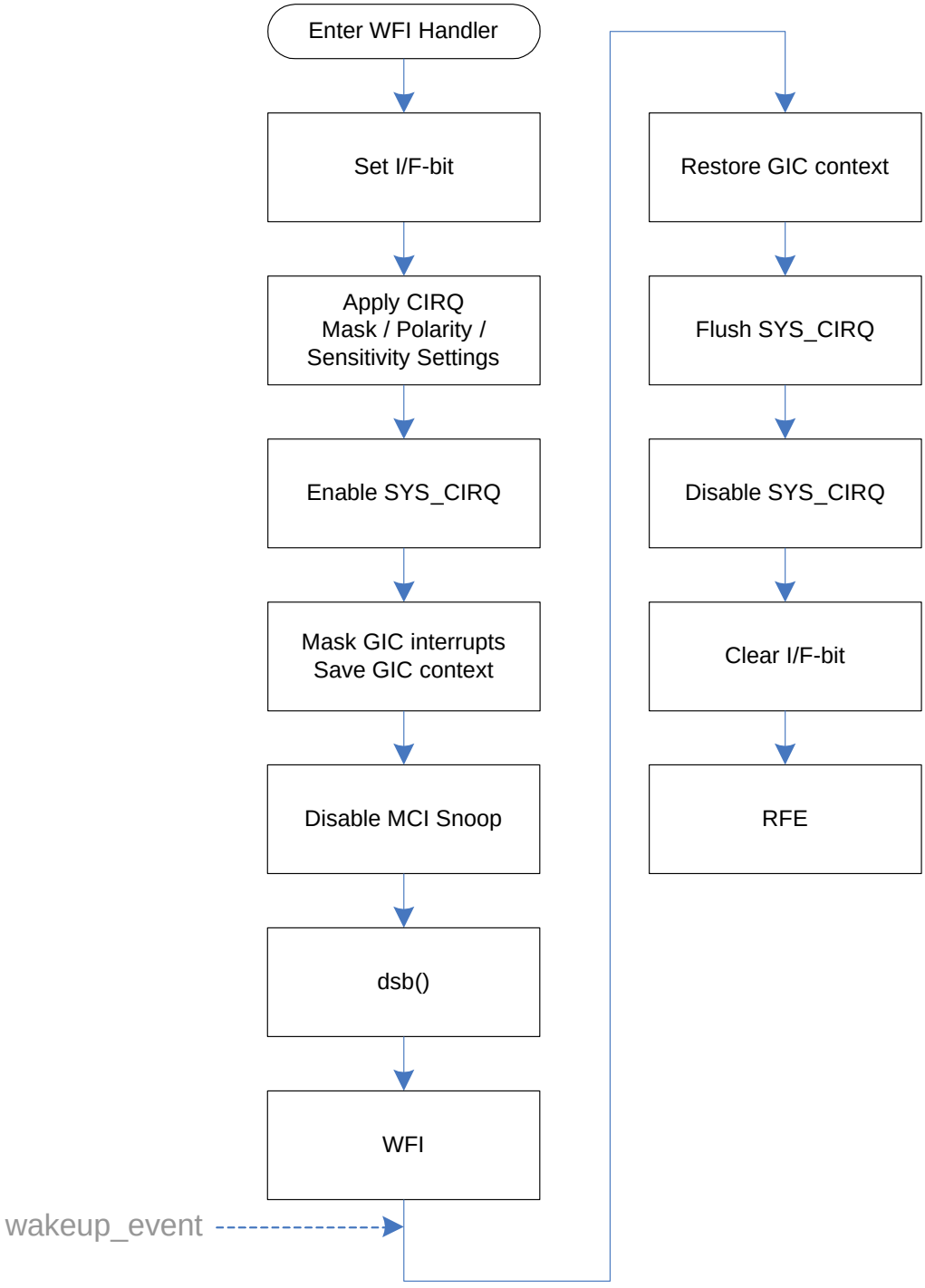
See chapter 1.3 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

3.3.5 Programming Guide

3.3.5.1 MCUSYS MTCMOS Sequence



3.3.5.2 SW Flow



3.4 Infrastructure System Configuration Module

3.4.1 Introduction

The infrastructure system configuration module (INFRACFG) provides reset, clock, bus-fabric and miscellaneous control signals in the infrastructure system.

3.4.2 Features

INFRACFG provides the following control signals to the functional blocks inside the infrastructure system:

- Software reset signals
- Clock gating control signals
- Dynamic clock management control signals
- Top AXI bus fabric control signals
- Dynamic clock management function
- MBIST control signals
- LDO control signals

3.4.2.1 DCM in Details

The Dynamic Clock Management (DCM) is a key technology to tradeoff between performance and power consumption for platform bus system. The bus clock can be slowed down or even gated off while the bus system is idle, depending on the configuration. The bus systems embedded with the DCM engine are infrasys bus fabric, peripheral bus fabric, and memory bus fabric. Figure 3-7 shows the concept of this scheme.

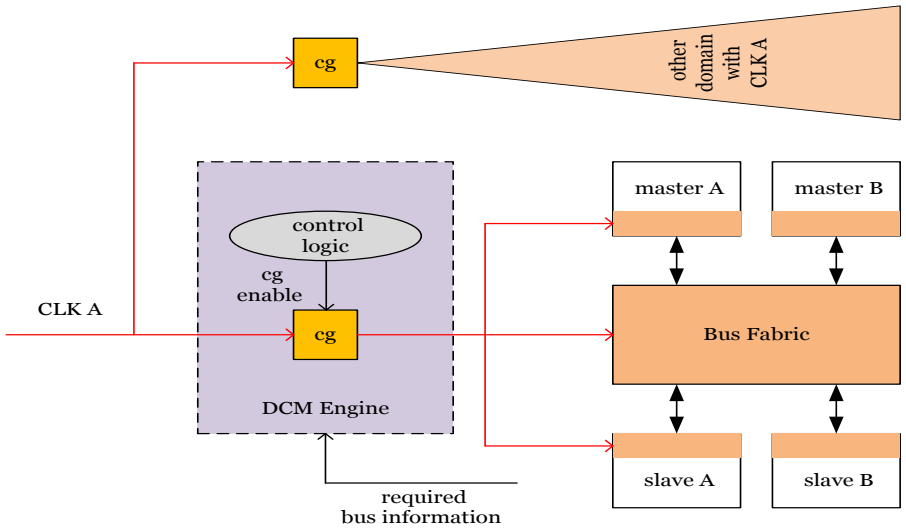


Figure 3-7. The proposed DCM scheme for bus low power consideration

3.4.2.2 AXI Fabric Control

The AXI fabric control registers help prevent the system bus from hanging up caused by improper access while some parts of the system are in the power-down state. See the figure Figure 3-8 for the location of SI nodes and the sleep protector and find the corresponding control registers in section 3.4.3 "Register Definition".

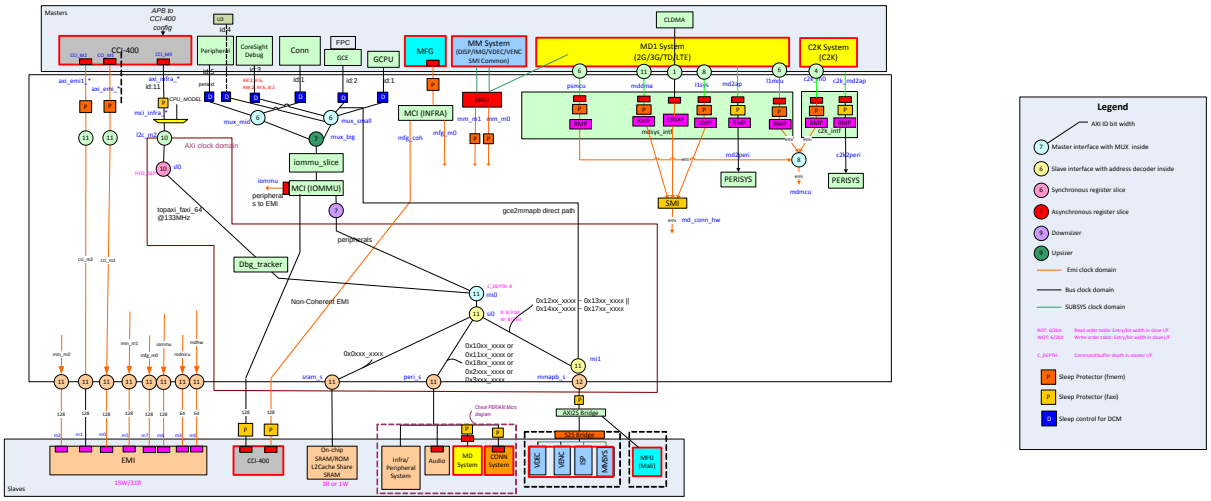


Figure 3-8. Top AXI fabric and control blocks

3.4.2.3 LDO Control

LDO supply power to SRAM bit-cell of CPU and GPU. Figure 3-9 shows the logic of this scheme.

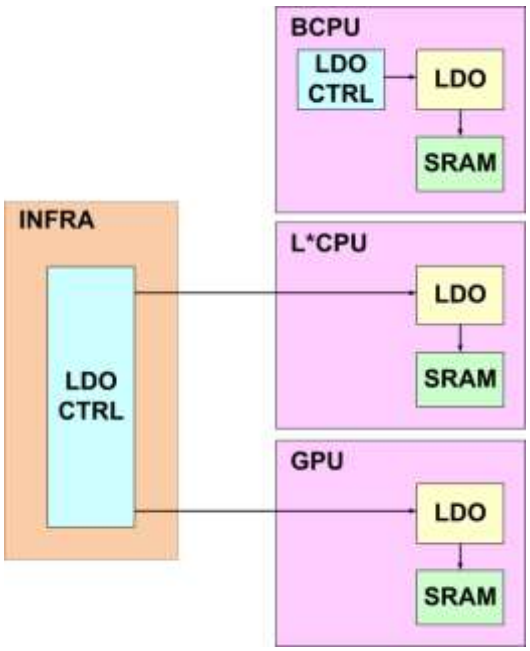


Figure 3-9. LDO control logic

3.4.3 Register Definition

See chapter 1.4 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

Register bank	Base address
infracfg_ao_reg	+10001000h
INFRA_AO_MBIST	+10001000h
infracfg_reg	+10201000h
INFRA_MBIST	+1020d000h
ldoctl	+10001000h

3.5 External Memory Interface

3.5.1 Introduction

The EMI (External Memory Interface) controller schedules requests from the masters and issues commands to DRAMC in an efficiency way. The block conducts flow control for DRAMC and masters to avoid DRAM stall or data overflow or underflow. It also minimizes the latency of processor path to enhance the performance and tries to increase the DRAM efficiency. The block also informs clock control to gate the clock when it does not find any transaction right now. This block is designed to supply 466MHz bus clock frequency and can be scaled down to 400MHz and 333MHz.

3.5.2 Features

The EMI controller receives AXI master commands and issues them to the DRAM controller. It supports all AXI transaction type commands except for the fixed and cache commands. There are plenty of schedule options to schedule the command, which are:

- Starvation control
- Bandwidth limiter
- High priority
- Page hit control
- Read/write turn around prevent control
- Memory protect unit (DRAM & APB)

3.5.3 Block Diagram

In MT6797, the DRAM controller connects three systems via eight AXI ports and supports connecting two channel DRAM controllers. Each channel supports two rank DRAM devices at the same time. For cortex APMCU system, two 128-bit AXI ports are provided for the connection. For the multimedia system, two 128-bit AXI ports are provided for the connection. For the GPU, a 128-bit AXI port is provided for the connection. For the peripherals, a 128-bit AXI port is provided for the connection. Besides, there are two 64-bit AXI ports for connecting to modem MCU, and modem 2G/3G/4G HW. In the DRAM controller, the APB interface is for programming registers. Then you can initialize DRAM or other parameter settings.

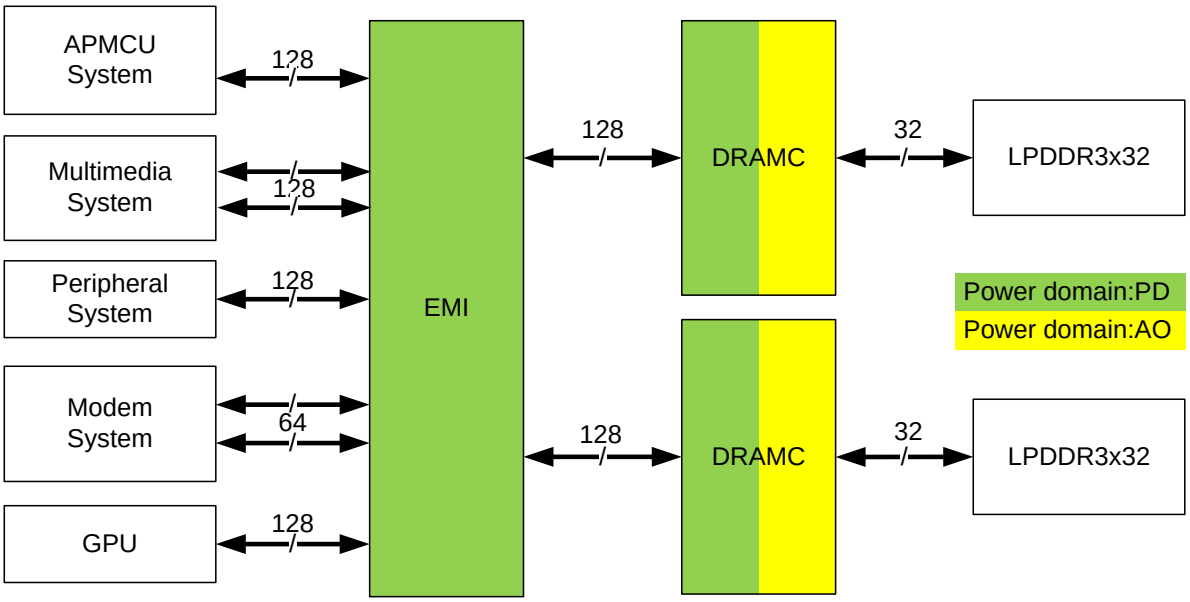


Figure 3-10. EMI/DRAM controller top connection

3.5.4 Register Definition

See chapter 1.5 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

Register bank	Base address
EMI_MPU_REG	+10200000h
EMI_REG Base address	+10203000h

3.6 AP_DMA

3.6.1 Introduction

There is always a DMA in a platform. The purpose of DMA is performing data transfer between different slaves. There are several slaves in a platform, and the major one is external memory, e.g. DRAM. There are also internal SRAM and some slave ports for the peripheral to transfer data. For saving software efforts, DMA delivers a virtual FIFO concept to help the software maintain read and write pointer when the software accesses data from a ring buffer. As the bus goes more and more efficient, the old DMA still utilizes the AHB bus protocol and may decrease its performance. Another problem is that when the old DMA meets byte alignment addresses or byte alignment sizes, it will need some software efforts to help solve head and tail non word alignment problems or let DMA to simply issues single-1-byte requests to conquer the byte-alignment problem. This will harm the overall system because the single-1-byte transaction is quite inefficient. The DMA efficiency is now improved by increasing its bus efficiency, including data buffering and overcoming byte alignment problems.

3.6.2 Features

APDMA has the following DMA engines.

- I2C DMA engine*10
- BTIF TX DMA engine*1
- BTIF RX DMA engine*1
- UART TX DMA engine*4
- UART RX DMA engine*4
- HIF DMA engine*1

The DMA engines and corresponding peripheral devices are listed below.

Table 3-5. Relationship between engines and devices

Engine	Peripheral device
HIF0	Connsys(sdctl)
I2C_0 ~ I2C_9	I2C_0 ~ I2C_9
UART0 ~ UART3	UART0 ~ UART3
BTIF	BTIF

3.6.3 Block Diagram

Figure 3-11 is the basic block diagram of AP_DMA. There are total 12 channels in DMA. The external AXI interface is connected to the peripheral AXI bus fabric to provide external memory access ability. The internal AXI interface is also connected to the peripheral AXI bus fabric and is re-directed to

related peripherals, e.g. HIF, I2C, BTIF and UART. A memory block is used as a buffer which makes the transfer on the AXI bus interface more efficient. An APB interface is used to program registers for both global registers and local registers existing in every individual DMA channel.

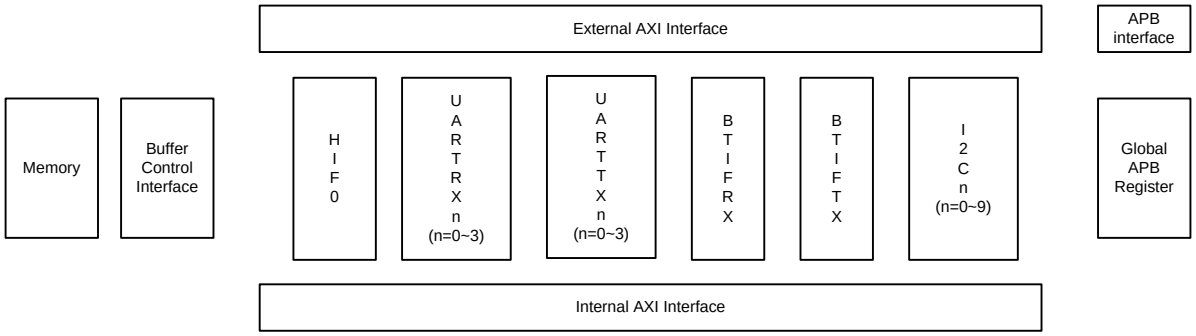


Figure 3-11. APDMA block diagram

3.6.4 Programming Guide

3.6.4.1 Warm Reset and Hard Reset

Warm reset and hard reset exist in DMA global control and each DMA engine. When warm reset is set, the engine will be reset after the current transaction is finished, and therefore the warm reset will not cause any bus hang. Conversely, when hard reset is set, the engine will be reset immediately, and therefore the bus may go down due to unfinished (and will never finish) transaction.

Mechanism of global warm reset

When the software is to re-start all engines or re-clear all engines in DMA, it can set the global WARM_RST to 1 and wait for (poll) all global running status to be 0. Next, set WARM_RST back to 0 to finish the global warm reset.

Mechanism of global hard reset

When the software is to re-start all engines or re-clear all engines in DMA without waiting for any period of time, it can set the global HARD_RST to 1 then back to 0 to finish the global hard reset. Note that this may break the bus protocol and cause system hang.

3.6.4.2 Pause and Resume

There are pause and resume functions available for general DMA and peripheral DMA. The mechanism is as the following:

1. Start DMA. (Program necessary settings, then set EN = 1.)
2. Pause DMA. (Set PAUSE = 1.)

3. Resume DMA. (Set PAUSE = 0.)
4. Wait for DMA to finish. (EN will become 0, and flag will be set to 1.)

The software can repeat step 2 and 3 many times when DMA is running and monitor the idle bit to see if DMA is stopped. DMA will not pause immediately and will wait for the last transaction to be finished.

3.6.4.3 Access Security Region

To access the memory security region, the security and domain bits must be specified before channel configuration and DMA is enabled. After SEC_EN and GSEC are set, access DMA registers with APB secure access. When DMA finishes the transmission, the secure and domain bits will become default value. Configure the bits again if DMA transmission is going to start.

1. Set register DOMAIN_CFG as the AP domain.
2. Set register SEC_EN=1 and GSEC=1.

3.6.5 Programmed Sequence for Different Types of Channels

3.6.5.1 Peripheral DMA with Burst Length equals 1 (e.g. I2C)

1. Configure DMA registers.
   AP_DMA_I2C_*_CON (dir)
   AP_DMA_I2C_*_TX_MEM_ADDR, AP_DMA_I2C_*_TX_LEN (TX)
   AP_DMA_I2C_*_RX_MEM_ADDR, AP_DMA_I2C_*_RX_LEN (RX)
2. Set interrupt enable=1 .
   AP_DMA_I2C_*_INT_EN
3. Wait for interrupt.
4. Clear interrupt flag.
   AP_DMA_I2C_*_INT_FLAG

3.6.5.2 Peripheral DMA with Configurable Burst Length (e.g. HIF)

1. Configure DMA registers.
   AP_DMA_HIF_*_CON (dir, burst_length)
   AP_DMA_HIF_*_MEM_ADDR
   AP_DMA_HIF_*_LEN
2. Set interrupt enable=1 .
   AP_DMA_HIF_*_INT_EN
3. Set enable=1.
   AP_DMA_HIF_*_EN

4. Wait for interrupt.
5. Clear interrupt flag.
   AP_DMA_HIF_*_INT_FLAG

3.6.5.3 Virtual FIFO DMA TX (e.g. UART_TX, BTIF_TX)

1. Configure registers.
   AP_DMA_UART_*_TX_VFF_ADDR
   AP_DMA_UART_*_TX_VFF_LEN, AP_DMA_UART_*_TX_VFF_THRE
   AP_DMA_UART_*_TX_VFF_WPT
2. Write data to EMI and update SW write_pointer.
   AP_DMA_UART_*_TX_VFF_WPT
3. Clear interrupt (repeat step 3-6 till finished).
   AP_DMA_UART_*_TX_INT_FLAG
4. Set interrupt enable =1.
   AP_DMA_UART_*_TX_INT_EN
5. Set enable=1 (first time).
   AP_DMA_UART_*_TX_EN
6. Wait for interrupt.
7. Set stop=1 if finished.
   AP_DMA_UART_*_STOP

3.6.5.4 Virtual FIFO DMA RX (e.g. UART_RX, BTIF_RX)

1. Configure registers.
   AP_DMA_UART_*_RX_VFF_ADDR
   AP_DMA_UART_*_RX_VFF_LEN, AP_DMA_UART_*_RX_VFF_THRE
   AP_DMA_UART_*_RX_VFF_RPT
   AP_DMA_UART_*_RX_FLOW_CTRL_THRE
2. Set interrupt enable =1.
   AP_DMA_UART_*_RX_INT_EN
3. Set enable=1 (first time).
   AP_DMA_UART_*_RX_EN
4. Wait for interrupt (repeat step 4-6 till finished).
5. Read data from EMI; update SW read_pointer.
   AP_DMA_UART_*_RX_VFF_RPT
6. Clear interrupt flag.
   AP_DMA_UART_*_RX_INT_FLAG
7. Set stop=1 if finished.
   AP_DMA_UART_*_RX_STOP

3.6.6 Register Definition

3.6.6.1 Global Control Registers

There are several registers put together for the software to monitor. However, if the software is to change them, they can only be written through individual DMA registers. The global register is only for the software to watch all DMA running statuses and interrupt flags together. Note that the security ability of all global registers belongs to the GSEC_EN bit. When this bit is set to 1, only the security transaction can write the global register (global reset and global slow-down). If a non-security read transaction is issued to read the interrupt flag or running status, only statuses of non-security engines can be reported, and others will always be 0.

3.6.6.2 Register Definition

See chapter 1.6 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

4 Clock and Power Control

4.1 Top Clock Generator

4.1.1 Introduction

This chapter introduces the top clock generator (TOPCKGEN) and the clock architecture.

4.1.2 Features

TOPCKGEN is responsible for generating the following clock signals:

- Free clock generation for whole chip
- Infrastructure and peripheral system clock, including the top level AXI fabric clock
- Multimedia system clock
- Pad macro clocks to be synchronized with one of the above system

The module TOPCKGEN provides clock source selection. Each clock has several clock source selection and can be turned off as well. When switching certain clock from frequency A to frequency B, make sure frequency A and B are available.

It comprises glitch-free clock MUX and digital clock divider to generate various clock frequencies.

4.1.3 Block Diagram

4.1.3.1 Clock Architecture

There are clock generators not only in the top level hierarchy but also in every partition/system. Figure 4-1 shows the location of the top level clock generator.

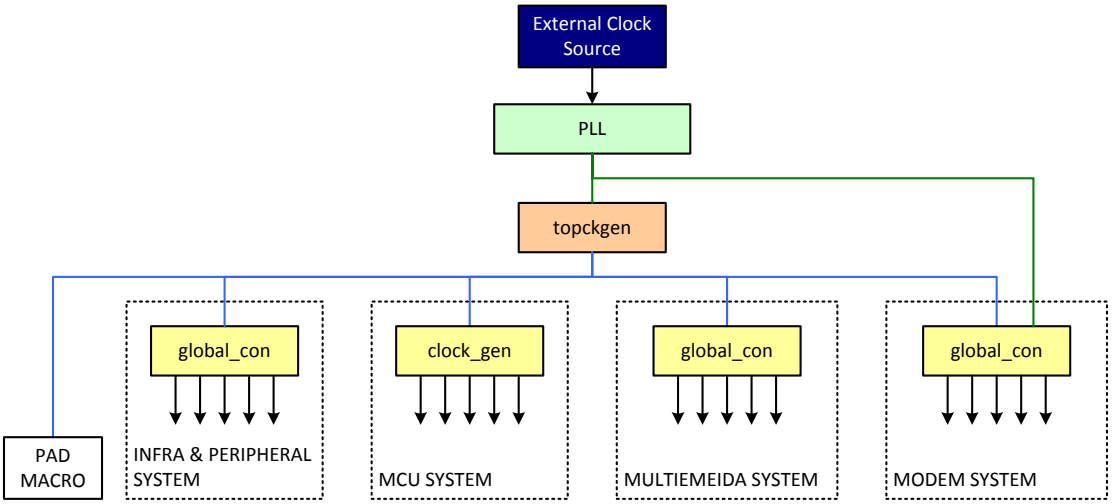


Figure 4-1. Block diagram of clock architecture

4.1.3.2 Clock Multiplier

Clock selection and generation have similar structure (see Figure 4-2). Several clock sources are provided. Choose one by specified register setting. The turn-off bit is provided as well to stop the clock output.

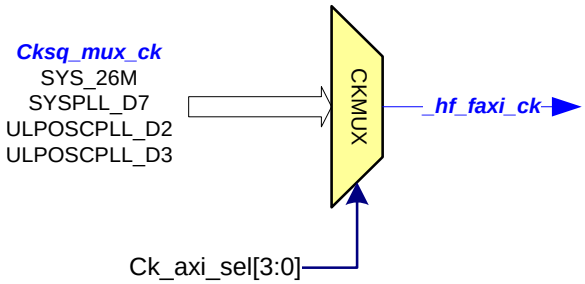


Figure 4-2. Example of clock multiplier

4.1.4 Clock PLL

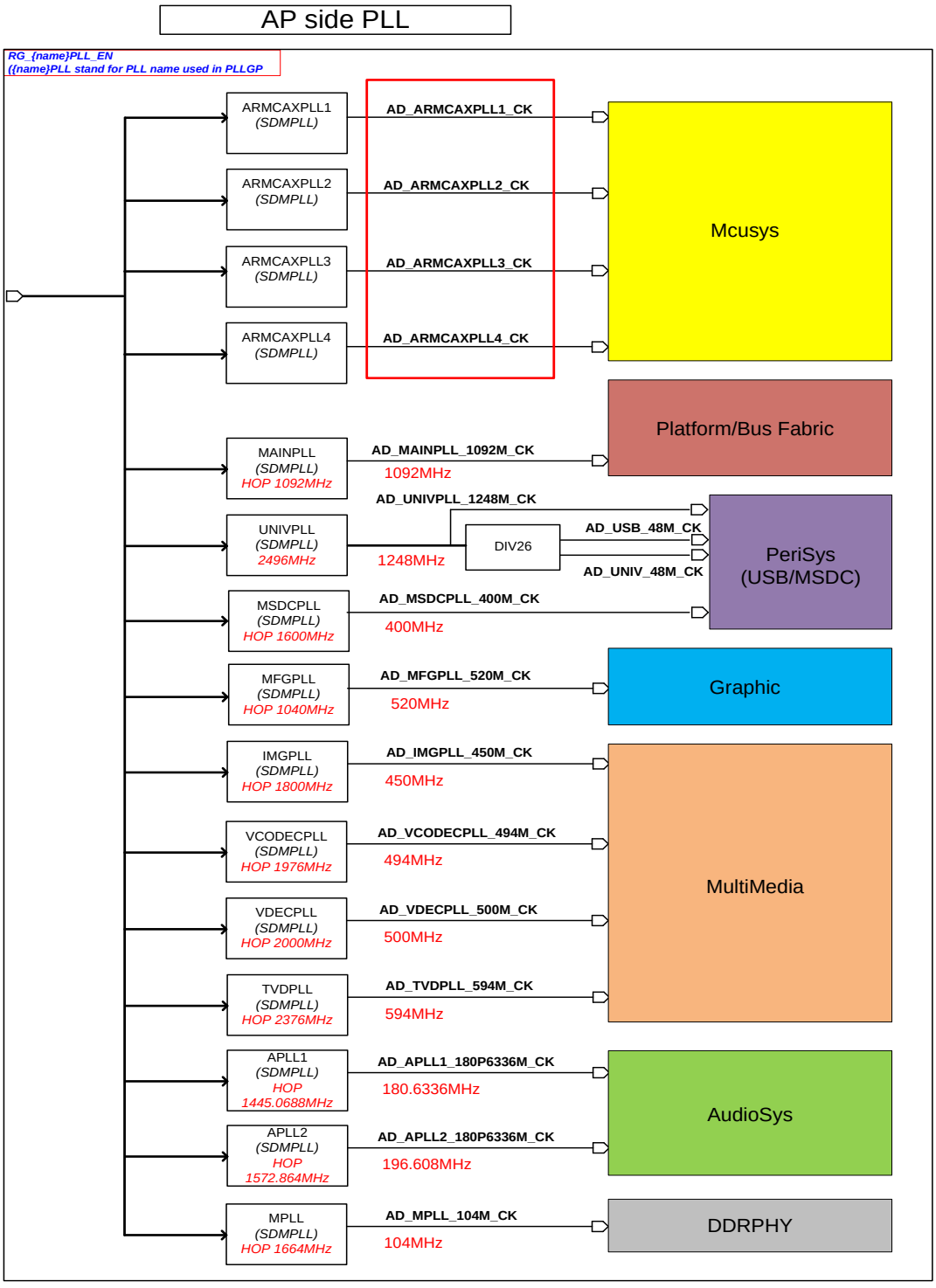


Figure 4-3. PLL block diagram

4.1.5 PLL Related Control

The following table lists all PLLs inside the application system.

The enabling of PLL can be switched between software control and hardware control. The hardware control is from SCPSYS.

The hopping and SSC features can be switched between software control and hardware control. The hardware control is from FHCTL.

Table 4-1. PLL related control

PLL	Capability	Control by FHCTL	Control by SPM
ARMCAXPLL1	Hopping, SSC	Y	N
ARMCAXPLL2	Hopping, SSC	Y	N
ARMCAXPLL3	Hopping, SSC	Y	N
ARMCAXPLL4	Hopping, SSC	Y	N
MAINPLL	Hopping, SSC	Y	Y
UNIVPLL	Fix	N	Y
MSDCPLL	Hopping, SSC	Y	N
MFGPLL	Hopping, SSC	Y	N
IMGPLL	Hopping, SSC	Y	N
VCODECPLL	Hopping, SSC	Y	N
VDECPLL	Hopping, SSC	Y	N
TVDPLL	Hopping, SSC	Y	N
MPLL	Hopping, SSC	Y	N
APLL1	Fix	N	N
APLL2	Fix	N	N
MIPI	SSC	N	N
MEMPLL	Hopping, SSC	Y	N

4.1.6 Clock Gating

The clock gating for module TOPCKGEN is listed in the table below where DCM and turn-off settings are provided.

Table 4-2. Clock gating settings

Register name	Bit	Default	Function name	Description
CLK_MODE	[8]	1'b0	pdn_md_32k	Turns off 32K clock source to MD
CLK_SCP_CFG_0	[0]	1'b0	sc_26ck_off_en	Turns on scpsys control path to gate 26MHz
	[1]	1'b0	sc_mem_ck_off_en	Turns on scpsys control path to gate

Register name	Bit	Default	Function name	Description
				DDRPHY
	[2]	1'b0	sc_axick_off_en	Turns on scpsys control path to gate hf_faxi_ck
	[5]	1'b0	sc_md_32k_off_en	Turns on scpsys control path to gate MD 32KHz
	[9]	1'b0	sc_mac_26m_off_en	Turns on scpsys control path to gate MIPI 26MHz
CLK_SCP_CFG_1	[0]	1'b0	sc_axi_26m_sel_en	Turns on scpsys control path to switch hf_faxi_ck to 26MHz
	[2]	1'b0	sc_scpck_26m_sel_en	Turns on scpsys control path to switch hf_fscp_ck to 26MHz

4.1.7 Frequency Meter

There is one frequency meter inside TOPCKGEN. There are two input clock sources: one is for PLLs and TEST clock, the other is for clocks generated from TOPCKGEN.

It has PAD output that can observe frequency directly instead of reading results from the frequency meter through DEBUG_MON[0].

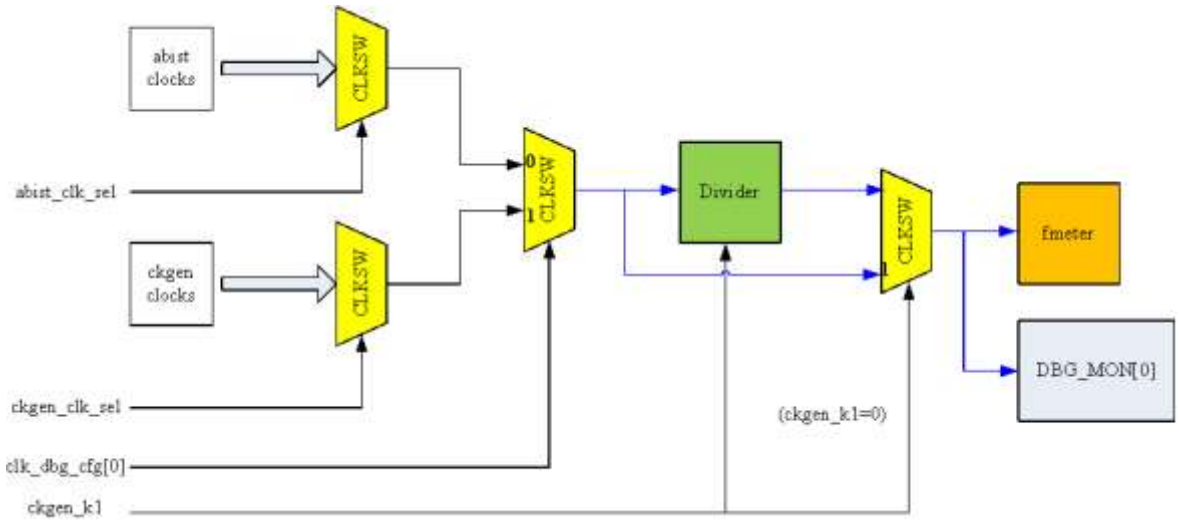


Figure 4-4. TOPCKGEN FMETER structure

4.1.8 Register Definition

See chapter 2.1 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

4.1.9 Programming Guide

4.1.9.1 Clock Off

- The clock can be turned on/off through changing the value of pdn_*. However, this control cannot be switched along with clk_*_sel and clk_*_inv.
- It is recommended to change pdn_* with SET and CLEAR function provided by CLK_CFG_*_SET and CLK_CFG_*_CLR. Because there may be clock with multi-bit pdn_* which are planned to avoid read modify write from different sub-systems (APSYS, MDSYS) at the same time.
- SET and CLEAR function of CLK_CFG_* is a solution to avoid read modify write from different sub-systems.

4.1.9.2 Clock Switching

- Make sure clock A and clock B are available before changing the setting of clk_*_sel. If switched to a non-exist clock, the clock switch will be stuck until non-exist clock is turned on to free the clock switch.
- Supports multi-clock switching at the same time (without changing pdn_*)

Switching from clock A to clock B

1. Make sure clock B is ready.
2. Change clk_*_sel.
3. Write *_ck_update
4. Wait until chg_sta = 1'b0 (optional).
5. Turn off clock A (optional).

4.1.9.3 Switch AXI to 26MHz by SCPSYS

The reflection time is about 17T 26MHz if all clocks are counted as 26MHz. Refer to the following formula:

$$4T\ Bus\ Clock_{(*1)} + 4T\ Current\ Clock_{(*2)} + 5T\ Reference\ Clock_{(*3)} + 1T\ Bus\ Clock_{(*4)} + 3T\ Target\ Clock_{(*5)}$$

Comment	Description
Bus clock	26MHz (in current project)
Ref clock	26MHz, balance with bus clock
*1	2T Sync + 1 T Control
*2	3T sync
*3	4T sync
*4	1T control. For async CLKSW like hf_fmem_ck

Comment	Description
	used, it will be 2T sync.
*5	2T sync

4.1.9.4 Frequency Meter

There are two frequency meters embedded inside TOPCKGEN.

1. Set fmeter_en to 1'b1.
2. Choose frequency meter source by clk_dbg_cfg[0]
3. Choose target clock by changing abist_clk_sel / ckgen_clk_sel.
4. Change ckgen_k1 for dividing target clock (optional).
5. Change reference clock by changing clk_exc (optional).
6. Change ckgen_load_cnt (optional).
7. Trigger frequency meter by set ckgen_tri_cal = 1b'1.
8. Wait until ckgen_tri_cal = 1'b0.
9. Read frequency meter result from ckgen_cal_cnt.
   - freq(target) = (ckgen_k1 + 1)*[freq(reference clock)*ckgen_cal_cnt]/(ckgen_load_cnt + 1)

4.2 Top Reset Generator Unit

4.2.1 Introduction

The top reset generator unit (TOPRGU) generates reset signals and distributes to each system. A watchdog timer is also included in this module.

4.2.2 Features

- Hardware reset signals for the whole chip
- Software controllable reset for each system (except for infrastructure and apmixedsys system)
- Watchdog timer
- Reset output signals for companion chips

4.2.3 Block Diagram

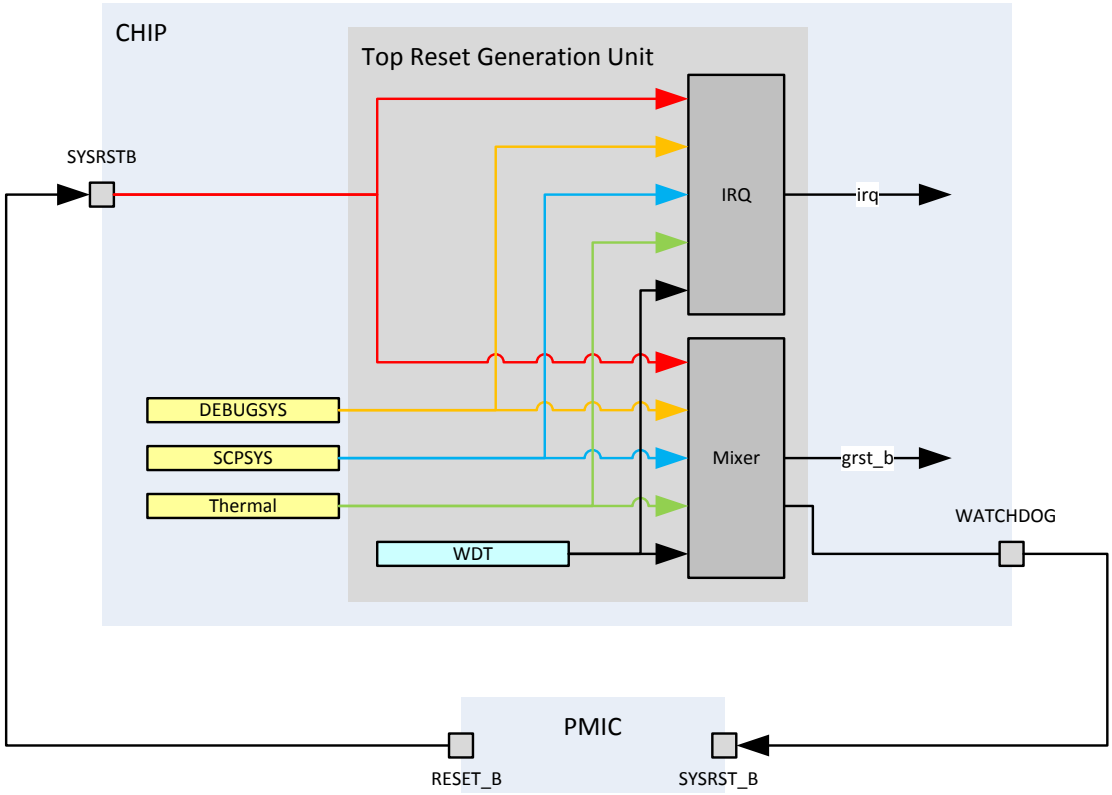


Figure 4-5. Block diagram of top reset generation unit

4.2.4 Register Definition

See chapter 2.2 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

4.3 PMIC Wrapper

4.3.1 Introduction

The PMIC wrapper is the bridge for communication between AP and PMIC.

