MEDIATEK

everyday genius

# MT6737 LTE Smartphone Application Processor Functional Specification

| | |
|---|---|
| Version: | 1.0 |
| Release date: | 2016-01-15 |

Specifications are subject to change without notice.

# Document Revision History

| Revision | Date | Author | Description |
|---|---|---|---|
| 1.0 | 2016-01-15 | Yi-Chih Huang | First release |
| | | | |

# Table of Contents

## Lists of Tables

## Lists of Figures

# 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

MT6737, with integrated Bluetooth, FM, WLAN and GPS modules, is a highly integrated baseband platform incorporating modem, application processing and connectivity subsystems to enable LTE smart phone applications. MT6737 integrates a Quad-core ARM® Cortex-A53 MPCore™ operating up to 1.25GHz, an ARM® Cortex-R4 MCU and a powerful multi-standard video accelerator. MT6737 interfaces to LPDDR2/3 optimal performance and also supports booting from eMMC to minimize the overall BOM cost. In addition, an extensive set of interfaces are included to interface to cameras, touch-screen displays, and MMC/SD cards.

The application processor, a Quad-core ARM® Cortex-A53 MPCore™ which includes a NEON multimedia processing 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 while 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 included to provide advanced multimedia applications and services such as streaming audio and video, a multitude of decoders and encoders such as H.264 and MPEG-4. Audio supports include FR, HR, EFR, AMR FR, AMR HR and Wide-Band AMR vocoders, polyphonic ringtones and advanced audio functions such as echo cancellation, hands-free speakerphone operation and noise cancellation.

An ARM® Cortex-R4, DSP, and 2G and 3G coprocessors provide a powerful modem subsystem capable of supporting LTE Cat 4 (150Mbps), Category 24 (42 Mbps) HSDPA downlink and Category 7 (11 Mbps) HSUPA uplink data rates, Category 14 (2.8Mbps) TD-HSDPA downlink and Category 6 (2.2Mbps) TD-HSUPA uplink as well as Class 12 GPRS, EDGE.

MT6737 also embodies wireless communication device, including WLAN, Bluetooth and GPS. With four advanced radio technologies integrated into one single chip, MT6737 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. The small footprint with low-power consumption greatly reduces the PCB layout resource.

## 1.1 Highlighted Features Integrated in MT6737

- Quad-core ARM® Cortex-A53 MPCore™ operating at 1.25GHz
- LPDDR2/LPDDR3 up to 3GB, 640MHz
- LTE Cat 4 (150Mbps)
- Embedded connectivity system including WLAN/BT/FM/GPS
- Resolution up to HD (1280*720)
- OpenGL ES 3.0 3D graphic accelerator
- ISP supports 13MP@28fps.
- MPEG4 720p @ 30fps encoder
- Speech codec (FR, HR, EFR, AMR FR, AMR HR and Wide-Band AMR)

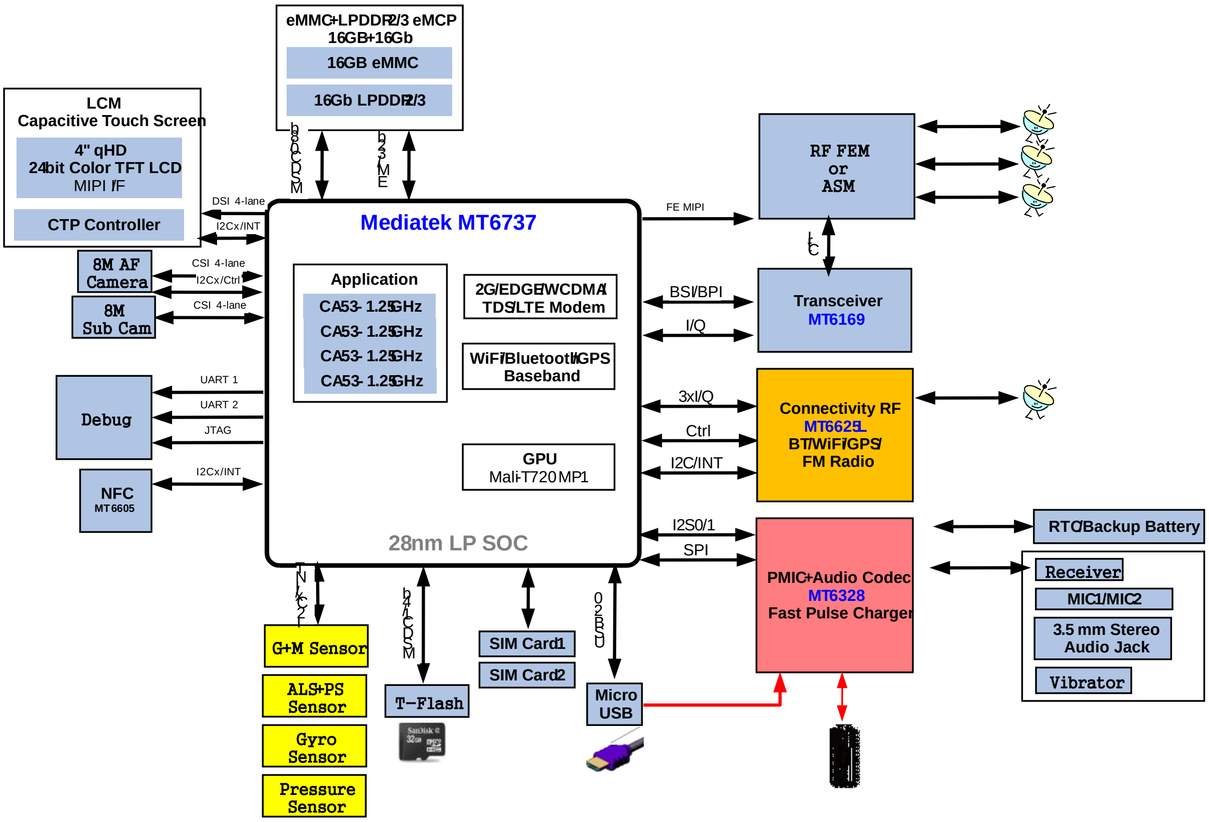


*Figure 1-1. High-level MT6737 functional block diagram*

## 1.2 Platform Features

- **General**
  - Smartphone two MCU subsystems architecture
  - eMMC bootloader

- **AP MCU subsystem**
  - Quad-core ARM® Cortex-A53 MPCoreTM operating at 1.25GHz
  - NEON multimedia processing engine with SIMDv2/VFPv4 ISA support
  - 32KB L1 I-cache and 32KB L1 D-cache
  - 256KB unified L2 cache
  - DVFS technology with adaptive operating voltage from 0.95V to 1.26V

- **MD MCU subsystem**
  - ARM® Cortex-R4 processor with maximum 600MHz operation frequency
  - 64KB I-cache, 64KB D-cache
  - 512KB TCM (tightly-coupled memory)
  - Coresonic DSP for running LTE modem tasks, with maximum 300MHz operation frequency
  - FD216 DSP for running modem/voice tasks, with maximum 250MHz 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 supported
  - Interface pins with RF and radio-related peripherals (antenna tuner, PA, ...)

- **External memory interface**
  - Supports LPDDR2/3 up to 3GB
  - 32-bit data bus width
  - Memory clock up to 640MHz
  - Supports self-refresh/partial self-refresh mode
  - Low-power operation
  - Programmable slew rate for memory controller’s IO pads
  - Supports dual rank memory device
  - Advanced bandwidth arbitration control

- **Security**
  - ARM® TrustZone® Security
  - Hardware Crypto Engine support

- **Peripherals**
  - USB2.0 high-speed OTG supporting 8 Tx and 8 Rx endpoints
  - eMMC5.0  supports
  - 4 UARTs for external devices and debugging interfaces
  - SPI for external devices
  - 4 I2C to control peripheral devices, e.g. CMOS image sensor, LCM or FM receiver module
  - I2S for connection with optional external hi-end audio codec
  - GPIOs
  - 2 sets of memory card controller supporting SD/SDHC/MS/MSPRO and MMC protocols
  - 1 set of programmable IRTX for remote control (Android API supported)
  - 1 set of IrDA
  - 6 sets of programmable PWM (1 channel reserved for IRTX software mode)

- **Operating conditions**
  - Core voltage: 1.05V

- Processor DVFS voltage : 0.95V~1.26V (Typ. 1.05V; Sleep mode 0.85V)
- Processor SRAM voltage : 1.05V~1.26V (Typ. 1.05V; Sleep mode 0.85V)
- GPU voltage : 1.05V/1.15V/1.25V
- I/O voltage: 1.8V/2.8V/3.3V
- Memory: 1.2V
- LCM interface: 1.8V
- Clock source: 26MHz, 32.768kHz

- **Package**
  - Type: VFBGA
  - 12.6mm*12.6mm
  - Height: 0.9mm maximum
  - Ball count: 641 balls
  - Ball pitch: 0.4mm

## 1.3 Modem Features

- **LTE**
  - FDD: up to 150 Mbps downlink, 50 Mbps uplink
  - TDD: up to 150 Mbps downlink, 50 Mbps uplink
  - 1.4 to 20 MHz RF bandwidth
  - 2*2 downlink SU-MIMO; 4*2 downlink SU-MIMO
  - IPv6, QoS
  - Inter-RAT capabilities with HSPA+, EDGE, and applicable backward-compatible modes
  - SNOW3G/ZUC cipher offload engine

- **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
  - Two 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
  - Supports TD-SCDMA Bands 34, 39 & 40 and Quad band GSM/EDGE
  - Circuit-switched voice and data, and 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 sidetone oscillation reduction
  - Digital sidetone generator with programmable gain
  - Two programmable acoustic compensation filters
  - GSM quad vocoders for adaptive multirate (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
  - Supports VAMOS (Voice services over Adaptive Multi-user channels on One Slot) technology in R9 spec

- **CDMA2000**
  - Supports CDMA2000 1xRTT (release 1 and Advanced ) and CDMA2000 HRPD/1xEV-DO Revision 1 and A.
  - Hybrid operation between 1x and HEPD
  - Simultaneous hybrid dual receiver (SHDR) support
  - 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.
  - Supports 1x Diversity

## 1.4 Multimedia Features

- **Display**
  - Supports portrait panel resolution up to HD (1280x720)
  - MIPI DSI interface (4 data lanes)
  - Embedded LCD gamma correction
  - Supports true colors
  - 4 overlay layers with per-pixel alpha channel and gamma table
  - Supports spatial and temporal dithering
  - Supports side-by-side format output to stereo 3D panel in both portrait and landscape modes
  - Supports color enhancement
  - Supports adaptive contrast enhancement
  - Supports image/video/graphic sharpness enhancement
  - Supports dynamic backlight scaling

- **Graphics**
  - OpenGL ES 1.1/2.0/3.0 3D graphic accelerator capable of processing 71.5M tri/sec and 650M pixel/sec @ 650MHz
  - OpenVG1.1 vector graphics accelerator

- **Image**
  - Integrated image signal processor supporting 13 MP
  - Supports video stabilization
  - Supports preference color adjustment
  - Supports noise reduction
  - Supports lens shading correction
  - Supports auto sensor defect pixel correction
  - Supports AE/AWB/AF
  - Supports edge enhancement (sharpness)
  - Supports face detection and visual tracking
  - Supports zero shutter delay image capture
  - Supports capturing full size image when recording video (up to 13M sensor)
  - Supports MIPI CSI-2 high-speed camera serial interface with 4 data lane (for main) + 4 data lane (for sub)
  - Hardware JPEG encoder: baseline encoding with 120M pixel/sec
  - YUV422/YUV420 color format and EXIF/JFIF format support
  - Hardware WebP decoder

- **Video**
  - H.264 decoder: baseline 1080p @ 30fps/40Mbps (HW: CBP)
  - H.264 decoder: main/high profile 1080p@30fps/40Mbps
  - Sorenson H.263/H.263 decoder: 1080p @ 30fps/40Mbps
  - MPEG-4 SP/ASP decoder: 1080p @ 30fps/40Mbps
  - DIVX4/DIVX5/DIVX6/DIVX HD/XVID decoder: 1080p @ 30fps/40Mbps
  - MPEG-4 encoder: Simple profile 720p@ 30fps
  - H.263 encoder: VGA@ 30fps (SW)

- **Audio**
  - Hardware sampling rates supported: 8kHz to 192kHz
  - Hardware Sample formats supported: 8-bit/16-bit/24-bit, Mono/Stereo
  - Hardware interfaces supported: DAI, I2S, PCM
  - Software 4-band IIR compensation filter to enhance loudspeaker responses
  - Software proprietary audio post-processing technologies: BesLoudness(MB-DRC), BesSurround, Android built-in post processing
  - Software audio encode: AMR-NB, AMR-WB, AAC, OGG, ADPCM
  - Software audio decode: WAV, MP3, MP2, AAC, AMR-NB, AMR-WB, MIDI, Vorbis, APE, AAC-plus v1, AAC-plus v2, FLAC, WMA, ADPCM

- **Speech (in DSP)**
  - Speech codec (FR, HR, EFR, AMR FR, AMR HR and Wide-Band AMR)
  - CTM
  - Noise reduction
  - Noise suppression
  - Noise cancellation
  - Dual-MIC noise cancellation
  - Echo cancellation
  - Echo suppression
  - Dual-MIC input
  - Digital MIC input

## 1.5 Connectivity Features

MT6737 includes four wireless connectivity functions:

- WLAN
- Bluetooth
- GPS
- FM Receiver

The RF parts of those four blocks are placed on chip MT6625L. With four advanced radio technologies integrated on one chip, MT6737/MT6625L 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 2.4 GHz Bluetooth, 2.4GHz/5GHz WLAN and 1.575 GHz 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
  - 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 (2.4GHz/5GHz) single stream 802.11 a/b/g/n MAC/BB/RF
  - 802.11 d/h/k compliant
  - Security: WFA WPA/WPA2 personal, WPS2.0, WAPI (hardware)
  - QoS: WFA WMM, WMM PS
  - 802.11n optional features: STBC, A-MPDU, Blk-Ack, RIFS, MCS Feedback, 20/40MHz coexistence (PCO), unscheduled PSMP
  - Supports 802.11w protected managed frames
  - Supports Wi-Fi HotSpot 2.0
  - Integrated 2.4GHz PA with max. 19dBm CCK output power and 5GHz PA with max. 17dBm OFDM 54Mbps output power
  - Typical Rx sensitivity with companion chip modem: -75dBm at 11g 54Mbps mode and -75.5dBm at 11a 54Mbps mode
  - Per packet TX power control

- **Bluetooth**
  - Bluetooth specification v2.1+EDR
  - Bluetooth specification 3.0+HS compliance
  - Bluetooth v4.0 Low Energy (LE)
  - Integrated PA with 6dBm (class 1) transmit power
  - Typical Rx sensitivity with companion chip modem: GFSK -92.5dBm, DQPSK -91.5dBm, 8-DPSK -86dBm
  - Best-in-class BT/Wi-Fi coexistence performance
  - Up to 4 piconets simultaneously with background inquiry/page scan
  - Supports Scatternet
  - Packet Loss Concealment (PLC) function for better voice quality
  - Low-power scan function to reduce power consumption in scan modes

- **GPS**
  - Supports GPS/QZSS/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 6dB design margin over 3GPP
  - Full A-GPS capability (E911/SUPL/EPO/HotStill)
  - Active interference cancellation for up to 8 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 operatio

## 1.6 General Descriptions

MediaTek's MT6737 hardware family is a highly integrated LTE System-on-Chip (SoC) which incorporates advanced features, e.g. LTE cat.4 modem, Quad-core ARM® Cortex-A53 MPCore™ operating at 1.25GHz, 3D graphics (OpenGL|ES 3.0), 13M camera ISP, LPDDR2 533/LPDDR3 640MHz and High-Definition 1080p video decoder. MT6737 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 28nm process, MT6737 is the brand-new generation smart phone SoC integrating MediaTek LTE cat.4 modem, 1.25GHz Quad-core ARM® Cortex-A53 MPCore™, 3D graphics and High-Definition 1080p video decoder.

***Rich in Features, High-Valued Product!***

To enrich the camera features, MT6737 equips a 13M camera ISP with advanced features e.g. auto focus, anti-handshake, auto sensor defect pixel correction, continuous video AF, face detection, burst shot, and panorama view.

***Incredible Browser Experience!***

The 1.25GHz Quad-core ARM® Cortex-A53 MPCore™ with NEON multimedia processing engine brings PC-like browser experiences and helps accelerate OpenGL|ES 3.0 3D Adobe Flash 10 rendering performance to an unbeatable level.

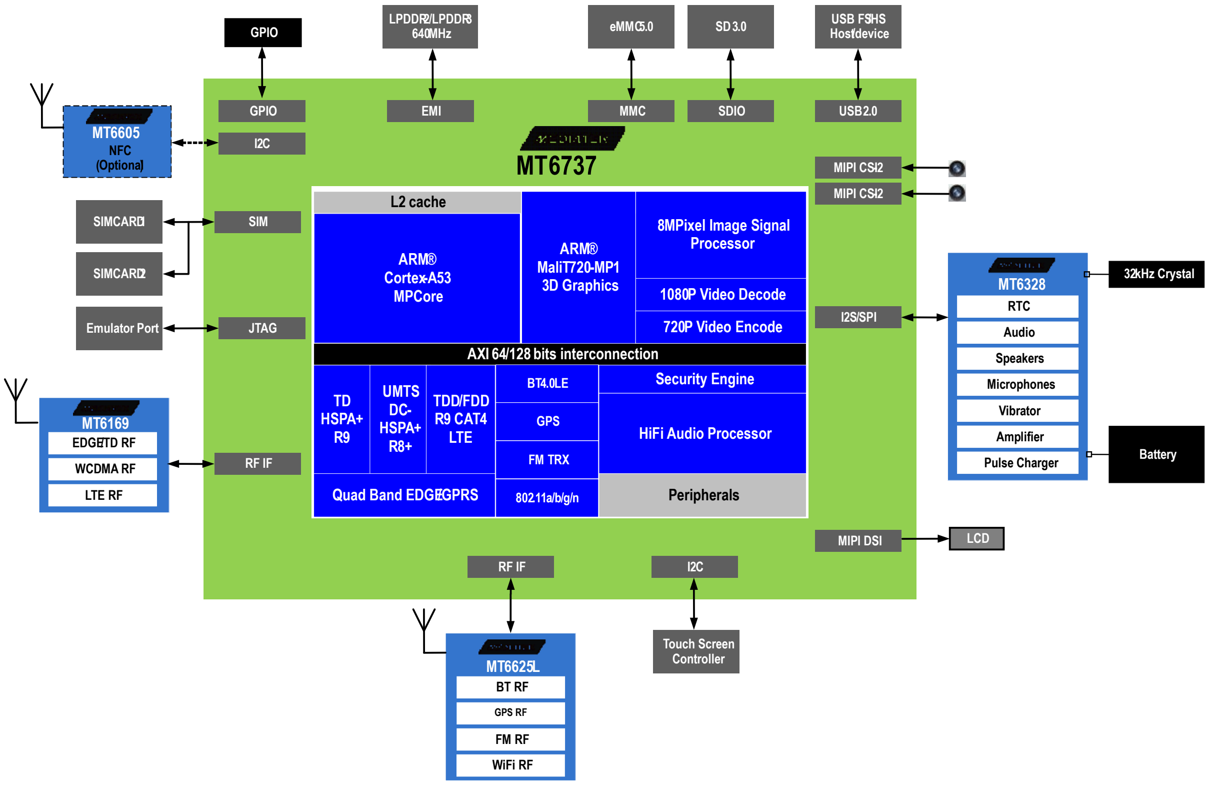


*Figure 1-2. Block diagram of MT6737*

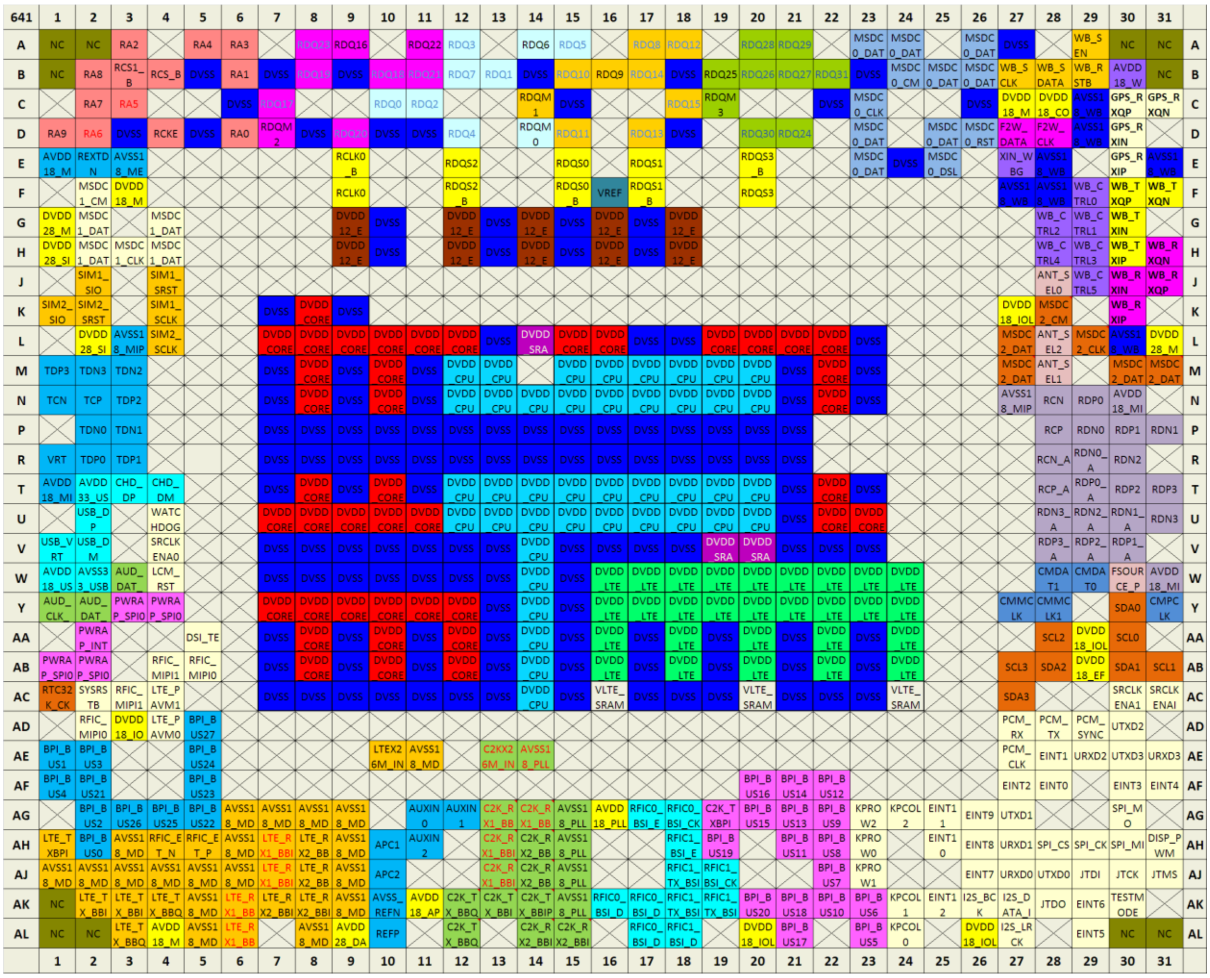
# 2 Product Description

## 2.1 Pin Description

### 2.1.1 Ball Map View

| 641 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | NC | NC | RA2 | | RA4 | RA3 | | RDQ17 | RDQ16 | | RDQ22 | RDQ0 | | RDQ6 | RDQ7 | | RDQ14 | RDQ15 | | RDQ27 | RDQ31 | | MSDC0_DAT | MSDC0_DAT | | MSDC0_DAT | DVSS | | WB_SEN | NC | NC | A |
| B | NC | RA8 | RCS1_B | RCS_B | DVSS | RA1 | DVSS | RDQ18 | DVSS | RDQ20 | RDQ19 | RDQ2 | RDQ4 | DVSS | RDQ8 | RDQ9 | RDQ11 | DVSS | RDQ25 | RDQ28 | RDQ30 | RDQ29 | DVSS | MSDC0_CM | MSDC0_DAT | MSDC0_DAT | WB_SCLK | WB_SDATA | WB_RSTB | AVDD18_W | NC | B |
| C | | RA7 | RA6 | | | DVSS | RDQ21 | | | RDQ1 | RDQ3 | | | RDQM1 | DVSS | | | RDQ13 | RDQM3 | | | DVSS | MSDC0_CLK | | | DVSS | DVDD18_M | DVDD18_CO | AVSS18_WB | GPS_RXQP | GPS_RXQN | C |
| D | RA9 | RA5 | DVSS | RCKE | DVSS | RA0 | RDQM2 | DVSS | RDQ23 | DVSS | DVSS | RDQ5 | | RDQM0 | RDQ10 | | RDQ12 | DVSS | | RDQ24 | RDQ26 | | MSDC0_DAT | | MSDC0_DAT | MSDC0_RST | F2W_DATA | F2W_CLK | AVSS18_WB | GPS_RXIN | | D |
| E | AVDD18_M | REXTDN | AVSS18_ME | | | | | | RCLK0_B | | | RDQS2 | | | RDQS0 | | RDQS1 | | | RDQS3_B | | | MSDC0_DAT | DVSS | MSDC0_DSL | | XIN_WBG | AVSS18_WB | | GPS_RXIP | AVSS18_WB | E |
| F | | MSDC1_CM | DVDD18_M | | | | | | RCLK0 | | | RDQS2_B | | | RDQS0_B | VREF | RDQS1_B | | | RDQS3 | | | | | | | AVSS18_WB | AVSS18_WB | WB_CTRL0 | WB_TXQP | WB_TXQN | F |
| G | DVDD28_M | MSDC1_DAT | | MSDC1_DAT | | | | | DVDD12_E | DVSS | | DVDD12_E | DVSS | DVDD12_E | DVSS | DVDD12_E | DVSS | DVDD12_E | | | | | | | | | | WB_CTRL2 | WB_CTRL1 | WB_TXIN | | G |
| H | DVDD28_SI | MSDC1_DAT | MSDC1_CLK | MSDC1_DAT | | | | | DVDD12_E | DVSS | | DVDD12_E | DVSS | DVDD12_E | DVSS | DVDD12_E | DVSS | DVDD12_E | | | | | | | | | | WB_CTRL4 | WB_CTRL3 | WB_TXIP | WB_RXQN | H |
| J | | SIM1_SIO | | SIM1_SRST | | | | | | | | | | | | | | | | | | | | | | | | ANT_SEL0 | WB_CTRL5 | WB_RXIN | WB_RXQP | J |
| K | SIM2_SIO | SIM2_SRST | | SIM1_SCLK | | | DVSS | DVDD_CORE | DVSS | | | | | | | | | | | | | | | | | | DVDD18_IOL | MSDC2_CM | | WB_RXIP | | K |
| L | | DVDD28_SI | AVSS18_MIP | SIM2_SCLK | | | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVSS | DVDD_SRA | DVDD_CORE | DVDD_CORE | DVSS | DVSS | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVSS | | | | MSDC2_DAT | ANT_SEL2 | MSDC2_CLK | AVSS18_WB | DVDD28_M | L |
| M | TDP3 | TDN3 | TDN2 | | | | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CPU | DVDD_CPU | | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVSS | DVDD_CORE | DVSS | | | | MSDC2_DAT | ANT_SEL1 | | MSDC2_DAT | MSDC2_DAT | M |
| N | TCN | TCP | TDP2 | | | | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVSS | DVDD_CORE | DVSS | | | | AVSS18_MIP | RCN | RDP0 | AVDD18_MI | | N |
| P | | TDN0 | TDN1 | | | | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | | | | | | | RCP | RDN0 | RDP1 | RDN1 | P |
| R | VRT | TDP0 | TDP1 | | | | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | | | | | | | RCN_A | RDN0_A | RDN2 | | R |
| T | AVDD18_MI | AVDD33_US | CHD_DP | CHD_DM | | | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVSS | DVDD_CORE | DVSS | | | | | RCP_A | RDP0_A | RDP2 | RDP3 | T |
| U | | USB_DP | | WATCHDOG | | | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVSS | DVDD_CORE | DVDD_CORE | | | | | RDN3_A | RDN2_A | RDN1_A | RDN3 | U |
| V | USB_VRT | USB_DM | | SRCLKENA0 | | | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVDD_CPU | DVSS | DVSS | DVSS | DVSS | DVDD_SRA | DVDD_SRA | DVSS | DVSS | DVSS | | | | | RDP3_A | RDP2_A | RDP1_A | | V |
| W | AVDD18_US | AVSS33_USB | AUD_DAT_ | LCM_RST | | | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVDD_CPU | DVSS | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | | | | CMDAT1 | CMDAT0 | FSOURCE_P | AVDD18_MI | W |
| Y | AUD_CLK_ | AUD_DAT_ | PWRAP_SPI0 | PWRAP_SPI0 | | | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVSS | DVDD_CPU | DVSS | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | | | CMMCLK | CMMCLK1 | | SDA0 | CMPCLK | Y |
| AA | | PWRAP_INT | | | DSI_TE | | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CPU | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | | | | SCL2 | DVDD18_IOL | SCL0 | | AA |
| AB | PWRAP_SPI0 | PWRAP_SPI0 | | RFIC_MIPI1 | RFIC_MIPI0 | | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CPU | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | | | SCL3 | SDA2 | DVDD18_EF | SDA1 | SCL1 | AB |
| AC | RTC32K_CK | SYSRSTB | RFIC_MIPI1 | LTE_PAVM1 | | | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVDD_CPU | DVSS | VLTE_SRAM | DVSS | DVSS | DVSS | VLTE_SRAM | DVSS | DVSS | DVSS | VLTE_SRAM | | | SDA3 | | | SRCLKENA1 | SRCLKENAI | AC |
| AD | | RFIC_MIPI0 | DVDD18_IO | LTE_PAVM0 | BPI_BUS27 | | | | | | | | | | | | | | | | | | | | | | PCM_RX | PCM_TX | PCM_SYNC | UTXD2 | | AD |
| AE | BPI_BUS1 | BPI_BUS3 | | | BPI_BUS24 | | | | | LTEX26M_IN | AVSS18_MD | | C2KX26M_IN | AVSS18_PLL | | | | | | | | | | | | | PCM_CLK | EINT1 | URXD2 | UTXD3 | URXD3 | AE |
| AF | BPI_BUS4 | BPI_BUS21 | | | BPI_BUS23 | | | | | | | | | | | | | | | BPI_BUS16 | BPI_BUS14 | BPI_BUS12 | | | | | EINT2 | EINT0 | | EINT3 | EINT4 | AF |
| AG | | BPI_BUS2 | BPI_BUS26 | BPI_BUS25 | BPI_BUS22 | AVSS18_MD | AVSS18_MD | AVSS18_MD | AVSS18_MD | | AUXIN0 | AUXIN1 | C2K_RX1_BB | C2K_RX1_BB | AVSS18_PLL | AVDD18_PLL | RFIC0_BSI_E | RFIC0_BSI_CK | C2K_TXBPI | BPI_BUS15 | BPI_BUS13 | BPI_BUS9 | KPROW2 | KPCOL2 | EINT11 | EINT9 | UTXD1 | | | SPI_MO | | AG |
| AH | LTE_TXBPI | BPI_BUS0 | AVSS18_MD | RFIC_ET_N | RFIC_ET_P | AVSS18_MD | LTE_RX1_BBI | LTE_RX2_BB | AVSS18_MD | APC1 | AUXIN2 | | C2K_RX1_BBI | C2K_RX2_BB | AVSS18_PLL | | | RFIC1_BSI_E | BPI_BUS19 | | BPI_BUS11 | BPI_BUS8 | KPROW0 | | EINT10 | EINT8 | URXD1 | SPI_CS | SPI_CK | SPI_MI | DISP_PWM | AH |
| AJ | AVSS18_MD | AVSS18_MD | AVSS18_MD | AVSS18_MD | AVSS18_MD | AVSS18_MD | LTE_RX1_BBI | LTE_RX2_BB | AVSS18_MD | APC2 | | | C2K_RX1_BBI | C2K_RX2_BB | AVSS18_PLL | | | RFIC1_TX_BSI | RFIC1_BSI_CK | | | BPI_BUS7 | KPROW1 | | | EINT7 | URXD0 | UTXD0 | JTDI | JTCK | JTMS | AJ |
| AK | NC | LTE_TX_BBI | LTE_TX_BBI | LTE_TX_BBQ | AVSS18_MD | LTE_RX1_BB | LTE_RX2_BBI | LTE_RX2_BBI | AVSS18_MD | AVSS_REFN | AVDD18_AP | C2K_TX_BBQ | C2K_TX_BBI | C2K_TX_BBIP | AVSS18_PLL | RFIC0_BSI_D | RFIC0_BSI_D | RFIC1_TX_BSI | RFIC1_TX_BSI | BPI_BUS20 | BPI_BUS18 | BPI_BUS10 | BPI_BUS6 | KPCOL1 | EINT12 | I2S_BCK | I2S_DATA_I | JTDO | EINT6 | TESTMODE | | AK |
| AL | NC | NC | LTE_TX_BBQ | AVDD18_M | AVSS18_MD | LTE_RX1_BB | | AVSS18_MD | AVDD28_DA | REFP | | C2K_TX_BBQ | | C2K_RX2_BBI | C2K_RX2_BBI | | RFIC0_BSI_D | RFIC1_BSI_D | | DVDD18_IOL | BPI_BUS17 | | BPI_BUS5 | KPCOL0 | | DVDD18_IOL | I2S_LRCK | | EINT5 | NC | NC | AL |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | |

*Figure 2-1. Ball map view for LPDDR3*



| 641 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | NC | NC | RA2 | ╳ | RA4 | RA3 | ╳ | RDQ23 | RDQ16 | ╳ | RDQ22 | RDQ3 | ╳ | RDQ6 | RDQ5 | ╳ | RDQ8 | RDQ12 | ╳ | RDQ28 | RDQ29 | ╳ | MSDC0_DAT | MSDC0_DAT | ╳ | MSDC0_DAT | DVSS | ╳ | WB_SEN | NC | NC | A |
| B | NC | RA8 | RCS1_B | RCS_B | DVSS | RA1 | DVSS | RDQ19 | DVSS | RDQ18 | RDQ21 | RDQ7 | RDQ1 | DVSS | RDQ10 | RDQ9 | RDQ14 | DVSS | RDQ25 | RDQ26 | RDQ27 | RDQ31 | DVSS | MSDC0_CM | MSDC0_DAT | MSDC0_DAT | WB_SCLK | WB_SDATA | WB_RSTB | AVDD18_W | NC | B |
| C | ╳ | RA7 | RA5 | ╳ | ╳ | DVSS | RDQ17 | ╳ | ╳ | RDQ0 | RDQ2 | ╳ | ╳ | RDQM1 | DVSS | ╳ | ╳ | RDQ15 | RDQM3 | ╳ | ╳ | DVSS | MSDC0_CLK | ╳ | ╳ | DVSS | DVDD18_M | DVDD18_CO | AVSS18_WB | GPS_RXQP | GPS_RXQN | C |
| D | RA9 | RA6 | DVSS | RCKE | DVSS | RA0 | RDQM2 | DVSS | RDQ20 | DVSS | DVSS | RDQ4 | ╳ | RDQM0 | RDQ11 | ╳ | RDQ13 | DVSS | ╳ | RDQ30 | RDQ24 | ╳ | MSDC0_DAT | ╳ | MSDC0_DAT | MSDC0_RST | F2W_DATA | F2W_CLK | AVSS18_WB | GPS_RXIN | ╳ | D |
| E | AVDD18_M | REXTDN | AVSS18_ME | ╳ | ╳ | ╳ | ╳ | ╳ | RCLK0_B | ╳ | ╳ | RDQS2 | ╳ | ╳ | RDQS0 | ╳ | RDQS1 | ╳ | ╳ | RDQS3_B | ╳ | ╳ | MSDC0_DAT | DVSS | MSDC0_DSL | ╳ | XIN_WBG | AVSS18_WB | ╳ | GPS_RXIP | AVSS18_WB | E |
| F | ╳ | MSDC1_CM | DVDD18_M | ╳ | ╳ | ╳ | ╳ | ╳ | RCLK0 | ╳ | ╳ | RDQS2_B | ╳ | ╳ | RDQS0_B | VREF | RDQS1_B | ╳ | ╳ | RDQS3 | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | AVSS18_WB | AVSS18_WB | WB_CTRL0 | WB_TXQP | WB_TXQN | F |
| G | DVDD28_M | MSDC1_DAT | ╳ | MSDC1_DAT | ╳ | ╳ | ╳ | ╳ | DVDD12_E | DVSS | ╳ | DVDD12_E | DVSS | DVDD12_E | DVSS | DVDD12_E | DVSS | DVDD12_E | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | WB_CTRL2 | WB_CTRL1 | WB_TXIN | ╳ | G |
| H | DVDD28_SI | MSDC1_DAT | MSDC1_CLK | MSDC1_DAT | ╳ | ╳ | ╳ | ╳ | DVDD12_E | DVSS | ╳ | DVDD12_E | DVSS | DVDD12_E | DVSS | DVDD12_E | DVSS | DVDD12_E | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | WB_CTRL4 | WB_CTRL3 | WB_TXIP | WB_RXQN | H |
| J | ╳ | SIM1_SIO | ╳ | SIM1_SRST | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ANT_SEL0 | WB_CTRL5 | WB_RXIN | WB_RXQP | J |
| K | SIM2_SIO | SIM2_SRST | ╳ | SIM1_SCLK | ╳ | ╳ | DVSS | DVDD_CORE | DVSS | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | DVDD18_IOL | MSDC2_CM | ╳ | WB_RXIP | ╳ | K |
| L | ╳ | DVDD28_SI | AVSS18_MIP | SIM2_SCLK | ╳ | ╳ | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVSS | DVDD_SRA | DVDD_CORE | DVDD_CORE | DVSS | DVSS | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVSS | ╳ | ╳ | ╳ | MSDC2_DAT | ANT_SEL2 | MSDC2_CLK | AVSS18_WB | DVDD28_M | L |
| M | TDP3 | TDN3 | TDN2 | ╳ | ╳ | ╳ | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CPU | DVDD_CPU | ╳ | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVSS | DVDD_CORE | DVSS | ╳ | ╳ | ╳ | MSDC2_DAT | ANT_SEL1 | ╳ | MSDC2_DAT | MSDC2_DAT | M |
| N | TCN | TCP | TDP2 | ╳ | ╳ | ╳ | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVSS | DVDD_CORE | DVSS | ╳ | ╳ | ╳ | AVSS18_MIP | RCN | RDP0 | AVDD18_MI | ╳ | N |
| P | ╳ | TDN0 | TDN1 | ╳ | ╳ | ╳ | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | RCP | RDN0 | RDP1 | RDN1 | P |
| R | VRT | TDP0 | TDP1 | ╳ | ╳ | ╳ | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | RCN_A | RDN0_A | RDN2 | ╳ | R |
| T | AVDD18_MI | AVDD33_US | CHD_DP | CHD_DM | ╳ | ╳ | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVSS | DVDD_CORE | DVSS | ╳ | ╳ | ╳ | ╳ | RCP_A | RDP0_A | RDP2 | RDP3 | T |
| U | ╳ | USB_DP | ╳ | WATCHDOG | ╳ | ╳ | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVDD_CPU | DVSS | DVDD_CORE | DVDD_CORE | ╳ | ╳ | ╳ | ╳ | RDN3_A | RDN2_A | RDN1_A | RDN3 | U |
| V | USB_VRT | USB_DM | ╳ | SRCLKENA0 | ╳ | ╳ | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVDD_CPU | DVSS | DVSS | DVSS | DVSS | DVDD_SRA | DVDD_SRA | DVSS | DVSS | DVSS | ╳ | ╳ | ╳ | ╳ | RDP3_A | RDP2_A | RDP1_A | ╳ | V |
| W | AVDD18_US | AVSS33_USB | AUD_DAT_ | LCM_RST | ╳ | ╳ | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVDD_CPU | DVSS | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | ╳ | ╳ | ╳ | CMDAT1 | CMDAT0 | FSOURCE_P | AVDD18_MI | W |
| Y | AUD_CLK_ | AUD_DAT_ | PWRAP_SPI0 | PWRAP_SPI0 | ╳ | ╳ | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVDD_CORE | DVSS | DVDD_CPU | DVSS | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | DVDD_LTE | ╳ | ╳ | CMMCLK | CMMCLK1 | ╳ | SDA0 | CMPCLK | Y |
| AA | ╳ | PWRAP_INT | ╳ | ╳ | DSI_TE | ╳ | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CPU | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | ╳ | ╳ | ╳ | SCL2 | DVDD18_IOL | SCL0 | ╳ | AA |
| AB | PWRAP_SPI0 | PWRAP_SPI0 | ╳ | RFIC_MIPI1 | RFIC_MIPI0 | ╳ | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CORE | DVSS | DVDD_CPU | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | DVSS | DVDD_LTE | ╳ | ╳ | SCL3 | SDA2 | DVDD18_EF | SDA1 | SCL1 | AB |
| AC | RTC32K_CK | SYSRSTB | RFIC_MIPI1 | LTE_PAVM1 | ╳ | ╳ | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVSS | DVDD_CPU | DVSS | VLTE_SRAM | DVSS | DVSS | DVSS | VLTE_SRAM | DVSS | DVSS | DVSS | VLTE_SRAM | ╳ | ╳ | SDA3 | ╳ | ╳ | SRCLKENA1 | SRCLKENAI | AC |
| AD | ╳ | RFIC_MIPI0 | DVDD18_IO | LTE_PAVM0 | BPI_BUS27 | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | PCM_RX | PCM_TX | PCM_SYNC | UTXD2 | ╳ | AD |
| AE | BPI_BUS1 | BPI_BUS3 | ╳ | ╳ | BPI_BUS24 | ╳ | ╳ | ╳ | ╳ | LTEX26M_IN | AVSS18_MD | ╳ | C2KX26M_IN | AVSS18_PLL | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | PCM_CLK | EINT1 | URXD2 | UTXD3 | URXD3 | AE |
| AF | BPI_BUS4 | BPI_BUS21 | ╳ | ╳ | BPI_BUS23 | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | ╳ | BPI_BUS16 | BPI_BUS14 | BPI_BUS12 | ╳ | ╳ | ╳ | ╳ | EINT2 | EINT0 | ╳ | EINT3 | EINT4 | AF |
| AG | ╳ | BPI_BUS2 | BPI_BUS26 | BPI_BUS25 | BPI_BUS22 | AVSS18_MD | AVSS18_MD | AVSS18_MD | AVSS18_MD | ╳ | AUXIN0 | AUXIN1 | C2K_RX1_BB | C2K_RX1_BB | AVSS18_PLL | AVDD18_PLL | RFIC0_BSI_E | RFIC0_BSI_CK | C2K_TXBPI | BPI_BUS15 | BPI_BUS13 | BPI_BUS9 | KPROW2 | KPCOL2 | EINT11 | EINT9 | UTXD1 | ╳ | ╳ | SPI_MO | ╳ | AG |
| AH | LTE_TXBPI | BPI_BUS0 | AVSS18_MD | RFIC_ET_N | RFIC_ET_P | AVSS18_MD | LTE_RX1_BBI | LTE_RX2_BB | AVSS18_MD | APC1 | AUXIN2 | ╳ | C2K_RX1_BBI | C2K_RX2_BB | AVSS18_PLL | ╳ | ╳ | RFIC1_BSI_E | BPI_BUS19 | ╳ | BPI_BUS11 | BPI_BUS8 | KPROW0 | ╳ | EINT10 | EINT8 | URXD1 | SPI_CS | SPI_CK | SPI_MI | DISP_PWM | AH |
| AJ | AVSS18_MD | AVSS18_MD | AVSS18_MD | AVSS18_MD | AVSS18_MD | AVSS18_MD | LTE_RX1_BBI | LTE_RX2_BB | AVSS18_MD | APC2 | ╳ | ╳ | C2K_RX1_BBI | C2K_RX2_BB | AVSS18_PLL | ╳ | ╳ | RFIC1_TX_BSI | RFIC1_BSI_CK | ╳ | ╳ | BPI_BUS7 | KPROW1 | ╳ | ╳ | EINT7 | URXD0 | UTXD0 | JTDI | JTCK | JTMS | AJ |
| AK | NC | LTE_TX_BBI | LTE_TX_BBI | LTE_TX_BBQ | AVSS18_MD | LTE_RX1_BB | LTE_RX2_BBI | LTE_RX2_BBI | AVSS18_MD | AVSS_REFN | AVDD18_AP | C2K_TX_BBQ | C2K_TX_BBI | C2K_TX_BBIP | AVSS18_PLL | RFIC0_BSI_D | RFIC0_BSI_D | RFIC1_TX_BSI | RFIC1_TX_BSI | BPI_BUS20 | BPI_BUS18 | BPI_BUS10 | BPI_BUS6 | KPCOL1 | EINT12 | I2S_BCK | I2S_DATA_I | JTDO | EINT6 | TESTMODE | ╳ | AK |
| AL | NC | NC | LTE_TX_BBQ | AVDD18_M | AVSS18_MD | LTE_RX1_BB | ╳ | AVSS18_MD | AVDD28_DA | REFP | ╳ | C2K_TX_BBQ | ╳ | C2K_RX2_BBI | C2K_RX2_BBI | ╳ | RFIC0_BSI_D | RFIC1_BSI_D | ╳ | DVDD18_IOL | BPI_BUS17 | ╳ | BPI_BUS5 | KPCOL0 | ╳ | DVDD18_IOL | I2S_LRCK | ╳ | EINT5 | NC | NC | AL |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | |

*Figure 2-2. Ball map view for LPDDR2*

### 2.1.2 Pin Coordinate

*Table 2-1. Pin coordinate (using LPDDR3)*

| Ball Loc. | Ball name | Ball Loc. | Ball name | Ball Loc. | Ball name |
|---|---|---|---|---|---|
| A1 | NC | M23 | DVSS | AA28 | SCL2 |
| A2 | NC | M27 | NC | AA29 | DVDD18_IOLB |
| A3 | RA2 | M28 | ANT_SEL1 | AA30 | SCL0 |
| A5 | RA4 | M30 | NC | AB1 | PWRAP_SPI0_MI |
| A6 | RA3 | M31 | NC | AB2 | PWRAP_SPI0_MO |
| A8 | RDQ17 | N1 | TCN | AB4 | RFIC_MIPI1_SDATA |
| A9 | RDQ16 | N2 | TCP | AB5 | RFIC_MIPI0_SDATA |
| A11 | RDQ22 | N3 | TDP2 | AB7 | DVSS |
| A12 | RDQ0 | N7 | DVSS | AB8 | DVDD_CORE |
| A14 | RDQ6 | N8 | DVDD_CORE | AB9 | DVSS |
| A15 | RDQ7 | N9 | DVSS | AB10 | DVDD_CORE |
| A17 | RDQ14 | N10 | DVDD_CORE | AB11 | DVSS |
| A18 | RDQ15 | N11 | DVSS | AB12 | DVDD_CORE |

| Ball Loc. | Ball name | Ball Loc. | Ball name | Ball Loc. | Ball name |
|---|---|---|---|---|---|
| A20 | RDQ27 | N12 | DVDD_CPU | AB13 | DVSS |
| A21 | RDQ31 | N13 | DVDD_CPU | AB14 | DVDD_CPU |
| A23 | MSDC0_DAT4 | N14 | DVDD_CPU | AB15 | DVSS |
| A24 | MSDC0_DAT5 | N15 | DVDD_CPU | AB16 | DVDD_LTE |
| A26 | MSDC0_DAT0 | N16 | DVDD_CPU | AB17 | DVSS |
| A27 | DVSS | N17 | DVDD_CPU | AB18 | DVDD_LTE |
| A29 | WB_SEN | N18 | DVDD_CPU | AB19 | DVSS |
| A30 | NC | N19 | DVDD_CPU | AB20 | DVDD_LTE |
| A31 | NC | N20 | DVDD_CPU | AB21 | DVSS |
| B1 | NC | N21 | DVSS | AB22 | DVDD_LTE |
| B2 | RA8 | N22 | DVDD_CORE | AB23 | DVSS |
| B3 | RCS1_B | N23 | DVSS | AB24 | DVDD_LTE |
| B4 | RCS_B | N27 | AVSS18_MIPIRX0 | AB27 | SCL3 |
| B5 | DVSS | N28 | RCN | AB28 | SDA2 |
| B6 | RA1 | N29 | RDP0 | AB29 | DVDD18_EFUSE |
| B7 | DVSS | N30 | AVDD18_MIPIRX0 | AB30 | SDA1 |
| B8 | RDQ18 | P2 | TDN0 | AB31 | SCL1 |
| B9 | DVSS | P3 | TDN1 | AC1 | RTC32K_CK |
| B10 | RDQ20 | P7 | DVSS | AC2 | SYSRSTB |
| B11 | RDQ19 | P8 | DVSS | AC3 | RFIC_MIPI1_SCLK |
| B12 | RDQ2 | P9 | DVSS | AC4 | LTE_PAVM1 |
| B13 | RDQ4 | P10 | DVSS | AC7 | DVSS |
| B14 | DVSS | P11 | DVSS | AC8 | DVSS |
| B15 | RDQ8 | P12 | DVSS | AC9 | DVSS |
| B16 | RDQ9 | P13 | DVSS | AC10 | DVSS |
| B17 | RDQ11 | P14 | DVSS | AC11 | DVSS |
| B18 | DVSS | P15 | DVSS | AC12 | DVSS |
| B19 | RDQ25 | P16 | DVSS | AC13 | DVSS |
| B20 | RDQ28 | P17 | DVSS | AC14 | DVDD_CPU |
| B21 | RDQ30 | P18 | DVSS | AC15 | DVSS |
| B22 | RDQ29 | P19 | DVSS | AC16 | VLTE_SRAM |
| B23 | DVSS | P20 | DVSS | AC17 | DVSS |
| B24 | MSDC0_CMD | P21 | DVSS | AC18 | DVSS |
| B25 | MSDC0_DAT2 | P28 | RCP | AC19 | DVSS |
| B26 | MSDC0_DAT1 | P29 | RDN0 | AC20 | VLTE_SRAM |
| B27 | WB_SCLK | P30 | RDP1 | AC21 | DVSS |
| B28 | WB_SDATA | P31 | RDN1 | AC22 | DVSS |
| B29 | WB_RSTB | R1 | VRT | AC23 | DVSS |
| B30 | AVDD18_WBG | R2 | TDP0 | AC24 | VLTE_SRAM |
| B31 | NC | R3 | TDP1 | AC27 | SDA3 |
| C2 | RA7 | R7 | DVSS | AC30 | SRCLKENA1 |
| C3 | RA6 | R8 | DVSS | AC31 | SRCLKENAI |
| C6 | DVSS | R9 | DVSS | AD2 | RFIC_MIPI0_SCLK |
| C7 | RDQ21 | R10 | DVSS | AD3 | DVDD18_IORB |
| C10 | RDQ1 | R11 | DVSS | AD4 | LTE_PAVM0 |
| C11 | RDQ3 | R12 | DVSS | AD5 | BPI_BUS27 |
| C14 | RDQM1 | R13 | DVSS | AD27 | PCM_RX |
| C15 | DVSS | R14 | DVSS | AD28 | PCM_TX |
| C18 | RDQ13 | R15 | DVSS | AD29 | PCM_SYNC |

| Ball Loc. | Ball name | Ball Loc. | Ball name | Ball Loc. | Ball name |
|---|---|---|---|---|---|
| C19 | RDQM3 | R16 | DVSS | AD30 | UTXD2 |
| C22 | DVSS | R17 | DVSS | AE1 | BPI_BUS1 |
| C23 | MSDC0_CLK | R18 | DVSS | AE2 | BPI_BUS3 |
| C26 | DVSS | R19 | DVSS | AE5 | BPI_BUS24 |
| C27 | DVDD18_MC0 | R20 | DVSS | AE10 | LTEX26M_IN |
| C28 | DVDD18_CONN | R21 | DVSS | AE11 | AVSS18_MD |
| C29 | AVSS18_WBG | R28 | RCN_A | AE13 | C2KX26M_IN |
| C30 | GPS_RXQP | R29 | RDN0_A | AE14 | AVSS18_PLLGP |
| C31 | GPS_RXQN | R30 | RDN2 | AE27 | PCM_CLK |
| D1 | RA9 | T1 | AVDD18_MIPITX | AE28 | EINT1 |
| D2 | RA5 | T2 | AVDD33_USB | AE29 | URXD2 |
| D3 | DVSS | T3 | CHD_DP | AE30 | UTXD3 |
| D4 | RCKE | T4 | CHD_DM | AE31 | URXD3 |
| D5 | DVSS | T7 | DVSS | AF1 | BPI_BUS4 |
| D6 | RA0 | T8 | DVDD_CORE | AF2 | BPI_BUS21 |
| D7 | RDQM2 | T9 | DVSS | AF5 | BPI_BUS23 |
| D8 | DVSS | T10 | DVDD_CORE | AF20 | BPI_BUS16 |
| D9 | RDQ23 | T11 | DVSS | AF21 | BPI_BUS14 |
| D10 | DVSS | T12 | DVDD_CPU | AF22 | BPI_BUS12 |
| D11 | DVSS | T13 | DVDD_CPU | AF27 | EINT2 |
| D12 | RDQ5 | T14 | DVDD_CPU | AF28 | EINT0 |
| D14 | RDQM0 | T15 | DVDD_CPU | AF30 | EINT3 |
| D15 | RDQ10 | T16 | DVDD_CPU | AF31 | EINT4 |
| D17 | RDQ12 | T17 | DVDD_CPU | AG2 | BPI_BUS2 |
| D18 | DVSS | T18 | DVDD_CPU | AG3 | BPI_BUS26 |
| D20 | RDQ24 | T19 | DVDD_CPU | AG4 | BPI_BUS25 |
| D21 | RDQ26 | T20 | DVDD_CPU | AG5 | BPI_BUS22 |
| D23 | MSDC0_DAT6 | T21 | DVSS | AG6 | AVSS18_MD |
| D25 | MSDC0_DAT7 | T22 | DVDD_CORE | AG7 | AVSS18_MD |
| D26 | MSDC0_RSTB | T23 | DVSS | AG8 | AVSS18_MD |
| D27 | F2W_DATA | T28 | RCP_A | AG9 | AVSS18_MD |
| D28 | F2W_CLK | T29 | RDP0_A | AG11 | AUXIN0 |
| D29 | AVSS18_WBG | T30 | RDP2 | AG12 | AUXIN1 |
| D30 | GPS_RXIN | T31 | RDP3 | AG13 | C2K_RX1_BBQP |
| E1 | AVDD18_MEMPLL | U2 | USB_DP | AG14 | C2K_RX1_BBQN |
| E2 | REXTDN | U4 | WATCHDOG | AG15 | AVSS18_PLLGP |
| E3 | AVSS18_MEMPLL | U7 | DVDD_CORE | AG16 | AVDD18_PLLGP |
| E9 | RCLK0_B | U8 | DVDD_CORE | AG17 | RFIC0_BSI_EN |
| E12 | RDQS2 | U9 | DVDD_CORE | AG18 | RFIC0_BSI_CK |
| E15 | RDQS0 | U10 | DVDD_CORE | AG19 | C2K_TXBPI |
| E17 | RDQS1 | U11 | DVDD_CORE | AG20 | BPI_BUS15 |
| E20 | RDQS3_B | U12 | DVDD_CPU | AG21 | BPI_BUS13 |
| E23 | MSDC0_DAT3 | U13 | DVDD_CPU | AG22 | BPI_BUS9 |
| E24 | DVSS | U14 | DVDD_CPU | AG23 | KPROW2 |
| E25 | MSDC0_DSL | U15 | DVDD_CPU | AG24 | KPCOL2 |
| E27 | XIN_WBG | U16 | DVDD_CPU | AG25 | EINT11 |
| E28 | AVSS18_WBG | U17 | DVDD_CPU | AG26 | EINT9 |
| E30 | GPS_RXIP | U18 | DVDD_CPU | AG27 | UTXD1 |
| E31 | AVSS18_WBG | U19 | DVDD_CPU | AG30 | SPI_MO |

| Ball Loc. | Ball name | Ball Loc. | Ball name | Ball Loc. | Ball name |
|---|---|---|---|---|---|
| F2 | MSDC1_CMD | U20 | DVDD_CPU | AH1 | LTE_TXBPI |
| F3 | DVDD18_MC1 | U21 | DVSS | AH2 | BPI_BUS0 |
| F9 | RCLK0 | U22 | DVDD_CORE | AH3 | AVSS18_MD |
| F12 | RDQS2_B | U23 | DVDD_CORE | AH4 | RFIC_ET_N |
| F15 | RDQS0_B | U28 | RDN3_A | AH5 | RFIC_ET_P |
| F16 | VREF | U29 | RDN2_A | AH6 | AVSS18_MD |
| F17 | RDQS1_B | U30 | RDN1_A | AH7 | LTE_RX1_BBIP |
| F20 | RDQS3 | U31 | RDN3 | AH8 | LTE_RX2_BBQN |
| F27 | AVSS18_WBG | V1 | USB_VRT | AH9 | AVSS18_MD |
| F28 | AVSS18_WBG | V2 | USB_DM | AH10 | APC1 |
| F29 | WB_CTRL0 | V4 | SRCLKENA0 | AH11 | AUXIN2 |
| F30 | WB_TXQP | V7 | DVSS | AH13 | C2K_RX1_BBIN |
| F31 | WB_TXQN | V8 | DVSS | AH14 | C2K_RX2_BBQP |
| G1 | DVDD28_MC1 | V9 | DVSS | AH15 | AVSS18_PLLGP |
| G2 | MSDC1_DAT3 | V10 | DVSS | AH18 | RFIC1_BSI_EN |
| G4 | MSDC1_DAT0 | V11 | DVSS | AH19 | BPI_BUS19 |
| G9 | DVDD12_EMI | V12 | DVSS | AH21 | BPI_BUS11 |
| G10 | DVSS | V13 | DVSS | AH22 | BPI_BUS8 |
| G12 | DVDD12_EMI | V14 | DVDD_CPU | AH23 | KPROW0 |
| G13 | DVSS | V15 | DVSS | AH25 | EINT10 |
| G14 | DVDD12_EMI | V16 | DVSS | AH26 | EINT8 |
| G15 | DVSS | V17 | DVSS | AH27 | URXD1 |
| G16 | DVDD12_EMI | V18 | DVSS | AH28 | SPI_CS |
| G17 | DVSS | V19 | DVDD_SRAM | AH29 | SPI_CK |
| G18 | DVDD12_EMI | V20 | DVDD_SRAM | AH30 | SPI_MI |
| G28 | WB_CTRL2 | V21 | DVSS | AH31 | DISP_PWM |
| G29 | WB_CTRL1 | V22 | DVSS | AJ1 | AVSS18_MD |
| G30 | WB_TXIN | V23 | DVSS | AJ2 | AVSS18_MD |
| H1 | DVDD28_SIM1 | V28 | RDP3_A | AJ3 | AVSS18_MD |
| H2 | MSDC1_DAT2 | V29 | RDP2_A | AJ4 | AVSS18_MD |
| H3 | MSDC1_CLK | V30 | RDP1_A | AJ5 | AVSS18_MD |
| H4 | MSDC1_DAT1 | W1 | AVDD18_USB | AJ6 | AVSS18_MD |
| H9 | DVDD12_EMI | W2 | AVSS33_USB | AJ7 | LTE_RX1_BBIN |
| H10 | DVSS | W3 | AUD_DAT_MISO | AJ8 | LTE_RX2_BBQP |
| H12 | DVDD12_EMI | W4 | LCM_RST | AJ9 | AVSS18_MD |
| H13 | DVSS | W7 | DVSS | AJ10 | APC2 |
| H14 | DVDD12_EMI | W8 | DVSS | AJ13 | C2K_RX1_BBIP |
| H15 | DVSS | W9 | DVSS | AJ14 | C2K_RX2_BBQN |
| H16 | DVDD12_EMI | W10 | DVSS | AJ15 | AVSS18_PLLGP |
| H17 | DVSS | W11 | DVSS | AJ18 | RFIC1_TX_BSI_D0 |
| H18 | DVDD12_EMI | W12 | DVSS | AJ19 | RFIC1_BSI_CK |
| H28 | WB_CTRL4 | W13 | DVSS | AJ22 | BPI_BUS7 |
| H29 | WB_CTRL3 | W14 | DVDD_CPU | AJ23 | KPROW1 |
| H30 | WB_TXIP | W15 | DVSS | AJ26 | EINT7 |
| H31 | WB_RXQN | W16 | DVDD_LTE | AJ27 | URXD0 |
| J2 | SIM1_SIO | W17 | DVDD_LTE | AJ28 | UTXD0 |
| J4 | SIM1_SRST | W18 | DVDD_LTE | AJ29 | JTDI |
| J28 | ANT_SEL0 | W19 | DVDD_LTE | AJ30 | JTCK |
| J29 | WB_CTRL5 | W20 | DVDD_LTE | AJ31 | JTMS |

| Ball Loc. | Ball name | Ball Loc. | Ball name | Ball Loc. | Ball name |
|---|---|---|---|---|---|
| J30 | WB_RXIN | W21 | DVDD_LTE | AK1 | NC |
| J31 | WB_RXQP | W22 | DVDD_LTE | AK2 | LTE_TX_BBIP |
| K1 | SIM2_SIO | W23 | DVDD_LTE | AK3 | LTE_TX_BBIN |
| K2 | SIM2_SRST | W24 | DVDD_LTE | AK4 | LTE_TX_BBQN |
| K4 | SIM1_SCLK | W28 | CMDAT1 | AK5 | AVSS18_MD |
| K7 | DVSS | W29 | CMDAT0 | AK6 | LTE_RX1_BBQP |
| K8 | DVDD_CORE | W30 | FSOURCE_P | AK7 | LTE_RX2_BBIP |
| K9 | DVSS | W31 | AVDD18_MIPIRX1 | AK8 | LTE_RX2_BBIN |
| K27 | DVDD18_IOLT | Y1 | AUD_CLK_MOSI | AK9 | AVSS18_MD |
| K28 | NC | Y2 | AUD_DAT_MOSI | AK10 | AVSS_REFN |
| K30 | WB_RXIP | Y3 | PWRAP_SPI0_CK | AK11 | AVDD18_AP |
| L2 | DVDD28_SIM2 | Y4 | PWRAP_SPI0_CSN | AK12 | C2K_TX_BBQN |
| L3 | AVSS18_MIPITX | Y7 | DVDD_CORE | AK13 | C2K_TX_BBIN |
| L4 | SIM2_SCLK | Y8 | DVDD_CORE | AK14 | C2K_TX_BBIP |
| L7 | DVDD_CORE | Y9 | DVDD_CORE | AK15 | AVSS18_PLLGP |
| L8 | DVDD_CORE | Y10 | DVDD_CORE | AK16 | RFIC0_BSI_D2 |
| L9 | DVDD_CORE | Y11 | DVDD_CORE | AK17 | RFIC0_BSI_D0 |
| L10 | DVDD_CORE | Y12 | DVDD_CORE | AK18 | RFIC1_TX_BSI_CK |
| L11 | DVDD_CORE | Y13 | DVSS | AK19 | RFIC1_TX_BSI_EN |
| L12 | DVDD_CORE | Y14 | DVDD_CPU | AK20 | BPI_BUS20 |
| L13 | DVSS | Y15 | DVSS | AK21 | BPI_BUS18 |
| L14 | DVDD_SRAM | Y16 | DVDD_LTE | AK22 | BPI_BUS10 |
| L15 | DVDD_CORE | Y17 | DVDD_LTE | AK23 | BPI_BUS6 |
| L16 | DVDD_CORE | Y18 | DVDD_LTE | AK24 | KPCOL1 |
| L17 | DVSS | Y19 | DVDD_LTE | AK25 | EINT12 |
| L18 | DVSS | Y20 | DVDD_LTE | AK26 | I2S_BCK |
| L19 | DVDD_CORE | Y21 | DVDD_LTE | AK27 | I2S_DATA_IN |
| L20 | DVDD_CORE | Y22 | DVDD_LTE | AK28 | JTDO |
| L21 | DVDD_CORE | Y23 | DVDD_LTE | AK29 | EINT6 |
| L22 | DVDD_CORE | Y24 | DVDD_LTE | AK30 | TESTMODE |
| L23 | DVSS | Y27 | CMMCLK | AL1 | NC |
| L27 | NC | Y28 | CMMCLK1 | AL2 | NC |
| L28 | ANT_SEL2 | Y30 | SDA0 | AL3 | LTE_TX_BBQP |
| L29 | NC | Y31 | CMPCLK | AL4 | AVDD18_MD |
| L30 | AVSS18_WBG | AA2 | PWRAP_INT | AL5 | AVSS18_MD |
| L31 | DVDD28_MC2 | AA5 | DSI_TE | AL6 | LTE_RX1_BBQN |
| M1 | TDP3 | AA7 | DVSS | AL8 | AVSS18_MD |
| M2 | TDN3 | AA8 | DVDD_CORE | AL9 | AVDD28_DAC |
| M3 | TDN2 | AA9 | DVSS | AL10 | REFP |
| M7 | DVSS | AA10 | DVDD_CORE | AL12 | C2K_TX_BBQP |
| M8 | DVDD_CORE | AA11 | DVSS | AL14 | C2K_RX2_BBIN |
| M9 | DVSS | AA12 | DVDD_CORE | AL15 | C2K_RX2_BBIP |
| M10 | DVDD_CORE | AA13 | DVSS | AL17 | RFIC0_BSI_D1 |
| M11 | DVSS | AA14 | DVDD_CPU | AL18 | RFIC1_BSI_D0 |
| M12 | DVDD_CPU | AA15 | DVSS | AL20 | DVDD18_IOLB |
| M13 | DVDD_CPU | AA16 | DVDD_LTE | AL21 | BPI_BUS17 |
| M15 | DVDD_CPU | AA17 | DVSS | AL23 | BPI_BUS5 |
| M16 | DVDD_CPU | AA18 | DVDD_LTE | AL24 | KPCOL0 |

| Ball Loc. | Ball name | Ball Loc. | Ball name | Ball Loc. | Ball name |
|---|---|---|---|---|---|
| M17 | DVDD_CPU | AA19 | DVSS | AL26 | DVDD18_IOLB |
| M18 | DVDD_CPU | AA20 | DVDD_LTE | AL27 | I2S_LRCK |
| M19 | DVDD_CPU | AA21 | DVSS | AL29 | EINT5 |
| M20 | DVDD_CPU | AA22 | DVDD_LTE | AL30 | NC |
| M21 | DVSS | AA23 | DVSS | AL31 | NC |
| M22 | DVDD_CORE | AA24 | DVDD_LTE | | |