4.3.2 Feature

- Fast auto SPI format generator for PMIC register read/write
- APB3.0 bus lock scheme when SPI is busy
- Manual SPI format generator
- Supports access to dual PMICs
- Single and dual I/O SPI mode support for PMIC
- Single IO mode support only for Switching Charger
- Separated frequency between controller and SPI

4.3.3 Block Diagram

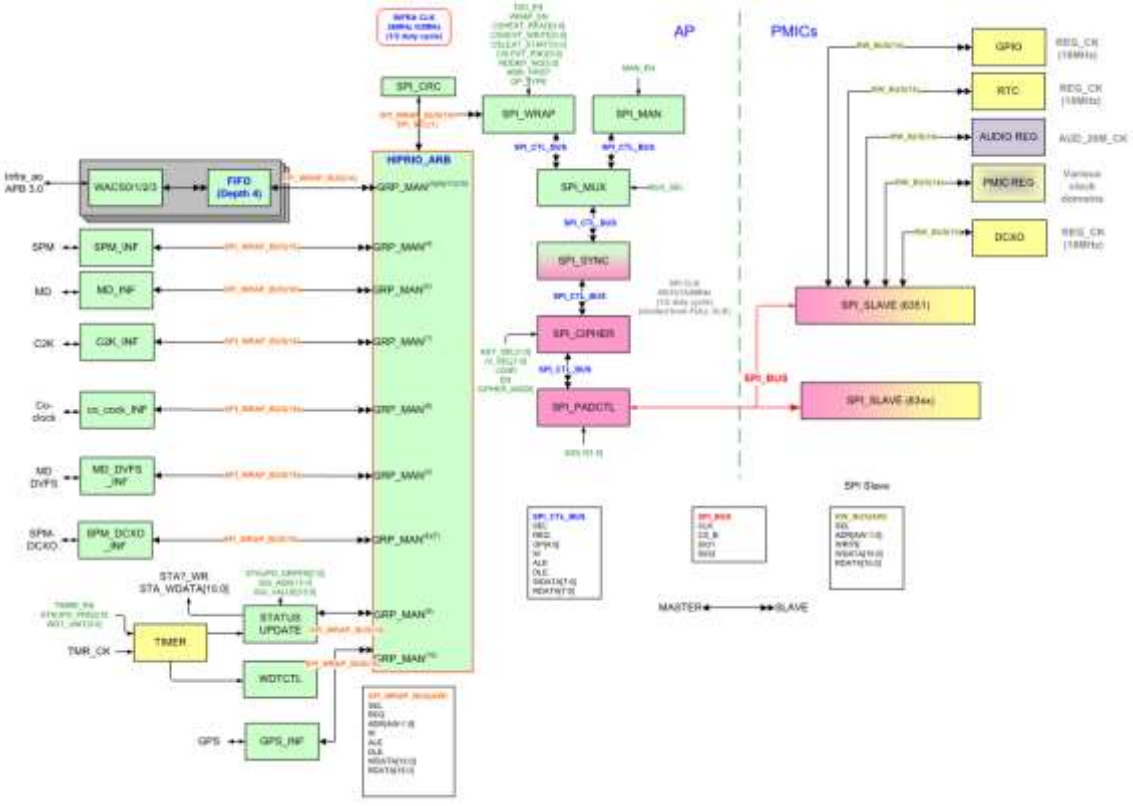


Figure 4-6. PMIC_WRAP architecture

4.3.4 Register Definition

See chapter 2.3 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

Register bank	Base address
PMIC_WRAP	+1000D000h
PMIC_WRAP_P2P	+100AB000h

5 Peripherals

5.1 Pericfg Controller

5.1.1 Introduction

The pericfg controller controls the clock and bus setting of peripheral subsys. The hardware DCM (Dynamic Clock Management) of the peripheral subsys is also controlled in the pericfg controller.

5.1.2 Features

- Supports DCM control of peripheral subsys
- Supports bus setting (bandwidth limit/way enable/...) of peripheral subsys

5.1.3 Block Diagram

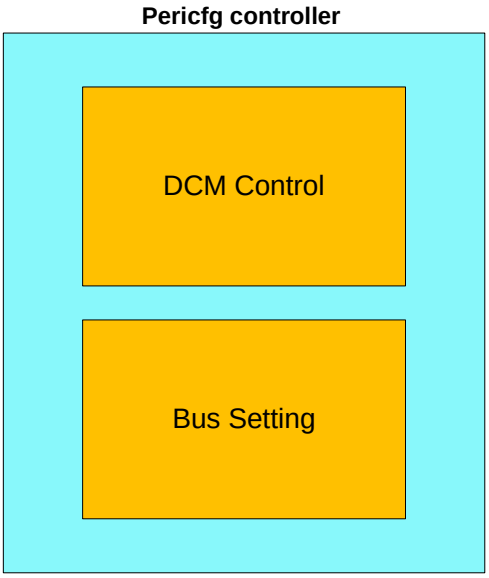


Figure 5-1. Block diagram of pericfg controller

5.1.4 Register Definition

See chapter 3.1 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.2 GPIO Control

5.2.1 General Descriptions

There are 222 I/O pins can be programmed as multiple purpose, including GPIO, NAND, SPI, etc. By setting up the GPIO_MODE register, specific IO is selected for specific function.

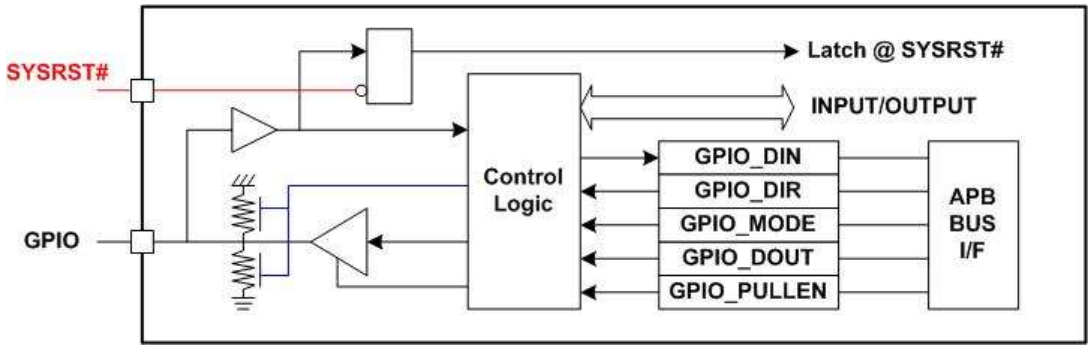


Figure 5-2. GPIO block diagram

All functions should comply with the priority rule. When there are more than one IO set as the same output function, all of the selected IOs are able to output specific signals. When there are more than one IO set as the same input (or bi-directional) function, only the IO with the largest GPIO index works functionally.

Table 5-1. Summary of configuration register bases of GPIOs

Register bank	Base address
GPIO_BASE	0x10005000
IOCFG_L_BASE	0x10002000
IOCFG_B_BASE	0x10002400
IOCFG_R_BASE	0x10002800
IOCFG_T_BASE	0x10002C00

Table 5-2. Summary of configuration registers of GPIOs

Name	IES	SMT	PU	PD	PUPD
PAD_ANC_DAT_MOSI	IO_CFG_R_BASE+ 0x0000[19:11]	IO_CFG_R_BASE+ 0x0030[12:7]	IO_CFG_R_BASE+ 0x00D0[18:10]	IO_CFG_R_BASE+ 0x00B0[18:10]	-
PAD_AUD_CLK_MOSI	IO_CFG_R_BASE+ 0x0000[19:11]	IO_CFG_R_BASE+ 0x0030[12:7]	IO_CFG_R_BASE+ 0x00D0[18:10]	IO_CFG_R_BASE+ 0x00B0[18:10]	-
PAD_AUD_DAT_MISO	IO_CFG_R_BASE+ 0x0000[19:11]	IO_CFG_R_BASE+ 0x0030[12:7]	IO_CFG_R_BASE+ 0x00D0[18:10]	IO_CFG_R_BASE+ 0x00B0[18:10]	-

Name	IES	SMT	PU	PD	PUPD
PAD_AUD_DAT_MOSI	IO_CFG_R_BASE+0x0000[19:11]	IO_CFG_R_BASE+0x0030[12:7]	IO_CFG_R_BASE+0x00D0[18:10]	IO_CFG_R_BASE+0x00B0[18:10]	-
PAD_AUD_INTN	IO_CFG_B_BASE+0x0020[16:12]	IO_CFG_B_BASE+0x0060[6:4]	IO_CFG_B_BASE+0x0180[12:8]	IO_CFG_B_BASE+0x0150[6:2]	-
PAD_AUD_PDN	IO_CFG_B_BASE+0x0010[23:21]	IO_CFG_B_BASE+0x0050[22:21]	IO_CFG_B_BASE+0x0170[11:9]	IO_CFG_B_BASE+0x0140[15:13]	-
PAD_BPI_BUS0	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS1	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS10	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS11	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS12_ANT0	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS13_ANT1	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS14_ANT2	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS15_ANT3	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS16_VM0	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS17_VM1	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS18_SWP0	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS19_SWP1	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS2	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS20_SWP2	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS21_SWP3	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS22_DET0	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS23_DET1	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS3	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS4	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-
PAD_BPI_BUS5	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS6	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS7	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_BPI_BUS8	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-

Name	IES	SMT	PU	PD	PUPD
PAD_BPI_BUS9	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_CAM_CLK0	IO_CFG_L_BASE+0x0000[9:0]	IO_CFG_L_BASE+0x0040[4:0]	IO_CFG_L_BASE+0x00B0[9:0]	IO_CFG_L_BASE+0x00D0[9:0]	-
PAD_CAM_CLK1	IO_CFG_L_BASE+0x0000[9:0]	IO_CFG_L_BASE+0x0040[4:0]	IO_CFG_L_BASE+0x00B0[9:0]	IO_CFG_L_BASE+0x00D0[9:0]	-
PAD_CAM_CLK2	IO_CFG_L_BASE+0x0000[9:0]	IO_CFG_L_BASE+0x0040[4:0]	IO_CFG_L_BASE+0x00B0[9:0]	IO_CFG_L_BASE+0x00D0[9:0]	-
PAD_CAM_PDN1	IO_CFG_L_BASE+0x0000[9:0]	IO_CFG_L_BASE+0x0040[4:0]	IO_CFG_L_BASE+0x00B0[9:0]	IO_CFG_L_BASE+0x00D0[9:0]	-
PAD_CAM_PDN2	IO_CFG_L_BASE+0x0000[9:0]	IO_CFG_L_BASE+0x0040[4:0]	IO_CFG_L_BASE+0x00B0[9:0]	IO_CFG_L_BASE+0x00D0[9:0]	-
PAD_CAM_RST0	IO_CFG_L_BASE+0x0000[9:0]	IO_CFG_L_BASE+0x0040[4:0]	IO_CFG_L_BASE+0x00B0[9:0]	IO_CFG_L_BASE+0x00D0[9:0]	-
PAD_CAM_RST1	IO_CFG_L_BASE+0x0000[9:0]	IO_CFG_L_BASE+0x0040[4:0]	IO_CFG_L_BASE+0x00B0[9:0]	IO_CFG_L_BASE+0x00D0[9:0]	-
PAD_CAM_RST2	IO_CFG_L_BASE+0x0000[9:0]	IO_CFG_L_BASE+0x0040[4:0]	IO_CFG_L_BASE+0x00B0[9:0]	IO_CFG_L_BASE+0x00D0[9:0]	-
PAD_CONN_BT_CLK	IO_CFG_L_BASE+0x0010[24:16]	IO_CFG_L_BASE+0x0050[4:0]	IO_CFG_L_BASE+0x00C0[21:13]	IO_CFG_L_BASE+0x00E0[21:13]	-
PAD_CONN_BT_DATA	IO_CFG_L_BASE+0x0010[24:16]	IO_CFG_L_BASE+0x0050[4:0]	IO_CFG_L_BASE+0x00C0[21:13]	IO_CFG_L_BASE+0x00E0[21:13]	-
PAD_CONN_HRST_B	IO_CFG_L_BASE+0x0010[8:7]	IO_CFG_L_BASE+0x0040[15]	-	IO_CFG_L_BASE+0x00D0[31:30]	-
PAD_CONN_TOP_CLK	IO_CFG_L_BASE+0x0010[24:16]	IO_CFG_L_BASE+0x0050[4:0]	IO_CFG_L_BASE+0x00C0[21:13]	IO_CFG_L_BASE+0x00E0[21:13]	-
PAD_CONN_TOP_DATA	IO_CFG_L_BASE+0x0010[24:16]	IO_CFG_L_BASE+0x0050[4:0]	IO_CFG_L_BASE+0x00C0[21:13]	IO_CFG_L_BASE+0x00E0[21:13]	-
PAD_CONN_WB_PTA	IO_CFG_L_BASE+0x0010[24:16]	IO_CFG_L_BASE+0x0050[4:0]	IO_CFG_L_BASE+0x00C0[21:13]	IO_CFG_L_BASE+0x00E0[21:13]	-
PAD_CONN_WF_CTRL0	IO_CFG_L_BASE+0x0010[24:16]	IO_CFG_L_BASE+0x0050[4:0]	IO_CFG_L_BASE+0x00C0[21:13]	IO_CFG_L_BASE+0x00E0[21:13]	-
PAD_CONN_WF_CTRL1	IO_CFG_L_BASE+0x0010[24:16]	IO_CFG_L_BASE+0x0050[4:0]	IO_CFG_L_BASE+0x00C0[21:13]	IO_CFG_L_BASE+0x00E0[21:13]	-
PAD_CONN_WF_CTRL2	IO_CFG_L_BASE+0x0010[24:16]	IO_CFG_L_BASE+0x0050[4:0]	IO_CFG_L_BASE+0x00C0[21:13]	IO_CFG_L_BASE+0x00E0[21:13]	-
PAD_DPI_CK	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_D0	IO_CFG_B_BASE+0x0000[1:0]	IO_CFG_B_BASE+0x0050[0]	-	IO_CFG_B_BASE+0x0130[1:0]	-
PAD_DPI_D1	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_D10	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_D11	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_D2	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_D3	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_D4	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_D5	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-

Name	IES	SMT	PU	PD	PUPD
PAD_DPI_D6	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_D7	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_D8	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_D9	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_DE	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_HSYNC	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DPI_VSYNC	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_DRVBUS	IO_CFG_B_BASE+0x0000[31:15]	IO_CFG_B_BASE+0x0050[9:6]	IO_CFG_B_BASE+0x0160[20:4]	IO_CFG_B_BASE+0x0130[22:6]	-
PAD_DSI_TE	IO_CFG_T_BASE+0x0000[9:6]	IO_CFG_T_BASE+0x0030[3:0]	IO_CFG_T_BASE+0x0090[4:0]	IO_CFG_T_BASE+0x0070[4:0]	-
PAD_EINT0	IO_CFG_B_BASE+0x0000[14:8]	IO_CFG_B_BASE+0x0050[5:3]	IO_CFG_B_BASE+0x0160[3:0]	IO_CFG_B_BASE+0x0130[5:2]	-
PAD_EINT1	IO_CFG_R_BASE+0x0010[17:8]	IO_CFG_R_BASE+0x0030[29:26]	IO_CFG_R_BASE+0x00E0[17:8]	IO_CFG_R_BASE+0x00C0[17:8]	-
PAD_EINT10	IO_CFG_R_BASE+0x0010[17:8]	IO_CFG_R_BASE+0x0030[29:26]	IO_CFG_R_BASE+0x00E0[17:8]	IO_CFG_R_BASE+0x00C0[17:8]	-
PAD_EINT11	IO_CFG_B_BASE+0x0000[31:15]	IO_CFG_B_BASE+0x0050[9:6]	IO_CFG_B_BASE+0x0160[20:4]	IO_CFG_B_BASE+0x0130[22:6]	-
PAD_EINT12	IO_CFG_B_BASE+0x0000[31:15]	IO_CFG_B_BASE+0x0050[9:6]	IO_CFG_B_BASE+0x0160[20:4]	IO_CFG_B_BASE+0x0130[22:6]	-
PAD_EINT13	IO_CFG_B_BASE+0x0000[31:15]	IO_CFG_B_BASE+0x0050[9:6]	IO_CFG_B_BASE+0x0160[20:4]	IO_CFG_B_BASE+0x0130[22:6]	-
PAD_EINT14	IO_CFG_B_BASE+0x0000[31:15]	IO_CFG_B_BASE+0x0050[9:6]	IO_CFG_B_BASE+0x0160[20:4]	IO_CFG_B_BASE+0x0130[22:6]	-
PAD_EINT15	IO_CFG_B_BASE+0x0000[31:15]	IO_CFG_B_BASE+0x0050[9:6]	IO_CFG_B_BASE+0x0160[20:4]	IO_CFG_B_BASE+0x0130[22:6]	-
PAD_EINT16	IO_CFG_B_BASE+0x0000[31:15]	IO_CFG_B_BASE+0x0050[9:6]	IO_CFG_B_BASE+0x0160[20:4]	IO_CFG_B_BASE+0x0130[22:6]	-
PAD_EINT2	IO_CFG_R_BASE+0x0010[17:8]	IO_CFG_R_BASE+0x0030[29:26]	IO_CFG_R_BASE+0x00E0[17:8]	IO_CFG_R_BASE+0x00C0[17:8]	-
PAD_EINT3	IO_CFG_R_BASE+0x0010[17:8]	IO_CFG_R_BASE+0x0030[29:26]	IO_CFG_R_BASE+0x00E0[17:8]	IO_CFG_R_BASE+0x00C0[17:8]	-
PAD_EINT4	IO_CFG_R_BASE+0x0010[17:8]	IO_CFG_R_BASE+0x0030[29:26]	IO_CFG_R_BASE+0x00E0[17:8]	IO_CFG_R_BASE+0x00C0[17:8]	-
PAD_EINT5	IO_CFG_R_BASE+0x0010[17:8]	IO_CFG_R_BASE+0x0030[29:26]	IO_CFG_R_BASE+0x00E0[17:8]	IO_CFG_R_BASE+0x00C0[17:8]	-
PAD_EINT6	IO_CFG_R_BASE+0x0010[17:8]	IO_CFG_R_BASE+0x0030[29:26]	IO_CFG_R_BASE+0x00E0[17:8]	IO_CFG_R_BASE+0x00C0[17:8]	-
PAD_EINT7	IO_CFG_R_BASE+0x0010[17:8]	IO_CFG_R_BASE+0x0030[29:26]	IO_CFG_R_BASE+0x00E0[17:8]	IO_CFG_R_BASE+0x00C0[17:8]	-
PAD_EINT8	IO_CFG_L_BASE+0x0010[24:16]	IO_CFG_L_BASE+0x0050[4:0]	IO_CFG_L_BASE+0x00C0[21:13]	IO_CFG_L_BASE+0x00E0[21:13]	-
PAD_EINT9	IO_CFG_R_BASE+0x0010[17:8]	IO_CFG_R_BASE+0x0030[29:26]	IO_CFG_R_BASE+0x00E0[17:8]	IO_CFG_R_BASE+0x00C0[17:8]	-
PAD_I2S0_BCK	IO_CFG_B_BASE+0x0010[16:12]	IO_CFG_B_BASE+0x0060[10:7]	IO_CFG_B_BASE+0x0170[4:0]	IO_CFG_B_BASE+0x0140[8:4]	-
PAD_I2S0_DI	IO_CFG_B_BASE+0x0010[16:12]	IO_CFG_B_BASE+0x0060[10:7]	IO_CFG_B_BASE+0x0170[4:0]	IO_CFG_B_BASE+0x0140[8:4]	-
PAD_I2S0_LRCK	IO_CFG_R_BASE+0x0010[17:8]	IO_CFG_R_BASE+0x0030[29:26]	IO_CFG_R_BASE+0x00E0[17:8]	IO_CFG_R_BASE+0x00C0[17:8]	-

Name	IES	SMT	PU	PD	PUPD
PAD_I2S0_MCK	IO_CFG_B_BASE+0x0010[16:12]	IO_CFG_B_BASE+0x0060[10:7]	IO_CFG_B_BASE+0x0170[4:0]	IO_CFG_B_BASE+0x0140[8:4]	-
PAD_I2S1_BCK	IO_CFG_B_BASE+0x0020[16:12]	IO_CFG_B_BASE+0x0060[6:4]	IO_CFG_B_BASE+0x0180[12:8]	IO_CFG_B_BASE+0x0150[6:2]	-
PAD_I2S1_DO	IO_CFG_B_BASE+0x0020[16:12]	IO_CFG_B_BASE+0x0060[6:4]	IO_CFG_B_BASE+0x0180[12:8]	IO_CFG_B_BASE+0x0150[6:2]	-
PAD_I2S1_LRCK	IO_CFG_B_BASE+0x0010[11:8]	IO_CFG_B_BASE+0x0050[17:15]	IO_CFG_B_BASE+0x0160[28:25]	IO_CFG_B_BASE+0x0140[3:0]	-
PAD_I2S1_MCK	IO_CFG_B_BASE+0x0010[20:17]	IO_CFG_B_BASE+0x0050[20:18]	IO_CFG_B_BASE+0x0170[8:5]	IO_CFG_B_BASE+0x0140[12:9]	-
PAD_I2S2_DI	IO_CFG_B_BASE+0x0020[16:12]	IO_CFG_B_BASE+0x0060[6:4]	IO_CFG_B_BASE+0x0180[12:8]	IO_CFG_B_BASE+0x0150[6:2]	-
PAD_I2S3_DO	IO_CFG_B_BASE+0x0010[16:12]	IO_CFG_B_BASE+0x0060[10:7]	IO_CFG_B_BASE+0x0170[4:0]	IO_CFG_B_BASE+0x0140[8:4]	-
PAD_IDDIG	IO_CFG_T_BASE+0x0000[9:6]	IO_CFG_T_BASE+0x0030[3:0]	IO_CFG_T_BASE+0x0090[4:0]	IO_CFG_T_BASE+0x0070[4:0]	-
PAD_INT_SIM1	IO_CFG_B_BASE+0x0000[7:2]	IO_CFG_B_BASE+0x0050[2:1]	-	-	-
PAD_INT_SIM2	IO_CFG_B_BASE+0x0020[20:17]	IO_CFG_B_BASE+0x0060[13:11]	IO_CFG_B_BASE+0x0180[3:0]	IO_CFG_B_BASE+0x0150[0]	-
PAD_JTCK	IO_CFG_L_BASE+0x0010[4:0]	IO_CFG_L_BASE+0x0040[13:11]	IO_CFG_L_BASE+0x00C0[5:0]	IO_CFG_L_BASE+0x00E0[5:0]	-
PAD_JTDI	IO_CFG_L_BASE+0x0010[4:0]	IO_CFG_L_BASE+0x0040[13:11]	IO_CFG_L_BASE+0x00C0[5:0]	IO_CFG_L_BASE+0x00E0[5:0]	-
PAD_JTDO	IO_CFG_L_BASE+0x0010[4:0]	IO_CFG_L_BASE+0x0040[13:11]	IO_CFG_L_BASE+0x00C0[5:0]	IO_CFG_L_BASE+0x00E0[5:0]	-
PAD_JTMS	IO_CFG_B_BASE+0x0010[23:21]	IO_CFG_B_BASE+0x0050[22:21]	IO_CFG_B_BASE+0x0170[11:9]	IO_CFG_B_BASE+0x0140[15:13]	-
PAD_JTRST_B	IO_CFG_L_BASE+0x0010[4:0]	IO_CFG_L_BASE+0x0040[13:11]	IO_CFG_L_BASE+0x00C0[5:0]	IO_CFG_L_BASE+0x00E0[5:0]	-
PAD_KPCOL0	IO_CFG_B_BASE+0x0000[7:2]	IO_CFG_B_BASE+0x0050[2:1]	-	-	-
PAD_KPCOL1	IO_CFG_B_BASE+0x0000[7:2]	IO_CFG_B_BASE+0x0050[2:1]	-	-	-
PAD_KPCOL2	IO_CFG_B_BASE+0x0000[7:2]	IO_CFG_B_BASE+0x0050[2:1]	-	-	-
PAD_KPROW0	IO_CFG_R_BASE+0x0000[27:20]	IO_CFG_R_BASE+0x0030[23:16]	IO_CFG_R_BASE+0x00D0[26:19]	IO_CFG_R_BASE+0x00B0[26:19]	-
PAD_KPROW1	IO_CFG_B_BASE+0x0000[7:2]	IO_CFG_B_BASE+0x0050[2:1]	-	-	-
PAD_KPROW2	IO_CFG_B_BASE+0x0000[7:2]	IO_CFG_B_BASE+0x0050[2:1]	-	-	-
PAD_LCM_RST	IO_CFG_T_BASE+0x0000[9:6]	IO_CFG_T_BASE+0x0030[3:0]	IO_CFG_T_BASE+0x0090[4:0]	IO_CFG_T_BASE+0x0070[4:0]	-
PAD_MISC_MIPI_CK_0	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-
PAD_MISC_MIPI_CK_1	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-
PAD_MISC_MIPI_CK_2	IO_CFG_T_BASE+0x0010[23:0]	IO_CFG_T_BASE+0x0030[10:4]	IO_CFG_T_BASE+0x00A0[23:0]	IO_CFG_T_BASE+0x0080[23:0]	-
PAD_MISC_MIPI_CK_3	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-
PAD_MISC_MIPI_DO_0	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-

Name	IES	SMT	PU	PD	PUPD
PAD_MISC_MIPI_DO_1	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-
PAD_MISC_MIPI_DO_2	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-
PAD_MISC_MIPI_DO_3	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-
PAD_MSDC0_CLK	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC0_CMD	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC0_DAT0	IO_CFG_B_BASE+0x0020[24:21]	IO_CFG_B_BASE+0x0060[16:14]	IO_CFG_B_BASE+0x0180[7:4]	IO_CFG_B_BASE+0x0150[1]	-
PAD_MSDC0_DAT1	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC0_DAT2	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC0_DAT3	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC0_DAT4	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC0_DAT5	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC0_DAT6	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC0_DAT7	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC0_DSL	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC0_RSTB	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_MSDC1_CLK	IO_CFG_B_BASE+0x0020[30:25]	IO_CFG_R_BASE+0x0030[2:0]	-	-	IO_CFG_B_BASE+0x0100[23:18]
PAD_MSDC1_CMD	IO_CFG_B_BASE+0x0020[20:17]	IO_CFG_B_BASE+0x0060[13:11]	IO_CFG_B_BASE+0x0180[3:0]	-	IO_CFG_B_BASE+0x0100[26:24]
PAD_MSDC1_DAT0	IO_CFG_B_BASE+0x0020[30:25]	IO_CFG_R_BASE+0x0030[2:0]	-	-	IO_CFG_B_BASE+0x0100[23:18]
PAD_MSDC1_DAT1	IO_CFG_B_BASE+0x0020[30:25]	IO_CFG_R_BASE+0x0030[2:0]	-	-	IO_CFG_B_BASE+0x0100[23:18]
PAD_MSDC1_DAT2	IO_CFG_B_BASE+0x0020[30:25]	IO_CFG_R_BASE+0x0030[2:0]	-	-	IO_CFG_B_BASE+0x0100[23:18]
PAD_MSDC1_DAT3	IO_CFG_B_BASE+0x0020[30:25]	IO_CFG_R_BASE+0x0030[2:0]	-	-	IO_CFG_B_BASE+0x0100[23:18]
PAD_PWRAP_SPI0_CK	IO_CFG_R_BASE+0x0000[19:11]	IO_CFG_R_BASE+0x0030[12:7]	IO_CFG_R_BASE+0x00D0[18:10]	IO_CFG_R_BASE+0x00B0[18:10]	-
PAD_PWRAP_SPI0_CSN	IO_CFG_R_BASE+0x0000[19:11]	IO_CFG_R_BASE+0x0030[12:7]	IO_CFG_R_BASE+0x00D0[18:10]	IO_CFG_R_BASE+0x00B0[18:10]	-
PAD_PWRAP_SPI0_MI	IO_CFG_L_BASE+0x0010[15:9]	IO_CFG_L_BASE+0x0040[19:16]	IO_CFG_L_BASE+0x00C0[12:6]	IO_CFG_L_BASE+0x00E0[12:6]	-
PAD_PWRAP_SPI0_MO	IO_CFG_R_BASE+0x0000[19:11]	IO_CFG_R_BASE+0x0030[12:7]	IO_CFG_R_BASE+0x00D0[18:10]	IO_CFG_R_BASE+0x00B0[18:10]	-
PAD_RFIC0_BSI_CK	-	-	IO_CFG_T_BASE+0x0090[4:0]	IO_CFG_T_BASE+0x0070[4:0]	-
PAD_RFIC0_BSI_D0	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-

Name	IES	SMT	PU	PD	PUPD
PAD_RFIC0_BSI_D1	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-
PAD_RFIC0_BSI_D2	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-
PAD_RFIC0_BSI_EN	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-
PAD_RTC32K_CK	IO_CFG_L_BASE+0x0000[11:10]	IO_CFG_L_BASE+0x0040[5]	IO_CFG_L_BASE+0x00B0[11:10]	IO_CFG_L_BASE+0x00D0[11:10]	-
PAD_SCL0	IO_CFG_L_BASE+0x0000[9:0]	IO_CFG_L_BASE+0x0040[4:0]	IO_CFG_L_BASE+0x00B0[9:0]	IO_CFG_L_BASE+0x00D0[9:0]	-
PAD_SCL1	IO_CFG_L_BASE+0x0000[27:12]	IO_CFG_L_BASE+0x0040[10:6]	IO_CFG_L_BASE+0x00B0[27:12]	IO_CFG_L_BASE+0x00D0[27:12]	-
PAD_SCL2	IO_CFG_L_BASE+0x0010[6:5]	IO_CFG_L_BASE+0x0040[14]	-	IO_CFG_L_BASE+0x00D0[29:28]	-
PAD_SCL3	IO_CFG_B_BASE+0x0010[16:12]	IO_CFG_B_BASE+0x0060[10:7]	IO_CFG_B_BASE+0x0170[4:0]	IO_CFG_B_BASE+0x0140[8:4]	-
PAD_SCL4	IO_CFG_B_BASE+0x0010[5:4]	IO_CFG_B_BASE+0x0050[13]	-	IO_CFG_B_BASE+0x0130[28:27]	-
PAD_SCL5	IO_CFG_B_BASE+0x0010[7:6]	IO_CFG_B_BASE+0x0050[14]	-	IO_CFG_B_BASE+0x0130[30:29]	-
PAD_SCL6	IO_CFG_R_BASE+0x0000[19:11]	IO_CFG_R_BASE+0x0030[12:7]	IO_CFG_R_BASE+0x00D0[18:10]	IO_CFG_R_BASE+0x00B0[18:10]	-
PAD_SCL7	IO_CFG_R_BASE+0x0010[3:2]	IO_CFG_R_BASE+0x0030[14]	IO_CFG_R_BASE+0x00E0[3:2]	IO_CFG_R_BASE+0x00C0[3:2]	-
PAD_SDA0	IO_CFG_B_BASE+0x0000[1:0]	IO_CFG_B_BASE+0x0050[0]	-	IO_CFG_B_BASE+0x0130[1:0]	-
PAD_SDA1	IO_CFG_R_BASE+0x0010[1:0]	IO_CFG_R_BASE+0x0030[13]	-	IO_CFG_R_BASE+0x00C0[1:0]	-
PAD_SDA2	IO_CFG_R_BASE+0x0000[27:20]	IO_CFG_R_BASE+0x0030[23:16]	IO_CFG_R_BASE+0x00D0[26:19]	IO_CFG_R_BASE+0x00B0[26:19]	-
PAD_SDA3	IO_CFG_L_BASE+0x0010[8:7]	IO_CFG_L_BASE+0x0040[15]	-	IO_CFG_L_BASE+0x00D0[31:30]	-
PAD_SDA4	IO_CFG_B_BASE+0x0010[3:0]	IO_CFG_B_BASE+0x0050[12:10]	IO_CFG_B_BASE+0x0160[24:21]	IO_CFG_B_BASE+0x0130[26:23]	-
PAD_SDA5	IO_CFG_B_BASE+0x0010[5:4]	IO_CFG_B_BASE+0x0050[13]	-	IO_CFG_B_BASE+0x0130[28:27]	-
PAD_SDA6	IO_CFG_R_BASE+0x0010[3:2]	IO_CFG_R_BASE+0x0030[14]	IO_CFG_R_BASE+0x00E0[3:2]	IO_CFG_R_BASE+0x00C0[3:2]	-
PAD_SDA7	IO_CFG_R_BASE+0x0010[5:4]	IO_CFG_R_BASE+0x0030[15]	IO_CFG_R_BASE+0x00D0[30:29]	IO_CFG_R_BASE+0x00C0[5:4]	-
PAD_SIM1_SCLK	IO_CFG_B_BASE+0x0020[11:0]	IO_CFG_B_BASE+0x0060[3:0]	IO_CFG_B_BASE+0x0170[23:12]	IO_CFG_B_BASE+0x0140[27:16]	IO_CFG_B_BASE+0x0100[17:6]
PAD_SIM1_SIO	IO_CFG_B_BASE+0x0020[20:17]	IO_CFG_B_BASE+0x0060[13:11]	IO_CFG_B_BASE+0x0180[3:0]	-	IO_CFG_B_BASE+0x0100[26:24]
PAD_SIM1_SRST	IO_CFG_B_BASE+0x0020[20:17]	IO_CFG_B_BASE+0x0060[13:11]	IO_CFG_B_BASE+0x0180[3:0]	-	IO_CFG_B_BASE+0x0100[26:24]
PAD_SIM2_SCLK	IO_CFG_R_BASE+0x0010[5:4]	IO_CFG_R_BASE+0x0030[15]	IO_CFG_R_BASE+0x00D0[30:29]	IO_CFG_R_BASE+0x00C0[5:4]	-
PAD_SIM2_SIO	IO_CFG_B_BASE+0x0020[24:21]	IO_CFG_B_BASE+0x0060[16:14]	IO_CFG_B_BASE+0x0180[7:4]	-	IO_CFG_B_BASE+0x0100[29:27]
PAD_SIM2_SRST	IO_CFG_B_BASE+0x0020[24:21]	IO_CFG_B_BASE+0x0060[16:14]	IO_CFG_B_BASE+0x0180[7:4]	-	IO_CFG_B_BASE+0x0100[29:27]
PAD_SPI0_CK	IO_CFG_R_BASE+0x0010[1:0]	IO_CFG_R_BASE+0x0030[13]	-	IO_CFG_R_BASE+0x00C0[1:0]	-
PAD_SPI0_CS	IO_CFG_B_BASE+0x0000[14:8]	IO_CFG_B_BASE+0x0050[5:3]	IO_CFG_B_BASE+0x0160[3:0]	IO_CFG_B_BASE+0x0130[5:2]	-
PAD_SPI0_MI	IO_CFG_B_BASE+0x0000[14:8]	IO_CFG_B_BASE+0x0050[5:3]	IO_CFG_B_BASE+0x0160[3:0]	IO_CFG_B_BASE+0x0130[5:2]	-

Name	IES	SMT	PU	PD	PUPD
PAD_SPI0_MO	IO_CFG_B_BASE+0x0000[14:8]	IO_CFG_B_BASE+0x0050[5:3]	IO_CFG_B_BASE+0x0160[3:0]	IO_CFG_B_BASE+0x0130[5:2]	-
PAD_SPI1_CK	IO_CFG_R_BASE+0x0010[7:6]	IO_CFG_R_BASE+0x0030[24]	IO_CFG_R_BASE+0x00E0[7:6]	IO_CFG_R_BASE+0x00C0[7:6]	-
PAD_SPI1_CS	IO_CFG_B_BASE+0x0010[3:0]	IO_CFG_B_BASE+0x0050[12:10]	IO_CFG_B_BASE+0x0160[24:21]	IO_CFG_B_BASE+0x0130[26:23]	-
PAD_SPI1_MI	IO_CFG_B_BASE+0x0010[3:0]	IO_CFG_B_BASE+0x0050[12:10]	IO_CFG_B_BASE+0x0160[24:21]	IO_CFG_B_BASE+0x0130[26:23]	-
PAD_SPI1_MO	IO_CFG_B_BASE+0x0010[3:0]	IO_CFG_B_BASE+0x0050[12:10]	IO_CFG_B_BASE+0x0160[24:21]	IO_CFG_B_BASE+0x0130[26:23]	-
PAD_SPI2_CK	IO_CFG_B_BASE+0x0010[7:6]	IO_CFG_B_BASE+0x0050[14]	-	IO_CFG_B_BASE+0x0130[30:29]	-
PAD_SPI2_CS	IO_CFG_B_BASE+0x0010[11:8]	IO_CFG_B_BASE+0x0050[17:15]	IO_CFG_B_BASE+0x0160[28:25]	IO_CFG_B_BASE+0x0140[3:0]	-
PAD_SPI2_MI	IO_CFG_B_BASE+0x0010[11:8]	IO_CFG_B_BASE+0x0050[17:15]	IO_CFG_B_BASE+0x0160[28:25]	IO_CFG_B_BASE+0x0140[3:0]	-
PAD_SPI2_MO	IO_CFG_B_BASE+0x0010[11:8]	IO_CFG_B_BASE+0x0050[17:15]	IO_CFG_B_BASE+0x0160[28:25]	IO_CFG_B_BASE+0x0140[3:0]	-
PAD_SPI3_CK	IO_CFG_B_BASE+0x0010[20:17]	IO_CFG_B_BASE+0x0050[20:18]	IO_CFG_B_BASE+0x0170[8:5]	IO_CFG_B_BASE+0x0140[12:9]	-
PAD_SPI3_CS	IO_CFG_B_BASE+0x0010[20:17]	IO_CFG_B_BASE+0x0050[20:18]	IO_CFG_B_BASE+0x0170[8:5]	IO_CFG_B_BASE+0x0140[12:9]	-
PAD_SPI3_MI	IO_CFG_B_BASE+0x0020[16:12]	IO_CFG_B_BASE+0x0060[6:4]	IO_CFG_B_BASE+0x0180[12:8]	IO_CFG_B_BASE+0x0150[6:2]	-
PAD_SPI3_MO	IO_CFG_B_BASE+0x0010[20:17]	IO_CFG_B_BASE+0x0050[20:18]	IO_CFG_B_BASE+0x0170[8:5]	IO_CFG_B_BASE+0x0140[12:9]	-
PAD_SRCLKENA0	IO_CFG_R_BASE+0x0000[27:20]	IO_CFG_R_BASE+0x0030[23:16]	IO_CFG_R_BASE+0x00D0[26:19]	IO_CFG_R_BASE+0x00B0[26:19]	-
PAD_SRCLKENA1	IO_CFG_R_BASE+0x0000[27:20]	IO_CFG_R_BASE+0x0030[23:16]	IO_CFG_R_BASE+0x00D0[26:19]	IO_CFG_R_BASE+0x00B0[26:19]	-
PAD_SRCLKENAI0	IO_CFG_R_BASE+0x0000[27:20]	IO_CFG_R_BASE+0x0030[23:16]	IO_CFG_R_BASE+0x00D0[26:19]	IO_CFG_R_BASE+0x00B0[26:19]	-
PAD_SRCLKENAI1	IO_CFG_R_BASE+0x0000[27:20]	IO_CFG_R_BASE+0x0030[23:16]	IO_CFG_R_BASE+0x00D0[26:19]	IO_CFG_R_BASE+0x00B0[26:19]	-
PAD_SYSRSTB	IO_CFG_R_BASE+0x0000[27:20]	IO_CFG_R_BASE+0x0030[23:16]	IO_CFG_R_BASE+0x00D0[26:19]	IO_CFG_R_BASE+0x00B0[26:19]	-
PAD_TDM_BCK	IO_CFG_L_BASE+0x0010[15:9]	IO_CFG_L_BASE+0x0040[19:16]	IO_CFG_L_BASE+0x00C0[12:6]	IO_CFG_L_BASE+0x00E0[12:6]	-
PAD_TDM_DATA0	IO_CFG_L_BASE+0x0010[15:9]	IO_CFG_L_BASE+0x0040[19:16]	IO_CFG_L_BASE+0x00C0[12:6]	IO_CFG_L_BASE+0x00E0[12:6]	-
PAD_TDM_DATA1	IO_CFG_L_BASE+0x0010[15:9]	IO_CFG_L_BASE+0x0040[19:16]	IO_CFG_L_BASE+0x00C0[12:6]	IO_CFG_L_BASE+0x00E0[12:6]	-
PAD_TDM_DATA2	IO_CFG_L_BASE+0x0010[15:9]	IO_CFG_L_BASE+0x0040[19:16]	IO_CFG_L_BASE+0x00C0[12:6]	IO_CFG_L_BASE+0x00E0[12:6]	-
PAD_TDM_DATA3	IO_CFG_L_BASE+0x0010[15:9]	IO_CFG_L_BASE+0x0040[19:16]	IO_CFG_L_BASE+0x00C0[12:6]	IO_CFG_L_BASE+0x00E0[12:6]	-
PAD_TDM_LRCK	IO_CFG_B_BASE+0x0020[30:25]	IO_CFG_R_BASE+0x0030[2:0]	-	-	IO_CFG_B_BASE+0x0100[23:18]
PAD_TDM_MCK	IO_CFG_L_BASE+0x0010[15:9]	IO_CFG_L_BASE+0x0040[19:16]	IO_CFG_L_BASE+0x00C0[12:6]	IO_CFG_L_BASE+0x00E0[12:6]	-
PAD_TDP0	IO_CFG_B_BASE+0x0020[24:21]	IO_CFG_B_BASE+0x0060[16:14]	IO_CFG_B_BASE+0x0180[7:4]	-	IO_CFG_B_BASE+0x0100[29:27]
PAD_TESTMODE	IO_CFG_T_BASE+0x0000[9:6]	IO_CFG_T_BASE+0x0030[3:0]	IO_CFG_T_BASE+0x0090[4:0]	IO_CFG_T_BASE+0x0070[4:0]	-
PAD_URXD0	IO_CFG_L_BASE+0x0010[6:5]	IO_CFG_L_BASE+0x0040[14]	-	IO_CFG_L_BASE+0x00D0[29:28]	-
PAD_UTXD0	IO_CFG_L_BASE+0x0000[11:10]	IO_CFG_L_BASE+0x0040[5]	IO_CFG_L_BASE+0x00B0[11:10]	IO_CFG_L_BASE+0x00D0[11:10]	-

Name	IES	SMT	PU	PD	PUPD
PAD_UTXD1	IO_CFG_R_BASE+0x0010[7:6]	IO_CFG_R_BASE+0x0030[24]	IO_CFG_R_BASE+0x00E0[7:6]	IO_CFG_R_BASE+0x00C0[7:6]	-
PAD_VOW_CLK_MISO	IO_CFG_R_BASE+0x0000[19:11]	IO_CFG_R_BASE+0x0030[12:7]	IO_CFG_R_BASE+0x00D0[18:10]	IO_CFG_R_BASE+0x00B0[18:10]	-
PAD_WATCHDOG	-	-	IO_CFG_R_BASE+0x00D0[26:19]	IO_CFG_R_BASE+0x00B0[26:19]	-
PAD_WF_IP	IO_CFG_T_BASE+0x0000[22:10]	IO_CFG_T_BASE+0x0030[21:11]	IO_CFG_T_BASE+0x0090[17:5]	IO_CFG_T_BASE+0x0070[17:5]	-

5.2.2 Register Definition

See chapter 3.2 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

Register bank	Base address
GPIO_BASE	0x10005000
IOCFG_L_BASE	0x10002000
IOCFG_B_BASE	0x10002400
IOCFG_R_BASE	0x10002800
IOCFG_T_BASE	0x10002C00

5.3 Keypad Scanner

5.3.1 General Description

The keypad supports two types of keypads: 3*3 single keys and 3*3 configurable double keys.