### 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 (using LPDDR3)*

| Pin name | Type | Description | Power domain |
|---|---|---|---|
| **System** | | | |
| SYSRSTB | DI | System reset input | DVDD18_IORB |
| WATCHDOG | DO | Watchdog reset output | DVDD18_IORB |
| TESTMODE | DIO | Test mode | DVDD18_IOLB |
| RTC32K_CK | DIO | 32K clock input | DVDD18_IORB |
| SRCLKENAI | DIO | 26MHz co-clock enable input | DVDD18_IOLB |
| SRCLKENA0 | DIO | 26MHz co-clock enable output | DVDD18_IORB |
| SRCLKENA1 | DIO | 26MHz co-clock enable output | DVDD18_IOLB |
| **PMIC** | | | |
| PWRAP_SPI0_MO | DIO | PMIC SPI control interface | DVDD18_IORB |
| PWRAP_SPI0_MI | DIO | PMIC SPI control interface | DVDD18_IORB |
| PWRAP_SPI0_CSN | DIO | PMIC SPI control interface | DVDD18_IORB |
| PWRAP_SPI0_CK | DIO | PMIC SPI control interface | DVDD18_IORB |
| PWRAP_INT | DIO | PMIC SPI control interface | DVDD18_IORB |
| AUD_CLK_MOSI | DIO | PMIC audio input interface | DVDD18_IORB |
| AUD_DAT_MOSI | DIO | PMIC audio input interface | DVDD18_IORB |
| AUD_DAT_MISO | DIO | PMIC audio input interface | DVDD18_IORB |
| **SIM** | | | |
| SIM1_SIO | DIO | SIM1 data, PMIC interface | DVDD18_MC1 |

| Pin name | Type | Description | Power domain |
|---|---|---|---|
| SIM1_SRST | DIO | SIM1 reset, PMIC interface | DVDD18_MC1 |
| SIM1_SCLK | DIO | SIM1 clock, PMIC interface | DVDD18_MC1 |
| SIM2_SIO | DIO | SIM2 data, PMIC interface | DVDD18_MC1 |
| SIM2_SRST | DIO | SIM2 reset, PMIC interface | DVDD18_MC1 |
| SIM2_SCLK | DIO | SIM2 clock, PMIC interface | DVDD18_MC1 |
| **JTAG** | | | |
| JTCK | DIO | JTCK | DVDD18_IOLB |
| JTDO | DIO | JTDO | DVDD18_IOLB |
| JTDI | DIO | JTDI | DVDD18_IOLB |
| JTMS | DIO | JTMS | DVDD18_IOLB |
| **LCD** | | | |
| DISP_PWM | DIO | Display PWM output | DVDD18_IOLB |
| DSI_TE | DIO | Parallel display interface tearing effect | DVDD18_IORB |
| LCM_RST | DIO | Parallel display interface reset signal | DVDD18_IORB |
| **I2S** | | | |
| I2S_DATA_IN | DIO | I2S data input pin | DVDD18_IOLB |
| I2S_BCK | DIO | I2S clock | DVDD18_IOLB |
| I2S_LRCK | DIO | I2S word select | DVDD18_IOLB |
| **PCM/I2S merge interface** | | | |
| PCM_TX | DIO | PCM audio interface | DVDD18_IOLB |
| PCM_CLK | DIO | PCM audio interface | DVDD18_IOLB |
| PCM_RX | DIO | PCM audio interface | DVDD18_IOLB |
| PCM_SYNC | DIO | PCM audio interface | DVDD18_IOLB |
| **EINT** | | | |
| EINT0 | DIO | External interrupt 0 | DVDD18_IOLB |
| EINT1 | DIO | External interrupt 1 | DVDD18_IOLB |
| EINT2 | DIO | External interrupt 2 | DVDD18_IOLB |
| EINT3 | DIO | External interrupt 3 | DVDD18_IOLB |
| EINT4 | DIO | External interrupt 4 | DVDD18_IOLB |
| EINT5 | DIO | External interrupt 5 | DVDD18_IOLB |
| EINT6 | DIO | External interrupt 6 | DVDD18_IOLB |
| EINT7 | DIO | External interrupt 7 | DVDD18_IOLB |
| EINT8 | DIO | External interrupt 8 | DVDD18_IOLB |
| EINT9 | DIO | External interrupt 9 | DVDD18_IOLB |
| EINT10 | DIO | External interrupt 10 | DVDD18_IOLB |
| EINT11 | DIO | External interrupt 11 | DVDD18_IOLB |
| EINT12 | DIO | External interrupt 12 | DVDD18_IOLB |
| **UART** | | | |
| URXD0 | DIO | UART0 RX | DVDD18_IOLB |
| UTXD0 | DIO | UART0 TX | DVDD18_IOLB |
| URXD1 | DIO | UART1 RX | DVDD18_IOLB |
| UTXD1 | DIO | UART1 TX | DVDD18_IOLB |

| Pin name | Type | Description | Power domain |
|---|---|---|---|
| URXD2 | DIO | UART2 RX | DVDD18_IOLB |
| UTXD2 | DIO | UART2 TX | DVDD18_IOLB |
| URXD3 | DIO | UART3 RX | DVDD18_IOLB |
| UTXD3 | DIO | UART3 TX | DVDD18_IOLB |
| **SPI** | | | |
| SPI_CS | DIO | SPI chip select | DVDD18_IOLB |
| SPI_MI | DIO | SPI data in | DVDD18_IOLB |
| SPI_MO | DIO | SPI data out | DVDD18_IOLB |
| SPI_CK | DIO | SPI clock | DVDD18_IOLB |
| **BPI** | | | |
| BPI_BUS0 | DIO | BPI1 BUS0 | DVDD18_IORB |
| BPI_BUS1 | DIO | BPI1 BUS1 | DVDD18_IORB |
| BPI_BUS2 | DIO | BPI1 BUS2 | DVDD18_IORB |
| BPI_BUS3 | DIO | BPI1 BUS3 | DVDD18_IORB |
| BPI_BUS4 | DIO | BPI1 BUS4 | DVDD18_IORB |
| BPI_BUS5 | DIO | BPI1 BUS5 | DVDD18_IOLB |
| BPI_BUS6 | DIO | BPI1 BUS6 | DVDD18_IOLB |
| BPI_BUS7 | DIO | BPI1 BUS7 | DVDD18_IOLB |
| BPI_BUS8 | DIO | BPI1 BUS8 | DVDD18_IOLB |
| BPI_BUS9 | DIO | BPI1 BUS9 | DVDD18_IOLB |
| BPI_BUS10 | DIO | BPI1 BUS10 | DVDD18_IOLB |
| BPI_BUS11 | DIO | BPI1 BUS11 | DVDD18_IOLB |
| BPI_BUS12 | DIO | BPI1 BUS12 | DVDD18_IOLB |
| BPI_BUS13 | DIO | BPI1 BUS13 | DVDD18_IOLB |
| BPI_BUS14 | DIO | BPI1 BUS14 | DVDD18_IOLB |
| BPI_BUS15 | DIO | BPI1 BUS15 | DVDD18_IOLB |
| BPI_BUS16 | DIO | BPI1 BUS16 | DVDD18_IOLB |
| BPI_BUS17 | DIO | BPI1 BUS17 | DVDD18_IOLB |
| BPI_BUS18 | DIO | BPI1 BUS18 | DVDD18_IOLB |
| BPI_BUS19 | DIO | BPI1 BUS19 | DVDD18_IOLB |
| BPI_BUS20 | DIO | BPI1 BUS20 | DVDD18_IOLB |
| BPI_BUS21 | DIO | BPI1 BUS21 | DVDD18_IORB |
| BPI_BUS22 | DIO | BPI1 BUS22 | DVDD18_IORB |
| BPI_BUS23 | DIO | BPI1 BUS23 | DVDD18_IORB |
| BPI_BUS24 | DIO | BPI1 BUS24 | DVDD18_IORB |
| BPI_BUS25 | DIO | BPI1 BUS25 | DVDD18_IORB |
| BPI_BUS26 | DIO | BPI1 BUS26 | DVDD18_IORB |
| BPI_BUS27 | DIO | BPI1 BUS27 | DVDD18_IORB |
| ANT_SEL0 | DIO | Antenna select 0 | DVDD18_IOLT |
| ANT_SEL1 | DIO | Antenna select 1 | DVDD18_IOLT |
| ANT_SEL2 | DIO | Antenna select 2 | DVDD18_IOLT |
| **VM** | | | |
| LTE_PAVM1 | DIO | PA mode selection | DVDD18_IORB |

| Pin name | Type | Description | Power domain |
|---|---|---|---|
| LTE_PAVM0 | DIO | PA mode selection | DVDD18_IORB |
| **BSI** | | | |
| RFIC1_BSI_EN | DIO | RFIC1 BSI enable | DVDD18_IOLB |
| RFIC1_BSI_CK | DIO | RFIC1 BSI clock | DVDD18_IOLB |
| RFIC1_BSI_D0 | DIO | RFIC1 BSI Data0 | DVDD18_IOLB |
| RFIC1_TX_BSI_EN | DIO | RFIC1 TX BSI enable | DVDD18_IOLB |
| RFIC1_TX_BSI_CK | DIO | RFIC1 TX BSI clock | DVDD18_IOLB |
| RFIC1_TX_BSI_D0 | DIO | RFIC1 TX BSI Data0 | DVDD18_IOLB |
| RFIC0_BSI_EN | DIO | RFIC0 BSI enable | DVDD18_IOLB |
| RFIC0_BSI_CK | DIO | RFIC0 BSI clock | DVDD18_IOLB |
| RFIC0_BSI_D2 | DIO | RFIC0 BSI Data2 | DVDD18_IOLB |
| RFIC0_BSI_D1 | DIO | RFIC0 BSI Data1 | DVDD18_IOLB |
| RFIC0_BSI_D0 | DIO | RFIC0 BSI Data0 | DVDD18_IOLB |
| RFIC_MIPI1_SCLK | DIO | RFIC MIPI1 SCLK | DVDD18_IORB |
| RFIC_MIPI1_SDATA | DIO | RFIC MIPI1 SDATA | DVDD18_IORB |
| RFIC_MIPI0_SCLK | DIO | RFIC MIPI0 SCLK | DVDD18_IORB |
| RFIC_MIPI0_SDATA | DIO | RFIC MIPI0 SDATA | DVDD18_IORB |
| C2K_TXBPI | DIO | C2K TXBPI | DVDD18_IOLB |
| LTE_TXBPI | DIO | LTE TXBPI | DVDD18_IORB |
| **MSDC0** | | | |
| MSDC0_DAT7 | DIO | MSDC0 data7 pin | DVDD18_MC0 |
| MSDC0_DAT6 | DIO | MSDC0 data6 pin | DVDD18_MC0 |
| MSDC0_DAT5 | DIO | MSDC0 data5 pin | DVDD18_MC0 |
| MSDC0_RSTB | DIO | MSDC0 reset output | DVDD18_MC0 |
| MSDC0_DAT4 | DIO | MSDC0 data4 pin | DVDD18_MC0 |
| MSDC0_DAT2 | DIO | MSDC0 data2 pin | DVDD18_MC0 |
| MSDC0_DAT3 | DIO | MSDC0 data3 pin | DVDD18_MC0 |
| MSDC0_CMD | DIO | MSDC0 command pin | DVDD18_MC0 |
| MSDC0_CLK | DIO | MSDC0 clock output | DVDD18_MC0 |
| MSDC0_DAT1 | DIO | MSDC0 data1 pin | DVDD18_MC0 |
| MSDC0_DAT0 | DIO | MSDC0 data0 pin | DVDD18_MC0 |
| **MSDC1** | | | |
| MSDC1_CLK | DIO | MSDC1 clock output | DVDD28_MC1/DVDD18_MC1 |
| MSDC1_CMD | DIO | MSDC1 command pin | DVDD28_MC1/DVDD18_MC1 |
| MSDC1_DAT0 | DIO | MSDC1 data0 pin | DVDD28_MC1/DVDD18_MC1 |
| MSDC1_DAT1 | DIO | MSDC1 data1 pin | DVDD28_MC1/DVDD18_MC1 |
| MSDC1_DAT2 | DIO | MSDC1 data2 pin | DVDD28_MC1/DVDD18_MC1 |
| MSDC1_DAT3 | DIO | MSDC1 data3 pin | DVDD28_MC1/DVDD18_MC1 |
| **WiFi/BT/GPS** | | | |
| WB_SDATA | DIO | WiFi/BT SPI control data | DVDD18_CONN |
| WB_SCLK | DIO | WiFi/BT SPI control clock | DVDD18_CONN |
| WB_SEN | DIO | WiFi/BT SPI control enable | DVDD18_CONN |

| Pin name | Type | Description | Power domain |
|---|---|---|---|
| WB_RSTB | DIO | WiFi/BT SPI control reset | DVDD18_CONN |
| F2W_CLK | DIO | FM clock | DVDD18_CONN |
| F2W_DATA | DIO | FM data | DVDD18_CONN |
| WB_CTRL0 | DIO | Data bus 0 | DVDD18_IOLT |
| WB_CTRL1 | DIO | Data bus 1 | DVDD18_IOLT |
| WB_CTRL2 | DIO | Data bus 2 | DVDD18_IOLT |
| WB_CTRL3 | DIO | Data bus 3 | DVDD18_IOLT |
| WB_CTRL4 | DIO | Data bus 4 | DVDD18_IOLT |
| WB_CTRL5 | DIO | Data bus 5 | DVDD18_IOLT |
| **EFUSE** | | | |
| FSOURCE_P | DIO | E-FUSE blowing power control | FSOURCE_P |
| **EMI** | | | |
| RCLK0 | DIO | DRAM clock 0 output | DVDD12_EMI |
| RCLK0_B | DIO | DRAM clock 0 output # | DVDD12_EMI |
| RCKE | DIO | DRAM command output CKE | DVDD12_EMI |
| RCS_B | DIO | DRAM chip select 0 # | DVDD12_EMI |
| RCS1_B | DIO | DRAM chip select 1 # | DVDD12_EMI |
| RA0 | DIO | DRAM address output 0 | DVDD12_EMI |
| RA1 | DIO | DRAM address output 1 | DVDD12_EMI |
| RA2 | DIO | DRAM address output 2 | DVDD12_EMI |
| RA3 | DIO | DRAM address output 3 | DVDD12_EMI |
| RA4 | DIO | DRAM address output 4 | DVDD12_EMI |
| RA5 | DIO | DRAM address output 5 | DVDD12_EMI |
| RA6 | DIO | DRAM address output 6 | DVDD12_EMI |
| RA7 | DIO | DRAM address output 7 | DVDD12_EMI |
| RA8 | DIO | DRAM address output 8 | DVDD12_EMI |
| RA9 | DIO | DRAM address output 9 | DVDD12_EMI |
| RDQM0 | DIO | DRAM DQM 0 | DVDD12_EMI |
| RDQM1 | DIO | DRAM DQM 1 | DVDD12_EMI |
| RDQM2 | DIO | DRAM DQM 2 | DVDD12_EMI |
| RDQM3 | DIO | DRAM DQM 3 | DVDD12_EMI |
| RDQS0 | DIO | DRAM DQS 0 | DVDD12_EMI |
| RDQS0_B | DIO | DRAM DQS 0 # | DVDD12_EMI |
| RDQS1 | DIO | DRAM DQS 1 | DVDD12_EMI |
| RDQS1_B | DIO | DRAM DQS 1 # | DVDD12_EMI |
| RDQS2 | DIO | DRAM DQS 2 | DVDD12_EMI |
| RDQS2_B | DIO | DRAM DQS 2 # | DVDD12_EMI |
| RDQS3 | DIO | DRAM DQS 3 | DVDD12_EMI |
| RDQS3_B | DIO | DRAM DQS 3 # | DVDD12_EMI |
| RDQ0 | DIO | DRAM data pin 0 | DVDD12_EMI |
| RDQ1 | DIO | DRAM data pin 1 | DVDD12_EMI |
| RDQ2 | DIO | DRAM data pin 2 | DVDD12_EMI |
| RDQ3 | DIO | DRAM data pin 3 | DVDD12_EMI |

| Pin name | Type | Description | Power domain |
|---|---|---|---|
| RDQ4 | DIO | DRAM data pin 4 | DVDD12_EMI |
| RDQ5 | DIO | DRAM data pin 5 | DVDD12_EMI |
| RDQ6 | DIO | DRAM data pin 6 | DVDD12_EMI |
| RDQ7 | DIO | DRAM data pin 7 | DVDD12_EMI |
| RDQ8 | DIO | DRAM data pin 8 | DVDD12_EMI |
| RDQ9 | DIO | DRAM data pin 9 | DVDD12_EMI |
| RDQ10 | DIO | DRAM data pin 10 | DVDD12_EMI |
| RDQ11 | DIO | DRAM data pin 11 | DVDD12_EMI |
| RDQ12 | DIO | DRAM data pin 12 | DVDD12_EMI |
| RDQ13 | DIO | DRAM data pin 13 | DVDD12_EMI |
| RDQ14 | DIO | DRAM data pin 14 | DVDD12_EMI |
| RDQ15 | DIO | DRAM data pin 15 | DVDD12_EMI |
| RDQ16 | DIO | DRAM data pin 16 | DVDD12_EMI |
| RDQ17 | DIO | DRAM data pin 17 | DVDD12_EMI |
| RDQ18 | DIO | DRAM data pin 18 | DVDD12_EMI |
| RDQ19 | DIO | DRAM data pin 19 | DVDD12_EMI |
| RDQ20 | DIO | DRAM data pin 20 | DVDD12_EMI |
| RDQ21 | DIO | DRAM data pin 21 | DVDD12_EMI |
| RDQ22 | DIO | DRAM data pin 22 | DVDD12_EMI |
| RDQ23 | DIO | DRAM data pin 23 | DVDD12_EMI |
| RDQ24 | DIO | DRAM data pin 24 | DVDD12_EMI |
| RDQ25 | DIO | DRAM data pin 25 | DVDD12_EMI |
| RDQ26 | DIO | DRAM data pin 26 | DVDD12_EMI |
| RDQ27 | DIO | DRAM data pin 27 | DVDD12_EMI |
| RDQ28 | DIO | DRAM data pin 28 | DVDD12_EMI |
| RDQ29 | DIO | DRAM data pin 29 | DVDD12_EMI |
| RDQ30 | DIO | DRAM data pin 30 | DVDD12_EMI |
| RDQ31 | DIO | DRAM data pin 31 | DVDD12_EMI |
| REXTDN | DIO | DRAM REXTDN pin | DVDD12_EMI |
| VREF | DIO |  | DVDD12_EMI |
| **CAM** |  |  |  |
| CMPCLK | DIO | Pixel clock from sensor | DVDD18_IOLB |
| CMMCLK | DIO | Master clock to sensor | DVDD18_IOLB |
| CMMCLK1 | DIO | Master clock1 to sensor | DVDD18_IOLB |
| CMDAT0 | DIO | CAM sensor Data0 | DVDD18_IOLB |
| CMDAT1 | DIO | CAM sensor Data1 | DVDD18_IOLB |
| **I2C0** |  |  |  |
| SCL0 | DIO | I2C0 clock | DVDD18_IOLB |
| SDA0 | DIO | I2C0 data | DVDD18_IOLB |
| **I2C1** |  |  |  |
| SCL1 | DIO | I2C1 clock | DVDD18_IOLB |
| SDA1 | DIO | I2C1 data | DVDD18_IOLB |
| **I2C2** |  |  |  |

| Pin name | Type | Description | Power domain |
|---|---|---|---|
| SCL2 | DIO | I2C2 clock | DVDD18_IOLB |
| SDA2 | DIO | I2C2 data | DVDD18_IOLB |
| **I2C3** | | | |
| SCL3 | DIO | I2C3 clock | DVDD18_IOLB |
| SDA3 | DIO | I2C3 data | DVDD18_IOLB |
| **ABB** | | | |
| C2K_RX2_BBQP | AIO | C2K downlink QP for diversity path | AVDD18_MD |
| C2K_RX2_BBQN | AIO | C2K downlink QN for diversity path | AVDD18_MD |
| C2K_RX2_BBIN | AIO | C2K downlink IN for diversity path | AVDD18_MD |
| C2K_RX2_BBIP | AIO | C2K downlink IPP for diversity path | AVDD18_MD |
| C2K_RX1_BBQP | AIO | C2K downlink QP for main path | AVDD18_MD |
| C2K_RX1_BBQN | AIO | C2K downlink QN for main path | AVDD18_MD |
| C2K_RX1_BBIN | AIO | C2K downlink IN for main path | AVDD18_MD |
| C2K_RX1_BBIP | AIO | C2K downlink IPP for main path | AVDD18_MD |
| C2K_TX_BBQP | AIO | C2K uplink QP | AVDD18_MD |
| C2K_TX_BBQN | AIO | C2K uplink QN | AVDD18_MD |
| C2K_TX_BBIN | AIO | C2K uplink IN | AVDD18_MD |
| C2K_TX_BBIP | AIO | C2K uplink IP | AVDD18_MD |
| C2KX26M_IN | AIO | 26MHz clock input for 2nd modem | AVDD18_MD |
| AUXIN0 | AIO | AuxADC external input channel 0 | AVDD18_AP |
| AUXIN1 | AIO | AuxADC external input channel 1 | AVDD18_AP |
| AUXIN2 | AIO | AuxADC external input channel 2 | AVDD18_AP |
| REFP | AIO | Positive reference port for internal circuit | AVDD18_AP |
| APC1 | AIO | Automatic power control for MD1 | AVDD28_DAC |
| APC2 | AIO | Automatic power control for MD2 | AVDD28_DAC |
| LTEX26M_IN | AIO | 26MHz clock input for AP & 1st modem | AVDD18_MD |
| LTE_RX2_BBQP | AIO | LTE downlink QP for diversity path | AVDD18_MD |
| LTE_RX2_BBQN | AIO | LTE downlink QN for diversity path | AVDD18_MD |
| LTE_RX2_BBIN | AIO | LTE downlink IN for diversity path | AVDD18_MD |
| LTE_RX2_BBIP | AIO | LTE downlink IPP for diversity path | AVDD18_MD |
| LTE_RX1_BBQP | AIO | LTE downlink QP for main path | AVDD18_MD |
| LTE_RX1_BBQN | AIO | LTE downlink QN for main path | AVDD18_MD |
| LTE_RX1_BBIN | AIO | LTE downlink IN for main path | AVDD18_MD |
| LTE_RX1_BBIP | AIO | LTE downlink IP for main path | AVDD18_MD |
| LTE_TX_BBQP | AIO | LTE uplink QP | AVDD18_MD |
| LTE_TX_BBQN | AIO | LTE uplink QN | AVDD18_MD |
| LTE_TX_BBIN | AIO | LTE uplink IN | AVDD18_MD |
| LTE_TX_BBIP | AIO | LTE uplink IP | AVDD18_MD |
| RFIC_ET_N | AIO | Envelop Tracking DAC output N | AVDD18_MD |
| RFIC_ET_P | AIO | Envelop Tracking DAC output P | AVDD18_MD |
| **WBG** | | | |

| Pin name | Type | Description | Power domain |
|---|---|---|---|
| XIN_WBG | AIO | WIFI/BT clock source | AVDD18_WBG |
| GPS_RXQN | AIO | RXQN for GPS RX | AVDD18_WBG |
| GPS_RXQP | AIO | RXQP for GPS RX | AVDD18_WBG |
| GPS_RXIN | AIO | RXIN for GPS RX | AVDD18_WBG |
| GPS_RXIP | AIO | RXIP for GPS RX | AVDD18_WBG |
| WB_TXQN | AIO | TXQN for WIFI/BT TX | AVDD18_WBG |
| WB_TXQP | AIO | TXQP for WIFI/BT TX | AVDD18_WBG |
| WB_TXIN | AIO | TXIN for WIFI/BT TX | AVDD18_WBG |
| WB_TXIP | AIO | TXIP for WIFI/BT TX | AVDD18_WBG |
| WB_RXQN | AIO | RXQN for WIFI/BT RX | AVDD18_WBG |
| WB_RXQP | AIO | RXQP for WIFI/BT RX | AVDD18_WBG |
| WB_RXIN | AIO | RXIN for WIFI/BT RX | AVDD18_WBG |
| WB_RXIP | AIO | RXIP for WIFI/BT RX | AVDD18_WBG |
| **MIPI** | | | |
| TDN3 | AIO | DSI0 lane3 N | DVDD18_MIPITX |
| TDP3 | AIO | DSI0 lane3 P | DVDD18_MIPITX |
| TDN2 | AIO | DSI0 lane2 N | DVDD18_MIPITX |
| TDP2 | AIO | DSI0 lane2 P | DVDD18_MIPITX |
| TCN | AIO | DSI0 CK lane N | DVDD18_MIPITX |
| TCP | AIO | DSI0 CK lane P | DVDD18_MIPITX |
| TDN1 | AIO | DSI0 lane1 N | DVDD18_MIPITX |
| TDP1 | AIO | DSI0 lane1 P | DVDD18_MIPITX |
| TDN0 | AIO | DSI0 lane0 N | DVDD18_MIPITX |
| TDP0 | AIO | DSI0 lane0 P | DVDD18_MIPITX |
| VRT | AO | External resistor for DSI bias<br>Connect 1.5K ohm 1% resistor to ground | DVDD18_MIPITX |
| RDN3 | AIO | CSI0 lane3 N | DVDD18_MIPIRX0 |
| RDP3 | AIO | CSI0 lane3 P | DVDD18_MIPIRX0 |
| RDN2 | AIO | CSI0 lane2 N | DVDD18_MIPIRX0 |
| RDP2 | AIO | CSI0 lane2 P | DVDD18_MIPIRX0 |
| RCN | AIO | CSI0 CK lane N | DVDD18_MIPIRX0 |
| RCP | AIO | CSI0 CK lane P | DVDD18_MIPIRX0 |
| RDN1 | AIO | CSI0 lane1 N | DVDD18_MIPIRX0 |
| RDP1 | AIO | CSI0 lane1 P | DVDD18_MIPIRX0 |
| RDN0 | AIO | CSI0 lane0 N | DVDD18_MIPIRX0 |
| RDP0 | AIO | CSI0 lane0 P | DVDD18_MIPIRX0 |
| RDN1_A | AIO | CSI1 lane1 N | DVDD18_MIPIRX1 |
| RDP1_A | AIO | CSI1 lane1 P | DVDD18_MIPIRX1 |
| RCN_A | AIO | CSI1 CK lane N | DVDD18_MIPIRX1 |
| RCP_A | AIO | CSI1 CK lane P | DVDD18_MIPIRX1 |
| RDN0_A | AIO | CSI1 lane0 N | DVDD18_MIPIRX1 |
| RDP0_A | AIO | CSI1 lane0 P | DVDD18_MIPIRX1 |

| Pin name | Type | Description | Power domain |
|---|---|---|---|
| RDN2_A | AIO | CSI1 lane 2 N | DVDD18_MIPIRX1 |
| RDP2_A | AIO | CSI1 lane 2 P | DVDD18_MIPIRX1 |
| RDN3_A | AIO | CSI1 lane 3 N | DVDD18_MIPIRX1 |
| RDP3_A | AIO | CSI1 lane 3 P | DVDD18_MIPIRX1 |
| **USB** | | | |
| USB_DP | AIO | USB port0 D+ differential data line | AVDD33_USB |
| USB_DM | AIO | USB port0 D- differential data line | AVDD33_USB |
| CHD_DP | AIO | BC1.1 Charger DP | AVDD33_USB |
| CHD_DM | AIO | BC1.1 Charger DM | AVDD33_USB |
| USB_VRT | AO | USB output for bias current; connect with 5.11K 1% Ohm to GND | AVDD18_USB |
| **Keypad** | | | |
| KPROW0 | AIO | Keypad row 0 | DVDD18_IOLB |
| KPROW1 | AIO | Keypad row 1 | DVDD18_IOLB |
| KPROW2 | AIO | Keypad row 2 | DVDD18_IOLB |
| KPCOL0 | AIO | Keypad column 0 | DVDD18_IOLB |
| KPCOL1 | AIO | Keypad column 1 | DVDD18_IOLB |
| KPCOL2 | AIO | Keypad column 2 | DVDD18_IOLB |
| **Analog power** | | | |
| DVDD18_PLLGP | P | Analog power input 1.8V for PLL | |
| AVDD18_AP | P | Analog power input 1.8V for AuxADC, TSENSE | |
| AVDD18_MD | P | Analog power input 1.8V for BBTX, BBRX, 2GBBTX | |
| AVDD18_MEMPLL | P | Analog power for MEMPLL | |
| AVDD18_USB | P | Analog power 1.8V for USB | |
| AVDD18_WBG | P | Analog power 1.8V for WiFi/BT/GPS | |
| DVDD18_MIPITX | P | Analog power for MIPI DSI | |
| DVDD18_MIPIRX0 | P | Analog power for MIPI CSI | |
| DVDD18_MIPIRX1 | P | Analog power for MIPI CSI | |
| AVDD28_DAC | P | Analog power input 2.8V for APC | |
| AVDD33_USB | P | Analog power 3.3V for USB port 1 | |
| **Digital power** | | | |
| DVDD18_IOLT | P | Digital power input for IO | - |
| DVDD18_IOLB | P | Digital power input for IO | - |
| DVDD18_IORB | P | Digital power input for IO | - |
| DVDD18_CONN | P | Digital power input for IO | - |
| EVDD18_EFUSE | P | Digital power input for efuse IO | - |
| DVDD18_MC0 | P | Digital power input for MSDC0 IO | - |
| DVDD18_MC1 | P | Digital power input for MSDC1 IO | - |
| DVDD28_MC1 | P | Digital power input for 1.8/3.3V MSDC IO | - |
| DVDD28_MC2 | P | Digital power input for 1.8/3.3V MSDC IO | - |

| Pin name | Type | Description | Power domain |
|---|---|---|---|
| DVDD12_EMI | P | Digital power input for 1.2V EMI | - |
| DVDD_TOP | P | Digital power input for core | - |
| DVDD_LTE | P | Digital power input for LTE | - |
| VLTE_SRAM | P | Digital power input for LTE SRAM | - |
| DVDD_CPU | P | Digital power input for processor | - |
| DVDD_SRAM | P | Digital power input for processor SRAM | - |
| **Analog ground** | | | |
| AVSS18_AP | G | analog ground | |
| AVSS18_MD | G | analog ground | |
| AVSS18_MEMPLL | G | analog ground | |
| AVSS18_WBG | G | analog ground | |
| AVSS_REFN | G | analog ground | |
| AVSS33_USB | G | analog ground | |
| DVSS18_MIPITX | G | analog ground | |
| DVSS18_MIPIRX0 | G | analog ground | |
| DVSS18_MIPIRX1 | G | analog ground | |
| AVSS33_USB | G | analog ground | |
| **Digital ground** | | | |
| DVSS | G | Digital ground | - |

*Table 2-4. Acronym for the 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 |
| 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] | | | |
| **eMMC interface** | | | | | | |
| MSDC0_CLK | LO | 1 | - | 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_RSTB | HO | 1 | - | DIOH4, DIOL4 | No Need | IO Type 4 |
| MSDC0_DSL | I | 1 | PD | DIOH4, DIOL4 | No Need | IO Type 4 |
| **SD card interface** | | | | | | |
| MSDC1_CLK | LO | 1 | - | 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 |
| **SIM interface** | | | | | | |
| SIM1_SCLK | I | 0 | PD | DIOH7, DIOL7 | No Need | IO Type 7 |
| SIM1_SRST | I | 0 | PD | DIOH7, DIOL7 | No Need | IO Type 7 |
| SIM1_SIO | I | 0 | PD | DIOH7, DIOL7 | No Need | IO Type 7 |
| SIM2_SCLK | I | 0 | PD | DIOH7, DIOL7 | No Need | IO Type 7 |
| SIM2_SRST | I | 0 | PD | DIOH7, DIOL7 | No Need | IO Type 7 |
| SIM2_SIO | I | 0 | PD | DIOH7, DIOL7 | No Need | IO Type 7 |
| **Audio interface** | | | | | | |
| AUD_CLK_MOSI | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| AUD_DAT_MISO | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| AUD_DAT_MOSI | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| **LCD control** | | | | | | |
| DISP_PWM | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| LCM_RST | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| DSI_TE | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| **CAM interface** | | | | | | |
| CMPCLK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| CMMCLK | I | 0 | PD | DIOH4, DIOL4 | No Need | IO Type 4 |
| CMMCLK1 | I | 0 | PD | DIOH4, DIOL4 | No Need | IO Type 4 |
| CMPDAT0 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| CMPDAT1 | 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).

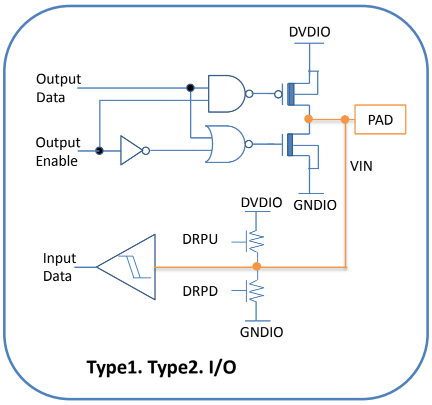
[2] The column “Aux” for “Reset” means the default aux. funtion number shown in Table “Pin Multiplexing, Capability and Settings”.

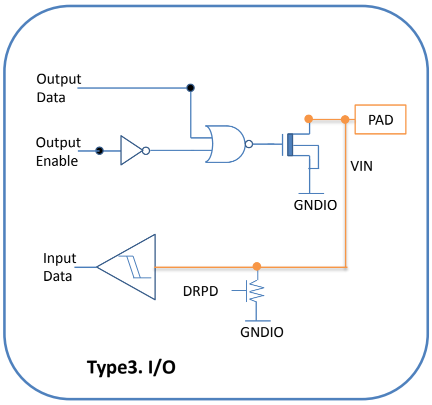
[3] The column “PU/PD” for “Reset” means if there is internal pull-up or pull-down when the pin is input in the reset state.

| Name | Reset | | | Output drivability | Termination when not used | IO type |
|---|---|---|---|---|---|---|
| | State[1] | Aux[2] | PU/PD[3] | | | |
| **PMIC SPI interface** | | | | | | |
| PWRAP_SPI0_CK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| PWRAP_SPI0_CSN | I | 0 | PU | DIOH1, DIOL1 | No Need | IO Type 1 |
| PWRAP_SPI0_MI | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| PWRAP_SPI0_MO | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| **I2C interface** | | | | | | |
| SCL0 | I | 1 | - | DIOH3, DIOL3 | No Need | IO Type 3 |
| SCL1 | I | 1 | - | DIOH3, DIOL3 | No Need | IO Type 3 |
| SCL2 | I | 1 | - | DIOH3, DIOL3 | No Need | IO Type 3 |
| SCL3 | I | 1 | - | DIOH3, DIOL3 | No Need | IO Type 3 |
| SDA0 | I | 1 | - | DIOH3, DIOL3 | No Need | IO Type 3 |
| SDA1 | I | 1 | - | DIOH3, DIOL3 | No Need | IO Type 3 |
| SDA2 | I | 1 | - | DIOH1, DIOL1 | No Need | IO Type 1 |
| SDA3 | I | 1 | - | DIOH3, DIOL3 | No Need | IO Type 3 |
| **RFIC interface** | | | | | | |
| RFIC0_BSI_EN | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC0_BSI_CK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC0_BSI_D0 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC0_BSI_D1 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC0_BSI_D2 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC1_BSI_EN | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC1_BSI_CK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC1_BSI_D0 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC1_TX_BSI_EN | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC1_TX_BSI_CK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC1_TX_BSI_D0 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC_MIPI1_SCLK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC_MIPI1_SDATA | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC_MIPI0_SCLK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| RFIC_MIPI0_SDATA | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| **BPI interface** | | | | | | |
| BPI_BUS0 | I | 0 | PD | DIOH6, DIOL6 | No Need | IO Type 6 |
| BPI_BUS1 | I | 0 | PD | DIOH6, DIOL6 | No Need | IO Type 6 |
| BPI_BUS2 | I | 0 | PD | DIOH6, DIOL6 | No Need | IO Type 6 |
| BPI_BUS3 | I | 0 | PD | DIOH6, DIOL6 | No Need | IO Type 6 |
| BPI_BUS4 | I | 0 | PD | DIOH6, DIOL6 | No Need | IO Type 6 |
| BPI_BUS5 | I | 0 | PD | DIOH6, DIOL6 | No Need | IO Type 6 |
| BPI_BUS6 | I | 0 | PD | DIOH5, DIOL5 | No Need | IO Type 5 |
| BPI_BUS7 | I | 0 | PD | DIOH5, DIOL5 | No Need | IO Type 5 |
| BPI_BUS8 | I | 0 | PD | DIOH5, DIOL5 | No Need | IO Type 5 |
| BPI_BUS9 | I | 0 | PD | DIOH5, DIOL5 | No Need | IO Type 5 |
| BPI_BUS10 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS11 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS12 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS13 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS14 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS15 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS16 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS17 | 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] | | | |
| BPI_BUS18 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS19 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS20 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS21 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS22 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS23 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS24 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS25 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS26 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| BPI_BUS27 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| LTE_TXBPI | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| C2K_TXBPI | I | 0 | PD | DIOH2, DIOL2 | No Need | IO Type 2 |
| **Connectivity RF interface** | | | | | | |
| WB_CTRL0 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| WB_CTRL1 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| WB_CTRL2 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| WB_CTRL3 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| WB_CTRL4 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| WB_CTRL5 | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| F2W_DATA | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| F2W_CLK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| WB_SCLK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| WB_SDATA | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| WB_SEN | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| WB_RSTB | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| **Keypad** | | | | | | |
| 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 | LO | 1 | PD | 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** | | | | | | |
| I2S0_MCK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S0_BCK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S0_LRCK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S0_DI | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S1_MCK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S1_BCK | I | 0 | PD | DIOH2, DIOL2 | No Need | IO Type 2 |
| I2S1_LRCK | I | 0 | PD | DIOH2, DIOL2 | No Need | IO Type 2 |
| I2S1_DO | I | 0 | PD | DIOH2, DIOL2 | No Need | IO Type 2 |
| I2S2_MCK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S2_BCK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S2_LRCK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S2_DI | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S3_MCK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S3_BCK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S3_LRCK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| I2S3_DO | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |

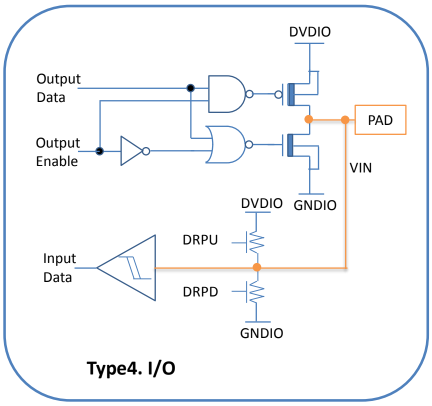
| Name | Reset | | | Output drivability | Termination when not used | IO type |
|---|---|---|---|---|---|---|
| | State[1] | Aux[2] | PU/PD[3] | | | |
| **SPI interface** | | | | | | |
| SPI_CSB | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| SPI_CLK | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| SPI_MO | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| SPI_MI | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| **System/reset clock enable** | | | | | | |
| WATCHDOG | HO | 1 | - | DIOH1, DIOL1 | No Need | IO Type 1 |
| SRCLKENA0 | HO | 1 | - | DIOH1, DIOL1 | No Need | IO Type 1 |
| SRCLKENA1 | LO | 1 | - | DIOH1, DIOL1 | No Need | IO Type 1 |
| SRCLKENAI | I | 1 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| IDDIG | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |
| USB_DRVBUS | I | 0 | PD | DIOH1, DIOL1 | No Need | IO Type 1 |

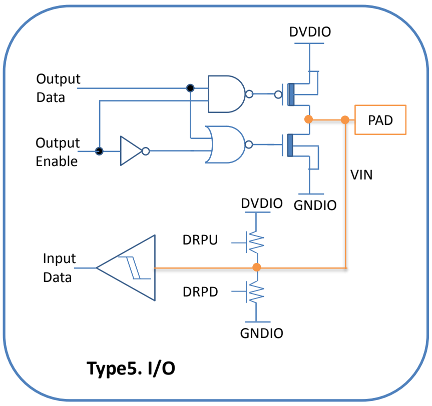




**Type1. GPIO 28 type**
**(2mA, 4mA, 6mA, 8mA)**
**Type2. GPIO 4G type**
**(4mA, 8mA, 12mA, 16mA)**

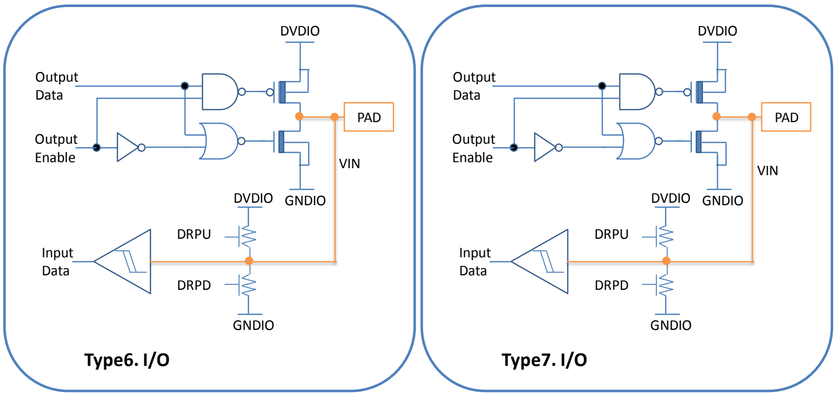
**Type 3. Open-drain IO**





**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 2-3. 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 | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| MSDC0_CLK | 0 | GPIO174 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 1 | MSDC0_CLK | O | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC0_CMD | 0 | GPIO172 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 1 | MSDC0_CMD | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC0_DAT0 | 0 | GPIO175 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 1 | MSDC0_DAT0 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC0_DAT1 | 0 | GPIO176 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 1 | MSDC0_DAT1 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC0_DAT2 | 0 | GPIO177 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 1 | MSDC0_DAT2 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC0_DAT3 | 0 | GPIO178 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 1 | MSDC0_DAT3 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/C U/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| MSDC0_DAT4 | 0 | GPIO179 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 1 | MSDC0_DAT4 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC0_DAT5 | 0 | GPIO180 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 1 | MSDC0_DAT5 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC0_DAT6 | 0 | GPIO181 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 1 | MSDC0_DAT6 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC0_DAT7 | 0 | GPIO182 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 1 | MSDC0_DAT7 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC0_RSTB | 0 | GPIO183 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 1 | MSDC0_RSTB | O | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC0_DSL | 0 | GPIO173 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 1 | MSDC0_DSL | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC1_CLK | 0 | GPIO167 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 1 | MSDC1_CLK | O | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 2 | LTE_MD32_JTAG_TCK | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 3 | C2K_TCK | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 4 | TDD_TCK | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 5 | CONN_DSP_JCK | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 6 | JTCK | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 7 | CONN_MCU_AICE_TCKC | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC1_CMD | 0 | GPIO166 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 1 | MSDC1_CMD | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 2 | LTE_MD32_JTAG_TMS | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 3 | C2K_TMS | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 4 | TDD_TMS | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 5 | CONN_DSP_JMS | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 6 | JTMS | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 7 | CONN_MCU_AICE_TMSC | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC1_DAT0 | 0 | GPIO168 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 1 | MSDC1_DAT0 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 2 | LTE_MD32_JTAG_TDI | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 3 | C2K_TDI | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 4 | TDD_TDI | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 5 | CONN_DSP_JDI | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 6 | JTDI | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC1_DAT1 | 0 | GPIO169 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
|  | 1 | MSDC1_DAT1 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| | 2 | LTE_MD32_JTAG_TDO | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 3 | C2K_TDO | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 4 | TDD_TDO | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 5 | CONN_DSP_JDO | O | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 6 | JTDO | O | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC1_DAT2 | 0 | GPIO170 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 1 | MSDC1_DAT2 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 2 | LTE_MD32_JTAG_TRST | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 3 | C2K_NTRST | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 4 | TDD_TRSTN | I | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 5 | CONN_DSP_JINTP | O | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 6 | DM_JTINTP | O | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| MSDC1_DAT3 | 0 | GPIO171 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 1 | MSDC1_DAT3 | IO | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 3 | C2K_RTCK | O | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| | 6 | TDD_TXD | O | CU, CD | 2/4/6/8/10/12/14/16mA | 0 |
| SIM1_SCLK | 0 | GPIO18 | IO | CU, CD | 4/8/12/16mA | 0 |
| | 1 | MD1_SIM1_SCLK | O | CU, CD | 4/8/12/16mA | 0 |
| | 2 | MD2_SIM1_SCLK | O | CU, CD | 4/8/12/16mA | 0 |
| | 3 | MD1_SIM2_SCLK | O | CU, CD | 4/8/12/16mA | 0 |
| | 4 | MD2_SIM2_SCLK | O | CU, CD | 4/8/12/16mA | 0 |
| SIM1_SRST | 0 | GPIO164 | IO | CU, CD | 4/8/12/16mA | 0 |
| | 1 | MD_SIM1_SRST | O | CU, CD | 4/8/12/16mA | 0 |
| | 2 | MD_SIM2_SRST | O | CU, CD | 4/8/12/16mA | 0 |
| | 3 | UIM1_RST | O | CU, CD | 4/8/12/16mA | 0 |
| | 4 | UIM0_RST | O | CU, CD | 4/8/12/16mA | 0 |
| SIM1_SIO | 0 | GPIO165 | IO | CU, CD | 4/8/12/16mA | 0 |
| | 1 | MD_SIM1_SDAT | IO | CU, CD | 4/8/12/16mA | 0 |
| | 2 | MD_SIM2_SDAT | IO | CU, CD | 4/8/12/16mA | 0 |
| | 3 | UIM1_IO | IO | CU, CD | 4/8/12/16mA | 0 |
| | 4 | UIM0_IO | IO | CU, CD | 4/8/12/16mA | 0 |
| SIM2_SCLK | 0 | GPIO160 | IO | CU, CD | 4/8/12/16mA | 0 |
| | 1 | MD_SIM2_SCLK | O | CU, CD | 4/8/12/16mA | 0 |
| | 2 | MD_SIM1_SCLK | O | CU, CD | 4/8/12/16mA | 0 |
| | 3 | UIM0_CLK | O | CU, CD | 4/8/12/16mA | 0 |
| | 4 | UIM1_CLK | O | CU, CD | 4/8/12/16mA | 0 |
| SIM2_SRST | 0 | GPIO161 | IO | CU, CD | 4/8/12/16mA | 0 |
| | 1 | MD_SIM2_SRST | O | CU, CD | 4/8/12/16mA | 0 |
| | 2 | MD_SIM1_SRST | O | CU, CD | 4/8/12/16mA | 0 |
| | 3 | UIM0_RST | O | CU, CD | 4/8/12/16mA | 0 |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| | 4 | UIM1_RST | O | CU, CD | 4/8/12/16mA | 0 |
| SIM2_SIO | 0 | GPIO162 | IO | CU, CD | 4/8/12/16mA | 0 |
| | 1 | MD_SIM2_SDAT | IO | CU, CD | 4/8/12/16mA | 0 |
| | 2 | MD_SIM1_SDAT | IO | CU, CD | 4/8/12/16mA | 0 |
| | 3 | UIM0_IO | IO | CU, CD | 4/8/12/16mA | 0 |
| | 4 | UIM1_IO | IO | CU, CD | 4/8/12/16mA | 0 |
| AUD_CLK_MOSI | 0 | GPIO143 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | AUD_CLK_MOSI | O | CU, CD | 2/4/6/8mA | 0 |
| AUD_DAT_MISO | 0 | GPIO144 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | AUD_DAT_MISO | I | CU, CD | 2/4/6/8mA | 0 |
| | 3 | AUD_DAT_MOSI | O | CU, CD | 2/4/6/8mA | 0 |
| AUD_DAT_MOSI | 0 | GPIO145 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | AUD_DAT_MOSI | O | CU, CD | 2/4/6/8mA | 0 |
| | 3 | AUD_DAT_MISO | I | CU, CD | 2/4/6/8mA | 0 |
| DISP_PWM | 0 | GPIO69 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | DISP_PWM | O | CU, CD | 2/4/6/8mA | 0 |
| | 2 | PWM1 | O | CU, CD | 2/4/6/8mA | 0 |
| | 3 | LTE_MD32_JTAG_TRST | I | CU, CD | 2/4/6/8mA | 0 |
| | 4 | TDD_TRSTN | I | CU, CD | 2/4/6/8mA | 0 |
| | 5 | ANT_SEL7 | O | CU, CD | 2/4/6/8mA | 0 |
| | 6 | DM_JTINTP | O | CU, CD | 2/4/6/8mA | 0 |
| LCM_RST | 0 | GPIO146 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | LCM_RST | O | CU, CD | 2/4/6/8mA | 0 |
| DSI_TE | 0 | GPIO147 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | DSI_TE | I | CU, CD | 2/4/6/8mA | 0 |
| CMDAT0 | 0 | GPIO42 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | CMDAT0 | I | CU, CD | 2/4/6/8mA | 0 |
| | 2 | CMCSD0 | I | CU, CD | 2/4/6/8mA | 0 |
| | 3 | CMMCLK1 | O | CU, CD | 2/4/6/8mA | 0 |
| | 6 | ANT_SEL5 | O | CU, CD | 2/4/6/8mA | 0 |
| | 7 | CLKM5 | O | CU, CD | 2/4/6/8mA | 0 |
| CMDAT1 | 0 | GPIO43 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | CMDAT1 | I | CU, CD | 2/4/6/8mA | 0 |
| | 2 | CMCSD1 | I | CU, CD | 2/4/6/8mA | 0 |
| | 3 | CMFLASH | O | CU, CD | 2/4/6/8mA | 0 |
| | 4 | MD_EINT0 | I | CU, CD | 2/4/6/8mA | 0 |
| | 5 | CMMCLK1 | O | CU, CD | 2/4/6/8mA | 0 |
| | 6 | CLKM4 | O | CU, CD | 2/4/6/8mA | 0 |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| | 7 | DBG_MON_A10 | IO | CU, CD | 2/4/6/8mA | 0 |
| CMPCLK | 0 | GPIO44 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | CMPCLK | I | CU, CD | 2/4/6/8mA | 0 |
| | 2 | CMCSK | I | CU, CD | 2/4/6/8mA | 0 |
| | 3 | CMCSD2 | I | CU, CD | 2/4/6/8mA | 0 |
| | 4 | KCOL3 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 5 | SRCLKENAI2 | I | CU, CD | 2/4/6/8mA | 0 |
| | 6 | PWM0 | O | CU, CD | 2/4/6/8mA | 0 |
| | 7 | DBG_MON_A11 | IO | CU, CD | 2/4/6/8mA | 0 |
| CMMCLK | 0 | GPIO45 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | CMMCLK0 | O | CU, CD | 2/4/6/8mA | 0 |
| | 7 | DBG_MON_A12 | IO | CU, CD | 2/4/6/8mA | 0 |
| CMMCLK1 | 0 | GPIO46 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | CMMCLK1 | O | CU, CD | 2/4/6/8mA | 0 |
| | 2 | IDDIG | I | CU, CD | 2/4/6/8mA | 0 |
| | 3 | LTE_MD32_JTAG_TRST | I | CU, CD | 2/4/6/8mA | 0 |
| | 4 | TDD_TRSTN | I | CU, CD | 2/4/6/8mA | 0 |
| | 5 | DM_JTINTP | O | CU, CD | 2/4/6/8mA | 0 |
| | 6 | KCOL6 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 7 | DBG_MON_A13 | IO | CU, CD | 2/4/6/8mA | 0 |
| PWRAP_SPI0_CK | 0 | GPIO141 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | PWRAP_SPICK_I | O | CU, CD | 2/4/6/8mA | 0 |
| PWRAP_SPI0_CSN | 0 | GPIO142 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | PWRAP_SPICS_B_I | O | CU, CD | 2/4/6/8mA | 0 |
| PWRAP_SPI0_MI | 0 | GPIO138 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | PWRAP_SPIDO | IO | CU, CD | 2/4/6/8mA | 0 |
| | 2 | PWRAP_SPIDI | IO | CU, CD | 2/4/6/8mA | 0 |
| PWRAP_SPI0_MO | 0 | GPIO139 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | PWRAP_SPIDI | IO | CU, CD | 2/4/6/8mA | 0 |
| | 2 | PWRAP_SPIDO | IO | CU, CD | 2/4/6/8mA | 0 |
| WATCHDOG | 0 | GPIO149 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | WATCHDOG | O | CU, CD | 2/4/6/8mA | 0 |
| SRCLKENA0 | 0 | GPIO148 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | SRCLKENA0 | O | CU, CD | 2/4/6/8mA | 0 |
| SRCLKENA1 | 0 | GPIO56 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | SRCLKENA1 | O | CU, CD | 2/4/6/8mA | 0 |
| SCL0 | 0 | GPIO48 | IO | CD | | 0 |
| | 1 | SCL0_0 | IO | CD | | 0 |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| SCL1 | 0 | GPIO50 | IO | CD | | 0 |
| | 1 | SCL1_0 | IO | CD | | 0 |
| SCL2 | 0 | GPIO52 | IO | CD | | 0 |
| | 1 | SCL2_0 | IO | CD | | 0 |
| SCL3 | 0 | GPIO54 | IO | CD | | 0 |
| | 1 | SCL3_0 | IO | CD | | 0 |
| | 3 | IDDIG | I | CD | | 0 |
| SDA0 | 0 | GPIO47 | IO | CD | | 0 |
| | 1 | SDA0_0 | IO | CD | | 0 |
| SDA1 | 0 | GPIO49 | IO | CD | | 0 |
| | 1 | SDA1_0 | IO | CD | | 0 |
| SDA2 | 0 | GPIO51 | IO | CD | | 0 |
| | 1 | SDA2_0 | IO | CD | | 0 |
| SDA3 | 0 | GPIO53 | IO | CD | | 0 |
| | 1 | SDA3_0 | IO | CD | | 0 |
| | 3 | IDDIG | I | CD | | 0 |
| RFIC0_BSI_EN | 0 | GPIO110 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | RFIC0_BSI_EN | O | CU, CD | 2/4/6/8mA | 0 |
| | 4 | SPM_BSI_EN | O | CU, CD | 2/4/6/8mA | 0 |
| RFIC0_BSI_CK | 0 | GPIO111 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | RFIC0_BSI_CK | O | CU, CD | 2/4/6/8mA | 0 |
| | 4 | SPM_BSI_CLK | O | CU, CD | 2/4/6/8mA | 0 |
| RFIC0_BSI_D0 | 0 | GPIO112 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | RFIC0_BSI_D0 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 4 | SPM_BSI_D2 | IO | CU, CD | 2/4/6/8mA | 0 |
| RFIC0_BSI_D1 | 0 | GPIO113 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | RFIC0_BSI_D1 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 4 | SPM_BSI_D1 | IO | CU, CD | 2/4/6/8mA | 0 |
| RFIC0_BSI_D2 | 0 | GPIO114 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | RFIC0_BSI_D2 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 4 | SPM_BSI_D0 | IO | CU, CD | 2/4/6/8mA | 0 |
| RFIC1_BSI_EN | 0 | GPIO104 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | RFIC1_BSI_EN | O | CU, CD | 2/4/6/8mA | 0 |
| | 5 | C2K_RX_BSI_EN | O | CU, CD | 2/4/6/8mA | 0 |
| RFIC1_BSI_CK | 0 | GPIO105 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | RFIC1_BSI_CK | O | CU, CD | 2/4/6/8mA | 0 |
| | 2 | PCM1_SYNC | IO | CU, CD | 2/4/6/8mA | 0 |
| | 5 | C2K_RX_BSI_CLK | O | CU, CD | 2/4/6/8mA | 0 |
| RFIC1_BSI_D0 | 0 | GPIO106 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | RFIC1_BSI_D0 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 5 | C2K_RX_BSI_DATA | IO | CU, CD | 2/4/6/8mA | 0 |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| RFIC1_TX_BSI_EN | 0 | GPIO107 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | RFIC1_BSI_D1 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 5 | C2K_TX_BSI_EN | O | CU, CD | 2/4/6/8mA | 0 |
| RFIC1_TX_BSI_CK | 0 | GPIO108 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | RFIC1_BSI_D2 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 5 | C2K_TX_BSI_CLK | O | CU, CD | 2/4/6/8mA | 0 |
| RFIC1_TX_BSI_D0 | 0 | GPIO109 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 5 | C2K_TX_BSI_DATA | IO | CU, CD | 2/4/6/8mA | 0 |
| PAD_RFIC_MIPI1_SCLK | 0 | GPIO133 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | MIPI1_SCLK | O | CU, CD | 2/4/6/8mA | 0 |
| PAD_RFIC_MIPI1_SDATA | 0 | GPIO134 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | MIPI1_SDATA | IO | CU, CD | 2/4/6/8mA | 0 |
| PAD_RFIC_MIPI0_SCLK | 0 | GPIO135 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | MIPI0_SCLK | O | CU, CD | 2/4/6/8mA | 0 |
| PAD_RFIC_MIPI0_SDATA<br>B | 0 | GPIO136 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | MIPI0_SDATA | IO | CU, CD | 2/4/6/8mA | 0 |
| BPI_BUS0 | 0 | GPIO119 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | BPI_BUS0 | O | CU, CD | 2/4/6/8mA | 0 |
|  | 7 | DBG_MON_B28 | IO | CU, CD | 2/4/6/8mA | 0 |
| BPI_BUS1 | 0 | GPIO120 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | BPI_BUS1 | O | CU, CD | 2/4/6/8mA | 0 |
|  | 7 | DBG_MON_B29 | IO | CU, CD | 2/4/6/8mA | 0 |
| BPI_BUS2 | 0 | GPIO121 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | BPI_BUS2 | O | CU, CD | 2/4/6/8mA | 0 |
|  | 7 | DBG_MON_B30 | IO | CU, CD | 2/4/6/8mA | 0 |
| BPI_BUS3 | 0 | GPIO122 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | BPI_BUS3 | O | CU, CD | 2/4/6/8mA | 0 |
|  | 7 | DBG_MON_B31 | IO | CU, CD | 2/4/6/8mA | 0 |
| BPI_BUS4 | 0 | GPIO123 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | BPI_BUS4 | O | CU, CD | 2/4/6/8mA | 0 |
|  | 7 | DBG_MON_B32 | IO | CU, CD | 2/4/6/8mA | 0 |
| BPI_BUS5 | 0 | GPIO87 | IO | CU, CD | 2/4/6/8mA | 0 |
|  | 1 | BPI_BUS5 | O | CU, CD | 2/4/6/8mA | 0 |
|  | 2 | LTE_C2K_BPI_BUS5 | O | CU, CD | 2/4/6/8mA | 0 |
|  | 5 | C2K_BPI_BUS5 | O | CU, CD | 2/4/6/8mA | 0 |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| | 7 | DBG_MON_B0 | IO | CU, CD | 2/4/6/8mA | 0 |
| BPI_BUS6 | 0 | GPIO88 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | BPI_BUS6 | O | CU, CD | 2/4/6/8mA | 0 |
| | 2 | LTE_C2K_BPI_BUS6 | O | CU, CD | 2/4/6/8mA | 0 |
| | 3 | Reserved | - | CU, CD | 2/4/6/8mA | 0 |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | 0 |
| | 5 | C2K_BPI_BUS6 | O | CU, CD | 2/4/6/8mA | 0 |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | 0 |
| | 7 | DBG_MON_B1 | IO | CU, CD | 2/4/6/8mA | 0 |
| BPI_BUS7 | 0 | GPIO89 | IO | CU, CD | 2/4/6/8mA | 0 |
| | 1 | BPI_BUS7 | O | CU, CD | 2/4/6/8mA | 0 |
| | 2 | LTE_C2K_BPI_BUS7 | O | CU, CD | 2/4/6/8mA | 0 |
| | 3 | CLKM0 | O | CU, CD | 2/4/6/8mA | 0 |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_BPI_BUS7 | O | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_B2 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS8 | - | GPIO9- | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | BPI_BUS8 | O | CU, CD | 2/4/6/8mA | - |
| | 2 | LTE_C2K_BPI_BUS8 | O | CU, CD | 2/4/6/8mA | - |
| | 3 | CLKM1 | O | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_BPI_BUS8 | O | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_B3 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS9 | - | GPIO91 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | BPI_BUS9 | O | CU, CD | 2/4/6/8mA | - |
| | 2 | LTE_C2K_BPI_BUS9 | O | CU, CD | 2/4/6/8mA | - |
| | 3 | CLKM2 | O | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_BPI_BUS9 | O | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_B4 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS1- | - | GPIO92 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | BPI_BUS1- | O | CU, CD | 2/4/6/8mA | - |
| | 2 | LTE_C2K_BPI_BUS1- | O | CU, CD | 2/4/6/8mA | - |
| | 3 | CLKM3 | O | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_BPI_BUS1- | O | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
|  | 7 | DBG_MON_B5 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS11 | - | GPIO93 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | BPI_BUS11 | O | CU, CD | 2/4/6/8mA | - |
|  | 2 | LTE_C2K_BPI_BUS11 | O | CU, CD | 2/4/6/8mA | - |
|  | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 5 | C2K_BPI_BUS11 | O | CU, CD | 2/4/6/8mA | - |
|  | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 7 | DBG_MON_B6 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS12 | - | GPIO94 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | BPI_BUS12 | O | CU, CD | 2/4/6/8mA | - |
|  | 2 | LTE_C2K_BPI_BUS12 | O | CU, CD | 2/4/6/8mA | - |
|  | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 5 | C2K_BPI_BUS12 | O | CU, CD | 2/4/6/8mA | - |
|  | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 7 | DBG_MON_B7 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS13 | - | GPIO95 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | BPI_BUS13 | O | CU, CD | 2/4/6/8mA | - |
|  | 2 | LTE_C2K_BPI_BUS13 | O | CU, CD | 2/4/6/8mA | - |
|  | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 5 | C2K_BPI_BUS13 | O | CU, CD | 2/4/6/8mA | - |
|  | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 7 | DBG_MON_B8 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS14 | - | GPIO96 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | BPI_BUS14 | O | CU, CD | 2/4/6/8mA | - |
|  | 2 | LTE_C2K_BPI_BUS14 | O | CU, CD | 2/4/6/8mA | - |
|  | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 5 | C2K_BPI_BUS14 | O | CU, CD | 2/4/6/8mA | - |
|  | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 7 | DBG_MON_B9 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS15 | - | GPIO97 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | BPI_BUS15 | O | CU, CD | 2/4/6/8mA | - |
|  | 2 | LTE_C2K_BPI_BUS15 | O | CU, CD | 2/4/6/8mA | - |
|  | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_BPI_BUS15 | O | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_B1- | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS16 | - | GPIO98 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | BPI_BUS16 | O | CU, CD | 2/4/6/8mA | - |
| | 2 | LTE_C2K_BPI_BUS16 | O | CU, CD | 2/4/6/8mA | - |
| | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_BPI_BUS16 | O | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_B11 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS17 | - | GPIO99 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | BPI_BUS17 | O | CU, CD | 2/4/6/8mA | - |
| | 2 | LTE_C2K_BPI_BUS17 | O | CU, CD | 2/4/6/8mA | - |
| | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_BPI_BUS17 | O | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_B12 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS18 | - | GPIO1-- | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | BPI_BUS18 | O | CU, CD | 2/4/6/8mA | - |
| | 2 | LTE_C2K_BPI_BUS18 | O | CU, CD | 2/4/6/8mA | - |
| | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_BPI_BUS18 | O | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_B13 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS19 | - | GPIO1-1 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | BPI_BUS19 | O | CU, CD | 2/4/6/8mA | - |
| | 2 | LTE_C2K_BPI_BUS19 | O | CU, CD | 2/4/6/8mA | - |
| | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_BPI_BUS19 | O | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_B14 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS2- | - | GPIO1-2 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | BPI_BUS2- | O | CU, CD | 2/4/6/8mA | - |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
|  | 2 | LTE_C2K_BPI_BUS2- | O | CU, CD | 2/4/6/8mA | - |
|  | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 5 | C2K_BPI_BUS2- | O | CU, CD | 2/4/6/8mA | - |
|  | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 7 | DBG_MON_B15 | IO | CU, CD | 2/4/6/8mA | - |
| BPI_BUS21 | - | GPIO124 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | BPI_BUS21 | O | CU, CD | 2/4/6/8mA | - |
|  | 2 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 5 | DPI_HSYNC1 | O | CU, CD | 2/4/6/8mA | - |
|  | 6 | KCOL2 | IO | CU, CD | 2/4/6/8mA | - |
|  | 7 | TDD_TXD | O | CU, CD | 2/4/6/8mA | - |
| BPI_BUS22 | - | GPIO125 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | BPI_BUS22 | O | CU, CD | 2/4/6/8mA | - |
|  | 2 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 5 | DPI_VSYNC1 | O | CU, CD | 2/4/6/8mA | - |
|  | 6 | KROW2 | IO | CU, CD | 2/4/6/8mA | - |
|  | 7 | MD_URXD | I | CU, CD | 2/4/6/8mA | - |
| BPI_BUS23 | - | GPIO126 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | BPI_BUS23 | O | CU, CD | 2/4/6/8mA | - |
|  | 2 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 5 | DPI_CK1 | O | CU, CD | 2/4/6/8mA | - |
|  | 6 | I2S2_MCK | O | CU, CD | 2/4/6/8mA | - |
|  | 7 | MD_UTXD | O | CU, CD | 2/4/6/8mA | - |
| BPI_BUS24 | - | GPIO127 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | BPI_BUS24 | O | CU, CD | 2/4/6/8mA | - |
|  | 2 | Reserved | - | CU, CD | 2/4/6/8mA | - |
|  | 3 | CONN_MCU_DBGI_N | I | CU, CD | 2/4/6/8mA | - |
|  | 4 | EXT_FRAME_SYNC | I | CU, CD | 2/4/6/8mA | - |
|  | 5 | DPI_DE1 | O | CU, CD | 2/4/6/8mA | - |
|  | 6 | SRCLKENAI1 | I | CU, CD | 2/4/6/8mA | - |
|  | 7 | URXD- | I | CU, CD | 2/4/6/8mA | - |
| BPI_BUS25 | - | GPIO128 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | BPI_BUS25 | O | CU, CD | 2/4/6/8mA | - |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| | 2 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 3 | GPS_FRAME_SYNC | O | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | I2S2_DI | I | CU, CD | 2/4/6/8mA | - |
| | 6 | PTA_RXD | I | CU, CD | 2/4/6/8mA | - |
| | 7 | UTXD- | O | CU, CD | 2/4/6/8mA | - |
| BPI_BUS26 | - | GPIO129 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | BPI_BUS26 | O | CU, CD | 2/4/6/8mA | - |
| | 2 | DISP_PWM | O | CU, CD | 2/4/6/8mA | - |
| | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | I2S2_LRCK | O | CU, CD | 2/4/6/8mA | - |
| | 6 | PTA_TXD | O | CU, CD | 2/4/6/8mA | - |
| | 7 | LTE_URXD | I | CU, CD | 2/4/6/8mA | - |
| BPI_BUS27 | - | GPIO13- | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | BPI_BUS27 | O | CU, CD | 2/4/6/8mA | - |
| | 2 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | I2S2_BCK | O | CU, CD | 2/4/6/8mA | - |
| | 6 | IRTX_OUT | O | CU, CD | 2/4/6/8mA | - |
| | 7 | LTE_UTXD | O | CU, CD | 2/4/6/8mA | - |
| LTE_TXBPI | - | GPIO118 | IO | CU, CD | 4/8/12/16mA | - |
| | 1 | TXBPI | I | CU, CD | 4/8/12/16mA | - |
| C2K_TXBPI | - | GPIO1-3 | IO | CU, CD | 4/8/12/16mA | - |
| | 1 | C2K_TXBPI | I | CU, CD | 4/8/12/16mA | - |
| WB_CTRL- | - | GPIO13 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | WB_CTRL- | IO | CU, CD | 2/4/6/8mA | - |
| | 3 | C2K_ARM_EINT- | IO | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_A- | IO | CU, CD | 2/4/6/8mA | - |
| WB_CTRL1 | - | GPIO14 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | WB_CTRL1 | IO | CU, CD | 2/4/6/8mA | - |
| | 3 | C2K_ARM_EINT1 | IO | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_A1 | IO | CU, CD | 2/4/6/8mA | - |
| WB_CTRL2 | - | GPIO15 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | WB_CTRL2 | IO | CU, CD | 2/4/6/8mA | - |
| | 3 | C2K_ARM_EINT2 | IO | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_A2 | IO | CU, CD | 2/4/6/8mA | - |
| WB_CTRL3 | - | GPIO16 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | WB_CTRL3 | IO | CU, CD | 2/4/6/8mA | - |
| | 3 | C2K_ARM_EINT3 | IO | CU, CD | 2/4/6/8mA | - |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
|  | 7 | DBG_MON_A3 | IO | CU, CD | 2/4/6/8mA | - |
| WB_CTRL4 | - | GPIO17 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | WB_CTRL4 | IO | CU, CD | 2/4/6/8mA | - |
|  | 3 | C2K_DM_EINT- | IO | CU, CD | 2/4/6/8mA | - |
|  | 4 | WATCHDOG | O | CU, CD | 2/4/6/8mA | - |
|  | 7 | DBG_MON_A4 | IO | CU, CD | 2/4/6/8mA | - |
| WB_CTRL5 | - | GPIO18 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | WB_CTRL5 | IO | CU, CD | 2/4/6/8mA | - |
|  | 2 | C2K_DM_EINT1 | IO | CU, CD | 2/4/6/8mA | - |
|  | 7 | DBG_MON_A5 | IO | CU, CD | 2/4/6/8mA | - |
| F2W_DATA | - | GPIO184 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | F2W_DATA | I | CU, CD | 2/4/6/8mA | - |
|  | 2 | MRG_CLK | O | CU, CD | 2/4/6/8mA | - |
|  | 3 | C2K_DM_EINT2 | IO | CU, CD | 2/4/6/8mA | - |
|  | 4 | PCM-_CLK | O | CU, CD | 2/4/6/8mA | - |
| F2W_CLK | - | GPIO185 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | F2W_CK | I | CU, CD | 2/4/6/8mA | - |
|  | 2 | MRG_DI | I | CU, CD | 2/4/6/8mA | - |
|  | 3 | C2K_DM_EINT3 | IO | CU, CD | 2/4/6/8mA | - |
|  | 4 | PCM-_DI | I | CU, CD | 2/4/6/8mA | - |
| WB_SCLK | - | GPIO187 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | WB_SCLK | O | CU, CD | 2/4/6/8mA | - |
|  | 2 | MRG_DO | O | CU, CD | 2/4/6/8mA | - |
|  | 4 | PCM-_DO | O | CU, CD | 2/4/6/8mA | - |
| WB_SDATA | - | GPIO188 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | WB_SDATA | IO | CU, CD | 2/4/6/8mA | - |
|  | 2 | MRG_SYNC | O | CU, CD | 2/4/6/8mA | - |
|  | 4 | PCM-_SYNC | O | CU, CD | 2/4/6/8mA | - |
| WB_SEN | - | GPIO189 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | WB_SEN | O | CU, CD | 2/4/6/8mA | - |
|  | 4 | UTXD3 | O | CU, CD | 2/4/6/8mA | - |
|  | 5 | URXD3 | I | CU, CD | 2/4/6/8mA | - |
| WB_RSTB | - | GPIO186 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | WB_RSTB | O | CU, CD | 2/4/6/8mA | - |
|  | 4 | URXD3 | I | CU, CD | 2/4/6/8mA | - |
|  | 5 | UTXD3 | O | CU, CD | 2/4/6/8mA | - |
| KPCOL- | - | GPIO84 | IO | CU, CD | 2/4/6/8mA | - |
|  | 1 | KCOL- | IO | CU, CD | 2/4/6/8mA | - |
|  | 2 | URTS- | O | CU, CD | 2/4/6/8mA | - |
|  | 3 | CONN_MCU_DBGACK_N | O | CU, CD | 2/4/6/8mA | - |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| | 4 | SCL2 | IO | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_TDO | IO | CU, CD | 2/4/6/8mA | - |
| | 6 | AUXIF_CLK | O | CU, CD | 2/4/6/8mA | - |
| | 7 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| KPCOL1 | - | GPIO85 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | KCOL1 | IO | CU, CD | 2/4/6/8mA | - |
| | 2 | UCTS- | I | CU, CD | 2/4/6/8mA | - |
| | 3 | UCTS1 | I | CU, CD | 2/4/6/8mA | - |
| | 4 | SDA2 | IO | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_TMS | I | CU, CD | 2/4/6/8mA | - |
| | 6 | AUXIF_ST | O | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_A31 | IO | CU, CD | 2/4/6/8mA | - |
| KPCOL2 | - | GPIO86 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | KCOL2 | IO | CU, CD | 2/4/6/8mA | - |
| | 2 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 3 | URTS1 | O | CU, CD | 2/4/6/8mA | - |
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_RTCK | O | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_A32 | IO | CU, CD | 2/4/6/8mA | - |
| KPROW- | - | GPIO81 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | KROW- | IO | CU, CD | 2/4/6/8mA | - |
| | 2 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 3 | CONN_MCU_DBGI_N | I | CU, CD | 2/4/6/8mA | - |
| | 4 | CORESONIC_SWCK | I | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_TCK | I | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | C2K_DM_EINT1 | IO | CU, CD | 2/4/6/8mA | - |
| KPROW1 | - | GPIO82 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | KROW1 | IO | CU, CD | 2/4/6/8mA | - |
| | 2 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 3 | CONN_MCU_TRST_B | I | CU, CD | 2/4/6/8mA | - |
| | 4 | CORESONIC_SWD | IO | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_NTRST | I | CU, CD | 2/4/6/8mA | - |
| | 6 | USB_DRVVBUS | O | CU, CD | 2/4/6/8mA | - |
| | 7 | C2K_DM_EINT2 | IO | CU, CD | 2/4/6/8mA | - |
| KPROW2 | - | GPIO83 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | KROW2 | IO | CU, CD | 2/4/6/8mA | - |
| | 2 | USB_DRVVBUS | O | CU, CD | 2/4/6/8mA | - |
| | 3 | Reserved | - | CU, CD | 2/4/6/8mA | - |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| | 4 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 5 | C2K_TDI | I | CU, CD | 2/4/6/8mA | - |
| | 6 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 7 | C2K_DM_EINT3 | IO | CU, CD | 2/4/6/8mA | - |
| SRCLKENAI | - | GPIO55 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | SRCLKENAI- | I | CU, CD | 2/4/6/8mA | - |
| | 2 | PWM2 | O | CU, CD | 2/4/6/8mA | - |
| | 3 | CLKM5 | O | CU, CD | 2/4/6/8mA | - |
| | 4 | CORESONIC_SWD | IO | CU, CD | 2/4/6/8mA | - |
| | 5 | ANT_SEL6 | O | CU, CD | 2/4/6/8mA | - |
| | 6 | KROW5 | IO | CU, CD | 2/4/6/8mA | - |
| | 7 | DISP_PWM | O | CU, CD | 2/4/6/8mA | - |
| SPI_CSB | - | GPIO65 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | SPI_CSA | O | CU, CD | 2/4/6/8mA | - |
| | 2 | EXT_FRAME_SYNC | I | CU, CD | 2/4/6/8mA | - |
| | 3 | I2S3_MCK | O | CU, CD | 2/4/6/8mA | - |
| | 4 | KROW2 | IO | CU, CD | 2/4/6/8mA | - |
| | 5 | GPS_FRAME_SYNC | O | CU, CD | 2/4/6/8mA | - |
| | 6 | PTA_RXD | I | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_A22 | IO | CU, CD | 2/4/6/8mA | - |
| SPI_CLK | - | GPIO66 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | SPI_CKA | O | CU, CD | 2/4/6/8mA | - |
| | 2 | USB_DRVVBUS | O | CU, CD | 2/4/6/8mA | - |
| | 3 | I2S3_BCK | O | CU, CD | 2/4/6/8mA | - |
| | 4 | KCOL2 | IO | CU, CD | 2/4/6/8mA | - |
| | 5 | Reserved | - | CU, CD | 2/4/6/8mA | - |
| | 6 | PTA_TXD | O | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_A23 | IO | CU, CD | 2/4/6/8mA | - |
| SPI_MO | - | GPIO68 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | SPI_MOA | O | CU, CD | 2/4/6/8mA | - |
| | 2 | SPI_MIA | I | CU, CD | 2/4/6/8mA | - |
| | 3 | I2S3_LRCK | O | CU, CD | 2/4/6/8mA | - |
| | 4 | PTA_TXD | O | CU, CD | 2/4/6/8mA | - |
| | 5 | ANT_SEL4 | O | CU, CD | 2/4/6/8mA | - |
| | 6 | URTS1 | O | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_A25 | IO | CU, CD | 2/4/6/8mA | - |
| SPI_MI | - | GPIO67 | IO | CU, CD | 2/4/6/8mA | - |
| | 1 | SPI_MIA | I | CU, CD | 2/4/6/8mA | - |
| | 2 | SPI_MOA | O | CU, CD | 2/4/6/8mA | - |
| | 3 | I2S3_DO | O | CU, CD | 2/4/6/8mA | - |
| | 4 | PTA_RXD | I | CU, CD | 2/4/6/8mA | - |

| Name | Aux. function | Aux. name | Aux. type | PU/PD/CU/CD | Driving | SMT |
|---|---|---|---|---|---|---|
| | 5 | IDDIG | I | CU, CD | 2/4/6/8mA | - |
| | 6 | UCTS1 | I | CU, CD | 2/4/6/8mA | - |
| | 7 | DBG_MON_A24 | IO | CU, CD | 2/4/6/8mA | - |