The 3*3 keypad can be divided into two parts: 1) The keypad interface including 8 columns and 8 rows (see Figure 5-3 and Figure 5-4); 2) The key detection block provides key pressed, key released and de-bounce mechanisms.

Each time the key is pressed or released, i.e. something different in the 8x8 matrix, the key detection block senses the change and recognizes if a key has been pressed or released. Whenever the key status changes and is stable, a KEYPAD IRQ will be issued. The MCU can then read the key(s) pressed directly in the KP_MEM1, KP_MEM2, KP_MEM3, KP_MEM4 and KP_MEM5 registers. To ensure the key pressed information is not missed, the status register in keypad will not be read-cleared by the APB read command. The status register can only be changed by the key-pressed detection FSM.

This keypad detects one or two keys pressed simultaneously with any combination. Figure 5-7 shows the one key pressed condition. Figure 5-8(a) and Figure 5-8(b) illustrate the cases of two keys pressed. Since the key pressed detection depends on the HIGH or LOW level of the external keypad interface, if the keys are pressed at the same time, and there exists a key that is on the same column and the same row with other keys, the pressed key cannot be correctly decoded. For example, if there are three key pressed: key1 = (x1, y1), key2 = (x2, y2), and key3 = (x1, y2), both key3 and key4 = (x2, y1) will be detected, and therefore they cannot be distinguished correctly. Hence, the keypad detects only one or two keys pressed simultaneously in any combination. More than two keys pressed simultaneously in a specific pattern will retrieve the wrong information.

The 3*3 keypad supports a 3*3*2 = 18 keys matrix. The 18 keys are divided into 9 sub groups, and each group consists of 2 keys and a 20 ohm resistor. Besides the limitation of the 3*3 keypad, 3*3 keypad has another limitation, which is it cannot detect two keys pressed simultaneously when the two keys are in one group, i.e. the 3*3 keypad cannot detect key 0 and key 1 pressed simultaneously or key 14 and key 15 pressed simultaneously.

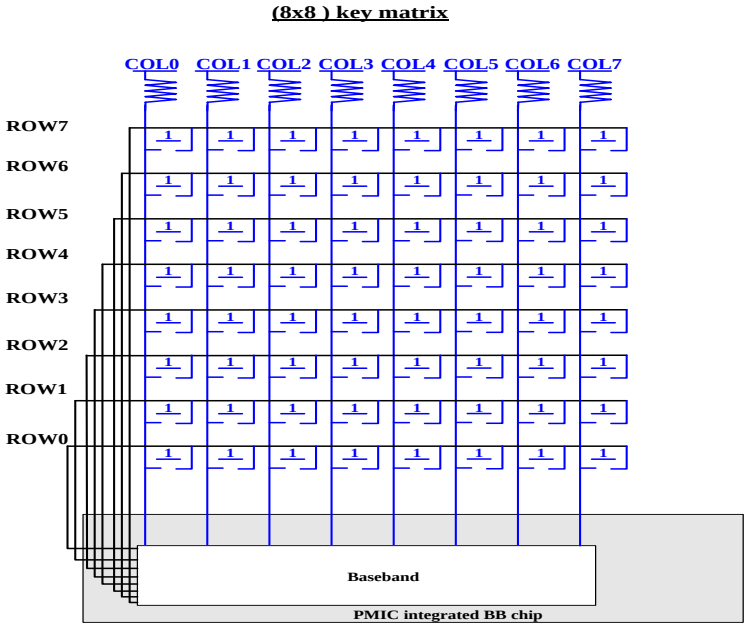


Figure 5-3. 3x3 keypad matrix (9 keys)

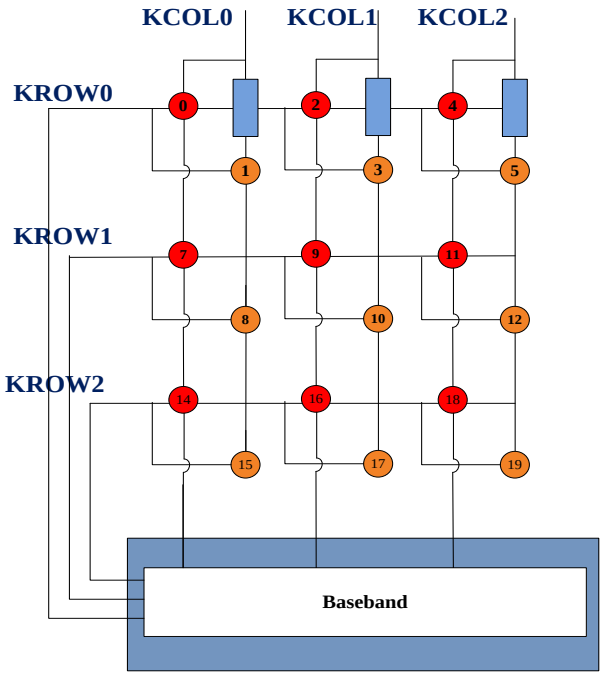


Figure 5-4. 3x3 keypad matrix (18 keys)

5.3.2 Waveform

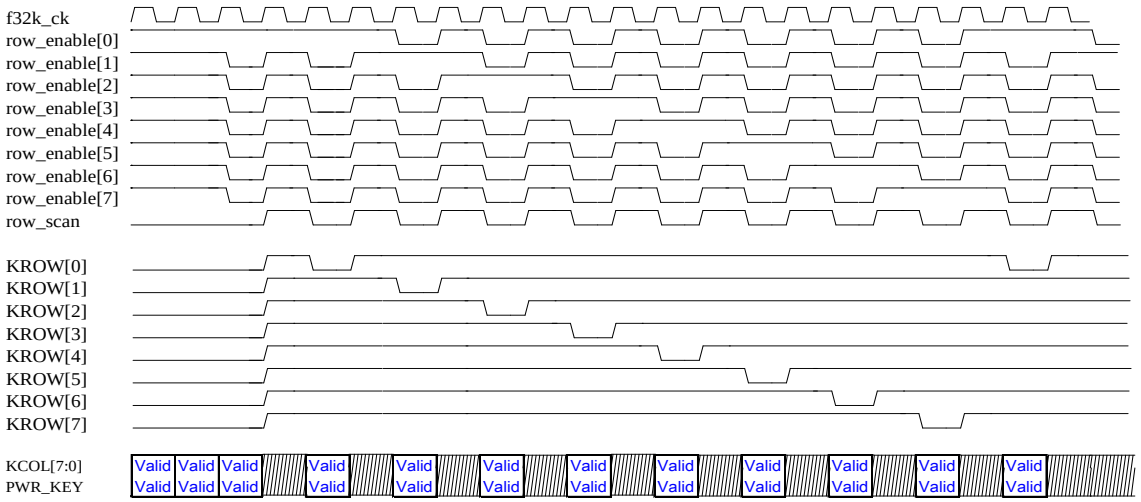


Figure 5-5. 3x3 single keypad scan waveform

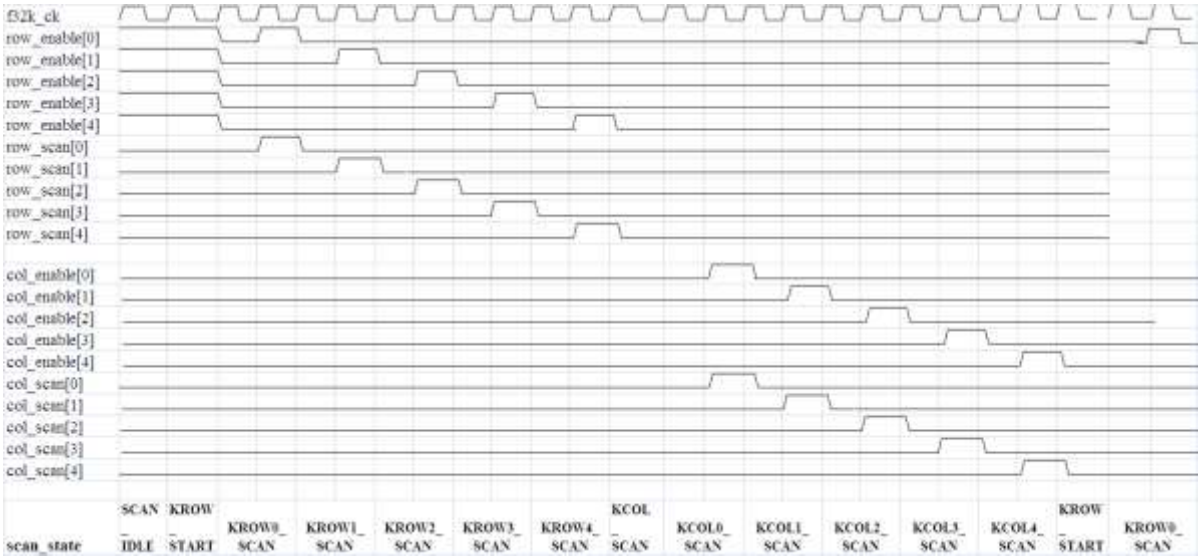


*Figure 5-6. 3*3 double keypad scan waveform*

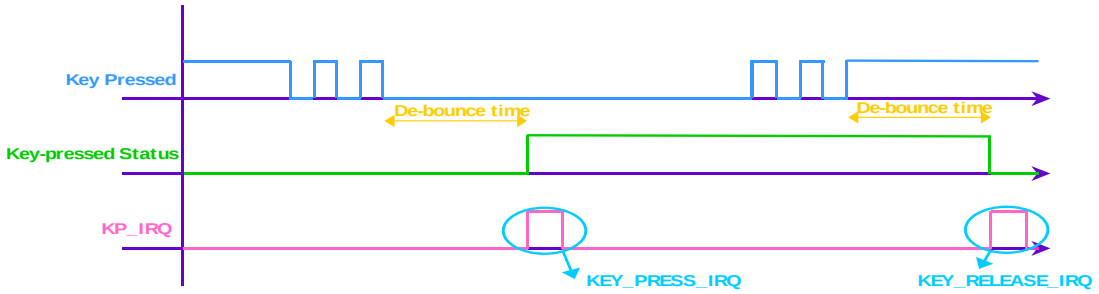


Figure 5-7. One key pressed with de-bounce mechanism denoted

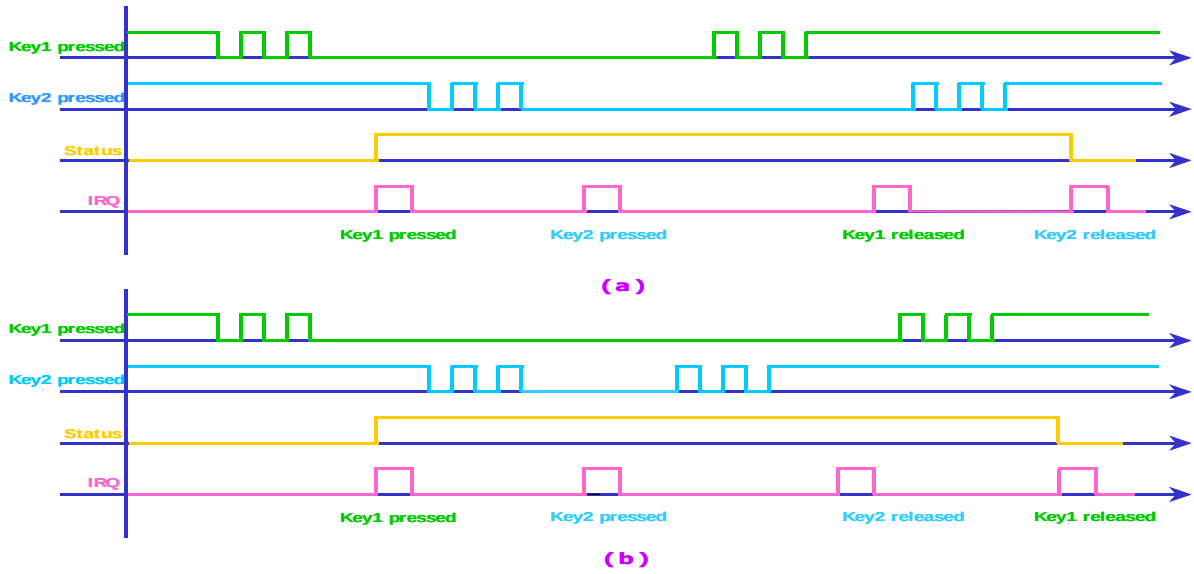


Figure 5-8. (a) Two keys pressed, case 1; (b) Two keys pressed, case 2

5.3.3 Register Definition

See chapter 3.3 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.4 UART

5.4.1 Introduction

The baseband chipset houses four UARTs. UARTs provide full duplex serial communication channels between the baseband chipset and external devices.

UART has both M16C450 and M16550A modes of operation, which are compatible with a range of standard software drivers. The extensions are designed to be broadly software compatible with 16550A variants, but certain areas offer no consensus.

In common with M16550A, the UART supports word lengths from 5 to 8 bits, an optional parity bit and one or two stop bits and is fully programmable by an 8-bit CPU interface. A 16-bit programmable baud rate generator and an 8-bit scratch register are included, together with separate transmit and receive FIFOs. Two modem control lines and a diagnostic loop-back mode are provided. UART also includes two DMA handshake lines, indicating when the FIFOs are ready to transfer data to the CPU. Interrupts can be generated from any of the ten sources.

Note that UART is designed so that all internal operation is synchronized by the CLK signal. This synchronization results in minor timing differences between the UART and industry standard 16550A device, which means that the core is not clock for clock identical to the original device.

After hardware reset, UART will be in M16C450 mode; its FIFOs can then be enabled and UART can enter M16550A mode. UART has further additional functions beyond the M16550A mode. Each of the extended functions can be selected individually under software control.

UART provides more powerful enhancements than the industry-standard 16550:

Hardware flow control
This feature is very useful when the ISR latency is hard to predict and control in the embedded applications. The MCU is relieved of having to fetch the received data within a fixed amount of time.

Note that in order to enable the enhancements, the enhanced mode bit, EFR[4], must be set. If EFR[4] is not set, IER[7:4], FCR[5:4], cannot be written and MCR[7] cannot be read. The enhanced mode bit ensures that UART is backward compatible with the software that has been written for 16C450 and 16550A devices.

5.4.2 Features

- Provides four channels
- DMA, polling or interrupt operation
- Supports word lengths from five to eight bits, with an optional parity bit and one or two stop bits

- Four UART ports for hardware automatic flow control (UART0, UART1, UART2, UART3)
- Supports baud rates from 110bps up to 961,200bps
- Baud rate auto detection function

5.4.3 Block Diagram

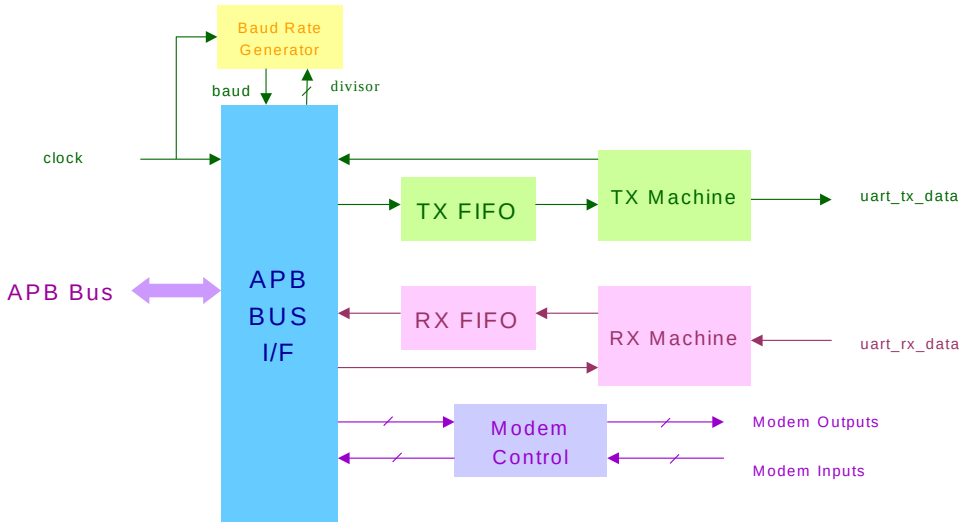


Figure 5-9. Block diagram of UART

5.4.4 Register Definition

UART number	Base address	DMA mode	HW flow control
AP UART0	0x11002000	V	V
AP UART1	0x11003000	V	V
AP UART2	0x11004000	V	V
AP UART3	0x11005000	V	V
SCP UART0	0x100A9000	X	X
SCP UART1	0x100AE000	X	X

There are six UART IPs in this SOC. The usage of the registers is the same except that the base address must be changed to respective one. Note that AP UART2 is connected to C2KSYS in the SOC, and do not connect to pin-mux.

See chapter 3.4 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.4.5 Programming Guide

5.4.5.1 Auto Baud Rate Detection

UART detects the baud rate used automatically. Follow the steps below:

1. Set up register autobaud_en to start detecting the data.
2. Send data of ASCII code “AT” or “at” to UART from the connected host, e.g. the PC.
3. Check if the “AT” or “at” is received. If received, the setting is now already set for further transmission.

5.4.5.2 Transmission

Follow the steps below for UART transmission:

1. Use the autobaud function to set up the baud rate or set up the parameters by yourself. The settings needed can be found in register DLL, DLM ,HIGH SPEED.
2. After setting up the baud rate, start the transmission by filling the TX FIFO and receiving data from RX FIFO.
3. Virtual FIFO can also be used for the transmission. To use the virtual FIFO, you need APDMA settings (refer to details in the APDMA section).

5.5 USB 2.0 High Speed Dual-Role Controller

5.5.1 Introduction

The USB controller is configured for supporting 5 endpoints to receive packets and 5 endpoints to send packets except for endpoint 0. These endpoints can be individually configured in the software to handle either Bulk transfers, Interrupt transfers or Isochronous transfers. There are 8 DMA channels and the embedded RAM size is configurable size up to 8K bytes. The embedded RAM can be dynamically configured to each endpoint. As the host for point-to-point communications, the controller maintains a frame counter and automatically schedules SOF, Isochronous, Interrupt and Bulk transfers.

5.5.2 Features

The following table lists the unified USB IP features.

Feature	Description
USB specifications	USB2.0 Host
Enhanced feature	Generic Host QMU
Endpoint	5 TX 5 RX EP0
DMA channel	5
Embedded RAM	Up to 8KB
UTMI+ interface	UTMI+ 16b
CPU slave interface	AHB asynchronous design
DMA master interface	AHB busy free asynchronous design

5.5.3 USB Controller Block Diagram

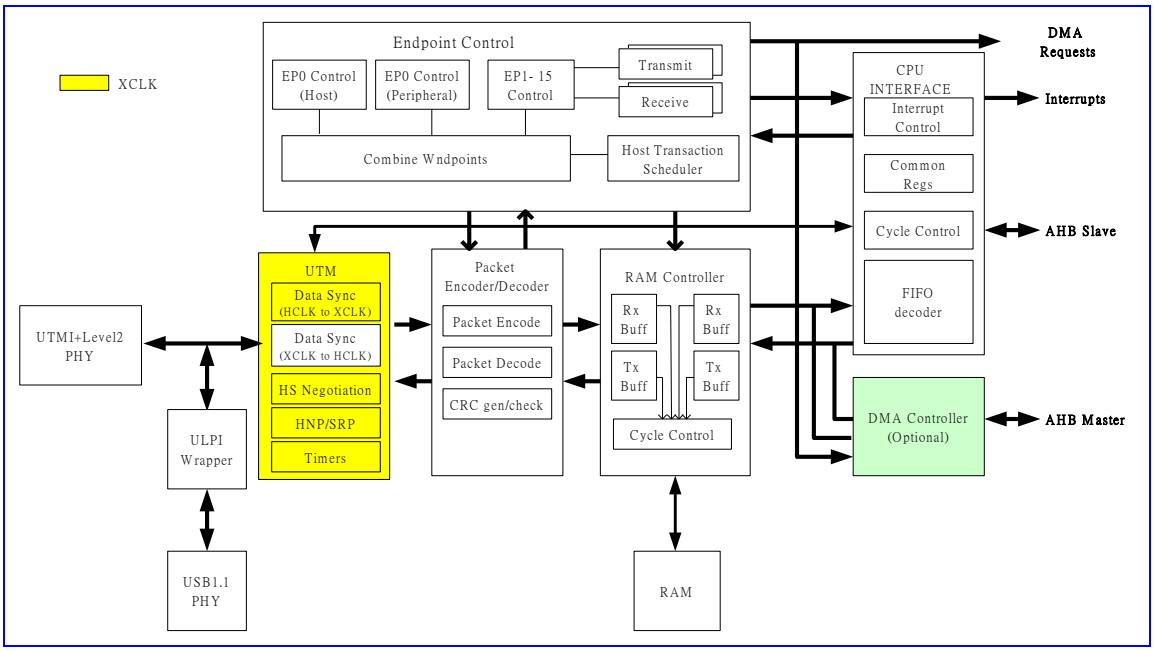


Figure 5-10. USB controller block diagram

5.5.4 Register Definition

Registers accessed using byte manipulation are marked in blue columns. Byte accessing registers can be accessed using word manipulation. Word accessing registers cannot be accessed using the byte manipulation.

Register address	Register name	Manipulation (Byte/Word)	Acronym
Common Registers			
USB + 0000h	Function address register	Byte	FADDR
USB + 0001h	Power management register	Byte	POWER
USB + 0002h	TX interrupt status register	Byte	INTRTX
USB + 0004h	RX interrupt status register	Byte	INTRRX
USB + 0006h	TX interrupt enable register	Byte	INTRTXE
USB + 0008h	RX interrupt enable register	Byte	INTRRXE
USB + 000Ah	Common USB interrupts register	Byte	INTRUSB
USB + 000Bh	Common USB interrupts enable register	Byte	INTRUSBE
USB + 000Ch	Frame number register	Byte	FRAME
USB + 000Eh	Endpoint selecting index register	Byte	INDEX
USB + 000Fh	Test mode enable register	Byte	TESTMODE

Register address	Register name	Manipulation (Byte/Word)	Acronym
Indexed EndPoint CSR Region *n stands for endpoint number.* *For example, endpoint 1's n = 1. Valid n = 1 ~ MaxEndPoint.* *MaxEndPoint is hardware configured and the maximum is 15.*			
USB + 0010h ~ USB + 001Fh	It maps to CSR EP0 ~ EPx depending on the INDEX register. For example, if INDEX is n, address 0010h ~ 001Fh are mapped to 0x(100+10*n)h ~ 0x(100+10*n+F)h.	Byte	Indexed CSR
USB + 0020h	USB endpoint 0 FIFO register	Byte	FIFO0
USB + 0020h +(n)*4 h	USB endpoint n FIFO register	Byte	FIFOn
OTG, Dynamic FIFO, Version Registers			
USB + 0060h	OTG device control register	Byte	DEVCTL
USB + 0061h	Power up counter register	Byte	PWRUPCNT
USB + 0062h	TX FIFO size register	Byte	TXFIFOSZ
USB + 0063h	RX FIFO size register	Byte	RXFIFOSZ
USB + 0064h	TX FIFO address register	Byte	TXFIFOADD
USB + 0066h	RX FIFO address register	Byte	RXFIFOADD
USB + 006Ch	Hardware capability register	Byte	HWCAPS
USB + 006Eh	Hardware sub version register	Byte	HWSVERS
Hardware Configuration, Special Setting Registers			
USB + 0070h	USB bus performance register 1	Byte	BUSPERF1
USB + 0072h	USB bus performance register 2	Byte	BUSPERF2
USB + 0074h	USB bus performance register 3	Byte	BUSPERF3
USB + 0078h	Information about number of TX and RX register	Byte	EPINFO
USB + 0079h	Information about the width of RAM and the number of DMA channel register	Byte	RAMINFO
USB + 007Ah	Info. about delay to be applied register	Byte	LINKINFO
USB + 007Bh	Vbus pulsing charge register	Byte	VPLEN
USB + 007Ch	Time buffer available on HS transactions register	Byte	HS_EOF1
USB + 007Dh	Time buffer available on FS transactions register	Byte	FS_EOF1
USB + 007Eh	Time buffer available on LS transactions register	Byte	LS_EOF1
USB + 007Fh	Reset information register	Byte	RST_INFO
USB + 0080h	RX data toggle set/status register	Word	RXTOG
USB + 0082h	RX data toggle enable register	Word	RXTOGEN
USB + 0084h	TX data toggle set/status register	Word	TXTOG
USB + 0086h	TX data toggle enable register	Word	TXTOGEN
Level1 interrupt Control/Status registers			
USB + 00A0h	USB Level 1 interrupt status register	Byte	USB_L1INTS
USB + 00A4h	USB Level 1 interrupt unmask register	Byte	USB_L1INTM
USB + 00A8h	USB Level 1 interrupt polarity register	Byte	USB_L1INTP

Register address	Register name	Manipulation (Byte/Word)	Acronym
USB + 00ACh	USB Level 1 interrupt control register	Byte	USB_L1INTC
Non-indexed EndPoint CSR Region *n stands for endpoint number.* *For example, endpoint 1's n = 1. Valid n = 1 ~ MaxEndPoint.* *MaxEndPoint is hardware configured and the maximum is 15.*			
USB + 0102h	EP0 control status register	Byte	CSR0
USB + 0108h	EP0 received bytes register	Byte	COUNT0
USB + 010Bh	NAK limit register	Byte	NAKLIMT0
USB + 010Fh	Core configuration register	Byte	CONFIGDATA
USB + 0100h +(n)*10h	TXMAP register	Byte	TXMAP(n)
USB + 0102h +(n)*10h	TX CSR register	Byte	TXCSR(n)
USB + 0104h +(n)*10h	RXMAP register	Byte	RXMAP(n)
USB + 0106h +(n)*10h	RX CSR register	Byte	RXCSR(n)
USB + 0108h +(n)*10h	RX Count register	Byte	RXCOUNT(n)
USB + 010Ah +(n)*10h	TXType register	Byte	TXTYPE(n)
USB + 010Bh +(n)*10h	TXInterval register	Byte	TXINTERVAL(n)
USB + 010Ch +(n)*10h	RXType register	Byte	RXTYPE(n)
USB + 010Dh +(n)*10h	RXInterval register	Byte	RXINTERVAL(n)
USB + 010Fh +(n)*10h	Configured FIFO size register	Byte	FIFOSIZE(n)
DMA Channels Control Registers *M stands for DMA channel number.* *For example, DMA channel 1's M = 1. Valid M = 1 ~ MaxDMAChannel.* *MaxDMAChannel is hardware configured and the maximum is 8.*			
USB + 0200h	DMA interrupt status register (word access only)	Word	DMA_INTR
USB + 0210h	DMA limiter register (word access only)	Word	DMA_LIMITER
USB + 0220h	DMA configuration register (word access only)	Word	DMA_CONFIG
USB + 0204h +(M-1)*10h	DMA channel M control register (word access only)	Word	DMA_CNTL_M
USB + 0208h +(M-1)*10h	DMA channel M address register (word access only)	Word	DMA_ADDR_M
USB + 020Ch +(M-1)*10h	DMA channel M byte count register (word access only)	Word	DMA_COUNT_M
EndPoint RX Packet Count Register			

Register address	Register name	Manipulation (Byte/Word)	Acronym
n stands for endpoint number. For example, endpoint 1's n = 1. Valid n = 1 ~ MaxEndPoint. MaxEndPoint is hardware configured and the maximum is 15.			
USB + 0300h +(n)*4h	EPn RXPktCount register	Word	EPnRXPKTCOUNT
Host/Hub Control Registers (Host mode only registers) *n stands for endpoint number. For example, endpoint 1's n = 1. Valid n = 1 ~ MaxEndPoint. MaxEndPoint is hardware configured and maximum is 15.*			
USB + 0480h +8*n h	Transmit endpoint n function address	Word	TXFUNCADDR
USB + 0482h +8*n h	Transmit endpoint n hub/port address	Word	TXHUBADDR
USB + 0484h +8*n h	Receive endpoint n function address	Word	RXFUNCADDR
USB + 0486h +8*n h	Receive endpoint n hub/port address	Word	RXHUBADDR
Debug Function Registers			
USB + 0600h	Debug flag selection control (byte 0, 1, 2, 3)	Word	DFC0R, DFC1R
USB + 0604h	Timing test mode	Word	TM1
USB + 0605h	No response error count	Word	TM1
USB + 0606h	Debug flag UTMI11 sub group selection	Word	DFC2R
USB + 0608h	Hardware version control register	Word	HWVER_DATE
USB + 0610h	Packet sequence record control/OpState record control	Word	PSR_CTRL/ OSR_CTRL
USB + 0611h ~ 0616h	Packet sequence record filter and trigger setting	Word	PSR_CTRL
USB + 0620h ~ 0637h	Debug register	Word	DBG_PRB
USB + 0640h ~ 065fh	Packet sequence PID data/OpState record Data	Word	PSR_DATA/ OSR_DATA
USB + 0684h	SRAM address register	Word	SRAMA
USB + 0688h	SRAM data register (word access only)	Word	SRAMD
USB + 0690h	RISC_SIZE register	Word	RISC_SIZE
USB + 0700h	Reserved register	Word	RESREG
USB + 0704h	HW TXPktRdy	Word	HWTPR
USB + 0708h	HW TXPktRdy enable register	Word	HWTPR_EN
USB + 070Ch	HW TXPktRdy error detection register	Word	HWTPR_ERR

See chapter 3.5 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.6 USBPHY Register File

The full features include USB2.0 PHYD and PHYA macro control registers. A frequency meter for USB2.0 PHYA monitor clock is also included. The registers can be accessed by I2C interface (FT). The default mode is accessing registers by AHB. After 0xfe (I2C access only) is configured to 8'h01, the register file will be in the I2C mode (accessing registers by I2C).

5.6.1 Features

- USB2.0
  - USB2.0 PHYD control registers for PHYD macro setting
  - USB2.0 PHYA control registers for PHYA characteristic tuning
  - Force USB2.0 UTMI interface for FT tests
  - Force USB2.0 PHY analog power-down in ATPG mode
  - Frequency meter for USB2.0 PHYA monitor clock
- USB1.1
  - USB1.1 PHYA control registers for PHYA characteristic tuning
  - Force USB1.1 PHY interface for FT tests
  - Force USB1.1 PHY analog power-down in ATPG mode
- Accessing PHY registers by AHB slave interface
- Accessing PHY registers by I2C interface

5.6.2 USBPHY Register File Block Diagram

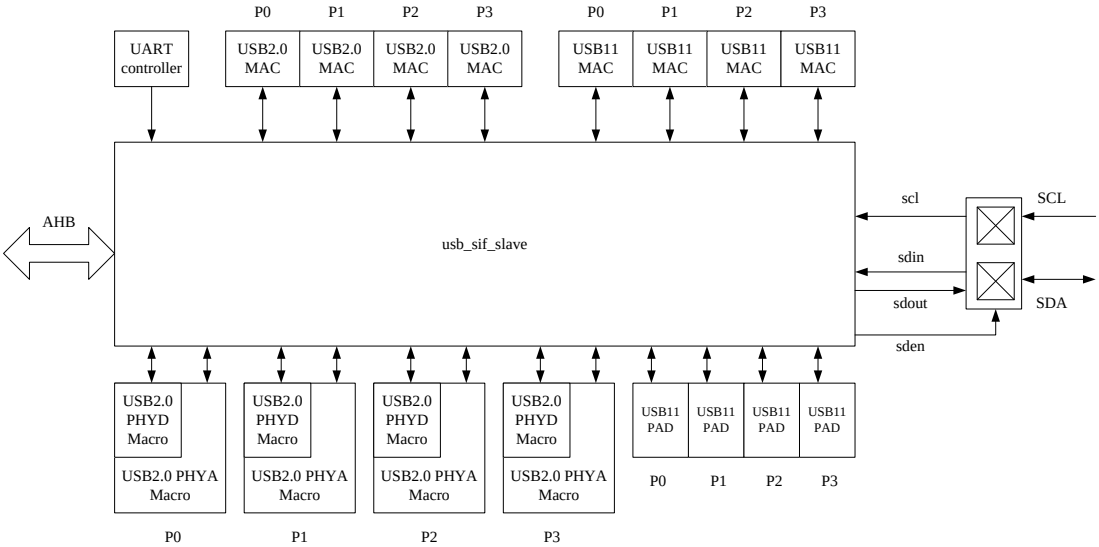
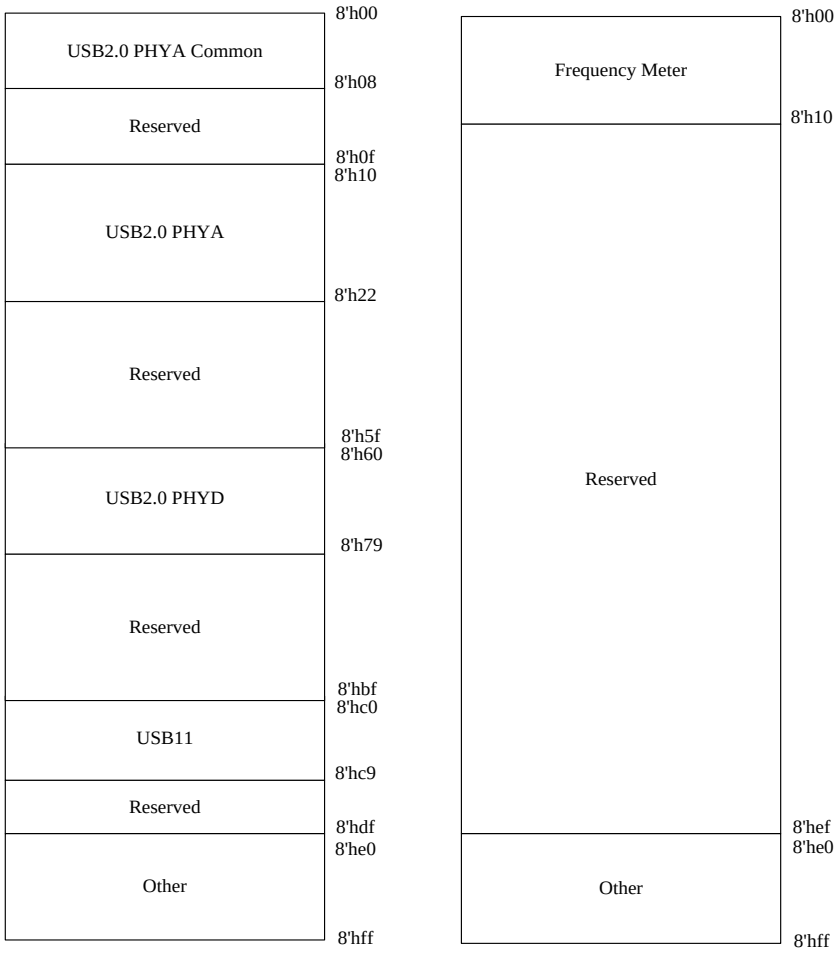


Figure 5-11. USB controller block diagram

5.6.3 Register Definition

- Page base
  - Every page register contains 0x00h ~ 0xefh (USB2.0+ USB1.1)
  - Frequency meter registers in one page (@ 0xff = 8'h0f)
  - 0xf0~0xff: Global register
  - 0xfe[0]: I2C mode, the default value is 0 (accessing register by AHB interface)
    - If switched to I2C mode, configuring 0xfe[0] to be 1'h1 is required.
    - I2C access only
  - 0xffh (RG_PAGE): I2C page register
    - I2C access only
- I2C
  - Accessing different pages by setting up RG_PAGE
  - Default device number: 7'h60
    (*If there are more than one hier, the device number of the second hier. will be 7'h61.)
  - Accessing different pages by setting up RG_PAGE (0xff)
  - Port 0 USB PHY register page: RG_PAGE value: 8'h00
  - Port 1 USB PHY register page: RG_PAGE value: 8'h01
  - Port 2 USB PHY register page: RG_PAGE value: 8'h02
  - Port 3 USB PHY register page: RG_PAGE value: 8'h03
  - Frequency meter register page: RG_PAGE value: 8'h0f
- AHB
  - Accessing different pages by different base addresses
  - Supports max. four ports. Base address: 800h, 900h, a00h, b00h
    - Port 0 register base address: 800h
    - Port 1 register base address: 900h
    - Port 2 register base address: a00h
    - Port 3 register base address: b00h
    - Frequency meter registers base address: f00h



5.6.3.1 Function Address Register

See chapter 3.6 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.7 SPI Interface Controller

5.7.1 Introduction

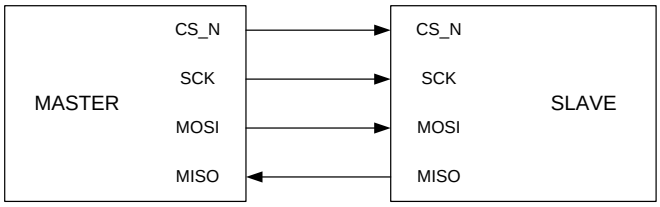


Figure 5-12. Pin connection between SPI master and SPI slave

The SPI interface is a bit-serial, four-pin transmission protocol. Figure 5-12 is an example of the connection between the SPI master and SPI slave. In MT6797, the SPI interface controller is a master responsible of the data transmission with the slave.

5.7.2 Pin Description

Table 5-3. SPI controller interface of MT6797

Signal name	Type	Description
CS_N	O	Low active chip selection signal
EWAIT	O	The (bit) serial clock
MOSI	O	Data signal from master output to slave input
MISO	I	Data signal from slave output to master input

5.7.2.1 Transmission Formats

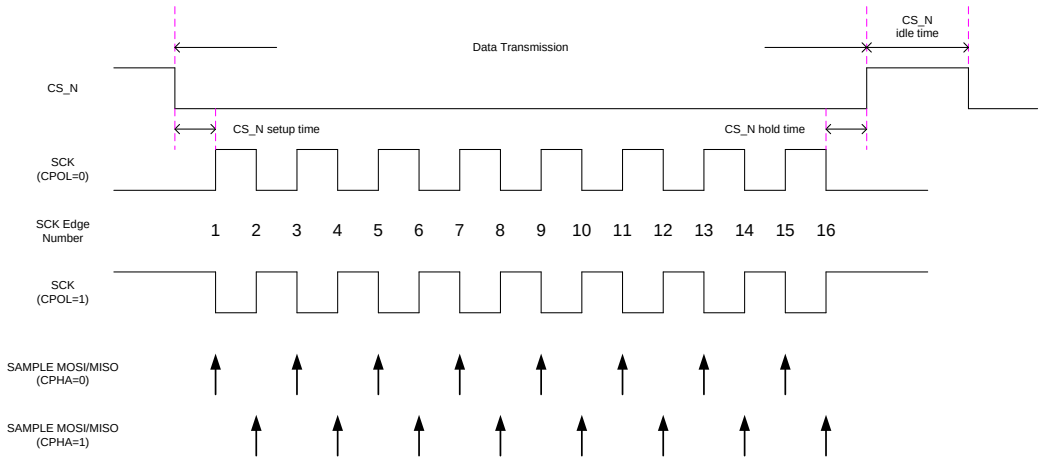


Figure 5-13. SPI transmission formats

Figure 5-13 shows the waveform during the SPI transmission. The low active CS_N determines the start point and end point of one transaction. The CS_N setup time, hold time and idle time are also depicted.