## 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 |
|---|---|---|---|---|
| AVDD18_PLLGP<br>AVDD18_MEMPLL<br>AVDD18_MDPLLGP | Analog power input 1.8V for PLL | 1.7 | 1.9 | V |
| AVDD18_AP | Analog power input 1.8V for AuxADC, TSENSE | 1.7 | 1.9 | V |
| AVDD18_MD | Analog power input 1.8V for BBTX, BBRX | 1.7 | 1.9 | V |
| AVDD28_DAC | Analog power input 2.8V for APC | 2.66 | 2.94 | V |
| DVDD18_MIPITX1 | Analog power for MIPI DSI | 1.7 | 1.9 | V |
| DVDD18_MIPIRX-<br>DVDD18_MIPIRX1 | Analog power for MIPI CSI- & CSI1 | 1.7 | 1.9 | V |
| AVDD33_USB_P-<br>AVDD33_USB_P1 | Analog power 3.3V for USB | 3.135 | 3.465 | V |
| AVDD18_USB | Analog power 1.8V for USB | 1.7 | 1.9 | V |
| AVDD18_WBG | Analog power 1.8V for connectivity ABB | 1.7 | 1.9 | V |
| DVDD18_IO!<br>DVDD18_IO2<br>DVDD18_IO3<br>DVDD18_BIAS1<br>DVDD18_BIAS2<br>DVDD18_BIAS3 | Digital power input for 1.8V IO | 1.62 | 1.98 | V |
| DVDD28_BPI1<br>DVDD28_BPI2 | Digital power input for BPI | 1.7 | 3.6 | V |
| DVDD18_MSDC- | Digital power input for MSDC- | 1.62 | 1.98 | V |
| DVDD28_MSDC1 | Digital power input for MSDC1 | 1.7 | 3.6 | V |
| DVDD28_SIM1 | Digital power input for SIM1 | 1.7 | 3.3 | V |
| DVDD28_SIM2 | Digital power input for SIM2 | 1.7 | 3.3 | V |
| DDRV<br>DDRV_CLK<br>DDRV_VREF | Digital power input for DRAM | 1.14 | 1.3 | V |
| DVDD_DVFS | Digital power input for DVFS | 0.95 | 1.31 | V |
| DVDD_LTE | Digital power input for LTE | 0.95 | 1.155 | V |
| DVDD_SRAM | Digital power input for SRAM | 0.95 | 1.31 | V |
| DVDD_TOP | Digital power input for TOP | 0.95 | 1.31 | V |

***Warning:*** *Stressing the device beyond the absolute maximum ratings may cause permanent damage. These are stress ratings only.*

### 2.2.2 Recommended Operating Conditions

***Table 2-9. Recommended operating conditions for power supply***

| Symbol or pin name | Description | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| AVDD18_PLLGP<br>AVDD18_MEMPLL<br>AVDD18_MDPLLGP | Analog power input 1.8V for PLL | 1.7 | 1.8 | 1.9 | V |
| AVDD18_AP | Analog power input 1.8V for AuxADC, TSENSE | 1.71 | 1.8 | 1.9 | V |
| AVDD18_MD | Analog power input 1.8V for BBTX, BBRX | 1.71 | 1.8 | 1.9 | V |
| AVDD28_DAC | Analog power input 2.8V for APC | 2.66 | 2.8 | 2.94 | V |
| DVDD18_MIPITX1 | Analog power for MIPI DSI | 1.71 | 1.8 | 1.89 | V |
| AVDD33_USB_P-<br>AVDD33_USB_P1 | Analog power for MIPI CSI- & CSI1 | 1.71 | 1.8 | 1.89 | V |
| AVDD33_USB | Analog power 3.3V for USB | 3.135 | 3.3 | 3.465 | V |
| AVDD18_USB | Analog power 1.8V for USB | 1.71 | 1.8 | 1.89 | V |
| AVDD18_WBG | Analog power 1.8V for connectivity ABB | 1.71 | 1.8 | 1.89 | V |
| DVDD28_BPI1<br>DVDD28_BPI2 | Digital power input for BPI | 1.7 | 1.8 | 1.95 | V |
| | | 2.66 | 2.8 | 2.94 | |
| DVDD18_IO!<br>DVDD18_IO2<br>DVDD18_IO3<br>DVDD18_BIAS1<br>DVDD18_BIAS2<br>DVDD18_BIAS3 | Digital power input for 1.8V IO | 1.62 | 1.8 | 1.98 | V |
| DVDD18_MSDC- | Digital power input for MSDC- | 1.62 | 1.8 | 1.98 | V |
| DVDD28_MSDC1 | Digital power input for MSDC1 | 1.7 | 1.8 | 1.95 | V |
| | | 2.7 | 3.3 | 3.6 | |
| DVDD28_SIM1<br>DVDD28_SIM2 | Digital power input for SIM1/SIM2 | 2.7 | 3.3 | 3.6 | V |
| | | 1.7 | 1.8 | 1.9 | |
| DDRV<br>DDRV_CLK<br>DDRV_VREF | Digital power input for EMI (LPDDR2/3) | 1.14 | 1.2 | 1.3 | V |
| DVDD_DVFS | Digital power input for DVFS | 0.95 | 1.15 | 1.31 | V |
| DVDD_LTE | Digital power input for LTE | 0.95 | 1.05- | 1.155 | V |
| DVDD_SRAM | Digital power input for SRAM | 0.95 | 1.15 | 1.31 | V |
| DVDD_TOP | Digital power input for TOP | 0.95 | 1.15 | 1.31 | V |

### 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 AC Electrical Characteristics and Timing Diagram

### 2.2.4.1 External Memory Interface for LPDDR3

The external memory interface, shown in Figure 2-4, Figure 2-5 and Figure 2-6, is used to connect LPDDR3 device for MT6737. 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 2-10 summarizes the symbol definition and the related timing specifications.

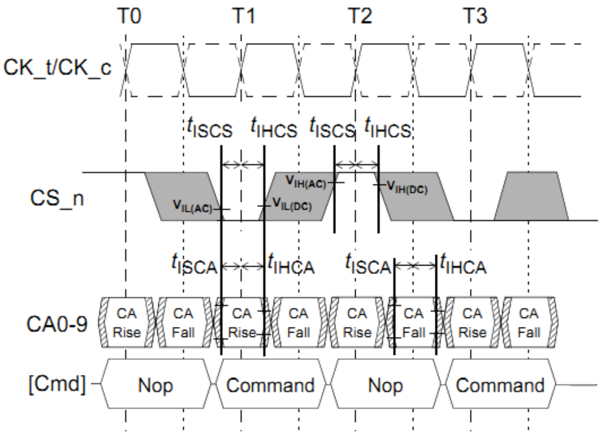


*Figure 2-4. Basic timing parameter for LPDDR3 commands*

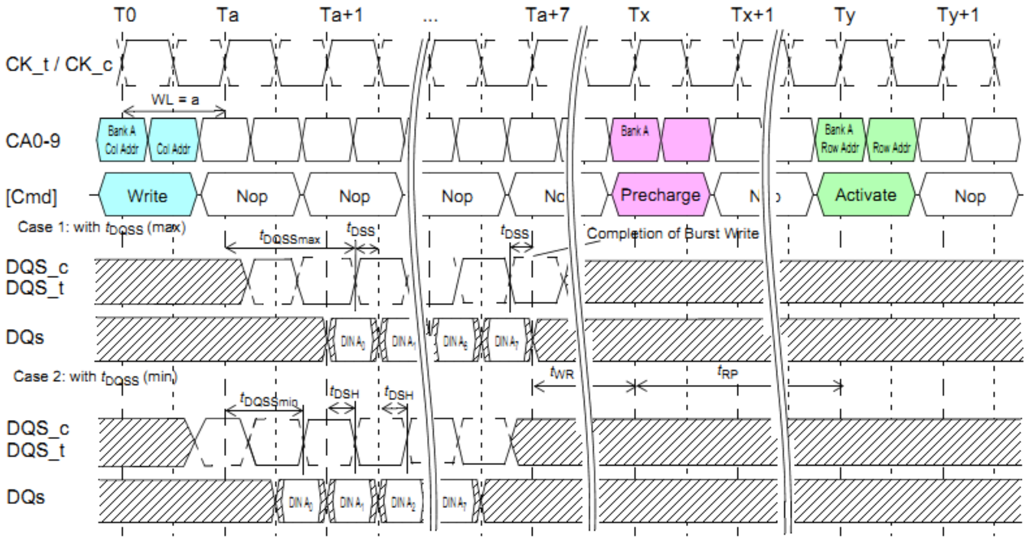


*Figure 2-5. Basic timing parameter for LPDDR3 write*

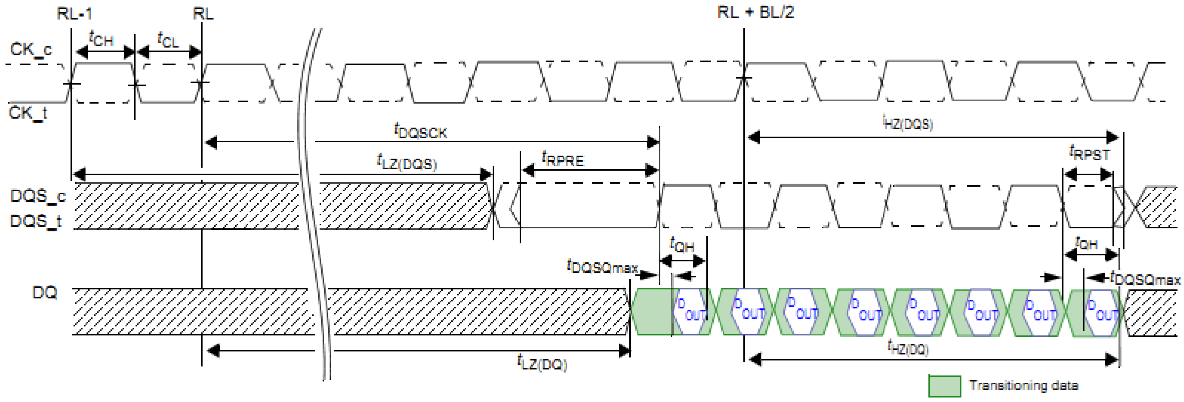


*Figure 2-6. Basic LPDDR3 read timing parameter*

*Table 2-10. 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 |
| tHZ(DQS) | DQS high-impedance time from CK/CK' | | | tDQSCK (MAX) – 0.1 | ns |

| Symbol | Description | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| 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)<br>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 |
| 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 |

| Symbol | Description | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| 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 |

### 2.2.4.2 External Memory Interface for LPDDR2

The external memory interface, shown in Figure 2-7, Figure 2-8 and Figure 2-9, is used to connect LPDDR2 device for MT6737. It includes pins ED_CLK, ED_CLK_B, ECKE, ECS#, EBA[2:0], EDQS[3:0], EDQS#[3:0], EA[9:0] and ED[31:0]. Table 2-11 summarizes the symbol definition and the related timing specifications.

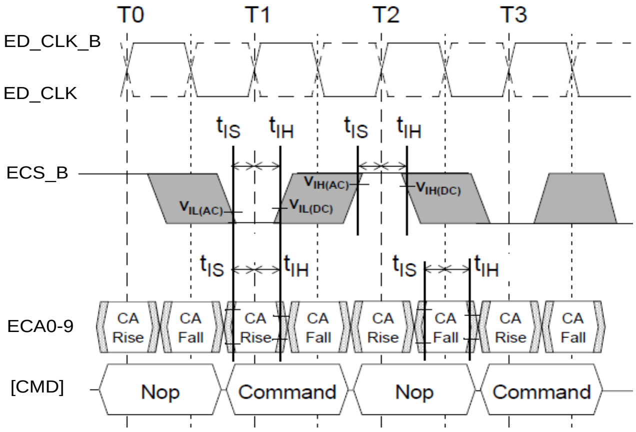


*Figure 2-7. Basic timing parameter for LPDDR2 commands*

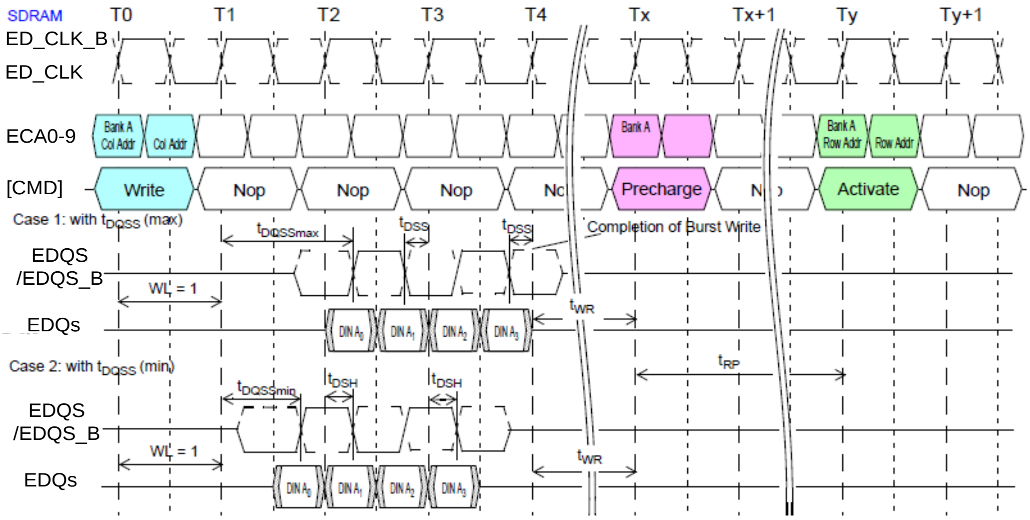
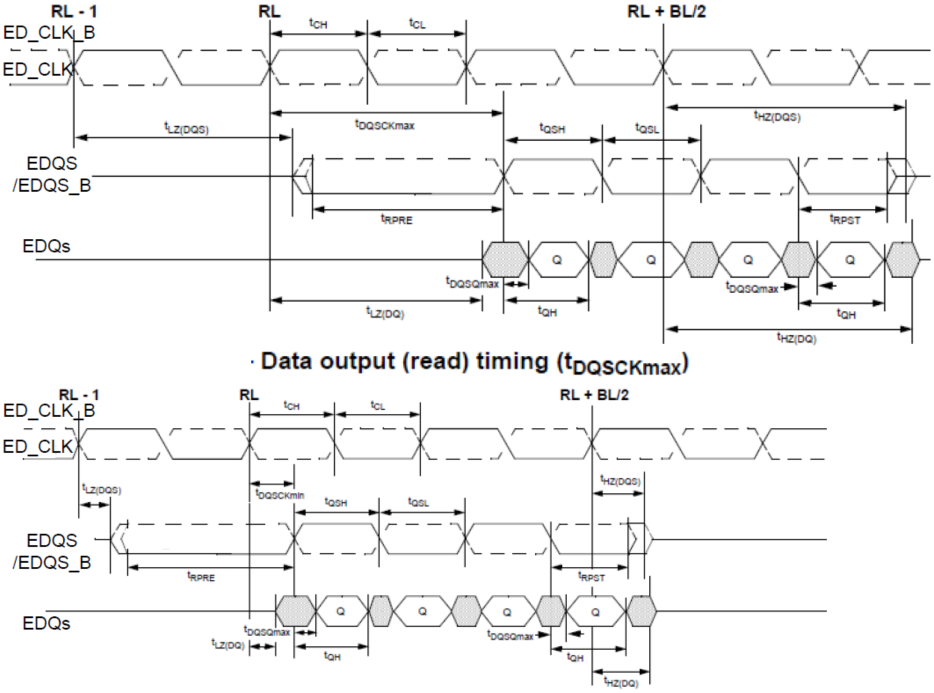


*Figure 2-8. Basic timing parameter for LPDDR2 write*



Data output (read) timing ($t_{DQSCKmax}$)

Data output (read) timing ($t_{DQSCKmin}$)

*Figure 2-9. Basic timing parameter for LPDDR2 read*

*Table 2-11. LPDDR2 AC timing parameter table of external memory interface*

| Symbol | Description | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| tCK | Clock cycle time | 3.75 | | 8 | 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 |
| tHP | Clock half period | 0.45 | | 0.55 | tCK |
| tDS | DQ & DM input setup time | 0.43 | | | ns |
| tDH | DQ & DM input hold time | 0.43 | | | ns |
| 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 |
| tIS | Address & control input setup time | 0.46 | | | ns |
| tIH | Address & control input hold time | 0.46 | | | ns |
| tLZ(DQS) | DQS low-impedance time from CK/CK’ | tDQSCK (Min.) – 300 | | | ns |
| tHZ(DQS) | DQS high-impedance time from CK/CK’ | tDQSCK (Max.) – 100 | | | ns |
| tLZ(DQ) | DQ low-impedance time from CK/CK’ | tDQSCK (Min.) – [1.4*tQHS (Max.)] | | | ns |

| Symbol | Description | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| tHZ(DQ) | DQ high-impedance time from CK/CK' | tDQSCK (Max.) + [1.4*tDQSQ (Max.)] | | | ns |
| tDQSQ | DQS-DQ skew | 0.34 | | | ns |
| tQHP | Data half period | Min. (tQSH, tQSL) | | | tCK |
| tQHS | Data hold skew factor | 0.4 | | | ns |
| tQH | DQ/DQS output hold time from DQS | tQHP – tQHS | | | 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 |
| tMRW | MODE register Write command period | 5 | | | tCK |
| tMRR | MODE register Read command period | 2 | | | tCK |
| tRPRE | Read preamble | 0.9 | | 1.1 | tCK |
| tRPST | Read postamble | tCL – 0.05 | | | tCK |
| tRAS | ACTIVE to PRECHARGE command period | 3 | | | tCK |
| tRC | ACTIVE to ACTIVE command period | 6 | | | tCK |
| tRFC | AUTO REFRESH to ACTIVE/AUTO REFRESH command period | 56 | | | tCK |
| tRCD | ACTIVE to READ or WRITE delay | 3 | | | tCK |
| tRP | PRECHARGE command period | 3 | | | tCK |
| tRRD | ACTIVE bank A to ACTIVE bank B delay | 2 | | | tCK |
| tWR | WRITE recovery time | 3 | | | tCK |
| tWTR | Internal write to READ command time | 2 | | | tCK |
| tXSR | SELF REFRESH exit to the next valid command | 40 | | | tCK |
| tXP | EXIT power-down to the next valid command delay | 2 | | | tCK |
| tCKE | CKE min. pulse width (high & low pulse width) | 2 | | | tCK |

## 2.3 System Configuration

### 2.3.1 Mode Selection

*Table 2-12. Mode selection*

| Pin name | Description |
|---|---|
| KCOL0 | 0: Force USB download mode in bootrom<br>1: NA (default) |

### 2.3.2 Constant Tie Pins

*Table 2-13. Constant tied pins*

| Pin name | Description |
|---|---|
| TESTMODE | Test mode (tied to GND) |

## 2.4 Power-on Sequence

The power-on/off sequence with XTAL is shown in the following figure:

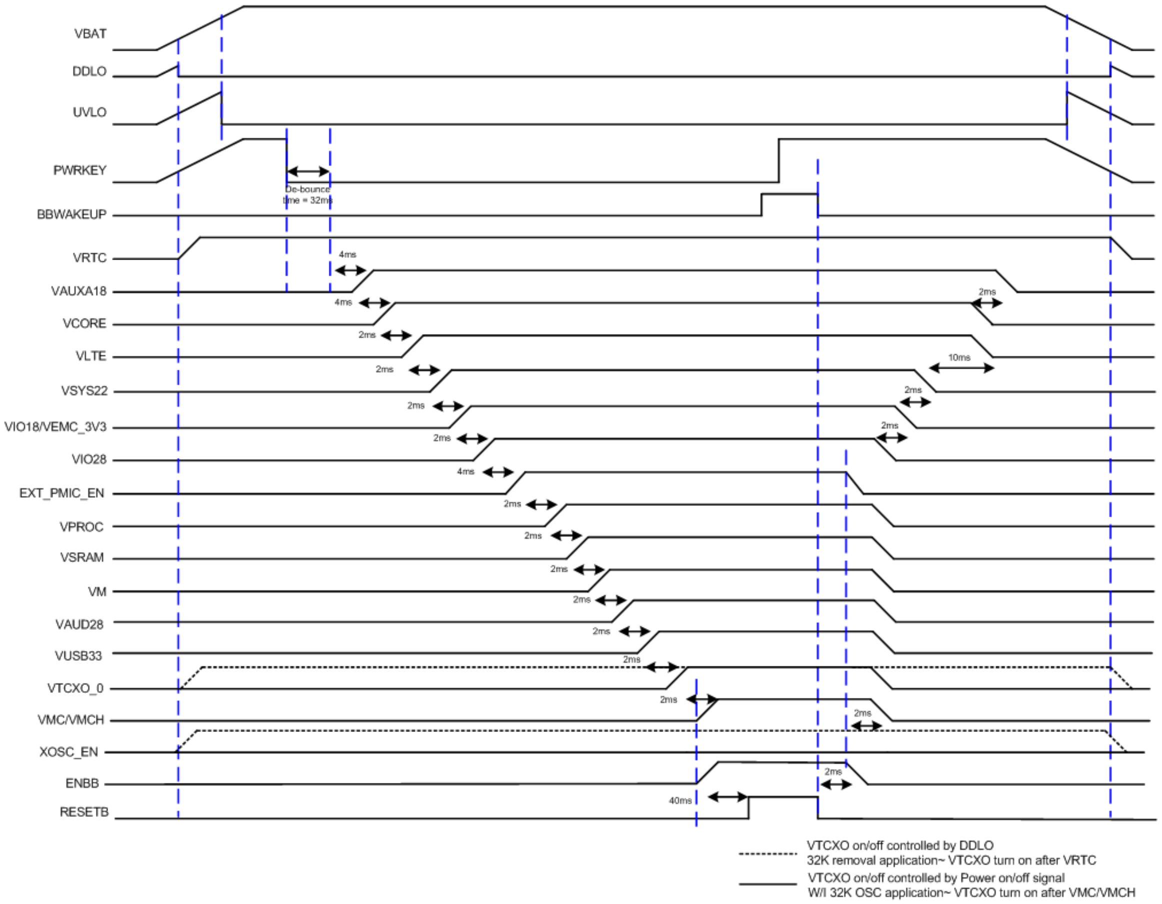


*Figure 2-10 Power on/off Sequence by pressing PWRKEY*

## 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: A DAC output to control buck-converter for envelop tracking technique.
- RF control: Two DACs for automatic power control (APC) is included.The outputs are provided to external RF power amplifiers respectively.
- Auxiliary ADC: Provides an ADC for the battery and other auxiliary analog functions monitoring.
- Clock generation: One clock-squarer for shaping the input sinwave clock and 20 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:

- LTE_BBRX
- C2K_BBRX
- LTE_BBTX
- C2K_BBTX
- ETDAC
- APC-DAC
- AUXADC
- Phase locked loop
- Temperature sensor

## 2.5.3 Block Diagram

### 2.5.3.1 LTE_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: 4 high performance sigma-delta ADCs perform I/Q digitization for further digital signal processing.

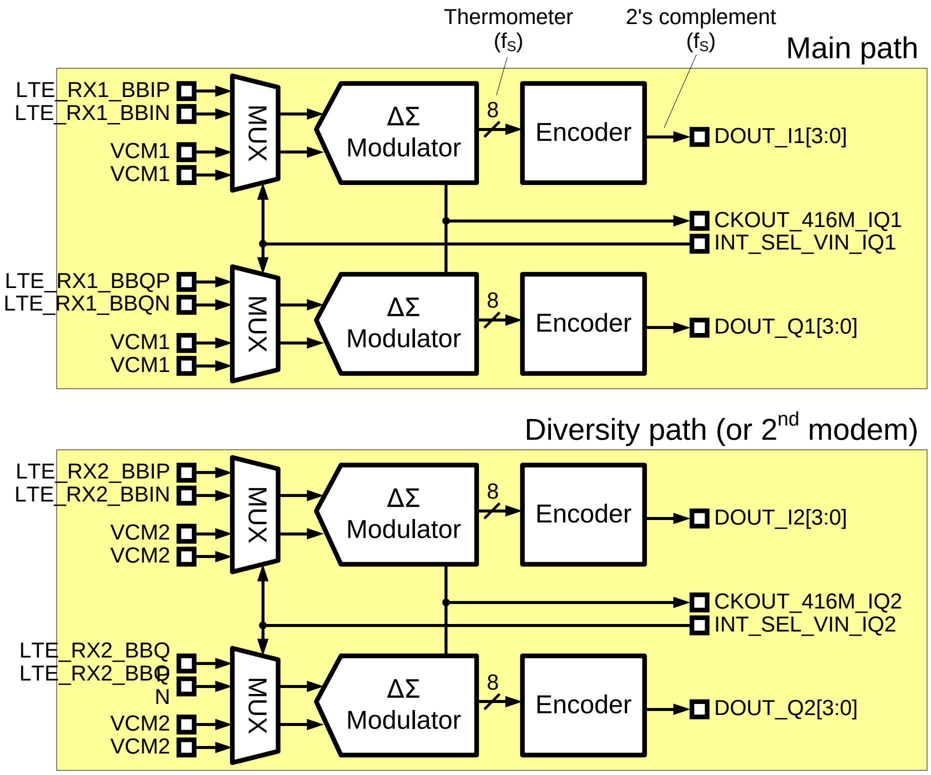


*Figure 2-11. Block diagram of LTE_BBRX-ADC*

#### 2.5.3.1.2 Functional Specifications

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

*Table 2-14. Baseband downlink specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| VIN | Differential analog input voltage (peak-to-peak) | | | 2.4 | V |

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| ICM | Common mode input current magnitude | | | 1 | uA |
| VCM | Common mode input voltage | 0.65 | 0.7 | 0.75 | V |
| FC | Input clock frequency<br>– Clock rate (LTE HB mode)<br>– Clock rate (LTE LB mode)<br>– Clock rate (DC mode)<br>– Clock rate (SC mode & GSM mode) | | 416<br>208<br>416<br>208 | | MHz |
| | Input clock duty cycle | 49.5 | 50 | 50.5 | % |
| | Input clock period jitter, DC mode | | | 0.14 | % (rms) |
| | Input clock period jitter, SC mode & GSM mode | | | 0.61 | % (rms) |
| RIN | Differential input resistance<br>– LTE HB mode<br>– LTE LB mode<br>– DC mode<br>– SC mode & GSM mode | 2.8<br>5.6<br>5.6<br>11.2 | 4<br>8<br>8<br>16 | 5.2<br>10.4<br>10.4<br>20.8 | kΩ |
| FS | Output sampling rate | | 416/208 | | MSPS |
| VOS | Differential input referred offset | | | 10 | mV |
| SIN | Signal to in-band noise<br>– LTE HB mode, 2.4Vpp (10.2MHz) sinewave, 1kHz ~ 9MHz band<br>– LTE LB mode, 2.4Vpp (5.2MHz) sinewave, 1kHz ~ 4.5MHz band<br>– DC mode, 2.4Vpp (5.2MHz) sinewave, 400kHz ~ 4.6MHz band<br>– SC mode, 2.4Vpp (2.7MHz) sinewave, 1kHz ~ 2.1MHz band<br>– GSM mode: 2.4Vpp(570kHz) sinewave, 70kHz ~ 270kHz band | 70<br>70<br>72<br>72<br>83 | 73<br>73<br>75<br>75<br>86 | | dB<br>dB<br>dB |
| DVDD18 | Digital power supply | 1.7 | 1.8 | 1.9 | V |
| AVDD18 | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | −20 | | 80 | °C |
| | Current consumption (per channel, 1 ADC)<br>– Power-up<br>– Power-down | | | 4.5<br>1 | mA<br>uA |

### 2.5.3.2 C2K_BBRX

#### 2.5.3.2.1 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: 4 high performance sigma-delta ADCs perform I/Q digitization for further digital signal processing.

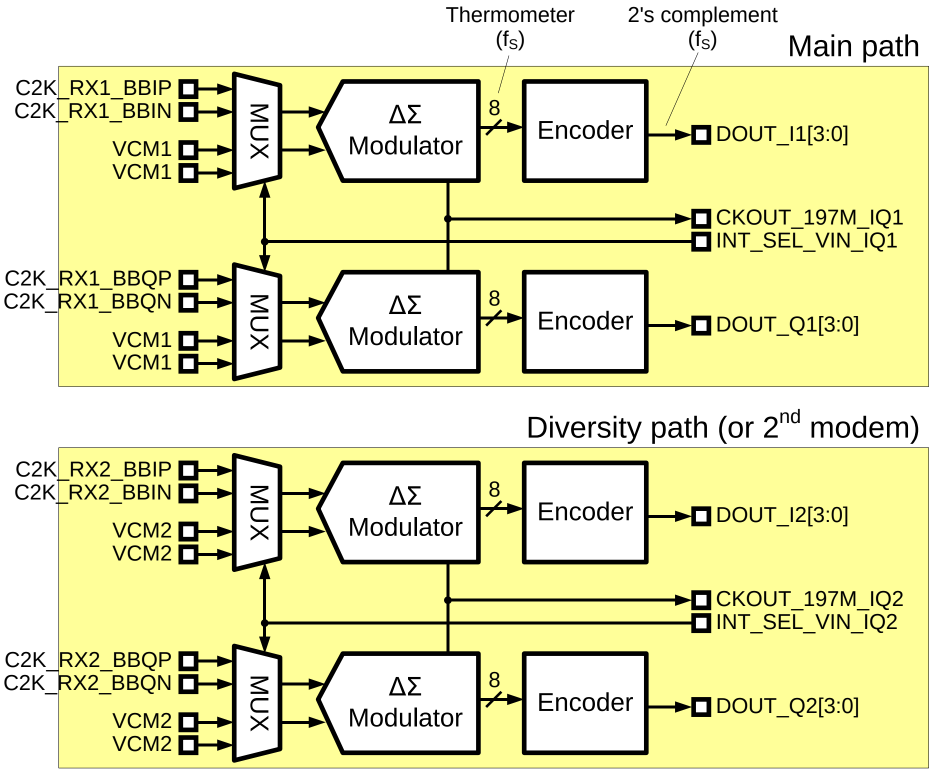


*Figure 2-12. Block diagram of C2K_BBRX-ADC*

### 2.5.3.3 Functional Specifications

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

*Table 2-15. Baseband downlink specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| VIN | Differential analog input voltage (peak-to-peak) | | | 2.4 | V |
| ICM | Common mode input current magnitude | | | 1 | uA |
| VCM | Common mode input voltage | 0.65 | 0.70 | 0.75 | V |
| FC | Clock rate | | 196.608 | | MHz |
| | Input clock duty cycle | 49.5 | 50 | 50.5 | % |
| | Input clock period jitter | | | 0.61 | % (rms) |
| RIN | Differential input resistance | 11.2 | 16 | 20.8 | kΩ |
| FS | Output sampling rate | | 196.608 | | MSPS |
| VOS | Differential input referred offset | | | 10 | mV |
| SIN | Signal to in-band noise<br>– 2.4Vpp (1.6MHz) sinewave, 1kKHz ~ 640kHz band | 79 | 82 | | dB |
| DVDD18 | Digital power supply | 1.7 | 1.8 | 1.9 | V |
| AVDD18 | Analog power supply | 1.7 | 1.8 | 1.9 | V |

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| T | Operating temperature | −20 | | 80 | °C |
| | Current consumption (per channel, 1 ADC)<br>– Power-up<br>– Power-down | | | <br>2<br>1 | <br>mA<br>uA |

### 2.5.3.4 LTE_BBTX

#### 2.5.3.4.1 Block Descriptions

BBTX includes two channel DACs with the 1st order low pass filter. The DACs are PMOS current-steering topology with NMOS constant sinking current and the active RC filter performs current to voltage buffer.

The bitwidth of DACs is 11-bit which is encoded into 7 bits of thermometer code and 8 binary code by digital hard macro inside BBTX layout. The encoded bits are timing synchronized by D-type flip-flop which is toggled by the analog local clock. The MD-PLL delivers 832MHz differential clock to BBTX. A clock divider translates the 832MHz to 416MHz for DACs and AFIFO inside mixedsys.

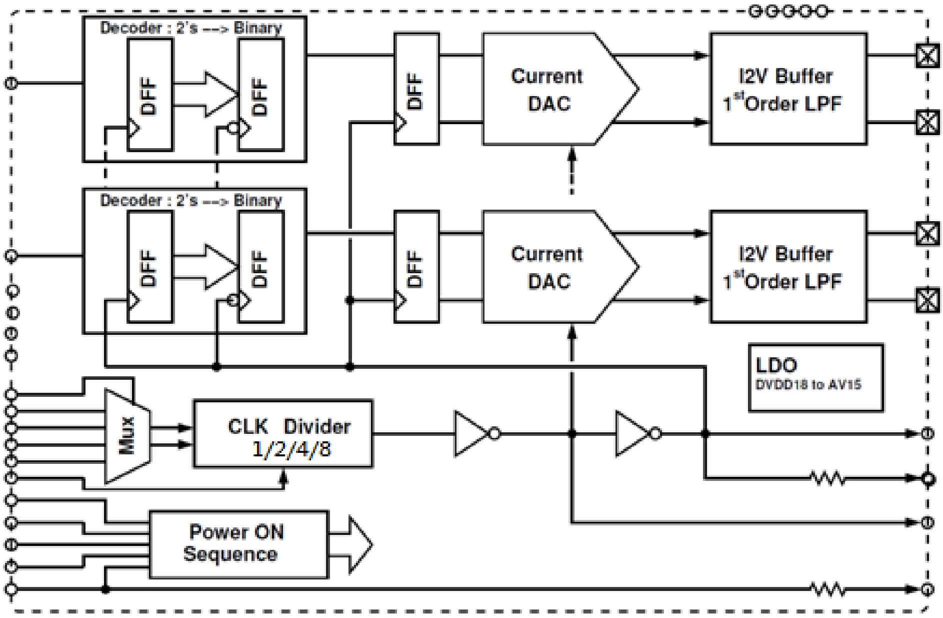


*Figure 2-13. Block diagram of LTE_BBTX*

#### 2.5.3.4.2 Functional Specifications

*Table 2-16. LTE_BBTX specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| $V_{ocm}$ | DC output common mode voltage | 0.615 | 0.65 | 0.685 | V |
| $I_K$ | HF leakage current @ supply,<br>Irms @416*2 = 832MHz | | | 3.5 | uA |
| $V_{fs}$ | DAC output swing | | 2100 | | mV |
| N | DAC resolution | | 11.0 | | bit |
| $F_s$ | Sampling clock | | 416 | | MHz |
| $I_{mis}$ | 1-sgma DAC unit cell mismatch | | | 1 | % |
| $G_{mis}$ | 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 |
| $N_{OOB}$ | Output noise level @25MHz | | 40 | | nVrms/sqrt(Hz) |
| Dinb | Inband Droop | | 0.1 | | dB |
| DNL | | | 1 | | LSB |
| INL | | | 2 | | LSB |
| IM3 | In-band two-tone test swing<br>V1=V2=290/sqrt(2)mV | | -58 | -55 | dBc |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption<br>– Power-up<br>– Power-down | | <br>6.5<br>10 | | <br>mA<br>uA |

### 2.5.3.5 C2K_BBTX

#### 2.5.3.5.1 Block Descriptions

BBTX includes two channels of DACs with the first order low pass filter. The DACs are PMOS current-steering topology with NMOS constant sinking current, and the active RC filter performs current to the voltage buffer.

The bitwidth of DACs is 10-bit which is encoded into 7 bits of thermometer code and 7 binary code by mixedsys hardware. The encoded bits are timing synchronized by D-type flip-flop which is toggled by the analog local clock. MD-PLL2 deliver 393.216MHz differential clock to BBTX. A clock divider buffered the 393.216MHz to AFIFO inside the mixedsys.

The IO power, DVDD18_MD, is regulated to a voltage around 1.55V to supply analog component, and the required bias currents are generated by BBRX.

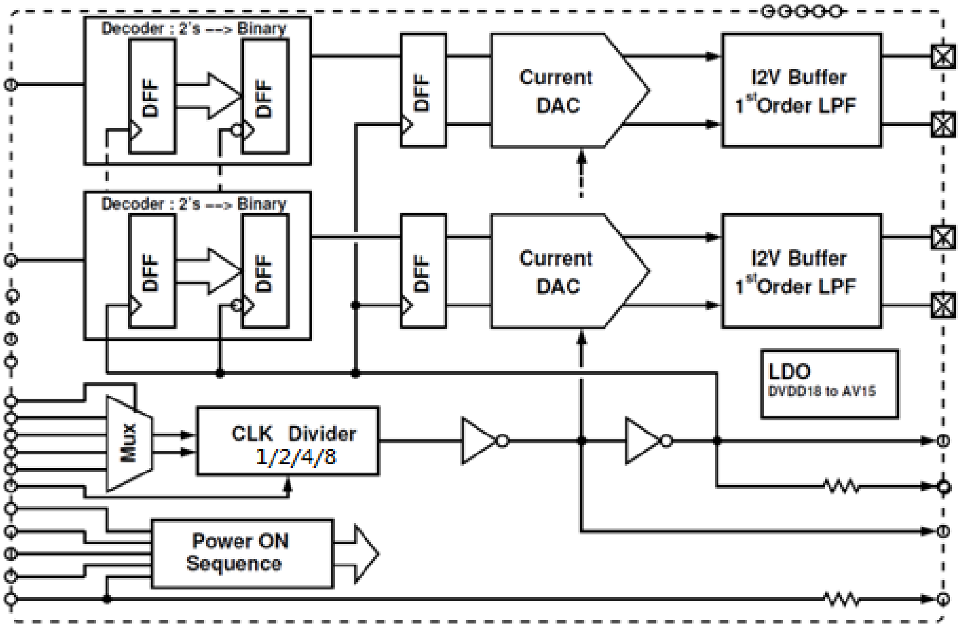


*Figure 2-14. Block diagram of C2K_BBTX*

### 2.5.3.5.2 Functional Specifications

*Table 2-17. C2K_BBTX specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit. |
|---|---|---|---|---|---|
| $V_{ocm}$ | DC output common mode voltage | 0.615 | 0.65 | 0.685 | V |
| $I_K$ | HF Leakage current @ supply,<br>Irms @416*2=832MHz | | | 3.5 | uA |
| $V_{fs}$ | DAC output swing | | 2100 | | mV |
| N | DAC resolution | | 10.0 | | bit |
| $F_s$ | Sampling clock | | 393.216 | | MHz |
| $I_{mis}$ | 1-sgma DAC unit cell mismatch | | | 1 | % |
| $G_{mis}$ | 3-sigma I/Q gain mismatch | -0.2 | | 0.2 | dB |
| $V_{os\_T}$ | 3-sigma output differential DC offset over temp. | | | 4 | mV |
| $V_{os}$ | 3-sigma output differential DC offset | | | 10 | mV |
| $F_{3dB}$ | 3dB corner freq. | 20 | 25 | 30 | MHz |
| $S_{LPF}$ | LPF selectivity @832MHz | 28 | | | dB |
| $N_{OOB}$ | Output noise level @45MHz | | 15.1 | 30.1. | nVrms/sqrt(Hz) |
| CN | Signal to noise ratio@45MHz | | -146 | -140 | dBc/Hz |
| IM3 | In-band two-tone test swing<br>V1=V2=290/sqrt(2) mV | | -60 | -56 | dBc |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption<br>– Power-up | | 6.5 | | mA |

| Symbol | Parameter | Min. | Typ. | Max. | Unit. |
|---|---|---|---|---|---|
| | – Power-down | | 10 | | uA |

### 2.5.3.6 ETDAC

#### 2.5.3.6.1 Block Descriptions

The ETDAC (Envelope Tracking DAC) provides analog envelope signal to external ET modulator. It includes:

1. 11-bit D/A converter: Converts digital modulated signals to analog domain. The input to the DAC is sampled at 416MHz rate with the 11-bit resolution.
2. Smoothing filter: The low-pass filter performs smoothing function for DAC output signals with a 20/40MHz 1st-order Butterworth frequency response.

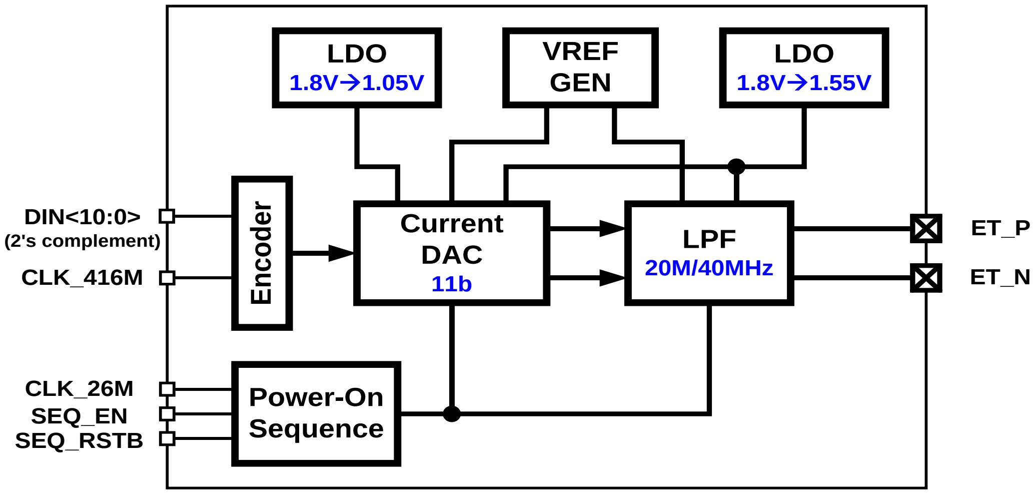


*Figure 2-15. Block diagram of ETDAC*

#### 2.5.3.6.2 Functional Specfications

*Table 2-18. ETDAC specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| N | Resolution | | 11 | | Bit |
| FS | Sampling rate | | 416 | | MSPS |
| IM3 | 3rd order Intermodulation distortion | | -60 | -50 | dB |
| | Output swing (full swing) | | 2 | | Vppd |
| VOCM | Output CM voltage | 0.6 | | 0.85 | V |
| | Output capacitance (single-ended) | | | 10 | PF |
| | Output resistance (differential) | | 100 | | KΩ |

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| DNL | Differential nonlinearity | -1 | | +1 | LSB |
| INL | Integral nonlinearity | -2 | | +2 | LSB |
| FCUT | Filter -3dB cutoff frequency (calibrated) | | 20/40 | | MHz |
| DVDD | Digital power supply | 0.95 | 1.05 | 1.15 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption<br>– Power-up<br>– Power-down | | 4.5<br>10 | | mA<br>uA |

### 2.5.3.7 APC-DAC

#### 2.5.3.7.1 Block Descriptions

See the figure below. APC-DAC is designed to produce a single-ended output signal at APC pin. Two APC-DACs provide two separate output signals (APC1 and APC2).

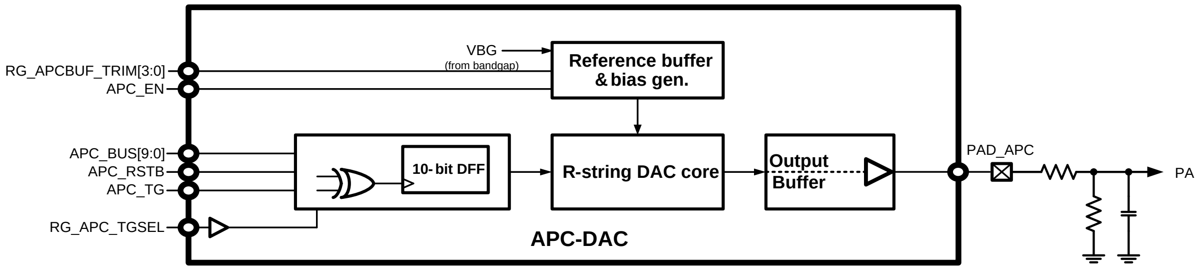


*Figure 2-16. Block diagram of APC-DAC (same architecture for two APC-DACs)*

#### 2.5.3.7.2 Functional Specifications

See the table below for the functional specifications of the APC-DAC (apply to both APC-DACs).

*Table 2-19. APC-DAC specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| N | Resolution | | 10 | | Bit |
| $F_S$ | Clock rate | 1.0833 | | 2.1666 | MS/s |
| SNDR | Signal-to-noise-and-distortion ratio (10kHz sine wave with 1.0V swing) | | 50 | | dB |
| $T_S$ | 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 |

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| DNL | Differential nonlinearity (code 30 ~ 970) | | ±1.0 | | LSB |
| INL | Integral nonlinearity (code 30 ~ 970) | | ±2.0 | | LSB |
| DVDD | Digital power supply | 0.81 | 1.0 | 1.1 | V |
| AVDD | Analog power supply | 2.6 | 2.8 | 3.0 | V |
| T | Operating temperature | −20 | | 85 | °C |
| $I_{ON}$ | Current consumption (power-on state) | | 450 | | uA |
| $I_{OFF}$ | Current consumption (power-down state) | | | 20 | uA |

### 2.5.3.8 AUXADC

#### 2.5.3.8.1 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.

See Table 2-20 for brief descriptions of AUXADC input channels.

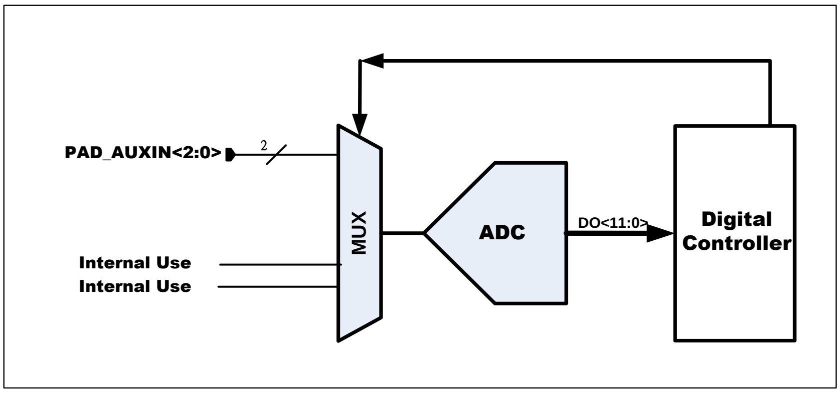


*Figure 2-17. Block diagram of AUXADC*

*Table 2-20. Definitions of AUXADC channels*

| AUXADC channel ID | Description |
|---|---|
| Channel 0 | External use (AUX_IN0) |
| Channel 1 | External use (AUX_IN1) |
| Channel 2 | NA |

| AUXADC channel ID | Description |
|---|---|
| Channel 3 | NA |
| Channel 4 | NA |
| Channel 5 | NA |
| Channel 6 | NA |
| Channel 7 | NA |
| Channel 8 | NA |
| Channel 9 | NA |
| Channel 10 | Internal use |
| Channel 11 | Internal use |
| Channel 12 | External use (AUX_IN2) |
| Channel 13 | NA |
| Channel 14 | NA |
| Channel 15 | NA |

### 2.5.3.8.2 Functional Specifications

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

*Table 2-21. 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<br>Unselected channel<br>Selected channel | | <br>50<br>4 | | <br>fF<br>pF |
| RIN | Input resistance<br>Unselected channel | <br>20 | | | <br>MΩ |
| | Clock latency | | N+8 | | 1/FC |
| DNL | Differential nonlinearity | | +1.0/-1.0 | | LSB |
| INL | Integral nonlinearity | | +2.0/-2.0 | | LSB |
| SINAD | Signal to noise and distortion ratio (1kHz full swing input & 1.0833MHz clock rate) | 56 | 64 | | dB |
| DVDD | Digital power supply | 0.81 | 1.0 | 1.1 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption<br>Power-up<br>Power-down | | <br>600<br>1 | | <br>uA<br>uA |
| | Accuracy- before trim | | | +-75 | mV |
| | Accuracy- after trim | | | +-10 | mV |

### 2.5.3.9 Clock Squarer

#### 2.5.3.9.1 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.

#### 2.5.3.9.2 Functional Specifications

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

***Table 2-22. 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 |
| DVDD | Digital power supply | 0.81 | 1.0 | 1.1 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 500 | | uA |

### 2.5.3.10 Phase Locked Loop

#### 2.5.3.10.1 Block Descriptions

There are total 17 PLLs in PLL macro separated into 2 groups, providing several clocks for CPU, BUS, modem, analog modem, MSDC and image-sensor.

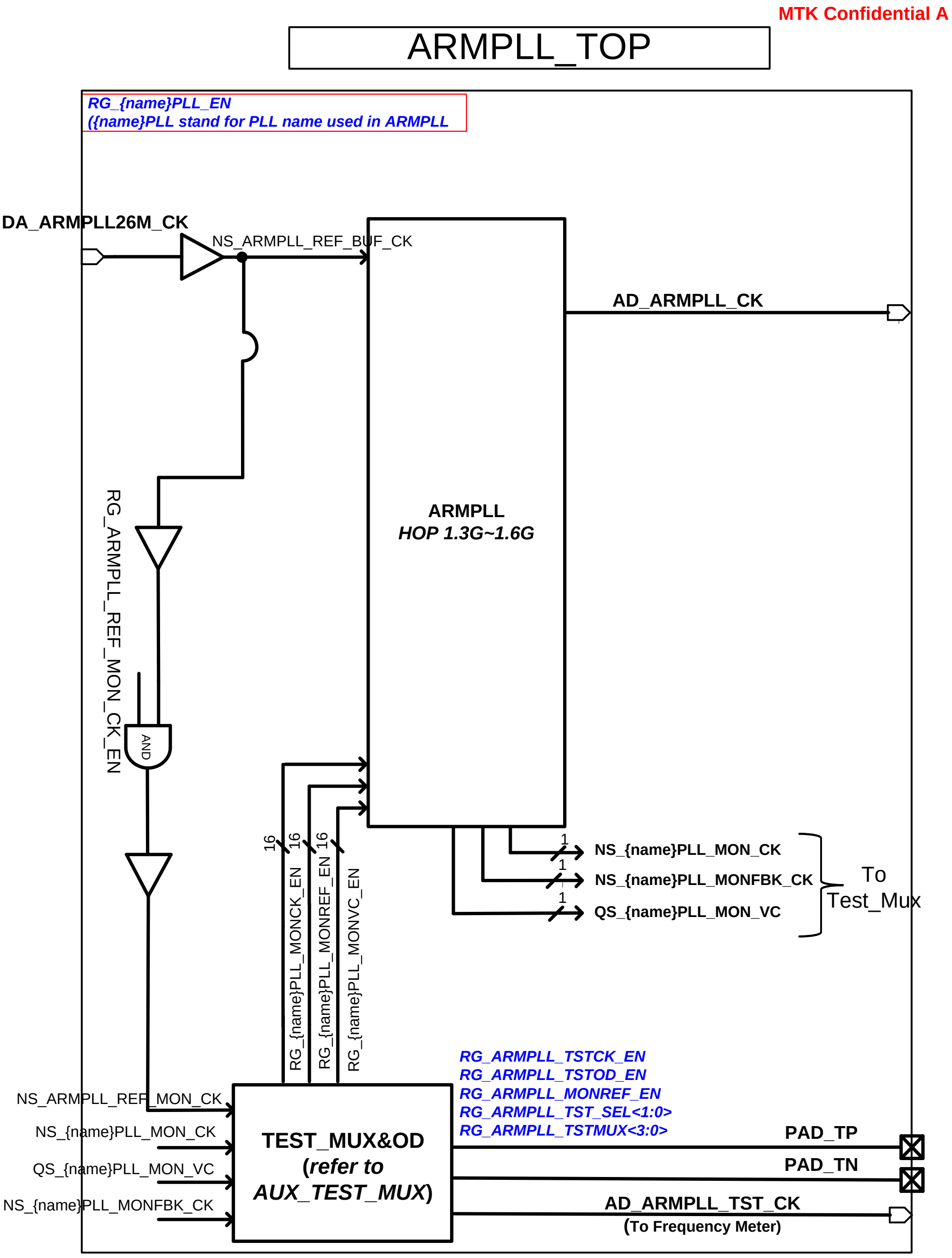


*Figure 2-18. Block diagram of ARMPLL*

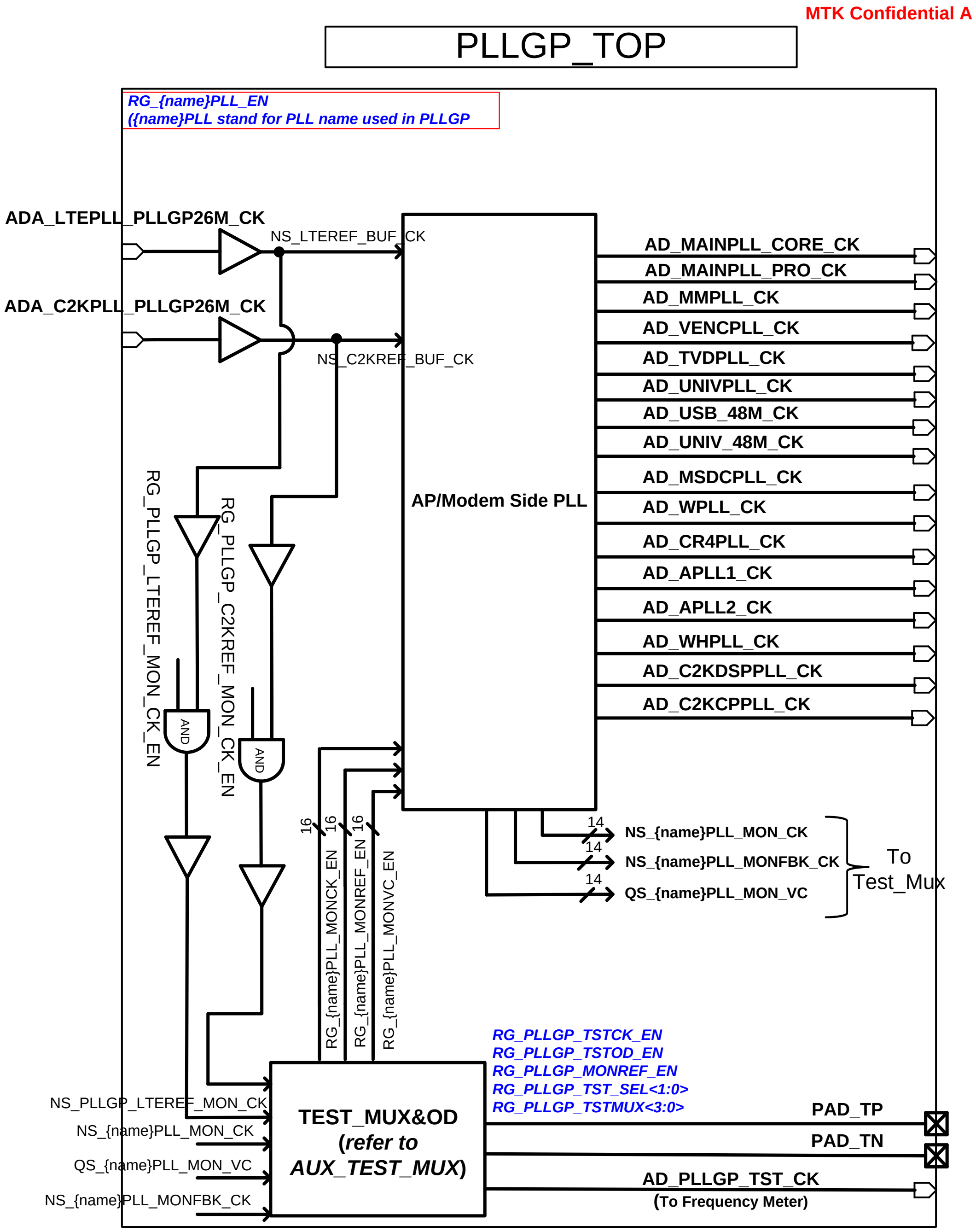


*Figure 2-19. Block diagram of PLLGP*

### 2.5.3.10.2 Functional Specifications

See the table below for the functional specifications of PLL.

*Table 2-23. ARMPLL specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | | 1300 | | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 45 | 50 | 55 | % |
| | Output clock jitter (period jitter) | | 30 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 1 | | mA |
| | Power-down current consumption | | | 1 | uA |

*Table 2-24. MAINPLL specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | | 1092 | | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 45 | 50 | 55 | % |
| | Output clock jitter (period jitter) | | 30 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 1 | | mA |
| | Power-down current consumption | | | 1 | uA |

*Table 2-25. MMPLL specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | | 450 | | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 47 | 50 | 53 | % |
| | Output clock jitter (period jitter) | | 60 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

### Table 2-26. UNIVPLL specifications

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | N/A | 1248 | N/A | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 45 | 50 | 55 | % |
| | Output clock jitter (period jitter) | | 30 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

### Table 2-27. MSDCPLL specifications

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | | 800 | | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 45 | 50 | 55 | % |
| | Output clock jitter (period jitter) | | 60 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

### Table 2-28. WPLL specifications

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | N/A | 491.52 | N/A | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 47 | 50 | 53 | % |
| | Output clock jitter (rms period jitter) | | 60 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

*Table 2-29. WHPLL specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | N/A | 500.5 | N/A | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 47 | 50 | 53 | % |
| | Output clock jitter (rms period jitter) | | 60 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

*Table 2-30. C2KCPPLL specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | N/A | 780 | N/A | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 47 | 50 | 53 | % |
| | Output clock jitter (period jitter) | | 60 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

*Table 2-31. C2KDSPPLL specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | | 340 | | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 47 | 50 | 53 | % |
| | Output clock jitter (period jitter) | | 60 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

*Table 2-32. CR4PLL specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | | 1196 | | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 47 | 50 | 53 | % |
| | Output clock jitter (period jitter) | | 30 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

*Table 2-33. VENCPLL specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | | 295.75 | | MHz |
| | Settling time | | 20 | | us |
| | Output clock duty cycle | 47 | 50 | 53 | % |
| | Output clock jitter (period jitter) | | 60 | | ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

*Table 2-34. TVDPLL specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | | 148.5 | | MHz |
| | Settling time | | 20 | | Us |
| | Output clock duty cycle | 47 | 50 | 53 | % |
| | Output clock jitter (period jitter) | | 60 | | Ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

*Table 2-35. LTEDSPPLL specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | | 320 | | MHz |
| | Settling time | | 20 | | Us |
| | Output clock duty cycle | 47 | 50 | 53 | % |
| | Output clock jitter (period jitter) | | 60 | | Ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

*Table 2-36. APLL1 specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Fin | Input clock frequency | | 26 | | MHz |
| Fout | Output clock frequency | | 98.304 | | MHz |
| | Settling time | | 20 | | Us |
| | Output clock duty cycle | 47 | 50 | 53 | % |
| | Output clock jitter (period jitter) | | 60 | | Ps |
| DVDD | Digital power supply | 0.945 | 1.05 | 1.155 | V |
| AVDD | Analog power supply | 1.7 | 1.8 | 1.9 | V |
| T | Operating temperature | -20 | | 80 | °C |
| | Current consumption | | 0.8 | | mA |
| | Power-down current consumption | | | 1 | uA |

### 2.5.3.11 Temperature Sensor

#### 2.5.3.11.1 Block Descriptions

In order to monitor the temperature of CPUs, several temperature sensors are provided. The temperature sensor is made of substrate BJT in the CMOS process. The voltage output of temperature sensor is measured by AUXADC.

#### 2.5.3.11.2 Functional Specifications

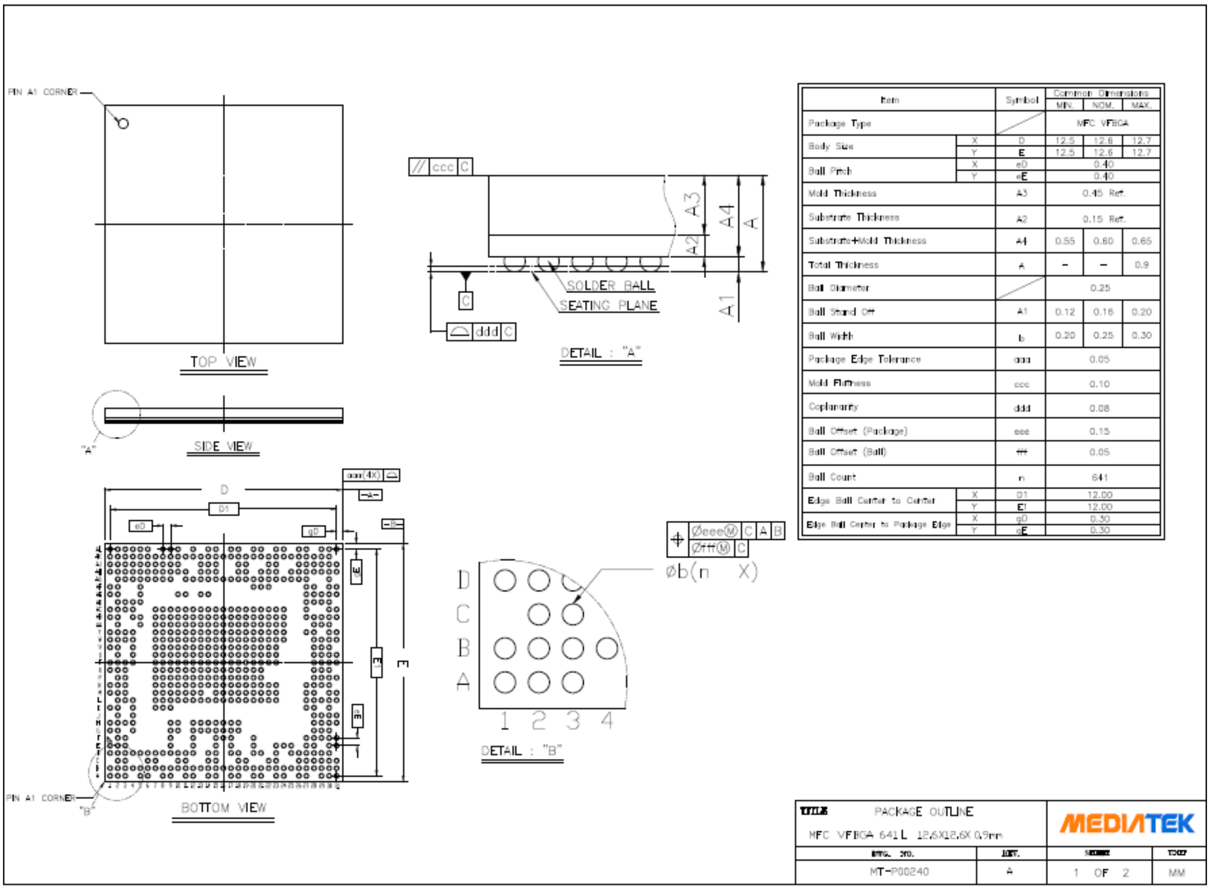
See the table below for the functional specifications of temperature sensor.

*Table 2-37. Temperature sensor specifications*

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| | Resolution | | 0.15 | | °C |
| | Temperature range | 0 | | 85 | °C |
| | Accuracy | -5 | | 5 | °C |
| | Active current | | 60 | | uA |
| | Quiescent current | | 12 | | uA |

## 2.6 Package Information

### 2.6.1 Package Outlines



| Item | | Symbol | Common Dimensions | | |
|---|---|---|---|---|---|
| | | | MIN. | NOM. | MAX. |
| Package Type | | | MFC VFBGA | | |
| Body Size | X | D | 12.5 | 12.6 | 12.7 |
| | Y | E | 12.5 | 12.6 | 12.7 |
| Ball Pitch | X | eD | 0.40 | | |
| | Y | eE | 0.40 | | |
| Mold Thickness | | A3 | 0.45 Ref. | | |
| Substrate Thickness | | A2 | 0.15 Ref. | | |
| Substrate+Mold Thickness | | A4 | 0.55 | 0.60 | 0.65 |
| Total Thickness | | A | – | – | 0.9 |
| Ball Diameter | | | 0.25 | | |
| Ball Stand Off | | A1 | 0.12 | 0.16 | 0.20 |
| Ball Width | | b | 0.20 | 0.25 | 0.30 |
| Package Edge Tolerance | | aaa | 0.05 | | |
| Mold Flatness | | ccc | 0.10 | | |
| Coplanarity | | ddd | 0.08 | | |
| Ball Offset (Package) | | eee | 0.15 | | |
| Ball Offset (Ball) | | fff | 0.05 | | |
| Ball Count | | n | 641 | | |
| Edge Ball Center to Center | X | D1 | 12.00 | | |
| | Y | E1 | 12.00 | | |
| Edge Ball Center to Package Edge | X | gD | 0.30 | | |
| | Y | gE | 0.30 | | |

*Figure 2-20. Outlines and dimensions of VFBGA 12.6mm*12.6mm, 641 balls, 0.4mm pitch package*

### 2.6.2 Thermal Operating Specifications

*Table 2-38. Thermal operating specifications*

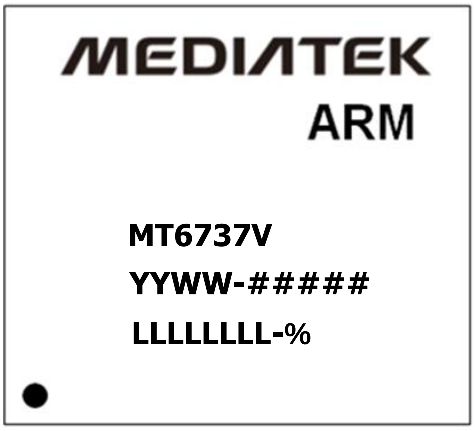
| Symbol | Description | Value | Unit | Note |
|---|---|---|---|---|
| | Max. operating junction temperature | 125 | °C | |
| | Package thermal resistances in nature convection | 37.65 | °C/Watt | |

### 2.6.3 Lead-free Packaging

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

## 2.7 Ordering Information

### 2.7.1 Top Marking Definition



- **YYWW: Date code**
- **%: Functional code**
- **#####: Subcontractor code**
- **LLLLLLLL: Die lot No.**

***Figure 2-21. Top marking of MT6737***

# 3 MCU and BUS Fabric

## 3.1 MCU System

### 3.1.1 Introduction

MCUSYS is a subsystem responsible for running operating system and application programs in MT6737. It comprises four Cortex-A53 cores into 1 cluster  and Generic Interrupt Controller (GIC). A 333MHz 128-bit AXI bus is directly connected to External Memory Interface (EMI) to minimize the access latency to DRAM and provide sufficient memory bandwidth. The peripheral system and on-chip storage are bridged through a 273MHz/64-bit AXI bus, and the outstanding capability of AXI protocol allows the system to exploit its maximum throughput from eight CPU cores.