CPOL defines the clock polarity in the transmission. Two types of polarity can be adopted, i.e. polarity 0 and polarity 1. Figure 5-13 shows both of the clock polarity (CPOL) as examples.

CPHA defines the legal timing to sample MOSI and MISO. Two different methods can be adopted.

5.7.3 Features

The features of the SPI controller (master) are:

Configurable CS_N setup time, hold time and idle time

- Programmable SCK high time and low time
- Configurable transmitting and receiving bit order
- Two configurable modes for the source of the data to be transmitted. 1) In TX DMA mode, the SPI controller automatically fetches the transmitted data (to be put on the MOSI line) from memory; 2) In TX FIFO mode, the data to be transmitted on the MOSI line are written to FIFO before the start of the transaction.
- Two configurable modes for destination of the data to be received. 1) In RX DMA mode, the SPI controller automatically stores the received data (from MISO line) to memory; 2) In RX FIFO mode, the received data keep being in RX FIFO of the SPI controller. The processor must read back the data by itself.
- Adjustable endian order from/to memory system
- Programmable byte length for transmission
- Unlimited length for transmission. This is achieved by the operation of PAUSE mode. In PAUSE mode, the CS_N signal will keep being active (low) after the transmission. At this time, the SPI controller is in PAUSE_IDLE state, ready to receive the resume command. The state transition is shown in Figure 5-14.
- Configurable option to control CS_N de-assert between byte transfers. The controller supports a special transmission format called CS_N de-assert mode. Figure 5-15 illustrates the waveform in this transmission format.

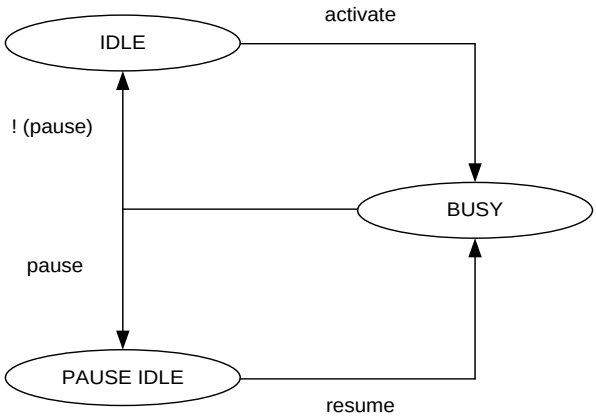


Figure 5-14. Operation flow with or without PAUSE mode

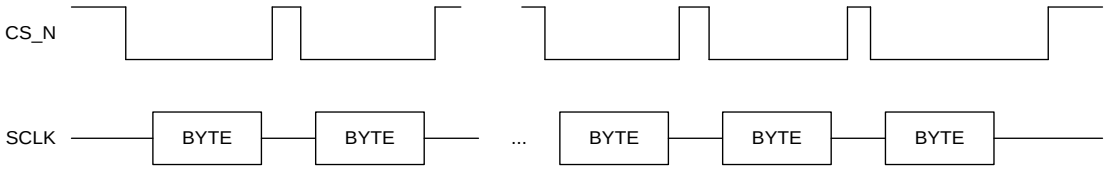


Figure 5-15. CS_N de-assert mode

5.7.4 Block Diagram

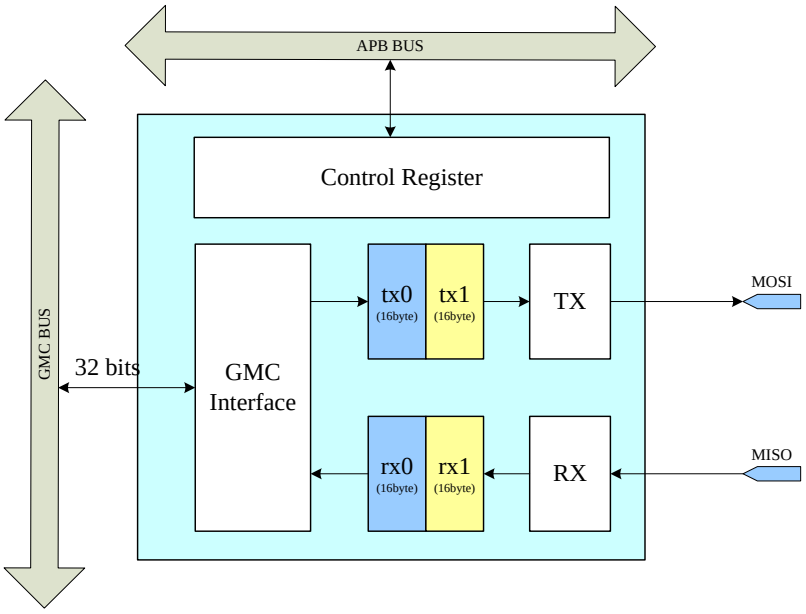


Figure 5-16. Block diagram of SPI

5.7.5 Register Definition

See chapger 3.7 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

Register bank	Base address
spi0	+1100a000h
spi1	+11012000h
spi2	+11018000h
spi3	+11019000h
spi4	+1101a000h
spi5	+1101b000h

There are six SPI IPs in this SOC. The usage of the registers is the same except that the base address must be changed to respective one.

5.8 AUXADC

5.8.1 Introduction

The auxiliary ADC unit is used to identify the plugged peripheral and perform temperature measurement. There are 16 input channels allowing diverse applications, such as temperature measurement and light sensor.

Each channel only operates in the immediate mode. The time-trigger mode is removed now. In the immediate mode, the A/D converter samples the value once only when the flag in the AUXADC_CON1 register is set. For example, if the flag IMM0 in AUXADC_CON1 is set, the A/D converter will sample the data for channel 0. The IMM flags have to be cleared and set again to initialize another sampling. The value sampled for channel 0 is stored in register AUXADC_DAT0, and the value for channel 1 is stored in register AUXADC_DAT1 and so on. If the AUTOSET(x) flag in the register AUXADC_CON0 is set, the auto-sampling function will be enabled in channel(x). The A/D converter samples the data for the channel(x) in which the corresponding data register is read. For example, in the case the AUTOSET0 flag is set. When the data register AUXADC_DAT0 is read, the A/D converter will sample the next value for channel 0 immediately.

If multiple channels are selected at the same time, the task will be performed sequentially on every selected channel from high to low channel. For example, if AUXADC_CON1 is set to 0x7f, i.e. all 7 channels are selected, the state machine in the unit will start sampling from channel 6 to channel 0 and saves the values of each input channel in respective registers. The same process also applies to the timer-triggered mode.

The PUWAIT_EN bit in register AUXADC_CON3 is used to power up the analog port in advance, ensuring that the power is ramped up to the stable state before A/D converter starts the conversion. The analog part is automatically powered down after the conversion is completed.

Besides, there are several embedded temperature sensors. The module accepts signals from module of thermal controller to measure the temperature of the embedded sensors. The measurement result is able to be read in the command registers in the module of thermal controller.

5.8.2 Features

Table 5-4 describes the features in the AUXADC module.

Table 5-4. AUXADC feature list

Item	Main function	Description
1	Immediate analog-digital conversion	In immediate mode, it supports auto-set option. In time-trigger mode, it supports auto-clear option.

Item	Main function	Description
2	Background detection and interrupt	The related command registers: AUXADC_DET_VOLT, AUXADC_PERIOD, AUXADC_DEBT, AUXADC_SEL
3	Temperature measurement	

5.8.3 Block Diagram

SW controls the AUXADC through the APB bus. Once the hardware receives the command, it will trigger AUXADC channel sampling automatically. SW polls the status register or waits for interrupts from the CPU.

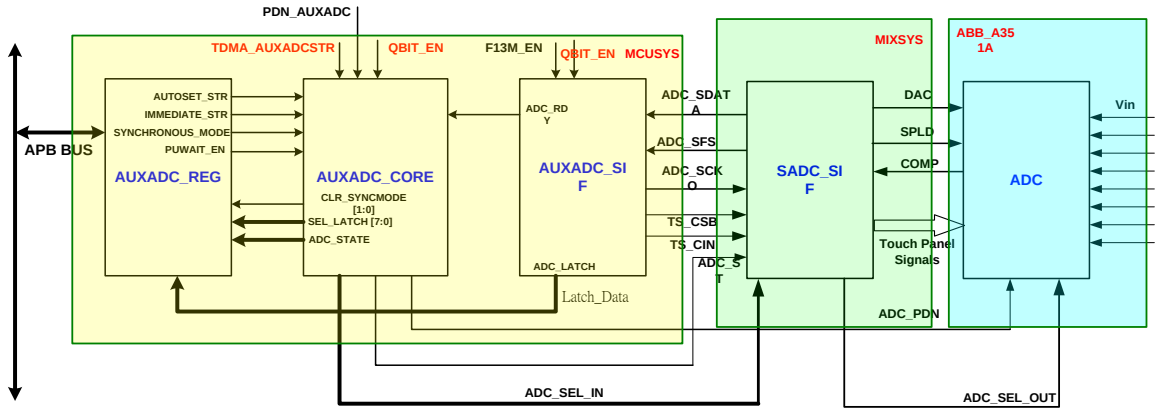


Figure 5-17. Block diagram of AUXADC

5.8.4 Theory of Operation

5.8.4.1 SAR ADC

Successive-approximation-register (SAR) ADC provides low power consumption, cost-effective and medium resolution. The AUXADC is SAR ADC architecture.

Here is an 8-bit conversion example. V_{REF} is the reference voltage of AUXADC.

The AUXADC implements a binary search algorithm. An initial register V_{DA} value compared to the input voltage V_{IN} is the mid-value between (2^8-1) and 0. The value represents $V_{REF}/2$. If V_{IN} is bigger than V_{DA}, the output of comparison will be 1, and the MSB-bit will be 1. On the contrary, the MSB bit will be 0. Subsequently, bit 7 will be set to 1, and another comparison is done. Bit 6 to bit 0 will be executed as the previous action. Then, the 8-bit digital value will be available.

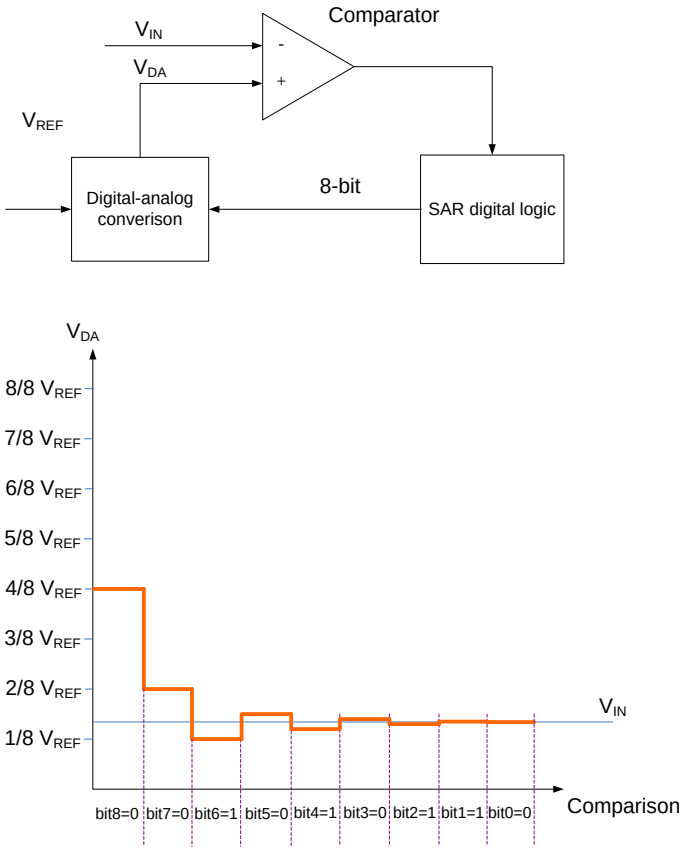


Figure 5-18. SAR ADC architecture and conversion

5.8.4.2 Design Partition

Table 5-5 is the design partition.

Table 5-5. AUXADC design partition

Sub module name (hier1)	Description
AUXADC	Top module
AUXADC_REG	APB command registers
AUXADC_SIF	ADC serial interface with the module SADC_SIF
AUXADC_DEBUG	Debugging signal selection
AUXADC_MONITOR	Background detection and generate interrupt
AUXADC_CORE	AUXADC state machine and handle sampling sequence
SADC_SIF	Generate signals to analog part and transfer ADC result to the module AUXADC

5.8.5 Register Definition

See chapter 3.8 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.9 I2C/SCCB Controller

5.9.1 Introduction

I2C (Inter-IC)/SCCB (Serial Camera Control Bus) is a two-wire serial interface. The two signals are SCL and SDA. SCL is a clock signal that is driven by the master. SDA is a bi-directional data signal that can be driven by either the master or the slave. This generic controller supports the master role and conforms to the I2C specification.

5.9.2 Features

- I2C compliant master mode operation
- Adjustable clock speed for LS/FS mode operation
- Supports 7-bit/10-bit addressing
- Supports high-speed mode
- Supports slave clock extension
- START/STOP/REPEATED START condition
- Manual transfer mode
- Multi-write per transfer
- Multi-read per transfer
- Multi-transfer per transaction
- Combined format transfer with length change capability.
- Active drive/wired-and I/O configuration
- Repeated start multiple transfer

5.9.2.1 Manual Transfer Mode

The controller offers manual mode.

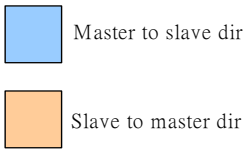
When the manual mode is selected, in addition to the slave address register, the controller has a built-in 8-byte deep FIFO which allows MCU to prepare up to eight bytes of data for a write transfer, or read up to eight bytes of data for a read transfer.

5.9.2.2 Transfer Format Support

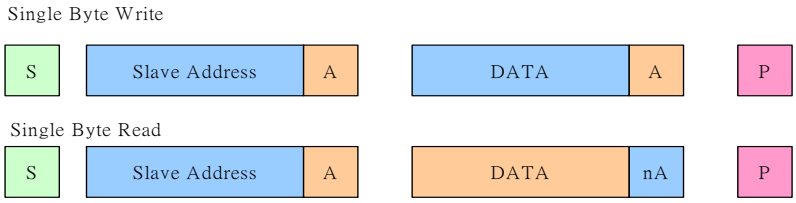
This controller is designed to be as generic as possible in order to support a wide range of devices that may utilize different combinations of transfer formats. Here are the transfer format types supported through different software configurations:

Wording convention note

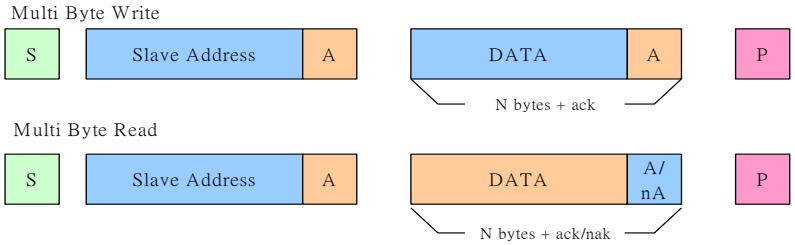
- Transfer = Anything encapsulated within a Start and Stop or Repeated Start.
- Transfer length = Number of bytes within the transfer
- Transaction = This is the top unit. Everything combined equals 1 transaction.
- Transaction length = Number of transfers to be conducted.



Single byte access

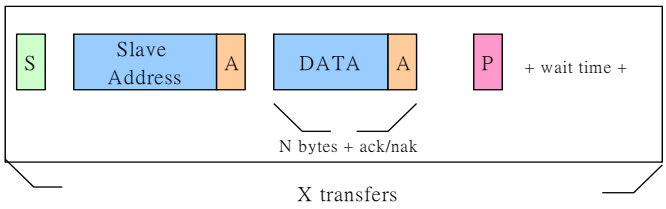


Multi byte access

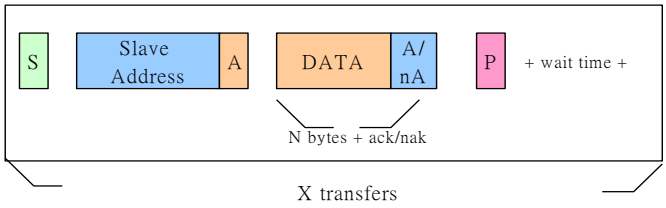


Multi-byte transfer + multi-transfer (same direction)

Multi Byte Write + Multi Transfer

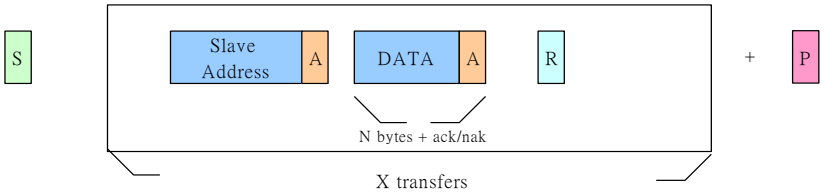


Multi Byte Read + Multi Transfer

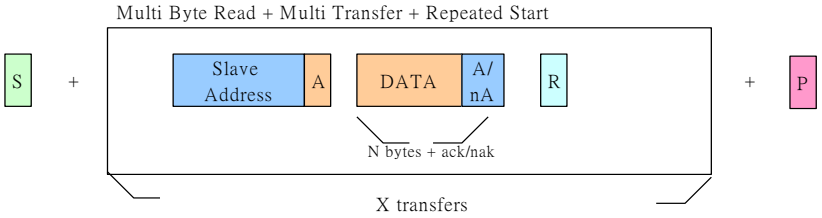


Multi-byte transfer + multi-transfer w RS (same direction)

Multi Byte Write + Multi Transfer + Repeated Start



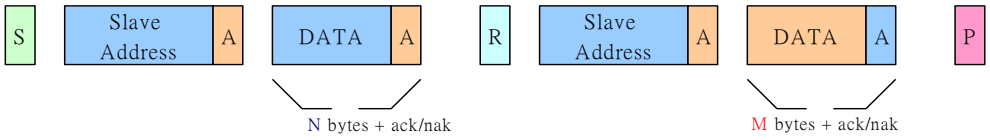
Multi Byte Read + Multi Transfer + Repeated Start



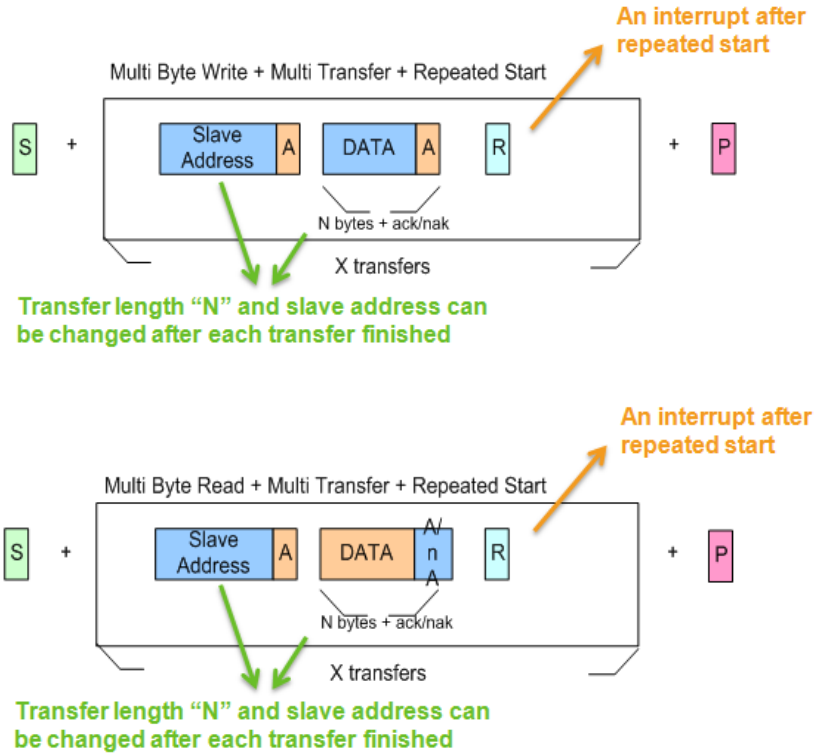
Combined write/read with Repeated Start (direction change)

Note: Only supports write and then read sequence. Read and then write is not supported.

Combined Multi Byte Write + Multi Byte Read



Repeated start multiple transfer (write/read) *(I2C_dual does not support this mode.)*



5.9.3 Block Diagram

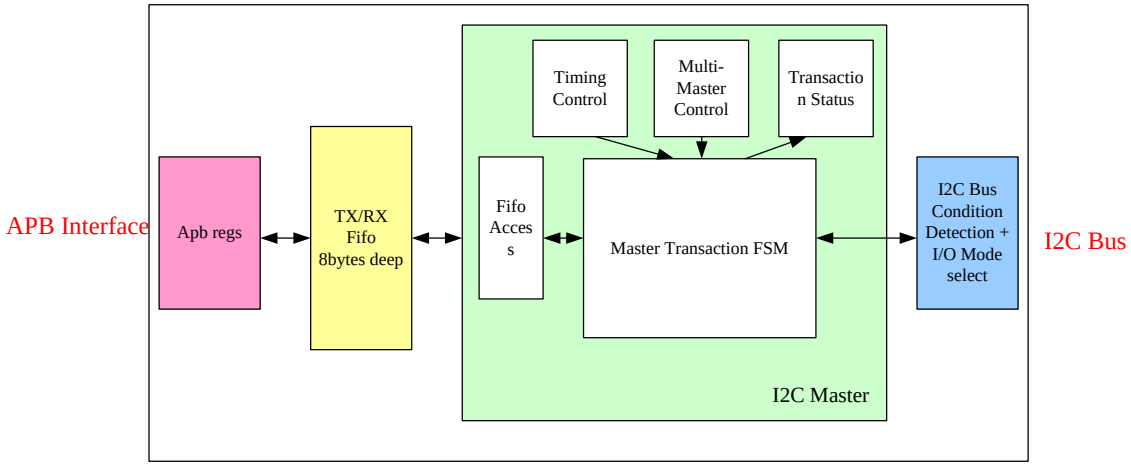


Figure 5-19. Block diagram of I2C

5.9.4 Register Definition

I2C bus	I2C number	Base address	Feature
I2C0	I2C0	0x11007000	Supports DMA
I2C1	I2C1	0x11008000	Supports DMA
I2C2	I2C2	0x11009000	Supports DMA
	I2C2_imm	0x11013000	Supports DMA
I2C3	I2C3	0x1100D000	Supports DMA
	I2C3_imm	0x11014000	Supports DMA
I2C4	I2C4	0x11011000	Supports DMA
I2C5	I2C5	0x1101C000	Supports DMA
I2C6(APPM)	I2C6 (I2C_dual)	0x1100E000	Channel 1 supports DMA
I2C7(GPUPM)	I2C7	0x11010000	Supports DMA

There are seven I2C buses in this SOC. The usage of the registers is the same except that the base address must be changed to respective one.

There are only one I2C_dual bus in this SOC. Note that I2C_dual does not support **repeated start multiple transfer (write/read).**

See chapter 3.9 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.10 Pulse-Width Modulation (PWM)

5.10.1 Introduction

Seven generic pulse-width modulators are implemented to generate pulse sequences with programmable frequency and duration for LCD backlight, charging or other purposes. Before enabling PWM, the pulse sequences must be prepared in the memory or registers. Then PWM will read the pulse sequences to generate random waveform to meet all kinds of applications (see Figure 5-20).

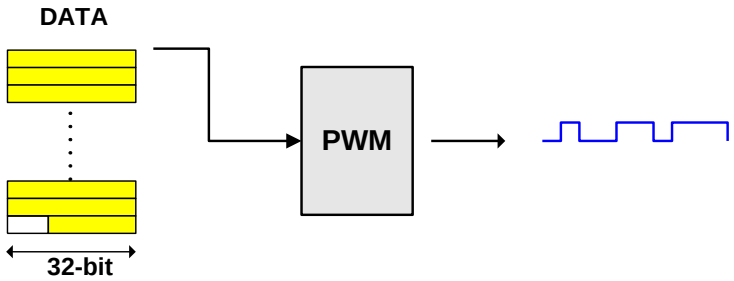
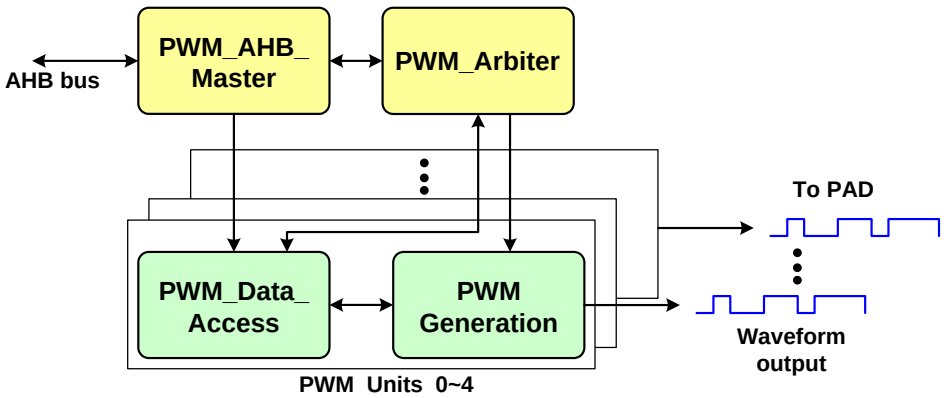


Figure 5-20. Generation procedure of PWM

5.10.2 Features

- Old mode, FIFO mode
- Periodical memory and random mode
- Sequential output mode and 3DLCM mode

5.10.3 Block Diagram



5.10.4 Register Definition

See chapter 3.10 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.10.5 Clock Source Selection

Mode	OLD_MODE	CLKSEL_OLD	CLKSEL	CLKSRC
Old mode	1	Don't care	0	bclk
Old mode	1	1	1	32kHz (used in old mode only)
Old mode	1	0	1	bclk/1625
FIFO mode	0	Don't care	0	bclk
FIFO mode	0	Don't care	1	bclk/1625

5.10.6 Output Frequency and Duty Cycle

Mode	Duty cycle	Output frequency
Old mode	PWM_THRESH/(PWM_DATA_WIDTH + 1)	SRCLK/[CLK_DIV*(PWM_DATA_WIDTH + 1)]
FIFO mode	Output freq. = (SRCLK*duty cycle)/4	

Note: bclk can be selected as 26MHz or 66MHz(bus clock) by PWM_CK_26M_SEL.

5.11 General-Purpose Timer (GPT)

5.11.1 Introduction

The GPT includes five 32-bit timers and one 64-bit timer. Each timer has four operation modes, which are ONE-SHOT, REPEAT, KEEP-GO and FREERUN, and can operate on one of the two clock sources, RTC clock (32.768kHz) and system clock (13MHz).

5.11.2 Features

The four operation modes for GPT are ONE-SHOT, REPEAT, KEEP-GO and FREERUN. See Table 5-6 for the functions of each mode.

Table 5-6. Operation mode of GPT

Mode	Auto stop	Interrupt	Increases when EN=1 and ...	When COUNTn equals COMPAREn	Example: Compare is set to 2 **Bold means interrupt*
ONE-SHOT	Yes	Yes	Stops when COUNTn equals to COMPAREn	EN is reset to 0.	0,1,**2**,2,2,2,2,2,2,2,2,2,...
REPEAT	No	Yes		Count is reset to 0.	0,1,**2**,0,1,**2**,0,1,**2**,0,1,**2**...
KEEP-GO	No	Yes	Reset to 0 when overflow		0,1,**2**,3,4,5,6,7,8,9,10,...
FREERUN	No	No	Reset to 0 when overflow		0,1,2,3,4,5,6,7,8,9,10,...

Each timer can be programmed to select the clock source, RTC clock (32.76kHz) or system clock (13MHz). After the clock source is determined, the division ratio of the selected clock can be programmed. The division ratio can be fine-granulated as 1, 2, 3, 4 to 13 and coarse-granulated as 16, 32 and 64.

5.11.3 Block Diagram

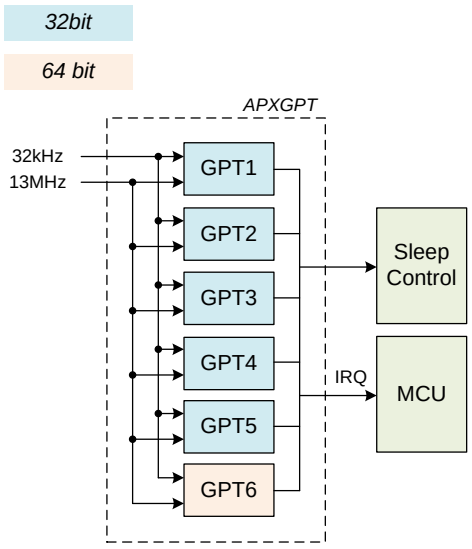


Figure 5-21. Block diagram of APXGPT

5.11.4 Register Definition

See chapter 3.11 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.11.5 Programming Guide

To program and use GPT, note that:

- The counter value can be read any time even when the clock source is RTC clock.
- The compare value can be programmed any time.

For the GPT6 64-bit timer, the read operation of the 64-bit timer value will be separated into two APB reads since an APB read is of 32-bit width. To perform the read of 64-bit timer value, the lower word should be read first then the higher word. The read operation of lower word freezes the "read value" of the higher word but does not freeze the timer counting. This ensures that the separated read operation acquires the correct timer value. If both two tasks, e.g. task A and task B, perform the read of 64-bit timer value, task A first reads the lower word of the value, and task B reads the lower word of the value. Either of the tasks reads the higher word of timer value, and the obtained value will be the time when task B reads the lower word of timer value. To guarantee task A reads the correct 64-bit timer value, some software procedures are required, e.g. the semaphore.

5.12 IRTX

5.12.1 Introduction

IRTX module provides the “Consumer IR” feature. According to Wikipedia, *“Consumer IR, consumer infrared, or CIR, refers to a wide variety of devices employing the infrared electromagnetic spectrum for wireless communications. Most commonly found in television remote controls, infrared ports are equally ubiquitous in consumer electronics, such as PDAs, laptops, and computers. The functionality of CIR is as broad as the consumer electronics that carry it. For instance, a television remote control can convey a "channel up" command to the television, while a computer might be able to surf the internet solely via CIR. The type, speed, bandwidth, and power of the transmitted information depends on the particular CIR protocol employed.”*

MediaTek’s IRTX provides three main stream CIR protocols: NEC, RC5 and RC6. It also supports Android IR API in which the OS version is above KitKat.

5.12.2 NEC Protocol

5.12.2.1 Introduction

The NEC protocol uses pulse distance encoding of the bits. Each pulse is a 560µs long 38kHz carrier burst (about 21 cycles). A logical "1" takes 2.25ms to transmit, while a logical "0" takes only half of that, i.e. 1.125ms, as shown in Figure 5-22. The recommended carrier duty-cycle is 1/4 or 1/3.

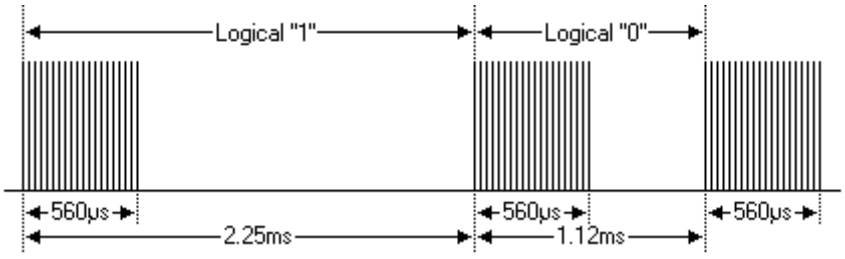


Figure 5-22. Logic representation for NEC protocol

Figure 5-23. Pulse train in transmission of NEC protocol

The picture above shows the typical pulse train of the NEC protocol. With this protocol, the LSB is transmitted first. In this case Address $59 and Command $16 is transmitted. A message is started by a 9ms AGC burst, which is used to set up the gain of the earlier IR receivers. This AGC burst is then followed by a 4.5ms space, which is then followed by the Address and Command. Address and Command are transmitted twice. In the second time, all bits are inverted and can be used for verification of the received message. The total transmission time is constant because every bit is repeated with its inverted length.

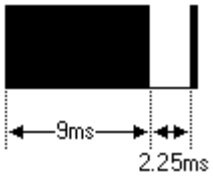


Figure 5-24. A message is started by a 9ms AGC burst

A command is transmitted only once even when the key on the remote control remains pressed. Every 110ms a repeat code is transmitted for as long as the key remains down. This repeated code is simply a 9ms AGC pulse followed by a 2.25ms space and a 560µs burst.

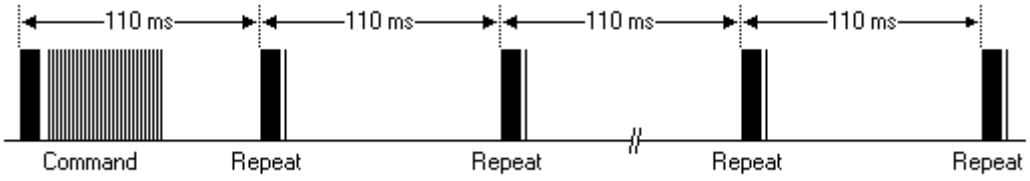


Figure 5-25. A repeat code is transmitted every 110ms

5.12.2.2 Features

- 8-bit address and 8-bit command length
- Address and command are transmitted twice for reliability.
- Pulse distance modulation
- 38kHz carrier frequency
- 1.125ms or 2.25ms bit time

5.12.3 Philips RC-5 Protocol

5.12.3.1 Introduction

The protocol uses bi-phase modulation (or so-called Manchester coding) of a 36kHz IR carrier frequency. All bits are of equal length to 1.778ms in this protocol, with half of the bit time filled with a burst of the 36kHz carrier and the other half being idle. A logical zero is represented by a burst in the first half of the bit time. A logical one is represented by a burst in the second half of the bit time. The pulse/pause ratio of the 36kHz carrier frequency is 1/3 or 1/4 to reduce power consumption.

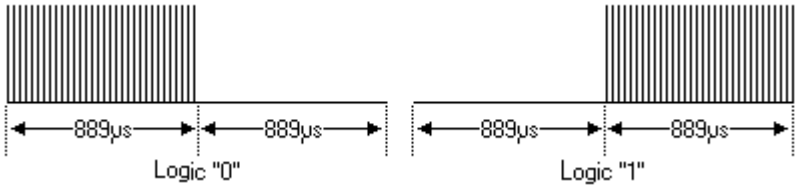


Figure 5-26. Coding method for RC5 protocol

5.12.3.2 Features

- 5-bit address and 6-bit command length (7 command bits for RC5X)
- Bi-phase coding (aka Manchester coding)
- 36kHz carrier frequency
- 1.778ms constant bit time (64 cycles of 36 kHz)
- Manufacturer Philips

5.12.4 Philips RC-6 Protocol

5.12.4.1 Introduction

RC-6 is the successor of the RC-5 protocol. Like RC-5 the new RC-6 protocol is also defined by Philips. It is a very versatile and well defined protocol. Because of this versatility its original definition is many pages long. Here we only summarize the most important properties of this protocol. RC-6 signals are modulated on a 36kHz Infra Red carrier. The duty cycle of this carrier has to be between 25% and 50%.

Data are modulated using Manchester coding; this means that each bit (or symbol) has both a mark and space in the output signal. If the symbol is a "1", the first half of the bit time will be a mark and the second half a space. If the symbol is a "0", the first half of the bit time will be a space and the second half a mark.

The main timing unit is 1t, which is 16 times the carrier period (1/36k*16 = 444μs). With RC-6, total different symbols are defined:

- The leader pulse has a 6t mark time (2.666ms) and 2t space time (0.889ms). This leader pulse is normally used to set up the gain of the IR receiver unit.

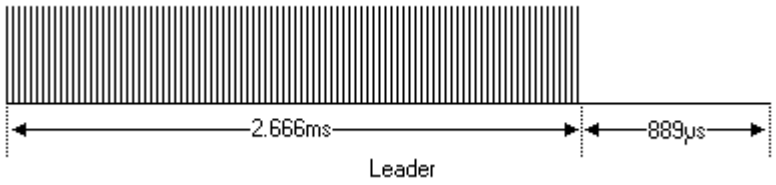


Figure 5-27. Lead pulse in RC6 protocol

- Normal bits have 1t mark time (0.444ms) and 1t space time (0.444ms). "0" and "1" are encoded by the position of the mark and space in the bit time.

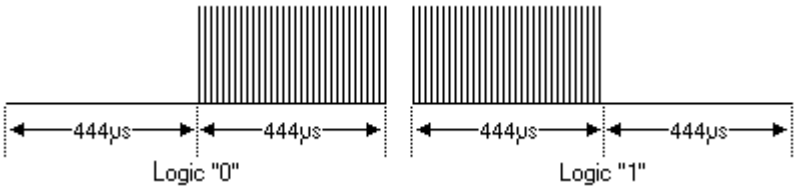


Figure 5-28. Coding method for RC6 protocol

- Trailer bits have 2t mark time (0.889ms) and 2t space time (0.889ms). "0" and "1" are encoded by the position of the mark and space in the bit time.

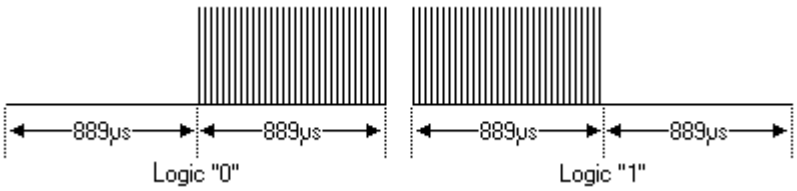


Figure 5-29. Trailer pulse in RC6 protocol

The leader and trailer symbols are only used in the header field of the messages.

5.12.4.2 Features

- Different modes of operation, depending on the intended use

- Dedicated Philips modes and OEM modes
- Variable command length, depending on the operation mode
- Bi-phase coding (aka Manchester coding)
- 35kHz carrier frequency
- Manufacturer Philips

5.12.5 Register Definition

See chapter 3.12 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.13 Audio System

5.13.1 Introduction

The audio system provides the audio data exchange ability between AP, internal modem and external components. The interfaces are listed as the following:

- Master/Slave I2S input interface with SRC x 1
- Master I2S output x 2
- Master I2S input x 1
- Slave PCM interface for internal MODEM x 1
- Master/Slave PCM interface with SRC for internal/external MODEM x 1
- Proprietary audio interface for MTK PMIC x 1
- 8-ch Master I2S output x1 with TDM support

5.13.2 Features

- Audio playing
  - Supports 8, 11.025, 12, 16, 22.05, 24, 32, 44.1, 48, 96, and 192kHz sampling rate output
  - Supports playing stereo data
- Audio recording
  - Supports 8, 16, 32, 48kHz sampling rate recording
  - Support 48, 96, 192kHz sampling rate high resolution recordint
  - Supports stereo recording
- Speech
  - Supports dual MIC
  - Supports 8/16kHz sampling rate recording
  - Supports side tone filter
  - Master/Slave PCM interface with SRC function
- I2S
  - Supports master/slave input mode
  - Supports master output mode
  - Supports 16/24-bit stereo data
  - Supports 8, 11.025, 12, 16, 22.05, 24, 32, 44.1, 48, 88.2, 96, 176.4, and 192kHz sampling rate in master mode
  - Supports EIAJ/I2S format
- TDM
  - Supports 2, 4, and 8 channels master output mode
  - Supports 16/24-bit stereo data

– Supports 8, 11.025, 12, 16, 22.05, 24, 32, 44.1, 48, 88.2, 96, 176.4, and 192kHz sampling rate in master mode

- Hardware gain function with higher resolution to enhance the audio quality and flexibility of interconnection
- Flexible interconnection system to make data exchange between interfaces without intervention of CPU
- Feed-Forward Active Noise Cancellation (FF *ANC*)
- Voice Wakeup (VOW)

5.13.3 Block Diagram

The diagram and table below show the flexibility on the interconnection between audio interfaces.