MCUSYS supports DVFS technology which allows CPU to run at different frequency/voltage configurations for different application requirements. When in standby mode, MCUSYS can be completely shut down to further save power consumption and optimize the battery usage on mobile devices.

### 3.1.2 Features

#### 3.1.2.1 Cluster 0, Cortex-A53 Specifications

- Four-core ARM® Cortex-A53 MPCore™ operating at 1.25GHz
- Supports ARMv8-A architecture for both 32 and 64-bit execution state
- Supports NEON multimedia processing engine with SIMDv2/VFPv4 ISA
- Optional support ARMv8 Cryptographic extension
- 32KB L1 I-cache and 32KB L1 D-cache
- 256KB unified L2 cache for CPU cluster
- DVFS technology with adaptive operating voltage from 0.95V to 1.25V
- Supports ARM Jazelle technology

#### 3.1.2.2 Clock Modes between Clusters and AXI Bus Fabric

The CPU and AXI bus fabric is synchronous and with integer clock ratio for example 1:1/1:2/1:4 for the best system performance. CPU and AXI bus fabric also support Dynamic Clock Management (DCM) mechanism to dynamically turn off the clock when no transactions are on the bus interface. CPU, cluster and AXI bus fabric can also support DVFS technology to lower power consumption.

### 3.1.3 Interrupt Controller

MT6737 uses ARM GIC400 interrupt controller for interrupt management. GIC400 is embedded inside MCUSYS alongside CCI to minimize the interrupt handling latency. For the interrupt connected to GIC400, see the table below for details. The GIC interrupts are separated into 2 categories, the Private Peripheral Interrupts (PPI) and Shared Peripheral Interrupts (SPI). PPI occupies the first 32 interrupt slots in GIC and are banked for each CPU core. SPI begins from the 33rd interrupt and is shared by all CPU cores.

*Table 3-1. Interrupt request list for Cortex-A53*

| GIC ID | Interrupt source/name | Polarity | Trigger type |
|---|---|---|---|
| 0 | Software generated interrupt 0 | H | Edge |
| 1 | Software generated interrupt 1 | H | Edge |
| 2 | Software generated interrupt 2 | H | Edge |
| 3 | Software generated interrupt 3 | H | Edge |
| 4 | Software generated interrupt 4 | H | Edge |
| 5 | Software generated interrupt 5 | H | Edge |
| 6 | Software generated interrupt 6 | H | Edge |
| 7 | Software generated interrupt 7 | H | Edge |
| 8 | Software generated interrupt 8 | H | Edge |
| 9 | Software generated interrupt 9 | H | Edge |
| 10 | Software generated interrupt 10 | H | Edge |
| 11 | Software generated interrupt 11 | H | Edge |
| 12 | Software generated interrupt 12 | H | Edge |
| 13 | Software generated interrupt 13 | H | Edge |
| 14 | Software generated interrupt 14 | H | Edge |
| 15 | Software generated interrupt 15 | H | Edge |
| 16 | (Reserved) | - | - |
| 17 | (Reserved) | - | - |
| 18 | (Reserved) | - | - |
| 19 | (Reserved) | - | - |
| 20 | (Reserved) | - | - |
| 21 | (Reserved) | - | - |
| 22 | (Reserved) | - | - |
| 23 | (Reserved) | - | - |
| 24 | (Reserved) | - | - |
| 25 | Virtual maintenance interrupt | L | Level |
| 26 | Hypervisor timer event | L | Level |
| 27 | Virtual timer event | L | Level |
| 28 | Legacy nFIQ | L | Level |
| 29 | Secure physical timer event | L | Level |
| 30 | Non-secure physical timer event | L | Level |
| 31 | Legacy nIRQ | L | Level |
| 32 | nIRQOUT[0] | L | Level |

| GIC ID | Interrupt source/name | Polarity | Trigger type |
|---|---|---|---|
| 33 | nIRQOUT[1] | L | Level |
| 34 | nIRQOUT[2] | L | Level |
| 35 | nIRQOUT[3] | L | Level |
| 40 | nPMUIRQ[0] | L | Level |
| 41 | nPMUIRQ[1] | L | Level |
| 42 | nPMUIRQ[2] | L | Level |
| 43 | nPMUIRQ[3] | L | Level |
| 48 | nCNTHPIRQ[0] | L | Level |
| 49 | nCNTHPIRQ[1] | L | Level |
| 50 | nCNTHPIRQ[2] | L | Level |
| 51 | nCNTHPIRQ[3] | L | Level |
| 56 | nCNTVIRQ[0] | L | Level |
| 57 | nCNTVIRQ[1] | L | Level |
| 58 | nCNTVIRQ[2] | L | Level |
| 59 | nCNTVIRQ[3] | L | Level |
| 64 | CNTPSIRQ[0] | H | Level |
| 65 | CNTPSIRQ[1] | H | Level |
| 66 | CNTPSIRQ[2] | H | Level |
| 67 | CNTPSIRQ[3] | H | Level |
| 72 | CNTPNSIRQ[0] | H | Level |
| 73 | CNTPNSIRQ[1] | H | Level |
| 74 | CNTPNSIRQ[2] | H | Level |
| 75 | CNTPNSIRQ[3] | H | Level |
| 80 | EXTERRIRQ | H | Level |
| 82 | mp0_CTIIRQ_sync[0] | H | Level |
| 83 | mp0_CTIIRQ_sync[1] | H | Level |
| 84 | mp0_CTIIRQ_sync[2] | H | Level |
| 85 | mp0_CTIIRQ_sync[3] | H | Level |
| 90 | CCI_EVNTCNTOVFL[0] | H | Level |
| 91 | CCI_EVNTCNTOVFL[1] | H | Level |
| 92 | CCI_EVNTCNTOVFL[2] | H | Level |
| 93 | CCI_EVNTCNTOVFL[3] | H | Level |
| 94 | CCI_EVNTCNTOVFL[4] | H | Level |
| 95 | CCI_ERRORIRQ | H | Level |
| 96 | mcu_xgpt_irq[0] | H | Level |
| 97 | mcu_xgpt_irq[1] | H | Level |
| 98 | mcu_xgpt_irq[2] | H | Level |
| 99 | mcu_xgpt_irq[3] | H | Level |
| 100 | mcu_xgpt_irq[4] | H | Level |
| 101 | mcu_xgpt_irq[5] | H | Level |
| 102 | mcu_xgpt_irq[6] | H | Level |
| 103 | mcu_xgpt_irq[7] | H | Level |
| 104 | usb_mcu_irq_b[0] | L | Level |

| GIC ID | Interrupt source/name | Polarity | Trigger type |
|---|---|---|---|
| 105 | usb_mcu_irq_b[1] | L | Level |
| 106 | ts_irq_b | L | Edge |
| 107 | ts_batch_irq_b | L | Edge |
| 108 | lowbattery_irq_b | L | Edge |
| 109 | pwm_irq_b | L | Level |
| 110 | therm_ctrl_irq_b | L | Level |
| 111 | msdc0_irq_b | L | Level |
| 112 | msdc1_irq_b | L | Level |
| 113 | (Reserved) | - | - |
| 114 | (Reserved) | - | - |
| 115 | ap_hif_irq_b | L | Level |
| 116 | i2c0_irqb | L | Level |
| 117 | i2c1_irqb | L | Level |
| 118 | i2c2_irqb | L | Level |
| 119 | i2c3_irqb | L | Level |
| 120 | (Reserved) | - | - |
| 121 | (Reserved) | - | - |
| 122 | btif_irq_b | L | Edge |
| 123 | uart0_irq_b | L | Level |
| 124 | uart1_irq_b | L | Level |
| 125 | uart2_irq_b | L | Level |
| 126 | uart3_irq_b | L | Level |
| 127 | nfiecc_irq_b | L | Level |
| 128 | nfi_irq_b | L | Level |
| 129 | dma_irq[0] (HIF0) | L | Level |
| 130 | dma_irq[1] (IRDA) | L | Level |
| 131 | dma_irq[2] (I2C0) | L | Level |
| 132 | dma_irq[3] (I2C1) | L | Level |
| 133 | dma_irq[4] (I2C2) | L | Level |
| 134 | dma_irq[5] (I2C3) | L | Level |
| 135 | dma_irq[6] (uart0_tx) | L | Level |
| 136 | dma_irq[7] (uart0_rx) | L | Level |
| 137 | dma_irq[8] (uart1_tx) | L | Level |
| 138 | dma_irq[9] (uart1_rx) | L | Level |
| 139 | dma_irq[10] (uart2_tx) | L | Level |
| 140 | dma_irq[11] (uart2_rx) | L | Level |
| 141 | dma_irq[12] (uart3_tx) | L | Level |
| 142 | dma_irq[13] (uart3_rx) | L | Level |
| 143 | dma_irq[14] (uart4_tx) | L | Level |
| 144 | dma_irq[15] (uart4_rx) | L | Level |
| 145 | (Reserved) | - | - |
| 146 | (Reserved) | - | - |
| 147 | (Reserved) | - | - |

| GIC ID | Interrupt source/name | Polarity | Trigger type |
|---|---|---|---|
| 148 | (Reserved) | - | - |
| 149 | (Reserved) | - | - |
| 150 | spi0_irq_b | L | Level |
| 151 | msdc0_wakeup_ps_irq | H | Edge |
| 152 | msdc1_wakeup_ps_irq | H | Edge |
| 153 | (Reserved) | - | - |
| 154 | (Reserved) | - | - |
| 155 | irda_irq | H | Level |
| 156 | irtx_irq | H | Level |
| 157 | ptp_fsm_irq_b | L | Level |
| 158 | conn2ap_btif_wakeup_out_b | L | Edge |
| 159 | (Reserved) | - | - |
| 160 | wdt_irq_b | L | Edge |
| 161 | (Reserved) | - | - |
| 162 | (Reserved) | - | - |
| 163 | (Reserved) | - | - |
| 164 | dcc_aparm_irq | L | Level |
| 165 | (Reserved) | - | - |
| 166 | aparm_domain_irq_b | L | Level |
| 167 | aparm_decerr_irq_b | L | Level |
| 168 | domain_abort_irq | H | Level |
| 169 | bus_dbg_tracker_irq_b | L | Level |
| 170 | cq_dma_gdma_irq_b | L | Level |
| 171 | (Reserved) | - | - |
| 172 | ccifo_ap_irq_b | L | Level |
| 173 | trng_irq_b | L | Level |
| 174 | cq_dma_gdma2_irq_b | L | Level |
| 175 | cq_dam_audio_irq_b | L | Level |
| 176 | afe_mcu_irq_b | L | Level |
| 177 | cldma_ap_irq | H | Level |
| 178 | mmu_irq_b | L | Level |
| 179 | mmu_sec_irq_b | L | Level |
| 180 | gce_secure_irq_b | L | Level |
| 181 | refresh_rate_int_pulse_b | L | Edge |
| 182 | gcpu_irq_b | L | Level |
| 183 | gce_irq_b | L | Level |
| 184 | apxgpt_irq_b | L | Level |
| 185 | eint_irq[0](arm_eint_irq) | H | Level |
| 186 | eint_event_irq_b | L | Level |
| 187 | eint_direct_irq[0] | H | Level |
| 188 | eint_direct_irq[1] | H | Level |
| 189 | eint_direct_irq[2] | H | Level |
| 190 | eint_direct_irq[3] | H | Level |

| GIC ID | Interrupt source/name | Polarity | Trigger type |
|---|---|---|---|
| 191 | dnl3_xpgt64_irq_b[0] | L | Level |
| 192 | dnl3_xpgt64_irq_b[1] | L | Level |
| 193 | dnl3_xpgt64_irq_b[2] | L | Level |
| 194 | dnl3_xpgt64_irq_b[3] | L | Level |
| 195 | pmic_wrap_err | H | Level |
| 196 | kp_irq_b | L | Edge |
| 197 | spm_irq_b[0] | L | Level |
| 198 | spm_irq_b[1] | L | Level |
| 199 | spm_irq_b[2] | L | Level |
| 200 | spm_irq_b[3] | L | Level |
| 201 | spm_irq_b[4] | L | Level |
| 202 | spm_irq_b[5] | L | Level |
| 203 | spm_irq_b[6] | L | Level |
| 204 | spm_irq_b[7] | L | Level |
| 205 | sej_apxgpt_irq_b | L | Level |
| 206 | sej_wdt_irq_b | L | Level |
| 207 | (Reserved) | - | - |
| 208 | smi_larb0_irq_b | L | Level |
| 209 | smi_larb1_irq_b | L | Level |
| 210 | smi_larb2_irq_b | L | Level |
| 211 | vdec_irq_b | L | Level |
| 212 | venc_irq_b | L | Level |
| 213 | jpgenc_irq_b | L | Level |
| 214 | seninf_irq_b | L | Level |
| 215 | cam0_irq_b | L | Level |
| 216 | cam1_irq_b | L | Level |
| 217 | cam2_irq_b | L | Level |
| 218 | disp_mutex_irq_b | L | Level |
| 219 | mdp_rdma_irq_b | L | Level |
| 220 | mdp_rsz0_irq_b | L | Level |
| 221 | mdp_rsz1_irq_b | L | Level |
| 222 | mdp_tdshp_irq_b | L | Level |
| 223 | mdp_wdma_irq_b | L | Level |
| 224 | mdp_wrot_irq_b | L | Level |
| 225 | disp_ovl0_irq_b | L | Level |
| 226 | disp_ovl1_irq_b | L | Level |
| 227 | disp_rdma0_irq_b | L | Level |
| 228 | disp_rdma1_irq_b | L | Level |
| 229 | disp_wdma0_irq_b | L | Level |
| 230 | disp_color_irq_b | L | Level |
| 231 | disp_ccorr_irq_b | L | Level |
| 232 | disp_aal_irq_b | L | Level |
| 233 | disp_gamma_irq_b | L | Level |

| GIC ID | Interrupt source/name | Polarity | Trigger type |
|---|---|---|---|
| 234 | disp_dither_irq_b | L | Level |
| 235 | disp_ufoe_irq_b | L | Level |
| 236 | dsi0_irq_b | L | Level |
| 237 | dpi0_irq_b | L | Level |
| 238 | mmsys_top_irq_b | L | Level |
| 239 | (Reserved) | - | - |
| 240 | (Reserved) | - | - |
| 241 | (Reserved) | - | - |
| 242 | mfg_irq_b[0] | L | Level |
| 243 | mfg_irq_b[1] | L | Level |
| 244 | mfg_irq_b[2] | L | Level |
| 245 | (Reserved) | - | - |
| 246 | (Reserved) | - | - |
| 247 | (Reserved) | - | - |
| 248 | (Reserved) | - | - |
| 249 | (Reserved) | - | - |
| 250 | (Reserved) | - | - |
| 251 | (Reserved) | - | - |
| 252 | (Reserved) | - | - |
| 253 | md_wdt_irq_b | L | Edge |
| 254 | (Reserved) | - | - |
| 255 | (Reserved) | - | - |
| 256 | (Reserved) | - | - |
| 257 | conn_wdt_irq_b | L | Edge |
| 258 | wf_hif_int_b | L | Level |
| 259 | conn2ap_btif_wakeup_out_b | L | Level |
| 260 | bt_cvsd_int_b | L | Level |
| 261 | (Reserved) | - | - |
| 262 | (Reserved) | - | - |
| 263 | cirq_event_irq_b | L | Level |
| 264 | (Reserved) | - | - |
| 265 | (Reserved) | - | - |
| 266 | (Reserved) | - | - |
| 267 | (Reserved) | - | - |
| 268 | (Reserved) | - | - |
| 269 | (Reserved) | - | - |
| 270 | (Reserved) | - | - |
| 271 | (Reserved) | - | - |
| 272 | (Reserved) | - | - |
| 273 | sec_vio_abort_n | L | Level |

### 3.1.4 Register Definition

This section describes the registers defined in the MCUCFG_REG block in the MCU system.

See chapater 1.1 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 3.2 On-chip Memory Controller

### 3.2.1 Introduction

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

### 3.2.2 Features

The memory controller has the following features

- 64KB on-chip ROM, with memory access protection and detection
- 64 KB on-chip SRAM, with memory access protection and detection
- 128 KB L2 share SRAM
- Chip ID

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

*Table 3-2. Memory map of on-chip memory controller*

| Bank | Start address | End address | Size | Device |
|---|---|---|---|---|
| 0 | 0x0000_0000 | 0x0000_FFFF | 64KB | ROM |
| | 0x0010_0000 | 0x0010_FFFF | 64KB | SRAM |
| | 0x0020_0000 | 0x0021_FFFF | 128KB | L2 Share SRAM |

### 3.2.3 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.

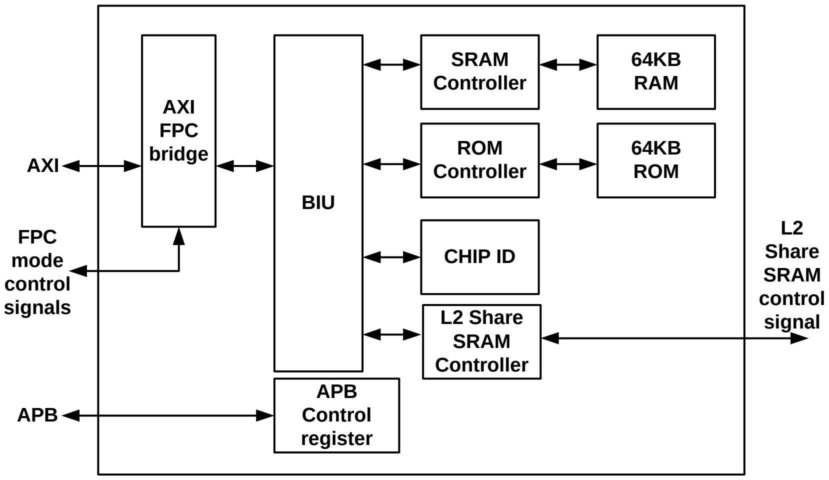


*Figure 3-1. Block diagram of on-chip memory controller*

### 3.2.3.1 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 *SRAMROM_BOOT_ADDR* register.

### 3.2.3.2 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.

### 3.2.3.3 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*.

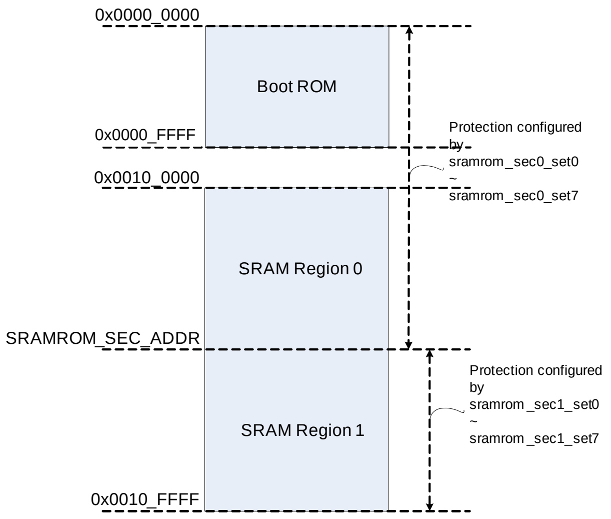

0x0000_0000
Boot ROM
0x0000_FFFF
Protection configured by sramrom_sec0_set0 ~ sramrom_sec0_set7
0x0010_0000
SRAM Region 0
SRAMROM_SEC_ADDR
Protection configured by sramrom_sec1_set0 ~ sramrom_sec1_set7
SRAM Region 1
0x0010_FFFF

***Figure 3-2. Security memory protection scheme***

### 3.2.4 Register Definition

See chapater 1.2 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 3.3 External Interrupt Controller

### 3.3.1 Introduction

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

### 3.3.2 Features

EINTC supports up to 213 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.

### 3.3.3 Block Diagram

Figure 3-3 is the block diagram of the external interrupt controller in MT6737. Every functional block is controlled by the corresponding control registers defined in next section.

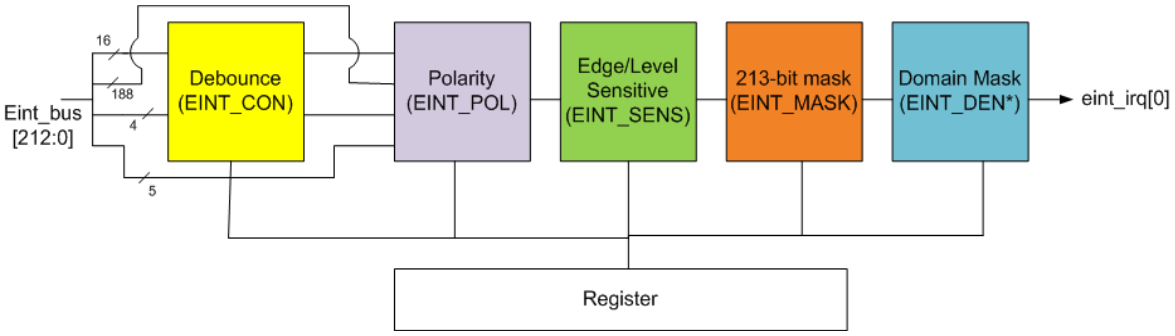


***Figure 3-3. Block diagram of external interrupt controller***

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 3-3. External interrupt request signal connection*

| IRQ name | AP MCU INTC |
|---|---|
| eint_irq[0] | IRQ[185] |
| eint_event_b | IRQ[186] |
| deint_irq[0] | IRQ[187] |
| deint_irq[1] | IRQ[188] |
| deint_irq[2] | IRQ[189] |
| deint_irq[3] | IRQ[190] |

*Table 3-4. Definitions of domains*

| Domain number | Target CPU/DSP |
|---|---|
| 0 | Application CPU |

*Table 3-5. EINT table*

| EINT Name | EINT number | Support HW debounce |
|---|---|---|
| GPIO[197:0] | 197~0<br>(137,143 cannot use) | Only 15~0 support debounce. |
| 6'd0 (unused) | 203~198 | N/A |
| USB_iddig | 204 | Yes |
| USB_vbusvalid | 205 | Yes |
| PMIC[1:0] | 207~206 | Yes |
| 5'd0 (unused) | 212~208 | N/A |

### 3.3.4 Register Definition

See chapater 1.3 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 3.3.5 Programming Guide

#### 3.3.5.1 Register Bit Set/Clear

For efficient bit set up/clear operation, the following registers have specific bit set/clear registers:

- EINT_MASK
- EINT_SOFT
- EINT_DBNC
- EINT_POL

Write 1 to the specified bit of the set/clear register to set up or clear the corresponding bit in the status register.

#### 3.3.5.2 Domain Control

For instance, if you would like int_bus[3] to trigger eint_irq[0], bit 3 in the EINT_D0EN register should be set.

#### 3.3.5.3 EINT De-bounce Control Sequence

1. Set up EINT_CON (PRESCALER, POL, CNT) and enable de-bounce (EN).
2. Wait for 3*32K (~100us).
3. Write RSTDBC (self-clear).
4. Wait for 3*32K (~100us).
5. Write EINT_INTACK to ack/clear all statuses.
6. Unmask EINT_MASK.

## 3.4 System Interrupt Controller

### 3.4.1 Introduction

For processors which have embedded interrupt controllers (GIC), the part of the MCUSYS will need to keep feeding clock and power to make 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 became 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, the MCUSYS can be completely turned off to improve system power consumption without losing interrupts.

### 3.4.2 Features

SYS_CIRQ supports up to 159 interrupts which can configure following attributes individually.

- Polarity inversion
- Edge/level trigger selection

The 159 interrupts will 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.

### 3.4.3 Block Diagram

Figure 3-4 is the system level block diagram of the system interrupt controller.

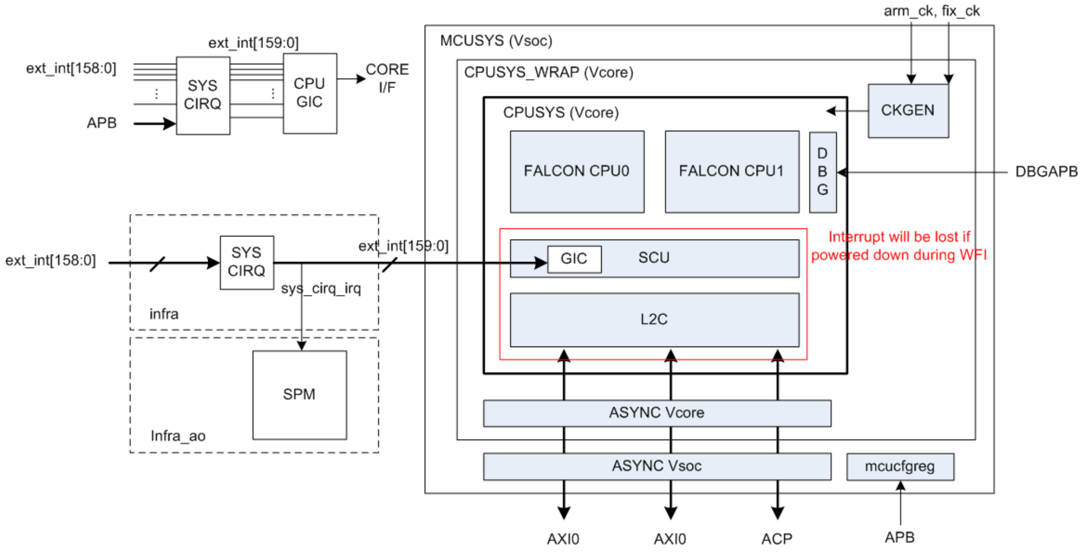


*Figure 3-4. System level block diagram of system interrupt controller*

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.

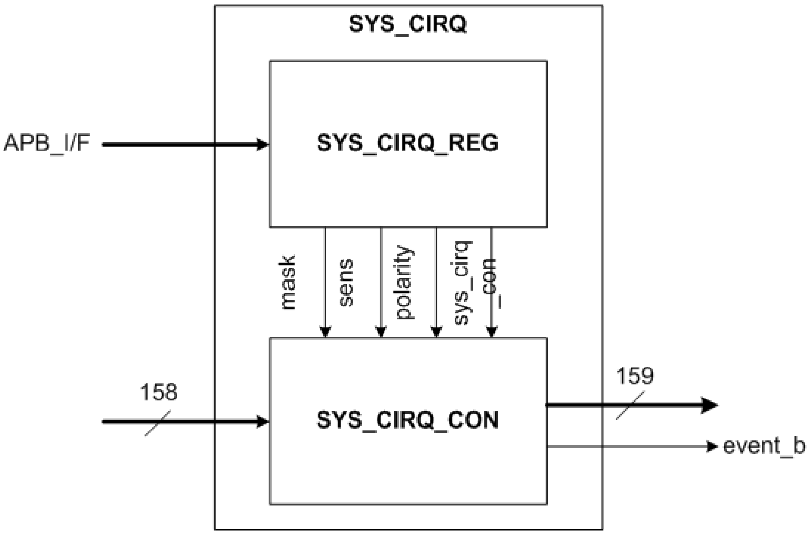


*Figure 3-5. Block diagram of system interrupt controller*

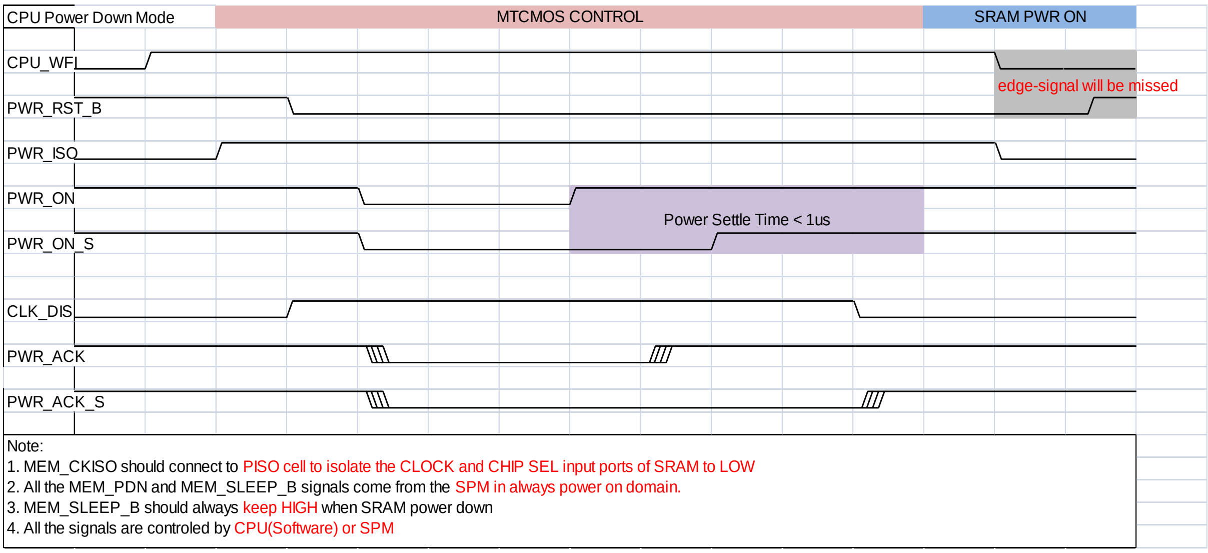
Figure 3-5 is the architecture of SYS_CIRQ. SYS_CIRQ_REG stores the mask/sensitivity/polarity attributes of each interrupt signal, and SYS_CIRQ_CON is used to mask and detect edge-triggered interrupts.

### 3.4.4 Register Definition

See chapater 1.4 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

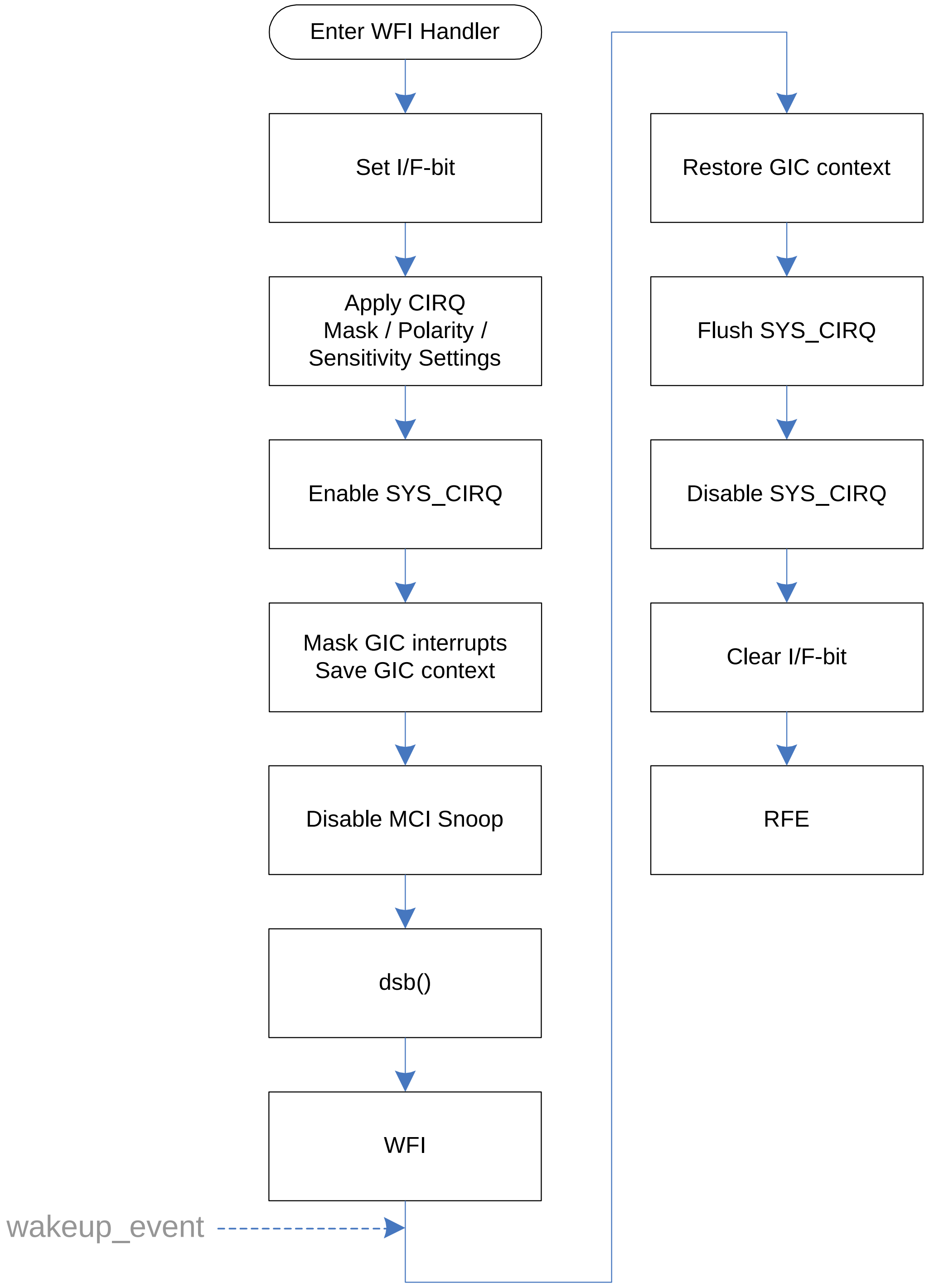
### 3.4.5 Programming Guide

#### 3.4.5.1 MCUSYS MTCMOS Sequence



Note:
1. MEM_CKISO should connect to PISO cell to isolate the CLOCK and CHIP SEL input ports of SRAM to LOW
2. All the MEM_PDN and MEM_SLEEP_B signals come from the SPM in always power on domain.
3. MEM_SLEEP_B should always keep HIGH when SRAM power down
4. All the signals are controled by CPU(Software) or SPM

#### 3.4.5.2 SW Flow


Enter WFI Handler
Set I/F-bit
Apply CIRQ Mask / Polarity / Sensitivity Settings
Enable SYS_CIRQ
Mask GIC interrupts Save GIC context
Disable MCI Snoop
dsb()
WFI
wakeup_event
Restore GIC context
Flush SYS_CIRQ
Disable SYS_CIRQ
Clear I/F-bit
RFE

## 3.5 Infrastructure System Configuration Module

### 3.5.1 Introduction

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

### 3.5.2 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

### 3.5.3 DCM in Details

The dynamic clock management function is used to slow down the clock frequency for power saving when the system is in idle state automatically.

Figure 3-6 gives a sample clock waveform when DCM is activated. In this example, the clock frequency in DCM mode is set to quarter of the original clock. The ratio of clock frequency slow-down is controlled by the INFRA_DCMFSEL register.

After the bus idle signal is low, it will take several cycles of latency to make the slow-down clock return to the normal frequency. The cycle number varies with the runtime status of the clock gating logic and will somehow cause minor performance impact. In order to minimize the impact when the system is in heavy load status, the INFRA_DCMDBC register controls the cycle count once the bus idle signal is asserted. Setting the de-bounce cycle to be longer and enabling the function will reduce the probability of the system entering the DCM mode.

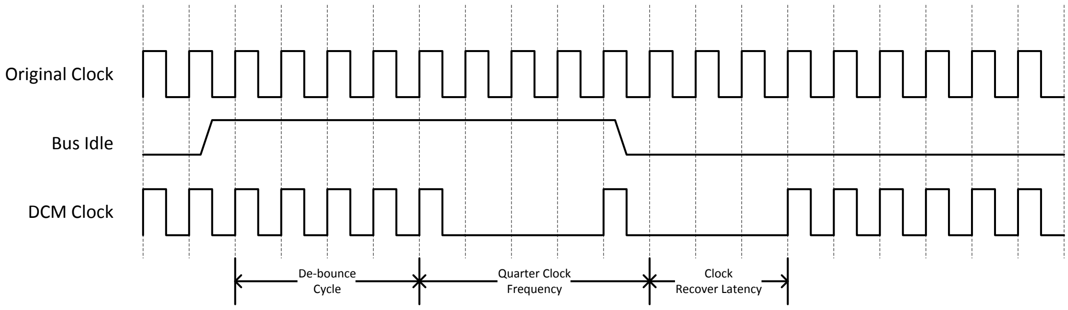


*Figure 3-6. DCM in action*

### 3.5.4 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 below for the location of SI node 0 to SI node 4 and the sleep protector and find the corresponding control registers in section 3.5.5.

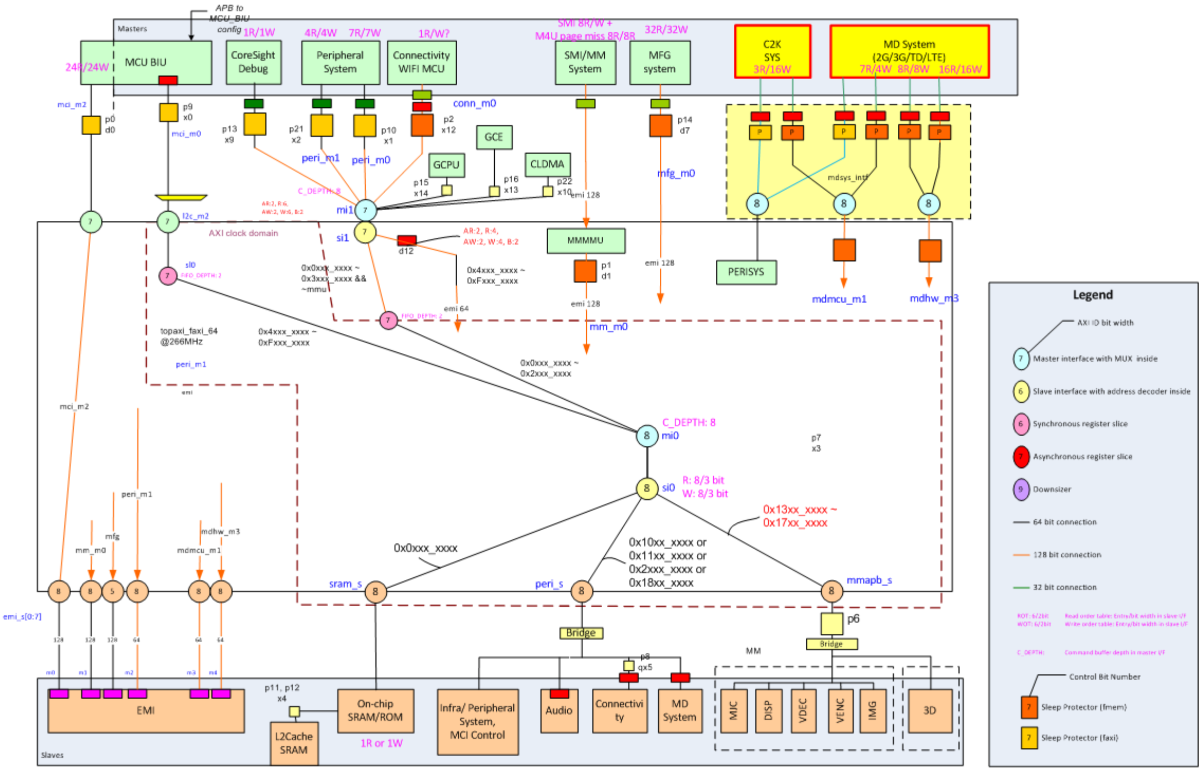


*Figure 3-7. Top AXI fabric and control blocks*

### 3.5.5 Register Definition

See chapater 1.5 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 3.6 External Memory Interface

### 3.6.1 Introduction

The EMI 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 320MHz bus clock frequency and can be scaled down to 234MHz.

### 3.6.2 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

### 3.6.3 Block Diagram

In MT6737, the DRAM controller connects three systems via six AXI ports and supports connecting two rank DRAM devices at the same time. For cortex APMCU system, we provide a 128-bit AXI port for the connection. For the multimedia system, we provide a 128-bit AXI port for the connection. Besides, we provide three 64-bit AXI ports for connecting to modem MCU, modem 2G/3G/4G HW and peripheral system. In the DRAM controller, the APB interface is for programming registers to initialize DRAM or other parameter settings.

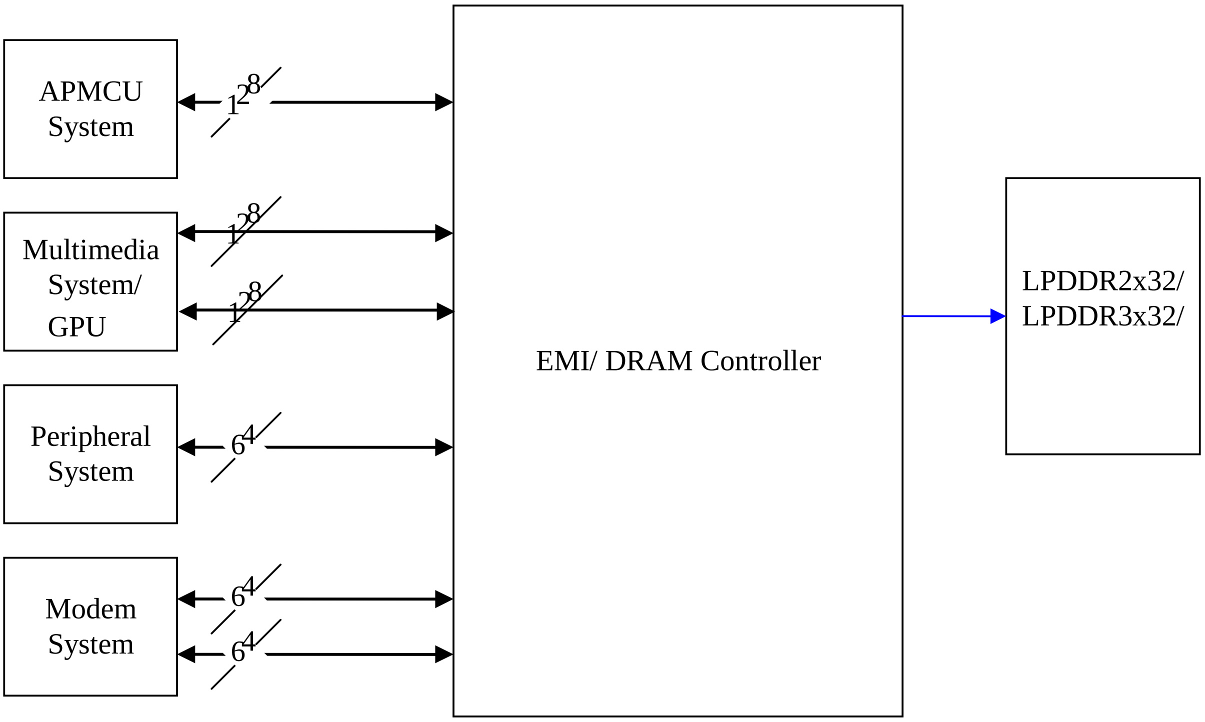


*Figure 3-8. EMI/DRAM controller top connection*

### 3.6.4 Register Definition

See chapater 1.6 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 3.6.5 Programming Guide

To start the EMI function, program the settings following the steps below:

1. Program to supporting one channel DRAM.
2. Program DRAM address mapping type, e.g. column bit number, row bit number and bank bit number. Set the address map to "row, ban, chn, col" or "ban[2], row, ban[1:0], chn, col" or "row, ban, col".
3. Set the DRAM bit number to 16 bits or 32 bits data bus.
4. Program the latency for each AXI interface for you to set the request to high priority when the age counter expires. (In the beginning, we treat all requests from one AXI port as the same ID.)
5. Allocate the bandwidth requirement for each AXI port and set up the bandwidth limiter value. Guarantee the total bandwidth that you do not set it to exceeding 100%.
6. Set the bandwidth limiter for each AXI ports to soft limit or hard limit.
7. Program the security region addresses and numbers.

## 3.7 DRAM Controller

### 3.7.1 Overview

DRAM controller supports the following DRAM bus configuration:
1. LPDDR2 32-bit @ 533MHz (1,066M bps/per bit channel)
2. LPDDR3 32-bit @ 640MHz (1,280M bps/per bit channel)

See the table below for the DRAM bus signals:

*Table 3-6. DRAM bus signal list (refer to DRAMC side)*

| Signal name | Type | Description |
|---|---|---|
| CK0/CK1 | Input | DRAM clock signal |
| CK0#/CK1# | Input | DRAM clock invert signal |
| MA[9:0] | Input | Address for all memories/CA bus for LPDDR2/3 |
| CKE | Input | Clock enable signal for DRAM |
| CS# [1:0] | Input | RANK1~RANK0 selection signal |
| DQ[31:0] | I/O | Data bus for LPDDR2/LPDDR3 |
| DQM[1:0] | Input | Data mask |
| DQS[3:0] | I/O | Data strobe |
| DQS#[3:0] | I/O | Differential data strobe in LPDDR2/3 |
| REXTDN | I/O | Output driving calibration |

See below for the DRAM bus command truth table:

*Table 3-7. DRAM bus command truth table (LPDDR2/3)*

| | | SDR Command Pins | | | DDR CA pins (10) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SDRAM Command | NVM Command | CKE | | CS_N | CA0 | CA1 | CA2 | CA3 | CA4 | CA5 | CA6 | CA7 | CA8 | CA9 | CK_t EDGE |
| | | CK_t(n-1) | CK_t(n) | | | | | | | | | | | | |
| MRW | MRW | H | H | L | L | L | L | L | MA0 | MA1 | MA2 | MA3 | MA4 | MA5 | ↑ |
| | | | | X | MA6 | MA7 | OP0 | OP1 | OP2 | OP3 | OP4 | OP5 | OP6 | OP7 | ↓ |
| MRR | MRR | H | H | L | L | L | L | H | MA0 | MA1 | MA2 | MA3 | MA4 | MA5 | ↑ |
| | | | | X | MA6 | MA7 | X | | | | | | | | ↓ |
| Refresh (per bank)[11] | - | H | H | L | L | L | H | L | X | | | | | | ↑ |
| | | | | X | X | | | | | | | | | | ↓ |
| Refresh (all bank) | - | H | H | L | L | L | H | H | X | | | | | | ↑ |
| | | | | X | X | | | | | | | | | | ↓ |
| Enter Self Refresh | Enter Power Down | H | L | L | L | L | H | X | | | | | | | ↑ |
| | | X | | X | X | | | | | | | | | | ↓ |
| Activate (bank) | Activate (row buffer) | H | H | L | L | H | R8/a15 | R9/a16 | R10/a17 | R11/a18 | R12/a19 | BA0 | BA1 | BA2 | ↑ |
| | | | | X | R0/a5 | R1/a6 | R2/a7 | R3/a8 | R4/a9 | R5/a10 | R6/a11 | R7/a12 | R13/a13 | R14/a14 | ↓ |
| Write (bank) | Write (RDB) | H | H | L | H | L | L | RFU | RFU | C1 | C2 | BA0 | BA1 | BA2 | ↑ |
| | | | | X | AP[3,4] | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 | ↓ |
| Read (bank) | Read (RDB) | H | H | L | H | L | H | RFU | RFU | C1 | C2 | BA0 | BA1 | BA2 | ↑ |
| | | | | X | AP[3,4] | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 | ↓ |
| Precharge (pre bank, all bank) | Preactive (RAB) | H | H | L | H | H | L | H | AB/a30 | X/a31 | X/a32 | BA0 | BA1 | BA2 | ↑ |
| | | | | X | X/a20 | X/a21 | X/a22 | X/a23 | X/a24 | X/a25 | X/a26 | X/a27 | X/a28 | X/a29 | ↓ |
| BST | BST | H | H | L | H | H | L | L | X | | | | | | ↑ |
| | | | | X | X | | | | | | | | | | ↓ |
| Enter Deep Power Down | Enter Power Down | H | L | L | H | H | L | X | | | | | | | ↑ |
| | | X | | X | X | | | | | | | | | | ↓ |
| NOP | NOP | H | H | L | H | H | H | X | | | | | | | ↑ |
| | | | | X | X | | | | | | | | | | ↓ |
| Maintain PD, SREF, DPD (NOP) | Maintain Power Down (NOP) | L | L | L | H | H | H | X | | | | | | | ↑ |
| | | | | X | X | | | | | | | | | | ↓ |
| NOP | NOP | H | H | H | X | | | | | | | | | | ↑ |
| | | | | X | X | | | | | | | | | | ↓ |
| Maintain PD, SREF, DPD (NOP) | Maintain Power Down (NOP) | L | L | H | X | | | | | | | | | | ↑ |
| | | | | X | X | | | | | | | | | | ↓ |
| Enter Power Down | Enter Power Down | H | L | H | X | | | | | | | | | | ↑ |
| | | X | | X | X | | | | | | | | | | ↓ |
| Exit PD, SREF, DPD | Exit Power Down | L | H | H | X | | | | | | | | | | ↑ |
| | | X | | X | X | | | | | | | | | | ↓ |

These tables are applied when CKE is asserted at the clock cycle before CS# is asserted. Read and write accesses to the DDR SDRAM are burst oriented. The accesses start at a selected location and continue for a programmed number of locations in a programmed sequence. The accesses begin with the registration of an ACTIVE command, followed by a READ or WRITE command. The address bits registered coincident with the ACTIVE command are used to select the bank and row to be accessed. The address bits registered coincident with the READ or WRITE command are used to select the bank and the starting column location for the burst access.

As with standard SDRAMs, the pipelined, multi-bank architecture of DDR SDRAMs allows for concurrent operation, thereby providing high effective bandwidth by hiding row precharge and activation time.

The DDR SDRAM operates from a differential clock (CK and CK#). Commands (address and control signals) are registered at every positive and negative edges of CK for LPDDR2/3. The input data are registered on both edges of DQS, and the output data are referenced to both edges of DQS, as well as to both edges of CK. DQS is center-aligned with data for WRITEs. Without DLL inside mobile DRAM's (LPDDR2/3), DQS is not edge-aligned with data for READs.

The commands for LPDDR2/3 SDRAM are encoded in MA0 ~ MA9 and transfer at double rate of clock frequency such as DQ.

Other key points of MT6737 DRAM controller:

- Supports column address bit number from 8 to 11
- Supports DRAM burst length 4 and 8
- Supports maximum LPDDR2/3 8G-bit device

### 3.7.2 Features

- 128-bit data bus interface with BE[15:0] for write data mask
- LEN has 4-bit, 0 ~ 15 DLE's/WDLE's
- Supports END_SIZE for optimize utilization rate
- Supports WDLE for split transaction
- Supports HDLE (DLE64) side band signal for early responses
- Supports high-priority side band signal for reducing request latency
- A request cannot cross page boundary
- Supports 2X frequency mode (frequency ratio of DRAMC:DRAM = 1:2) for timing optimization
- Supports dual-scheduler function for 1T command rate under 2X frequency mode for performance optimization
- Supports power-down and self-refresh for power saving
- Supports clock stop for power saving
- Supports input DQS/DQ timing calibration for PVT variation
- Supports read/write command out of order control

#### 3.7.2.1 Reference

- LPDDR2 spec: http://www.jedec.org/download/search/JESD209-2.pdf

### 3.7.3 Block Diagram

The major blocks of DRAM controller are command decoder, command pool, bus scheduler, timing controller DDR PHY and response generator.

The requests from Arbiter are pushed to command pool to wait for execution in order. The bus scheduler inspects the precharge/active pool and command FIFO and decides which DRAM bus command, e.g. PRECHARGE, ACTIVE, READ or WRITE, is issued to the DRAM bus. The goals of bus scheduler are to raise the bus utilization rate and lower the response latency. The timing control unit is responsible for the integrity of DRAM bus timing such as pre-charge to active delay (tRP), active to command delay (tRCD) and bus turnaround time. The bus scheduler refers to the information and choose the next DRAM bus command. The DRAM interface unit is responsible for generating DRAM bus commands, transmitting data and DQS to DDR DRAM and receiving data and DQS from DDR DRAM. The response generator produces the response signals for all DRAM agents such as DLE and RDAT.

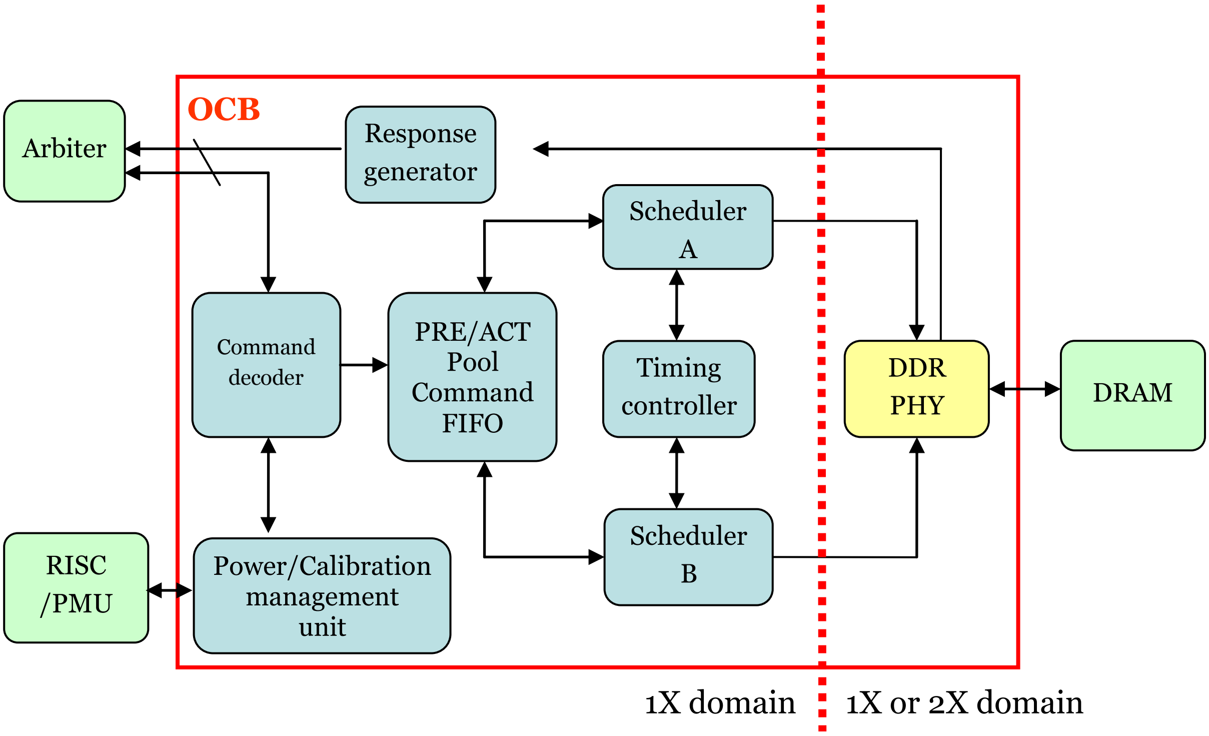


*Figure 3-9. Block diagram of DRAM controller*

### 3.7.4 Register Definition

See chapater 1.7 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

### 3.7.5 Programming Guide

**<u>Steps of DRAM initialization</u>**

1. Set up DRAM AC timing parameter.
2. Follow DRAM spec to complete DRAM initialization including mode register programming.
3. Enable calibration for DQ/DQS window.
4. Set up refresh rate counter.
5. Normal operation.

## 3.8 AP_DMA

### 3.8.1 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.

### 3.8.2 Features

APDMA has the following DMA engines.

- HIF DMA engine*1
- IRDA DMA engine*1
- I2C DMA engine*4
- UART0 TX DMA engine*1
- UART0 RX DMA engine*1
- UART1 TX DMA engine*1
- UART1 RX DMA engine*1
- UART2 TX DMA engine*1
- UART2 RX DMA engine*1
- UART3 TX DMA engine*1
- UART3 RX DMA engine*1
- UART4 TX DMA engine*1
- UART4 RX DMA engine*1
- UART4 TX DMA engine*1
- UART4 RX DMA engine*1

The DMA engines and corresponding peripheral devices are listed below.

***Table 3-8. Relationship between engines and devices***

| Engine | Peripheral device |
|---|---|
| HIF_0 | Connsys (sdctl) |
| IRDA_0 | IRDA |
| I2C_0 ~ I2C_3 | I2C_0 ~ I2C_3 |

| Engine | Peripheral device |
|---|---|
| UART0 ~ UART3 | UART0 ~ UART3 |
| UART4 | BTIF |

### 3.8.3 Block Diagram

Figure 3-10 is the basic block diagram of AP_DMA. There are total 15 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 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.

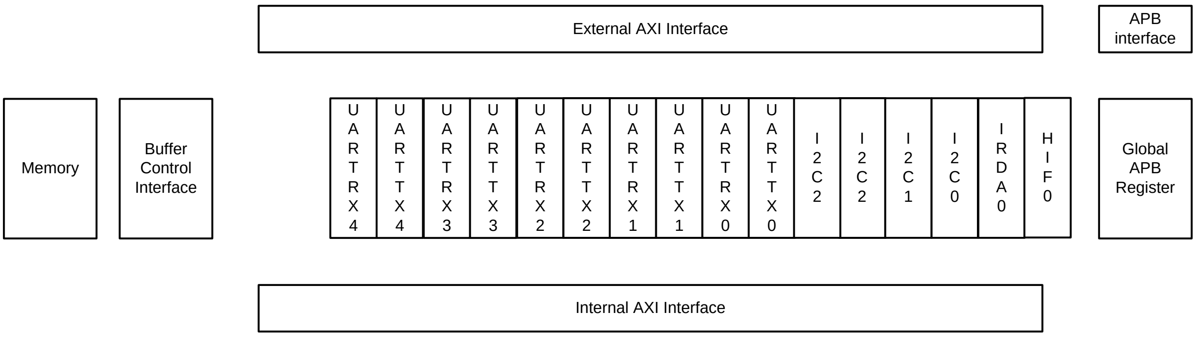


*Figure 3-10. APDMA block diagram*

### 3.8.4 Register Definition

#### 3.8.4.1 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.

See chapater 1.8 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 3.9 CQDMA

### 3.9.1 Introduction

There is always a DMA in a platform. The purpose of general DMA is performing data transfer between different slaves and purpose of audio DMA is performing data transfer to audio system from DRAM. 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.

### 3.9.2 Features

CQDMA has the following DMA engines.

- GDMA engine
- Audio DMA engine

### 3.9.3 Block Diagram

Figure 3-11 is the basic block diagram of CQ_DMA. There is one channel for general DMA and one channel for audio DMA. The external AXI interface is connected to DRAM to provide external memory access ability. 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.

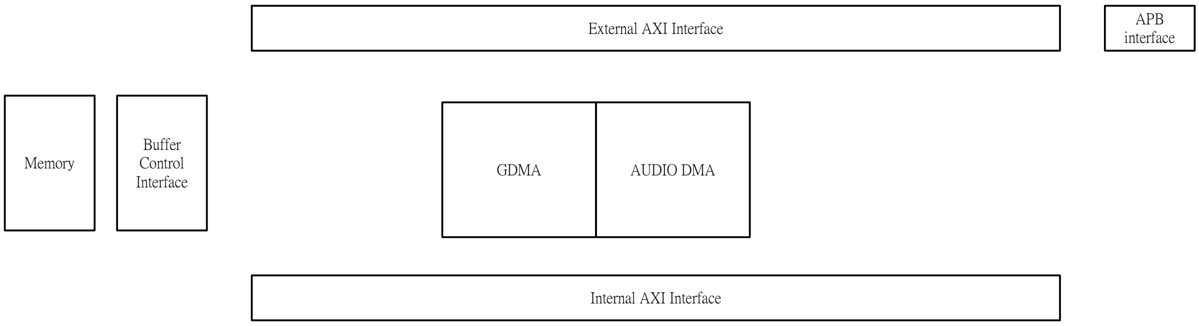


*Figure 3-11. CQDMA block diagram*

### 3.9.4 Register Definition

See chapater 1.9 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

# 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.

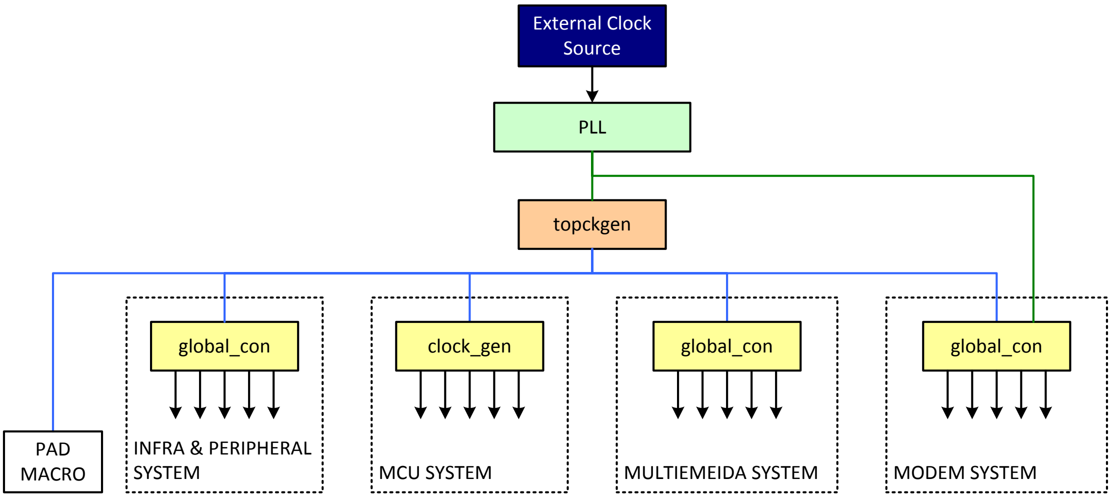


*Figure 4-1. Block diagram of clock architecture*

### 4.1.3.2 Clock Multiplixer

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.

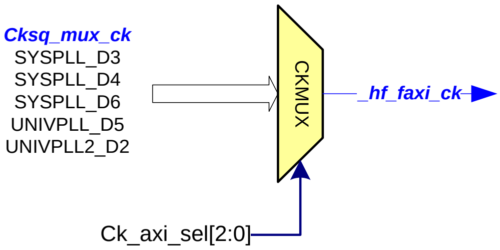


*Figure 4-2. Example of clock multiplixer*

### 4.1.4 Clock PLL

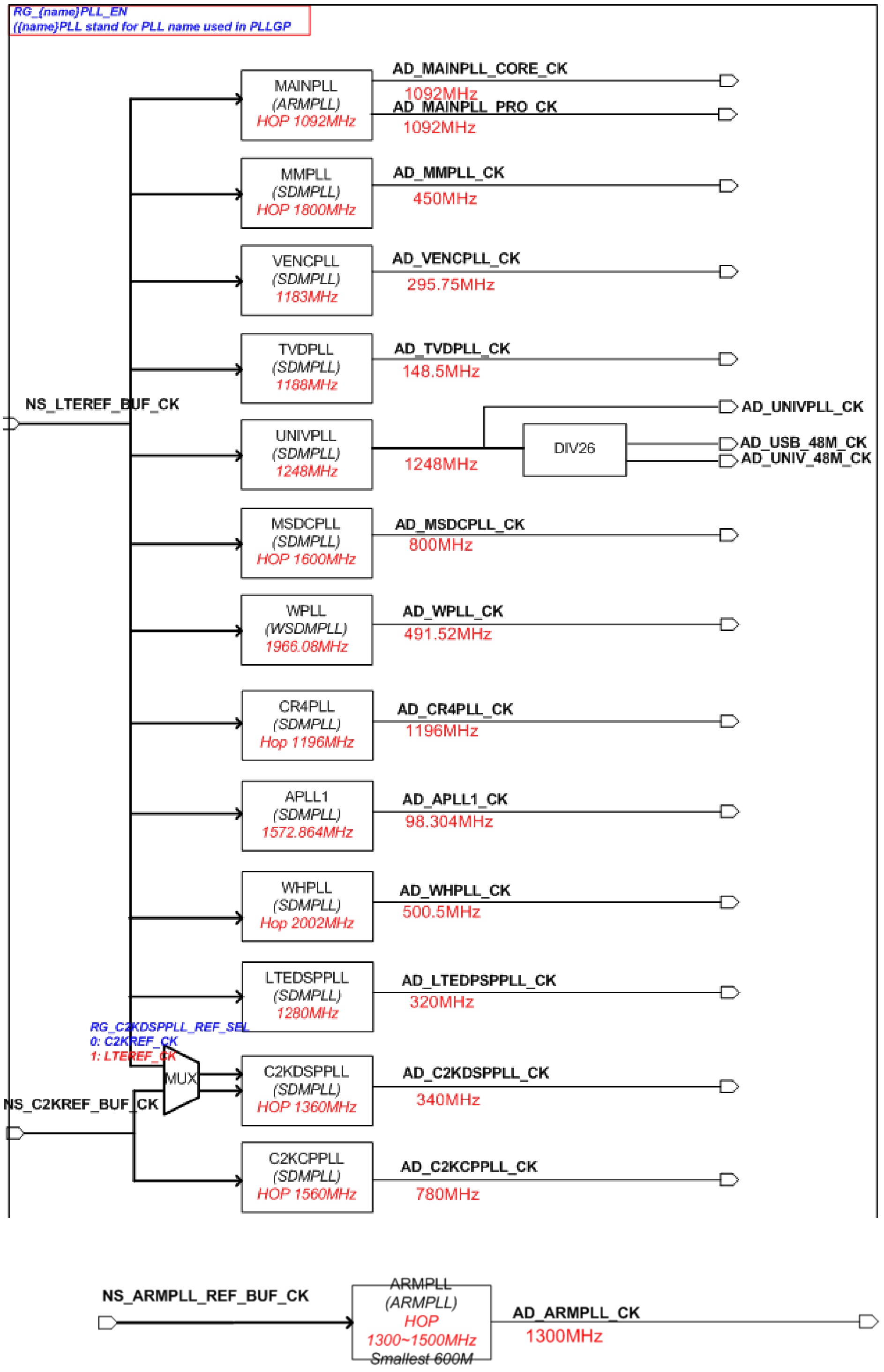


*Figure 4-3. PLL block diagram*

### 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 |
|---|---|---|---|
| ARMPLL | Hopping, SSC | Y | Y |
| MAINPLL | Hopping, SSC | Y | Y |
| UNIVPLL | Fix | N | Y |
| MSDCPLL | Hopping, SSC | Y | N |
| MMPLL | Hopping, SSC | Y | N |
| TVDPLL | Hopping, SSC | Y | N |
| VENCPLL | Hopping, SSC | Y | N |
| MEMPLL | Hopping, SSC | Y | Y |
| MIPIPLL | SSC | N | N |
| USB20_PHYA | Fix | N | N |
| APLL1 | Hopping, SSC | N | N |
| WPLL | Fix | By MD | By MD |
| WHPLL | Hopping, SSC | By MD | By MD |
| CR4PLL | Hopping, SSC | By MD | By MD |
| MDPLL | Fix | By MD | By MD |
| C2KCPPLL | Hopping, SSC | By MD | By MD |
| C2KDSPPLL | Hopping, SSC | By MD | By MD |
| MDPLL2_SUB1 (ABB) | Fix | By MD | By MD |
| MDPLL2_SUB2 (ABB) | Fix | By MD | By MD |
| LTEDSPPLL | Hopping, SSC | By MD | By MD |

### 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 |
| DCM_CFG | [7] | 1'b0 | dcm_enable | Enables hf_faxi_ck DCM |

| Register name | Bit | Default | Function name | Description |
|---|---|---|---|---|
| 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 DDRPHY |
|  | [2] | 1’b0 | sc_axick_off_en | Turns on scpsys control path to gate hf_faxi_ck |
|  | [4] | 1’b0 | sc_armck_off_en | Turns on scpsys control path to gate CA7 hf_farm_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 |
|  | [10] | 1’b0 | sc_armca15ck_off_en | Turns on scpsys control path to gate CA15 hf_farm_ck |
| CLK_SCP_CFG_1 | [0] | 1’b0 | sc_axi_26m_sel_en | Turns on scpsys control path to switch hf_faxi_ck to 26MHz |
|  | [4] | 1’b0 | sc_axick_dcm_dis_en | Turns on scpsys control path to disable DCM of hf_faxi_ck |

### 4.1.7 Frequency Meter

There are two frequency meters inside TOPCKGEN. One is for PLLs and TEST clock, called abist_fmeter. The other is for clocks generated from TOPCKGEN, called ckgen_fmeter.
Both structures have PAD output that can observe frequency directly instead of reading results from the frequency meter. Abist_fmeter is outputted to CLKM[0] and ckgen_fmeter is outputted to DEBUG_MON[2].