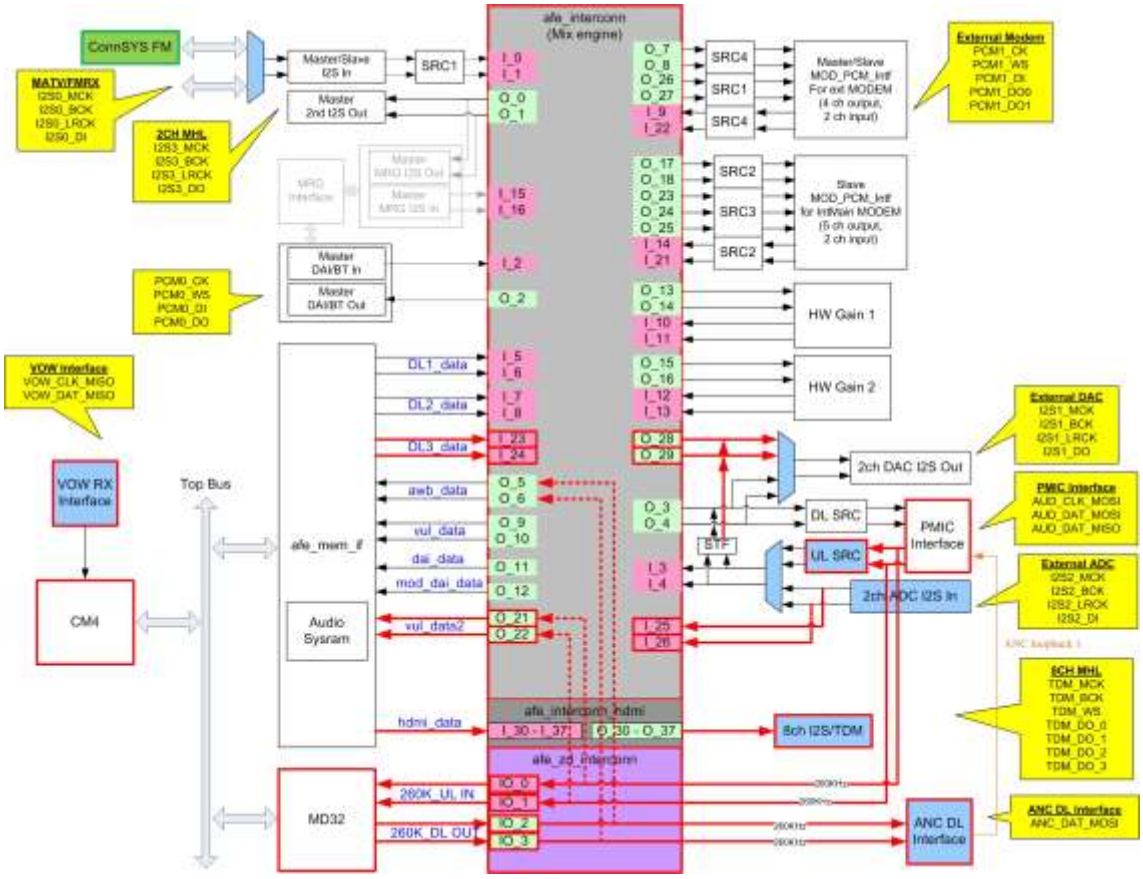


Figure 5-30. Block diagram of audio sys

5.13.4 Register Definition

See chapter 3.13 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.14 BTIF

5.14.1 Overview

Bluetooth Interface (BTIF) is designed in SOC (BT+GSM) in order to be instead of the UART interface between BT chip and baseband chip. As the UART design, BTIF is an APB slave and can transmit or receive data by MCU access or through DMA/VFIFO.

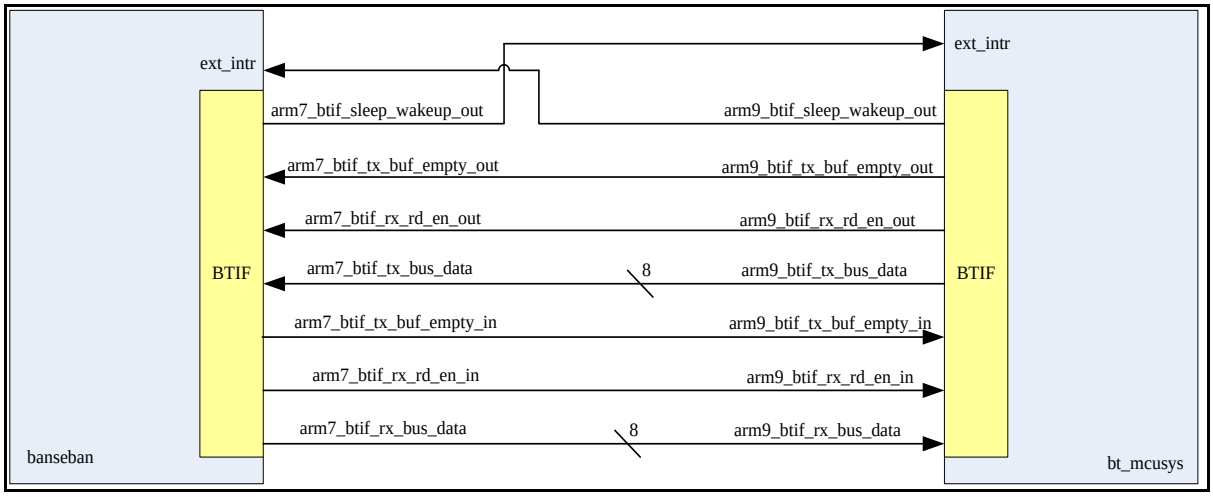


Figure 5-31. Interface connection between BT and baseband system

Detailed block diagram of BTIF is shown in Figure 5-32 and Table 5-7. The Control Register (CR) of BTIF can be set by MCU or DMA through APB interface. The CR of BTIF in btif_inntr_reg can setup FIFO enable, interrupt, wakeup event and so on. Btif_TX_fifo and btif_RX_fifo are used to temporarily store the transmitted and received data. The BTIF transmission is asynchronous handshake. Therefore, btif_RX_async is required to sync the received data.

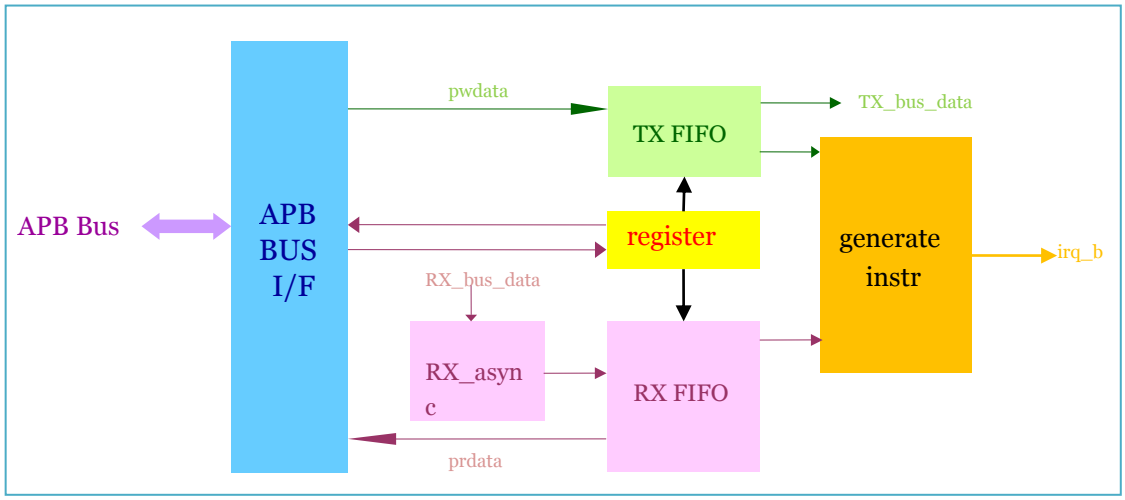


Figure 5-32. Block diagram of BTIF

Table 5-7. BTIF: Design partition

Sub module name (hier1)	Description
btif_intr_reg	APB bus configures UART register and generates interrupt.
btif_RX_async	Synchronously receives control and data signal
btif_RX_fifo	Receives data from baseband
btif_TX_fifo	Transmits data to baseband

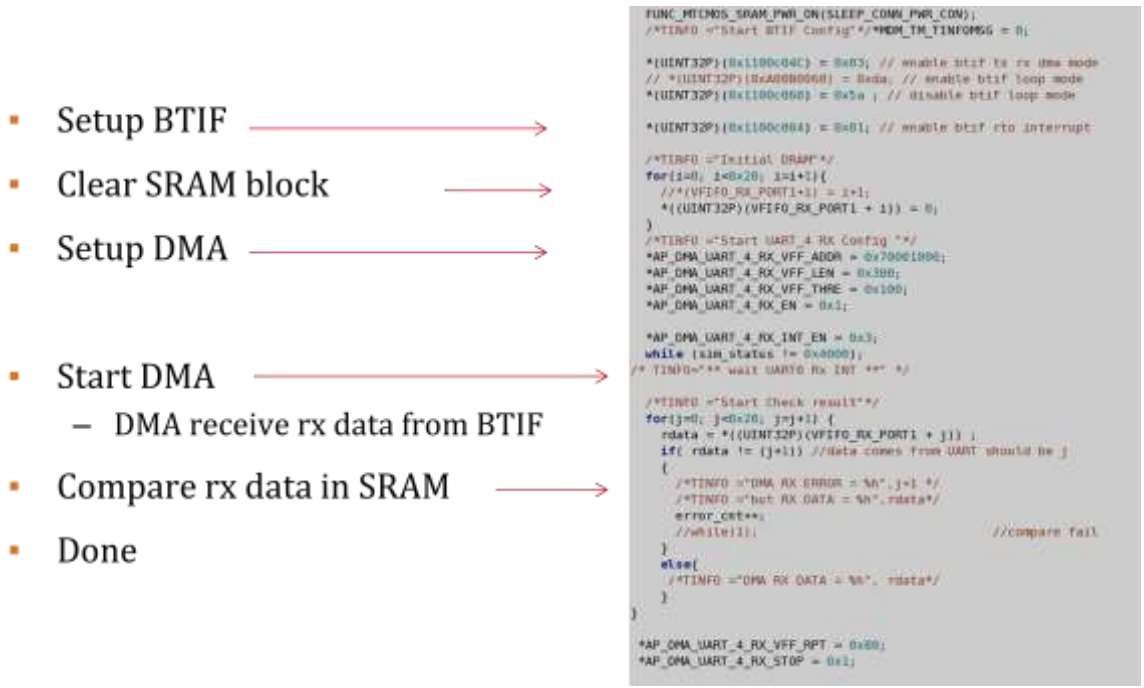
5.14.2 Theory of Operation

Table 5-8 lists the function of BTIF for test. There are four test items. The first is to test the transmitted data; the second received data and the third the interrupt functionality. Finally, the register is configured to verify the settings of BTIF.

Table 5-8. BTIF functions

Item	Main function	Description
1	TX FIFO	Transmits data to baseband
2	RX FIFO	Receives data from baseband
3	Interrupt	Generates BTIF interrupt
4	Register	APB bus configures BTIF register.

The software programming guide is listed below.



In simulating the whole chip, four patterns are used to verify BTIF, as shown in the following table.

Table 5-9. Test patterns for whole chip simulation

Test list	Test name	Description
conn_mcu_btif.list	conn_btif	Tests DMA + BTIF transmitted and received data
conn_mcu_btif_toggle.list	conn_btif_toggle	Tests toggle coverage of interface
conn_mcu_btif_swrst.list	conn_btif_swrst	Tests interrupt
spm_wakeup.list	conn_btif_wake_ap	Tests wakeup signal to interrupt other subsys

5.14.3 Register Definition

See chapter 3.14 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.15 USIM

5.15.1 Introduction

USIM is a peripheral IP used to connect CPU with SIM card due to that the electrical behavior of SIM card is constrained by ISOIEC7816-3 SPEC, CPU touch SIM card through USIM.

5.15.1.1 Data Format

The USIM interface is a half duplex asynchronous communication port and its data format (i.e. the character frame in the left of document) is composed of ten consecutive bits: a start bit in state Low, eight information bits, and a tenth bit used for parity check. The character frame can be divided into two modes as the following:

Direct Convention Mode (Field CONV of R_USIMx_CONF = 0)

SB D0 D1 D2 D3 D4 D5 D6 D7 PB

SB: Start bit (in state Low)

Dx: Data byte (LSB is the first and logic level ONE is in state High)

PB: Parity check bit

Inverse Convention Mode (Field CONV of R_USIMx_CONF =1)

SB N7 N6 N5 N4 N3 N2 N1 N0 PB

SB: Start bit (in state Low)

***Nx*:** Data byte (MSB is the first and logic level ONE is in state Low)

PB: Parity check bit

If the receiver gets a wrong parity bit, it will respond by pulling the SIMDATA Low to inform the transmitter, and the transmitter will retransmit the character frame.

When the USIM interface is the transmitter, it will take total 14 bits guard period whether the error response appears (as shown in Figure 5-33). If the USIM card shows the error response, which starts 0.5 bit after the PB, it may last for 1~2 bit periods, and the USIM interface will retransmit the previous character again else it will transmit the next character.

When the USIM interface is the receiver, it starts error response 0.5 bit after the PB and last for 1.5 bit period. The USIM card will retransmit the previous character after the response. Therefore, the USIM interface has to detect the start bit of the character then.

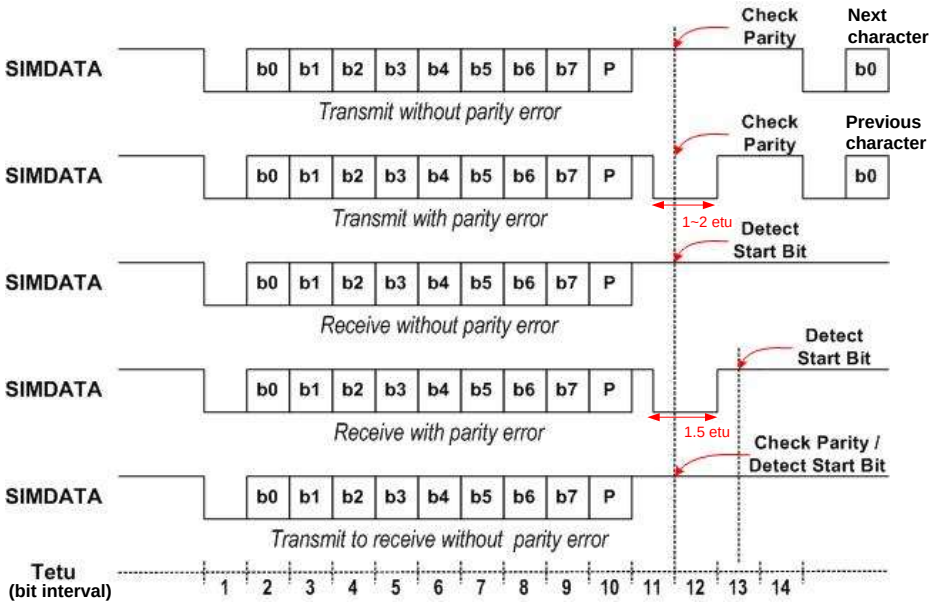


Figure 5-33. USIM interface character frame timing diagram

Any received character frame will be decoded to a byte by the USIM interface according to the convention mode. MCU can read the bytes from FIFO by reading register R_USIMx_DATA. On the other hand, MCU can write bytes into FIFO by writing the register. Then the USIM interface will encode these bytes to character frames according to the convention mode and send them to the USIM card.

5.15.2 Features

- Supports IR and AL type SIM card
- Supports DMA and CPU data transfer
- Supports T=0 and T=1 protocol

5.15.3 Design Partition

Table 5-10. Sub-module description

Module name	Description
usim_async	Syncs two clock domain signals
usim_atrcon	Activates/Deactivates/Warm resets SIM card as ISOIEC7816-3 SPEC
usim_clock	Generates many frequency clocks to each module from clock f13m_ck
usim_dma	Controls the hand shake with DMA interface
usim_fifo	There is an FIFO in this module; for the half-duplex behavior, this module will separate read/write priority of masters.

Module name	Description
usim_irqcon	Creates interrupt
usim_reg	Register control
usim_stb	Monitors SIM card bus status and syncs simdata_in signal
usim_topt1	T=0/T=1protocol control
usim_debug_flag	Debug control
usim_uart	Translates bit sequence to byte and byte to bit sequence

5.15.4 Register Definition

See chapter 3.15 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.16 WIFI-HIF

5.16.1 General Description

This document defines the interface for WIFI-HIF.

WIFI-HIF is the control and data interface for APMCU to send commands and data to the WIFI radio subsystem and get the command response and data from WIFI radio subsystem. The WIFI radio subsystem includes the N9 processor, PSE, WIFI MAC, and WIFI radio.

The WIFI-HIF design is based on SDIO IP and supports SDIO commands, including the command CMD52, CMD53, and the command response CMD3, CMD5, CMD7, and CMD11. The SDIO IO is replaced by AHB bus. WIFI-HIF is the AHB slave.

5.16.1.1 WIFI-HIF Interface

- APMCU direct access WIFI-HIF through AHB slave interface
- APMCU programs APDMA to access WIFI-HIF.
- APMCU reads/writes WIFI-HIF by SDIO-like command.
- In WIFI-HIF, the APMCU HOST Driver Domain registers are described in chapter 5.16.4, and can be accessed by APMCU via AHB bus (0x180F_0000 in CONNSYS) using SDIO-like command interfaces which are described and exampled in chapter 5.16.2.

In WIFI-HIF, the N9 Firmware Domain registers are described in chapter 5.16.5 and can be accessed by N9 via AHB1 bus (0x5003_0000 in CONNSYS).

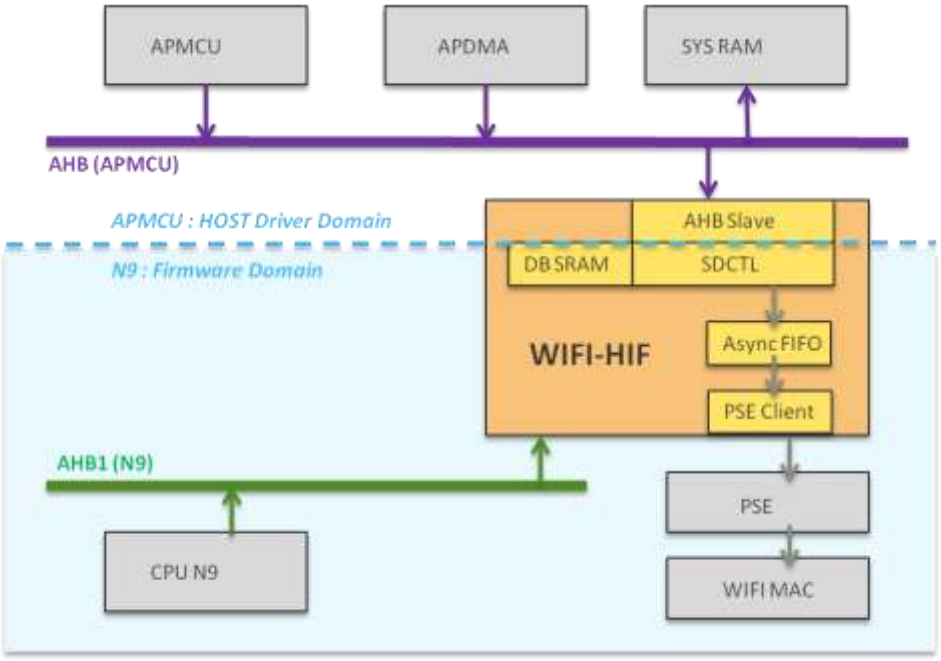


Figure 5-34. CONNSYS WIFI-HIF architecture

5.16.1.2 WIFI-HIF Features

- WIFI-IF is used for WIFI radio subsystem.
- WLAN TX packet de-aggregation and WLAN RX packet aggregation
- WLAN WHISR/RX enhanced read mode
- CIS information for WIFI-HIF
- func0, func1, and func2 capabilities exist in WIFI-HIF, but func2 is not used in Indium.

5.16.2 WIFI-HIF Command

WIFI-HIF has defined several commands for MCU-APMCU. These commands follow SDIO specification and have their own addresses. MCU-APMCU also needs to follow the predefined rules to access these commands. By these SDIO-like commands, MCU-APMCU can not only check WIFI-HIF status but also read/write WIFI-HIF driver domain CR.

5.16.2.1 WIFI-HIF Command List

WIFI-HIF has defined several commands, which are similar to SDIO commands. The following table lists all commands and the corresponding addresses. Each command has its own command information. Set up the command information first before sending the command. All the commands and their information are described in chapter 5.16.2.4.

Table 5-11. WIFI-HIF command list

Module name: WIF_HIF Base address: (+180f0000h)

Address	Name	Width	Register Function
50200000	SDIO_CMD_BASE	32	CMD52/CMD53 command information
50200004	CMD52	32	CMD52 port (read/write SDIO on funco)
50200008	CMD3	32	CMD3 port (Read only: R6 response)
5020000C	CMD5	32	CMD5 port (Read only: R4 response)
50200010	CMD7	32	CMD7 port (Read only: R1 response)
50200014	CMD11	32	CMD11 port (Read only: R1 response)
50201000	CMD53	32	CMD53 port (General read/write SDIO)

5.16.2.2 WIFI-HIF Command and Register Scope

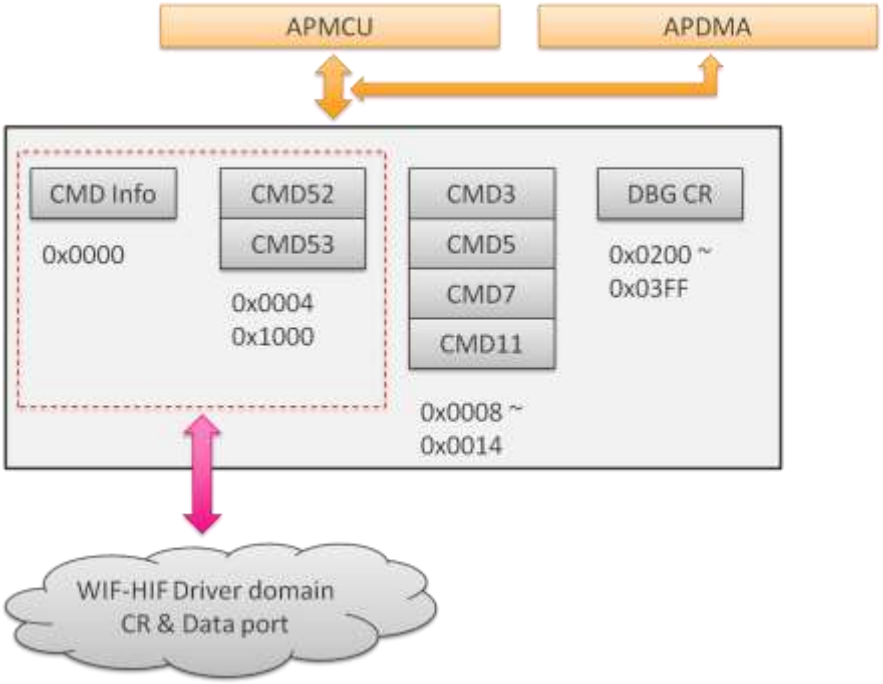


Figure 5-35. WIFI-HIF command memory space

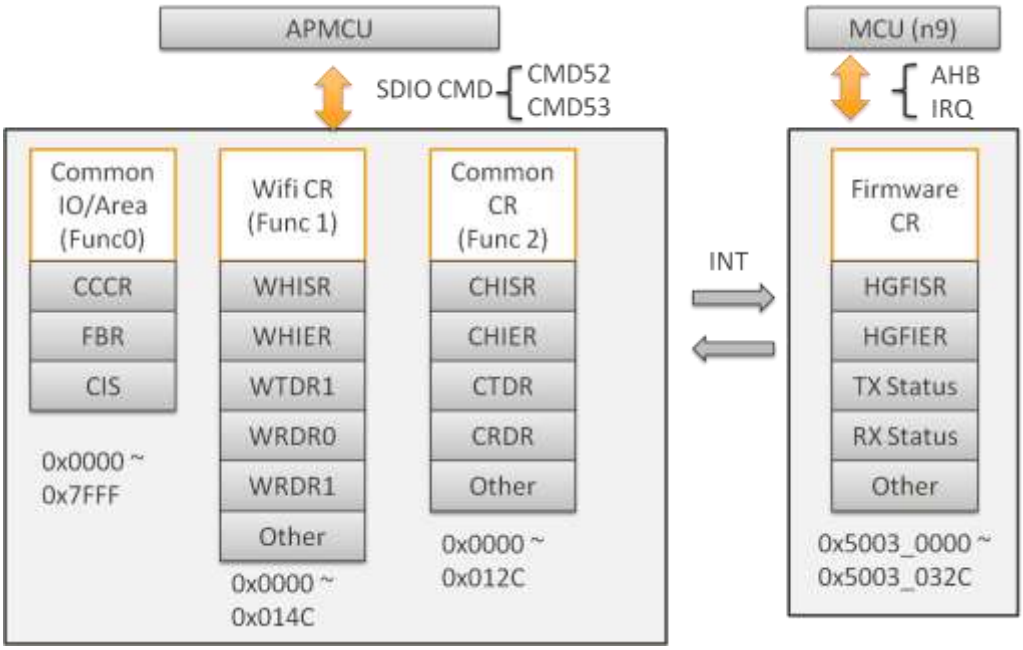


Figure 5-36. WIFI-HIF register space

5.16.2.3 WIFI-HIF Commands Description

WIFI-HIF defines several SDIO-like commands. Different commands have different command information and their own responses. The information is compatible to SDIO specification. This chapter gives details of each command.

5.16.2.4 CMD5 (IO_SEND_OP_COND Command)

Command Information

The tables below show the CMD5 command information. The function of CMD5 for SDIO cards is used to inquire about the voltage range needed by the I/O card. The normal response to CMD5 is R4. See Table 5-16 for the 32-bit WIFI-HIF R4 response.

Table 5-12. CMD5 command information format

0	0	0	0	0	0	0	S18R	IO/OCR
31	30	29	28	27	26	25	24	23-0

Table 5-13. CMD5 command information

Bits	Identifier	Description
31:25	Reserved	
24	S18R	Switch to 1.8V request
23:0	I/O OCR	Operation conditions register for MCU. The supported minimum and maximum VDD values. The layout of the OCR is shown in Table 5-14.

Table 5-14. OCR values for CMD5

I/O OCR bit position	VDD voltage window (Volts)
0-7	Reserved
8	2.0-2.1
9	2.1-2.2
10	2.2-2.3
11	2.3-2.4
12	2.4-2.5
13	2.5-2.6
14	2.6-2.7
15	2.7-2.8
16	2.8-2.9
17	2.9-3.0
18	3.0-3.1
19	3.1-3.2
20	3.2-3.3
21	3.3-3.4
22	3.4-3.5
23	3.5-3.6

Command Response (R4)

The CMD5 operation responds with WIFI-HIF unique response, R4.

Table 5-15. Response R4 format

C	Number of I/O functions	Memory present	Stuff bits	S18A	I/O OCR
31	30-28	27	26-25	24	23-0

Table 5-16. CMD5 response R4

Bits	Identifier	Description
31	Card ready	Set SDIO to 1 if the card is ready to operate after initialization. MCU can check I/O availability by this bit.

Bits	Identifier	Description
30:28	Number of I/O functions	Indicates the total number of I/O functions supported by this card. Note that the common area present on all I/O cards at Function 0 is not included in this count. The I/O function shall be implemented sequentially beginning at Function 1.
27	Memory present	It is set to 0. WIFI-HIF is I/O only.
26:25	Stuff bits	Reserved to 0
24	S18A	Switching to 1.8V is accepted.
23:0	I/O OCR	Operation Conditions Register for SDIO card. The default I/O OCR in the SDIO card is 0x7F_0000.

Note: CMD5 is used to inquire voltage support range in SDIO card. MCU needs to set the OCR match I/O OCR in the SDIO card. When the OCR value is matched between MCU and SDIO card, CMD5 will respond card ready bit to 1.

5.16.2.4.1 CMD3 (SEND_RELATIVE_ADDRESS)

Command Information

The tables below show the CMD3 command information. The function of CMD3 for SDIO cards is used for asking the SDIO card to publish a new relative address (RCA). The normal response to CMD3 is R6. See Table 5-20 for the 32-bit WIFI-HIF R6 response.

Table 5-17. CMD3 Command information format

Reserved to 0
31-0

Table 5-18. CMD3 command information

Bits	Identifier	Description
31:0	Reserved	The CMD3 command information is reserved to 0; MCU needs to set up command information to 0 before sending CMD3.

Command Response (R6)

Table 5-19. Response R6 format

RCA																														
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

Table 5-20. CMD3 response R

Bits	Identifier	Description
31:	RCA	The CMD3 operation responds with WIFI-HIF unique response, R6. Each time MCU sends CMD3, SDIO card will increase its own RCA by 1 and respond with new RCA value.

5.16.2.4.2 CMD7 (SELECT/DESELECT_CARD)

Command Information

The tables below show the CMD7 command information. The function of CMD7 for SDIO cards is used for selecting card by its own relative address (RCA) and gets deselected by any other address. The normal response to CMD7 is R1. See Table 5-24 for the 32-bit WIFI-HIF R1 response.

Table 5-21. CMD7 command information format

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	RCA
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15-0

Table 5-22. CMD7 command information

Bits	Identifier	Description
31:16	Reserved	
15:0	RCA	In WIFI-HIF, MCU can check the current RCA by CMD3 and send new RCA by CMD7.

Command Response (R1)

The CMD7/CMD11 operation responds with WIFI-HIF unique response, R1. In WIFI-HIF, R1 responds specific card status in the SDIO card (see Table 5-24).

Table 5-23. Response R1 format

Card Status																														
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

Table 5-24. CMD7/CMD11 response R1 (card status)

Bits	Identifier	Value	Description
31	OUT_OF_RANGE	'0'=no error '1'= error	1. Command information of address field in CMD52/CMD53 is out of range (bigger than 0x0FFFF). 2. Funcion1/Function2 block size specified in CCCR is out of range (bigger than 0x800).
30	Reserved	0	
29	Reserved	0	

Bits	Identifier	Value	Description
28	Reserved	0	
27	Reserved	0	
26	Reserved	0	
25	Reserved	0	
24	Reserved	0	
23	COM_CRC_ERROR	'0'=no error '1'= error	CRC error of the previous command. In WIFI-HIF, AHB bus does not have CRC check, but it checks CMD53 read/write operation error.
22	ILLEGAL_COMMAND	'0'=no error '1'= error	Command is not supported in SDIO card.
21	Reserved	0	
20	Reserved	0	
19	ERROR	'0'=no error '1'= error	A general or unknown error occurs during the operation.
18	Reserved	0	
17	Reserved	0	
16	Reserved	0	
15	Reserved	0	
14	Reserved	0	
13	Reserved	0	
12:9	CURRENT_STATE	0xF	Reserved to 0xF in I/O mode
8:0	Reserved	0	

5.16.2.4.3 CMD11 (VOLTAGE_SWITCH)

Command Information

Table 5-26 shows the CMD11 command information. The function of CMD11 for SDIO cards is used for requesting SDIO card switch to 1.8V bus signal level. The normal response to CMD11 is R1. The 32-bit GEN3-SDIOWIFI-HIF R1 response is the same as CMD7 (see Table 5-24).

Table 5-25. CMD11 command format

Reserved to 0
31-0

Table 5-26. CMD11 command information

Bits	Identifier	Description
31:0	Reserved	The CMD11 command information is reserved to 0. MCU needs to set up command information to 0 before sending CMD11.

Command Response (R1)

The CMD11 operation responds with WIFI-HIF unique response, R1. In WIFI-HIF, R1 responds specific card status in the SDIO card. R1 response is the same as CMD7 response.

5.16.2.4.4 CMD52 (IO_RW_DIRECT)

Command Information

Table 5-28 provides the CMD52 command information. CMD52 is the simplest mean to access a single register within the total 128K of register space in any I/O function, including the common I/O area (CIA).

This command can only be used to read/write one byte data. A common use is to initialize CCCR and FBR registers in function0 or monitor CIS status of SDIO card. CMD52 does not have specific response; it is used for reading/writing registers in SDIO card.

Table 5-27. CMD52 command information format

R/W flag	Function number			RAW flag	Stuff	Register address	Stuff	Write data
31	30	29	28	27	26	25-9	8	7-0

Table 5-28. CMD52 command information

Bits	Identifier	Description
31	R/W flag	This bit determines the direction of the I/O operation. If this bit is 0, this command shall read data from the SDIO card at the address specified by the gunction number and register address to the host. If this bit is set to 1, the command shall write the bytes in the write data field to the I/O location addressed by the function number and register address.
30:28	Function number	The number of function within the I/O card you wish to read or write. Note that function 0 selects the common I/O area (CIA).
27	RAW flag	Reserved to 0. In WIFI-HIF, CMD52 does not support read-after-write function and does not respond with R5.
26	Stuff	Reserved to 0
25:9	Register address	This is the address of the byte of data inside of the selected function to read or write.

Bits	Identifier	Description
		There are 17 bits of address available so the register is located within the first 128K (131,072) addresses of that function.
8	Stuff	Reserved to 0
7:0	Write data or stuff bits	For a direct write command (R/W=1), this is the byte that is written to the selected address. For a direct read (R/W=0), this field is not used and should be set to 0.

5.16.2.4.5 CMD53 (IO_RW_EXTENDED)

This command allows the reading or writing of a large number of I/O registers with a single command. Since this is a data transfer command, it provides the highest possible transfer rate.

A common use is to read/write TX/RX data register in function1 and function2. It is also commonly used in reading/writing 32-bit register in function1 and function2. CMD53 does not have a specific response.

Command Information

Table 5-29. CMD53 command information format

R/W flag	Function number			Block mode	OP code	Register address	Byte/Block count
31	30	29	28	27	26	25-9	8-0

Table 5-30. CMD53 command information

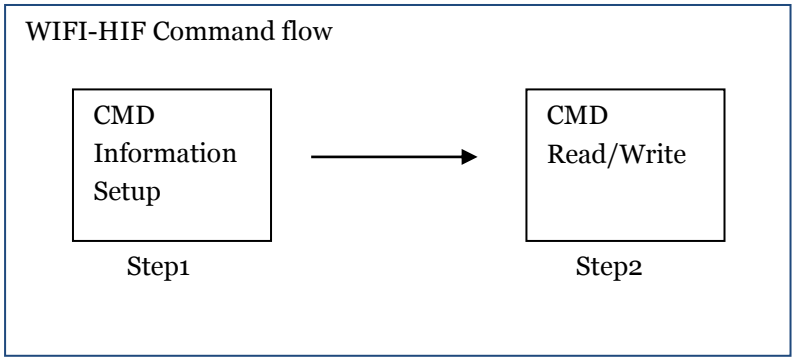
Bits	Identifier	Description
31	R/W flag	This bit determines the direction of the I/O operation. If this bit is 0, this command shall read data from the SDIO card at the address specified by the function number and register address to the host. If this bit is set to 1, the command shall write the bytes in the write data field to the I/O location addressed by the function number and register address.
30:28	Function number	The number of the function within the I/O card you wish to read or write. Note that function 0x00 selects the common I/O area (CIA).
27	Block mode	Block mode indicates that the read or write operation shall be performed on a block basis, rather than the normal byte basis. If this bit is set, the byte/block count value shall contain the number of blocks to be read/written. The block size for functions 1-7 is set by writing the block size to the I/O block size register in the FBR. The block size for function 0 is set by writing to the FN0 Block Size register in the CCCR.
26	OP code	OP Code 0 is used for reading or writing multiple bytes of data to/from a single I/O register address. OP Code 1 is used for reading or writing multiple bytes of data to/from an I/O register

Bits	Identifier	Description
		address that increment by 1 after each operation. Note: WIFI-HIF only supports OP-Code=1 in function0 CIS read operation.
25:9	Register address	Start address of I/O register to read or write. Range: [0x0FFFF:0]
8:0	Byte/Block count	If the command is operating on bytes (Block Mode = 0), this field will contain the number of bytes to read or write. Value of 0x000 will cause 512 bytes to be read or written. If the command is in block mode (Block Mode=1), the block count field will specify the number of data blocks to be transferred following this command. Note: It is not valid to specify infinite block transfer (Block count=0) in WIFI-HIF.

5.16.2.5 Usage of WIFI-HIF Command

In WIFI-HIF, all commands should follow the same rules.

Step 1: Set up command information in the address (Base_address + 0x0000_0000).
Step 2: Read or write data by specific command and its own address.



5.16.2.5.1 CMD5 Command Procedure

(1) CMD5_RD (read only)

Step 1: Disable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x1
Step 2: Set up command information
*(Base_address + 0x0000_0000) = cmd5_info
Step 3: Send CMD5
cmd_rdata = *(Base_address + 0x0000_000C)
Step 4: Enable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x0

5.16.2.5.2 CMD52 Command Procedure

(1) CMD52_WR (write)

```
Step 1: Disable WIF-HIF interrupt
*( Base_address + 0x0000_0200) = 0x1
Step 2: Set up command information
*( Base_address + 0x0000_0000) = cmd52_info
Step 3: Send CMD52
*(Base_address + 0x0000_0004) = 0;
Step 4: Enable WIF-HIF interrupt
*( Base_address + 0x0000_0200) = 0x0
```

Note: CMD52 write data are specified in the command information, but there is still the need to write dummy 0 into address (Base_address + 0x0000_0004) to active CMD52.

(2) CMD52_RD (read)

```
Step 1: Disable WIF-HIF interrupt
*( Base_address + 0x0000_0200) = 0x1
Step 2: Set up command information
*( Base_address + 0x0000_0000) = cmd52_info
Step 3: Send CMD52
cmd_rdata  = *( Base_address + 0x0000_0004)
Step 4: Enable WIF-HIF interrupt
*( Base_address + 0x0000_0200) = 0x0
```

Note:
“cmd5_info” and “cmd52_info” are user specified command information.
“cmd_wdata” is user specified write data.
“cmd_rdata” is SDIO response or read data.
All the examples of WIFI-HIF command are described in the appendix.

5.16.2.5.3 CMD52 Read/Write Example

CMD52 Write CR

Example: Writing data 0x03 to the address 0x04 in function1

Table 5-31. CMD52 write CR

R/W flag	Function number			RAW flag	Stuff	Register address	Stuff	Write data
1	0	0	1	0	0	0_0000_0000_0000_0100	0	0000_0011

Step 1: Set up command information.
Setup command information into address: (Base_address + 0x0000_0000).
CMD52_INFO: 0x9000_0803

Step 2: CMD52 write
Write dummy data 0 into address: (Base_address + 0x0000_0004) to activate write operation. Note that the CMD52 write data are specified in cmd52_info.