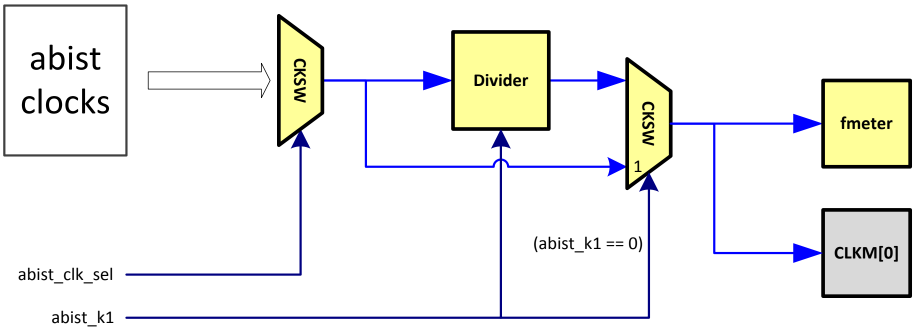


*Figure 4-4. ABIST FMETER structure*

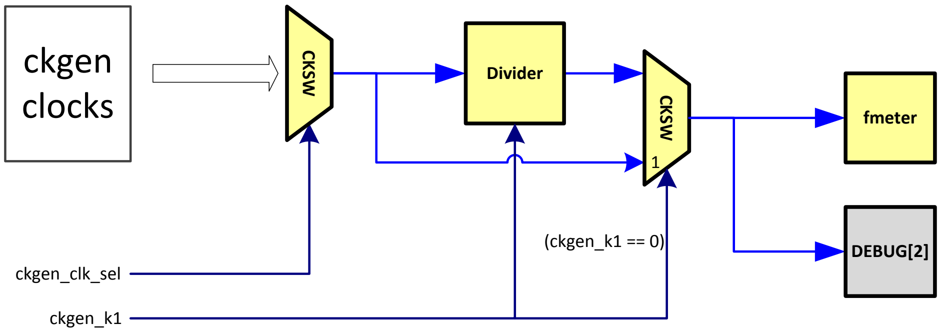


*Figure 4-5. CKGEN FMETER structure*

### 4.1.8 Register Definition

See chapater 2.1 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 4.2 Top Reset Generate 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

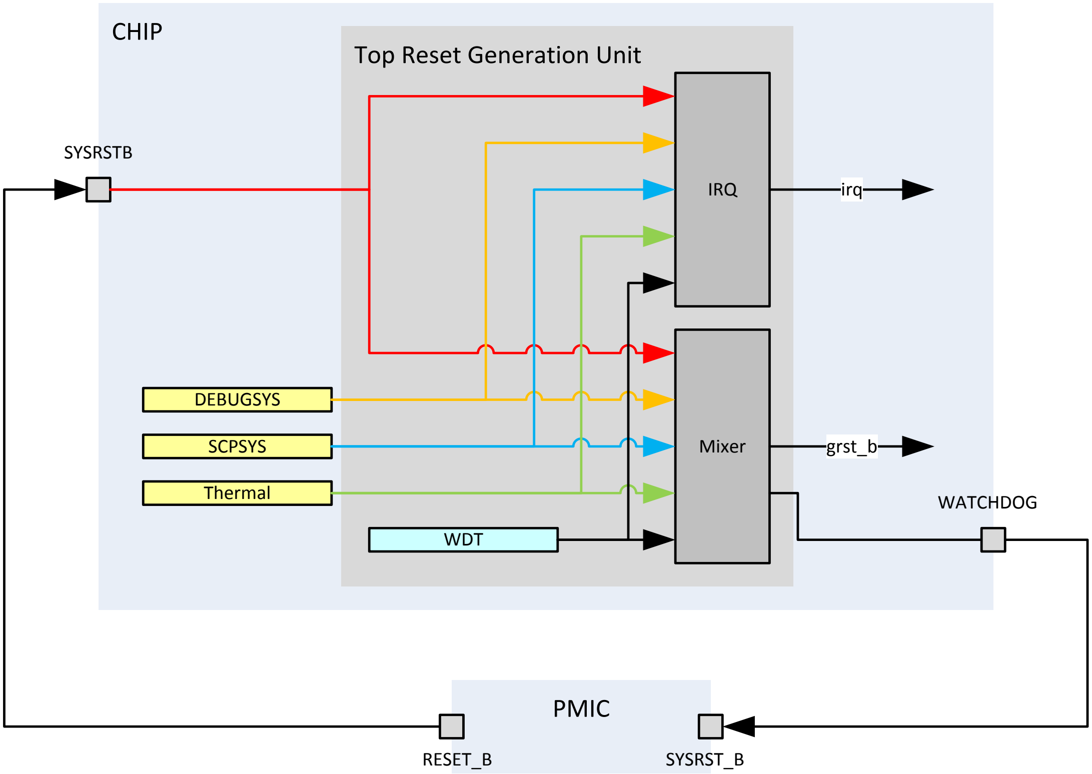


*Figure 4-6. Block diagram of top reset generation unit*

### 4.2.4 Register Definition

See chapater 2.2 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 4.2.5 Programming Guide

#### 4.2.5.1 TOPRGU Initial

Enable dual mode reset when TOPRGU is first initialized. Because WDT_MODE will not be reset and the dual mode will be disabled if system reset is triggered through WDT_SWRST, these registers will only be reset by SYSRSTB.

The following registers will not be reset by TOPRGU.

- WDT_MODE
- WDT_STA
- WDT_NONRST_REG
- WDT_NONRST_REG2
- WDT_REQ_MODE
- WDT_REQ_IRQ_EN
- WDT_DEBUG_CTL

#### 4.2.5.2 Watchdog Timer

- Trigger WDT_RESTART right after WDT_LENGTH is updated.
- WDT_SWRST can be triggered without wdt_en set to 1'b1.
- It is recommended to trigger WDT_RESTART before setting wdt_en to 1'b1.

#### 4.2.5.3 IRQ Mode

Dual mode reset is default on. Therefore, all reset requests are default with IRQ mode enabled. This means the interrupt is triggered instead of triggering system reset immediately. If you would like to trigger system reset instead of interrupt, change the corresponding configuration of each reset request.

Each reset request can be configured as reset or IRQ separately.

#### 4.2.5.4 MDSYS and CONNSYS Watchdog Timout

MDSYS and CONNSYS have their own watchdog timer. When their watchdog timers expire, they notify AP through interrupts. AP then asserts software reset to MDSYS or CONNSYS.

- MDSYS
  - Enable bus protection to/from MDSYS.
  - Set md_rst = 1'b1.
  - Wait for 2T 32kHz then set md_rst = 1'b0.
  - Disable bus protection to/from MDSYS.
  - Set up MDSYS boot slave.
  - Inform MDSYS abnormal reset via CCIF after MDSYS is ready.

- CONSYS
  - conn_rst = 1'b1
  - Wait for 2T 32kHz then set conn_rst = 1'b0.

### 4.2.5.5 Dual Mode Reset

Dual mode reset is system reset after TOPRGU triggers interrupt. The watchdog timer needs to be enabled to complete this function.

In this mode, the watchdog timer will be ***AUTO-RESTART*** after interrupt is triggered. AP needs to clear WDT_STA after receiving interrupt from TOPRGU, or system reset will be triggered after watchdog timer expires.

- Set wdt_en = 1'b1.
- Set dual_mode = 1'b1.
- Set wdt_irq, thermal_irq, scpsys_irq or debug_irq to 1'b1.

### 4.2.5.6 DDR Protect

DDR protect (rg_ddr_protect_en) is useless when DDR reserved mode is enabled.

### 4.2.5.7 DDR Reserved Mode Reset

DDR reserved mode keeps data in DDR during system reset. In order to complete this function, DRAMC, DRMC_CONF, DDRPHY_CONF and EMI_CONF (optional) will not be reset.

- Enable DDR reserved mode when initializing TOPRGU.
- Wait for system reset to be triggered.
- [Optional] Check DDR reserved mode status (ddr_reserve_sta).
- After system reset, release DRAMC_CONF protect (set rg_dramc_conf_iso = 1'b0).
- Ensure related clocks of EMI, DRAMC, DDRPHY are ready (including PLL).
- Wait for dramc_sref_sta = 1'b1.
- Release DRAMC protect (set rg_dramc_iso = 1'b0).
- Release DRAMC self-refresh control (set rg_dramc_sref = 1'b0).
- Wait for dramc_sref_sta = 1'b0.

### 4.2.5.8 History

- EXT RESET is for NAND and PMIC, but there is exception.
  - MT6329 needs to disable WDT_MODE[2].
- WDT_LENGTH needs to be reset, or SYSTEM will enter RESET loop if WDT_LENGTH has been set to very short before reset trigger.

## 4.3 PMIC Wrapper

### 4.3.1 Introduction

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

### 4.3.2 Features

- 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

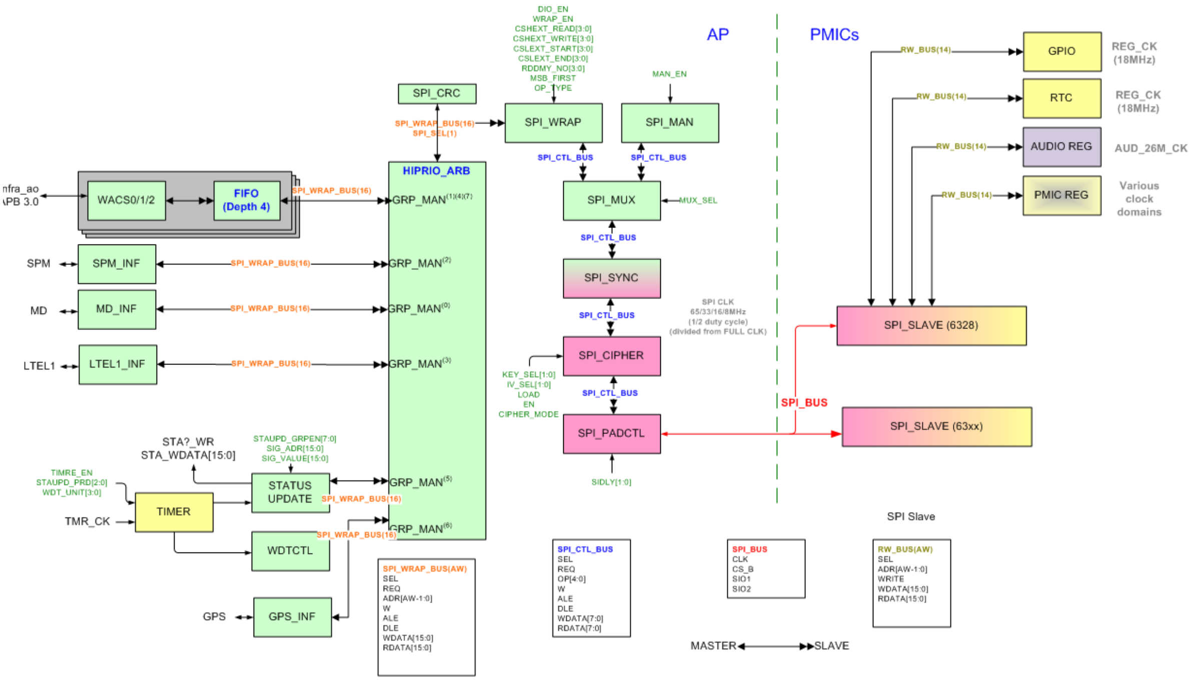


***Figure 4-7. PMIC_WRAP architecture***

### 4.3.4 Register Definition

See chapater 2.3 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 4.4 Frequency Hopping Controller

### 4.4.1 Introduction

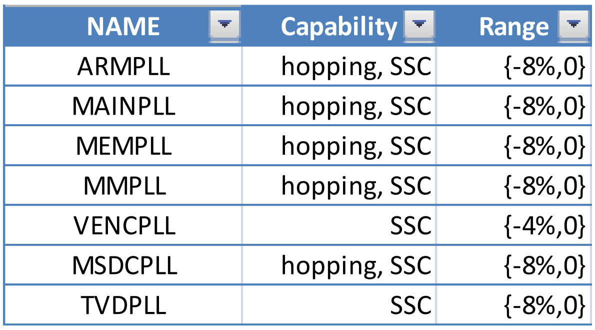
The frequency hopping controller helps AP resolve de-sense issues. The RF victims are 2G, 3G, BT, FM, Wi-Fi, etc. The aggressor in AP is the clock generated from PLL in ABB. The harmonic of all clock frequency may de-sense the band of RF system.

### 4.4.2 Features

The frequency hopping controller receives the command from CPU to trigger the following two mechanisms:

- Spread spectrum clocking
- Frequency hopping

In MT6737, there are 7 hopping PLLs:



| NAME | Capability | Range |
|---|---|---|
| ARMPLL | hopping, SSC | {-8%,0} |
| MAINPLL | hopping, SSC | {-8%,0} |
| MEMPLL | hopping, SSC | {-8%,0} |
| MMPLL | hopping, SSC | {-8%,0} |
| VENCPLL | SSC | {-4%,0} |
| MSDCPLL | hopping, SSC | {-8%,0} |
| TVDPLL | SSC | {-8%,0} |

### 4.4.3 Block Diagram

Whenever MCUSYS enters sleep mode, SRCLKENA from the sleep controller is de-asserted. The SRCLKENA from MCUSYS controls the power supply for the 13MHz/26MHz TCVCXO via the on-chip PMU. When the signal is de-asserted, LDO for TCVCXO in PMU will be turned off, and 13MHz/26MHz clock will stop.

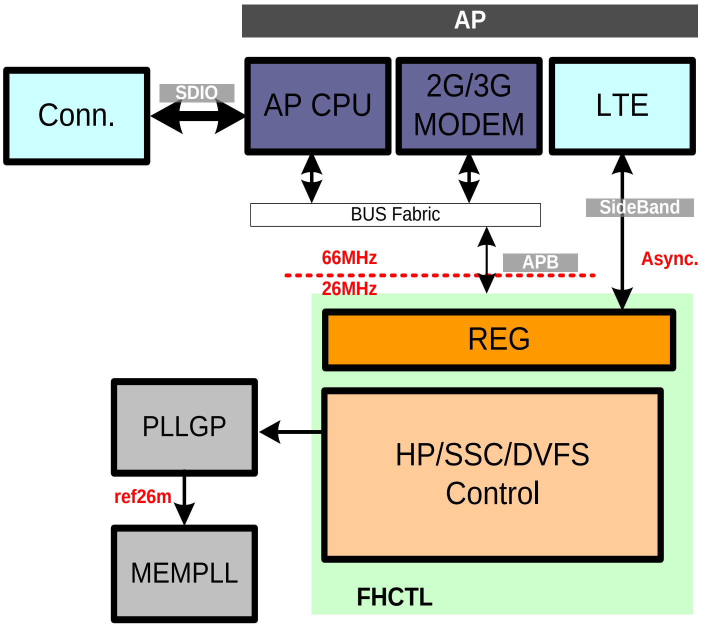


***Figure 4-8. Block diagram of frequency hopping controller***

### 4.4.4 Register Definition

See chapater 2.4 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

# 5 Peripherals

## 5.1 Pericfg Controller

### 5.1.1 Introduction

The pericfg controller is used to control the reset, clock and bus setting of peripheral subsys. Each module inside the peripheral subsys has its own software reset and clock gated control (power-down control). The hardware DCM (Dynamic Clock Management) of the peripheral subsys is also controlled in the pericfg controller. Beside AP MCU, the modem MCU can also use this pericfg controller to control specific modules clock gated control (power-down control).

### 5.1.2 Features

- Supports software reset control of each module inside peripheral subsys
- Supports clock gated control of the modules insider peripheral subsys by AP MCU
- Supports clock gated control of the modules insider peripheral subsys by Modem1 MCU
- Supports clock gated control of the modules insider peripheral subsys by Modem2 MCU
- Supports DCM control of peripheral subsys
- Supports bus setting (bandwidth limit/way enable/…) of peripheral subsys

### 5.1.3 Block Diagram

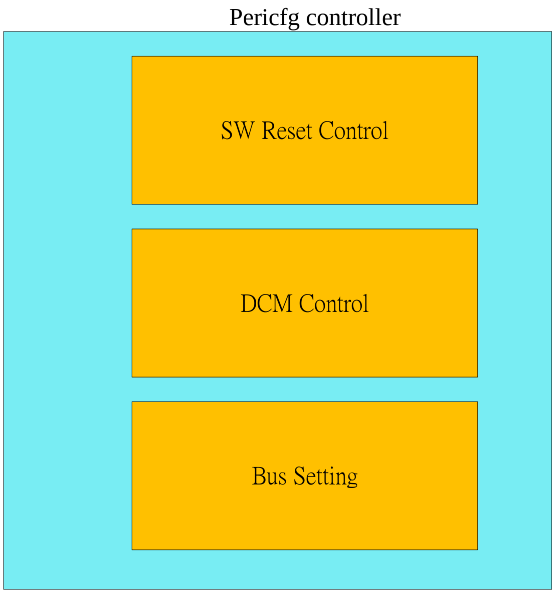


***Figure 5-1. Block diagram of pericfg controller***

### 5.1.4 Register Definition

See chapater 3.1 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 5.2 GPIO Control

### 5.2.1 General Descriptions

There are 198 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.

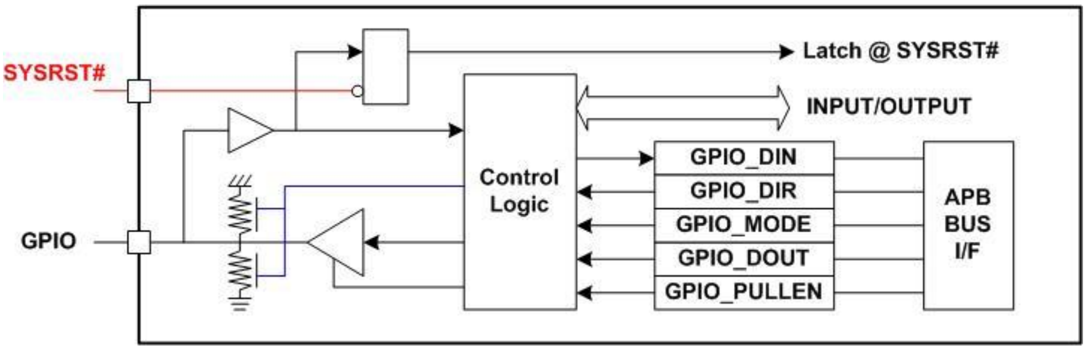


*Figure 5-2. GPIO block diagram*

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.

When the MIPI function is not used, related IOs can be switched to non-MIPI input-only function, like EINT. To enable the GPI function of MIPI, there are some configurations to be done before the related GPIO_MODE is set.

### 5.2.2 Register Definition

See chapater 3.2 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 5.3 Keypad Scanner

### 5.3.1 General Description

The keypad supports two types of keypads: 8*8 single keys and 3*3 configurable double keys.

The 8*8 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). The key detection block provides key pressed, key released and de-bounce mechanisms.

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 8*8 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.

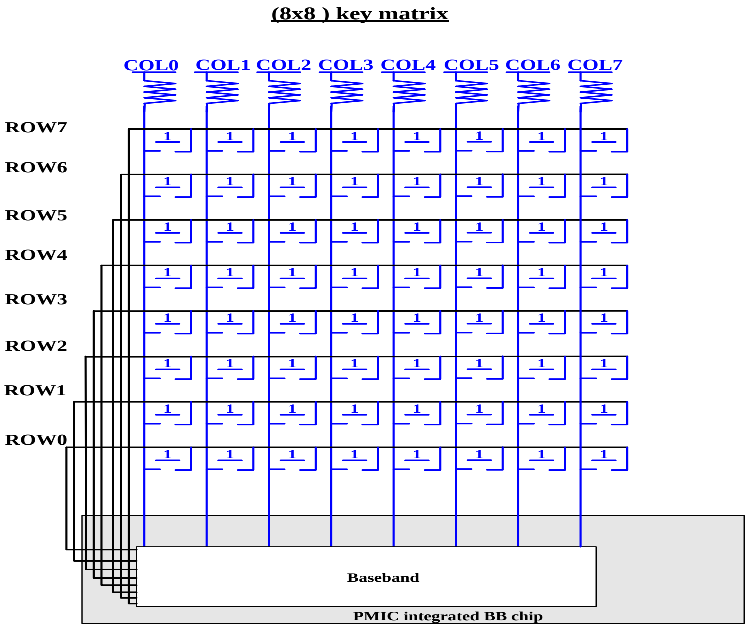


*Figure 5-3. 8x8 keypad matrix (64 keys)*

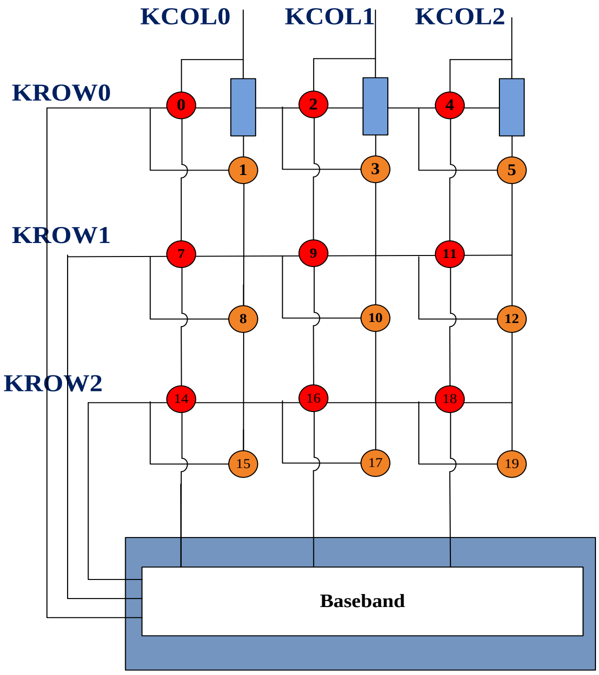


*Figure 5-4. 3x3 keypad matrix (18 keys)*

### 5.3.2 Waveform

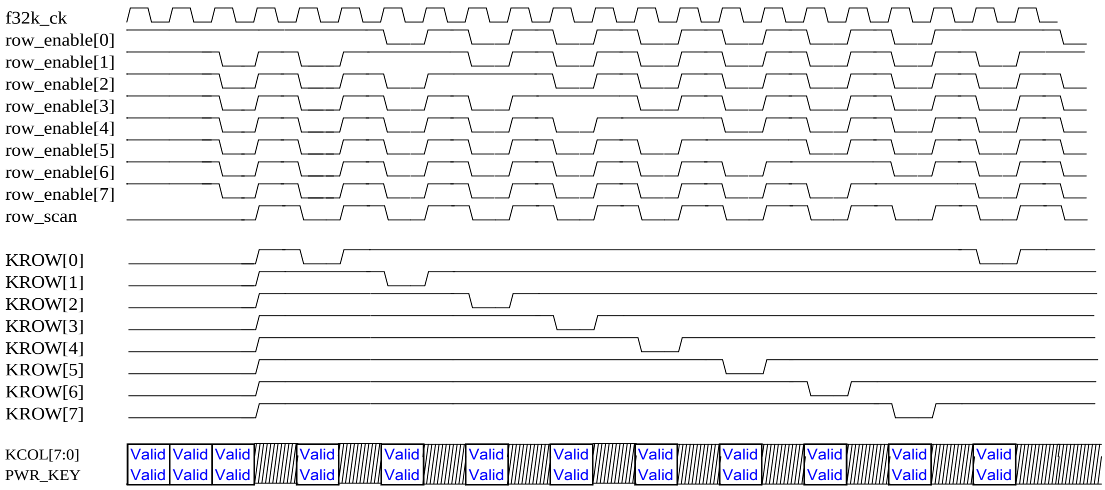


*Figure 5-5. 8x8 keypad scan waveform*

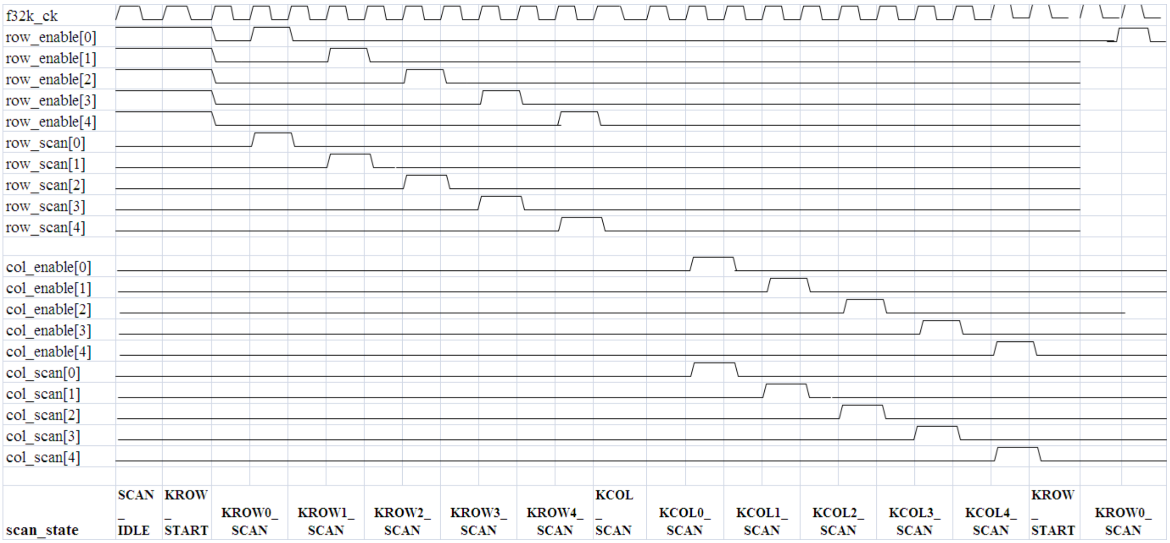


*Figure 5-6. 5*5 keypad scan waveform*

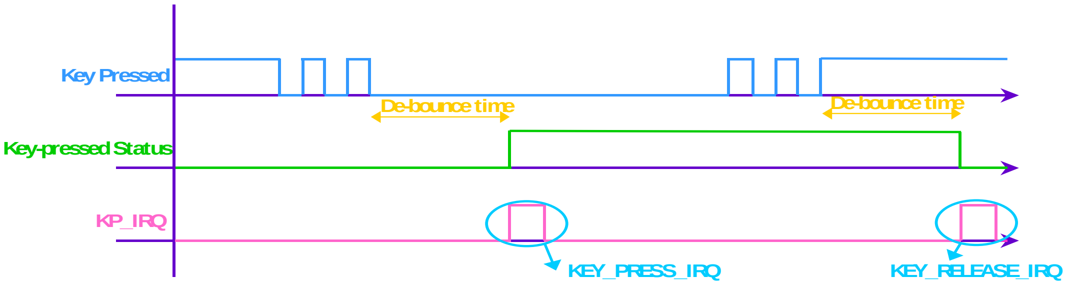


*Figure 5-7. One key pressed with de-bounce mechanism denoted*

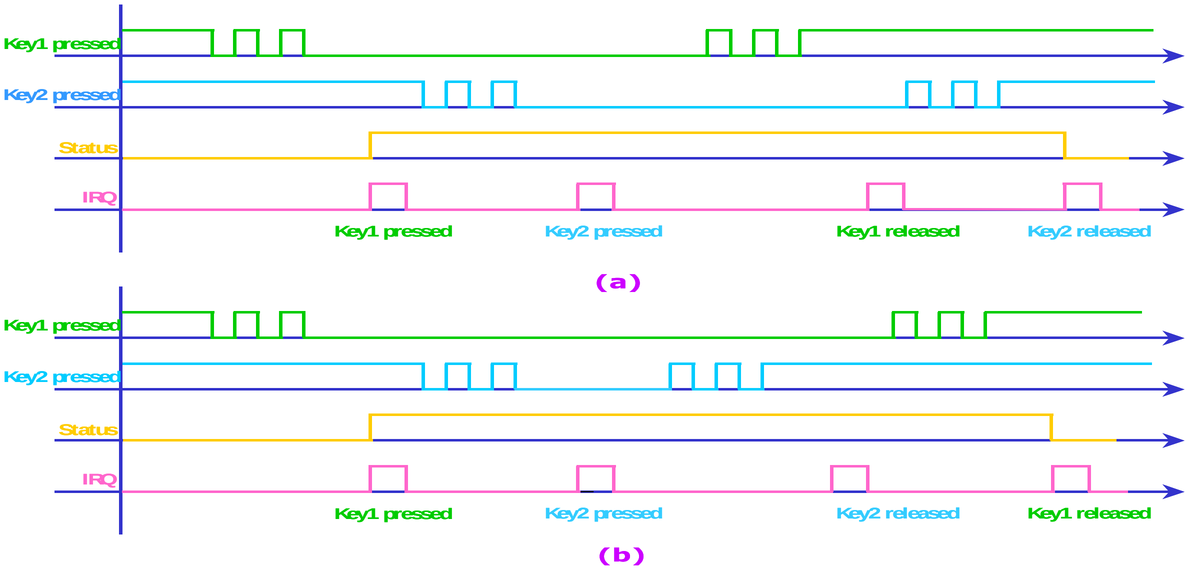


*Figure 5-8. (a) Two keys pressed, case 1; (b) Two keys pressed, case 2*

### 5.3.3 Register Definition

See chapater 3.3 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 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 4 channels
- DMA, polling or interrupt operation
- Supports word lengths from 5 to 8 bits, with an optional parity bit and one or two stop bits
- 4 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

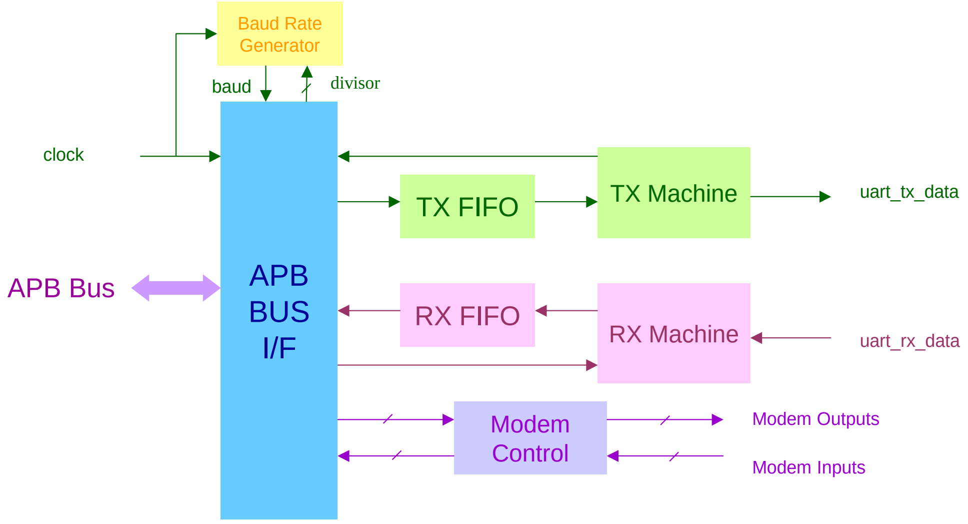


***Figure 5-9. Block diagram of UART***

### 5.4.4 Register Definition

| UART number | Base address | Feature |
|---|---|---|
| UART0 | 0x11002000 | Supports DMA, HW flow control |
| UART1 | 0x11003000 | Supports DMA, HW flow control |
| UART2 | 0x11004000 | Supports DMA, HW flow control |
| UART3 | 0x11005000 | Supports DMA, HW flow control |

There are four UART IPs in this SOC. The usage of the registers below are the same except that the base address must be changed to respective one.

See chapater 3.4 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 5.4.5 Programming Guide

#### 5.4.5.1 Auto Baud Rate Detection

UART can detect 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 8 endpoints to receive packets and 8 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 list | Description |
|---|---|
| USB specifications | USB2.0 OTG |
| Enhanced feature | Generic Host QMU<br>Generic Dev QMU |
| Endpoint | 8 Tx<br>8 Rx<br>EP0 |
| DMA channel | 8 |
| 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

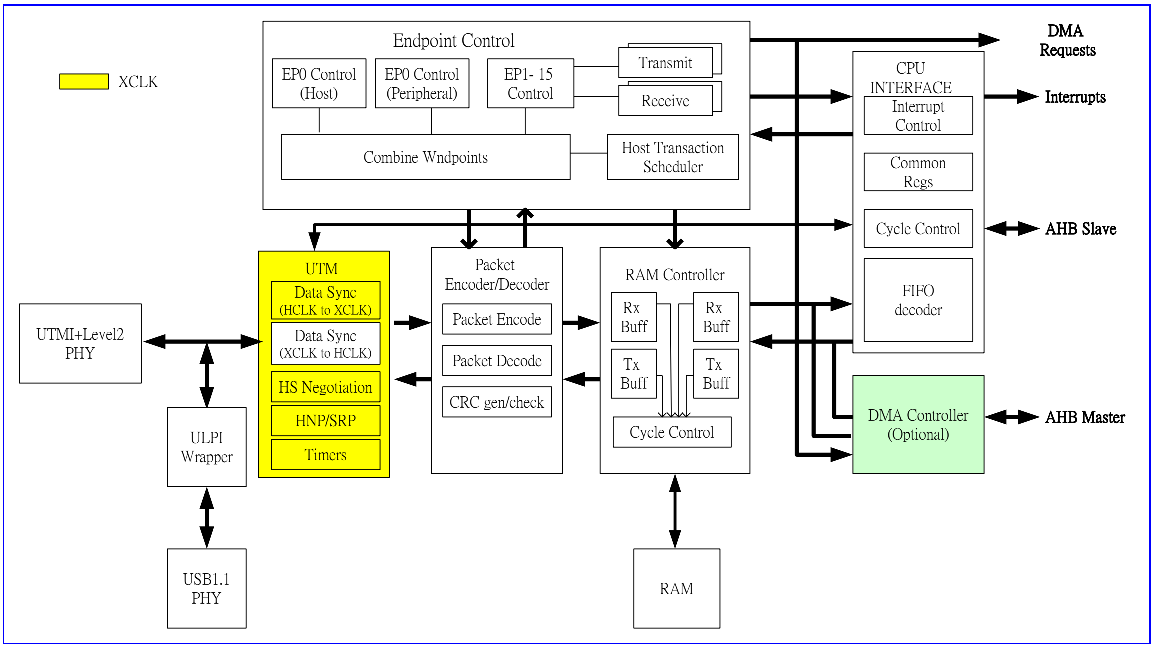


*Figure 5-10. USB controller block diagram*

### 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 |
| **Indexed EndPoint CSR Region**<br>*n stands for endpoint number.*<br>*For example, endpoint 1's n = 1. Valid n = 1 ~ MaxEndPoint.* | | | |

| Register address | Register name | Manipulation (Byte/Word) | Acronym |
|---|---|---|---|
| *MaxEndPoint is hardware configured and the maximum is 15.* | | | |
| USB + 0010h<br>~<br>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 |
| USB + 00ACh | USB Level 1 interrupt control register | Byte | USB_L1INTC |
| **Non-indexed EndPoint CSR Region**<br>*n stands for endpoint number.*<br>*For example, endpoint 1's n = 1. Valid n = 1 ~ MaxEndPoint.*<br>*MaxEndPoint is hardware configured and the maximum is 15.* | | | |
| USB + 0102h | EP0 control status register | Byte | CSR0 |

| Register address | Register name | Manipulation (Byte/Word) | Acronym |
|---|---|---|---|
| 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**<br>*M stands for DMA channel number.*<br>*For example, DMA channel 1's M = 1. Valid M = 1 ~ MaxDMAChannel.*<br>*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**<br>*n stands for endpoint number. For example, endpoint 1's n = 1. Valid n = 1 ~ MaxEndPoint.*<br>*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)**<br>*n stands for endpoint number. For example, endpoint 1's n = 1. Valid n = 1 ~ MaxEndPoint.*<br>*MaxEndPoint is hardware configured and maximum is 15.* | | | |
| USB + 0480h +8*n h | Transmit endpoint n function address | Word | TXFUNCADDR |

| Register address | Register name | Manipulation (Byte/Word) | Acronym |
|---|---|---|---|
| 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 chapater 3.5 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 5.6 USBPHY Register File

The full features include USB2.0 PHYD and PHYA macro control registers. It also includes a frequency meter for USB2.0 PHYA monitor clock. 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
- Accessing PHY registers by AHB slave interface
- Accessing PHY registers by I2C interface

### 5.6.2 USBPHY Register File Block Diagram

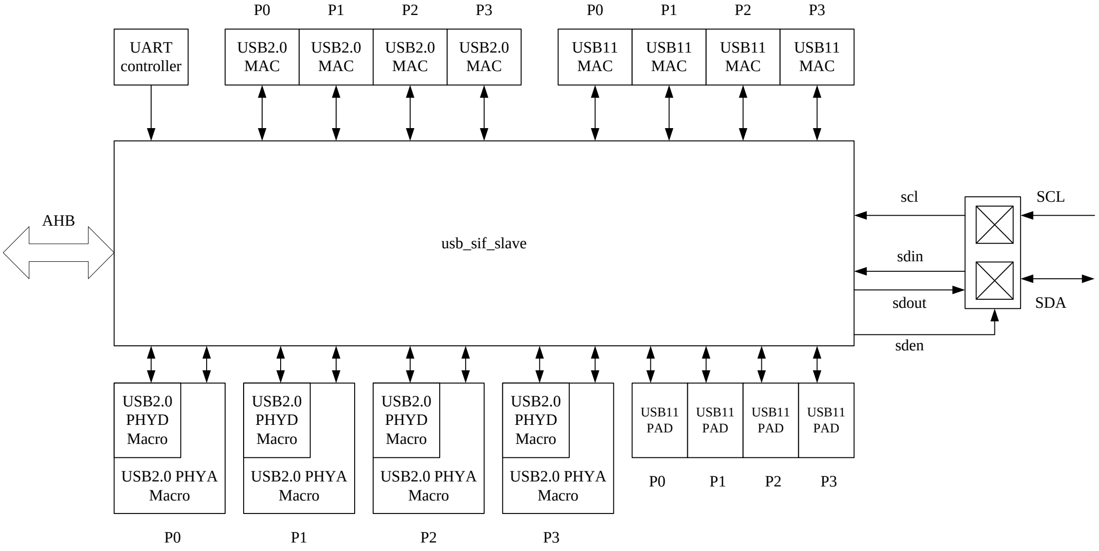
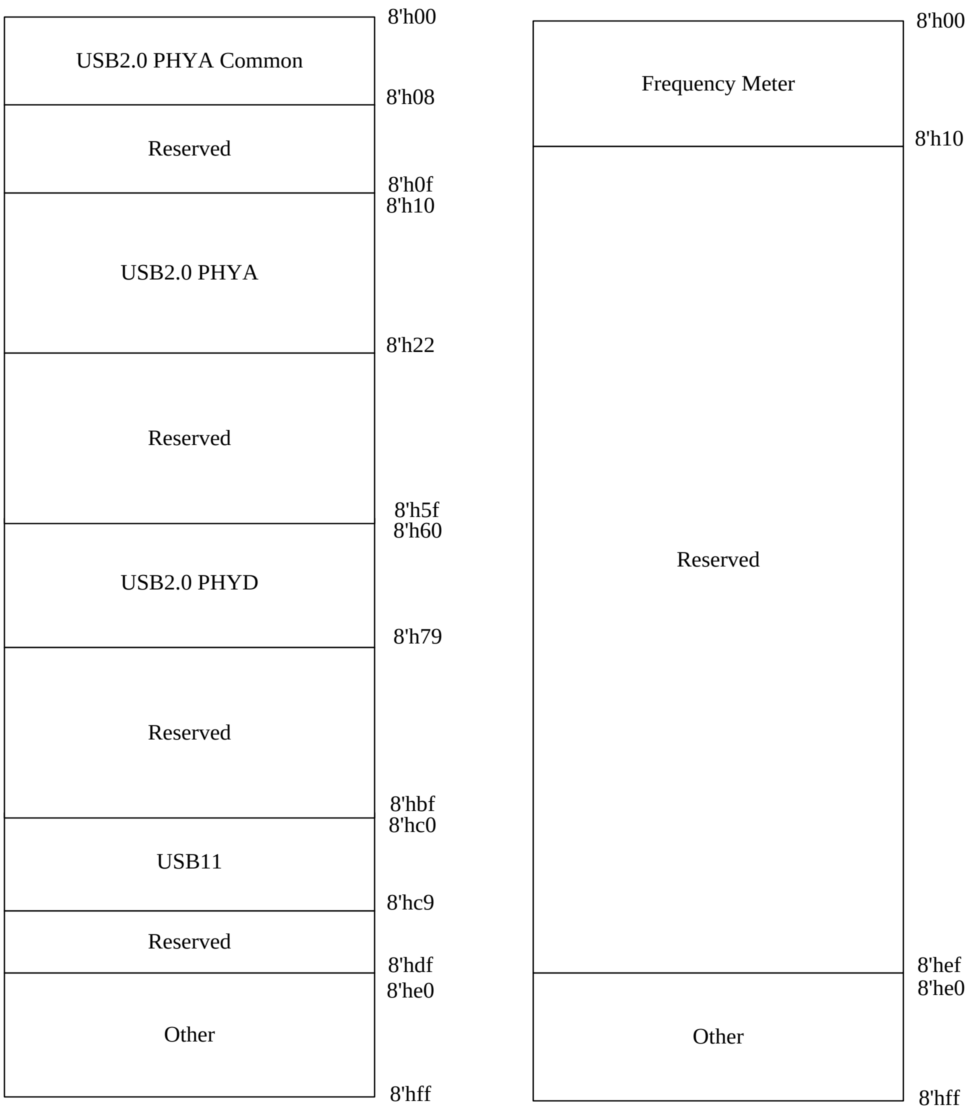


*Figure 5-11. USBPHY RegFile block diagram*

### 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
  - Supporting 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



See chapater 3.6 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 5.7 SPI Interface Controller

### 5.7.1 Introduction

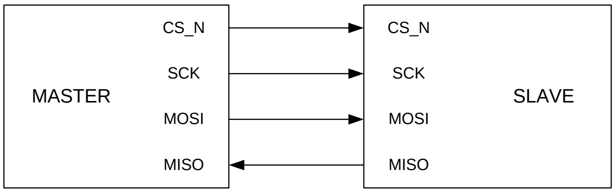


***Figure 5-12. Pin connection between SPI master and SPI slave***

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. The SPI interface controller is a master responsible of the data transmission with the slave.

### 5.7.2 Pin Description

***Table 5-1. SPI controller interface***

| Signal name | Type | Description |
|---|---|---|
| CS_N | O | Low active chip selection signal |
| SCK | 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 |

### 5.7.3 Transmission Formats

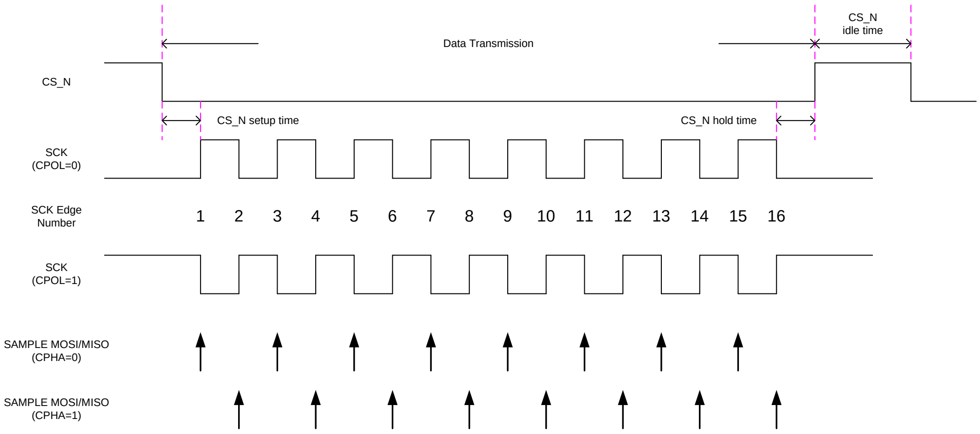


***Figure 5-13. SPI transmission formats***

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 is an example of both clock polarities (CPOL).

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

### 5.7.4 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.

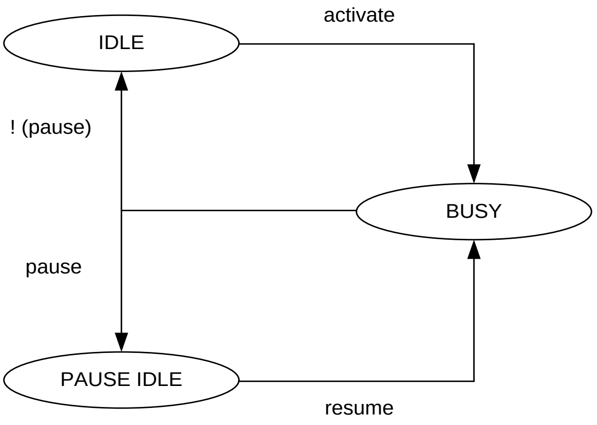


*Figure 5-14. Operation flow with or without PAUSE mode*

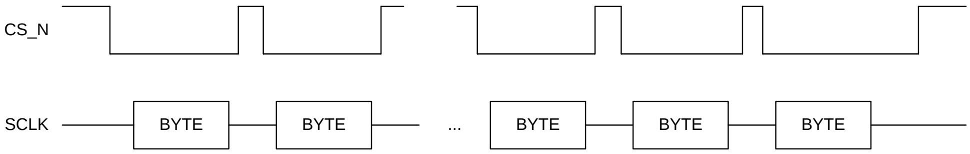


*Figure 5-15. CS_N de-assert mode*

### 5.7.5 Block Diagram

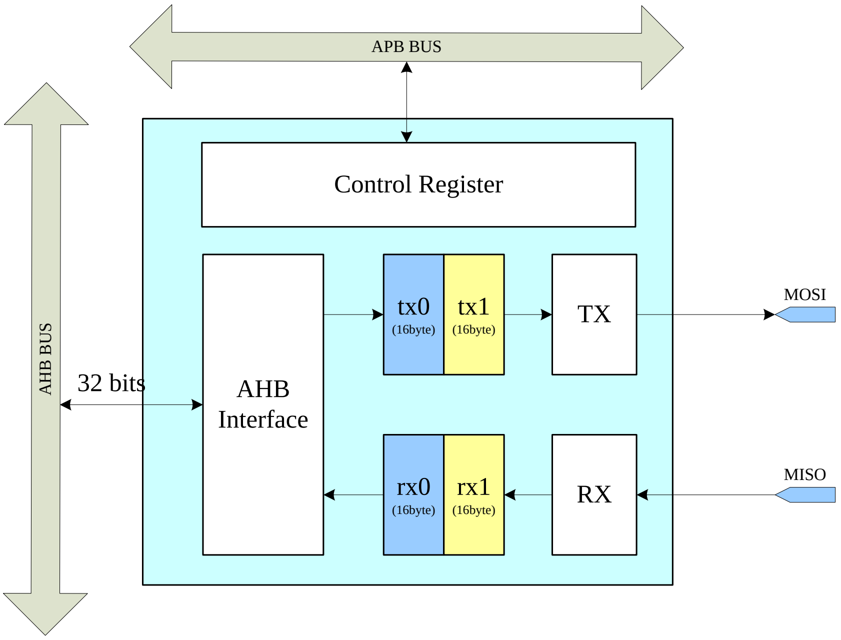


*Figure-5-16. Block diagram of SPI*

### 5.7.6 Register Definition

See chapater 3.7 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

### 5.7.7 Programming Guide

Follow the steps below to perform SPI transmission:

1. Prepare the data in the memory with its start address to be the “source address”.
2. Set up the timing and protocol for the SPI transmission (see Figure 5-13 for detailed setup parameters).
3. Fill the “destination address”, which is the start address that you would like to place the received data, and “source address”, which is the start address to place the data to be transmitted, into register  SPI_RX_DST and SPI_TX_SRC, respectively.
4. Write 1 the CMD_ACT to start the transfer
5. Get the data received from the buffer prepared starting from “destination address”.

# 5.8 MSDC Controller

## 5.8.1 Introduction

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

- SD memory card specification version 3.0
- MMC/eMMC 5.0

## 5.8.2 Features

Each MSDC contains:

- Interface with MCU by AHB bus and AXI bus
- 32-bit access on AHB bus and 64 axi bus master
- 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
- 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 e-MMC boot-up mode and SD boot-up
- Supports HS400 mode (DDR200) to emmc50
- 256 programmable serial clock rates on SD/MMC bus from 100kHz to 208MHz
- Card detection capabilities

### 5.8.3 Block Diagram

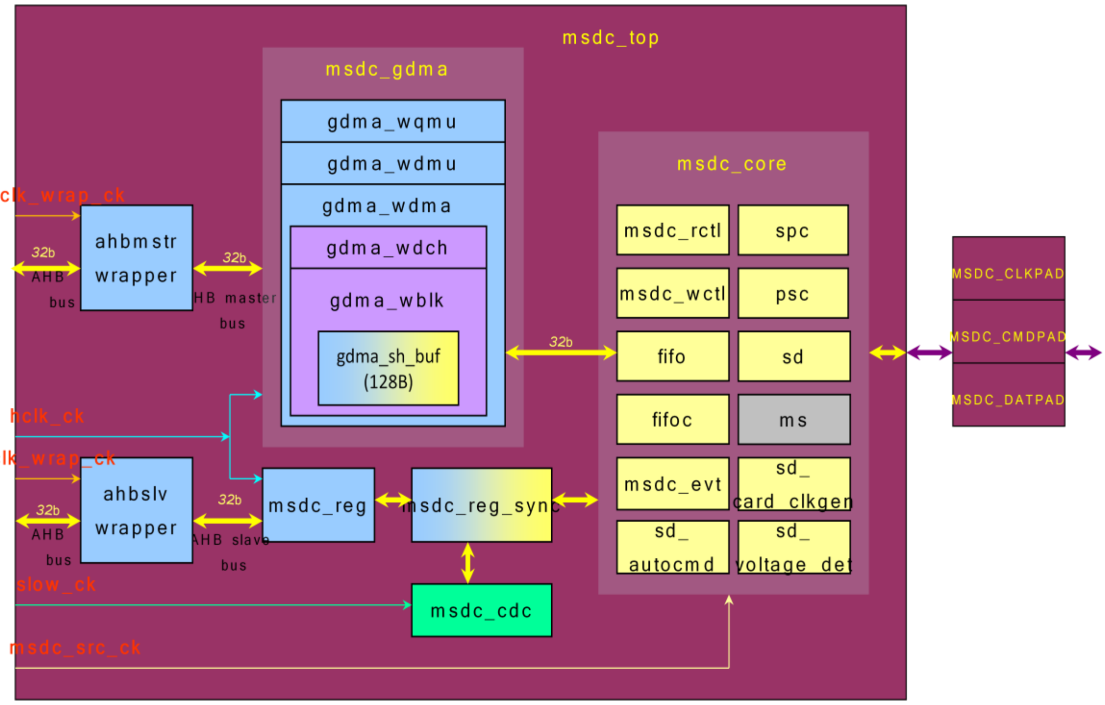


*Figure 5-17. Block diagram of MSDC controller*

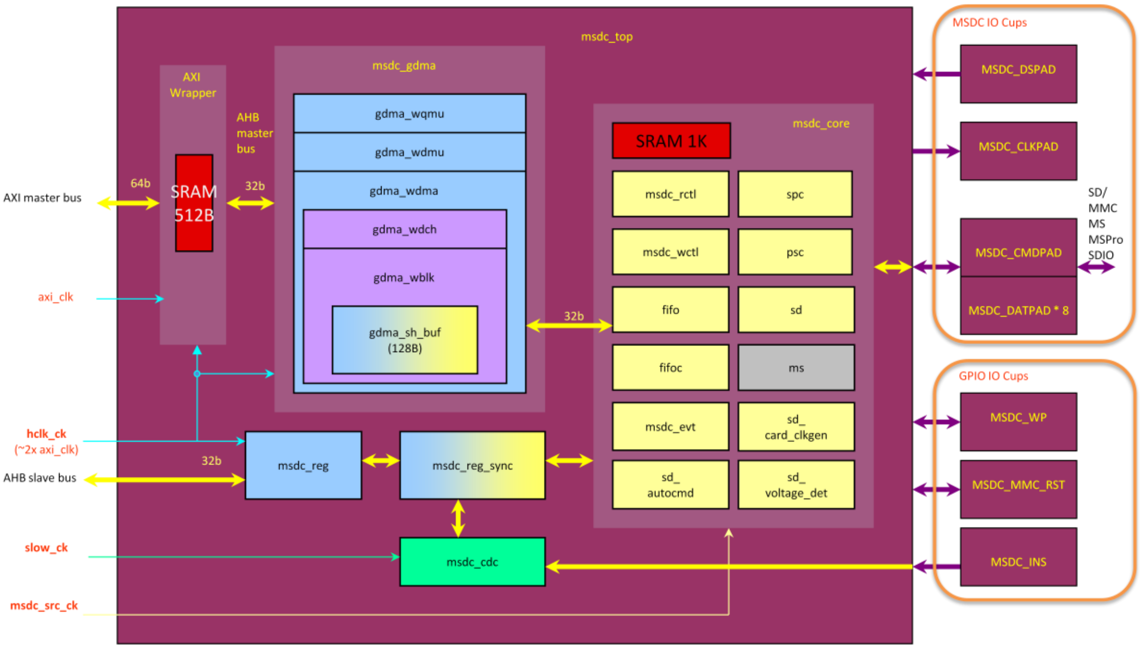


*Figure 5-18. Block diagram of MSDC controller for emmc50*

### 5.8.4 Register Definition

| MSDC number | Base address | Feature |
|---|---|---|
| MSDC0 | 0x11230000 | EMMC4.5/5.0 |
| MSDC1 | 0x11240000 | SD3.0/SDIO3.0 |

There are 2 MSDC IPs in this SOC. Use of the registers below are the same except that the base address needs to be changed to respective one.

See chapater 3.8 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 5.9 AUXADC

### 5.9.1 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.

### 5.9.2 Features

Table 5-2 describes the features in the AUXADC module.

***Table 5-2. AUXADC: feature list***

| Item | Main function | Description |
|---|---|---|
| 1 | Immediate analog-digital conversion | In immediate mode, it supports auto-set option.<br>In time-trigger mode, it supports auto-clear option. |
| 2 | Background detection and interrupt | The related command registers: AUXADC_DET_VOLT, AUXADC_PERIOD, AUXADC_DEBT, AUXADC_SEL |
| 3 | Temperature measurement | |

### 5.9.3 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.

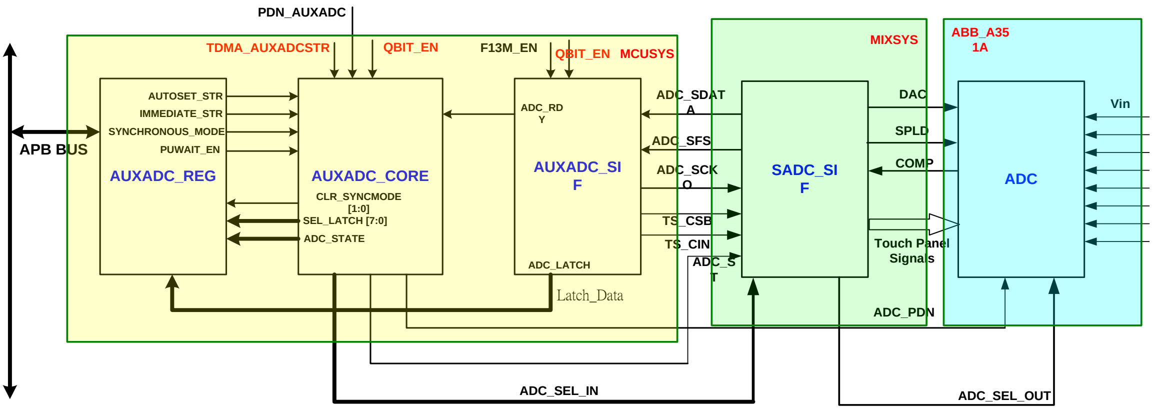


*Figure 5-19. Block diagram of AUXADC*

### 5.9.4 Theory of Operation

#### 5.9.4.1 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.

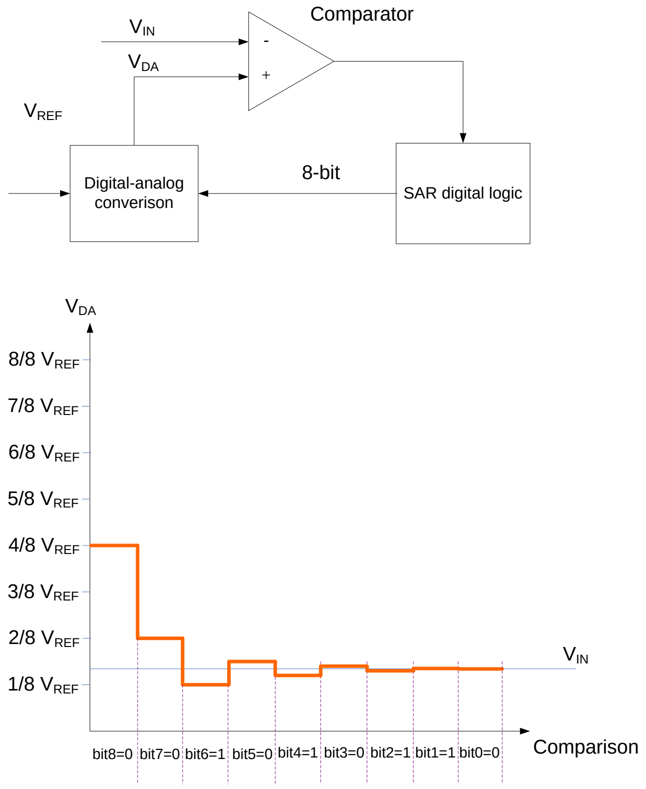


*Figure 5-20. SAR ADC architecture and conversion*

### 5.9.4.2 Design Partition

Table 5-3 shows the design partition.

*Table 5-3. 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 |

## 5.9.5 Register Definition

See chapater 3.9 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 5.10 I2C/SCCB Controller

### 5.10.1 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.

### 5.10.2 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

#### 5.10.2.1 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 8 bytes of data for a write transfer, or read up to 8 bytes of data for a read transfer.

#### 5.10.2.2 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.

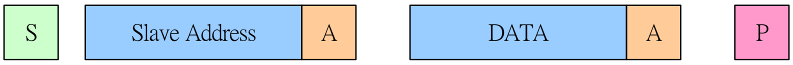
- Transaction length = Number of transfers to be conducted.

🟦 Master to slave dir

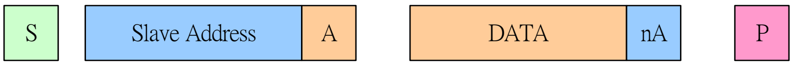
🟧 Slave to master dir

## Single byte access

Single Byte Write

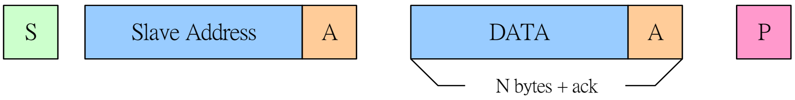


Single Byte Read

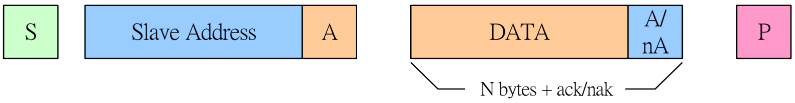


## Multi byte access

Multi Byte Write

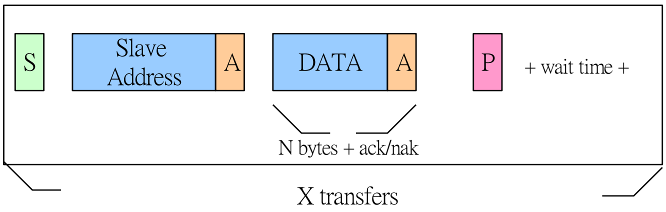


Multi Byte Read

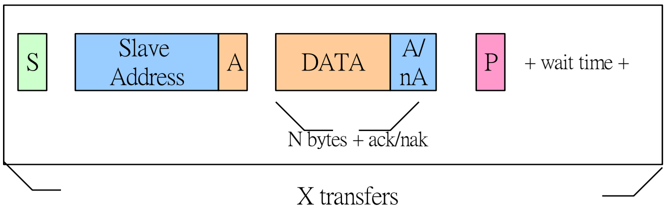


## Multi-byte transfer + multi-transfer (same direction)

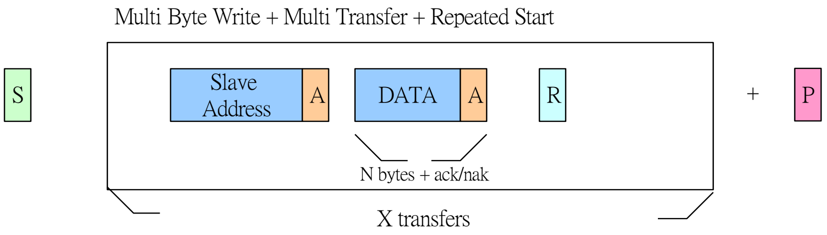
Multi Byte Write + Multi Transfer

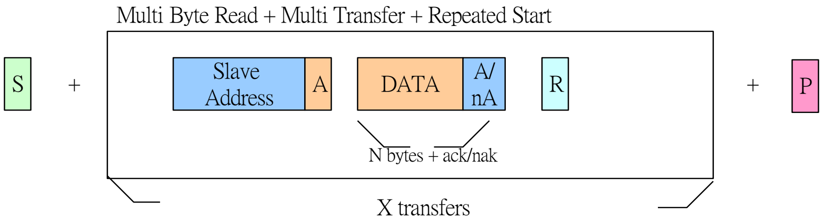


Multi Byte Read + Multi Transfer



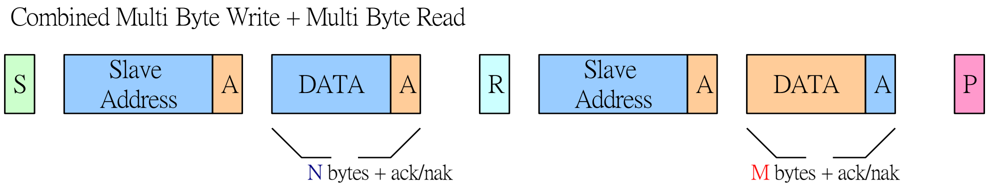
## Multi-byte transfer + multi-transfer w RS (same direction)



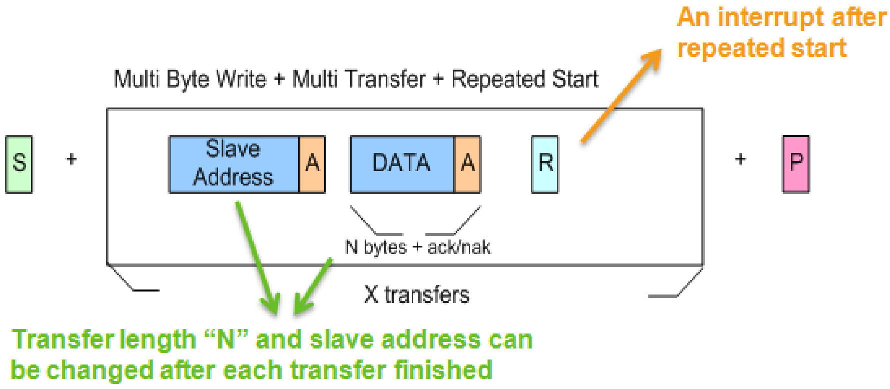


## Combined write/read with Repeated Start (direction change)

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

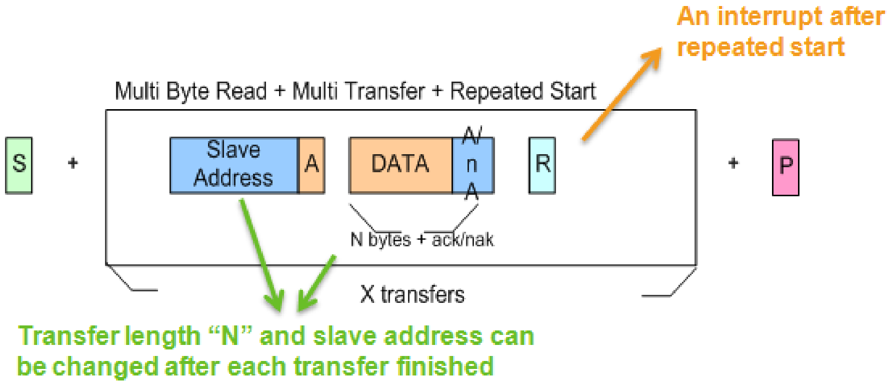


## Repeated start multiple transfer (write/read)



An interrupt after repeated start

Transfer length "N" and slave address can be changed after each transfer finished


An interrupt after repeated start
Multi Byte Read + Multi Transfer + Repeated Start
S
+
Slave Address
A
DATA
A/nA
R
+
P
N bytes + ack/nak
X transfers
Transfer length "N" and slave address can be changed after each transfer finished

### 5.10.3 Block Diagram

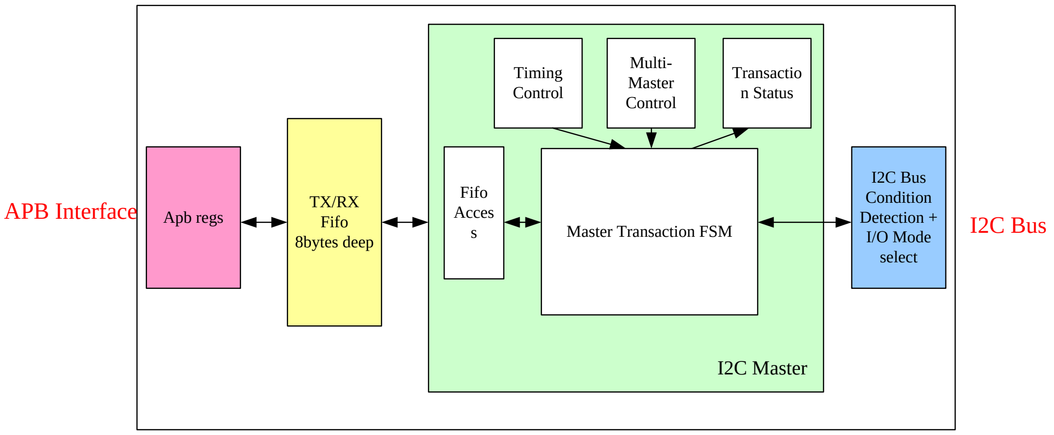


*Figure 5-21. Block diagram of I2C*

### 5.10.4 Register Definition

| I2C number | Base address | Feature |
|---|---|---|
| I2C0 | 0x11007000 | Supports DMA |
| I2C1 | 0x11008000 | Supports DMA |
| I2C2 | 0x11009000 | Supports DMA |
| I2C3 | 0x1100F000 | Supports DMA |

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

See chapater 3.10 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 5.11 Pulse-Width Modulation (PWM)

### 5.11.1 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 5-22).

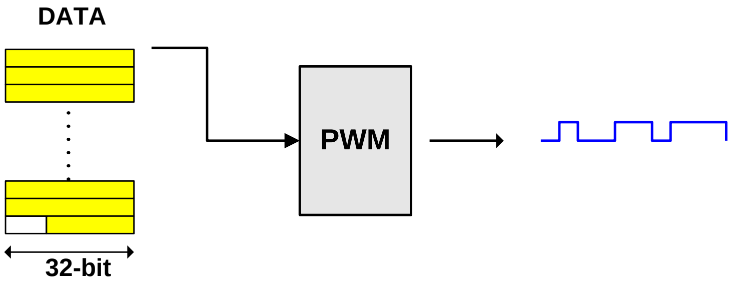
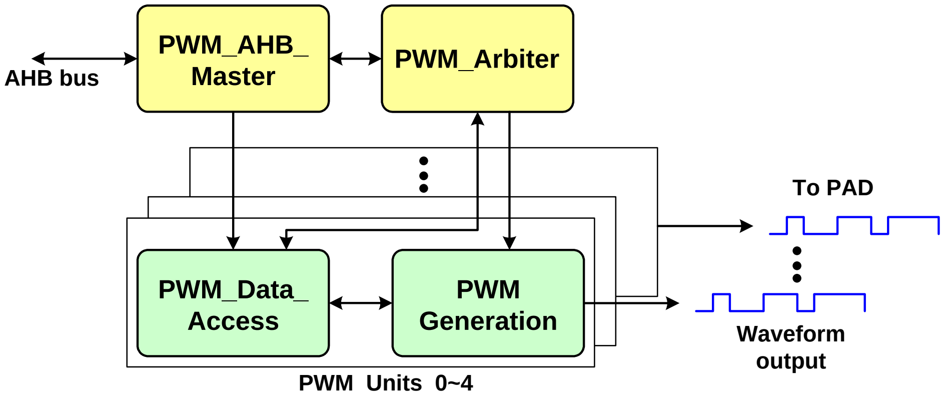


*Figure 5-22. Generation procedure of PWM*

### 5.11.2 Features

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

### 5.11.3 Block Diagram



### 5.11.4 Register Definition

See chapater 3.11 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 5.11.5 Clock Source Selection

| OLD_MODE | CLKSEL_OLD | CLKSEL | CLKSRC |
|---|---|---|---|
| **1** | 1 | 0 | bclk |
| **1** | 1 | 1 | 32KHz (used in old mode only) |
| **1** | 0 | 0 | bclk |
| **1** | 0 | 1 | bclk/1625 |
| **0** | Don't care | 0 | bclk |
| **0** | Don't care | 1 | bclk/1625 |

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

# 5.12 General-Purpose Timer (GPT)

## 5.12.1 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).

## 5.12.2 Features

The four operation modes for GPT are ONE-SHOT, REPEAT, KEEP-GO and FREERUN. See Table 5-4. ***Operation mode of GPT*** for the functions of each mode.

***Table 5-4. Operation mode of GPT***

| Mode | Auto Stop | Interrupt | Increases when EN=1 and ... | When COUNTn equals COMPAREn | Example: Compare is set to 2<br>_*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.

### 5.12.3 Block Diagram

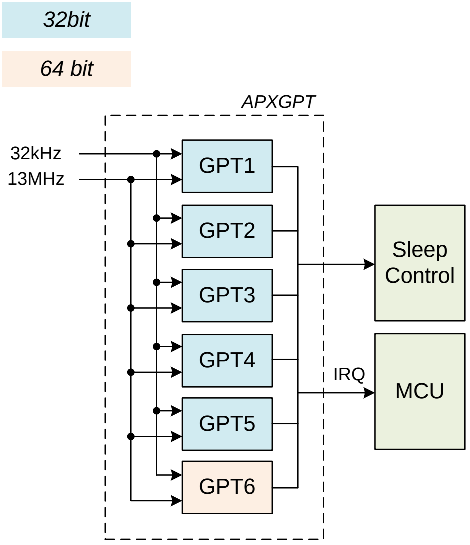


***Figure 5-23. Block diagram of APXGPT***

### 5.12.4 Register Definition

See chapater 3.12 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 5.12.5 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.

## 5.13 Thermal Controller

### 5.13.1 Introduction

On mobile platform, thermal management is very fundamental. The thermal management controls the platform computing performance to achieve the requirement and maintain the Raven within the temperature constraints. Operation under over-high temperature for a long time will have a risk of damage for Raven reliability.

In MT6737, it embeds several temperature sensors in possible hot spots on the die. The thermal controller module executes a periodic measurement for each hot spot. The temperature readings are readable for software.

In order to minimize the software effort of temperature monitoring, the thermal controller generates interrupts to microprocessors which informs the abnormal condition.

### 5.13.2 Features

- Supports up to three thermal sensors
- Periodic temperature measurement
- Temperature monitoring
- Different types of low pass filters for thermal sensor reading

### 5.13.3 Block Diagram

There are four microprocessors in MT6737. Their maximum frequency is up to over 1.3GHz and the transistor count is also large. The power consumed by microprocessors occupies a high percentage of the entire chip power consumption. Besides the microprocessors, EMI is also a source of power consumption because it provides high DRAM data bandwidth to other modules in MT6737.

The thermal controller is implemented for the software to operate MT6737 with a pre-defined temperature range. Or it will have function failure and reliability issues. According to temperature measurement, the system performance can be adjusted and the system design for power dissipation can also be monitored.

The hottest location in MT6737 may be different in different applications. Besides, when the thermal controller informs the software of an abnormal condition, the following power reduction action should be an efficient and low latency.

In general, there are two methods for embedded CMOS temperature sensor:

- VBE temperature sensor
- ▵ VBE temperature sensor (PTAT temperature sensor)

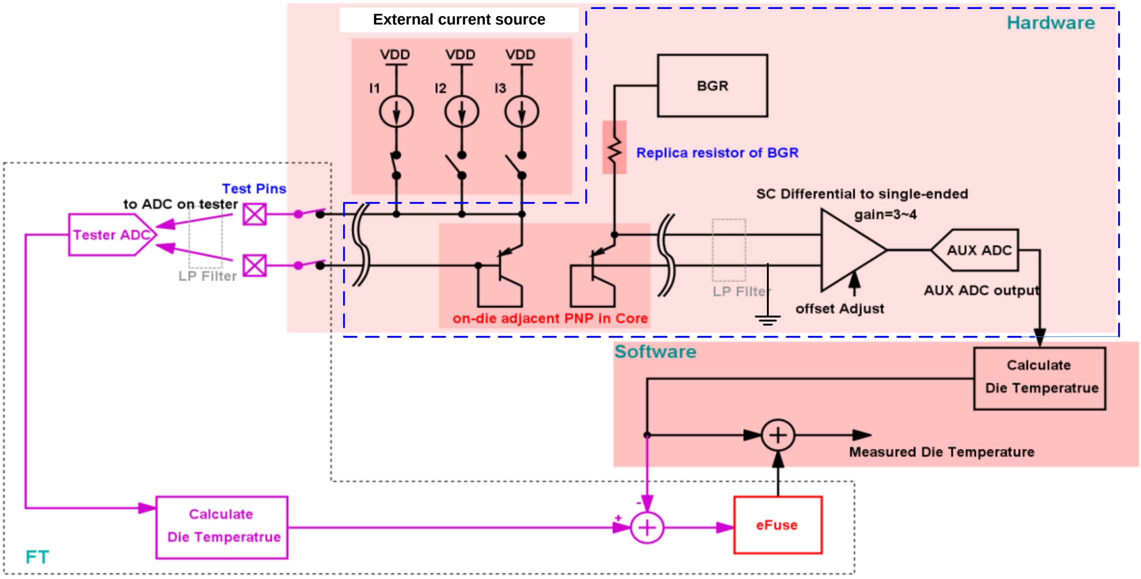


*Figure 5-24. Implementation of CMOS temperature sensor*

The accuracy target is 2°C when the temperature is in the range of 0 ~ 85°C. We use VBE as a CMOS sensor:

Consider

$$Ic = Is \cdot e^{\frac{V_{BE}}{\frac{nkT}{q}}}$$

$$\Rightarrow V_{BE} = \frac{nkT}{q} \cdot \ln \frac{I_C}{I_S}$$

- -1.665mV/°C in N40 process (TT, RTyp)

And use 12-bit SAR AUXADC for data conversion. The resolution of AUXADC is about 2.5V/$2^{12}$= 0.6mV. But for △ VBE method,

△ VBE = nkT/q x lnN

- k/q=0.086mV/°C

The resolution of AUXADC is not enough.

Considering offset, we calibrate the offset in FT and use e-fuse to store the calibrated value.

- △ VBE based approach for die temperature offset calibration.

There is no need to wait for thermal equilibrium in FT. However, it needs unused PAD and external bias circuit in FT.

1. Consider the BJT series resistor,

$$\Delta V_{BE} = \frac{\beta}{1+\beta} R_S (I_{C_1} - I_{C_2}) + \frac{nkT}{q} \cdot \ln \frac{I_{C_1}}{I_{C_2}} \frac{I_{S_2}}{I_{S_1}} = \frac{\beta}{1+\beta} R_S (I_{C_1} - I_{C_2}) + \frac{nkT}{q} \ln \frac{N}{M}$$

2. By applying IC1, IC2 and IC3, the three different current in FT, we can get the following equation and solve it for RS and T.

$$\Delta V_{BE1} = \frac{\beta}{1+\beta} R_S (I_{C_2} - I_{C_1}) + \frac{nkT}{q} \ln N_1$$

$$\Delta V_{BE2} = \frac{\beta}{1+\beta} R_S (I_{C_3} - I_{C_2}) + \frac{nkT}{q} \ln N_2$$

- Comparing the T measured via on chip ADC and the T got in the above equations, we can calibrate the temperature offset.

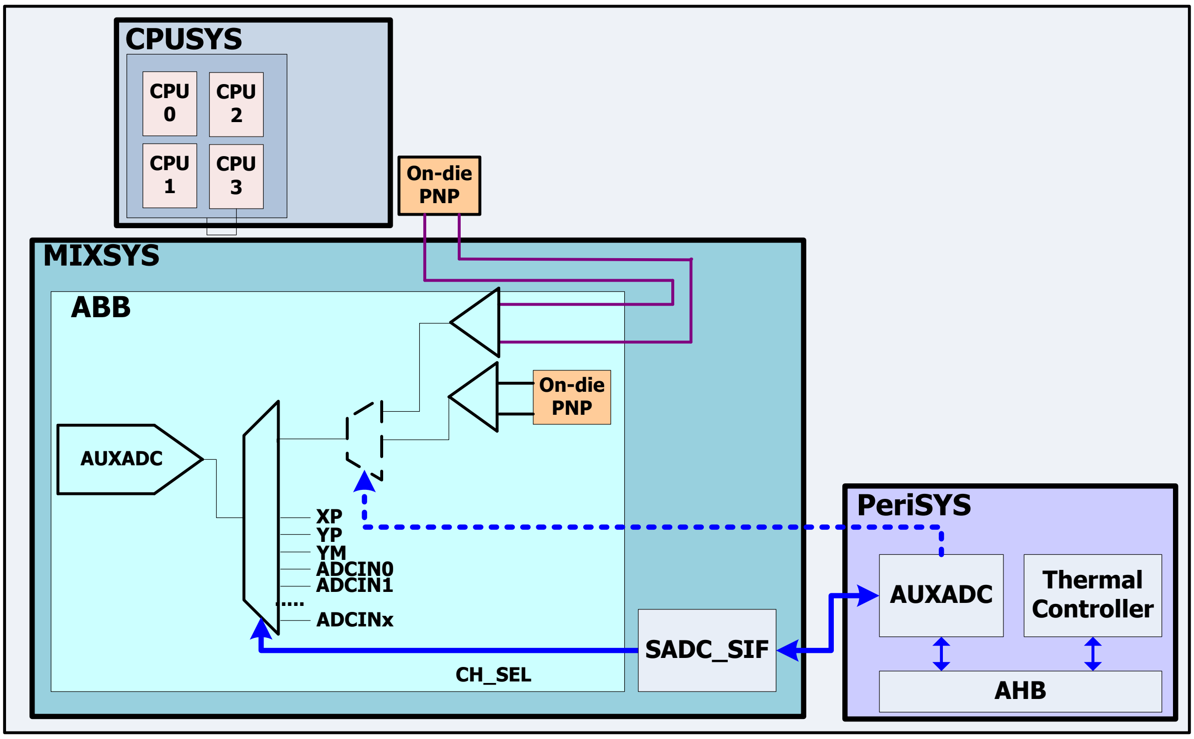


*Figure 5-25. Block diagram of system temperature measurement*

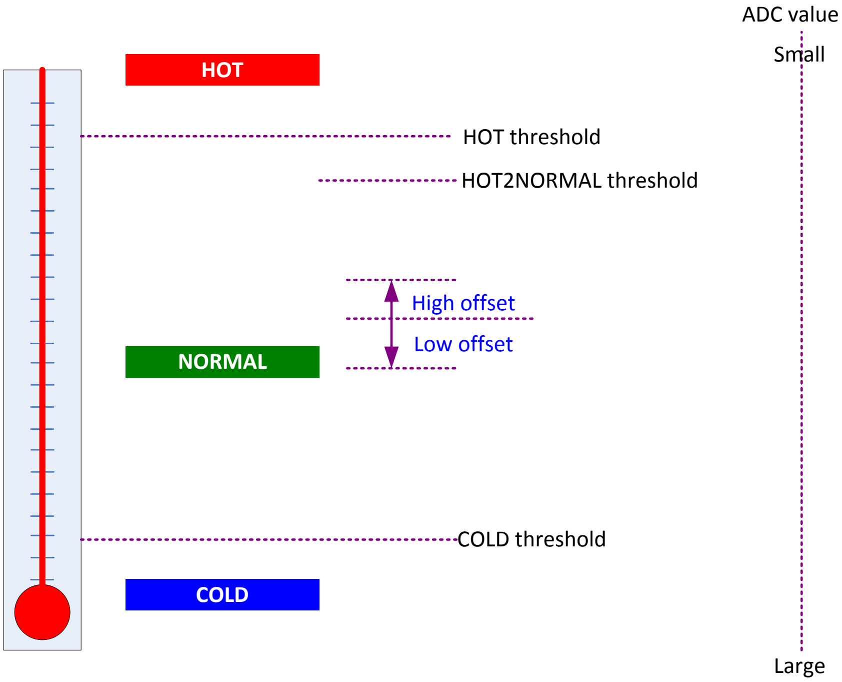
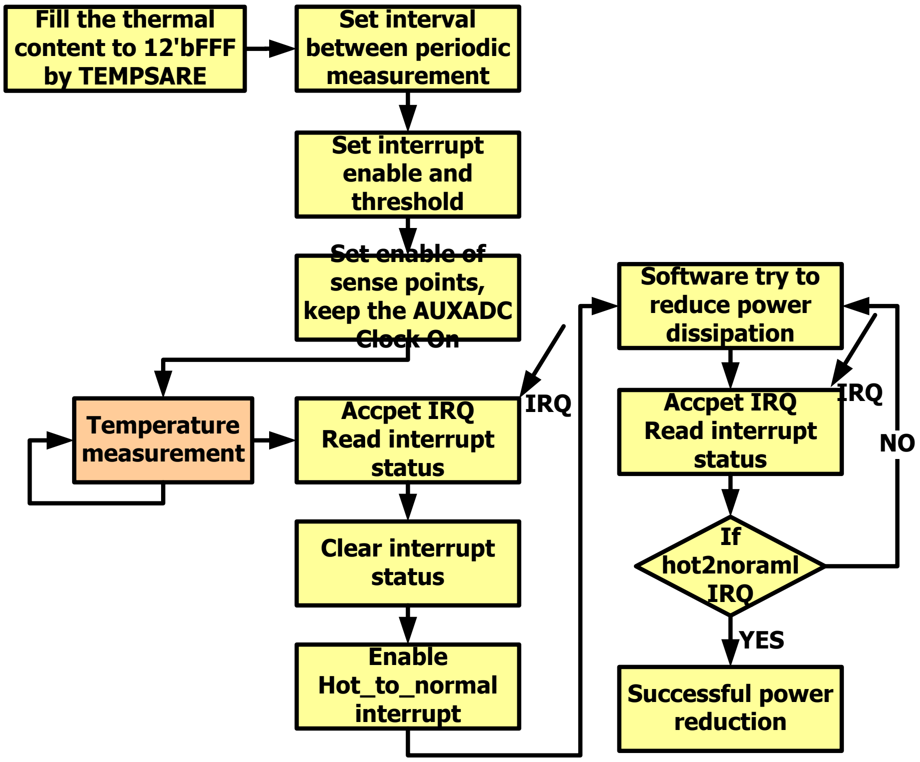


***Figure 5-26. Block diagram of system temperature measurement***

### 5.13.4 Register Definition

See chapater 3.13 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 5.13.5 Programming Guide


Fill the thermal content to 12'bFFF by TEMPSARE
Set interval between periodic measurement
Set interrupt enable and threshold
Set enable of sense points, keep the AUXADC Clock On
Temperature measurement
Accpet IRQ Read interrupt status
Clear interrupt status
Enable Hot_to_normal interrupt
IRQ
Software try to reduce power dissipation
Accpet IRQ Read interrupt status
IRQ
NO
If hot2noraml IRQ
YES
Successful power reduction

:

*Figure 5-27. Programming flow*

1. Fill the thermal content to 12'bFFF by accessing TEMPSPARE.

**vWriteREG(TEMPMONCTL1, 'h0);**
**vWriteREG(TEMPMONCTL2, 'h0);**
**vWriteREG(TEMPAHBPOLL, 'h0); // polling interval to check if temp sense is ready**
**vWriteREG(TEMPAHBTO, 'hFF); // Exceed this polling time, IRQ would be inserted**
**vWriteREG(TEMPSPARE0, 'h1FFF); // fill the TEMPSPARE0 register as 'h1FFF**
**vWriteREG(TEMPPNPMUXADDR, 32'hTS_CON1); // The adxadc mux address to select to Thermal channel, and please reference mixsys.doc**
**vWriteREG(TEMPADCENADDR, 32'TEMPSPARE1); // The adxadc enable address to trigger Thermal senser, please reference auxadc.doc**
**vWriteREG(TEMPADCVALIDADDR, 32'hTEMPSPARE1); // The adxadc status address to check if Thermal senser reading is valid, please reference auxadc.doc**
**vWriteREG(TEMPADCVOLTADDR, 32'hTEMPSPARE0); // The adxadc temperature address for the value read back from temp senser, please reference auxadc.doc**
**vWriteREG(TEMPRDCTRL, 'h0); // use TEMPSPARE0 as valid address**
**vWriteREG(TEMPADCVALIDMASK, 'h2c); // set adxadc valid polarity to 0**
**vWriteREG(TEMPMONCTL0, 'h0F); // enable all sense points include the debug one**
**Wait until the content of TEMPIMMD are filled by 'hFFF**

2. Set up interval between periodic temperature measurement, if the MODULE clock is 66MHz.

**vWriteREG(TEMPMONCTL1, 'h3FF); // counting unit is 1024*15.15ns=15.5 us**
**vWriteREG(TEMPMONCTL2, 'h3FF); // sensing interval is 1024*15.5us=15.87 ms**
**vWriteREG(TEMPAHBPOLL, 'h0F); // polling interval to check if temp sense is ready**
**vWriteREG(TEMPAHBTO, 'hFF); // Exceed this polling time, IRQ would be inserted**

```
vWriteREG(TEMPPNPMUXADDR, 32‘hTS_CON1); // The adxadc mux address to select to Thermal channel,
and please reference mixsys.doc
vWriteREG(TEMPADCENADDR, 32‘hAUXADC_CON1); // The adxadc enable address to trigger Thermal
senser, please reference auxadc.doc
vWriteREG(TEMPADCVALIDADDR, 32‘hAUXADC_CON3); // The adxadc status address to check if Thermal
senser reading is valid, please reference auxadc.doc
vWriteREG(TEMPADCVOLTADDR, 32‘hAUXADC_DAT11); // The adxadc temperature address for the value
read back from temp senser, please reference auxadc.doc
vWriteREG(TEMPRDCTRL, ‘h0); // use AUXADC_DAT11 as valid address
vWriteREG(TEMPADCVALIDMASK, ‘h2c); // set adxadc valid polarity to 0
```

3. Set up monitoring threshold and SPM wake up event.

```
vWriteREG(TEMPHTHRE, ‘hxxx); // set hot threshold
vWriteREG(TEMPCTHRE, ‘hxxx); // set cold threshold
vWriteREG(TEMPCTHRE, ‘hxxx); // set hot to normal threshold
vWriteREG(TEMPPROTCTL, ‘h20xxx); // set hot to wakeup event control
vWriteREG(TEMPPROTTC, ‘hxxx); // set hot to HOT wakeup event
vWriteREG(TEMPMONINT, ‘h8000001F); // enable interrupt
```

4. Enable sensing points.

```
vWriteREG(TEMPMONCTL0, ‘h07); // enable all three sense points
```

5. Accept IRQ

```
vReadREG(TEMPMONINTSTS); // read interrupt and clear interrupt status
```

6. Read temperature readings (optional)

```
vReadREG(TEMPMSR0); // read temperature reading of sense point 0
vReadREG(TEMPMSR1); // read tempe
rature reading of sense point 1
vReadREG(TEMPMSR2); // read temperature reading of sense point 1
```

7. Release pause of periodic temperature measurement

```
vWriteREG(TEMPMSRCTL1, vReadREG(TEMPMSRCTL1) & 0xFFFE);
```

**Immediate temperature measurement:**

After each immediate is done, the software must disable the immediate mode.

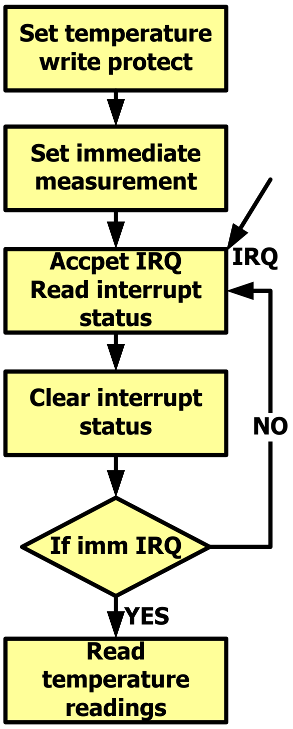

Set temperature write protect
Set immediate measurement
Accpet IRQ Read interrupt status
IRQ
Clear interrupt status
NO
If imm IRQ
YES
Read temperature readings

*Figure 5-28. Immediate measurement programming flow*

### 5.13.5.1 Interrupt Control Flow

The interrupt condition of high and low temperature monitoring is shown in Figure 5-29.

The software will accept interrupts while the following three conditions occur. The software determines which temperature sensor is to be monitored. Once the condition in any one of the three temperature sensors occurs, the interrupt will be issued.

The state machine is shown in Figure 5-30.

- Cold interrupt: When the temperature decreases to lower than the cold threshold form the normal temperature range, it means when the state of NORMAL transferred into the state of COLD.
- Hot interrupt: When the temperature increases to higher than the hot threshold from the temperature below the hot threshold.

The state of VERY_HOT indicates that temperature is higher than the hot threshold. The state of HOT1 indicates that the temperature is higher than the hot_to_normal threshold. NORAML state can not be transferred into VERY_HOT directly.

- Hot to normal temperature interrupt: When HOT2 state is transferred into the NORMAL state.

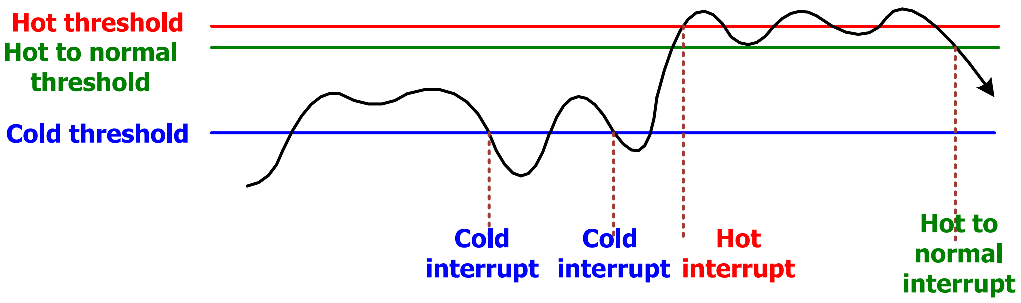


*Figure 5-29. Interrupt condition of high/low temperature monitoring*

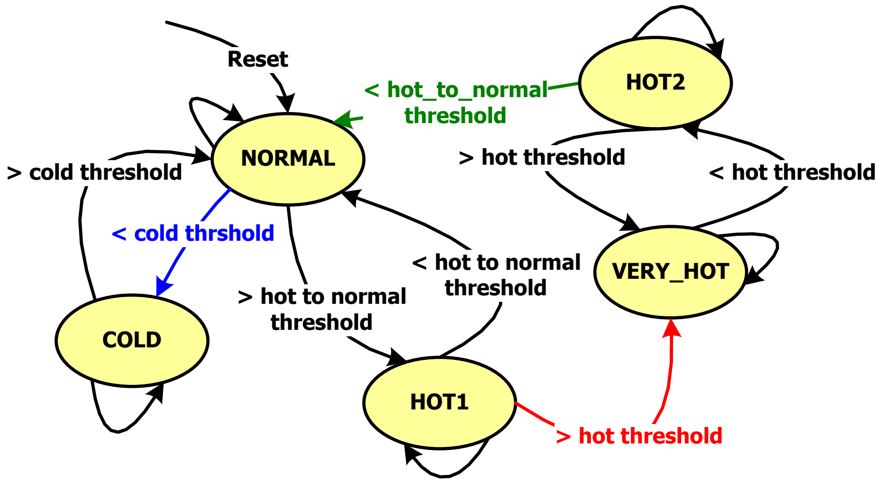


*Figure 5-30. Finite state machine of high/low temperature monitoring*

In Figure 5-30, when the software immediate measurement is enabled, the state will maintain the current state until the software disables the immediate mode. It is shown as the "*" mark.

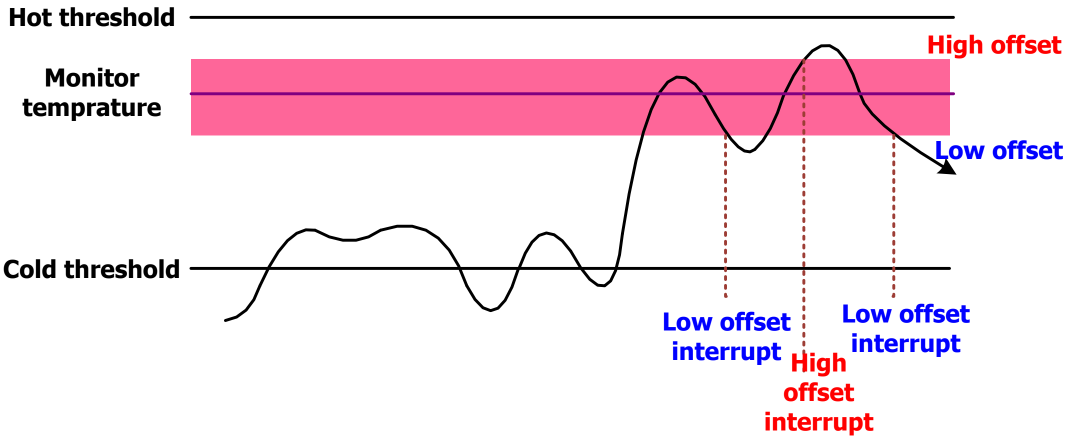


*Figure 5-31. Interrupt condition of high/low offset monitoring*

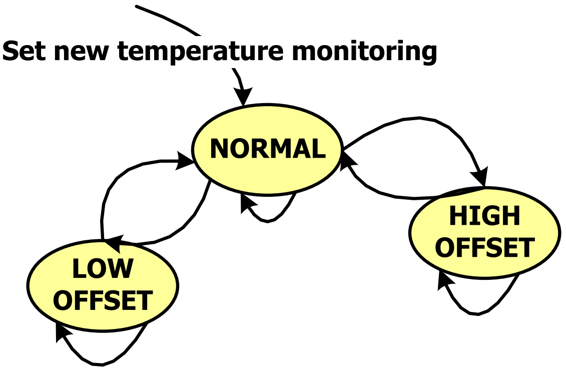


*Figure 5-32. Finite state machine of high/low offset monitoring*

## 5.14 IRTX

### 5.14.1 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.

### 5.14.2 NEC Protocol

#### 5.14.2.1 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 5-33. The recommended carrier duty-cycle is 1/4 or 1/3.

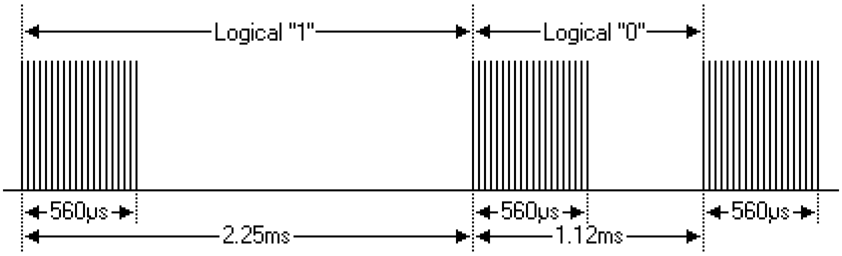


***Figure 5-33. Logic representation for NEC protocol***

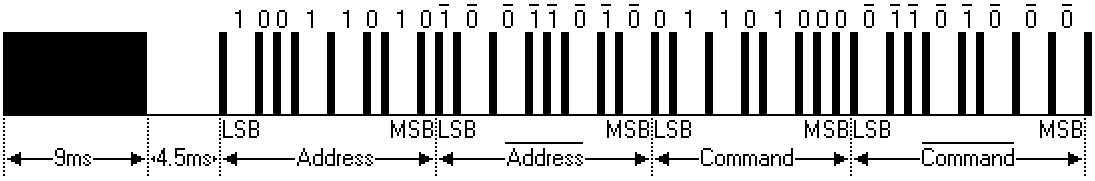


***Figure 5-34. Pulse train in transmission of NEC protocol***

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.

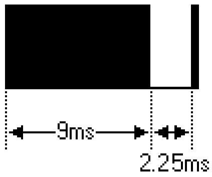


***Figure 5-35. A message is started by a 9ms AGC burst***

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.

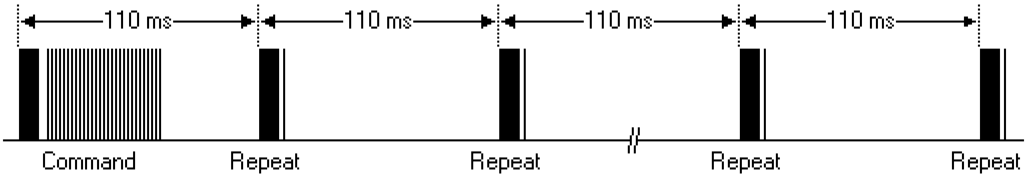


***Figure 5-36. A repeat code is transmitted every 110ms***

#### 5.14.2.2 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

### 5.14.3 Philips RC-5 Protocol

#### 5.14.3.1 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.

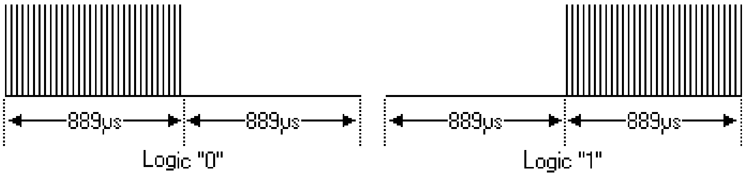


***Figure 5-37. Coding method for RC5 protocol***

### 5.14.3.2 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

## 5.14.4 Philips RC-6 Protocol

### 5.14.4.1 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.

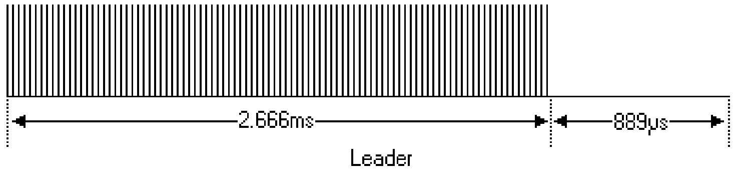


*Figure 5-38. Lead pulse in RC6 protocol*

- 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.

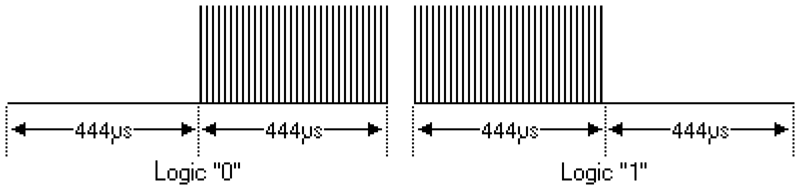


*Figure 5-39. Coding method for RC6 protocol*

- 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.

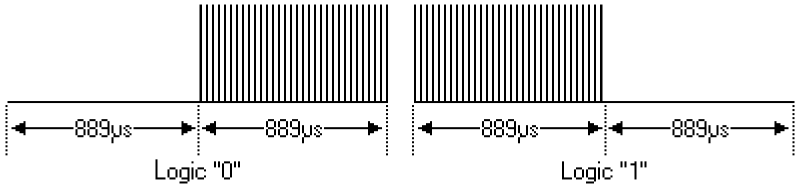


*Figure 5-40. Trailer pulse in RC6 protocol*

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

### 5.14.4.2 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

### 5.14.5 Register Definition

See chapater 3.14 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 5.15 IrDA

### 5.15.1 Introduction

IrDA framer, which is shown in Figure 5-41, is implemented to reduce the CPU loading for IrDA transmission. The IrDA framer functional block can be divided into two parts: (1) the transmitting part and (2) the receiving part. The transmitter part performs BOFs addition, byte stuffing, the addition of 16-bits FCS, and EOF appendence. The receiving part executes BOFs removal, ESC character removal, CRC checking, and EOF detection. In addition, the framer will perform 3/16 modulation and demodulation to connect to the IR transceiver. The transmitter and receiver need the DMA channel.

### 5.15.2 Features

- The IrDA framer supports IrDA SIR, MIR and FIR modes of operation.
- SIR mode includes operation from 9600bps ~ 115200bps.
- MIR includes operation at 567000bps or 1152000bps
- FIR mode includes operation at 4Mbps.

### 5.15.3 Block Diagram

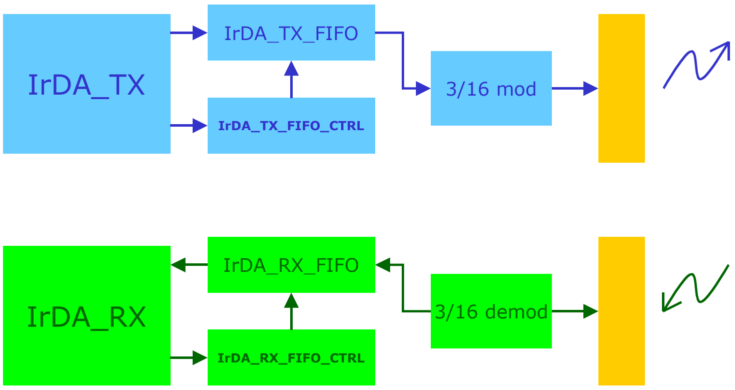


***Figure 5-41. Block diagram of IrDA***

### 5.15.4 Register Definition

See chapater 3.15 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 5.16 Audio System

### 5.16.1 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
- PCM/I2S merged interface for MTK connectivity IC x 1

### 5.16.2 Features

- Audio playing
  - Supports 8, 11.025, 12, 16, 22.05, 24, 32, 44.1, and 48kHz sampling rate output
  - Supports playing stereo data
- Audio recording
  - Supports 8, 16, 32, 48kHz sampling rate recording
  - 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
- PCM/I2S merged interface
  - 4-pin interface for concurrently supporting I2S and PCM
  - PCM supports 8k/16k Hz sampling rate
  - I2S supports 32, 44.1, and 48 kHz sampling rate
- 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

### 5.16.3 Block Diagram

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

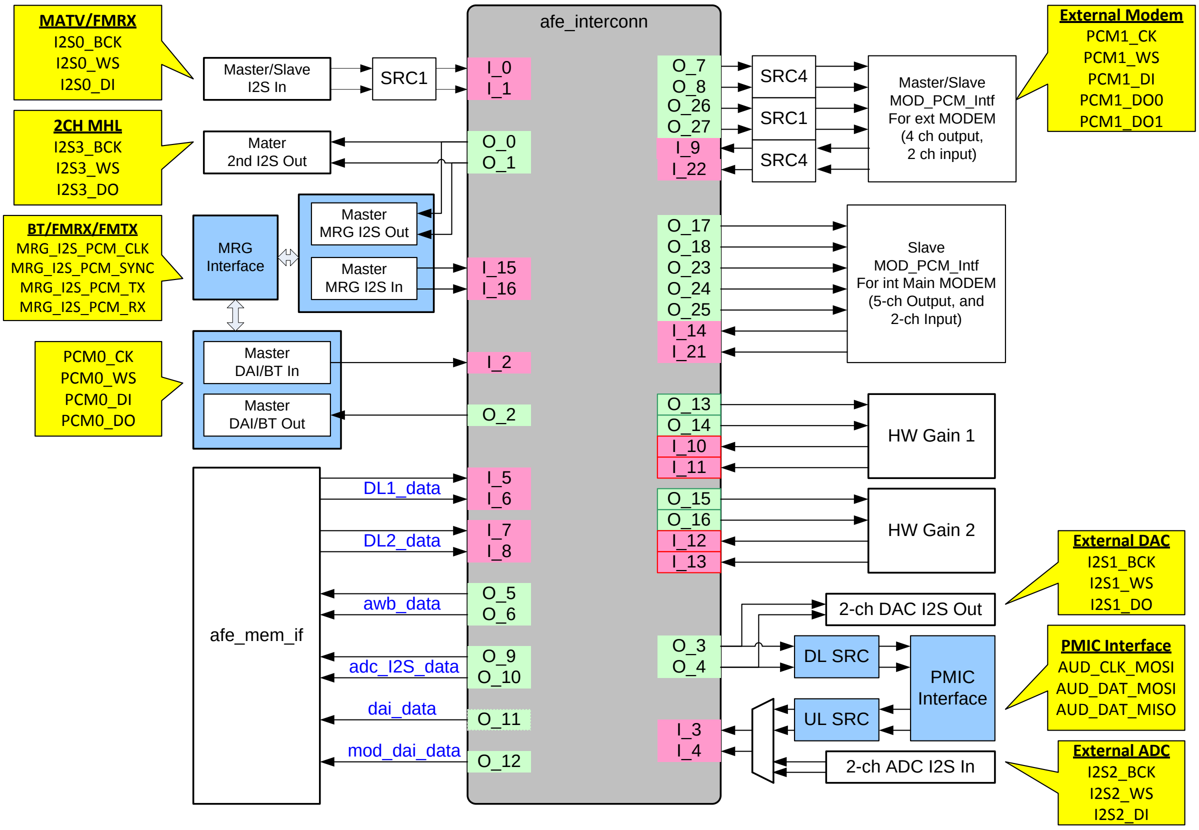


*Figure 5-42. Block diagram of audio sys*

*Table 5-5. Interface of audio sys*

| | Output | O_00 | O_01 | O_02 | O_03 | O_04 | O_05 | O_06 | O_07 | O_08 | O_09 | O_10 | O_11 | O_12 | O_13 | O_14 | O_15 | O_16 | O_17 | O_18 | ~~O_19~~ | ~~O_20~~ | O_21 | O_22 | O_23 | O_24 | O_25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Input | | I2S_2_LO | I2S_2_RO | DAI_O | DAC_I2S_LO | DAC_I2S_RO | MEM_AWB_LO | MEM_AWB_RO | MOD_DAI_LO | MOD_DAI_RO | MEM_ADC_LO | MEM_ADC_RO | MEM_DAI_O | MEM_MOD_DAI_O | GAIN1_LO | GAIN1_RO | GAIN2_LO | GAIN2_RO | SPCM_LO | SPCM_RO | ~~I2S3_LO~~ | ~~I2S3_RO~~ | MEM_ADC2_LO | MEM_ADC2_RO | SPCM2_LO | SPCM2_RO | SPCM3_LO |
| I_00 | I2S_I_L | RS | RS | RS | RS | RS | RS | x | S | S | S | S | x | x | S | S | RS | RS | S | S | ~~RS~~ | ~~RS~~ | S | x | S | x | S |
| I_01 | I2S_I_R | RS | RS | RS | RS | RS | x | RS | S | S | S | S | x | x | S | S | RS | RS | S | S | ~~RS~~ | ~~RS~~ | x | S | x | S | x |
| I_02 | DAI_I | S | S | S | S | S | S | x | S | S | x | x | S | x | S | S | S | S | S | S | ~~x~~ | ~~x~~ | x | x | x | x | x |
| I_03 | ADC_I2S_I_L | S | S | RS | S | S | S | x | S | x | S | x | x | x | S | S | S | S | S | x | ~~S~~ | ~~S~~ | S | x | S | x | S |
| I_04 | ADC_I2S_I_R | S | S | RS | S | S | x | S | x | S | x | S | x | x | S | S | S | S | x | S | ~~S~~ | ~~S~~ | x | S | x | S | x |
| I_05 | MEM_DL1_LI | RS | RS | RS | RS | RS | S | x | S | x | S | x | S | x | S | S | S | S | S | x | ~~RS~~ | ~~RS~~ | S | x | x | x | x |
| I_06 | MEM_DL1_RI | RS | RS | RS | RS | RS | x | S | x | S | x | S | x | S | S | S | S | S | x | S | ~~RS~~ | ~~RS~~ | x | S | x | x | x |
| I_07 | MEM_DL2_LI | RS | RS | RS | RS | RS | S | x | S | x | S | x | S | x | S | S | S | S | S | x | ~~RS~~ | ~~RS~~ | S | x | x | x | x |
| I_08 | MEM_DL2_RI | RS | RS | RS | RS | RS | x | S | x | S | x | S | x | S | S | S | S | S | x | S | ~~RS~~ | ~~RS~~ | x | S | x | x | x |
| I_09 | MOD_DAI_I | S | S | S | S | S | S | x | x | x | S | x | x | S | S | S | S | S | S | S | ~~S~~ | ~~S~~ | x | x | S | S | S |
| I_10 | GAIN1_LI | RS | x | RS | RS | x | S | x | S | x | S | S | S | S | x | x | x | x | S | x | ~~S~~ | ~~x~~ | S | S | S | x | S |
| I_11 | GAIN1_RI | x | RS | RS | x | RS | x | S | x | S | S | S | S | S | x | x | x | x | x | S | ~~x~~ | ~~S~~ | S | S | x | S | x |
| I_12 | GAIN2_LI | RS | x | RS | RS | x | S | x | S | x | S | S | S | S | x | x | x | x | S | x | ~~S~~ | ~~x~~ | S | S | x | x | x |
| I_13 | GAIN2_RI | x | RS | RS | x | RS | x | S | x | S | S | S | S | S | x | x | x | x | x | S | ~~x~~ | ~~S~~ | S | S | x | x | x |
| I_14 | SPCM_I | S | S | S | S | S | S | x | S | S | S | x | x | S | S | S | S | S | x | x | ~~S~~ | ~~S~~ | x | x | x | x | x |
| I_15 | MRG_I2S_LI | RS | RS | RS | RS | RS | RS | x | x | x | S | x | x | x | S | S | RS | S | x | x | ~~x~~ | ~~x~~ | S | x | x | x | x |
| I_16 | MRG_I2S_RI | RS | RS | RS | RS | RS | x | RS | x | x | x | S | x | x | S | S | S | RS | x | x | ~~x~~ | ~~x~~ | x | S | x | x | x |
| I_17 | ADC2_I2S_I_L | S | x | RS | S | x | S | x | S | x | S | x | S | x | S | S | S | S | S | x | ~~S~~ | ~~S~~ | S | x | S | x | S |
| I_18 | ADC2_I2S_I_R | x | S | RS | x | S | x | S | x | S | x | S | x | S | S | S | S | S | x | S | ~~S~~ | ~~S~~ | x | S | x | S | x |
| ~~I_19~~ | ~~MEM_DL1_2_LI~~ | ~~RS~~ | ~~x~~ | ~~RS~~ | ~~RS~~ | ~~RS~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~S~~ | ~~S~~ | ~~S~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~S~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~x~~ | ~~S~~ |
| ~~I_20~~ | ~~MEM_DL1_2_RI~~ | ~~x~~ | ~~RS~~ | ~~RS~~ | ~~RS~~ | ~~RS~~ | ~~x~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~S~~ | ~~S~~ | ~~S~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~S~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~x~~ | ~~S~~ | ~~x~~ |
| I_21 | SPCM2_I | S | S | S | S | S | x | S | S | S | x | S | x | S | S | S | S | S | x | x | ~~S~~ | ~~S~~ | x | x | x | x | x |

### 5.16.4 Register Definition

See chapater 3.16 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

# 6 Multimedia Subsystem

## 6.1 Multimedia Subsystem Configuration

### 6.1.1 Introduction

The multimedia subsystem contains multimedia controller, multimedia data path v2.0 (MDP 2.0) and display (DISP). The multimedia controller includes bus fabric control, Smart Memory Interface (SMI) control, memory access second level arbiter, and multimedia configuration. It plays the key role to handle different handshaking between infra subsystem, video subsystem, image subsystem and G3D subsystem. MDP 2.0 is the time sharing pipeline data flow controller to process resizing and rotation by memory access. The display pipeline outputs pixels to display interface with overlay, color enhancement, adaptive ambient light processing, color correction and gamma correction.

### 6.1.2 Features

The multimedia subsystem has the following features:

- APB bus fabric control center
- Smart Multimedia Interface (SMI) controller
- Second level arbiter for memory access request
- Multimedia Data Path v2.0. It has one read DMA, two resizers, one 2D-sharpness enhancement, one write DMA and one write rotator
- Two display pipe lines. One of the pipelines has its own read DMA, overlay, color engine, adaptive ambient light processing, color correction, gamma correction and display interface controller as basic components. The other one only includes read DMA 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 concurrent dual display output
- Display output interface: 1 DSI and 1 DPI.

### 6.1.3 Block Diagram

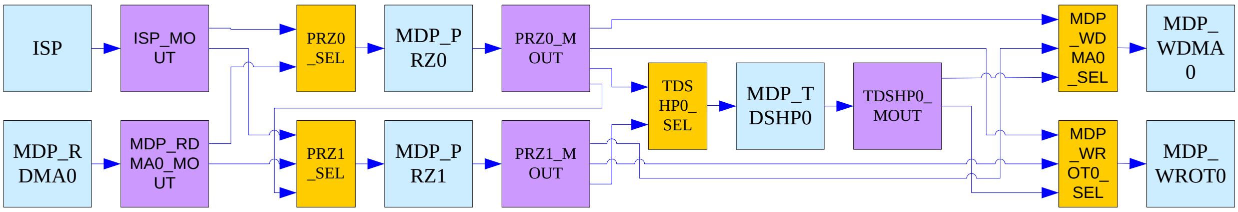


*Figure 6-1. MDP function blocks of multimedia partition*

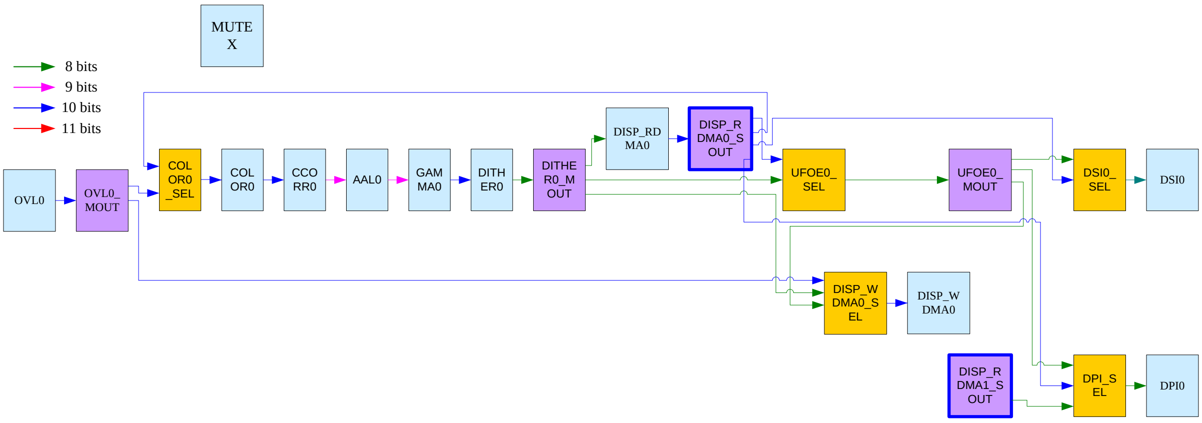


*Figure 6-2. DISP function blocks of multimedia partition*

### 6.1.4 Register Definition

See chapater 4.1 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 6.1.5 Programming Guide

Before we introduce the programming model, two basic operation modes and one terminology "STREAM" must be explained first to make the programming model reasonable. The basic operation modes are

1. Single mode:
   In this mode, one SW trigger only makes modules to process one frame. Usually, it is a memory-in-memory-output operation (e.g. memory -> RDMA0 -> PRZ0 -> WDMA0 -> memory), or an operation output to the display device with frame buffer (e.g. memory -> RDMA0-> color0-> gamma -> DSI0. SW must wait for one frame processing to be done before triggering the next frame.

2. Refresh mode:

In this mode, pixels must be outputted to the display device without frame buffer (e.g. DSI video mode and DPI). After a data path is configured as refresh mode and starts, it will process frame-by-frame automatically until the data path stops. The process follows the refresh timing (vsync, hsync and data enable) of the display output interface.

In the refresh mode, the display output interface refresh timing is asynchronous to software, and we must guarantee each module receives complete settings for one frame when it starts to process this frame. For this reason, a mutex between software and hardware is used to achieve this.

A mutex is used to specify a STREAM and to guarantee a complete setting of this STREAM is seen by modules in this STREAM.

- A STREAM means a data stream from Source to Sink.
- Source can be any module with read memory capability, which is RDMA or OVL.
- Sink can be any module with capability of outputting to external display devices or to memory, which is DSI0, DPI or WDMA/WROT.

In multimedia subsys, there are at most 8 STREAMs concurrently. Therefore, we provide 10 mutexes. Every mutex's function and register interface are the same.

Mutex has the following attributes, which can be set in dispsys_mutex.

- Source of SOF (Start Of Frame)
  It means the Sink of this STREAM is which module. Due to the display timing of a STREAM, when to start a frame/when to end a frame, is decided by the Sink, and the display subsys must know it to route the SOF signal to all modules in this mutex.
- Which modules are in the STREAM.

Because one mutex describes one STREAM, there is a constraint that one module can be set in only one mutex. In other words, one module cannot be set in more than one mutex.

## 6.2 SMI_COMMON

### 6.2.1 Introduction

SMI (Smart Multimedia Interface) is a simple bus protocol for multimedia engines to access system memories, including off-chip and on-chip memories. As in the block diagram, multimedia engines access EMI and MM memory through the SMI bus. The MM memory is part of SMI controller and designed to have short accessing latency and high bandwidth for multimedia engines. From the perspective of SMI, the multimedia engines are masters of the bus, and SMI itself is the slave of the bus. Furthermore, the MM memory is regarded as internal memory and EMI which is a type of external memory.

In this document, we focus on SMI_COMMON.

### 6.2.2 Features

- Auto clock gating for power reduction
- Arbitration among request from local arbiters to EMI
- Bandwidth/outstanding limiter
- Performance monitor
- Command throttling for reducing latency

### 6.2.3 Block Diagram

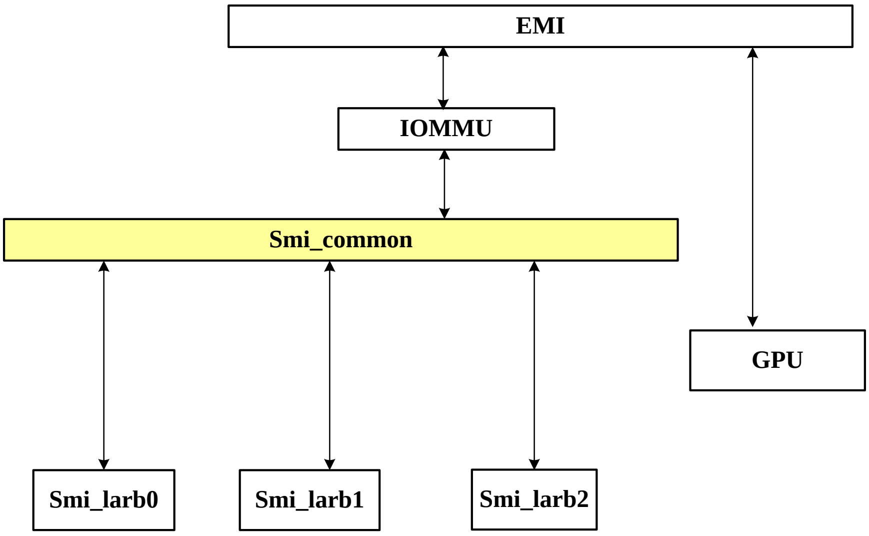


*Figure 6-3. SMI_COMMON and neighbor blocks*

### 6.2.4 Register Definition

See chapater 4.2 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 6.3 SMI_LARB

### 6.3.1 Introduction

The SMI (Smart Multimedia Interface) is a MediaTek proprietary interface used in multimedia systems. The SMI bus fabric deals with the complex bus interconnection and memory access transaction in a high-performance multimedia-rich system. The SMI bus fabric is separated into 2 parts for hierarchical arbitration. The SMI local arbiter is used for the first level arbitration for part of multimedia engines. The granted masters from SMI local arbiters are arbitrated at the second level arbiter at SMI common arbiter with other masters in other multimedia sub-systems.

### 6.3.2 Features

- 1st level arbitration of multimedia engines
- GMC/SMI protocol to AXI protocol conversion
- Supports bandwidth regulation for each master
- Performance monitor enables performance index measurement

### 6.3.3 Block Diagram

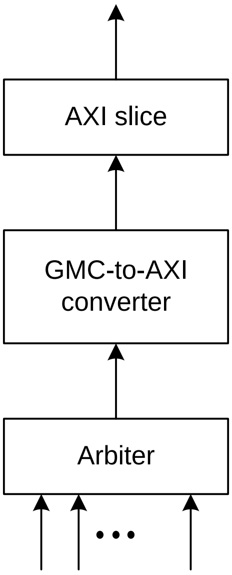


*Figure 6-4. SMI local arbiter block diagram*

### 6.3.4 Register Definition

The base addresses of local arbiters are listed below.

*Table 6-1. Base addresses of SMI local arbiters*

| Module | Base address |
|---|---|
| SMI_LARB0 | 0x1401_6000 |
| SMI_LARB1 | 0x1601_0000 |
| SMI_LARB2 | 0x1500_1000 |

The following register table is for SMI_LARB0. Replace the base address 0x14016000 with the base address of other SMI local arbiters when you are to program other SMI local arbiters.

See chapater 4.3 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 6.4 CAM

### 6.4.1 Introduction

Camera receives RAW and SOC sensor image data after processing of images and outputs YUV data to DRAM.

### 6.4.2 Features

The camera incorporates a feature-rich image signal processor to connect with a variety of image sensor components. This processor consists of timing generated unit (TG), lens/sensor compensation unit and image process unit

- Interface
  - Main cam: MIPI 4 lane/parallel interface
  - Sub cam: MIPI 2 lane/parallel interface
- Image capture resolution: Up to 13M
- Video recording resolution: Up to 720P
- Raw dump frame rate is 8M@30fps or 13M@15fps
- Supports video snapshot (up to 8M sensor), which enables user capture full size image while recording video.
- Image processing
  - Bad pixel compensation
  - Lens shading compensation
  - Demosaic
  - Color clipping
  - Gamma correction
  - Edge enhancement
  - Noise reduction with large kernel
  - Preference color adaptation
- 3A statistics and correction
- Flicker detection
- Electronic image stabilization for video
- High quality resizers

### 6.4.3 Register Definition

See chapater 4.4 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 6.5 SENINF_TOP (Sensor Interface)

### 6.5.1 Introduction

The seninf_top module transfers sensor signal into image pixels and pass them to ISP.

### 6.5.2 Features

- MIPI interface
  - Two 4-lanes MIPI interfaces
  - Virtual channel/Data type data interleaving
- Serial interface
  - 1 serial interface
- Parallel interface
  - 1 parallel interface
- Provides 3 sensor master clocks

### 6.5.3 Register Definition

See chapater 4.5 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 6.6 MDP_RDMA

### 6.6.1 Introduction

MDP RDMA is used to read images of multiple source format in memory and then output in scan-line sequence to the following engine. It supports several functions, such as

- Input image clipping, as illustrated in Figure 6-5.
- Different input formats, as listed in Table 6-2.
- Color conversion for RGB to YUV.
- Chroma upsample for cosited or non-consited YUV422/420 source

To support larger image size with cost burden, tile mode scheme is applied in MDP. MDP_RDMA is fully tile mode ready and can fulfill the single or tile mode operation. It also has a 3x3 color conversion inside. When the input is in RGB domain and output is in YCbCr domain, the 3x3 color conversion should be set well to get the correct data.

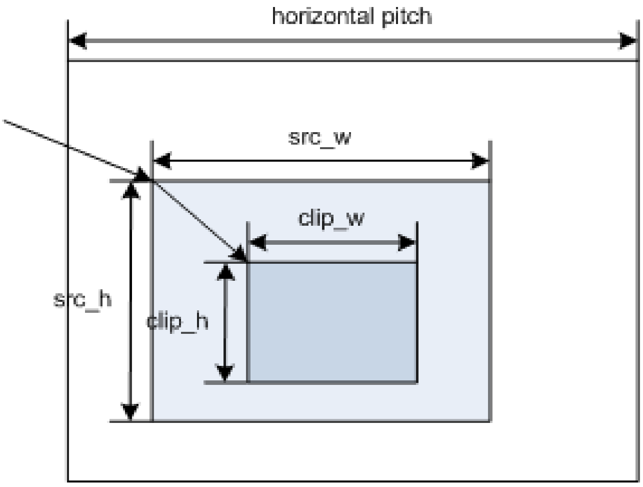


*Figure 6-5. Support clipping from source frame buffer*

*Table 6-2. Input format list*

| Input domain | Supported format |
|---|---|
| YCbCr | YCbCr_420_P_SW<br>YCbCr_420_SP_SW<br>YCbCr_420_SP_HW_BLOCK<br>YCbCr_420_SP_HW_BLOCK_INTERLACE<br>YCbCr_422_P_SW<br>YCbCr_422_SP_SW<br>YCbCr_422_I_SW<br>YCbCr_422_I_HW_BLOCK<br>Y Only |
| RGB | RGB565 |

| Input domain | Supported format |
|---|---|
| | RGB(BGR)888<br>ARGB(RGBA)8888<br>XRGB(RGBX)8888 |

### 6.6.2 Features

- Supports multiple format of input images
- Supports up to HD resolution 1280x800@60 fps without tile mode
- Supports YUV420/422 scanline 1/2/3 planes and RGB 16/24/32 bits
- Supports arbitrary byte swap for YUV or RGB source
- Supports cropping/clipping.
- Supports chroma up-sample filter to YUV444
- Supports default optimized or programmable RGB2YUV
- Tile mode ready; supports source width up to 131072 pixels
- Direct link to RSZ with YUV 444

### 6.6.3 Block Diagram

The internal pipeline of MDP_RDMA is shown in the following diagram, the input is from external memory via SMI read port, and output is direct linked to the next engine depending on the dispsys data-path setting, e.g. RSZ. There is also APB interface for S/W control, following the mutex protocol for advanced S/W control from disp_mutex.

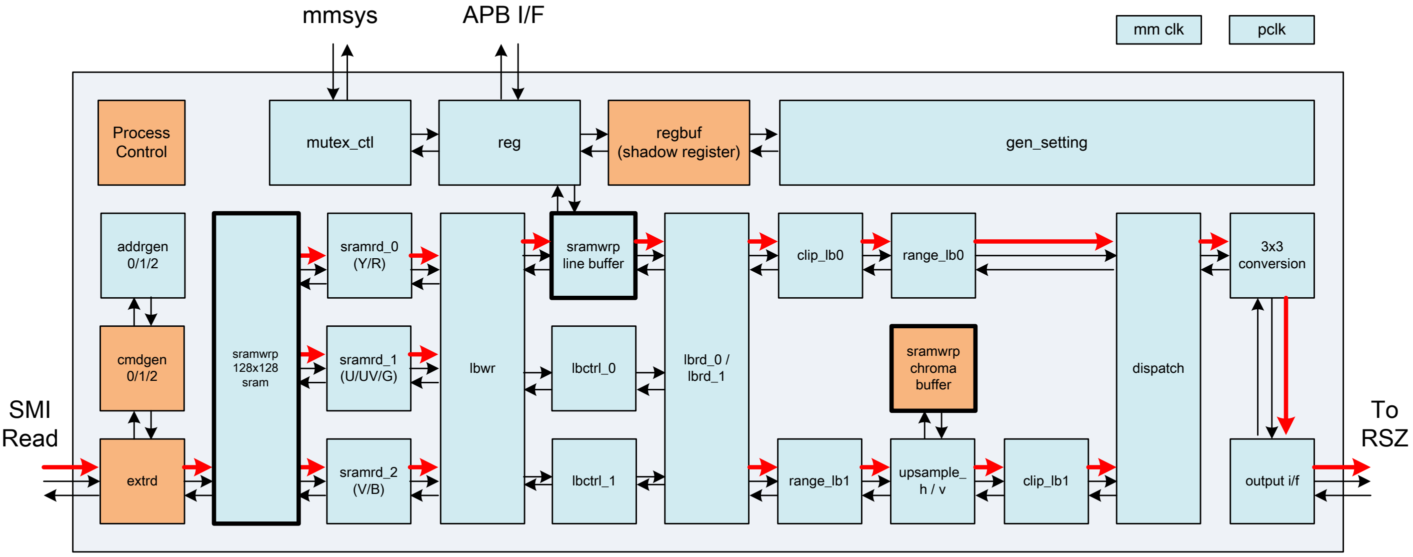


*Figure 6-6. Block diagram of DISP MDP_RDMA*

### 6.6.4 Register Definition

See chapater 4.6 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 6.6.5 Programming Guide

To enable MDP_RDMA to read a frame, SW can be programmed by the following method. Here we offer a quick example: With an YCbCr_420_P_SW 1920x1080 frame provided in DRAM, MDP_RDMA will try to clip it into 256x256 as the following example settings:

1. Release reset and enable all interrupt
   DISP_ROT_RESET = 0x0000_0000
   DISP_ROT_INTERRUPT_ENABLE = 0x0000_0007
2. Set up controls to determine the behavior of MDP_RDMA.
   DISP_ROT_CON = 0x0000_0000
3. Set up SMI configuration for performance issues.
   DISP_ROT_GMCIF_CON = 0x0000_1771
4. Set up input format.
   DISP_ROT_SRC_CON = 0x0000_0000
5. Set up base address 0 for the first plane of the source frame. If the source frame contains more than one plane, base address 1 and base address 2 should be set as well. Note that the offset of start address is relative to operations. Refer to appendix for the start address offset cases.
   DISP_ROT_SRC_BASE_0 = 0x0007_7880
   DISP_ROT_SRC_BASE_1 = 0x0080_6c40
   DISP_ROT_SRC_BASE_2 = 0x00A0_1040
6. Calculate the byte number of the line pitch. It is the distance from the first byte of the current line to the first byte of the next line. The value for sub frame is required for multiple planar source.
   DISP_ROT_MF_BKGD_SIZE_IN_BYTE = 0x0000_0780
   DISP_ROT_SF_BKGD_SIZE_IN_BYTE = 0x0000_0780
7. Set up the width and the height of the source frame before it is rotated.
   DISP_ROT_MF_SRC_SIZE = 0x0100_0100
8. Set up the clip size if necessary.
   DISP_ROT_MF_CLIP_SIZE = 0x0100_0100
9. Set up offset in source format if necessary.
   DISP_ROT_MF_OFFSET_1 = 0x0000_0000
10. Set up option for RGB to YUV color transform control registers and parameters if necessary.
    DISP_ROT_TRANSFORM_0 = 0x0000_0000
11. Assert ROTEN to enable MDP_RDMA to wait for update of the shadow register.
    DISP_ROT_EN = 0x0000_0001
12. Assert register update to start operation of MDP_RDMA. This control signal comes from dispsys mutex control.

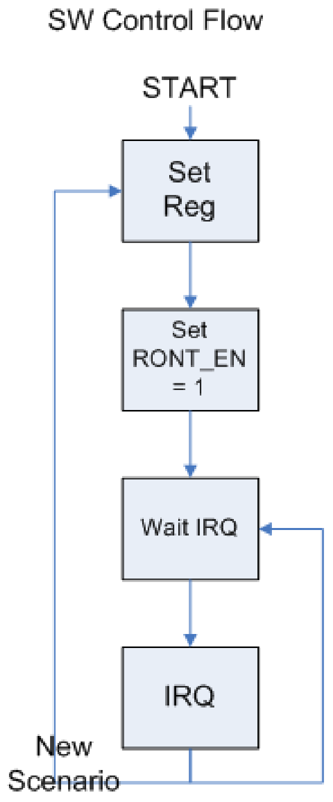


*Figure 6-7. SW control flow*

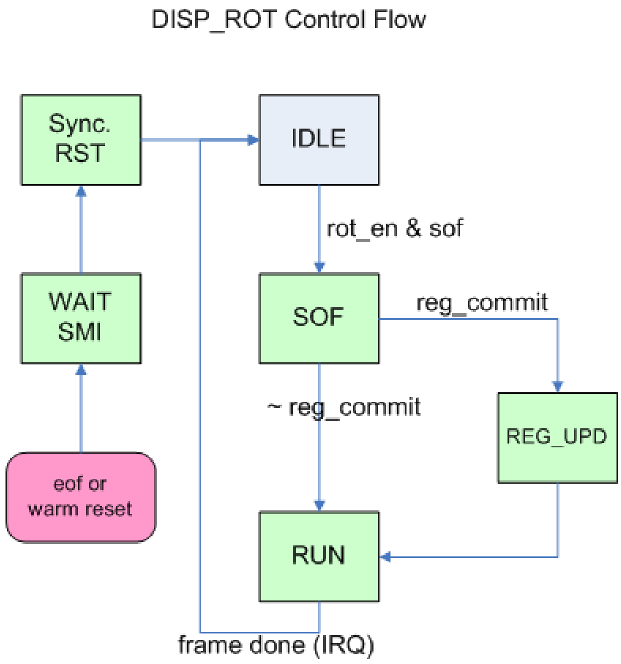


*Figure 6-8. HW FSM and mutex control flow*

## 6.7 MDP RSZ

### 6.7.1 Introduction

There are two resizer modules for MT6737 MDP, including MDP_RSZ0 and MDP_RSZ1. The algorithm of each may be slightly different for different project requirements. The table lists the corresponding algorithm spec. Table 6-3 shows the functional specification of MDP_RSZ0 and MDP_RSZ1. MDP_RSZ0 and MDP_RSZ1 scale for multiple purposes in MDP 2.0 structure, mainly for generating image for display, video codec, jpeg codec and FD. Digital zoom is accomplished by MDP_RSZ0 or MDP_RSZ1 in MDP 2.0. MDP_RSZ0 and MDP_RSZ1 supports YUV444 input and YUV444 output.

Table 6-4 shows the hardware specifications of MDP_RSZ0 and MDP_RSZ1. They support three scaling algorithms including 6-tap FIR, 4n-tap cubic accumulation and n-tap source accumulation. The maximum image width is 544.

***Table 6-3. Functional specifications of resizers***

| | MDP_RSZ0 | MDP_RSZ1 |
|---|---|---|
| Input data format | 8-bit YUV444 (unsigned) | 8-bit YUV444 (unsigned) |
| Output data format | 8-bit YUV444 (unsigned) | 8-bit YUV444 (unsigned) |
| Scaling ratio | Between 1/128x and 64x | Between 1/128x and 64x |
| Crop function | Supported | Supported |
| Function | 1. Digital zoom<br>2. MDP general-purpose resizing | 1. Digital zoom<br>2. MDP general-purpose resizing |
| Supported width (OTF) | N/A | N/A |
| Supported width (tile mode) | 6tap: 544<br>ntap: 544<br>4ntap:272 | 6tap: 544<br>ntap: 544<br>4ntap:272 |

*Table 6-4. Hardware specifications of resizers*

| | MDP_RSZ0 | MDP_RSZ1 |
|---|---|---|
| Luma interpolation method (H) | (1) 6 tap FIR<br>(2) 4n tap CA<br>(3) n tap SA | (1) 6 tap FIR<br>(2) 4n tap CA<br>(3) n tap SA |
| Luma interpolation method (V) | (1) 6 tap FIR<br>(2) 4n tap CA<br>(3) n tap SA | (1) 6 tap FIR<br>(2) 4n tap CA<br>(3) n tap SA |
| Chroma interpolation method (H) | (1) 6 tap FIR<br>(2) 2n tap TA<br>(3) n tap SA | (1) 6 tap FIR<br>(2) 2n tap TA<br>(3) n tap SA |
| Chroma interpolation method (V) | (1) 6 tap FIR<br>(2) 2n tap TA<br>(3) n tap SA | (1) 6 tap FIR<br>(2) 2n tap TA<br>(3) n tap SA |
| Interpolation order | H->V (xn tap)<br>V->H (6 tap) | H->V (xn tap)<br>V->H (6 tap) |
| Line buffer (frame mode) | N/A | N/A |
| Line buffer (tile mode) | 272x48x6 (Logical)<br>-> 136x96x6 | 272x48x6 (Logical)<br>-> 136x96x6 |

### 6.7.2 Theory of Operations

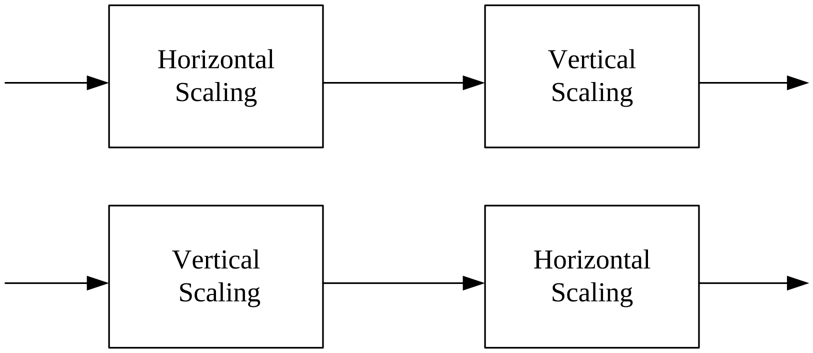


*Figure 6-9. Separate 1D FIR operaiton*

Basically, the rescaling procedure is composed of two separate 1D FIR operations (see Figure 6-9). There are three major types of 1D FIR operation, 6 tap FIR, 4n tap cubic accumulation and n tap source accumulation. These three algorithms have different rescaling characters and tap numbers. 6 tap is suitable for up-scaling and down-scaling (1X~1/2X). It is a fixed 6 tap FIR operation and needs 18-line buffer to do vertical scaling operation (see Figure 6-10).