CMD52 write operation example

Step 1: Disable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x1
Step 2: Set up command information
*(Base_address + 0x0000_0000) = CMD52_INFO
Step 3: CMD52 write
*(Base_address + 0x0000_0004) = 0;
Step 4: Enable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x0

CMD52 Read CR

Example: Read data 0x06 from the address 0x02 in the function0;

R/W flag	Function number			RAW flag	Stuff	Register address	Stuff	Write data
0	0	0	0	0	0	0_0000_0000_0000_0010	0	0000_0000

Step 1: Set up command information.
Set up command information into address: (Base_address + 0x0000_0000).
CMD52_INFO: 0x0000_0400

Step 2: CMD52 read
Read address: (Base_address + 0x0000_0004) will return SDIO register value, cmd_rdata = 0x06

CMD52 read operation example

Step 1: Disable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x1
Step 2: Set up command information
*(Base_address + 0x0000_0000) = CMD52_INFO
Step 3: CMD52 read
cmd_rdata = *(Base_address + 0x0000_0004)
Step 4: Enable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x0

5.16.2.5.4 CMD53 Read/Write CR Example

CMD53 write CR (byte mode)

Example: Writing 4 bytes of data 0x0000_0203 to the address 0x0004 in function1 and using byte mode transfer

R/W flag	Function number			Block Mode	OP Code	Register address	Byte/Block count
1	0	0	1	0	0	0_0000_0000_0000_0100	0_0000_0100

Step 1: Set up command information.
Set up command information into address: (Base_address + 0x0000_0000).
CMD53_INFO: 0x90000804

Step 2: CMD53 write
Write data into address: (Base_address + 0x0000_1000) to activate write operation.
CMD53_WDATA: 0x0000_0203

CMD53 write CR example (byte mode)

Step 1: Disable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x1
Step 2: Write command information
*(Base_address + 0x0000_0000) = CMD53_INFO
Step 3: CMD53 write
*(Base_address + 0x0000_1000) = CMD53_WDATA
Step 4: Enable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x0

CMD53 Read CR (byte mode)

Example: Reading 4 bytes of data 0x0010_1007 to the address 0x0010 in function1 and using byte mode transfer

R/W flag	Function number			Block mode	OP Code	Register address	Byte/Block count
0	0	0	1	0	0	0_0000_0000_0000_1010	0_0000_0100

Step 1: Set up command information.
Set up command information into address: (Base_address + 0x0000_0000).
CMD53_INFO: 0x10001404

Step 2: CMD53 read
Read data from address: (Base_address + 0x0000_1000) to activate read operation.
CMD53_RDATA: 0x0010_1007

CMD53 read CR example (byte mode)

Step 1: Disable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x1
Step 2: Write command information
*(Base_address + 0x0000_0000) = CMD53_INFO
Step 3: CMD53 write
CMD53_RDATA = *(Base_address + 0x0000_1000)
Step 4: Enable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x0

5.16.2.5.5 CMD53 Read/Write Data Port Example

5.16.2.5.5.1 CMD53 Read/Write Data Port (PIO Mode)

CMD53 consecutive write data port (byte mode)
Example: Writing 24 bytes data to the address 0x0034 in function1 and using byte mode transfer

R/W flag	Function number			Block mode	OP code	Register address	Byte/Block count
1	0	0	1	0	0	0_0000_0000_0011_0100	0_0001_1000

Step 1: Set up command information.
Set up command information into address: (Base_address + 0x0000_0000).
CMD53_INFO: 0x90006818

Step 2: CMD53 write
Write data into address: (Base_address + 0x0000_1000) to activate write operation.
Write data : CMD53_WDATA_0 ~ CMD53_WDATA_5

Consecutive write data port example (byte mode)

Step 1: Disable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x1
Step 2: Write command information
*(Base_address + 0x0000_0000) = CMD53_INFO
Step 3: CMD53 write
*(Base_address + 0x0000_1000) =
CMD53_WDATA_0
*(Base_address + 0x0000_1000) =
CMD53_WDATA_1
*(Base_address + 0x0000_1000) =
CMD53_WDATA_2
*(Base_address + 0x0000_1000) =
CMD53_WDATA_3
*(Base_address + 0x0000_1000) =
CMD53_WDATA_4
*(Base_address + 0x0000_1000) =
CMD53_WDATA_5
Step 4: Enable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x0

CMD53 consecutive read data port (byte mode)

Example: Reading 32 of bytes data from the address 0x0050 in function1 and using byte mode transfer

R/W flag	Function number			Block mode	OP code	Register address	Byte/Block count
0	0	0	1	0	0	0_0000_0000_0101_0000	0_0010_0010

Step 1: Set up command information.
Set up command information into address: (Base_address + 0x0000_0000).
CMD53_INFO: 0x1000A022

Step 2: CMD53 write
Read data from address: (Base_address + 0x0000_1000) to activate read operation.
CMD53_RDATA_0 ~CMD53_WDATA_7

Consecutive read data port example (byte mode)

Step 1: Disable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x1
Step 2: Write command information
*(Base_address + 0x0000_0000) = CMD53_INFO
Step 3: CMD53 read
CMD53_RDATA_0 = *(Base_address + 0x0000_1000)
CMD53_RDATA_1 = *(Base_address + 0x0000_1000)
CMD53_RDATA_2 = *(Base_address + 0x0000_1000)
CMD53_RDATA_3 = *(Base_address + 0x0000_1000)
CMD53_RDATA_4 = *(Base_address + 0x0000_1000)
CMD53_RDATA_5 = *(Base_address + 0x0000_1000)
CMD53_RDATA_6 = *(Base_address + 0x0000_1000)
CMD53_RDATA_7 = *(Base_address + 0x0000_1000)
Step 4: Enable WIF-HIF interrupt
*(Base_address + 0x0000_0200) = 0x0

5.16.2.5.5.2 CMD53 Read/Write Data Port with APDMA Single Mode (HIF_0)

CMD53 Write Data Port with APDMA (Block mode)

Example: Writing 768 bytes of data to the address 0x0034 in the function1 and using block mode transfer

(Block size = 512 bytes)

R/W flag	Function number			Block mode	OP code	Register address	Byte/Block count
1	0	0	1	1	0	0_0000_0000_0011_0100	0_0000_0010

Step 1: Prepare TX data in the sysram
Prepare 768 bytes of TX data in SYSRAM.
Prepare another 256 bytes of dummy 0 data in SYSRAM. (Pad zero into 1024 bytes.)

Step 2: Set up command information.
Set up command information into address: (Base_address + 0x0000_0000).
CMD53_INFO: 0x98006802

Step 3: APMCU program APDMA (HIF_0) to move data from SYSRAM into WIF-HIF
Set up APDMA control register and start to transfer.
Set AP_DMA_HIF_0_INT_EN = 0x1. (Enable APDMA interrupt.)
Set up AP_DMA_HIF_0_SRC_ADDR (Set up APDMA source address and memory address.)
Set AP_DMA_HIF_0_LEN = 1024. (Set up APDMA transfer length.)

Set AP_DMA_HIF_0_DST_ADDR = 0x180f_1000. (Set up WIFI_HIF CMD53 address.)
Set AP_DMA_HIF_0_CON = 0x0003_0000. (Set AXI Burst length = 4 and TX mode)
AP_DMA_HIF_0_CON[31]: 0x0 (single transfer mode)
AP_DMA_HIF_0_CON[17:16]: 0x3 (burst length setting)
AP_DMA_HIF_0_CON[2]: 0x0 (Disable general DMA slow-down.)
AP_DMA_HIF_0_CON[1]: 0x0 (Disable fixed pattern.)
AP_DMA_HIF_0_CON[0]: 0x0 (0: TX mode; 1: RX mode)
Set AP_DMA_HIF_0_EN = 0x1. (Start APDMA transfer.)

Note: DMA total transfer length (1024 bytes) should be the same as the setting in the command information (512 bytes*2 blocks).

CMD53 read data port with APDMA (block mode)
Example: Writing 768 bytes of data to the address 0x0050 in function1 and using block mode transfer

(Block size = 256 bytes)

R/W flag	Function number			Block mode	OP code	Register address	Byte/Block count
0	0	0	1	1	0	0_0000_0000_0101_0000	0_0000_0011

Step 1: Disable WIF-HIF interrupt
*(0x180F0200) = 0x1

Step 2: Set up command information.
Set up command information into address: (Base_address + 0x0000_0000).
CMD53_INFO: 0x1800A003

Step 3: APMCU programs APDMA to move data from WIFI–HIF into SYSRAM.
Set up APDMA control register and start to transfer.
Set AP_DMA_HIF_0_INT_EN = 0x1. (Enable APDMA interrupt.)
Set AP_DMA_HIF_0_SRC_ADDR = 0x180f_1000. (Set up WIFI_HIF CMD53 address.)
Set AP_DMA_HIF_0_LEN = 768. (Set up APDMA transfer length.)
Set AP_DMA_HIF_0_DST_ADDR = 0x180f_1000. (Set up APDMA destination address and memory address.)
Set AP_DMA_HIF_0_CON = 0x0003_0001. (Set AXI burst length = 4 and RX mode.)
AP_DMA_HIF_0_CON[31]: 0x0 (single transfer mode)
AP_DMA_HIF_0_CON[17:16]: 0x3 (burst length setting)
AP_DMA_HIF_0_CON[2]: 0x0 (Disable general DMA slow-down.)
AP_DMA_HIF_0_CON[1]: 0x0 (Disable fixed pattern.)
AP_DMA_HIF_0_CON[0]: 0x1 (0: TX mode; 1: RX mode)
Set AP_DMA_HIF_0_EN = 0x1. (Start APDMA transfer.)
Step 4: Poll AP_DMA_HIF_0_EN until finishing.

Step 5: Enable WIF-HIF interrupt.
*(0x180F0200) = 0x0

Note: DMA total transfer length (192*4 bytes) should be the same as the setting in the command information (256 bytes*3 blocks)

5.16.2.5.5.3 CMD53 Read/Write Data Port with APDMA Burst Mode (HIF_0)

CMD53 write data port with APDMA burst mode (block mode)
Example: Writing 768 bytes data to the address 0x0034 in function1 and using block mode transfer

(Block size = 512 bytes)

R/W flag	Function number			Block mode	OP code	Register address	Byte/Block count
1	0	0	1	1	0	0_0000_0000_0011_0100	0_0000_0010

Step 1: Prepare TX data in SYSRAM.
Prepare 768 bytes of TX data in SYSRAM.
Prepare another 256 bytes dummy 0 data in SYSRAM. (Pad zero into 1024 bytes.)

Step 2: Set up command information.
Set up command information into address: (Base_address + 0x0000_0000).
CMD53_INFO: 0x98006802

Step 3: APMCU programs APDMA (HIF_0) to move data from SYSRAM into WIF-HIF.
Set up APDMA control register and start to transfer.
Set AP_DMA_HIF_0_INT_EN = 0x1. (Enable APDMA interrupt.)
Set up AP_DMA_HIF_0_SRC_ADDR (Set up APDMA source address and memory address.)
Set AP_DMA_HIF_0_LEN = 1024. (Set up APDMA transfer length.)
Set AP_DMA_HIF_0_DST_ADDR = 0x180f_1000. (Set up WIFI_HIF CMD53 address.)
Set AP_DMA_HIF_0_CON = 0x8003_0000 (Set AXI burst length = 4 and TX mode.)
AP_DMA_HIF_0_CON[31]: 0x1 (burst transfer mode)
AP_DMA_HIF_0_CON[17:16]: 0x3 (burst length setting)
AP_DMA_HIF_0_CON[2]: 0x0 (Disable general DMA slow-down.)
AP_DMA_HIF_0_CON[1]: 0x0 (Disable fixed pattern.)
AP_DMA_HIF_0_CON[0]: 0x0 (0: TX mode; 1: RX mode)
Set AP_DMA_HIF_0_EN = 0x1. (Start APDMA transfer.)

Note: DMA total transfer length (1024 bytes) should be the same as the setting in the command information (512 bytes*2 blocks).

CMD53 read data port with APDMA burst mode (block mode)

Example: Writing 768 bytes data to the address 0x0050 in function1 and using block mode transfer.

(Block size = 256 bytes)

R/W flag	Function number			Block mode	OP code	Register address	Byte/Block count
0	0	0	1	1	0	0_0000_0000_0101_0000	0_0000_0011

Step 1: Disable WIF-HIF interrupt
*(0x180F0200) = 0x1

Step 2: Set up command information.
Set up command information into address: (Base_address + 0x0000_0000).
CMD53_INFO: 0x1800A003

Step 3: APMCU programs APDMA to move data from WIFI–HIF into SYSRAM
Set up APDMA control register and start to transfer.
Set AP_DMA_HIF_0_INT_EN = 0x1. (Enable APDMA interrupt.)
Set AP_DMA_HIF_0_SRC_ADDR = 0x180f_1000. (Set up WIFI_HIF CMD53 address.)
Set AP_DMA_HIF_0_LEN = 768. (Set up APDMA transfer length.)
Set AP_DMA_HIF_0_DST_ADDR = 0x180f_1000. (Set up APDMA destination address and memory address.)
Set AP_DMA_HIF_0_CON = 0x8003_0001. (Set AXI Burst length = 4 and RX mode.)
AP_DMA_HIF_0_CON[31]: 0x1 (single transfer mode)
AP_DMA_HIF_0_CON[17:16]: 0x3 (burst length setting)
AP_DMA_HIF_0_CON[2]: 0x0 (Disable general DMA slow-down.)
AP_DMA_HIF_0_CON[1]: 0x0 (Disable fixed pattern.)
AP_DMA_HIF_0_CON[0]: 0x1 (0: TX mode; 1: RX mode)
Set AP_DMA_HIF_0_EN = 0x1. (Start APDMA transfer.)

Step 4: Poll AP_DMA_HIF_0_EN until finishing.

Step 5: Enable WIF-HIF interrupt
*(0x180F0200) = 0x0

Note: DMA total transfer length (192*4 bytes) should be the same as the setting in the command information (256 bytes*3 blocks).

5.16.2.6 WIFI-HIF Command Golden Rules

Rule 1: All commands should set up CMD information then read/write command.

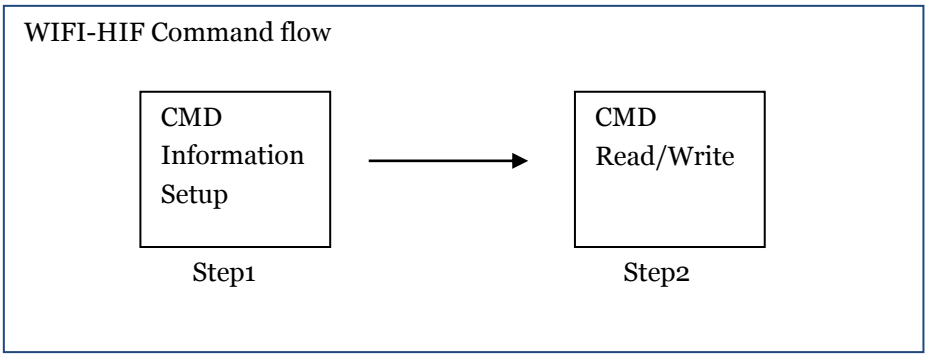


Figure 5-37. Command setup flow

Rule 2: Complete CMD53 read/write before starting the next transaction. This CMD transaction should be protected by disabling WIFI-HIF interrupt.

Case 1: CMD53 byte mode (Block_Mode=0)

1. Byte count should be 4-byte alignment (e.g. 0x4, 0x8, 0xC...).
2. Maximum transfer count: 128DW
3. CMD53 consecutive read/write DW count should be the same as the setting in the command information.

Example: Setting byte count = 0x18
Step 1: Disable WIFI-HIF interrupt.
*(0x180F0200) = 0x0001
Step 2: Write command information.
*(Base_address + 0x0000_0000) = CMD53_INFO
Step 3: CMD53 write
*(Base_address + 0x0000_1000) = CMD53_WDATA_0
*(Base_address + 0x0000_1000) = CMD53_WDATA_1
*(Base_address + 0x0000_1000) = CMD53_WDATA_2
*(Base_address + 0x0000_1000) = CMD53_WDATA_3
*(Base_address + 0x0000_1000) = CMD53_WDATA_4
*(Base_address + 0x0000_1000) = CMD53_WDATA_5
Step 4: Enable WIFI-HIF IRQ.
*(0x180F0200) = 0x0000

Case 2: CMD53 block mode (Block_Mode=1)

1. Block size setting is specified in Function0 register. Block size should be 4-byte alignment. (Default: 512 bytes, 0x200)
2. Block count setting should be more than 1.
3. Total transfer size: Block_counter*Block_size

4. CMD53 consecutive read/write DW count should be the same as the setting in the command information.

Example: Block_size: 512 bytes, block_count: 2
Step 1: Disable WIFI-HIF interrupt.
*(Base_address + 0x0000_0200) = 0x0001
Step 2: Write command information.
*(Base_address + 0x0000_0000) = CMD53_INFO
Step 3: CMD53 write
*(Base_address + 0x0000_1000) = CMD53_WDATA_0
*(Base_address + 0x0000_1000) = CMD53_WDATA_1
*(Base_address + 0x0000_1000) = CMD53_WDATA_2
…
*(Base_address + 0x0000_1000) = CMD53_WDATA_255
Step 4: Enable WIFI-HIF interrupt.
*(Base_address + 0x0000_0200) = 0x0000

Case 3: Using APDMA to transfer CMD53 data in block mode (Block_Mode=1)
1. Disable WIFI-HIF interrupt.
2. Set up CMD53 information.
3. Configure APDMA and start to transfer.
4. Poll APDMA Finish.
5. Enable WIFI-HIF interrupt.

5.16.3 WIFI-HIF TX/RX Protocol

5.16.3.1 WIFI-HIF TX Protocol

5.16.3.1.1 WIFI-HIF TX Packet Format

When there is a packet to be transmitted, the software driver should prepare the packet follow WIFI-MAC data format. The following figure is the brief description of WIFI TX descriptor. WIFI-HIF controller only parses the TX byte count information from the data packet.

The reserved part can be left for other SW/HW to parse or padded to 0 if unused.

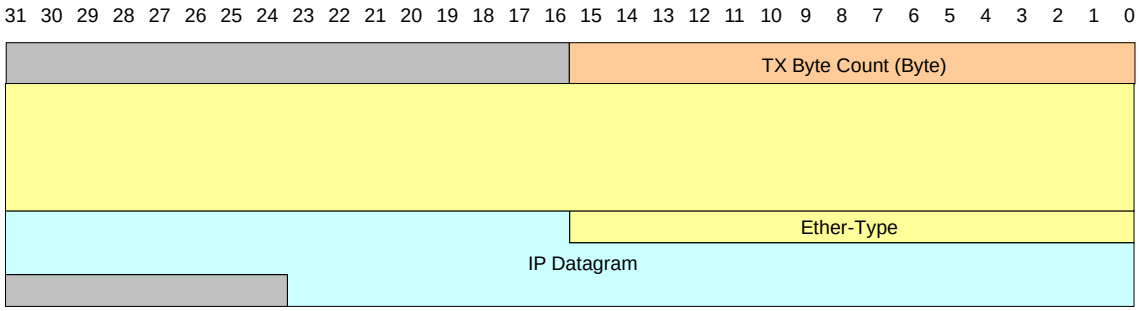


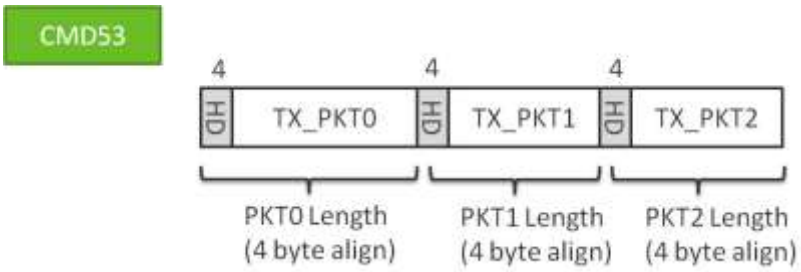
Figure 5-38. Brief WIFI-TX data format

DW	Bit	Description
0	15:0	TX byte count (unit: byte) This is the total byte count of the transmitted data structure (starting from TX byte count field to the ending byte of the TX buffer).

5.16.3.1.2 WIFI-HIF TX Aggregation

Software driver can also write multiple TX packets into data buffer, where the packets are concatenated in 4byte alignment. The access sequences are shown as below. Note that each packet should not be cut in transaction boundary, hence, if software driver needs to send multiple packets to WIF-HIF controller, it could use one command to transmit all these packets or it could use multiple commands to transmit these packets with that each packet should not cross transaction boundary. To achieve this, padding will be needed if the (concatenated) packet length cannot fit into a complete WIF-HIF data transition.

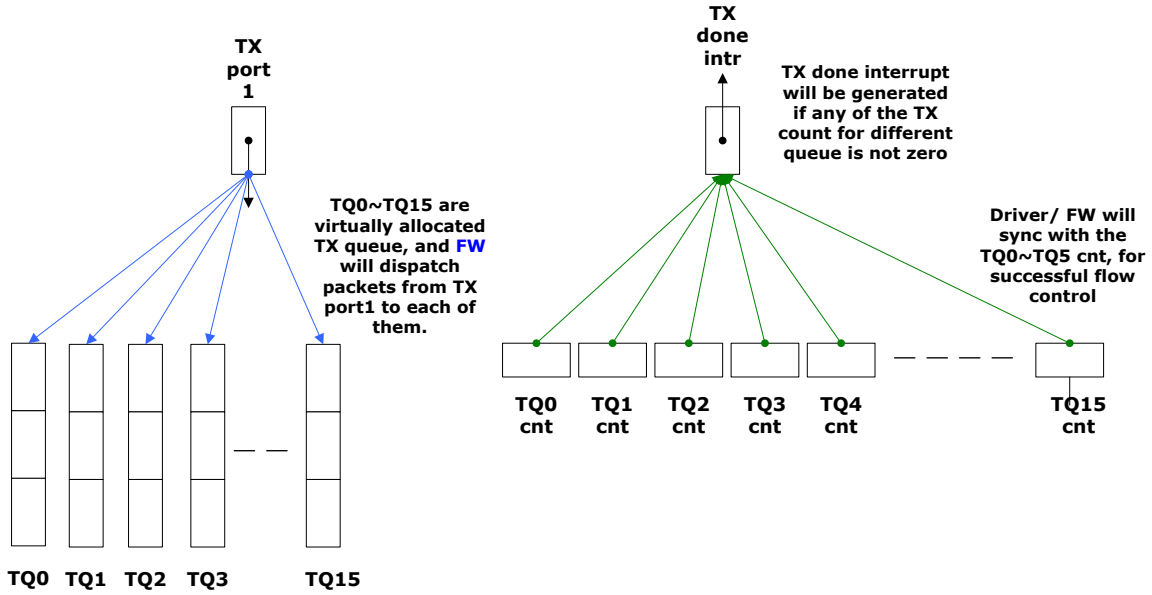
Note: HD means header. It has byte count information.



5.16.3.1.3 WIFI-HIF TX Buffer Control

There is one TX port for APMCU host driver to access TX queues through WIF-HIF controller. Since N9 has limited buffer size to receive data, it needs TX buffer control between APMCU host driver and N9 firmware.

For TX buffer control, it is done by the synchronization of the available TX packet buffer between APMCU host driver and N9 firmware. The TX packet buffer is negotiated between APMCU and N9 firmware on initialization. Driver should decrease the available packet count when sending packet to N9 firmware and stop senting the packet when available packet count reaches zero. When the packet buffer has been processed, N9 firmware will release buffer and send back TQCNT information to APMCU driver. APMCU host driver will get an interrupt and be able to read the released TX packet count via WIF-HIF controller.



N9 can allocate packet buffer into different TX queue to enhance TX buffer control by per TX queue management. Here is a example of TX queue mapping table. The mapping table can be different for each project depending on the system definition.

Table 5-32. TXQCNT mapping

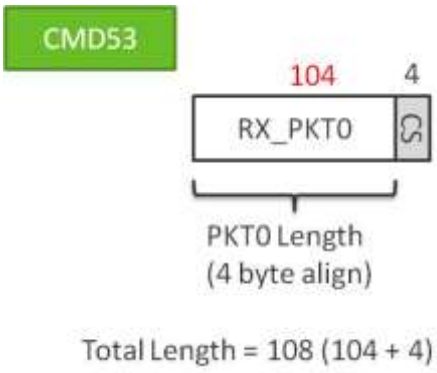
Mapping TXQ	Name	Mapping TXQ	Name
TXQ0_CNT	TXCAC0	TXQ1_CNT	TXCAC1
TXQ2_CNT	TXCAC2	TXQ3_CNT	TXCAC3
TXQ4_CNT	TXCAC4	TXQ5_CNT	TXCAC5
TXQ6_CNT	TXCAC6	TXQ7_CNT	TXCBMC

Mapping TXQ	Name	Mapping TXQ	Name
TXQ8_CNT	TXCBCN	TXQ9_CNT	TXCAC10
TXQ10_CNT	TXCAC11	TXQ11_CNT	TXCAC12
TXQ12_CNT	TXCAC13	TXQ13_CNT	TXCAC14
TXQ14_CNT	TXCFFA	TXQ15_CNT	TXCCPU

5.16.3.2 WIFI-HIF RX Protocol

5.16.3.2.1 WIFI-HIF RX Packet with Only One Packet

When APMCU host receives an RX_Done interrupt, it can read RX CR “WRPLR” to get packet length information and read RX data port “WRDR0” or “WRDR1” to get one packet. APMCU host should do 4-byte alignment first and read the extra four bytes to get extra RX_Infomation (checksum offload).



5.16.3.2.2 WIFI-HIF RX Packet with Multiple Packets (RX Aggregation)

If APMCU host wants to read out multiple RX packets at the same time, it should use interrupt enhance mode or RX enhance mode to get multiple RX length information.

5.16.3.2.2.1 Interrupt Enhance Data

Interrupt enhance mode is a protocol that APMCU host driver issues CMD53 to read out both WHISR (interrupt status) and other information at the same time. It should read out 112 bytes in one CMD53 read. All information includes WHISR, WTQCR0-7, RX0 number, RX1 number, RX0 length, RX1 length and mailbox information.

The following table lists the interrupt enhance information.

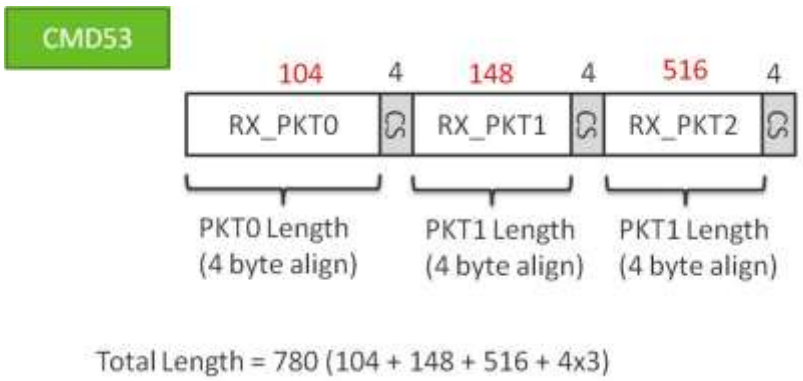
Table 5-33. Interrupt enhance data

DW	Bit	Description
0	31:0	WHISR It carries the interrupts status, which is the same in register WHISR.
1-8	31:0	WTQCR0-7 It carries the TX count status from N9 MCU.
9	31:0	Bit[31:16] : RX1 NUM It indicates the number of packets in RX port 1. RX0 NUM Bit[15:0] It indicates the number of packets in RX port 0.
10~17	15:0 31:16	RX0 LEN0~15 If RX0 NUM is not equal to 0, it indicates the per-packet length information queuing in RX0 port.
18~25	15:0 31:16	RX1 LEN0~15 If RX1 NUM is not equal to 0, it indicates the per-packet length information queuingd in RX1 port.
26	31:0	D2HRM0R It carries the received mailbox information from FW, which is the content of D2HRM0R.
27	31:0	D2HRM1R It carries the received mailbox information from FW, which is the content of D2HRM1R.

5.16.3.2.2.2 RX Aggregation by Interrupt Enhance Mode

When APMCU host receives multiple RX length information by interrupt enhance mode, WIFI driver can read out multiple packets from RX data port. For total packet size calculation, the driver should do 4-byte alignment on each packet length and add extra 4-byte length for checksum offload information. It calculates the sum of packet lengths and checksum to get the total byte count then reads out all RX packets on RX data port “WRDR0” or “WRDR1”.

The following figure is an example of 3 packet RX aggregation.



5.16.3.2.2.3 RX Enhance Mode Data Format

RX Enhance Mode is a protocol that APMCU host driver issues CMD53 to read out both RX data and interrupt enhance information at the same time. The following figure shows RX Enhance data format. It includes aggregated RX data packet, padding zero and Interrupt enhance information.

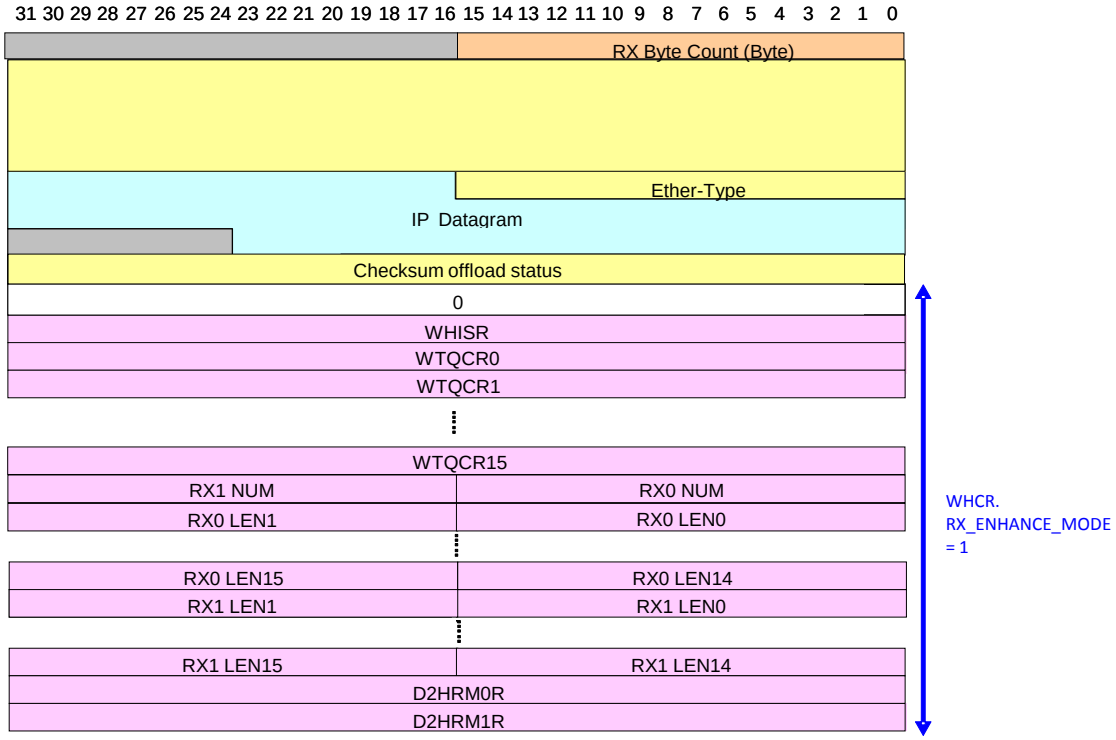
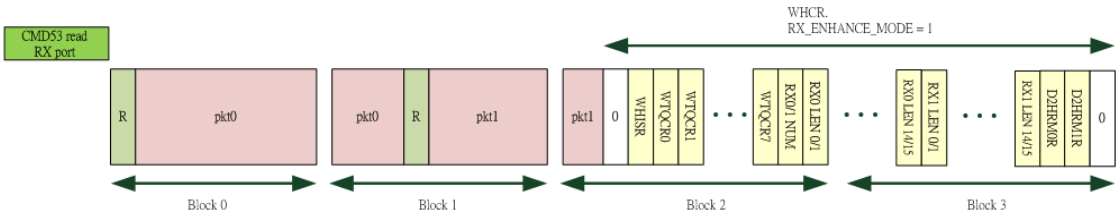


Figure 5-39. RX enhance data format

5.16.3.2.2.4 RX Aggregation by RX Enhance Mode

WIF-HIF controller supports extra function to read out RX packet and interrupt enhances information at the same time. This is the “RX enhance mode”.

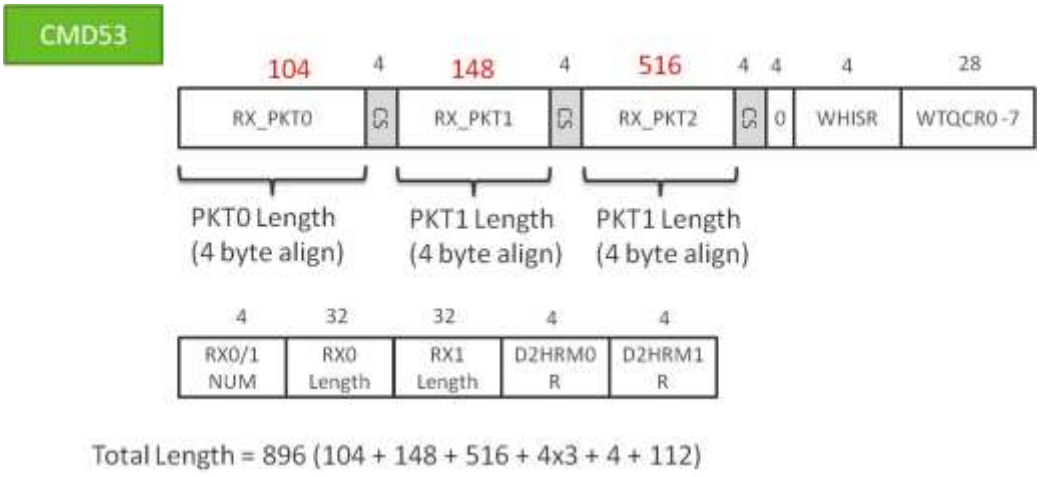
“RX enhance mode” can read out multiple RX data packets and “WHISR/TQ_CNT/RX0, RX1 number/RX0 length/RX1 length/mailbox” information.



Hardware limitation

When setting WHCR. RX_ENHANCE_MODE = 1, the host must read out both RX packets and interrupt enhance information in one CMD53 read transaction.

The following figure is an example of 3 packet RX aggregation with RX enhance mode.



5.16.4 WIFI-HIF Register Definitions

5.16.4.1 WIFI-HIF CIS

Three steps to set up SDIO CIS: First, use default value. Second, write it to SDIO controller by an external engine. Last, export an expand bus to the external to connect to the desired value. For the

first step, reference to kick-off CIS configuration table for the value. For the second and third step, the CIS value will be provided by SW and IP integrator.

The CIS value of CIS0 starts from 0x1000; CIS value for CIS1 starts from 0x2000 and CIS value for CIS2 starts from 0x3000.

5.16.4.1.1 Function 0 CIS Information

Function 0 CIS starts from address 0x1000; it is a read-only area. The host can use CMD52 or CMD53 to read CIS area. Note that if the host uses CMD53 to read from address less than 0x1000 and cross CIS area (e.g. read 128 bytes from 0x0ff8), it will not get the correct CIS value.

WIFI-HIF Function #0 CIS, size: 35 bytes, hard-coded except (0x0C~0x0D) field can be modified by EFUSE.

The following table is Function0 CIS information. CIS information is the device status which can be read out by CMD52.

Table 5-34. CIS information

Tuple	Offset	Value	Description
CISTPL_FUNCID	0x00	0x21	Tuple code
	0x01	0x02	Link to next tuple
	0x02	0x0C	Card function code
	0x03	0x00	Not used
CISTPL_FUNCE	0x04	0x22	Tuple code
	0x05	0x04	Link to next tuple
	0x06	0x00	Extended data
	0x07	0x00	The only block size (in bytes) that function 0 can support (512 bytes = 0x0200)
	0x08	0x02	
	0x09	0x5A	Transfer rate per data line is 50Mbit/sec (Bit 2:0 = 2, Bit 6:3 = b)
CISTPL_MANFID	0x0A	0x20	Tuple code
	0x0B	0x04	Link to next tuple
	0x0C	0x7A	SDIO manufacturer code (037A for MediaTek)
	0x0D	0x03	
	0x0E	0x37	Part number/revision number (7637 for MT7637)
	0x0F	0x76	
CISTPL_VERS_1	0x10	0x15	Tuple code
	0x11	0x10	Link to next tuple
	0x12	0x08	PC CARD std. Release 8.0

Tuple	Offset	Value	Description
	0x13	0x00	
	0x14	0x30	0
	0x15	0x32	2
	0x16	0x37	7
	0x17	0x39	9
	0x18	0x00	
	0x19	0x43	C
	0x1A	0x6F	O
	0x1B	0x6D	M
	0x1C	0x62	B
	0x1D	0x6F	O
	0x1E	0x00	
	0x1F	0x00	
	0x20	0x00	
	0x21	0xFF	
CISTPL_END	0x22	0xFF	End of a tuple chain

5.16.4.1.2 WIFI-HIF Function 1 CIS (WLAN)

Function 1 CIS starts from address 0x2000; it is read only area. The host can use CMD52 or CMD53 to read CIS area. Note that if the host uses CMD53 to read from address less than 0x2000 and cross CIS area (for example, read 128 bytes from 0x1ff8), it will not get the correct CIS value.

WIFI-HIF Function #1 CIS, size: 55 bytes, hard-coded except OCR field (0x14~0x17) can be modified by EFUSE.

Table 5-35. Function 1 CIS information

Tuple	Offset	Value	Description
CISTPL_FUNCID	0x00	0x21	Tuple code
	0x01	0x02	Link to next tuple
	0x02	0x0C	Card function code
	0x03	0x00	Not used
CISTPL_FUNCE	0x04	0x22	Tuple code
	0x05	0x2A	Link to next tuple
	0x06	0x01	Extended data
	0x07	0x01	Wakeup support (1: support)
	0x08	0x00	IO revision
	0x09	0x00	Product serial number
	0x0A	0x00	

Tuple	Offset	Value	Description
	0x0B	0x00	
	0x0C	0x00	
	0x0D	0x00	CSA size
	0x0E	0x00	
	0x0F	0x00	
	0x10	0x00	
	0x11	0x03	CSA property
	0x12	0x00	Maximum block size (512 bytes)
	0x13	0x02	
	0x14	0x00 **	Function OCR (3.1V0V~3.5V)
	0x15	0x00 **	
	0x16	0x7F **	
	0x17	0x00 **	
	0x18	0x64	Minimum current when operating (100 mA)
	0x19	0x96	Average current when operating (150 mA)
	0x1A	0xC8	Maximum current when operating (200 mA)
	0x1B	0x0A	Minimum current when standby (10 mA)
	0x1C	0x0F	Average current when standby (28 mA)
	0x1D	0x14	Maximum current when standby (20 mA)
	0x1E	0x32	Minimum data transfer bandwidth (50 KB)
	0x1F	0x00	
	0x20	0xDC	Optimum data transfer bandwidth (1500 KB)
	0x21	0x05	
	0x22	0x00	Required enable timeout value (No timeout value)
	0x23	0x00	
	0x24	0x96	Standard power mode average current for 3.3V
	0x25	0x00	
	0x26	0xC8	Standard power mode maximum current for 3.3V
	0x27	0x00	
	0x28	0x96	High power mode average current for 3.3V
	0x29	0x00	
	0x2A	0xC8	High power mode maximum current for 3.3V
	0x2B	0x00	
	0x2C	0x96	Low power mode average current for 3.3V
	0x2D	0x00	
	0x2E	0xC8	Low power mode maximum current for 3.3V
	0x2F	0x00	
CISTPL_MANFID	0x30	0x20	Tuple code
	0x31	0x04	Link to next tuple
	0x32	0x7A	SDIO manufacturer code 037A for MTK

Tuple	Offset	Value	Description
	0X33	0x03	
	0x34	0x02	
	0x35	0x79	0279 for func-1
CISTPL_END	0x36	0xFF	End of a tuple chain

Note : The default value is loaded from ASIC hard coded value, except that OCR can be modified by eFUSE setting.

5.16.4.2 SDIO Function0 Register

WIF-HIF interface supports SDIO function0 register. These registers are defined in the standard SDIO3.0 spec and can be accessed by CMD52 or CMD53 according to the spec.

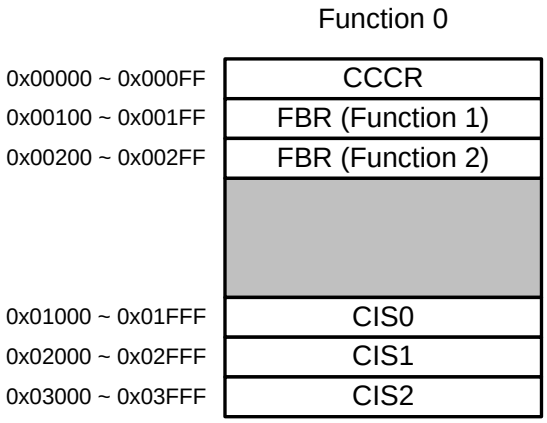


Table 5-36. Card common control registers (CCCR) – SDIO 3.0

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00	CCCR/SDIO revision	SDIO bit 3	SDIO bit 2	SDIO bit 1	SDIO bit 0	CCCR bit 3	CCCR bit 2	CCCR bit 1	CCCR bit 0
0x01	SD specification revision	RFU	RFU	RFU	RFU	SD bit 3	SD bit 2	SD bit 1	SD bit 0
0x02	I/O enable	RFU	RFU	RFU	RFU	RFU	IOE2	IOE1	RFU
0x03	I/O ready	RFU	RFU	RFU	RFU	RFU	IOR2	IOR1	RFU
0x04	INT enable	RFU	RFU	RFU	RFU	RFU	IEN2	IEN1	IENM
0x06	I/O abort	RFU	RFU	RFU	RFU	RES	AS2	AS1	AS0
0x09	Function C IS pointer	Pointer to Function CIS bit 7-0							
0x0A		Pointer to Function CIS bit 15-8							

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0B		RFU	RFU	RFU	RFU	RFU	RFU	RFU	CIS0 bit 16
0x10	FN0 block size	Function0 I/O block size bit 7-0							
0x11		Function0 I/O block size bit 15-8							
0x17-0xEF	RFU	Reserved for future use (RFU)							
0XF0-0xFF	Reserved for vendors	Area reserved for vendor unique registers							

Table 5-37. Default values of CCCR – SDIO 3.0

Field	Type	Default	Description
CCCRX	R	3 **	CCCR format version number
SDIOx	R	4 **	SDIO specification revision number
SDx	R	3 **	SD format version number
IOEx	R/W	0	Enable function
IORX	R	0	I/O function ready
IENx	R/W	0	Interrupt enable for function x.[4]
IENM	R/W	0	Interrupt enable master
ASx	W	0	Abort select abort an I/O read or write and free the SD bus, the function that is currently transferring data must be addressed.
RES	W	0	I/O CARD RESET setting the RES to 1 will cause all functions in an SDIO or combo card to perform a hard reset identical to a power-on.
FN0 Block Size	R/W	0	This 16-bit register sets up the block size for I/O block operations for Function 0 only.[5]

Table 5-38. Function basic register (FBR1/2) – SDIO 3.0

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x100	Function1 SDIO Interface code	RFU	RFU	RFU	RFU	Function1 Standard SDIO Function Interface Code			
0x109	Function1 CIS Pointer	Pointer to Function 1 CIS bit 7-0							
0x10A		Pointer to Function 1 CIS bit 15-8							

[4] To meet Interrupt Clear Timing (SDIO Spec Version 1.0, 8.1.8), software should use IENx to disable function 1 interrupt before clear MAC interrupt status (MCR offset: 0x0004 & 0x0008) and enable IENx after interrupt service routine is completed.
** Default value can use fixed default value or use an engine to set or use an external bus for them.
[5] In SD mode, if host sets function 0 block size with CMD52 larger than 2048, MT5921 will response with "OUT_OF_RANGE" in response R5.
In SD mode, if host sets function 0 block size with CMD52 to 0, MT5921 will not response error bit.
In SPI mode, if host sets function 0 block size with CMD52 larger than 2048, MT5921 will response with "PARAMETER_ERROR" in response R5.
In SPI mode, if host sets function 0 block size with CMD52 to 0, MT5921 will not response error bit.

Address	Register Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x10B		RFU	RFU	RFU	RFU	RFU	RFU	RFU	CIS1 Bit 16
0x110	FN1 Block Size	Function1 I/O Block Size bit 7-0							
0x111		Function1 I/O Block Size bit 15-8							
0x200	Function2 SDIO Interface code	RFU	RFU	RFU	RFU	Function2 Standard SDIO Function Interface Code			
0x209	Function2 CIS Pointer	Pointer to Function 2 CIS bit 7-0							
0x20A		Pointer to Function 2 CIS bit 15-8							
0x20B		RFU	RFU	RFU	RFU	RFU	RFU	RFU	CIS2 Bit 16
0x20C-0x20F		RFU	RFU	RFU	RFU	RFU	RFU	RFU	RFU
0x210	FN2 Block Size	Function1 I/O Block Size bit 7-0							
0x211		Function1 I/O Block Size bit 15-8							

Table 5-39. Default values of function basic register

Field	Type	Default	Description
Function1 SDIO Interface code	R	0 **	The standard I/O device code identifies those I/O functions, which implement the recommended standard interface as defined in a separate application specification.
Address pointer to Function CIS (FBR1)	R	0x02000	These three bytes make up a 24-bit pointer (only the lower 17 bits are used) to the start of the Card Information Structure (CIS) that is associated with each function.
Function1 Block Size	R/W	0x0200	For function1 CMD53 block size in block mode transfer.
Function2 SDIO Interface code	R	0 **	The standard I/O device code identifies those I/O functions, which implement the recommended standard interface as defined in a separate application specification.
Address pointer to Function CIS (FBR2)	R	0x03000	These three bytes make up a 24-bit pointer (only the lower 17 bits are used) to the start of the Card Information Structure (CIS) that is associated with each function.
Function2 Block Size	R/W	0x0200	For function2 CMD53 block size in block mode transfer.

5.16.5 Register Definition

Two methods to access HDCR:

1. In general, use CMD53 with byte mode and byte count = 4 to access one register. It is limited to starting in 4-byte boundary.
2. It is allowed to use CMD53 with block mode and block size = 4, block count = 1 to access one register. It is limited to starting in 4-byte boundary.
   (WIF_HIF_SDCTL_TOP_DMA)

SDIO firmware domain register is accessed by N9 MCU. N9 MCU uses firmware domain register to get SDIO information, interrupt status and other controls.
(WIF_HIF_SDCTL_TOP_FW)

See chapter 3.16 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

Register bank	Base address
WIF_HIF_SDCTL_TOP_DMA	+0h
WIF_HIF_SDCTL_TOP_FW	+50030000h
WIF_HIF_Connsys_HIF	+180f0000h

5.17 SSUSB

5.17.1 Introduction

The Super Speed Universal Serial Bus (SSUSB) IP provides a USB3.0 port with USB3.0 peripheral role and USB2.0 dual role capability. It contains a pair of U3 PHY and U3 MAC for super speed connection, and a pair of U2 PHY and U2 MAC for high-, full- and low-speed connection. The USB2.0 dual role capability allows the port to support On-The-Go (OTG) host and peripheral. When acting in USB2.0 host role, this port is controlled by the host controller (xHC) which is used to manage all devices connected through its root hub ports. When acting in peripheral role, the port is controlled by the Device (DEV) Controller.

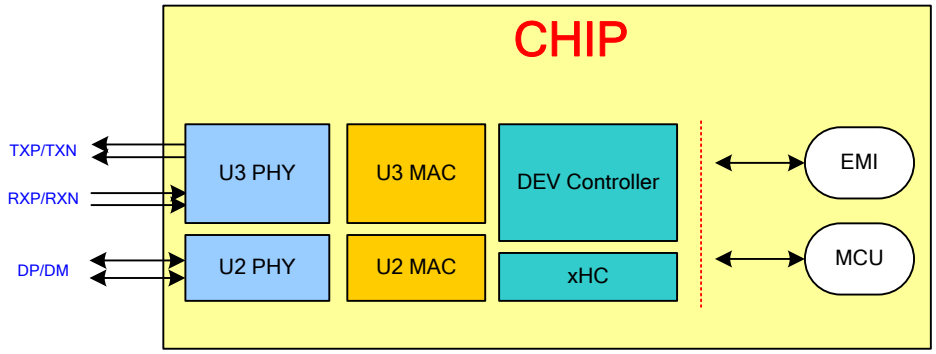


Figure 5-40. Block diagram of SSUSB IP

5.17.2 Features

- Configurable to be USB3.0 peripheral role
- Configurable to be USB2.0 host role
- Shared hardware on Dual function
- Proprietary application layer Device controller with linked list queue and scatter/gather DMA
- Extensible Host Controller Interface (xHCI) based Host controller
- Embedded USB3 PHY with 32-bit/125MHz PIPE interface
- Embedded USB2 PHY with 16-bit/30MHz UTMI interface
- High-/Full-Speed OTG host and peripheral compliant with OTG Supplement Version 2.0

5.17.3 Register Definition

See chapter 3.17 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

Register bank	Base address
ssusb_dev	+11271000h
ssusb_sifslv_u2phy_com_t20ulvt	+11290800h
ssusb_sifslv_u2phy_com_t28hpm	+11290800h
ssusb_usb2_csr	+11273400h
ssusb_sifslv_chiip	+11290300h
ssusb_sifslv_fmreg	+11290100h
ssusb_sifslv_ippc	+11280700h
ssusb_sifslv_u3phya_da_t20soc_tphy	+11290c00h
ssusb_sifslv_u3phyd_bank2_t20soc_tphy	+11290a00h
ssusb_sifslv_u3phyd_t20soc_tphy	+11290900h
ssusb_sifslv_spllc	+11290000h
ssusb_usb3_mac_csr	+11272400h
ssusb_usb3_sys_csr	+11272400h
ssusb_host	+11270000h
ssusb_xhci_u2_port	+11270420h
ssusb_sifslv_u3phya_t20soc_tphy	+11290b00h

5.17.4 Host Function Architecture

5.17.4.1 Host Features

- Supports linkage with high-, full- and low-speed device
- Lower Power Management (LPM) on U2 port
- Hardware configurable up to two USB2 ports with dedicated 480-Mbs bandwidth and shared DMA
- Control/Bulk/Interrupt/Isochronous PIPE type (currently no stream support)
- Compatible with connect to U2 Hub
- Smart scheduling algorithm
- Up to 15 devices
- Up to 32 endpoints configuration

5.17.4.2 Architecture Overview

The architecture of USB2.0 host is illustrated in. It includes one U2PHY and one MAC to handle protocol packets. All resources of endpoint and device are handled by xHCI controller. Firmware can dynamically allocate resource for different endpoints. The DMA channel of U2 port can be multiplexed in the system.

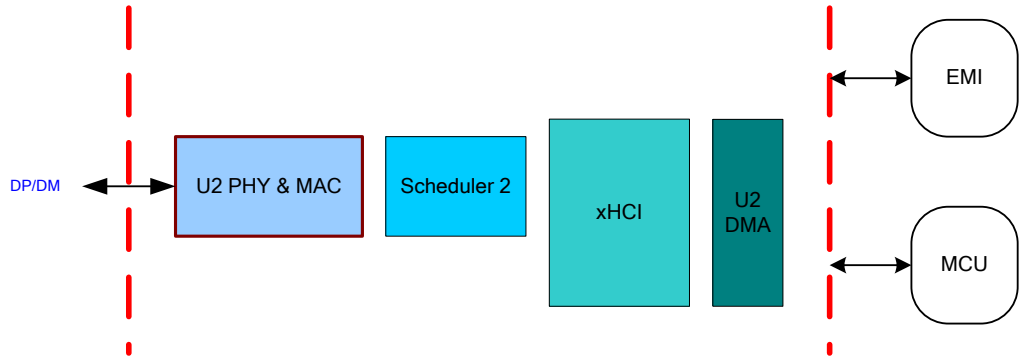


Figure 5-41. USBb2.0 host architecture

5.17.5 Device Function Architecture

5.17.5.1 Device Features

- Supports linkage with super- ,high- and full-speed host
- Embedded queue management function with scatter/gather DMA capability
- Lower Power Management (LPM) on U2 port
- U0/U1/U2/U3 state on U3 port
- 24 KB of on-chip data RAM
- Hardware configurable up to 8 OUT endpoints and 8 IN endpoints
- Hardware configurable up to 4 packet slots for each endpoint separately
- Firmware configurable FIFO size allocation for each endpoint separately
- Firmware configurable transfer type to Bulk/Interrupt/Isochronous for each endpoint

5.17.5.2 Architecture Overview

The architecture of device is illustrated in Figure 5-42. The linked-list queue of device is basically inherited from MTK Unified USB IP with similar descriptor definition.

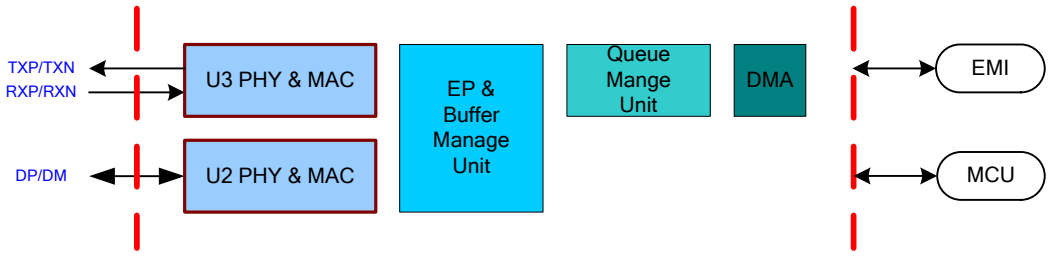


Figure 5-42. SSUSB device architecture

5.18 MSDC

5.18.1 Memory Stick and SD card Controller: MSDC0

5.18.1.1 Introduction

The MSDC (Memory Stick and SD card Controller) fully supports

- SD memory card specification version 3.0
- SDIO card specification version 3.0
- MMC/eMMC5.0

5.18.1.2 Features

- Interface with MCU by AHB bus
- 32-bit access on AHB bus
- 32-bit access for control registers
- 8-bit/16-bit/32-bit access for FIFO in PIO mode
- Built-in 128 bytes FIFO buffers for transmit and receive
- Built-in CRC circuit
- Supports Basic DMA mode, Basic Descriptor mode, and Enhanced Descriptor mode for SD/MMC
- Interrupt capabilities
- Does not support SPI mode for SD/MMC memory card
- Does not support suspend/resume for SD/MMC memory card
- Supports SD3.0 SDR104, data rate up to 208x4Mbps
- Supports SD3.0 DDR50, data rate up to 50x4x2Mbps (4-bit with clock dual edge)
- Supports eMMC HS400, data rate up to 200x8x2Mbps (8-bit with clock dual edge)
- Supports e-MMC boot-up mode
- 256 programmable serial clock rates on MS/SD/MMC bus from 100kHz to 208MHz
- Card detection capabilities

5.18.1.3 MSDC Controller Block Diagram

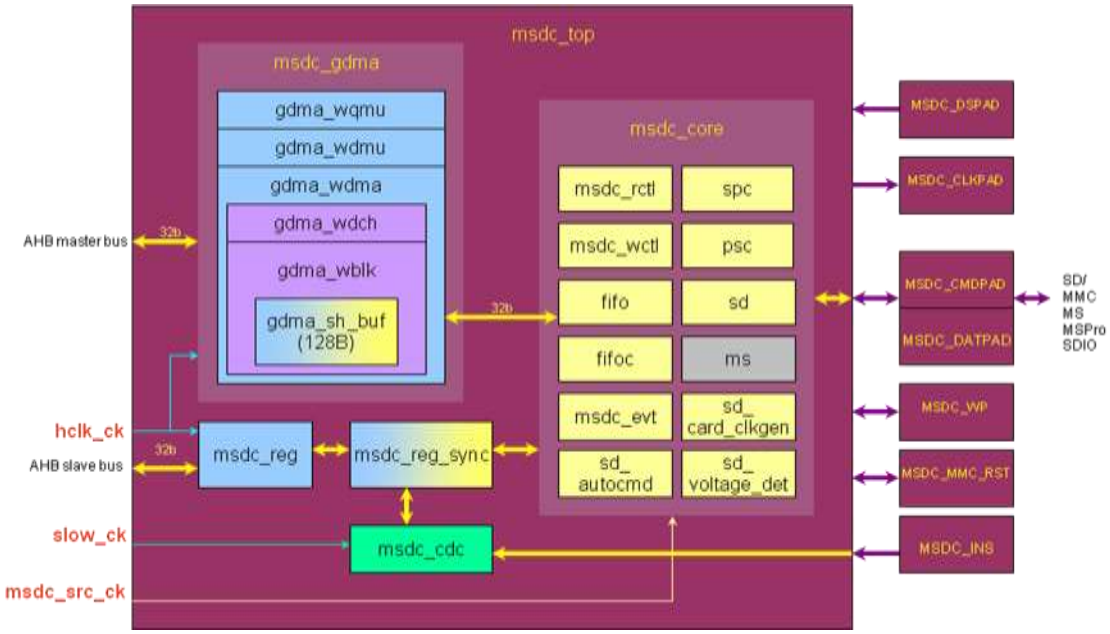


Figure 5-43. MSDC controller block diagram

5.18.1.4 Register Definition

See “MT6797 LTE-A Smartphone Application Processor Software Register Table”.

5.18.2 Memory Stick and SD card Controller: MSDC1

5.18.2.1 Introduction

The MSDC (Memory Stick and SD card Controller) fully supports

- SD memory card specification version 3.0
- SDIO card specification version 3.0
- MMC/eMMC4.5

5.18.2.2 Features

- Interface with MCU by AHB bus
- 32-bit access on AHB bus
- 32-bit access for control registers
- 8-bit/16-bit/32-bit access for FIFO in PIO mode

- Built-in 128 bytes FIFO buffers for transmit and receive
- Built-in CRC circuit
- Supports Basic DMA mode, Basic Descriptor mode, and Enhanced Descriptor mode for SD/MMC
- Interrupt capabilities
- Does not support SPI mode for SD/MMC memory card
- Does not support suspend/resume for SD/MMC memory card
- Supports SD3.0 SDR104, data rate up to 208x4Mbps
- Supports SD3.0 DDR50, data rate up to 50x4x2Mbps (4-bit with clock dual edge)
- Supports eMMC HS200, data rate up to 200x8Mbps (4-bit with clock dual edge)
- Supports SD boot-up mode
- 256 programmable serial clock rates on MS/SD/MMC bus from 100kHz to 208MHz
- Card detection capabilities

5.18.2.3 MSDC Controller Block Diagram

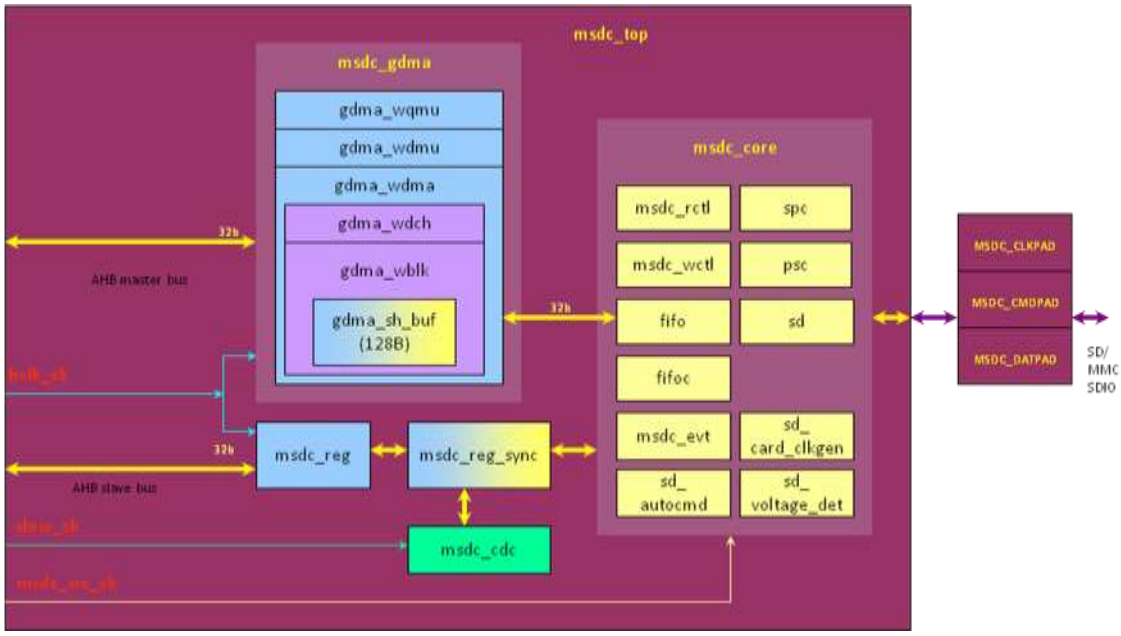


Figure 5-44. MSDC controller block diagram

5.18.2.4 Register Definition

See chapter 3.18 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

5.18.3 Memory Stick and SD card Controller: MSDC2

5.18.3.1 Introduction

The MSDC (Memory Stick and SD card Controller) fully supports

- SD memory card specification version 3.0
- SDIO card specification version 3.0

5.18.3.2 Features

- Interface with MCU by AHB bus
- 32-bit access on AHB bus
- 32-bit access for control registers
- 8-bit/16-bit/32-bit access for FIFO in PIO mode
- Built-in 128 bytes FIFO buffers for transmit and receive
- Built-in CRC circuit
- Supports Basic DMA mode, Basic Descriptor mode, and Enhanced Descriptor mode for SD/MMC
- Interrupt capabilities
- Does not support SPI mode for SD/MMC memory card
- Does not support suspend/resume for SD/MMC memory card
- Supports SD3.0 SDR104, data rate up to 208x4Mbps
- Supports SD3.0 DDR50, data rate up to 50x4x2Mbps (4-bit with clock dual edge)
- 256 programmable serial clock rates on MS/SD/MMC bus from 100kHz to 208MHz
- Card detection capabilities

5.18.3.3 MSDC Controller Block Diagram

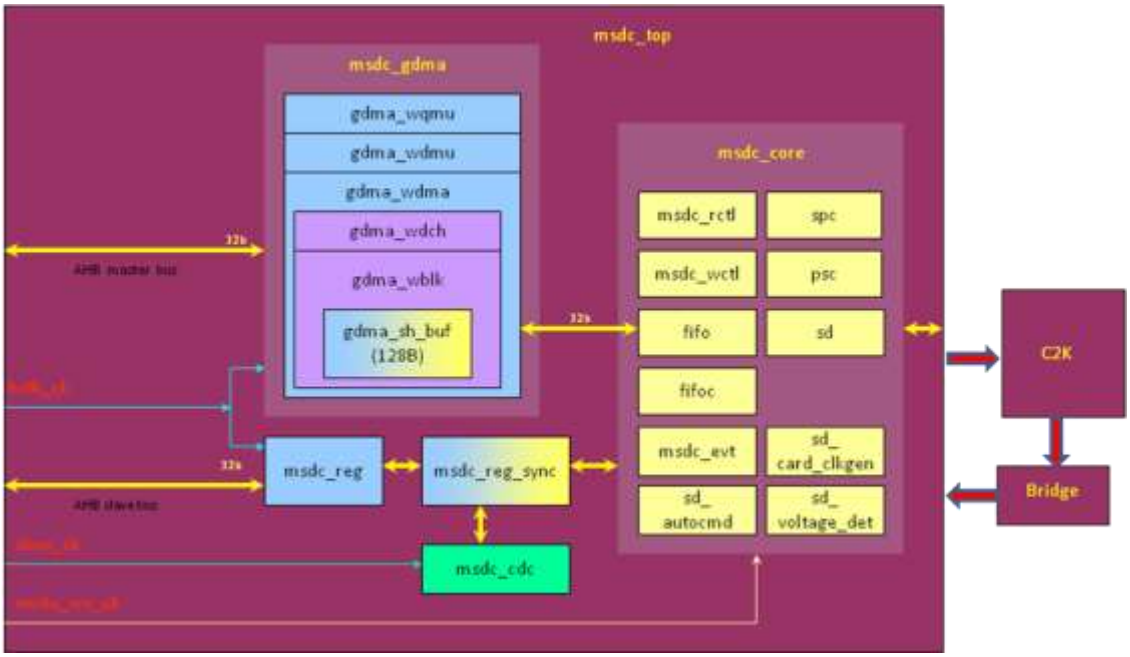


Figure 5-45. MSDC controller block diagram

5.18.3.4 Register Definition

See chapter 3.18 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part I)"*.

6 Multimedia

6.1 Display Controller

6.1.1 Introduction

The display controller contains multimedia data path v2.0 (MDP 2.0) and display pipeline (DISP). The MDP 2.0 is the time sharing pipeline data flow controller to process resizing and rotation by memory access. The display pipeline output pixels to display interface with overlay, color enhancement, adaptive ambient light processing, color correction, gamma correction, over drive, MTK proprietary UFO compression and VESA display stream compression.

6.1.2 Features

The multimedia subsystem has the following features:

- Multimedia Data Path v2.0. It has two read DMA, three resizer, one 2D-sharpness enhancements, one color enhancements, one write DMA and two write rotator.
- Two display pipe lines. The first pipeline has its own read DMA, overlay, color engine, adaptive ambient light processing, color correction, gamma correction, over drive, MTK proprietary UFO compression, VESA display stream compression and display interface controller as basic components. The second pipeline only includes read DMA, overlay and display interface controller.
- Supports 3D display
- Supports color enhancement engine
- Supports adaptive ambient light processing for backlight power saving and sunlight visibility improvement
- Supports color correction and gamma correction for accurate image reproduction
- Supports over drive to improve LCD response time and reduce motion blur slightly
- Supports MTK proprietary UFO compression for DSI interface
- Supports VESA display stream compression
- Supports concurrent dual display output
- Display output interface: Two DSI and one DPI. DPI interface can connect to external LVDS, HDMI, and MHL bridge chips to support these interface.

6.1.3 Block Diagram

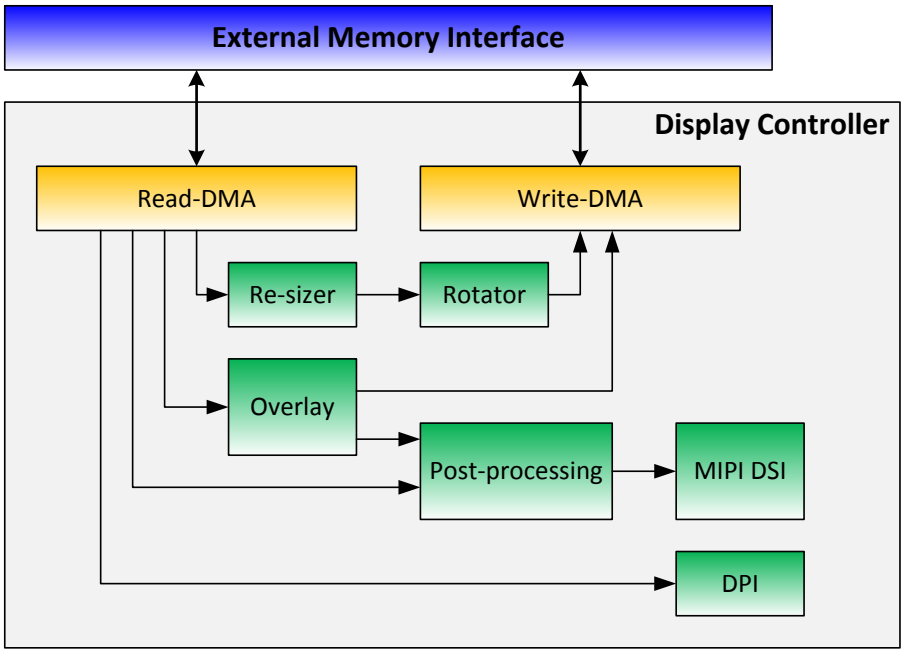


Figure 6-1. Display controller function blocks

6.1.4 Register Definition

See chapter 4.1 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part II)"*.

Register bank	Base address
MMSYS_CONFIG	+14000000h
DISP_UFOE	+14019000h
MDP_RDMA0	+14001000h
MDP_RDMA1	+14002000h
MDP_RSZ0	+14003000h
MDP_RSZ1	+14004000h
MDP_RSZ2 Base	+14005000h
MDP_WROT0	+14007000h
MDP_WROT1	+14008000h
MDP_TDSHP	+14009000h
DISP_OVL0	+1400b000h
DISP_OVL1	+1400c000h
DISP_OVL0_2L	+1400d000h

Register bank	Base address
DISP_OVL1_2L	+1400e000h
DISP_RDMA0	+1400f000h
MDP_WDMA	+14006000h
DISP_WDMA0	+14011000h
DISP_WDMA1	+14012000h
DISP_COLOR	+14013000h
MDP_COLOR	+1400a000h
DISP_CCORR	+14014000h
DISP_AAL	+14015000h
DISP_GAMMA	+14016000h
DISP_DITHER	+14018000h

6.2 DISPLAY PWM Generator

6.2.1 Introduction

The PWM generator provides PWM signals for the LED driver of mobile LCM.

6.2.2 Features

- Operating clock: 26MHz (default) or 104MHz
- Gradual PWM control

6.2.3 Register Definition

See chapter 4.2 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part II)"*.

6.2.4 Programming Guide

1. Turn on DISP_PWM operating clock.
2. Get MMSYS mutex.
3. Set up DISP_PWM_CON_0 and DISP_PWM_CON_1.
4. Write DISP_PWM_EN = 1.
5. Release MMSYS mutex.

6.3 DPI (Digital Parallel Interface)

6.3.1 Introduction

The DPI controller provides data to the companion chip, such as HDMI, MHL, or other bridge chips.

6.3.2 Features

- Programmable 2D/3D, progressive/interlaced timing generator
- Programmable EAV, SAV embedded sync. Timing
- Fixed-coefficient color space transform
- Supports RGB 8-bit/YUV444 8-bit/YUV422 8-bit,10-bit,12-bit output data format
- Supports YC MUX (CCIR656-like) output format
- Support s dual edge output format
- Supports secure display
- 3-tap chroma LPF
- Internal pattern generator

6.3.3 Register Definition

See chapter 4.3 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part II)"*.

6.3.4 Programming Guide

6.3.4.1 General Timing Programming

Programming DPI for one frame timing is easy. Figure 6-2 is illustration of the general timing relationship in DPI. The optional BG defines the region of background color in the image frame. The polarities of VS/HS/DE are all adjustable.

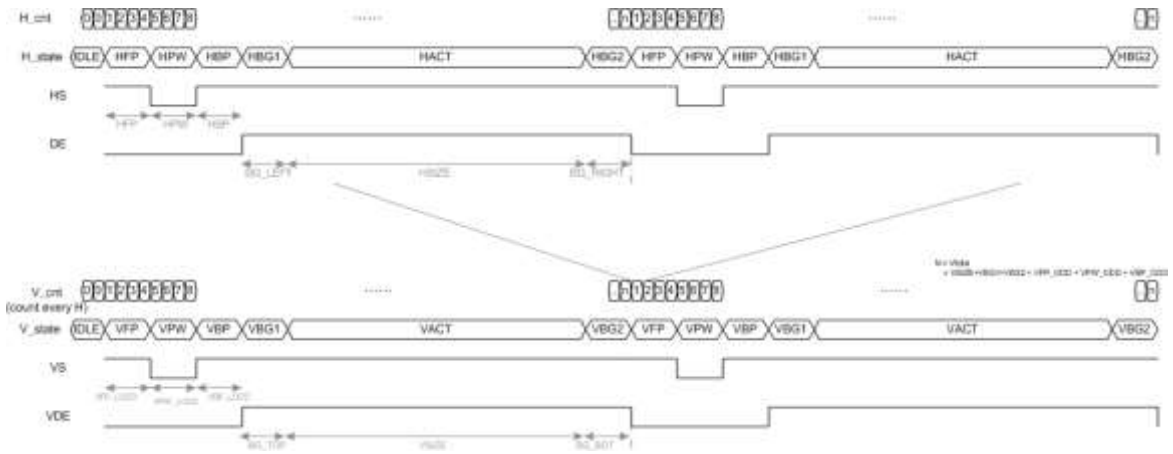


Figure 6-2. Illustration of general timing relationship

6.3.4.2 Data Timing Programming

The DPI engine supports four data/clock timing combinations in dual edge mode. In dual edge mode, one pixel is sent by 2 clock edges, and the order of sending MSB (High bits) and LSB (Low bits) is configurable. Another option is determining sending first half pixel by rising or falling clock edge. Figure 6-3 shows the data/clock timing and their corresponding settings in four cases.

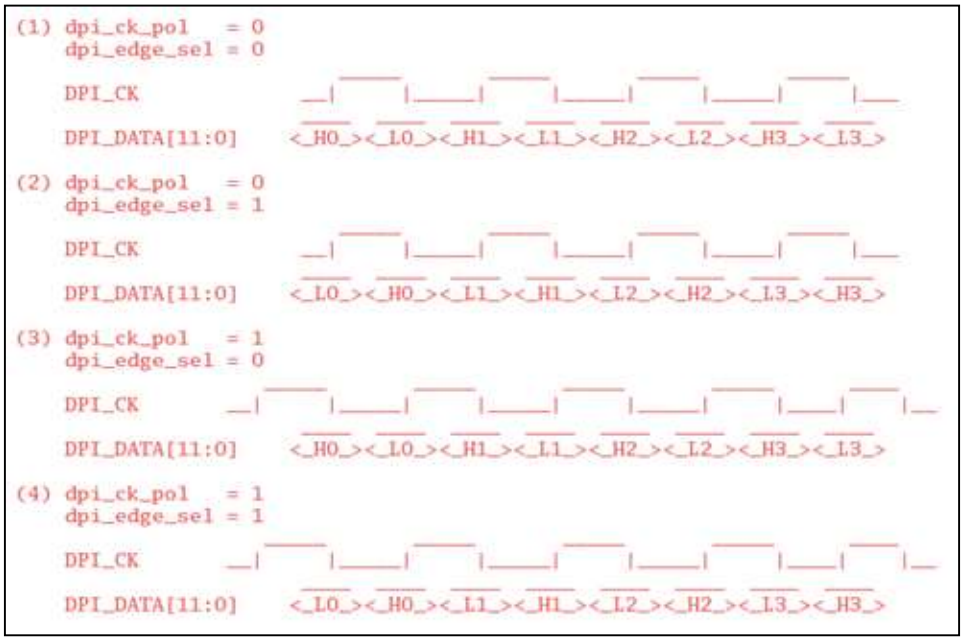


Figure 6-3. Data/Clock timing

Table 6-1 shows explicitly the data map of the four cases above.

Table 6-1. Data map

	Case 1		Case 2		Case 3		Case 4	
Pin name	Rising edge	Falling edge	Rising edge	Falling edge	Falling edge	Rising edge	Falling edge	Rising edge
D0	G4	B0	B0	G4	G4	B0	B0	G4
D1	G5	B1	B1	G5	G5	B1	B1	G5
D2	G6	B2	B2	G6	G6	B2	B2	G6
D3	G7	B3	B3	G7	G7	B3	B3	G7
D4	R0	B4	B4	R0	R0	B4	B4	R0
D5	R1	B5	B5	R1	R1	B5	B5	R1
D6	R2	B6	B6	R2	R2	B6	B6	R2
D7	R3	B7	B7	R3	R3	B7	B7	R3
D8	R4	G0	G0	R4	R4	G0	G0	R4
D9	R5	G1	G1	R5	R5	G1	G1	R5
D10	R6	G2	G2	R6	R6	G2	G2	R6
D11	R7	G3	G3	R7	R7	G3	G3	R7

6.3.4.3 Programming Flow

Figure 6-4 shows the DPI programming flow diagram. First, configure each timing register based on the target frame timing. Next, reset and enable DPI.

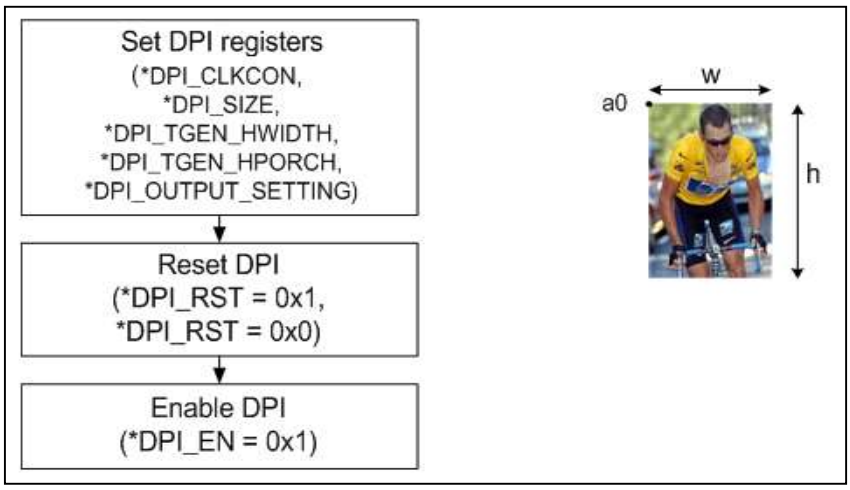


Figure 6-4. Programming flow diagram

6.4 Display Serial Interface

6.4.1 Introduction

The display serial interface (DSI) is based on MIPI Alliance Specification, supporting high-speed serial data transfer between host processor and peripheral devices such as display modules. DSI supports both video mode and command mode data transfer defined in MIPI spec, and it also provides bidirectional transimission with low-power mode to receive mesaages from the peripheral. DSI should work with MIPI_TX_Config module to obtain its engine clock to analog DPHY macro, and it should work with DMA engines in the previous stage of DISP path to read frame pixels from memory.

6.4.2 Features

The DSI engine has the following features for display serial interface:

- 1 clock lane and up to 4 data lanes
- Throughput up to 1G bps for 1 data lane
- Bidirectional data transmisstion in low-power mode in data lane 0
- Uni-directional data transmission in high-speed mode in data lane 0 ~ 3
- 128-enrty command queue for command transmission
- Supports 3 types of video modes: sync-event, sync-pulse, burst mode
- Pixel format of RGB565/RGB666/loosely RGB666/RGB888
- Supports non-continuous high-speed transmission in both clock/data lanes
- Supports command mode frame transmission free-run
- Supports peripheral TE and external TE signal detection
- Supports limited high-speed residual packet transmission during video mode blanking period
- Supports ultra-low power mode control
- Supports low frame-rate (LFR) technique
- Supports frame compression with UFOE module
- Supports frame compression with DSC module
- Dual DSI mode allows transmission with 2 clock lanes and 8 data lanes

6.4.3 Register Definition

See chapter 4.4 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part II)"*.

6.5 JPEG Encoder

6.5.1 Introduction

The hardware JPEG encoder implements the baseline mode of Standard ISO/IEC 10918-1. After initialization by software, the hardware JPEG encoder can generate the entire compressed file. Figure 6-5 shows the procedure of the JPEG encoder. The YUV pixel data are retrieved from the memory and grouped into 8x8 blocks. YUV422 one plane and YUV420 two plane formats are supported. When encoding, the first thing to do is turning the pixel data into the frequency domain using FDCT. After the quantization is done, the quantized DCT coefficients are encoded by RLE and VLC. Then the bitstream of JPEG file is generated.

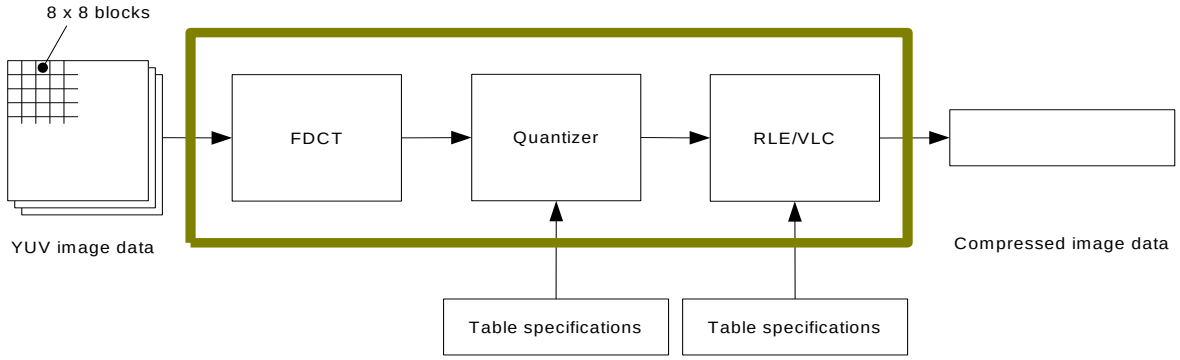


Figure 6-5. Procedure of JPEG encoder

6.5.2 Features

The JPEG encoder supports YUV 422 and 420 formats for color pictures. With software assistance and suitable destination memory address setting; JFIF/EXIF JPEG format can also be supported. For hardware reduction, it uses standard DC and AC Huffman tables for both luminance and chrominance components. To adjust the picture compression ratio and picture quality, there are 15 levels of quantization that can be programmed.

6.5.2.1 Software Reset Mechanism

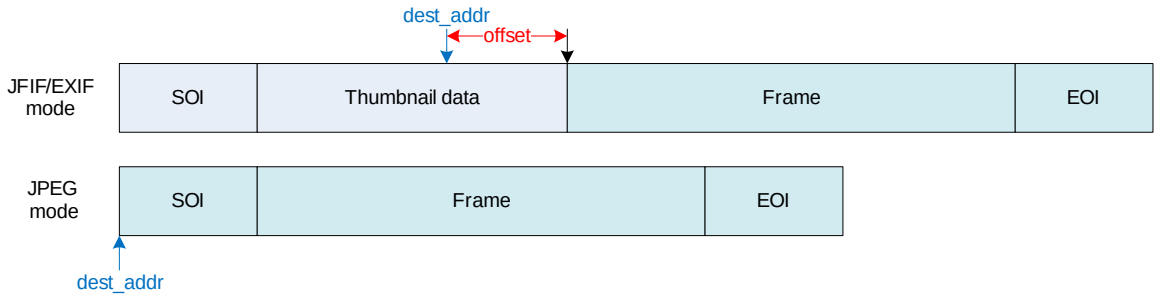
To avoid problem occurred with wrong SMI protocol, the software should do reset based on the following procedure.

1. The EN bit in JPGENC_CTL should be set to 0.
2. The software polls the GMC_IDLE bit in JPGENC_DEBUG_INFO0.

Only when the GMC_IDLE bit is 1 can the RSTB bit in JPGENC_RST be set to 0 to do the reset scheme. Be sure to follow this procedure to do software reset.

6.5.2.2 Byte Offset Address Setting

To align SMI bus bitwidth, the buffer address should be 16-byte aligned. However, to reduce software bitstream copy and concatenate effort, software can program a byte offset setting (from 0~15 bytes) to offset the start position of bitstream written out by hardware. The illustration of this feature is as the following.



- Example: If SOI starts at 0x0000,
  - JFIF/EXIF mode: SOI+thumbnail data length = 50 bytes
    - dest_addr = 0x0030
      offset = 0x0002

6.5.3 DRAM Buffer Requirement

- Source frame buffer
  - YUV422:
    - One frame buffer
    - Buffer size
      - {[(width_y*2) + 127]/128}*128*[(height_y + 7)/8]*8
  - YUV420: Two frame buffers
    - Two frame buffers
    - Buffer size
      - Y: [(width_y + 127)/128]*128*[(height_y + 15)/16]*16
      - UV: [(width_y + 127)/128]*128*[(height_y + 7)/8]*8
- Bitstream buffer
  - One buffer
  - Buffer size (suggested)
    - YUV422: width_y* height_y * 2
    - YUV420: width_y* height_y * 1.5

- Must be multiple of 128 bytes.

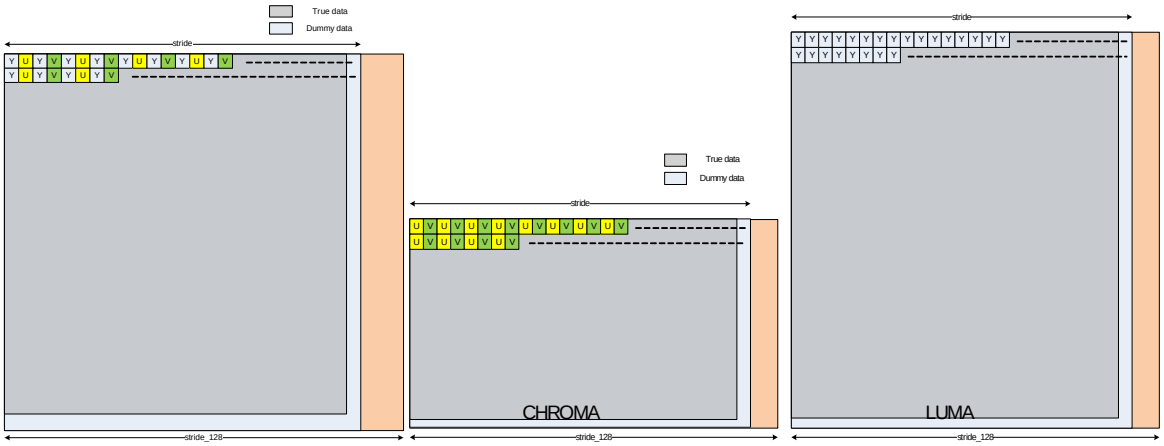


Figure 6-6. Memory footprint of source frame buffer

6.5.4 Register Definition

See chapter 4.5 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part II)"*.