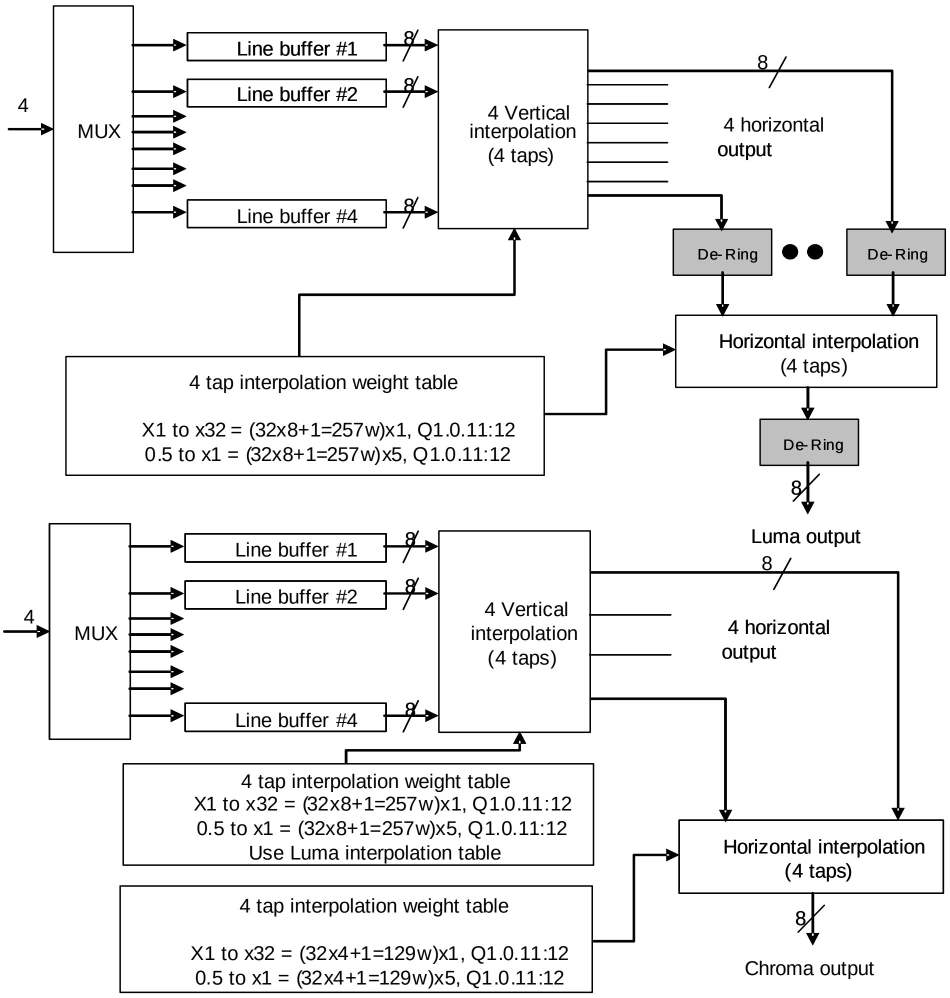
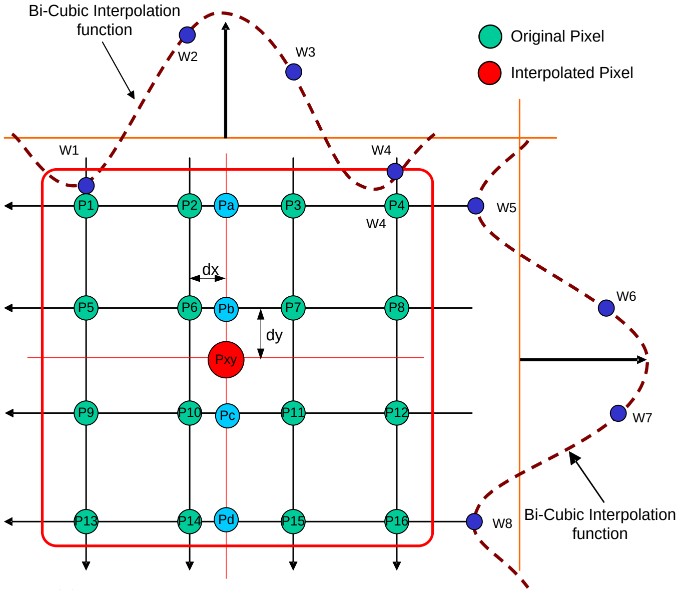

4
MUX
Line buffer #1
Line buffer #2
Line buffer #4
8
4 Vertical interpolation (4 taps)
8
4 horizontal output
De-Ring
De-Ring
Horizontal interpolation (4 taps)
4 tap interpolation weight table
X1 to x32 = (32x8+1=257w)x1, Q1.0.11:12
0.5 to x1 = (32x8+1=257w)x5, Q1.0.11:12
De-Ring
8
Luma output
4
MUX
Line buffer #1
Line buffer #2
Line buffer #4
4 Vertical interpolation (4 taps)
8
4 horizontal output
4 tap interpolation weight table
X1 to x32 = (32x8+1=257w)x1, Q1.0.11:12
0.5 to x1 = (32x8+1=257w)x5, Q1.0.11:12
Use Luma interpolation table
Horizontal interpolation (4 taps)
4 tap interpolation weight table
X1 to x32 = (32x4+1=129w)x1, Q1.0.11:12
0.5 to x1 = (32x4+1=129w)x5, Q1.0.11:12
8
Chroma output

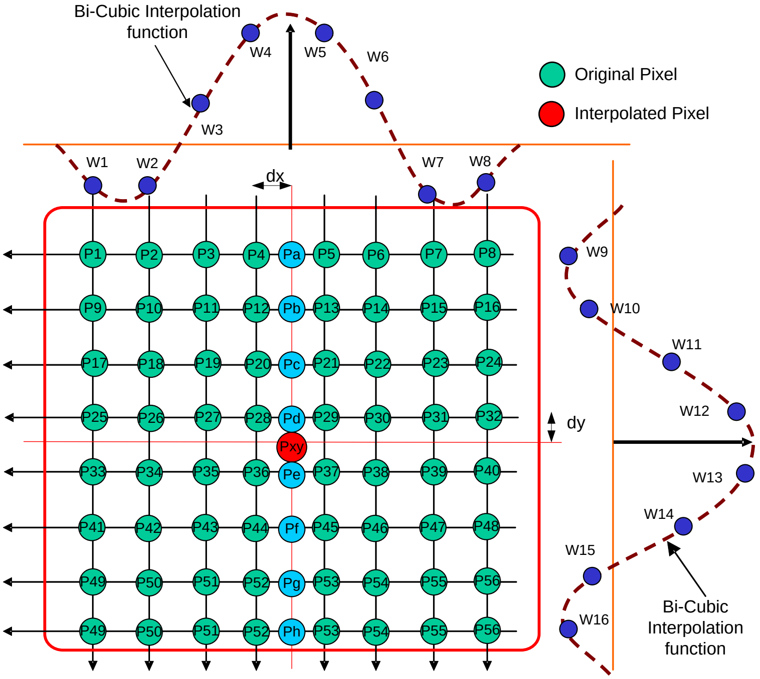
***Figure 6-10. 4 tap cubic block diagram (Luma and Chroma data, in MT6737 4tap changes to 6tap)***

4n tap cubic accumulation has better quality than 6 tap FIR when the down-scaling ratio is bigger than 1/2X. It is essential a variable tap FIR operation and its tap number is determined by scaling ratio (4 times n, n is scaling ratio). For example, when the scaling ratio is 1, the tap number is 4 (see Figure 6-11). When the scaling ratio is 1/2, the tap number is 8 (see Figure 6-12). In addition, n tap source accumulation is also a variable tap number FIR algorithm. It has poor sharpness and is an extremely low cost solution.



Weight_X = W1+W2+W3+W4;
Weight_Y = W5+W6+W7+W8;
Pa=(P1*W1 + P2*W2 + P3*W3 + P4*W4+Weight_X/2)/Weight_X
Pb=(P5*W1 + P6*W2 + P7*W3 + P8*W4+Weight_X/2)/Weight_X
Pc=(P9*W1 + P10*W2 + P11*W3 + P12*W4+Weight_X/2)/Weight_X
Pd=(P13*W1 + P14*W2 + P15*W3 + P16*W4+Weight_X/2)/Weight_X
Pxy=(Pa*W5 + Pb*W6 + Pc*W7 + Pd*W8+Weight_Y/2)/Weight_Y

*Figure 6-11. Cubic accumulation (scaling ratio = 1x)*



Weight_X = W1+W2+W3+W4+W5+W6+W7+W8;
Weight_Y = W9+W10+W11+W12+W13+W14+W15+W16;
Pa=(P1*W1 + P2*W2 + P3*W3 + P4*W4 + P5*W5 + P6*W6 + P7*W7 + P8*W8 + Weight_X/2)/Weight_X
Pb, Pc, Pd, Pe, Pf, Pg, Ph Is Interpolated In the same way
Pxy=(Pa*W9 + Pb*W10 + Pc*W11 + Pd*W12 + Pe*W13 + Pf*W14 + Pg*W15 + Ph*W16+ Weight_Y/2)/Weight_Y

*Figure 6-12. Cubic accumulation (scaling ratio = 1/2x)*

### 6.7.3 Programming Guide

#### 6.7.3.1 Suggested Algorithm

Corresponding to applied scaling ratio, here is the suggested algorithm for the resizer to choose from. The algorithm is suggested for the best quality. The principles are:

- 6tap: 32x ~ 1/2x
- 4ntap: 1/2x ~ 1/64x
- ntap: 1x ~ 1/128x

However, the suggested algorithm may contradict with limited supporting size in each resizer engine. Take CDRZ for example, as described in the figure below, if a certain application needs to use CDRZ to resize from 3000 to 1400, it is suggested to use 4ntap cubic accumulation algorithm. However, the supported size of 4ntap in CDRZ is 1152, which is smaller than 1400. Therefore, in this case choose ntap to be applied.

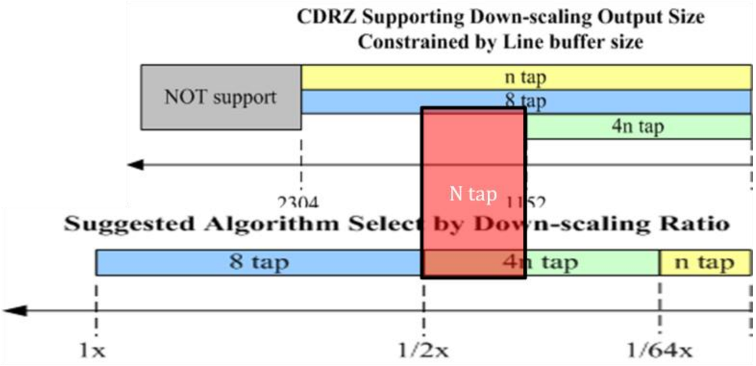

CDRZ Supporting Down-scaling Output Size
Constrained by Line buffer size
NOT support
n tap
8 tap
4n tap
2304
1152
N tap
Suggested Algorithm Select by Down-scaling Ratio
8 tap
4n tap
n tap
1x
1/2x
1/64x

*Figure 6-13. Suggested algorithm v.s. supported size*

#### 6.7.3.2 Coefficient Step Calculation

The coefficient step is calculated in different way for different algorithms. Refer to the descriptions below.

- 6tap:
  - Unit base: 32768 (2^15)
  - M_m1 = Input_size **– In_Int_Ofst – (In_Sub_Ofst==0? 0 : 1)** -1
  - N_m1 = Output_size -1
  - Cstep = (M_m1*UNIT+(N_m1>>1) )/N_m1
- 6tap:
  - Unit base: **65536** (2^16) (note the offset settings)
- 4ntap:
  - Unit base: 1048576 (2^20)
  - M_m1 = Input_size **– In_Int_Ofst – (In_Sub_Ofst==0? 0 : 1)** -1

  - N_m1 = Output_size -1
  - Cstep = (N_m1*UNIT+(M_m1-1) )/M_m1
  - Transfer offset from input to output
- N-tap:
  - Unit base: 1048576 (2^20)
  - M_m1 = Input_size **– In_Int_Ofst – (In_Sub_Ofst==0? 0 : 1)** -1
  - N_m1 = Output_size -1
  - Cstep = (N_m1*UNIT+(M_m1-1) )/M_m1
  - Transfer offset from input to output

### 6.7.3.3 Input/Output Offset Transfer

- For 6 tap interpolation, set “Input Offset” into register.
  - Set register Int_Ofst = In_Int_Ofst ;
  - Set register Sub_Ofst = In_Sub_Ofst ;
- For Accumulation, set “Output Offset” into register.
  - Transfer Input Offset to Output Offset
  - Out_Int_Ofst = (In_Int_Ofst*Cstep + In_Sub_Ofst*Cstep/Unit)/Unit
  - Out_Sub_Ofst = (In_Int_Ofst*Cstep + In_Sub_Ofst*Cstep/Unit) % Unit

### 6.7.3.4 Table Selection

- 6tap/6tap: Depends on scaling ratio
  - [1~32768]: Table 27 or Table 0~19
  - (32768:36409): Table 20
  - [36409:40961): Table 21
  - [40961:46812): Table 22
  - [46812:54614): Table 23
  - [46812:59579): Table 24
  - [59579:65537): Table 25
  - [65537~∞): Table 26
- 4n tap (downscaling only)
  - 1~19 ( blur ~ sharp ), 15 is recommended
- N tap (downscaling only): No table selection required

The entire programming guide is as the following steps:

- Decide input/output size.
- Check “Maximum Size” constraint.
- Calculate the ratio.
- Choose algorithm from “Suggested Algorithm”.
- Check “Supporting Size” constraint.
- Trigger Resizer and wait for IRQ.

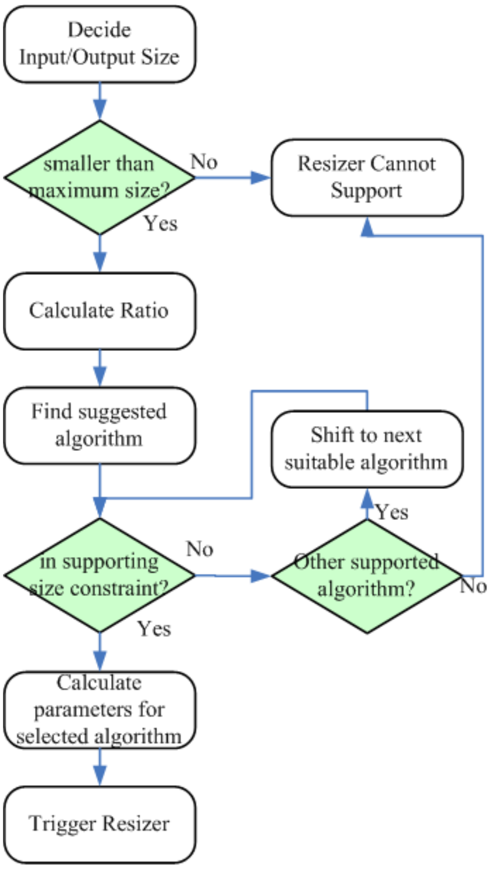

Decide Input/Output Size
smaller than maximum size?
No
Yes
Resizer Cannot Support
Calculate Ratio
Find suggested algorithm
Shift to next suitable algorithm
Yes
in supporting size constraint?
No
Other supported algorithm?
No
Yes
Calculate parameters for selected algorithm
Trigger Resizer

***Figure 6-14. Programming guide***

### 6.7.3.5 Programming Method

In MT6737, there are two programming methods

- CPU program through APB bus
- Command queue

Use proper programming method for different applications. In the video recording application, program CDRZ and MDP_RSZ0/MDP_RSZ1 by command queue to synchronize frame level setting between ISP  and MDP processing.

## 6.7.4 Register Definition

See chapater 4.7 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 6.8 MDP ROT_DMA (Multimedia Data Path-Rotation DMA)

### 6.8.1 Introduction

ROT_DMA is a write rotate DMA agent supporting 8 rotation/flip options. The users may hold the phone in any orientation, and we need to rotate the image in the correct direction.

### 6.8.2 Features

- Rotation Angels: 0°, 0° + H_Flip, 90°, 90° + H_Flip, 180°, 180° + H_Flip, 270°, and 270° + H_Flip as illustrated in Figure 6-15.
- Format and Footprint: YUV422 1/2/3 plane, YUV420 2/3 plane, RGB888, ARGB8888, RGB565, Y only
- Programmable RGB color matrix
- Dither engine

### 6.8.3 Block Diagram

See the figure below for the engine architecture, including color matrix, dither, sub-sampling, rotator and DMA engines.

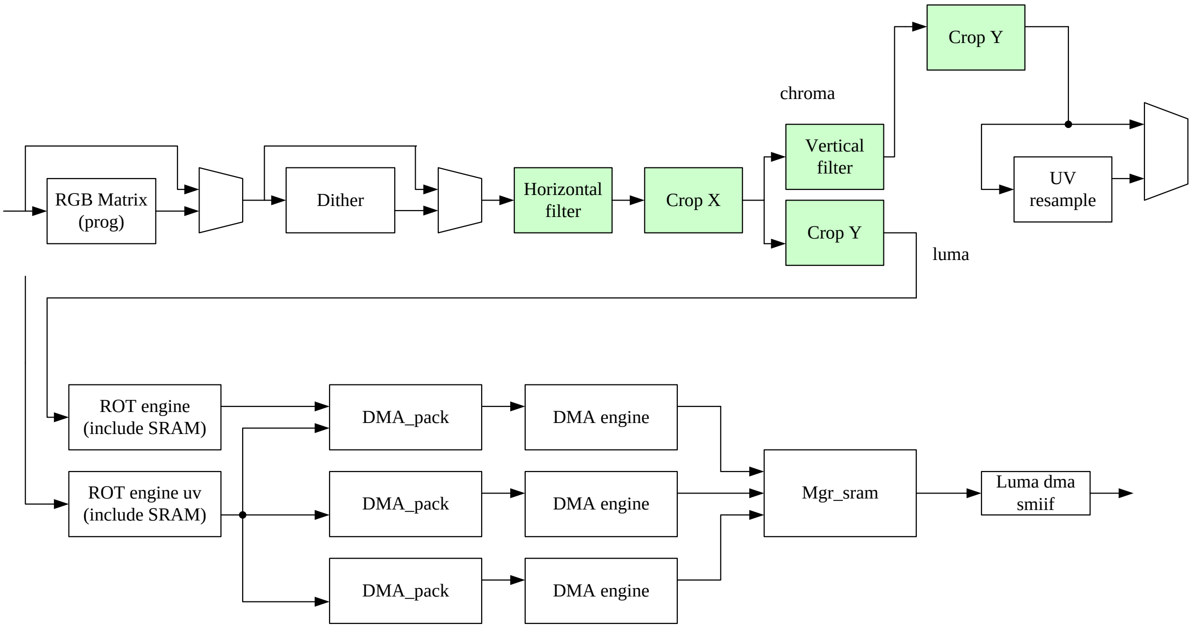


*Figure 6-15. Block diagram of ROT_DMA*

### 6.8.4 Register Definition

See chapater 4.8 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

### 6.8.5 Programming Guide

#### 6.8.5.1 Firmware Settings for Rotation/Flip

To output data to DRAM with rotation/flip, ROT_DMA should calculate the correct address to put the first data. Then the next data can be written in certain order which also depends on rotation/flip settings. However, the circuit for calculating the address to put the first data is only applied in the beginning of the entire image or tile. For cost effectiveness, the address offset for the first data compared to BASE_ADDR can be calculated by firmware and transferred into hardware by register OFST_ADDR. For all kinds of rotation/flip settings, the required positions of OFST_ADDR are different. The illustrations for the position of OFST_ADDR on DRAM footprint is shown in Figure 6-16 to Figure 6-19. Note that the xsize and ysize are defined according to input data scan-line direction. The direction of xsize is parallel to input scan-line direction, and the direction of ysize is perpendicular to input scan-line direction. The stride setting is defined according to DRAM footprint, which is the direction after rotation/flip.

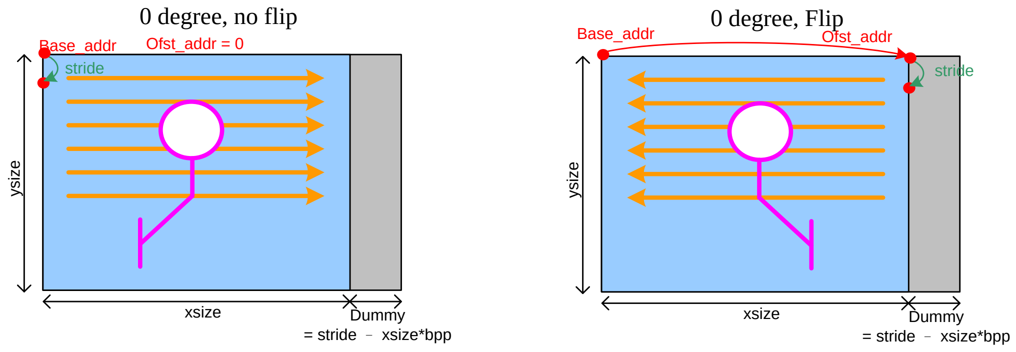


***Figure 6-16. Firmware settings of OFST_ADDR in 0° rotation (scan-line)***

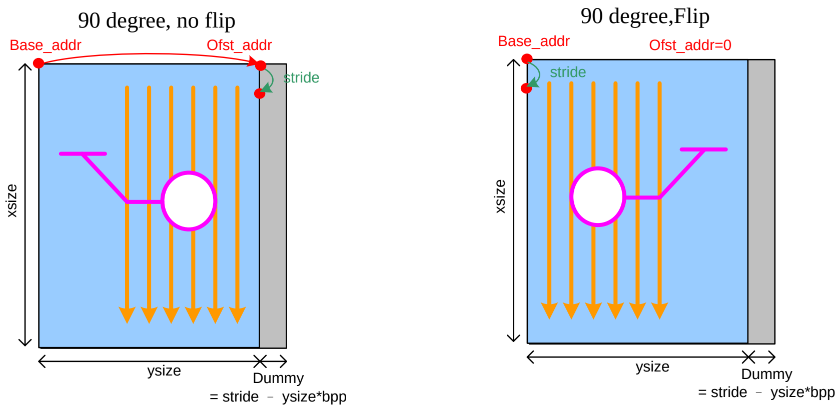


***Figure 6-17. Firmware settings of OFST_ADDR in 90° rotation (scan-line)***

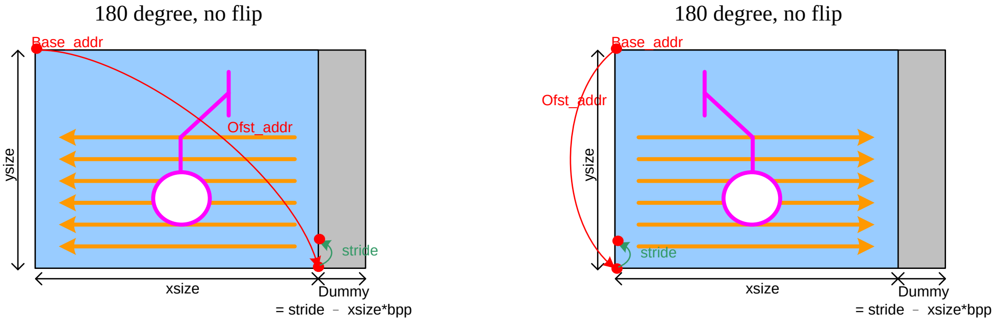


*Figure 6-18. Firmware settings of OFST_ADDR in 180° rotation (scan-line)*

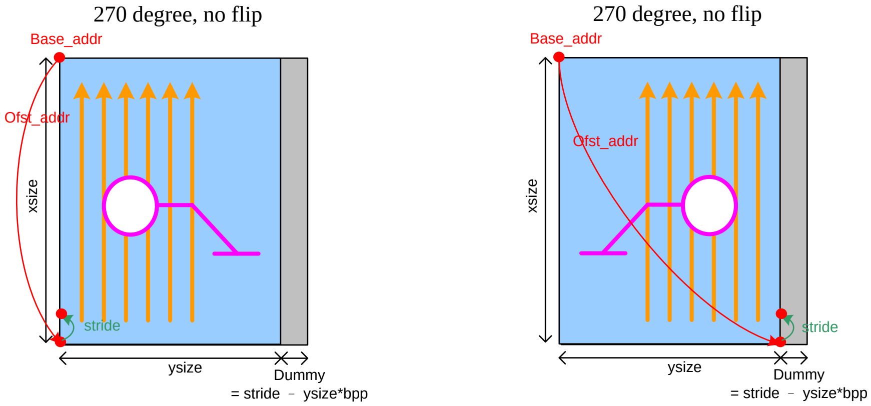


*Figure 6-19. Firmware settings of OFST_ADDR in 270° rotation (scan-line)*

Summary is given in Table 6-5.

*Table 6-5. ROT_DMA OFST_ADDR settings for rotation/flip*

| Rotation | No Flip<br>(Flip = 0) | Horizontal Flip<br>(Flip = 1) |
|---|---|---|
| 0 degree<br>(Rotation=0) | 0 | Xsize-1 |
| 90 degree<br>(Rotation=1) | Ysize-1 | 0 |
| 180 degree<br>(Rotation=2) | Stride*(Ysize-1) +(Xsize-1) | Stride*(Ysize-1) |
| 270 degree<br>(Rotation=3) | Stride*(xsize-1) | Stride*(Xsize-1) +(Ysize-1) |

For an interpolation based UV-resampler, which can perform better quality for YUV420, VIDO UVSEL should be set corresponding to different rotation/flip settings. According to Table 6-6, the suggested settings provide the proper chroma sampled points.

*Table 6-6. VIDO UV SEL for YUV420 format cooperated with CRSP*

| Rotation | Flip | UV_SELX | UV_SELY |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 |
| 2 | 0 | 1 | 1 |
| 2 | 1 | 0 | 1 |
| 3 | 0 | 0 | 1 |
| 3 | 1 | 0 | 1 |

#### 6.8.5.2 Working Buffer Height Setting

Setting up the buffer width and buffer height for ROT_DMA is necessary, no matter the rotation is enabled or not. Set N mod M = 0 for better performance. If N mod M is not 0, and you would like the efficiency to drop as little as possible, derive an algorithm for height calculation.

First of all, you need to know the buffer size for each format.

```
if (UYVY 2-plane or 3-plane)
     y_max_buf_size = 256x48
    uv_max_buf_size = 128x48
else if (YUV420)
     y_max_buf_size = 256x64
    uv_max_buf_size = 128x32
else
     y_max_buf_size = 256x32
    uv_max_buf_size = 256x32
```

Algorithm: (Width is tile width.)
Phase 1:
We can apply the algorithm on Y channel first to get the approximation first.

Coeff 1= floor (MAX_BUF_SIZE / WIDTH/2) *2
Coeff 2= Ceiling (WIDTH / Coeff1)
Buf_line_num= Ceiling (WIDTH/Coeff 2/ 4) *4 (To make sure if Buf_line_num mod 4 = 0)

If (buf_line_num > width)
Buf_line_num = ceiling (width / 4) * 4 (To make sure width >= Buf_line_num)

else if (Buf_line_num * Buf_line_num * Coeff 2 > MAX_BUF_SIZE) (buffer overflow)
Buf_line_num = buf_height.

Phase 2:
Check if the setting is over the buffer size or not.

```
Y_buf_check =0
Uv_buf_check =0

//internal buffer check
while ((y_buf_check !=0) | (uv_buf_check!=0)){

     // Y buffer check
     Internal_y_buf_width = Ceiling(width/buf_line_num) x buf_line_num //multiple of
buf_line_num and >= width
     Internal_y_buf_usage = Internal_y_buf_width x buf_line_num
     If (internal_y_buf_usage > y_max_buf_size){
         buf_line_num = buf_line_num -4
        Y_buf_check =0
          Uv_buf_check =0
     }else{
         Y_buf_check =1
     }

     // UV buffer check
     if (YUV422 rotate 0/180)
         Uv_blk_width = main_blk_width/2
         Uv_blk_line    = main_buf_line_num
     Else if ((YUV422 rotate 90/270)
         Uv_blk_width = main_blk_width
         Uv_blk_line    = main_buf_line_num/2
     Else if (YUV420)
         Uv_blk_width = main_blk_width/2
         Uv_blk_line    = main_buf_line_num/2
    Else
          Uv_blk_width = main_blk_width
         Uv_blk_line    = main_buf_line_num

     Internal_uv_buf_width = Ceiling(Uv_blk_width / Uv_blk_line)x Uv_blk_line

     Internal_uv_buf_usage = Uv_blk_width x Uv_blk_line
     If (internal_uv_buf_usage > uv_max_buf_size){
         Main_buf_line_num = maian_buf_line_num -4
        Y_buf_check =0
          Uv_buf_check =0
     }else{
         uv_buf_check =1
     }
}
```

### 6.8.5.3 SMI 256 Byte Boundary Restriction

#### 6.8.5.3.1 Methods to Handle Boundary Restriction

SMI has a restriction that any burst issued from DMA cannot cross the 256-byte address boundary, no matter in read or write action. The reason is from the bank configuration in SMI, where the 256-byte address is only the bank switching point.

Here we try to analyze the probability of crossing the 256-byte boundary by random access. As illustrated in Figure 6-20. The probability P is

P = (byte per burst length)/256

It depends on the designed burst length of ROT_DMA. For example, if the minimum burst length 4-16, the probability P = 64/256 = ¼, which means that 25% burst request will be separated. Though the total data amount is fixed, it increases 25% burst request times and will reduce DRAM access utility.

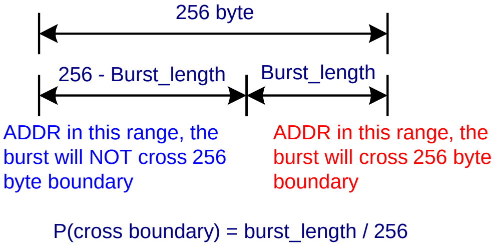

256 byte
256 - Burst_length
Burst_length
ADDR in this range, the burst will NOT cross 256 byte boundary
ADDR in this range, the burst will cross 256 byte boundary
P(cross boundary) = burst_length / 256

*Figure 6-20. Probability of crossing 256-byte boundary by random access*

To handle this restriction from SMI, there are two common solutions. One is adopting a burst separator between ROT_DMA and SMI port. As illustrated in Figure 6-21, the original DMA requests are represented in black arrow. Because it is not considered 256 byte boundary restriction, some requests should be cross the 256-byte boundary. The burst separator filters all requests from ROT_DMA, which allows the legal request pass as represented in blue arrows but separate the illegal request into two requests as represented in yellow arrows. Although it is simple to implement, it increases the total request number.

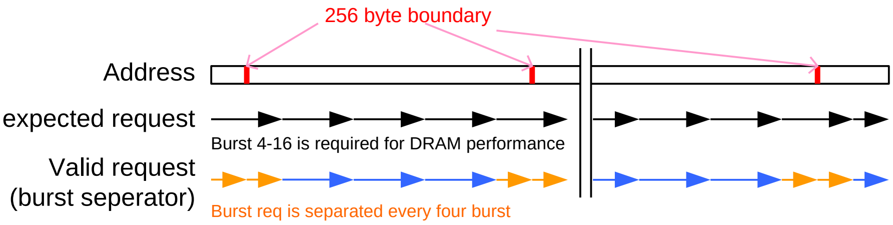

256 byte boundary
Address
expected request
Burst 4-16 is required for DRAM performance
Valid request
(burst seperator)
Burst req is separated every four burst

*Figure 6-21. Scan-line request without considering 256-byte boundary*

The second method is handling the 256-byte boundary restriction inside the control logics of ROT_DMA. After the data are stored in burst accumulator, ROT_DMA will adjust the burst length to meet the boundary when the target address is close to the 256-byte boundary. In common cases, the minimum burst length is a factor of 256 bytes; therefore the requests after the adjusted request will automatically align with the 256-byte boundary. This method is difficult to implement but can efficiently reduce the overhead of DMA requests (see Figure 6-22).

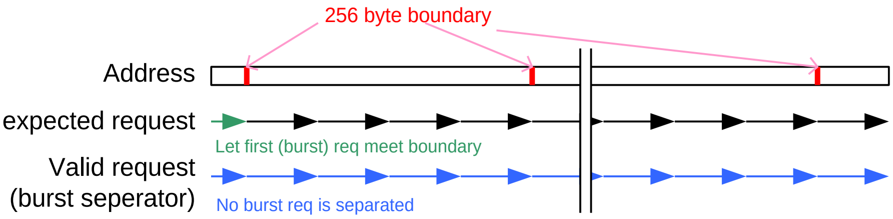


*Figure 6-22. Scan-line request considering 256-byte boundary*

### 6.8.5.3.2 SMI Boundary Restriction In Case of Rotation

The method described in the previous section is based on continuous scan-line access of DMA. It works only in 0° or 180° rotation with or without flip. Unfortunately, it does not always work in 90° or 270° rotation with or without flip. Considering a general condition with 90° or 270° rotation, if the stride is not multiple of 256 bytes, the distribution of 256-byte boundary will differ in every line as described in Figure 6-23. To handle this, the SMI_IF controller issues requests with different burst lengths according to the current address.

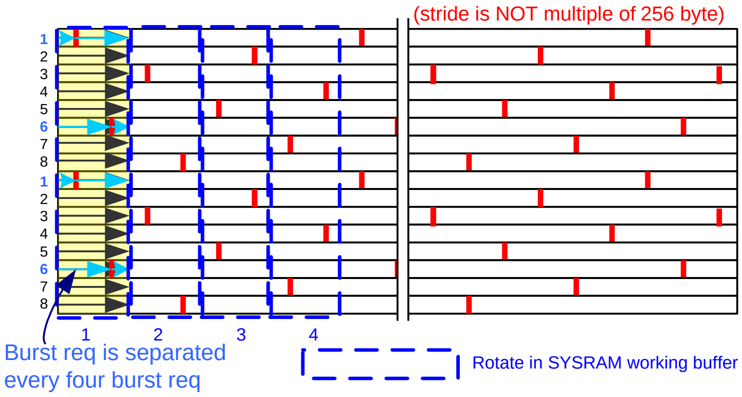

(stride is NOT multiple of 256 byte)
1
2
3
4
5
6
7
8
1
2
3
4
5
6
7
8
1
2
3
4
Burst req is separated
every four burst req
Rotate in SYSRAM working buffer

*Figure 6-23. 4-16 burst in 270° rotation with/without flip*

## 6.9 Display 2D Sharpness

### 6.9.1 Introduction

The sharpness function provides better picture quality for panel display. It restores the image details, sharpens the edge and provides a vivid feeling for pictures and videos.

### 6.9.2 Features

- 2-dimensional sharpness filter
- Peaking by color (PBC)

### 6.9.3 Block Diagram

Figure 6-24 is the block diagram of display 2D sharpness. The FIFO unit is used to pre-fetch pixel data, and thus the overall throughput can be improved in some situations.

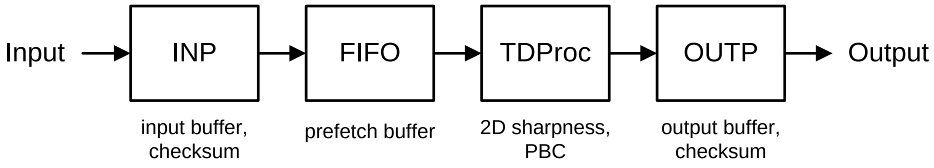


*Figure 6-24. Block diagram of display 2D sharpness*

Figure 6-25 is the block diagram of the sharpness core. The luminance data will be processed and modified in the sharpness function. The chrominance data will flow into PBC, but no modification at the output.

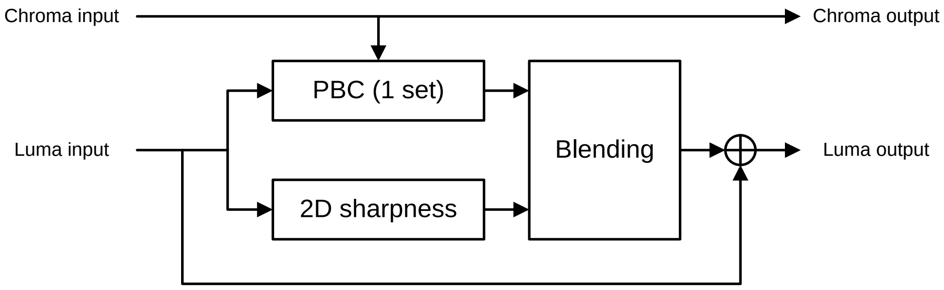
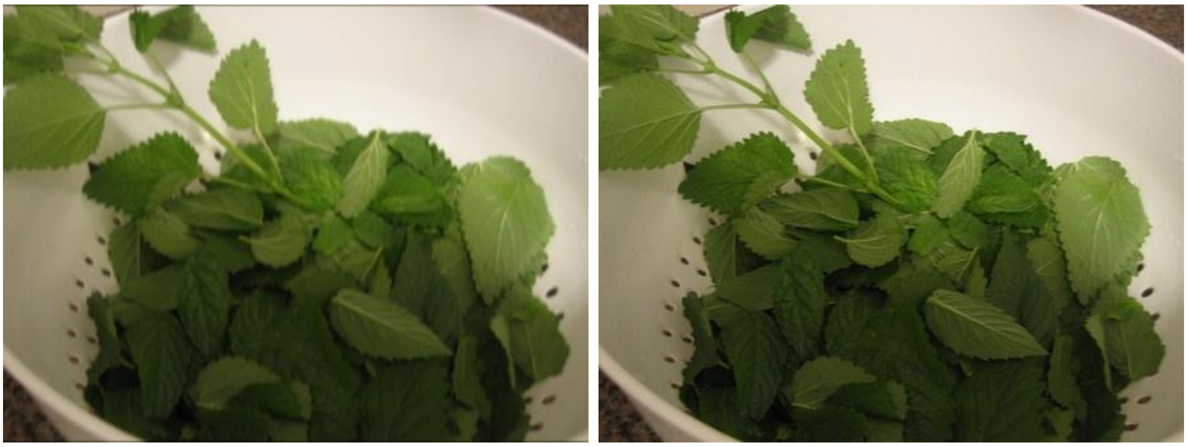


*Figure 6-25. Block diagram of sharpness core*

#### 6.9.3.1 2-dimensional Sharpness

The 2D filters are used to extract middle/high-frequency components from the input signal. Each extracted AC component is enhanced individually and then added back to the input signal. The enhancement units are composed by coring, gain, limit and clip.

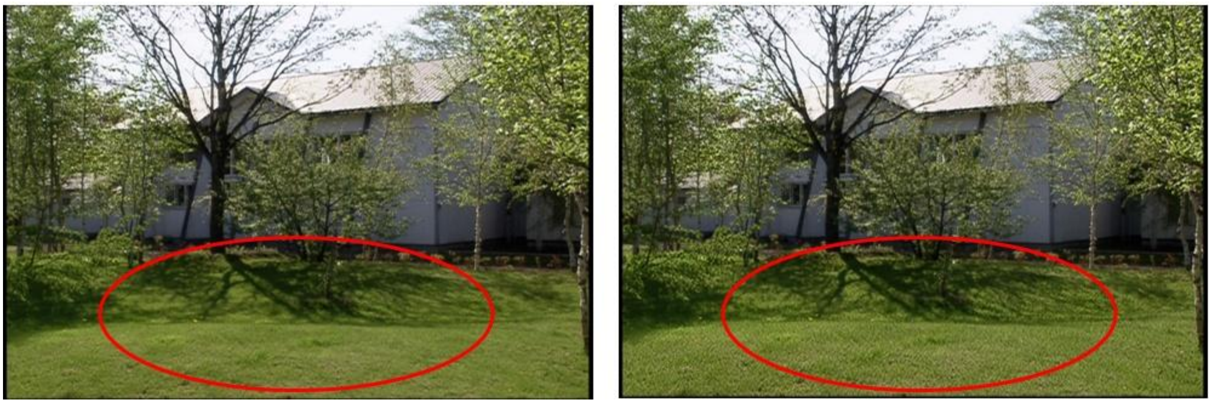


Before 2-dimensional sharpness After 2-dimensional sharpness

***Figure 6-26. Visual effect of 2-dimensional sharpness***

#### 6.9.3.2 Peaking by Color

Sometimes you may prefer different sharpness levels in different color tones. PBC is used for such preference. PBC detects at most three different colors and applies different sharpness levels to them.



Before green PBC After green PBC

***Figure 6-27. Visual effect of peaking by color***

### 6.9.4 Register Definition

See chapater 4.9 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

### 6.9.5 Programming Guide

#### 6.9.5.1 Sharpness Adjustment

There are total three possible usage scenarios for different situations.

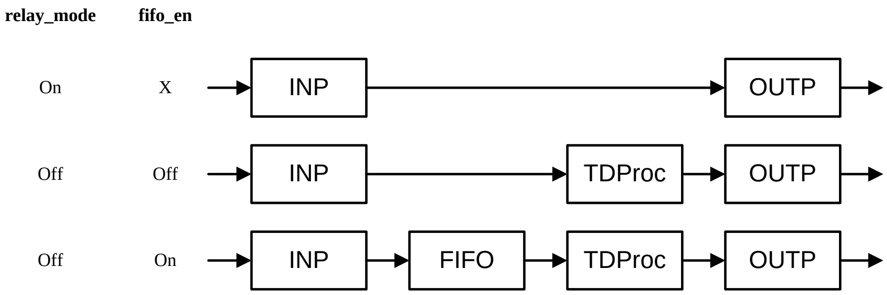


***Figure 6-28. Usage scenarios***

The sharpness function is designed to provide different sharpness levels. The sharpness level can be programmed by the gain curve control in the 2-dimensional sharpness unit. Higher sharpness level enhances more details of the image and sharpens the edge. However, sharpness setting that is too strong will induce some artificial side effects. Therefore, balance setting with enough sharpness level is preferred.

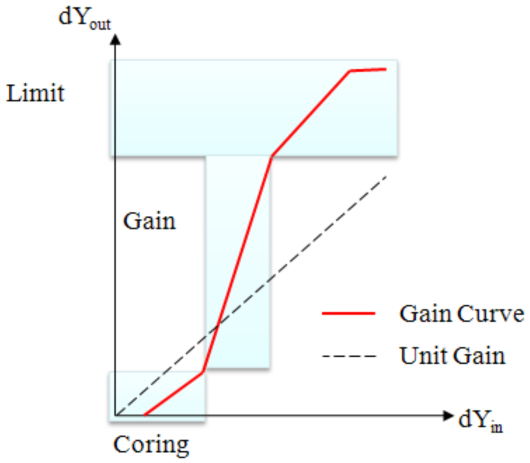
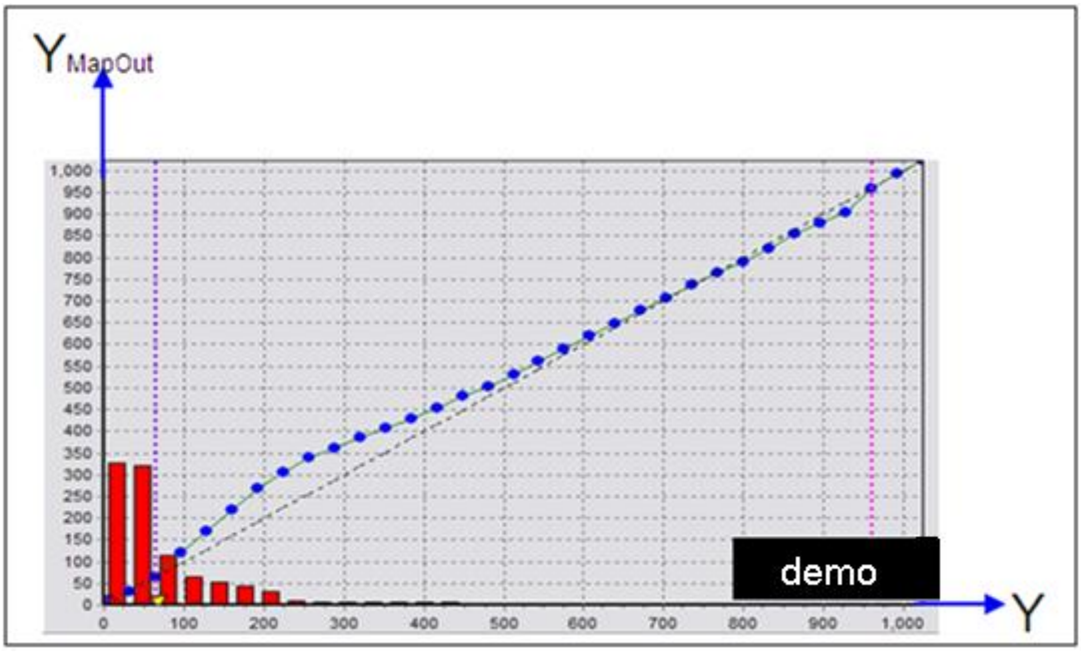


***Figure 6-29. Gain curve control of sharpness***

### 6.9.5.2 Luma Adjustment

The color processor applies a luma mapping curve to the input luma (see the figure below). The full luma range is equally divided into 16 segments, and registers Y_FTN_* are responsible for the adjustments.



## 6.10 DISP_OVL

### 6.10.1 Introduction

The display OVERLAY can do alpha blending up to 4 layers. The 4 layer sources can be from MEMORY/constant layer color. Four RDMA is included in overlay and 4 sets of color transform are also included.

### 6.10.2 Features

The following table describes the features of DISP_OVL.

| Item | Main function | Description |
|---|---|---|
| 1 | 4K2K resolution | Supports 4096x2160 |
| 2 | 4 layer overlay | Supports 4 layers of blending (w/o multiplier) |
| 3 | Color format uniform | Supports color format uniform and swap control (RGB565/RGB888/RGBA8888/ARGB8888/YUYV/YUV2) |
| 4 | 3D display | Interleave left and right image for 3D display (landscape and portrait mode) |
| 5 | Color conversion | Wide-gamma : sRGB/GPU/Adobe mode |
| 6 | Layer constant color | Constant color input for each layer source |
| 7 | Alpha blending | Supports pixel alpha blending +<br>SurfaceFlinger alpha blending (G2D) |
| 8 | Flexible ROI system | Supports individual color depth, window size, vertical and horizontal offset |
| 9 | Flip function | Vertical/Horizontal/180 degree flip function |

### 6.10.3 Block Diagram

There are four OVL_RDMA in DISP_OVL, and each OVL_RDMA contains a 128x128 single port SRAM (ping-pung buffer).

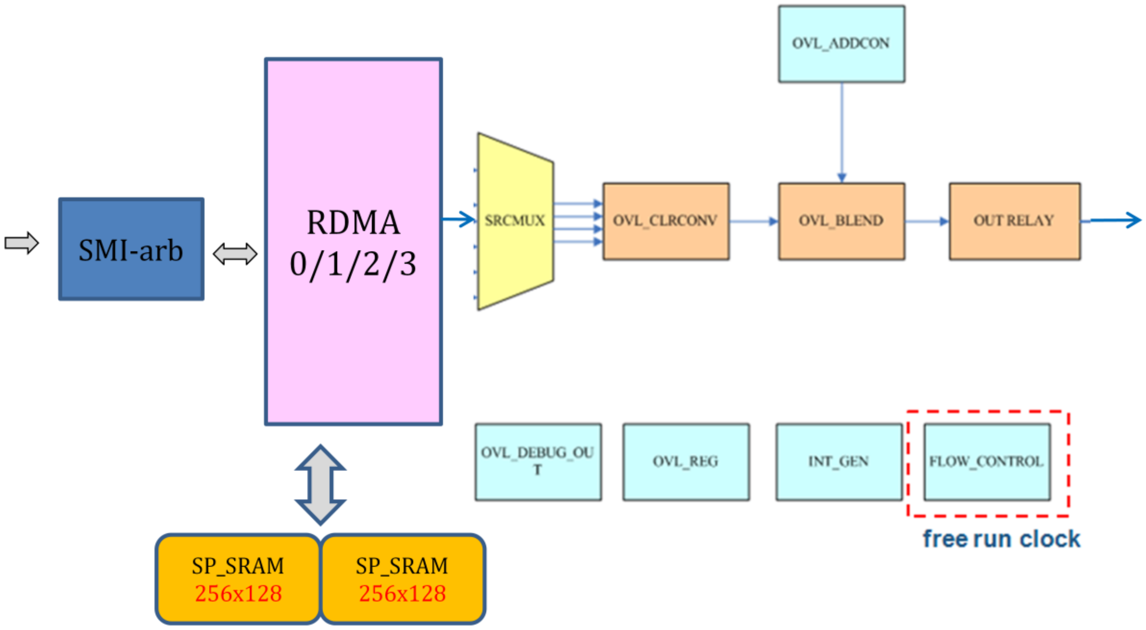


*Figure 6-30. Block diagram of DISP_OVL*

### 6.10.4 Register Definition

See chapater 4.10 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 6.10.5 Programming Guide

**Register settings:**

1. Configure DATAPATH (number of layers, layer source, color format, etc).
2. Configure RDMA parameters (addr/size/fmt/swap).
3. Configure layer parameters (layer size/offset/roi size).

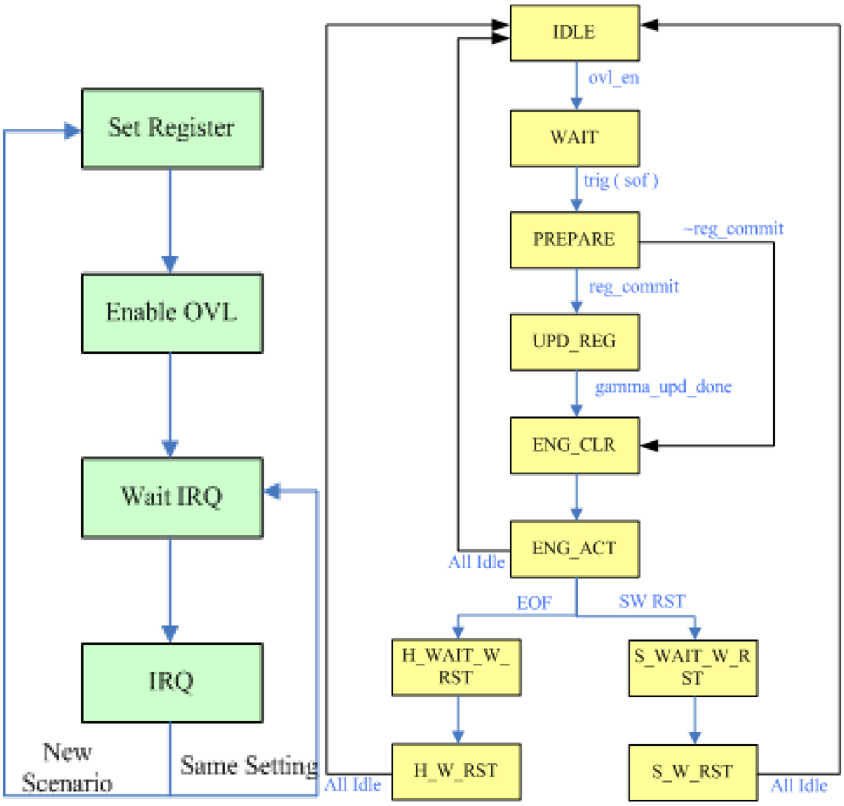


*Figure 6-31. Typical DISP_OVL programming procedure*

## 6.11 DIPS RDMA (Display Read Direct Memory Access)

### 6.11.1 Introduction

The RDMA engine is responsible of providing data to the interface engines, e.g. DSI, DBI and DPI. Because the interface engines need the real time service, the RDMA engine contains one line buffer to store the sufficient pixel data. It also detects the usage of data buffer to trigger the deep sleep mode of EMI.

### 6.11.2 Features

- Direct link input mode
- Memory input mode
  - Input format: YUYV422, UYVY422, YVYU422, UYVY422, RGB565, RGB888, ARGB8888
  - Input footprint: Raster-scan mode, 64 byte-aligned tile mode
  - Slow down mode
- Output control
  - Byte swap, RGB swap
  - Progressive mode, Interlace mode
  - Programmable YUV to RGB matrix
  - Non-stop output mode if the data buffer is under-running
- Buffer control
  - 240x16 byte data buffer (1280 pixels with RGB888 format)
  - Programmable request/pre-ultra/ultra control mechanism

### 6.11.3 Block Diagram

The following figure shows the detailed block diagram of RDMA engine. The clocks are automatically configured and one asynchronous FIFO is used for the output clock domain.

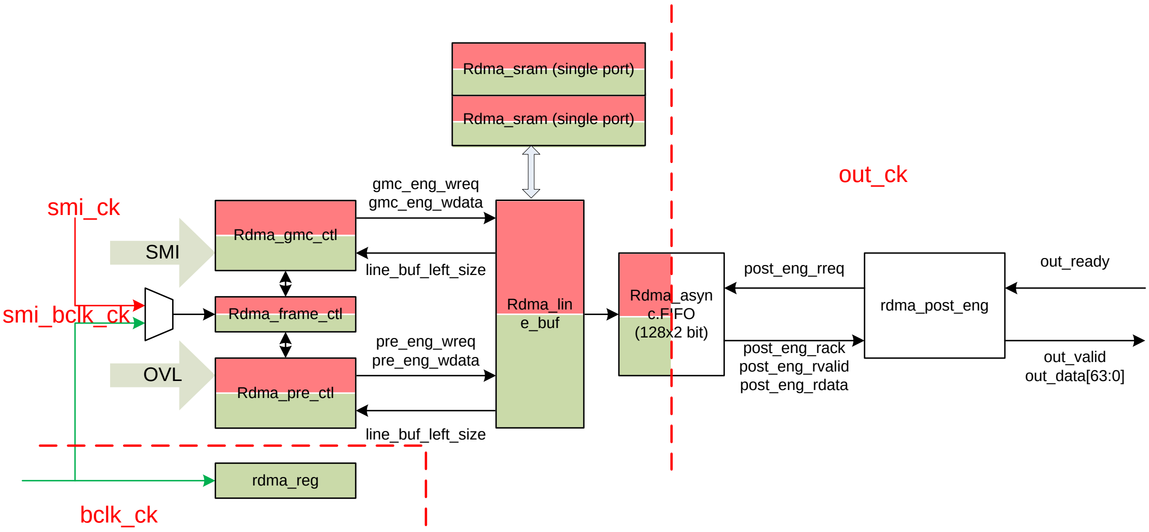


*Figure 6-32. Block diagram of RDMA engine*

## 6.11.4 Register Definition

See chapater 4.11 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 6.11.5 Programming Guide

The figure below shows the general programming sequence.

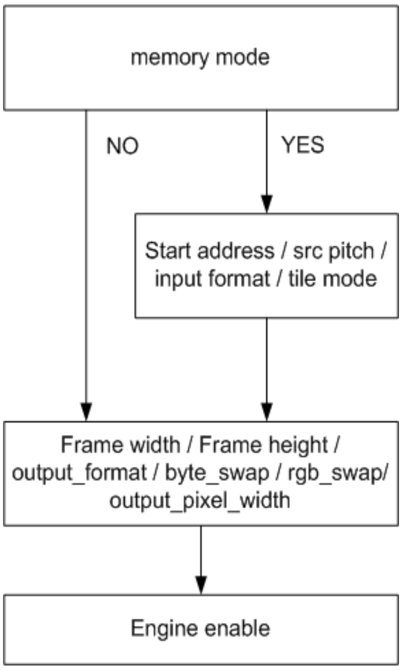

memory mode
NO
YES
Start address / src pitch / input format / tile mode
Frame width / Frame height / output_format / byte_swap / rgb_swap/ output_pixel_width
Engine enable

*Figure 6-33. General programming sequence*

The direct link mode can be easily configured by output frame width and height. On the other hand, the memory mode has several modes, which will be discussed in sub-sections.

#### 6.11.5.1 Memory Mode Control: Raster Scan Input, Progressive Output

1. Set 'MEM_MODE_START_ADDR' = Address of the first pixel.
2. Set 'MEM_MODE_SRC_PITCH' = Width of the source frame.

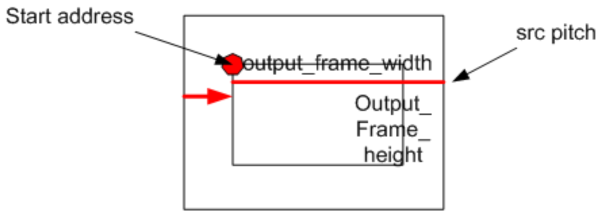


***Figure 6-34. Basic memory mode configuration***

#### 6.11.5.2 Memory Mode Control: Raster Scan Input, Interlace Output

1. Set 'MEM_MODE_START_ADDR' = Address of the first pixel.
2. Set 'MEM_MODE_SRC_PITCH' = ***Double*** of the width of the source frame.

#### 6.11.5.3 YUV to RGB Transfer Formula

$$\begin{pmatrix} \text{Y\_out} \\ \text{U\_out} \\ \text{V\_out} \end{pmatrix} = \begin{pmatrix} \text{C00} & \text{C01} & \text{C02} \\ \text{C10} & \text{C11} & \text{C12} \\ \text{C20} & \text{C21} & \text{C22} \end{pmatrix} \times \begin{pmatrix} \text{Y\_input + pre\_add\_0} \\ \text{U\_input + pre\_add\_1} \\ \text{V\_input + pre\_add\_2} \end{pmatrix} + \begin{pmatrix} \text{post\_add\_0} \\ \text{post\_add\_1} \\ \text{post\_add\_2} \end{pmatrix}$$

***Figure 6-35. Programmable color matrix***

#### 6.11.5.4 Byte Swap/RGB Swap

Set up the corresponding registers according to the data format on the memory.

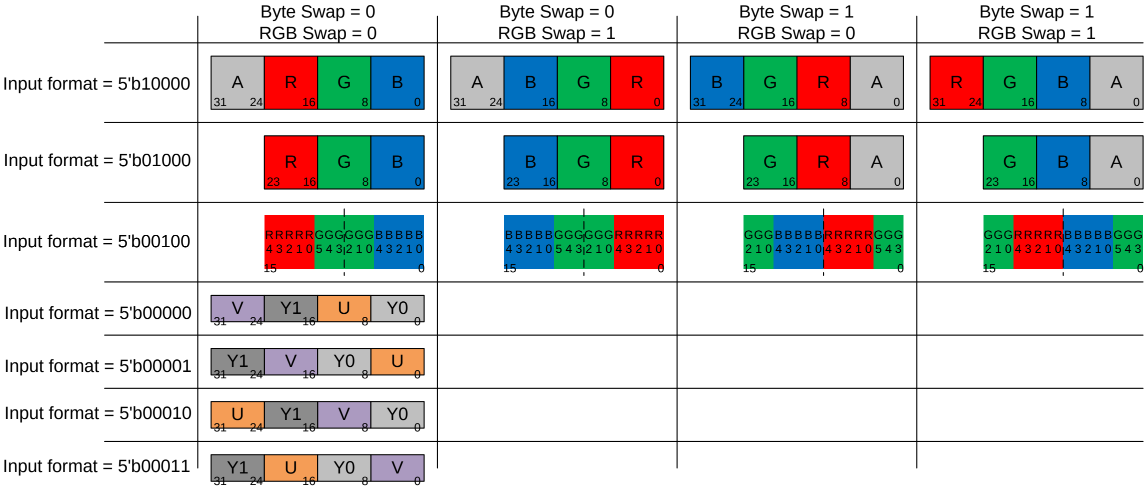


*Figure 6-36. Byte/RGB swap*

## 6.12 DISP_WDMA

### 6.12.1 Introduction

MDP_WDMA and DISP_WDMA have the same hardware architecture but slightly differ in SRAM size, which influences the line-width support and outstanding ability. WDMA does the job of DMA writing out the data in display/MDP pipeline into DRAM. In the following sections, WDMA is called DISP_WDMA.

### 6.12.2 Features

- Dither
- Programmable parameter color transform
- Input color format YUV444/RGB888
- Output format RGB565/RGB888/ARGB8888/UYVY/YV12/NV12/NV21
- 3-tap filter in horizontal and 2-tap filter in vertical for YUV420 down sample
- Byte swap/color swap/UV swap functions

### 6.12.3 Block Diagram

DISP_WDMA has a 256x128 two-port SRAM for DMA FIFO. One 320x64 single-port SRAM is for vertical filtering line_buffer (2560 pixels). For line-width larger than 2560, YUV420 vertical down sample must be drop-pixel scheme.

MDP_WDMA has a 128x128 two-port SRAM for DMA FIFO. One 64x64 single-port SRAM is for vertical filtering line_buffer (512 pixels). For line-width larger than 512, YUV420 vertical down sample must be drop-pixel scheme.

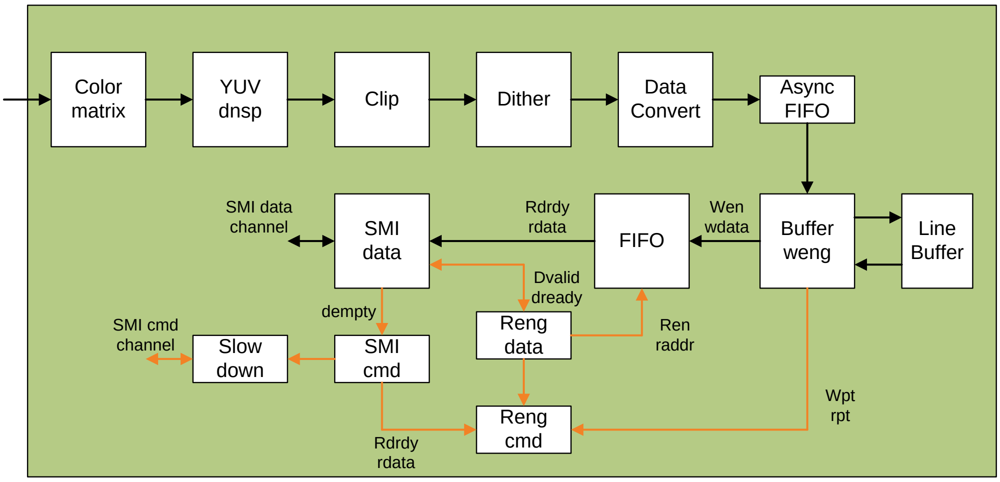


*Figure 6-37. Block diagram of DISP_WDMA*

### 6.12.4 Register Definition

See chapater 4.12 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

### 6.12.5 Programming Guide

The fundamental key parts are base address/input output format/data strides in memory. If color transform is needed, color transform matrix will also be needed. Other setting such as dither/filter are optional.

**MUST program**
WDMA_CFG
WDMA_SRC_SIZE
WDMA_CLIP_SIZE
WDMA_CLIP_COORD
WDMA_DST_ADDR0
WDMA_DST_W_IN_BYTE
WDMA_DST_ADDR1
WDMA_DST_ADDR2
WDMA_DST_UV_PITCH
WDMA_DST_ADDR_OFFSET0
WDMA_DST_ADDR_OFFSET1
WDMA_DST_ADDR_OFFSET2

The read result of WDMA_CT_DBG means the input counter of the WDMA.
The MSB part is line counter and the LSB part is pixel counter.
This can be used as debugging register.

The read result of WDMA_FLOW_CTRL_DBG means the current state of WDMA.

The state machine of the WDMA is as the following:

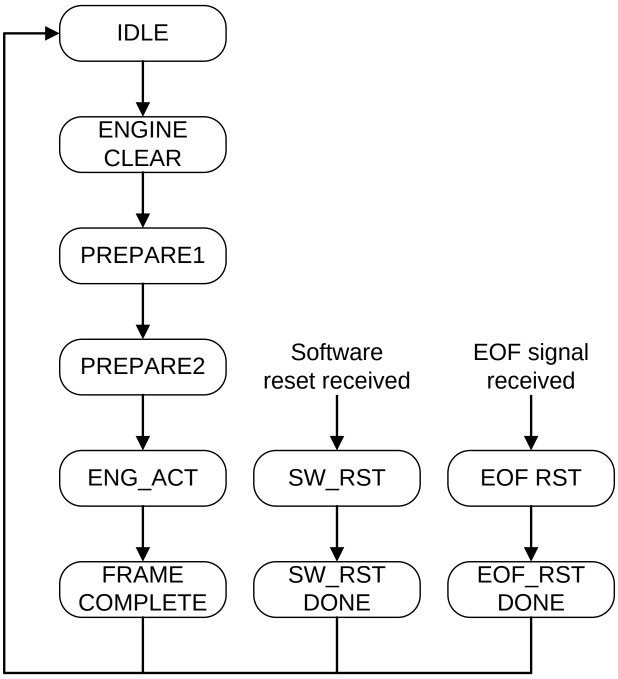


*Figure 6-38. Typical DISP_WDMA programming sequence*

## 6.13 Color Processor

### 6.13.1 Introduction

The color management engine is highly configurable and programmable; it fits different display panels and satisfies different user preferences for various purposes.

Color management is designed for two applications, getting better picture quality, and having one panel resemble the other in their output characteristics. With color processor, you can obtain better picture quality experience, and manufactures have identical and controllable production quality more easily.

A GUI tool is also provided as an easy and intuitive interface to color management.

### 6.13.2 Features

The color processor is very flexible for luma/saturation/hue adjustments. Main features are:

- Flexible architecture: PQ processing at many possible stages
- Input/output color space conversion
- Hue engine:
  - Partial hue: Modifies hue angle of specific hue phase
- Y engine:
  - Adaptive luma: Adaptively adjusts Y curve according to image content
  - Global contrast/brightness adjustment
  - Chroma boost: Compensates saturation value due to Y change
- Sat engine:
  - Partial S: Modifies saturation value of specific hue phase
  - Global saturation adjustment
- Histogram statistics: Includes Y histogram and chroma histogram

### 6.13.3 Block Diagram

The color engine provides various luma/saturation/hue adjustments. See Figure 6-39 for the block diagram.

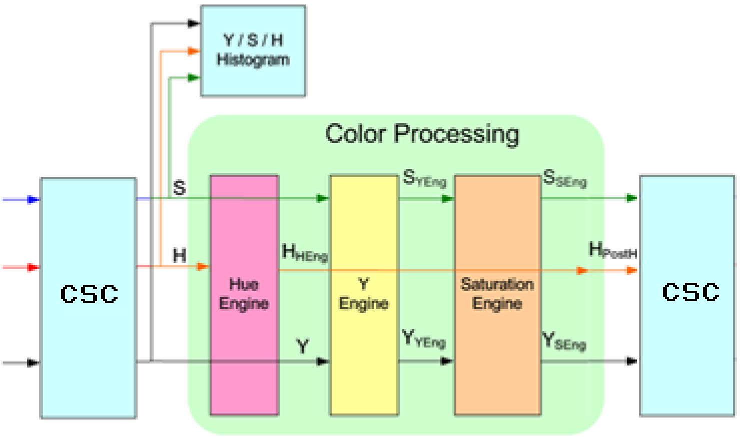


*Figure 6-39. Block diagram of color processor*

### 6.13.4 Register Definition

See chapater 4.13 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 6.13.5 Programming Guide

#### 6.13.5.1 Hue Adjustment

There are 20 hue phases for partial hue adjustment.

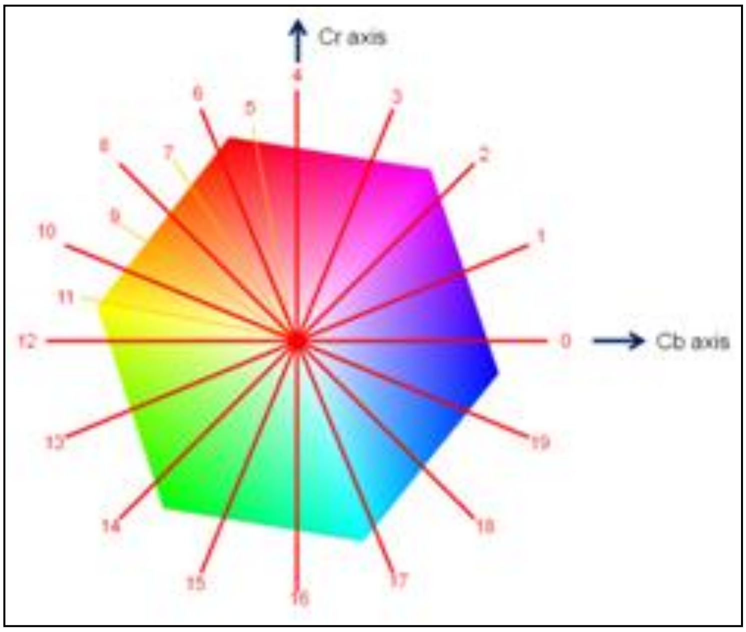


*Figure 6-40. 20 hue phases distribution*

Partial hue adjustment is achieved by HUE_TO_HUE_W_* registers. An example of partial hue adjustment is shown in Figure 6-41.

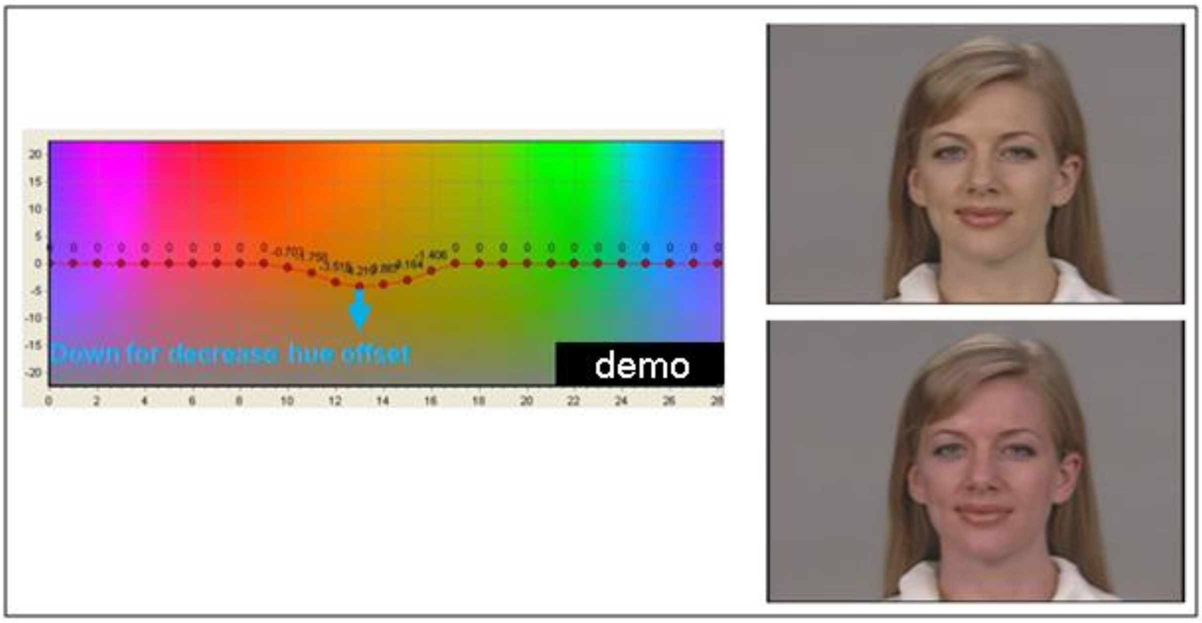


*Figure 6-41. Partial hue adjustment example*

### 6.13.5.2 Luma Adjustment

Global contrast and brightness adjustment are also provided for intuitive luma adjustment. The registers are G_CONTRAST and G_BRIGHTNESS (see Figure 6-42).