6.6 JPEG Decoder

6.6.1 Introduction

At present most images must be stored as JPEG format compressed files. In order to display this kind of file and boost image processing performance, the hardware JPEG decoder is developed. The JPEG decoder is designed to decode baseline JPEG file with general YUV sampling combinations. (Note that the progressive decoding is NOT supported in the hardware decoder).

To obtain the best speed performance, the JPEG decoder handles all portions of JPEG files except for the 17-byte SOF marker. The software only needs to program related control registers based on the SOF marker and wait for an interrupt coming from the hardware. Figure 6-7 shows the basic JPEG file structure and starting address that JPEG decoder needs. The information of DQT and DHT table is included in the JPEG file, which needs to be parsed by the JPEG decoder and store in the memory. Taking into consideration the limited size of memories, the hardware decoder also supports breakpoints insertion in huge JPEG files. Breakpoints insertion allows software to load partial JPEG file from the external flash into the internal memory if the JPEG file is too large to sit internally at a time.

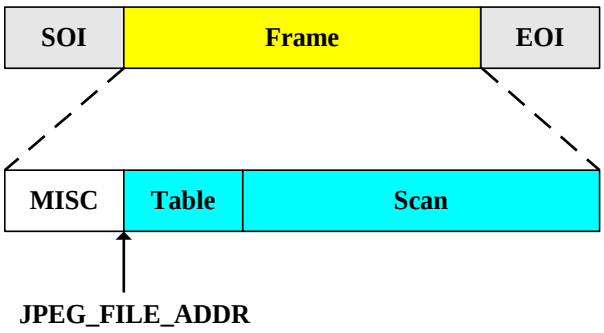


Figure 6-7. The basic structure of JPEG files

6.6.2 Features

The sleep controller receives the command from system software and controls the built-in power management unit to cut off the power supply of external clock source. The power of external clock source can be resumed by several events predefined by users. These events are issued by:

- Baseline JPEG decoding (No progressive decoding, no restart marker decoding, bypass to software solution)
- Sampling format limitation: See Figure 6-8
- Hardware table parsing (SW handle SOF parsing)
- Supports file breakpoint

- 1/2/4/8 block resize (some limitation due to format conversion, see Section 6.7.2.3)
- Supports MCU rows pause/resume
- Error handling by bitstream overflow detection
  The supported input sampling format of JPEG decoder is illustrated in Figure 6-8

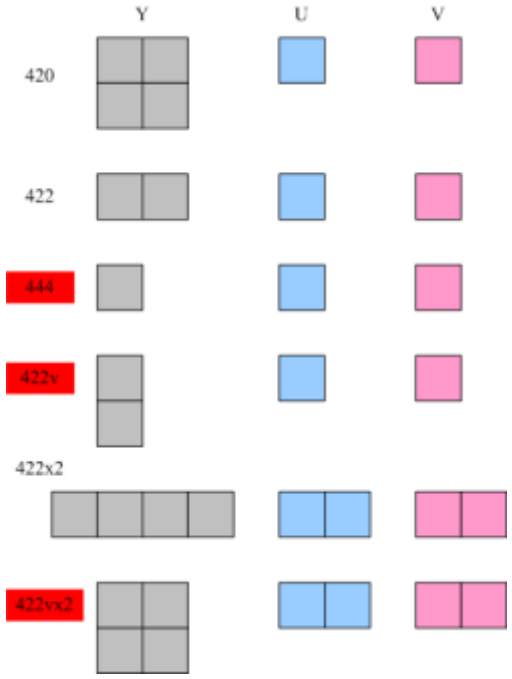


Figure 6-8. Supported sampling format of input JPEG file

For input sampling format that is not supported by the hardware, software solutions are adopted.

6.6.2.1 Software Reset Mechanism

To avoid unexpected memory access behavior, sot reset scheme is also developed for JPEG decoder (same procedure with JPEG encoder). First, assert the soft reset bit in JPGDEC_RESET. Then hardware will issue a done interrupt when its state is not busy. Software de-assert the soft reset bit and clear the interrupt status. Finally, assert the hard reset bit in JPGDEC_RESET to do reset process. To reset JPEG decoder and GDMA in the direct couple mode, perform soft reset mechanism to JPEG decoder and reset JPEG decoder first. Then perform GDMA soft reset mechanism to reset GDMA.

6.6.2.2 Operation Modes

6.6.2.2.1 Normal Mode (Full Frame Mode)

In normal mode, JPEG decoder will output full frame to DRAM. Then ISP/MDP takes over for resizing and image processing. For software usage, one set of frame buffer should be allocated to JPEG decoder, and then set corresponding configuration and trigger hardware for decoding. JPEG decoder will return interrupt when full frame is decoded.

6.6.2.2.2 Pause/ Resume Mode

In pause/resume mode, JPEG decoder will output multiple MCU rows to DRAM based on software configuration and pause. JPEG decoder will return interrupt when assigned MCU rows are decoded. Next configuration and trigger will resume JPEG decoder, and following MCU rows will be outputted correspondingly. For software usage, one set of MCU rows buffer should be allocated to JPEG decoder, when JPEG decoder finish assigned MCU rows and pause, software should do next configuration and trigger hardware until full frame is decoded. JPEG decoder will return interrupt when full frame is decoded.

6.6.2.3 Block Resizing

To save memory usage for JPEG image with extreme large size, block level resizing is supported by JPEG decoder. Simple 1/2, 1/4, 1/8 block level resize using 2-tap bilinear filter is adopted.
Since sampling format conversion is needed for JPEG decoder. Resize limitation exists for some source format. (eg. for YUV444 input, only 1/4 resize could be performed)

		Config			
src format	dst format (3/1-plane)	Y_rsz	UV_h_rsz	UV_v_rsz	Limitation on real rsz
420	420	real_rsz	real_rsz	real_rsz	No
422	422	real_rsz	real_rsz	real_rsz	No
444	422	real_rsz	1+real_rsz	real_rsz	~ 1/4
422v	420	real_rsz	1+real_rsz	real_rsz	~ 1/4
422 x 2	422	real_rsz	real_rsz	real_rsz	No
422v x 2	420	real_rsz	1+real_rsz	real_rsz	~ 1/4

6.6.2.4 Error Handling Mechanism

Error handling is first performed in software header parsing stage. If header is corrupted, software should not pass such file to hardware decoder. In hardware decoding stage, only part of error detection is supported (No error concealment capability). Several types of detection are supported: including invalid Huffman entry and block overflow. One flag will be raise when such error is detected

and error interrupt will be issued. Hardware will stop when 0xffd9 is encountered or the file size count is reached. If software would like to stop JPEG decoder when error detected, please use the soft reset scheme.

6.6.3 DRAM Buffer Requirement

For JPEG decoder, two types of buffer should be allocated by software, bitstream buffer and decoded frame buffer.

The bitstream buffer is used to store the jpeg file bitstream, the start address should be 16-byte aligned and the address position should be the first marker segment (such as DQT or DHT) after skipping all the APP markers. The break point address could be set based on the limited memory size. The related control register is JPEG_FILE_ADDR and JPEG_FILE_BRP. The allocated bitstream buffer size should be multiple of 128 bytes.

The decoded frame buffer is separate 3-plane planar format and. The supported color format is YUV422, YUV420 and grayscale Figure 6-9 shows an illustration of the decoded frame buffer example.

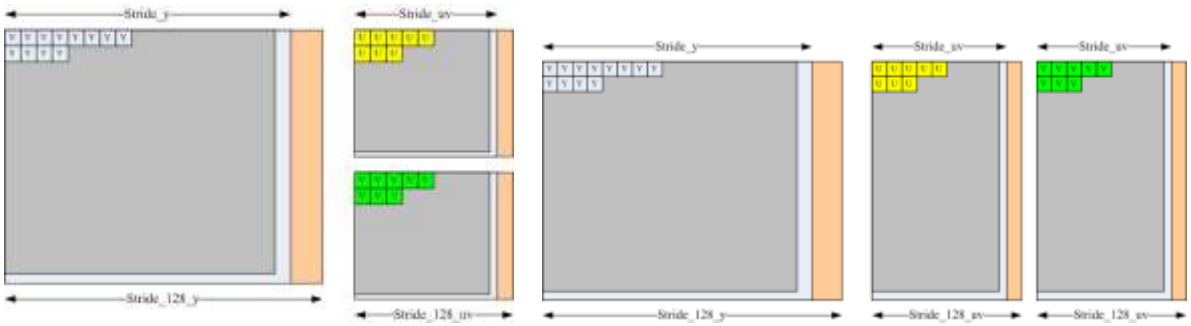


Figure 6-9. Memory footprint of source frame buffer

- Bitstream buffer
  - One buffer
  - Buffer size: depends on available memory buffer
    - **Must be multiple of 128 bytes**
    - Loading as many as possible is suggested for better decoding performance
    - Supports breakpoint feature
- Destination frame buffer
  - YUV422:
    - Three frame buffer
    - Buffer size
      - Y: [(width_y + 127)/128]*__128__*[(height_y + 7)/8]*8
      - U: [(width_u + 127)/128]*__128__*[(height_y + 7)/8]*8

  - V: [(width_v + 127)/128]*128*[(height_y + 7)/8]*8
- YUV420:
  - Three frame buffer
  - Buffer size
    - Y: [(width_y + 127)/128]*128*[(height_y + 15)/16]*16
    - U: [(width_u + 127)/128]*128*[(height_y + 7)/8]*8
    - V: [(width_v + 127)/128]*128*[(height_y + 7)/8]*8

6.6.4 Register Definition

See chapter 4.6 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part II)"*.

6.7 Video Decoder

6.7.1 Introduction

The video decoder (VDEC) supports multi-standard video compression formats, which greatly reduces the CPU loading and achieves high performance video decompression.

The video standard supported by VDEC includes:

- MPEG1/2
- H.264 CBP/MP/HP
- MPEG4 ASP
- DIVX3/DIVX4/DIVX5/DIVX6/DIVX HD/XVID
- Sorenson H.263, H.263
- De-Blocking Filter for MPEG2, H.263
- H.264 decoer Constraint Baseline Profile/Main/High Profile
- HEVC decoder main profile @ L5
- VC1, WMV9 SP @ main level, MP @ high level, AP @ L4
- VP8
- VP9

VDEC supports 4K 30fps under the limitation that picture size > 4K, (i.e. does not support picture width > 4096 or picture height > 2160).

6.7.2 Block Diagram

The architecture and core blocks of VDEC are shown in Figure 6-10, including the following parts: Entropy Decoder, IS/IQ/IT, MV Calculation, Intra Prediction, Motion Compensation and De-blocking Filter. The input to VDEC is a compressed video bitstream. After the decoding process, the reconstructed video will be sent to the display stage.

6.7.3 Interface

The interface of VDEC is shown in Figure 6-11. Related modules include SMI interface, APB interface and handshaking bus between VDEC and CDP. The output format supported by VDEC is:

- NV12_BLK (Video block mode), 420 format block mode, 2 plane (UV)
- NV12_BLK_FCM (video field compact mode), 420 format block mode, 2 plane (UV)

VDEC uses DRAM as bitstream input, working buffer, reference buffer and output, and DRAM access process is achieved by using SMI interface. Seven sets of SMI interfaces with 128 bits of read/write are

used including MC/PP/PP_WRAP/AVC_MV/PRED_RD/PRED_WR/VLD. In addition, all ports are EMI ports, i.e. no SYSRAM is required.

Register settings are passed to VDEC by APB interface. There is one set of APB interfaces.

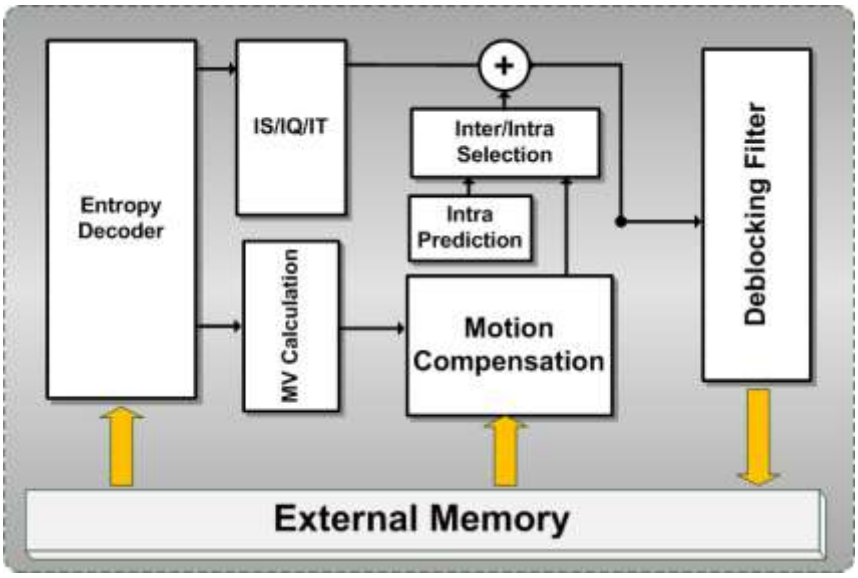


Figure 6-10. Block diagram of video decoder

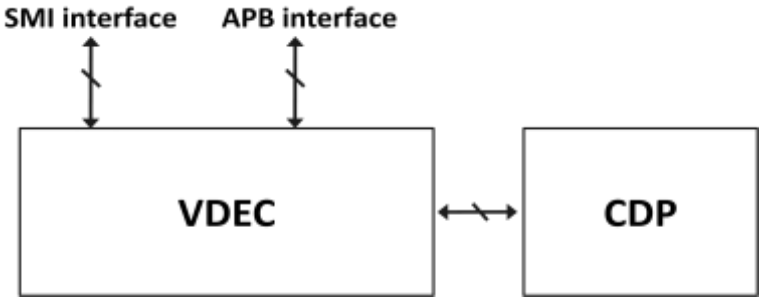


Figure 6-11. Interface of VDEC

6.7.4 Programming Guide

This section defines the recommended programming procedure for using VDEC to perform video decompression correctly. Moreover, this section also introduces common functions and settings, that is, these functions do not change from different video coding standards.

6.7.4.1 Base Settings

This section describes useful settings before programming VDEC, including clocks and base address of VDEC.

6.7.4.1.1 Clocks

Two clocks are required for VDEC:

- smi_clk : 500 MHz
- vdec_clk: 500 MHz

For more details, see the CKGEN register map.

6.7.4.1.2 VDEC Register Base Address

Base address of each sub-module is described in Table 6-2.

Table 6-2. VDEC base address

CS	BASE (hex)	HW cs	Comment
VDEC_MISC	0x16020000	cs_vdec_dv	Legacy naming is DV_BASE.
VLD	0x16021000	cs_vdec_vld	
VLD_TOP	0x16021800	cs_vdec_vld	Partial bank of VLD_BASE, i.e. (VLD | 0x800)
MC	0x16022000	cs_vdec_mc	
AVC_VLD	0x16023000	cs_avc_vld	
AVC_MV	0x16024000	cs_avc_mv	
PP	0x16025000	cs_vdec_pp	
VDEC_GCON	0x16000000		VDEC global control registers
SMI_LARB1	0x16010000		SMI LARB1 control registers

6.7.4.2 VDEC Hardware Architecture

Figure 6-12 is the hardware architecture.

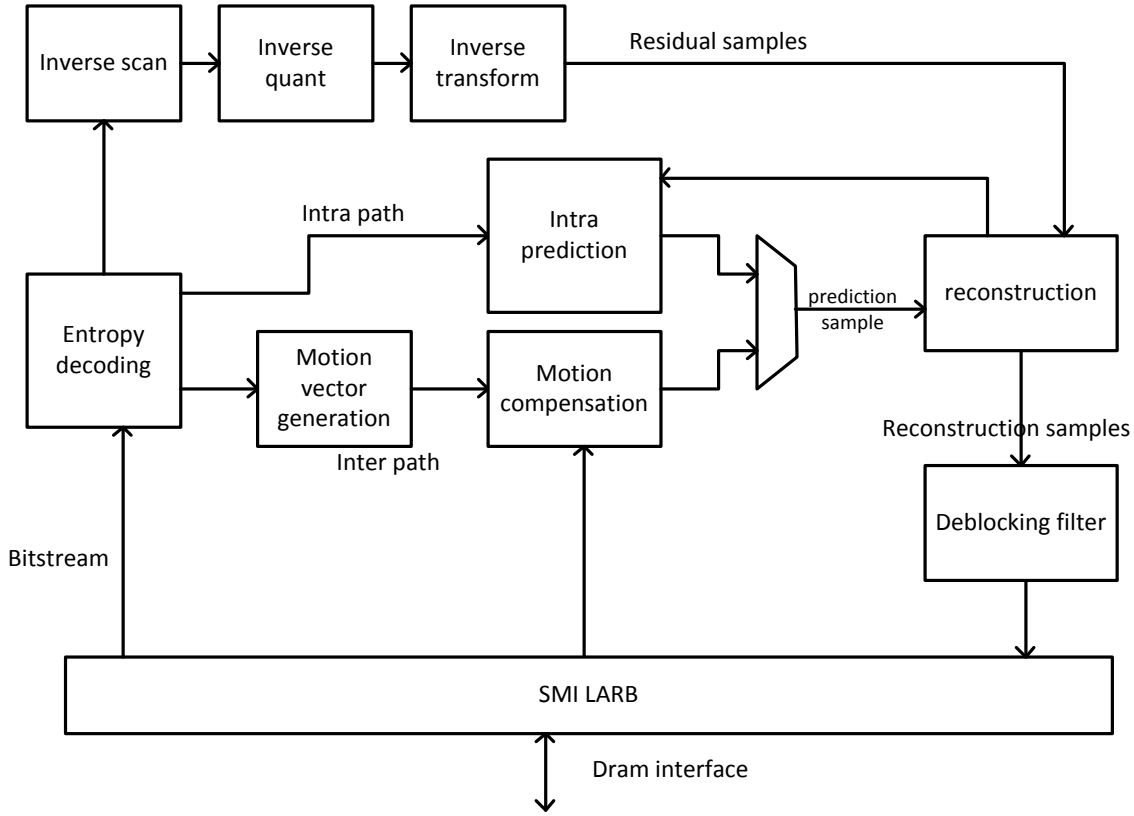


Figure 6-12. VDEC hardware architecture

6.7.4.3 Firmware Hardware Partition

Figure 6-13 is the typical decoding flow with FW/HW partition. SW takes charge frame level setting, including syntax decoded from frame header and proper DRAM buffer allocation, and triggers HW to decode the bitstream. An interrupt signal will be sent to CPU when HW finish decoding a frame and FW can reset HW and program setting for next frame if necessary. Details of FW decoding are described in section 6.7.4.4.

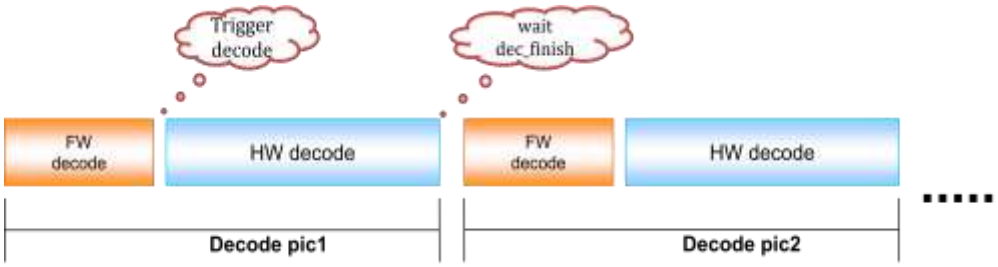


Figure 6-13. VDEC decoding flow

6.7.4.4 FW Decoding Flow

This section describes our recommended step-by-step FW setup flow. Each step should be followed to ensure a correct decoding process.

1. Soft reset
   a. Issue reset vld_reg_66[0] = 1.
   b. Turn on VDEC related clocks.
   c. Release soft reset vld_reg_66[0] = 0.
2. Initialize Bitstream DMAs and Barrel-Shifters.
3. Decode frame layer syntax.
4. Maintain DPB buffer.
5. HW registers setting
   a. SQT setting
   b. MV setting
   c. MC setting
   d. De-blocking (PP) setting
   e. VLD setting
6. Trigger HW decoding.
7. Wait for decoding picture to finish via VDEC interrupt (HW auto turns off VDEC related clocks).

6.7.4.4.1 Software Reset and Power Control Flow

Figure 6-14 describes the software reset and power control flow. Details of each step will be described in the following sections.

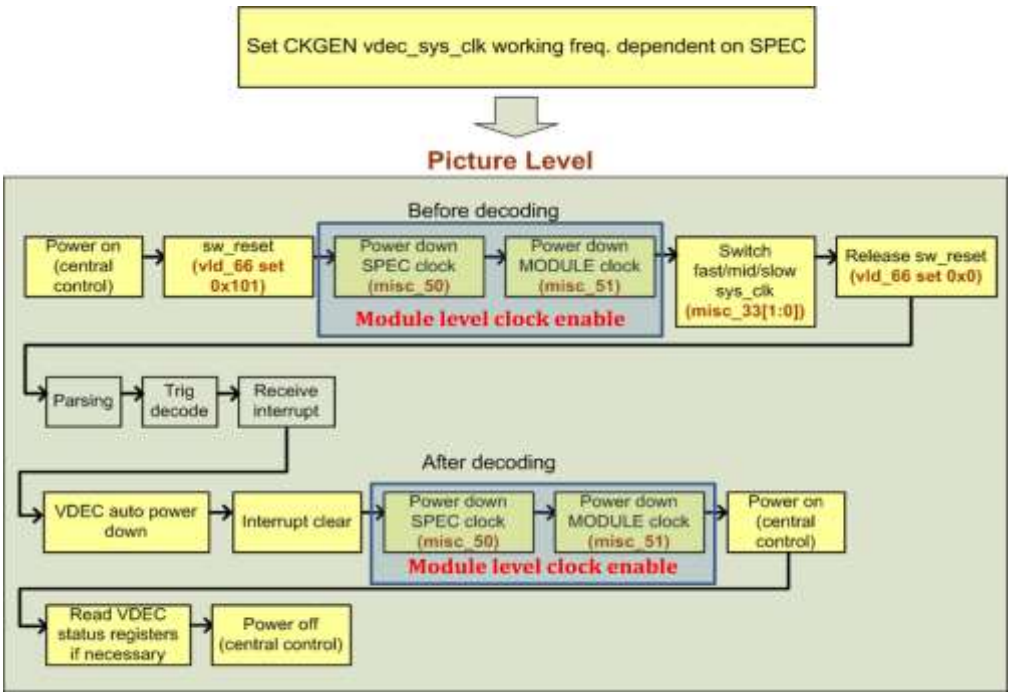


Figure 6-14. Software reset and power control

- ✓ Power on (central control)
  - Set VDEC_GCON reg 0 bit[0] (vdec subsys clock enable) = 1.
  - Enable global clocks of VDEC (dram clock, vdec_sys clock, bus clock).
- ✓ VDEC auto power-down
  - Auto turn off dram clock, vdec_sys clock.
  - Auto set pdn_con_spec (vdec_misc_reg 50) = 32'hffffffff.
  - Auto set pdn_con_module (vdec_misc_reg 51) = 32'hffffffff.
- ✓ Interrupt clear
  - No interrupt clear from CPU
  - Use internal VDEC RISC clear int
    - VDEC_MISC reg 41
      - Bit [0] (risc_clr_int_mode) always set to 1.
      - Bit [4] (risc_int_clr): internal VDEC RISC clear int
- ✓ Power off (central control)
  - Set VDEC_GCON reg 0 bit[0] (vdec subsys clock enable) = 0

6.7.4.4.2 Turn off VDEC Auto Power Down

FW can turn off VDEC auto power-down by disabling the setting of "auto turn off dram clk, vdec sys clock" and disabling "Auto set pdn_con_spec = 32'hffffffff & Auto set pdn_con_module = 32'hffffffff".

Related register settings are:

- Disable “Auto turn off dram clock, vdec_sys clock”
  - Set VDEC_GCON reg 6 bit[0] = 1
- Disable “Auto set pdn_con_spec = 32’hffffffff & Auto set pdn_con_module = 32’hffffffff”
  - Set VDEC_MISC reg 59 bit[0] = 1

The decoding process will change after FW turns off VDEC auto power-down control. The difference is shown in Figure 6-15.

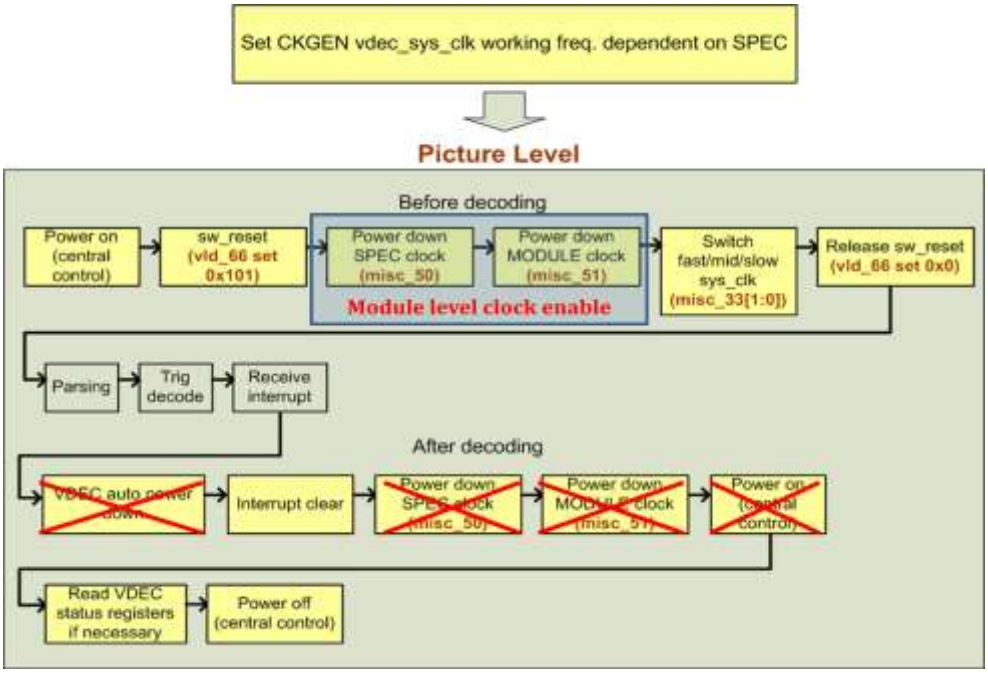


Figure 6-15. Software reset and power control (turn off VDEC auto power down)

6.7.4.4.3 Other Relative Registers about Power Control

FW can disable “Auto turn off dram clock, vdec_sys clock” when dec_error occurs by setting VDEC_GCON reg 6 bit[4] = 1.In addition, when dec_error occurs, there is **always no** “Auto set pdn_con_spec = 32’hffffffff & Auto set pdn_con_module = 32’hffffffff”.

6.7.4.4.4 Clock and Power-down Setting

a. Enable global clocks of VDEC (DRAM clock, vdec_sys clock, bus clock)
   i. Set VDEC_GCON reg 0 bit[0] (vdec subsys clock enable) = 1.
b. Enable SRAMs of SMI/VDEC.

i. ***SLEEP_IFR_PWR_CON = (**SLEEP_IFR_PWR_CON&0xffff00ff);
ii. ***SLEEP_VDE_PWR_CON = (**SLEEP_IFR_PWR_CON&0xffff00ff);

c. Set clock gating for each standard and modules
   i. Software reset: set VLD reg 66 = 0x101
   ii. Set **pdn_con_spec** & **pdn_con_module** setting for different spec
      1. **pdn_con_spec** (**VDEC_MISC reg 50 (0xC8)**)
         a) **pdn_con_module** (**VDEC_MISC reg 51 (0xCC)**)
   iii. set **vdec_sys_clk_sel** for different spec :
      1. **vdec_sys_clk_sel (VDEC_MISC reg 33(0x84))**
   iv. release software reset: reset VLD reg 66 = 0x0

VDEC system clock frequent, as well as the module level power down control, depends on different specs.

6.7.4.4.5 Interrupt Clear

In MT6797, there is no interrupt clear from CPU, i.e. interrupt clear should be achieved by FW setting up internal VDEC RISC. To perform interrupt clear, set up the following:

VDEC_MISC reg 41
- Bit [0] (risc_clr_int_mode) always set to 1
- Bit [4] (risc_int_clr): Internal VDEC RISC clear int

6.7.4.5 VDEC Break Function

This section describes the correct steps for FW to break VDEC before finishing decoding a picture.

1. Set up vdec_break.
   – Set VDEC_MISC reg 64 Bit [0] (vdec_break) = 1'b1.
2. Monitor status vdec_break_ok.
   – VDEC_MISC reg 65 Bit [0]: vdec_break_ok_0
   – VDEC_MISC reg 65 Bit [4]: vdec_break_ok_1
   – Monitor until both vdec_break_ok _0 = 1 && vdec_break_ok_1= 1.
3. Software reset
   – VLD reg 66 Bit [0]: vdec_sw_rst
   – Set vdec_sw_rst = 1, then reset vdec_sw_rst = 0
4. Turn off the clocks.
5. Finish.

6.7.5 Register Definition

See chapter 4.7 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part II)"*.

6.8 H.264/HEVC Video Encoder

6.8.1 Introduction

This design is main stream video encoder consisting of two video encoders: H.264, and HEVC. It is capable of encoding 1080P video at 60 frames per second (FPS) with promising superior video quality for H.264 and up to 2160P video at 30 FPS for HEVC. This IP supports various encoding methods that satisfy basic requirement of easy software controllability. Furthermore, with advanced encoding technology, it brings astonishing high quality and low memory bandwidth requirements. It also considers the usage of portable devices and provides several power saving capabilities.

The supported video codec and their capability are listed in the table below.

Table 6-3. Main features

H.264 encoder	Profile	High
	Level	4.2
	Speed	**1920x1088p@60fps**
HEVC encoder	Profile	Main
	Level	5.0
	Speed	**3840x2176p@30fps**

The video encoder takes DRAM as input, output, and working buffer. It reads input frame buffers, executes video encoding and writes encoded bitstream to output buffer. The driver software maintains all buffers and assign proper value to video encoder to allow hardware to work correctly. Figure 6-16 shows the procedure of the video encoder. YUV420 two plane scan-line (NV12/NV21), YUV420 three plane scan-line (YV12/I420) or MTK block formats are supported.

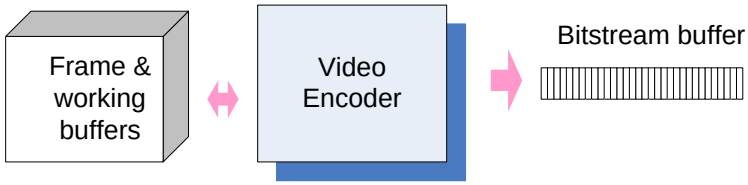


Figure 6-16. Procedure of video encoder

6.8.2 Block Diagram

Figure 6-17 is the brief IP architecture and local on-chip-bus architecture. The interface for controlling it consists of ARM APB bus and MediaTek proprietary SMI bus. It reports hardware event through interrupt or software polling. In addition, it adopts several SMI ports and one APB port. The video encoder is configured by software through APB interface. As the register is configured, the sequencer

will send the corresponding control signals to trigger sub-modules. DMA will acquire and store back the image data and bitstream from and to memory according to the configured address. ME conducts motion estimation to decide motion vector for later encoding. MC conducts motion compensation to give predicted pixel values. TQ conducts transform and quantization operation and write reconstructed pixels to DB and quantized transformed coefficient to EC. DB conducts de-blocking operation and allows DMA to store back the processed frame as the next frame's reference frame. EC conducts entropy encoding, and the coding can be variable length code, context based arithmetic code, or context based variable length code. The encoded bitstream will be written to memory by DMA.

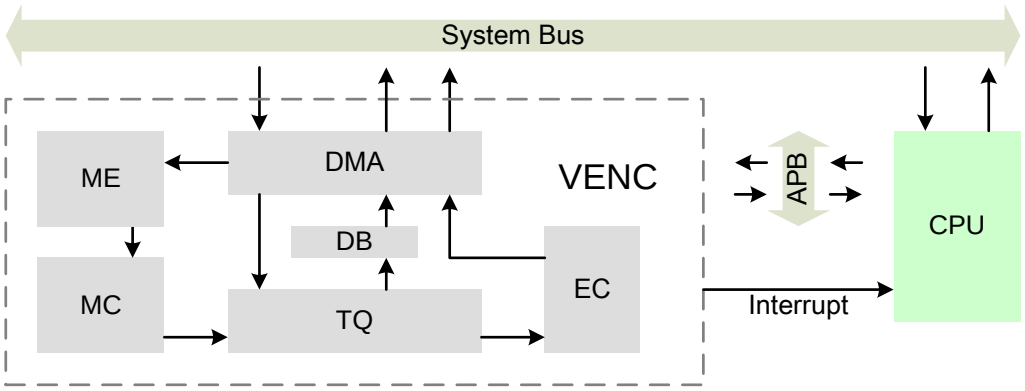


Figure 6-17. Block diagram of video encoder

6.8.3 Encoding Parameters in ARM Driver Code

This section describe parameters for basic parameters of video encoding in driver code, which shows as the following:

Code example

```
/*
 * struct venc_enc_prm - encoder settings for VENC_SET_PARAM_ENC used in
venc_if_set_param()
 * @yuv_fmt: input yuv format
 * @h264_profile: V4L2 defined H.264 profile
 * @h264_level: V4L2 defined H.264 level
 * @width: image width
 * @height: image height
 * @buf_width: buffer width
 * @buf_height: buffer height
 * @frm_rate: frame rate
 * @intra_period: intra frame period
 * @bitrate: target bitrate in kbps
```

Code example

```
 */
struct venc_enc_prm {
        enum venc_yuv_fmt yuv_fmt;
        unsigned int h264_profile;
        unsigned int h264_level;
        unsigned int width;
        unsigned int height;
        unsigned int buf_width;
        unsigned int buf_height;
        unsigned int frm_rate;
        unsigned int intra_period;
        unsigned int bitrate;
};
```

The following shows the definition of parameters video encoding.

Code example

```
enum venc_set_param_type {
        VENC_SET_PARAM_ENC = 1,
        VENC_SET_PARAM_FORCE_INTRA,
        VENC_SET_PARAM_ADJUST_BITRATE,
        VENC_SET_PARAM_I_FRAME_INTERVAL,
        VENC_SET_PARAM_ADJUST_FRAMERATE = 6,
        VENC_SET_PARAM_SKIP_FRAME,
        VENC_SET_PARAM_PREPEND_HEADER,
        VENC_SET_PARAM_NON_REF_P,
        VENC_SET_PARAM_TS_MODE,
};
```

6.8.4 Register Definition

See chapter 4.8 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part II)"*.

6.9 MFG

6.9.1 Introduction

MFG contains Mali-T880 MP4 GPU and clock/reset control logic. The Mali-T880 series of GPUs process extremely complicated graphics and perform general processing tasks assigned by the main application processor. This diagram shows the main components and interface of MFG.

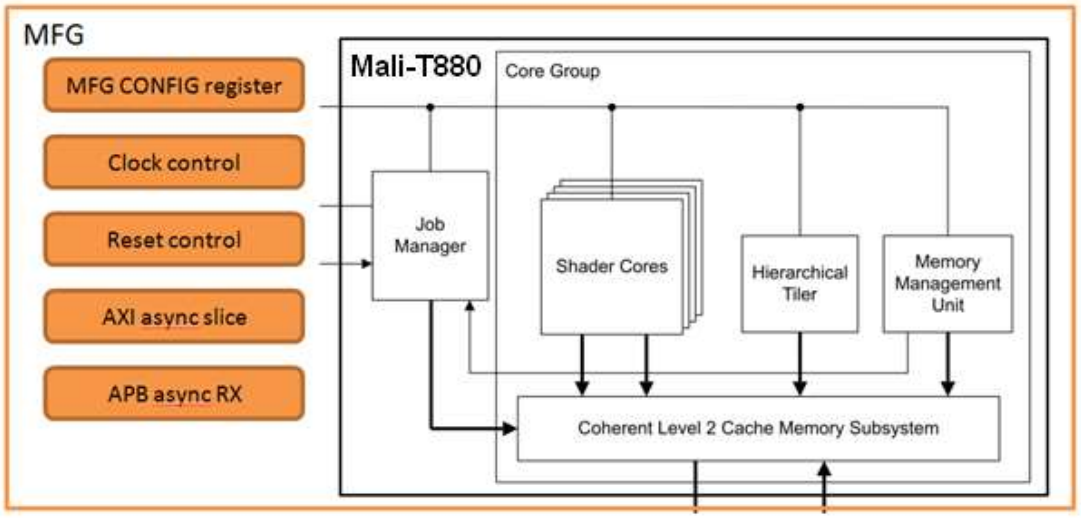


Figure 6-18. Block diagram and interface of MFG

6.9.2 Features

The Mali-T880 MP4 GPU includes the following features:

- A rich API feature set with high-performance support for both shader-based and fixed-function graphics APIs. These API graphics industry standards are:
  - *OpenGL ES 1.1 Specification* at *Khronos*
  - *OpenGL ES 2.0 Specification* at *Khronos*
  - *OpenGL ES 3.0 Specification* at *Khronos*
  - *OpenCL 1.0 Specification* at *Khronos*
  - *OpenCL 1.1 Specification* at *Khronos*
  - *OpenCL 1.2 Specification* at *Khronos*
- Anti-aliasing capabilities
- An effective core for *General Purpose computing on GPU* (GPGPU) applications
- High memory bandwidth and low power consumption for 3D graphics content.
- Image quality using double-precision FP64, and anti-aliasing.
- Frame buffer compression.

- Bus protocol
  - 128-bit AXI bus with up to read 64 outstanding and write 32 outstanding
  - 32-bit APB bus
- Level-2 cache
  - 512KB
  - 4-way set associative
- Performance
  - Triangle rate 350 M Tri/sec
  - Fill rate 2800 M pixels/sec
  - Shader rate 143.2 GFLOPS

6.9.3 Register Definition

See chapter 4.9 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part II)"*.

6.10 SENINF_TOP (Sensor Interface)

6.10.1 Introduction

The SENINF_TOP module transfers sensor signals into image pixels and passes them to ISP.

6.10.2 Features

- MIPI interface
  - Three MIPI interfaces
  - Virtual channel/data type data interleaving
- Serial interface
  - No serial interface
- Parallel interface
  - No parallel interface
- Four sensor master clocks

6.10.3 Register Definition

See chapter 4.10 of *"MT6797 LTE-A Smartphone Application Processor Software Register Table (Part II)"*.

Register bank	Base address
seninf_top	+1a040000h
mipi_RX_ana	+10217000h
mipi_TX_config0	+10215000h
mipi_TX_config1	+1021e000h
mipi_TX_cphy	+10218000h
seninf1	+1a040200h
seninf1_csi2	+1a040a00h
mipi_RX_config_csi0	+1a040800h
seninf1_mux	+1a040d00h
SENINF1_NCSI2	+1a040700h
seninf1_ocsi2	+1a040300h
SENINF1_TG	+1a040600h
seninf2	+1a041200h
seninf2_csi2	+1a041a00h
mipi_RX_config_csi1	+1a041800h
seninf2_mux	+1a041d00h
SENINF2_NCSI2	+1a041700h
seninf2_ocsi2	+1a041300h
SENINF2_TG	+1a041600h
seninf3	+1a042200h
seninf3_csi2	+1a042a00h
mipi_RX_config_csi2	+1a042800h
seninf3_mux	+1a042d00h
SENINF3_NCSI2	+1a042700h
seninf3_ocsi2	+1a042300h
SENINF3_TG	+1a042600h
seninf4	+1a043200h
seninf4_csi2	+1a043a00h
mipi_RX_config_csi3	+1a043800h
seninf4_mux	+1a043d00h
SENINF4_NCSI2	+1a043700h
seninf4_ocsi2	+1a043300h
SENINF4_TG	+1a043600h
seninf5_ccir656	+1a044400h
seninf5	+1a044200h
seninf5_mux	+1a044d00h
SENINF5_TG	+1a044600h
seninf6_mux	+1a045d00h
seninf7_mux	+1a046d00h

Register bank	Base address
seninf8_mux	+1a047d00h
SENINF_N3D_A	+1a040100h
SENINF_N3D_B	+1a041100h