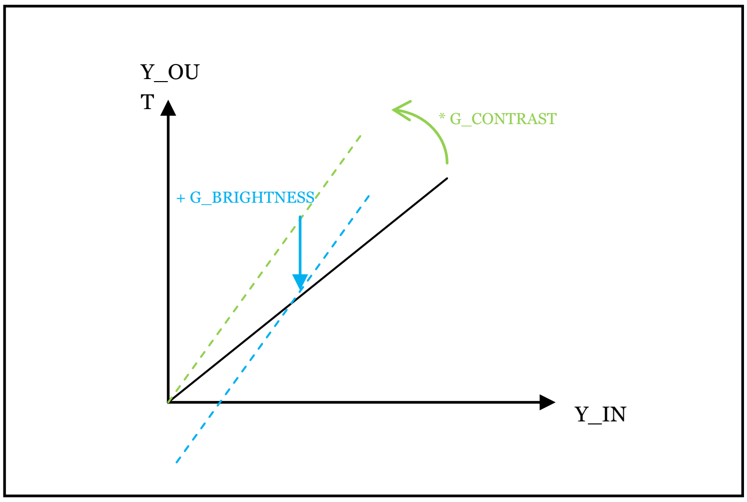


*Figure 6-42. Contrast/brightness adjustment example*

### 6.13.5.3 Saturation Adjustment

In human vision system, luminance increase results in pale images even the saturation is not decreased. Therefore, the color processor provides a “Chroma Boost” mechanism to detect the luminance increasing amount and decides the corresponding saturation boost level. The control registers are *_CBOOST_*, and the formula is illustrated in Figure 6-43.

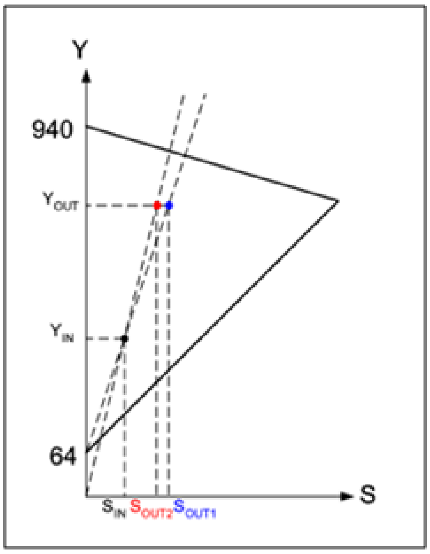


*Figure 6-43. Chroma boost demonstration*

For saturation adjustment, the color processor provides a flexible partial saturation adjustment. Similar to PartialHue, the hue plan is equally divided into 20 hue phases and saturation corresponding to each hue phase can be customized by three gains and two turning points. The adjustment for each hue phase are controlled by PARTIAL_SAT_GAIN*, PARTIAL_SAT_POINT*. Figure 6-44 shows the saturation adjustment in one hue phase.

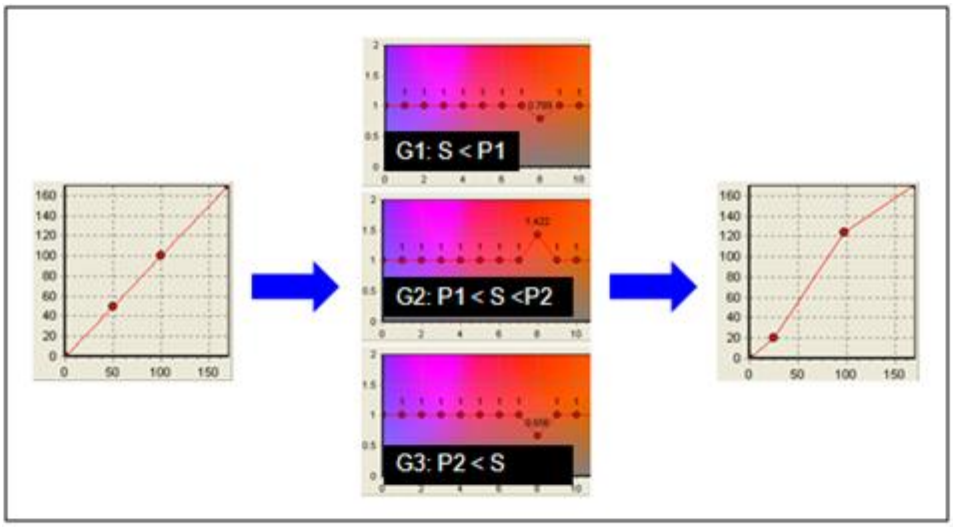


*Figure 6-44. Example of partial S adjustment for one hue phase*

A global saturation gain is provided for intuitive saturation adjustment. The control register is G_SATURATION (see Figure 6-45).

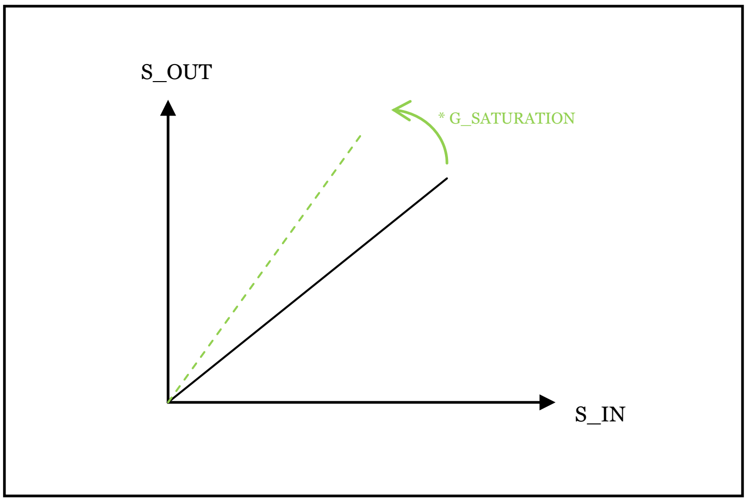


*Figure 6-45. Illustration of global saturation adjustment*

### 6.13.5.4 Color Matrix Conversion Programming

The color engine is equipped with input/output color matrix conversions. This makes possible that the color engine can operate in selectable processing stage. For example, the color engine can operate by RDMA engine's RGB or YCbCr outputs.

If RGB data are fed to the color engine, RGB to YCbCr conversion should be performed before color processing, and YCbCr to RGB conversion can be applied for the following engines.

# 6.14 DIPS CCORR (Display Color Correction Engine)

## 6.14.1 Introduction

The color correction engine changes the overall mixture of RGB colors to fit the characteristics of target panel.

## 6.14.2 Features

- Fixed-coefficient inverse gamma table with wide-gamut support
- Programmable 3X3 matrix
- Fixed-coefficient gamma table

## 6.14.3 Block Diagram

Figure 6-46 is the block diagram of color correction core.

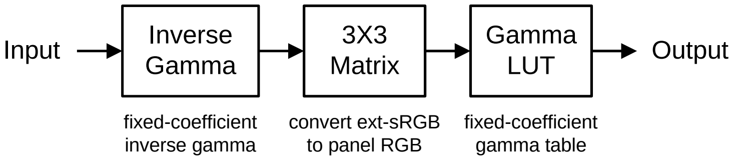
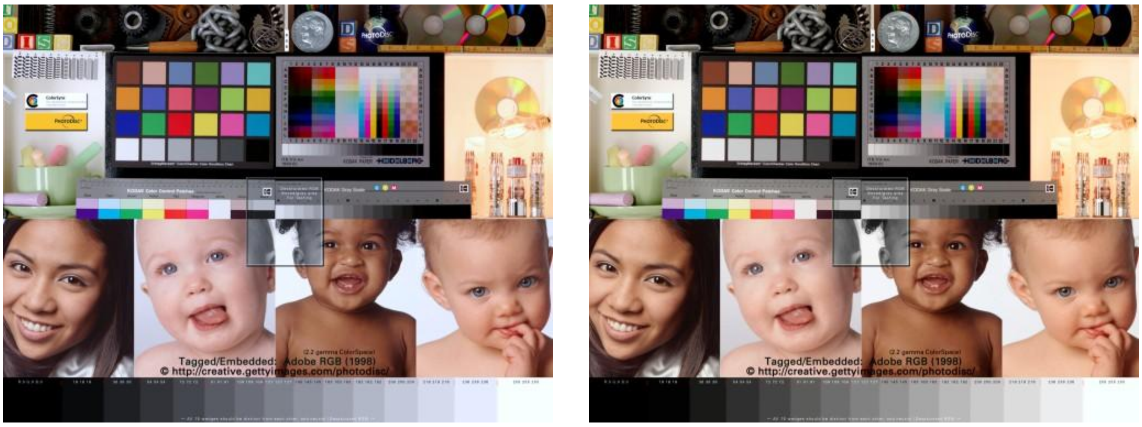

Input
Inverse Gamma
3X3 Matrix
Gamma LUT
Output
fixed-coefficient inverse gamma
convert ext-sRGB to panel RGB
fixed-coefficient gamma table

***Figure 6-46. Block diagram of color correction***

### 6.14.3.1 Color Correction

In order to display accurate image colors, the panel needs to match the standard sRGB color gamut. However, most LCD panels only display 65 to 80 percent of the sRGB color gamut. The OLED panels, on the other hand, can display over 130 percent of sRGB color gamut. With color correction, you can reproduce correct color on panels with different color gamuts.


Tagged/Embedded: Adobe RGB (1998)
© http://creative.gettyimages.com/photodisc/
Tagged/Embedded: Adobe RGB (1998)
© http://creative.gettyimages.com/photodisc/

Wide-gamut panel without color correction

Wide-gamut panel with color correction

***Figure 6-47. Visual effect of color correction***

## 6.14.4 Register Definition

See chapater 4.14 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 6.15 DIPS AAL (Display Adaptive Ambient Light Controller)

### 6.15.1 Introduction

The AAL processor, composed of content adaptive and ambient light adaptive, is responsible for backlight power saving and sunlight visibility improvement.

### 6.15.2 Features

- 33-bin weighted histogram
- DRE enhancement for sunlight visibility
- CABC compensation for backlight power saving

### 6.15.3 Block Diagram

Figure 6-48 is the block diagram of AAL processor.

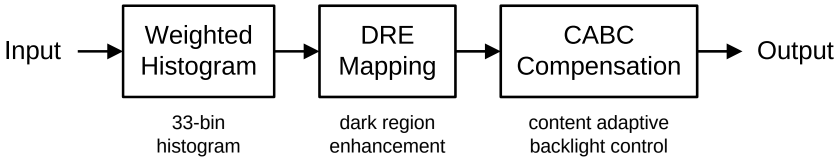
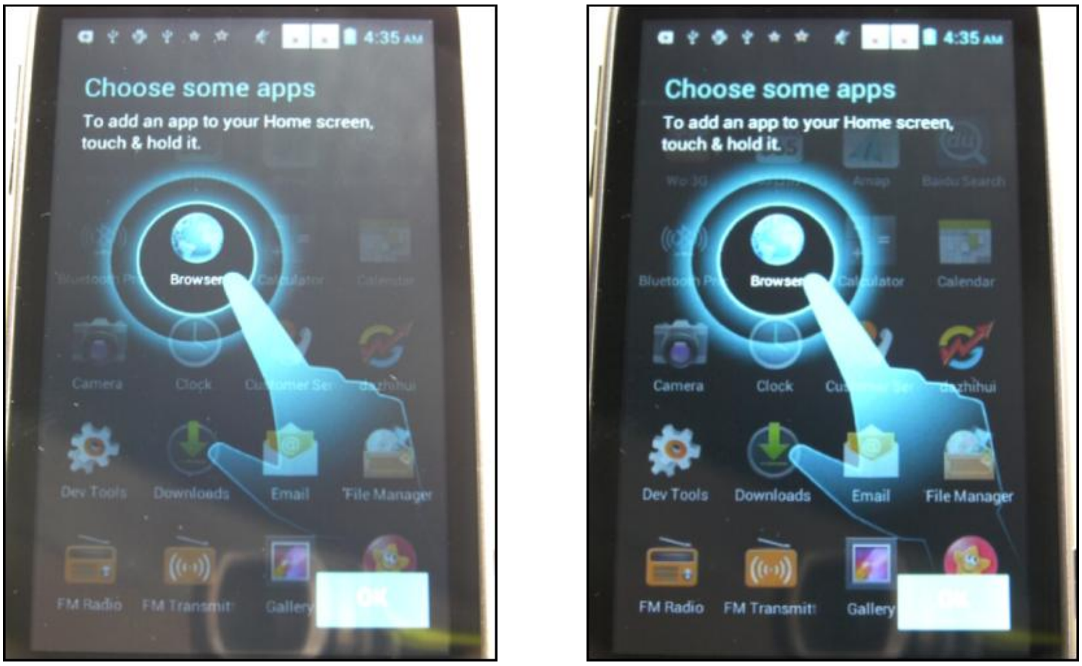


***Figure 6-48. Block diagram of AAL processor***

#### 6.15.3.1 DRE Enhancement

Dark region enhancement improve the visibility under sunlight.



Without DRE enhancement With DRE enhancement

***Figure 6-49. Visual effect of DRE enhancement***

## 6.15.4 Register Definition

See chapater 4.15 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 6.16 DIPS GAMMA (Display GAMMA Processing Engine)

### 6.16.1 Introduction

The GAMMA engine changes the overall mixture of RGB colors to fit the characteristics of target panel.

### 6.16.2 Features

- 10-bit gamma table with 512 entries
- Non-block gamma LUT programming

### 6.16.3 Gamma Correction

For accurate image reproduction, the transfer function of the panel should have a standard 2.20 gamma value. The gamma correction consists of three programmable look-up tables for RGB colors. It applies arbitrary mapping curve to compensate the incorrect transfer function of the panel.

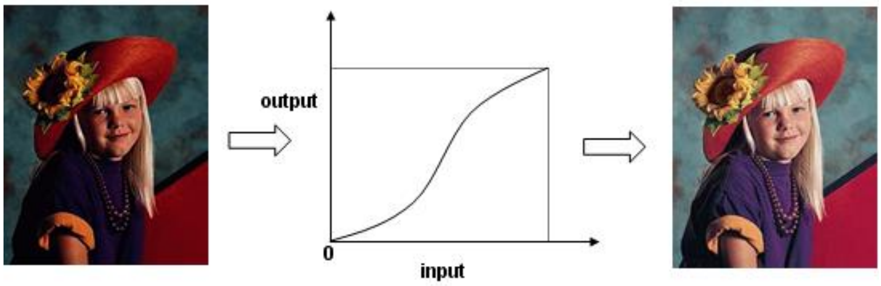


***Figure 6-50. Visual effect of gamma correction***

### 6.16.4 Register Definition

See chapater 4.16 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 6.17 DISP DITHER

### 6.17.1 Introduction

The dither engine decreases the RGB depth while migrates the loss of quality at the same time.

### 6.17.2 Register Definition

See chapater 4.17 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 6.18 DISPLAY PWM Generator

### 6.18.1 Introduction

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

### 6.18.2 Features

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

### 6.18.3 Register Definition

See chapater 4.18 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

### 6.18.4 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.

## 6.19 DPI (Digital Parallel Interface)

### 6.19.1 Introduction

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

### 6.19.2 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

### 6.19.3 Register Definition

See chapater 4.19 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 6.19.4 Programming Guide

#### 6.19.4.1 General Timing Programming

Programming DPI for one frame timing is easy. Figure 6-51 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.

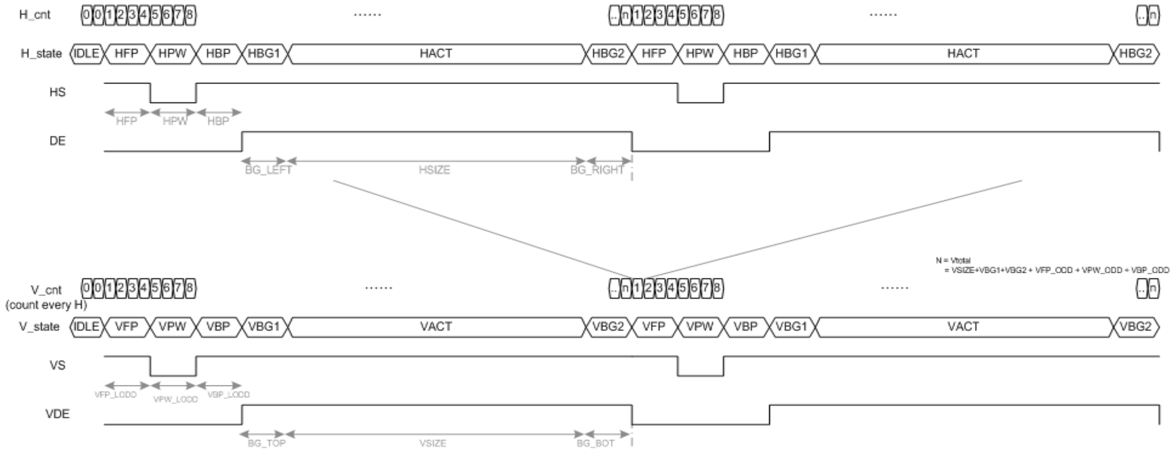


*Figure 6-51. Illustration of general timing relationship*

### 6.19.4.2 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 6-52 shows the data/clock timing and their corresponding settings in 4 cases.

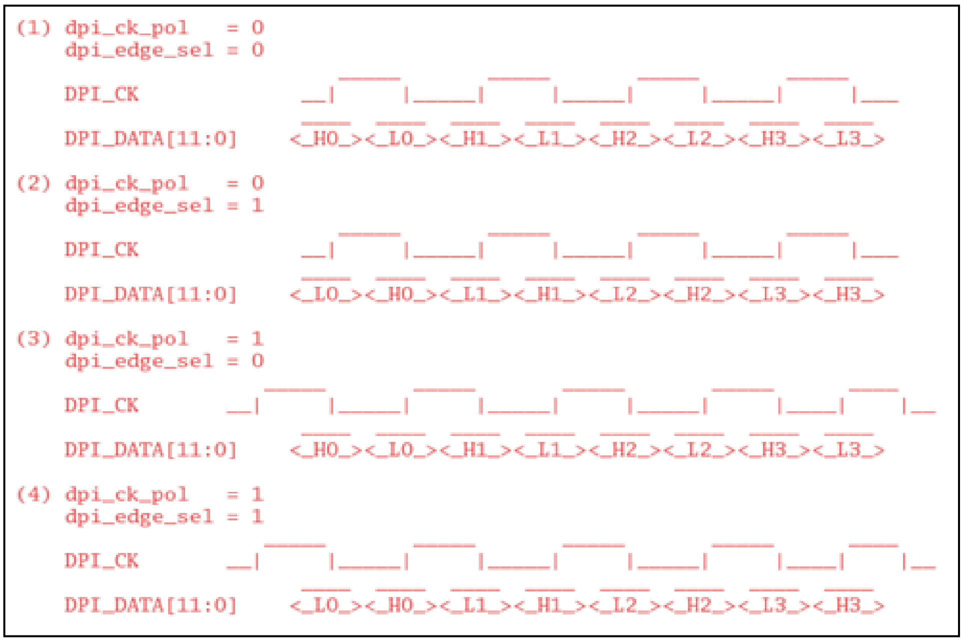


*Figure 6-52. Data/Clock timing*

Table 6-7 shows explicitly the data map of the 4 cases above.

*Table 6-7. 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 |

| | 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 |
| 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 |

### 6.19.4.3 Programming Flow

Figure 6-53 shows the DPI programming flow diagram. First, configure each timing register based on the target frame timing. Next, reset and enable DPI.

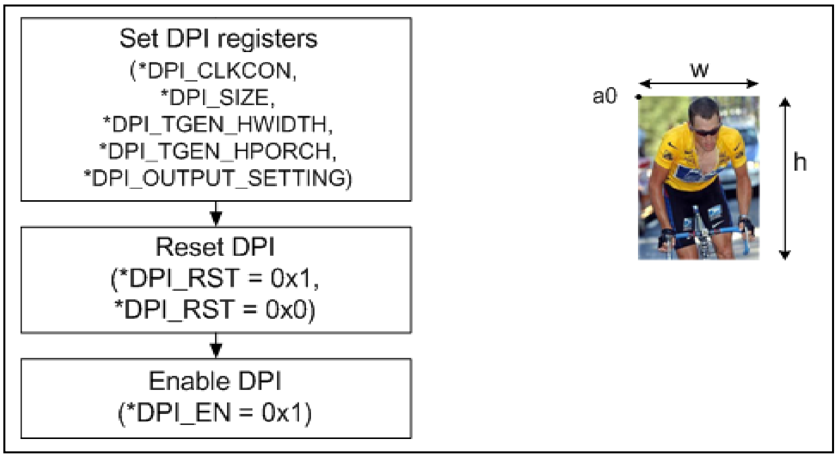


*Figure 6-53. Programming flow diagram*

## 6.20 Display Serial Interface

### 6.20.1 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.

### 6.20.2 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 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 frame compression with UFOE module
- Supports low frame-rate (LFR) technique

### 6.20.3 Register Definition

See chapater 4.20 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 6.20.4 Programming Guide

#### 6.20.4.1 Clock Control

After enabling PLL clock from MIPI DPHY macro and CG cells in MMSYS, a primary module clock enable shouble be set to turn on for the entire design (see Table 6-8). To disable clock for DSI, unset it. For more details on MIPI DPHY PLL clock control, please refer to the functional specification of MIPI TX Config module.

*Table 6-8. Sequence to enable module clock*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Enable primary module clock | W | DSI_COM_CON [1] | 1 |

### 6.20.4.2 DSI HS Clock Control

After enabling the module clock and properly setting up the related registers, it is necessary to enable the clock lane before starting high-speed data transmission by setting up the DSI_PHY_LCCON register. Please follow the steps shown in Table 6-9 to turn on the high-speed clock.

*Table 6-9. Sequence to enable high-speed clock*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Enable DSI high-speed clock | W | DSI_PHY_LCCON [0] | 1 |

To disable high-speed clock, follow steps in Table 6-10. Sequence of exiting ultra-low power mode on clock lane Note that the high-speed clock should be turned off before ultra-low power mode.

*Table 6-10. Sequence of exiting ultra-low power mode on clock lane*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Disable DSI high-speed clock | W | DSI_PHY_LCCON [0] | 0 |

### 6.20.4.3 DSI ULPS Enter Control

In cetain condition, the DISP subsys may be powered off after DSI entering the *ultra-low power state* (ULPS) to reduce more power consumption. ULPS is a protocol defined in MIPI spec and force LCM to enter standby scenario to save power. The sleep-in control in this module is shown in Table 6-11.

*Table 6-11. Sequence of sleep-in control (entering ultra-low power mode)*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Disable DSI high-speed clock | W | DSI_PHY_LCCON [0] | 0 |
| 2 | Data lane enter ultra-low power mode | W | DSI_PHY_LD0CON [1] | 1 |
| 3 | Clock lane enter ultra-low power mode | W | DSI_PHY_LCCON [1] | 1 |

### 6.20.4.4 DSI ULPS Exit Control

To wake up DSI and LCM from ULPM, executing an "ULPS-exit" procedure defined by MIPI spec is required. This procedure performs a protocol to get DP/DN signals from LP-00 through LP-10 to LP-11, where the LP-10 period should not be less than 1ms. The procedure can be easily controlled by the DSI sleep-out sequence with an interrupt indication in Table 6-12. Note that ULPS_WAKEUP_PRD should be changed according to different MIPI frequency to make sure wake-up time is not less than 1ms.

*Table 6-12. Sequence of sleep-out control (exiting ultra-low power mode)*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Enable sleep-out interrupt | W | DSI_INTEN [6] | 1 |
| 2 | Configure corresponding cycle counts for ULPS wakeup period according to current MIPI frequency (ex. 1Gbps) | W | DSI_TIME_CON0 [15:0] | 0x80 |
| 3 | Recover lane number (4-lane) | W | DSI_TXRX_CON [5:2] | 0xF |
| 4 | Sleep-out start | W | DSI_START [2] | 1 |
| 5 | Issue interrupt and sleep-out done | R | DSI_INTSTA [6] | 1 |
| 6 | Clear interrupt | W | DSI_INTSTA [6] | 1 |

### 6.20.4.5 DPHY Timing Control

All of the timing parameters defined in MIPI DPHY should be properly written in the DSI registers for correct timeing control. The writtern value is based on the DSI internal clock cycle period which is related to DPHY PLL clock settings through MIPI TX Config engine.

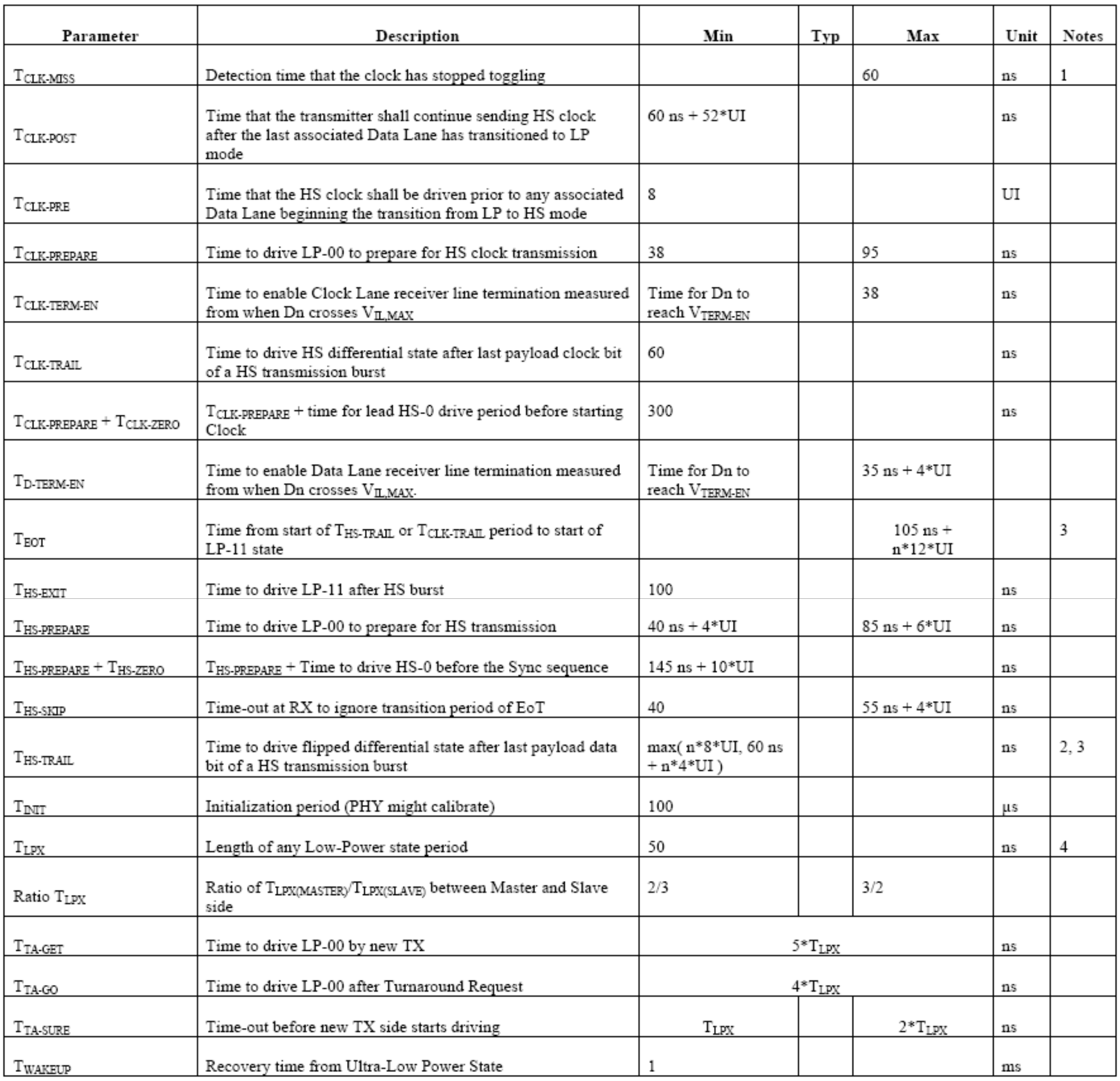
For example, the timing parameter $T_{HS\text{-}PREPARE}$ is mandatory to be between 40ns+4*UI and 85ns +6*UI, where UI means time interval, equal to the duration of any HS state on clock lane. If the clock lane is set to 500MHz frequency, as well as bit-rate 1Gbps, the UI should be 1ns. In other words, the value of $T_{HS\text{-}PREPARE}$ must between 44 ~ 91ns. The internal DSI clock is 4x divided by clock lane, as well as 125MHz. To satisfy $T_{HS\text{-}PREPARE}$, the register value DA_HS_PREP should be 6 ~ 11 (see Table 6-13).

*Table 6-13. DPHY timing parameters register settings*

| | Timing specification | Absolute time for UI: 1ns | DA_HS_PREP value |
|---|---|---|---|
| $T_{HS\text{-}PREPARE}$ | 40ns+4*UI ~ 85ns +6*UI | 44 ~ 91 ns | 6 ~ 11 |

Table 6-14 lists the timing parameters which should be configured in the DSI registers. Note that for different bit-rate requirements, the UI values are various. For more precise timing control, select DPHY clock as fast as possible. However, the faster DPHY clock is set, the more the power waste. A suitable clock is beneficial to the optimization of system power consumption.

*Table 6-14. DPHY global operation timing parameter defined by MIPI spec*



| Parameter | Description | Min | Typ | Max | Unit | Notes |
|---|---|---|---|---|---|---|
| $T_{CLK-MISS}$ | Detection time that the clock has stopped toggling | | | 60 | ns | 1 |
| $T_{CLK-POST}$ | Time that the transmitter shall continue sending HS clock after the last associated Data Lane has transitioned to LP mode | 60 ns + 52*UI | | | ns | |
| $T_{CLK-PRE}$ | Time that the HS clock shall be driven prior to any associated Data Lane beginning the transition from LP to HS mode | 8 | | | UI | |
| $T_{CLK-PREPARE}$ | Time to drive LP-00 to prepare for HS clock transmission | 38 | | 95 | ns | |
| $T_{CLK-TERM-EN}$ | Time to enable Clock Lane receiver line termination measured from when Dn crosses $V_{IL,MAX}$ | Time for Dn to reach $V_{TERM-EN}$ | | 38 | ns | |
| $T_{CLK-TRAIL}$ | Time to drive HS differential state after last payload clock bit of a HS transmission burst | 60 | | | ns | |
| $T_{CLK-PREPARE}$ + $T_{CLK-ZERO}$ | $T_{CLK-PREPARE}$ + time for lead HS-0 drive period before starting Clock | 300 | | | ns | |
| $T_{D-TERM-EN}$ | Time to enable Data Lane receiver line termination measured from when Dn crosses $V_{IL,MAX}$. | Time for Dn to reach $V_{TERM-EN}$ | | 35 ns + 4*UI | | |
| $T_{EOT}$ | Time from start of $T_{HS-TRAIL}$ or $T_{CLK-TRAIL}$ period to start of LP-11 state | | | 105 ns + n*12*UI | | 3 |
| $T_{HS-EXIT}$ | Time to drive LP-11 after HS burst | 100 | | | ns | |
| $T_{HS-PREPARE}$ | Time to drive LP-00 to prepare for HS transmission | 40 ns + 4*UI | | 85 ns + 6*UI | ns | |
| $T_{HS-PREPARE}$ + $T_{HS-ZERO}$ | $T_{HS-PREPARE}$ + Time to drive HS-0 before the Sync sequence | 145 ns + 10*UI | | | ns | |
| $T_{HS-SKIP}$ | Time-out at RX to ignore transition period of EoT | 40 | | 55 ns + 4*UI | ns | |
| $T_{HS-TRAIL}$ | Time to drive flipped differential state after last payload data bit of a HS transmission burst | max( n*8*UI, 60 ns + n*4*UI ) | | | ns | 2, 3 |
| $T_{INIT}$ | Initialization period (PHY might calibrate) | 100 | | | µs | |
| $T_{LPX}$ | Length of any Low-Power state period | 50 | | | ns | 4 |
| Ratio $T_{LPX}$ | Ratio of $T_{LPX(MASTER)}/T_{LPX(SLAVE)}$ between Master and Slave side | 2/3 | | 3/2 | | |
| $T_{TA-GET}$ | Time to drive LP-00 by new TX | 5*$T_{LPX}$ | | | ns | |
| $T_{TA-GO}$ | Time to drive LP-00 after Turnaround Request | 4*$T_{LPX}$ | | | ns | |
| $T_{TA-SURE}$ | Time-out before new TX side starts driving | $T_{LPX}$ | | 2*$T_{LPX}$ | ns | |
| $T_{WAKEUP}$ | Recovery time from Ultra-Low Power State | 1 | | | ms | |

The registers of timing parameters for data lanes and clock lane are illustrated in Figure 6-54 and Figure 6-55 respectivly. The registers for BTA (Bus turn-around) timing are illustrated in Figure 6-56.

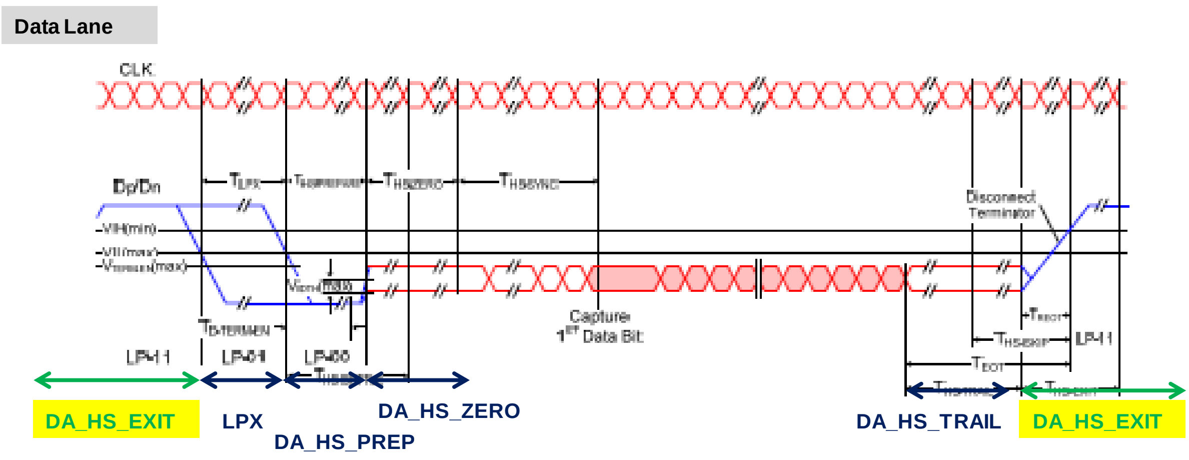


*Figure 6-54. Registers for data lane timing parameters*

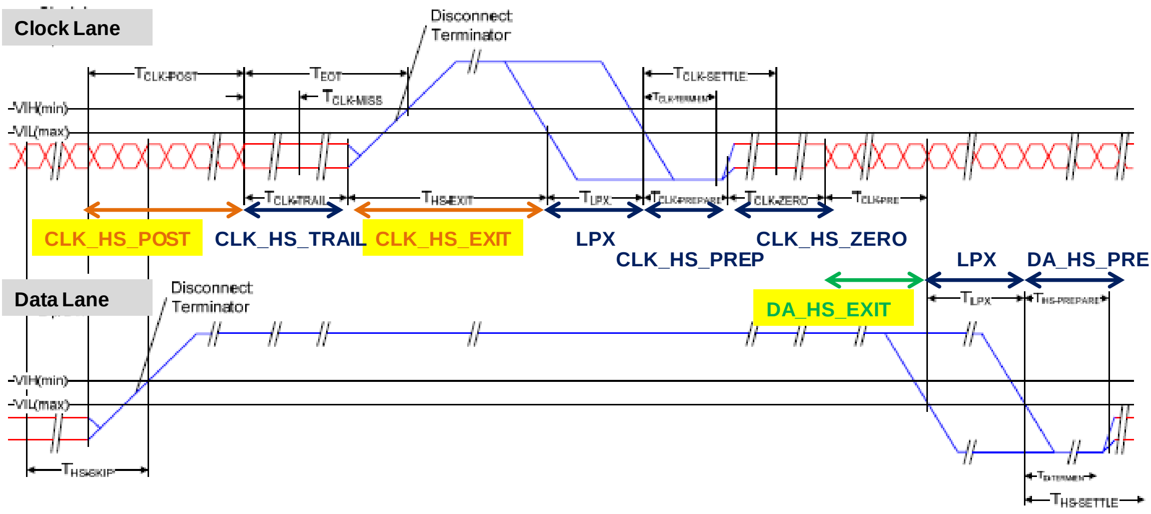


*Figure 6-55. Registers for clock lane timing paramaters*

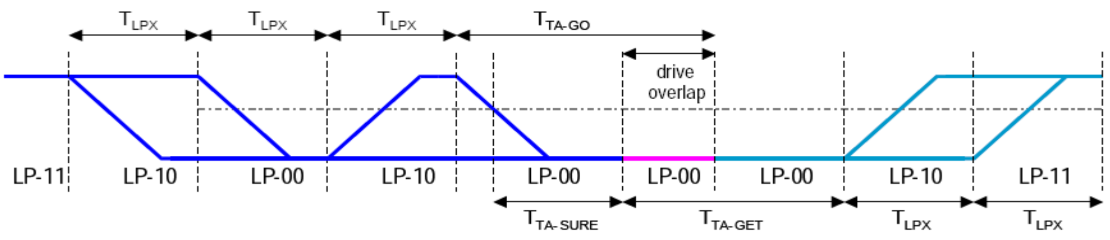


*Figure 6-56. Register for BTA timing parameters*

### 6.20.4.6 Command Mode

DSI supports command mode transmission through writing commands to a dedicated command queue. By configuring commands and triggering, the transmission can be executed sequentially.

#### 6.20.4.6.1 Command Queue

DSI has a dedicated command queue that are 32-bit wide and up to 128-entry deep, as shown in Figure 6-57. To simplify the settings for transmitting a packet in the command mode, the command queue is designed to categorize all possible transmission types and commands into four primary instructions and unifies all DSI specification commands into one or several 32-bit wide instructions. Figure 6-57 also illustrates a 32-bit instruction structure with the instruction format of CONFG byte.

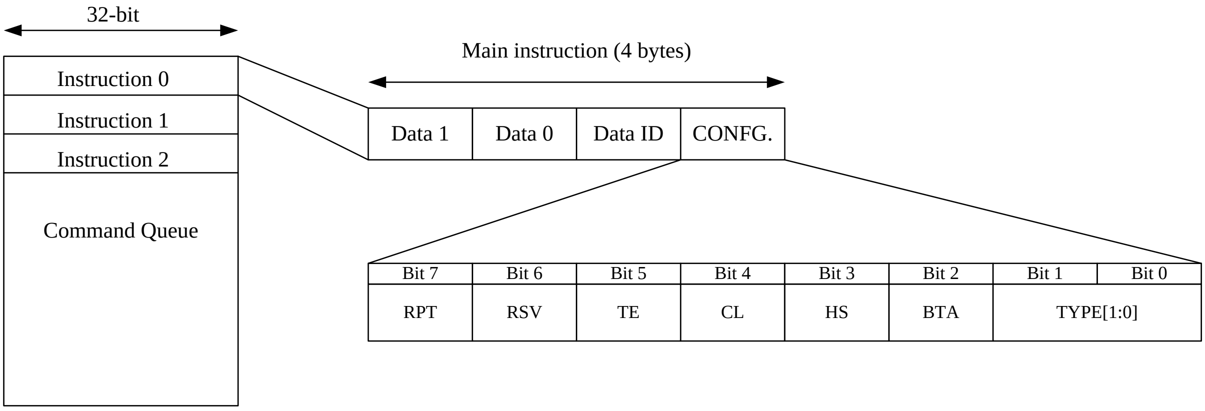


*Figure 6-57. DSI command queue instruction format*

Table 6-15 shows the descriptions of the CONFG byte to a instruction. For convenience's sake, virtual channel 0 is used for all packets in the following examples. Software programmers should take charge of the real virtual channel numbers to the slave devices.

*Table 6-15. Config. field description of main instruction*

| | Value | Function | Description |
|---|---|---|---|
| Type[1:0] | 00 | - | Used for DSI short packet read/write command |
| | 01 | - | Used for DSI frame buffer write command (long packet) |
| | 10 | - | Used for DSI generic long packet write command |
| | 11 | - | Used for DSI frame buffer read command (short packet) |
| BTA | 0 | Off | Turns around the DSI link after the DSI command is transmitted |
| | 1 | On | |
| HS | 0 | Off | Enables HS Tx transmission for this packet; otherwise transmit packet via LP Tx |
| | 1 | On | |
| CL | 0 | 8-bit | Selects command length for frame buffer read/write instruction. Only effective for type 1 and type 3 instructions. |
| | 1 | 16-bt | |

| | Value | Function | Description |
|---|---|---|---|
| TE | 0 | Off | Enables TE request. Will only turn around the DSI link without any packet transmission |
| | 1 | On | |
| Resv | - | - | Reserved for further use |
| Resv | - | - | Reserved for further use |

#### 6.20.4.6.2 Type-0 Instruction

Type-0 instruction is used to transmit short packets. Figure 6-58 lists the formats of type-0 instruction where (Data ID + Data 0 + Data 1) is constructed by a DSI short packet command (without ECC).

| Byte 3 | Byte 2 | Byte 1 | Byte 0 |
|---|---|---|---|
| Data 1 | Data 0 | Data ID | CONFG. |

***Figure 6-58. Type-0 instruction format***

Suppose we are to send "Turn On Peripheral" and "Color Mode On" commands which are transmitted via LP Tx and HS Tx respectively and request slave response after the second command finished, the descriptions can be translated into two 32-bit instructions and achieved by the steps illustrated in Table 6-16.

***Table 6-16. Type-0 Tx example***

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Fill command queue entry-0 | W | DSI_CMDQ_0 | 0x0000120C |
| 2 | Fill command queue entry-1 | W | DSI_CMDQ_1 | 0x00003200 |
| 3 | Set command count | W | DSI_CMDQ_CON | 0x2 |
| 4 | Start command | W | DSI_START [0] | 0x1 |
| 5 | Issue interrupt and receive slave response | R | DSI_INTSTA [0] | 0x1 |
| 6 | Clear interrupt status | W | DSI_INTSTA [0] | 0x1 |
| 7 | Read trigger status (Acknowledge) | R | DSI_RX_TRIG_STA [3:0] | 0x4 |
| 8 | Reponse read ack to module and go to next commends in queue | W | DSI_RX_RACK [0] | 0x1 |
| 9 | Issue interrupt and command done | R | DSI_INTSTA [1] | 0x1 |
| 10 | Clear interrupt status | W | DSI_INTSTA [1] | 0x1 |

#### 6.20.4.6.3 Type-1 Instruction

Type-1 command is used to write data into the frame buffer. As shown in Figure 6-59, there are 4 bytes constructing this type of instruction where Mem_start_0 and Mem_start_1 can be generic commands defined by slave vendors or DCS commands. Mem_start_1 is optional, i.e. the memory start/continue command can be single-byte as DCS defines. To indicate whether the DSI controller sends Mem_start_1 or not depends on the CL bit of the CONFG. byte. Since the length of frame butter to be updated is not a constant, this type of instruction may send several long packets to the slave. The payload data and length of each packet (excluding mem_start_0 and mem_start_1) is prepared by the RDMA controller which couples the output of image data path or layer overlay result to the DSI controller. For the first packet, mem_start_0 and mem_start_1 (if CL = 1) are used as the parameters to inform the slave that the host is starting to write the frame buffer. For the remaining packets, the register value MEM_CONTI[15:0] will be used as the parameters to inform the slave side to write these data following the last pixel of the previous packet. For more flexibility, Mem_start_0, Mem_start_1, MEM_CONTI[15:0] and CL are all programmable. However, it consumes only one entry of command queue.

The user needs to set up two registers to define the packet length and packet count for a frame-based type-1 transmission. Frame width in bytes should be set to DSI_PSCON, and frame height in lines should be set to DSI_VACT_NL, respectively. These two registers are used in both video and command mode frame data transmission.

| Byte 3 | Byte 2 | Byte 1 | Byte 0 |
|---|---|---|---|
| Mem start 1<br>(optional) | Mem start 0 | Data ID | CONFG. |

***Figure 6-59. Type-1 instruction format***

Follow the example in Table 6-17 to write the frame buffer via DCS command in the HS Tx mode.

***Table 6-17. Type-1 Tx example***

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Fill command queue entry-0 | W | DSI_CMDQ_0 | 0x002C3909 |
| 2 | Set command count | W | DSI_CMDQ_CON | 0x1 |
| 3 | Set memory continue command | W | DSI_MEM_CONTI [15:0] | 0x3C |
| 4 | Start command | W | DSI_START [0] | 0x1 |
| 5 | Issue interrupt and command done | R | DSI_INTSTA [1] | 0x1 |
| 6 | Clear interrupt status | W | DSI_INTSTA [1] | 0x1 |

#### 6.20.4.6.4 Type-2 Instruction

Type-2 instruction is used to send a long packet. As shown in Figure 6-60, this type of main instruction requires several sub-instructions which do not have CONFIG. To send a type-2 command, the user needs to write a CONFIG with TYPE = 2 and packet header information (Data ID + 2 byte word count) to enrty 0, and a series of data bytes in size of word count to the following entries, excluding ECC and checksum. The bytes in following entries will be treated as long packet data instead of the next instruction until the word count size is reached. The command queue count should be set as the number of multiple entries used.

The type-2 command is sent in LPTX mode due to memory latency in reading sub-instruction data. Besides, type-2 command should be sent individually without the next instruction followed. For the 32-entry command queue, the maximum word count for long packet is 124 bytes.

| Byte 3 | Byte 2 | Byte 1 | Byte 0 |
|---|---|---|---|
| WC 1 | WC 0 | Data ID | CONFG. |
| Data 3 | Data 2 | Data 1 | Data 0 |
| | | Data WC-1 | Data WC-2 |

***Figure 6-60. Type-2 instruction format***

See Table 6-18 below for the example of sending 3 parameters (0x33, 0x22, 0x11) by a generic long packet command with 3-byte word count.

***Table 6-18. Type-2 Tx example***

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Fill command queue entry-0 (data ID and word count) | W | DSI_CMDQ_0 | 0x00032902 |
| 2 | Fill command queue entry-1 (parameters) | W | DSI_CMDQ_1 | 0x00112233 |
| 3 | Set command count | W | DSI_CMDQ_CON | 0x2 |
| 4 | Start command | W | DSI_START [0] | 0x1 |
| 5 | Issue interrupt and command done | R | DSI_INTSTA [1] | 0x1 |
| 6 | Clear interrupt status | W | DSI_INTSTA [1] | 0x1 |

Note that the RPT bit of CONFIG. is designed for this type of instruction. It is useful for the NULL packet or blanking packet. For example, if a null packet is to be sent, only the main instruction (entry-0 of command queue) will be needed, the following payload data will be sent as “0”.

#### 6.20.4.6.5 Type-3 Instruction

Type-3 instruction is used for reading frame buffer. As shown in Figure 6-61, the format is the same as that of type-1. When this instruction is executed, the host will first send a short packet with memory start parameter given in byte 2 and byte 3 and automatically issue the next packet by memory continue parameters programmed in MEM_CONTI[15:0]. The number of total packets required to be sent depends on the FRM_BC and “maximum return packet size”. For example, if we are to read 1,024 bytes from the frame buffer in the slave, and the “maximum return packet size” is set to “4”, after the first short packet described in main instruction is sent, there will be another 255 short packets with memory continue parameters to be sent successively.

| Byte 3 | Byte 2 | Byte 1 | Byte 0 |
|---|---|---|---|
| Men start 1 (optional) | Mem start 0 | Data ID | CONFG. |

*Figure 6-61. Type-3 instruction format*

See Table 6-19 for the example of using type-3 instruction to perform the frame buffer read.

*Table 6-19. Type-3 Tx example*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Fill command queue entry-0 | W | DSI_CMDQ_0 | 0x002E0603 |
| 2 | Set command count | W | DSI_CMDQ_CON | 0x1 |
| 3 | Set memory read continue command | W | DSI_MEM_CONTI[15:0] | 0x3E |
| 4 | Start command | W | DSI_START [0] | 0x1 |
| 5 | Issue interrupt and receive slave response | R | DSI_INTSTA [0] | 0x1 |
| 6 | Read receive data bytes | R | DSI_RX_DATA03 | - |
| 7 | Clear interrupt status | W | DSI_INTSTA [0] | 0x1 |
| 8 | Reponse read ack to module and go to next commends in queue | W | DSI_RX_RACK [0] | 0x1 |
| 9 | Issue interrupt and command done | R | DSI_INTSTA [1] | 0x1 |
| 10 | Clear interrupt status | W | DSI_INTSTA [1] | 0x1 |

#### 6.20.4.6.6 Command Mode Status Debug Register

Sometimes an inproper configuration settings to DSI, DISP paths and MUTEX may cause hanged issues on DSI because command mode control needs to wait for memory data from DMA in previous stages. As a result, a debugging registers DSI_STATE_DBG6 is designed for users to observe internal states of DSI controller in order to reduce debugging efforts. Table 6-20 shows the status mapping to the register bit [14:0]. The user can check the on-hot bit to find incorrect settings.

*Table 6-20. Command mode status in debugging register*

| Bit | Value | Description |
|---|---|---|
| 0 | 0x0001 | Idle (wait command) |
| 1 | 0x0002 | Reads command queue for packet header |
| 2 | 0x0004 | Sends type-0 command |
| 3 | 0x0008 | Waits for data from the previous module to send type-1 command |
| 4 | 0x0010 | Sends type-1 command |
| 5 | 0x0020 | Sends type-2 command |
| 6 | 0x0040 | Reads command queue for paket data |
| 7 | 0x0080 | Sends type-3 command |
| 8 | 0x0100 | Sends BTA |
| 9 | 0x0200 | Waits for RX-read data |

| Bit | Value | Description |
|---|---|---|
| 10 | 0x0400 | Waits for software RACK for RX-read data |
| 11 | 0x0800 | Waits for peripheral TE |
| 12 | 0x1000 | Gets TE signaling |
| 13 | 0x2000 | Waits for software RACK for peripheral TE |
| 14 | 0x4000 | Waits for external TE |

### 6.20.4.7 Video Mode

DSI supports video mode traffic sequences, including sync pulse mode, sync event mode and burst mode. To facilitate the translation of the parameters of packets, see the timing diagrams below for the corresponding register settings.

#### 6.20.4.7.1 Sync-Pulse Mode

A non-burst sync-pulse mode enables the periphera to accurately reconstructoriginal video timing, including sync pulse widths. A timing diagram for sync-pulse mode is shown in Figure 6-62, which also shows registers mappings to lines of VSA/VBP/VACT/VFP periods

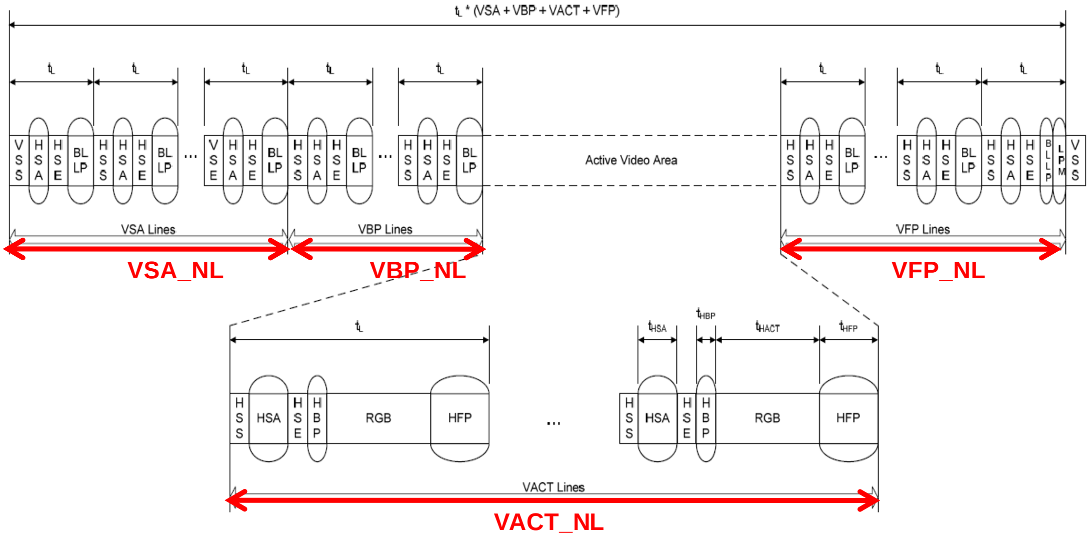


*Figure 6-62. Non-burst transmission: sync-pulse mode*

A detailed timing diagram for one line period is shown in Figure 6-63. In this diagram, the user needs to base on real timing parameters of HPW/HBP/HFP and pixel format to figure out the correct value to registers. In addition, a slight adjustment for HFP_WC should also be performed because it needs to leave some space time for HS preparation time due to non-continuous data lane transmission. The formulas for these registers are shown in Figure 6-64.

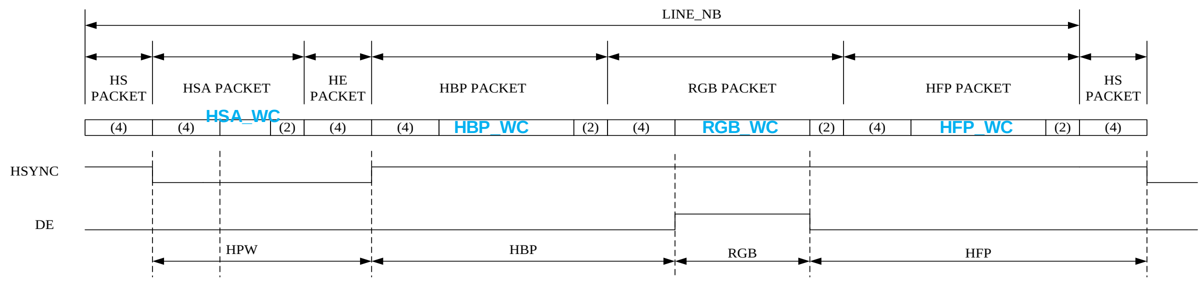


*Figure 6-63. Sync-pulse mode line period*

- HSA_WC = HPW * BPP - 10
- HBP_WC = HBP * BPP - 10
- RGB_WC = active_pixel * BPP
- HFP_WC = HFP * BPP - 12 - data_init_cycle * lane_num
- data_init_cycle = T_hs-exit + T_lpx + T_hs-prep + T_hs-zero

*Figure 6-64. Sync-pulse mode word-count parameters*

#### 6.20.4.7.2 Sync-Event Mode

A non-burst sync-event mode is similar to the pulse-sync mode, but accurate reconstruction of sync pulse widths is not required. Therefore, a single sync event is substituted. The timing diagram is shown in Figure 6-65.

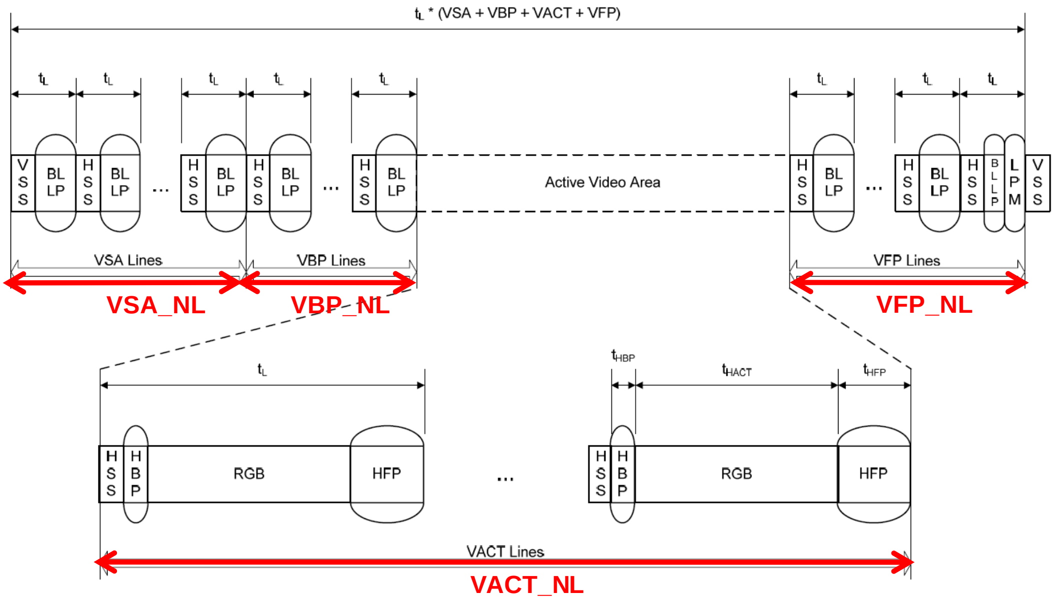


*Figure 6-65. Non-burst transmission: Sync-event mode*

A detailed timing diagram for one line period is shown in Figure 6-66, and the register settings of word count is listed in Figure 6-67.

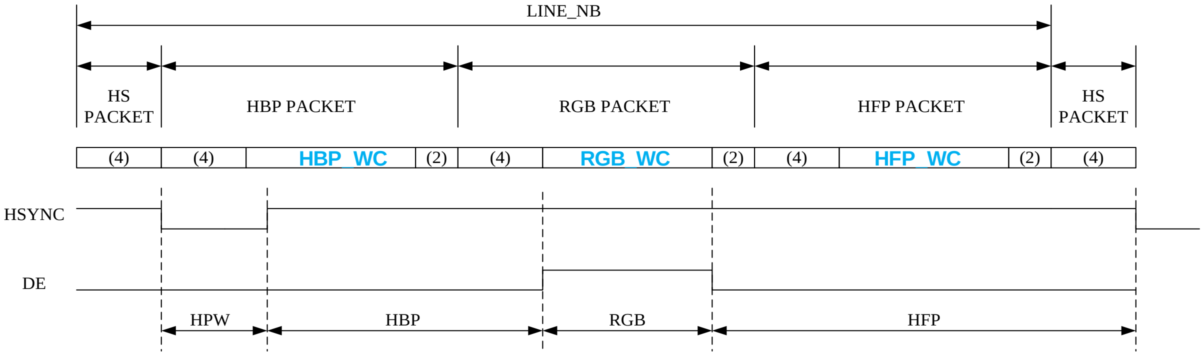
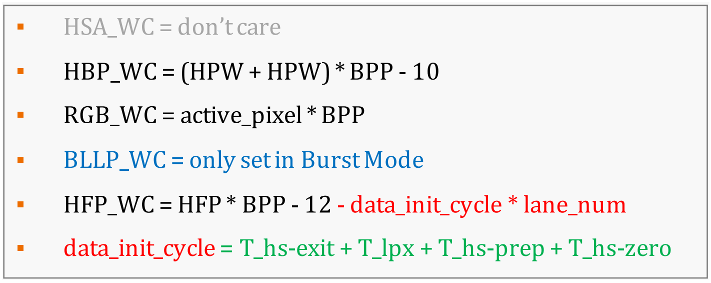


*Figure 6-66. Sync-event mode line period*



- HSA_WC = don't care
- HBP_WC = (HPW + HPW) * BPP - 10
- RGB_WC = active_pixel * BPP
- BLLP_WC = only set in Burst Mode
- HFP_WC = HFP * BPP - 12 - data_init_cycle * lane_num
- data_init_cycle = T_hs-exit + T_lpx + T_hs-prep + T_hs-zero

*Figure 6-67. Sync-event mode word-count parameters*

### 6.20.4.7.3 Burst Mode

A burst mode let RGB pixel packets time-compressed, leaving more time during a scan line for LP mode to save power or for multiplexing other transmissions onto DSI link. The timing diagram is shown in Figure 6-65.

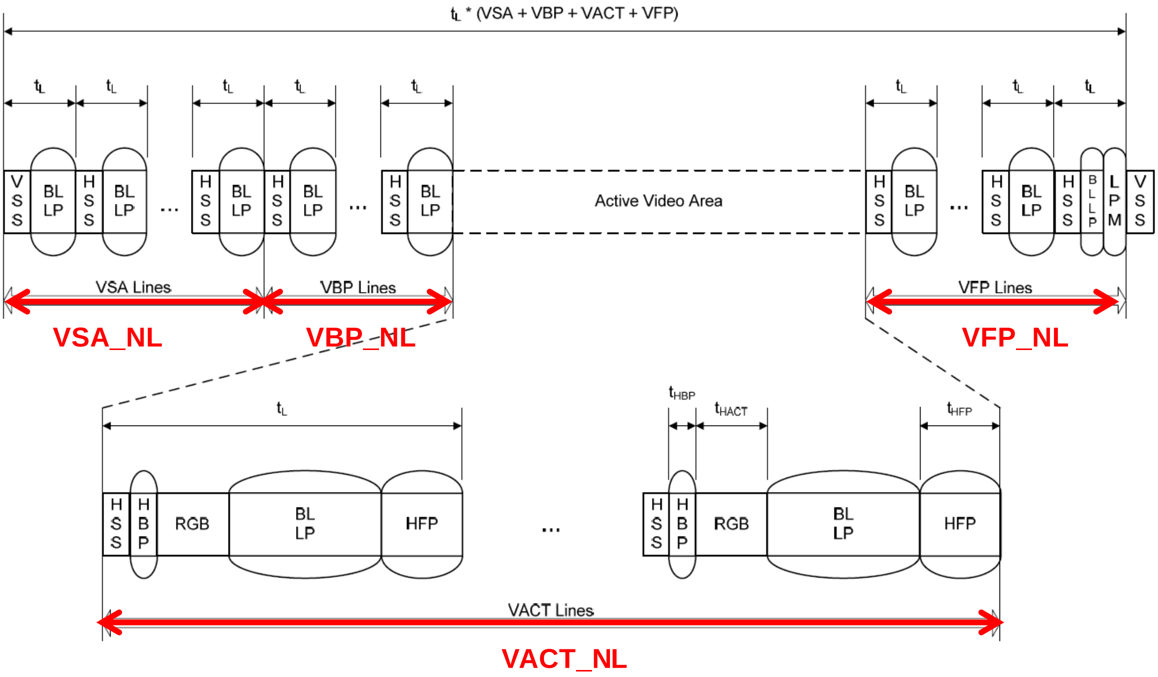


*Figure 6-68. Burst mode transmission*

Detailed timing diagram for one line period of burst mode is almost the same as sync-event mode shown in Figure 6-66 and Figure 6-67 except that the BLLC_WC register should be set.

### 6.20.4.8 Peripheral TE Detection

#### 6.20.4.8.1 TE Signaling

DSI has ability to receive peripheral TE (Tearing Effect) signals via BTA process. Before starting to receive TE signals from the peripheral, make sure a DCS command of “set_tear_on” is sent and register configuration of TE is enabled in the peripheral to avoid TE hanged issue. Here is an example in Table 6-21 to show how to trigger TE commands by command queue.

*Table 6-21. Example of TE signaling detection*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Fill command queue : DCS “set_tear_on” | W | DSI_CMDQ_0 | 0x00351500 |
| 2 | Fill command queue : get TE | W | DSI_CMDQ_1 | 0x00000020 |
| 3 | Fill command queue : type-1 command | W | DSI_CMDQ_2 | 0x002C3909 |
| 4 | Set command count | W | DSI_CMDQ_CON | 0x3 |
| 5 | Set memory write continue command | W | DSI_MEM_CONTI[15:0] | 0x3C |
| 6 | Start command | W | DSI_START [0] | 0x1 |
| 7 | Issue interrupt and receive TE | R | DSI_INTSTA [2] | 0x1 |

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 8 | Read trigger status (TE) | R | DSI_RX_TRIG_STA [3:0] | 0x2 |
| 9 | Clear interrupt status | W | DSI_INTSTA [2] | 0x1 |
| 10 | Reponse read ack to module and go to next commends in queue | W | DSI_RX_RACK [0] | 0x1 |
| 11 | Issue interrupt and command done | R | DSI_INTSTA [1] | 0x1 |
| 12 | Clear interrupt status | W | DSI_INTSTA [1] | 0x1 |

#### 6.20.4.8.2 External TE Pin

In certain case we may use external TE pin instead of TE signals for some reasons. DSI supports such mechanism to issue external TE interrupt signals. Refer to the sequences shown in Table 6-22 for detailed control information.

*Table 6-22. Example of external TE pin detection*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Enable external TE interrupt | W | DSI_INTEN [4] | 0x1 |
| 2 | Enable external TE | W | DSI_TXRX_CON [10] | 0x1 |
| 3 | Select external TE polarity (active-high) | W | DSI_TXRX_CON [9] | 0x0 |
| 4 | Issue interrupt and receive external TE | R | DSI_INTSTA [4] | 0x1 |
| 5 | Clear interrupt status | W | DSI_INTSTA [4] | 0x1 |

### 6.20.4.9 Video Mode Extra Packet Transmission

DSI supports sending out an extra short/long packet during video mode transmission. This operation must use a set of special registers instead of the original command queue. The single extra packet is sent once and immediately when the user triggers the VM_CMD_START bit of register DSI_START. Refer to the registers with prefix of "DSI_VM_CMD" for more detailed descriptions.

#### 6.20.4.9.1 Short Packet

To transmit an extra short packet in the video mode, follow the example illustrated in Table 6-23.

*Table 6-23. Example of short packet transmission in video mode*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Enable vm-cmd function | W | DSI_VM_CMD_CON [0] | 0x1 |
| 2 | Select short packet to be sent | W | DSI_VM_CMD_CON [1] | 0x0 |
| 3 | Enable sending period (VBP, VFP) | W | DSI_VM_CMD_CON [5:3] | 0x6 |
| 4 | Fill short packet ID and bytes 0~1 | W | DSI_VM_CMD_CON [31:8] | 0x001535 |
| 5 | Enable vm-cmd done interrupt | W | DSI_INTEN [5] | 0x1 |
| 6 | Start vm-cmd transmission | W | DSI_START [16] | 0x1 |

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 7 | Issue interrupt and vm-cmd transmission done | R | DSI_INTSTA [5] | 0x1 |
| 8 | Clear interrupt status | W | DSI_INTSTA [5] | 0x1 |

#### 6.20.4.9.2 Long Packet

To transmit an extra long packet in the video mode, follow the example illustrated in Table 6-24.

***Table 6-24. Example of long packet transmission in video mode***

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Enable vm-cmd function | W | DSI_VM_CMD_CON [0] | 0x1 |
| 2 | Select long packet to be sent | W | DSI_VM_CMD_CON [1] | 0x1 |
| 3 | Enable sending period (VSA, VBP, VFP) | W | DSI_VM_CMD_CON [5:3] | 0x7 |
| 4 | Fill long packet ID and word count | W | DSI_VM_CMD_CON [31:8] | 0x000739 |
| 5 | Fill long packet data byte 0~3 | W | DSI_VM_CMD_DATA0 | 0x332211FF |
| 6 | Fill long packet data byte 4~6 | W | DSI_VM_CMD_DATA1 | 0x00665544 |
| 7 | Enable vm-cmd done interrupt | W | DSI_INTEN [5] | 0x1 |
| 8 | Start vm-cmd transmission | W | DSI_START [16] | 0x1 |
| 9 | Issue interrupt and vm-cmd transmission done | R | DSI_INTSTA [5] | 0x1 |
| 10 | Clear interrupt status | W | DSI_INTSTA [5] | 0x1 |

# 6.21 MIPI TX Configuration Module

## 6.21.1 Introduction

The MIPI TX configuration module, as well as MIPI_TX_Config described in the following sections, is used to control MIPI TX related registers for MIPI DPHY macro. This analog macro includes design of SDM PLL, bandgap, LDO core, lane control, GPI pads, output enable, etc. It provides MIPI DSI high-speed clock up to 1.5Gbps, as well as 768MHz and provides primary clocks to DSI module.

## 6.21.2 Features

The following functions of MIPI DPHY can be controlled by this module.

- Bandgap control
- LDO core power and configuration
- SDM PLL configuration
- SSC control
- Analog function related settings and status
- Output pads control and electronic features
- GPI pads control
- Software-control mode for each lanes

## 6.21.3 Register Definition

See chapater 4.21 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 6.21.4 Programming Guide

### 6.21.4.1 MIPI PLL Initial Sequence

Figure 6-69 illustrates the waveform of the initial sequence to enable power of MIPI DPHY macro and initialize MIPI PLL. The user should follow the steps described in Table 6-23 to perform waveform behavior for MIPI DPHY control. In this example, 520MHz clock is set.

If Efuse option is realized, please read the Efuse results and write to RG_DSI_BG_R1_TRIM[3:0] & RG_DSI_BG_R2_TRIM[3:0] before RG_DSI_BG_CORE_EN is asserted.

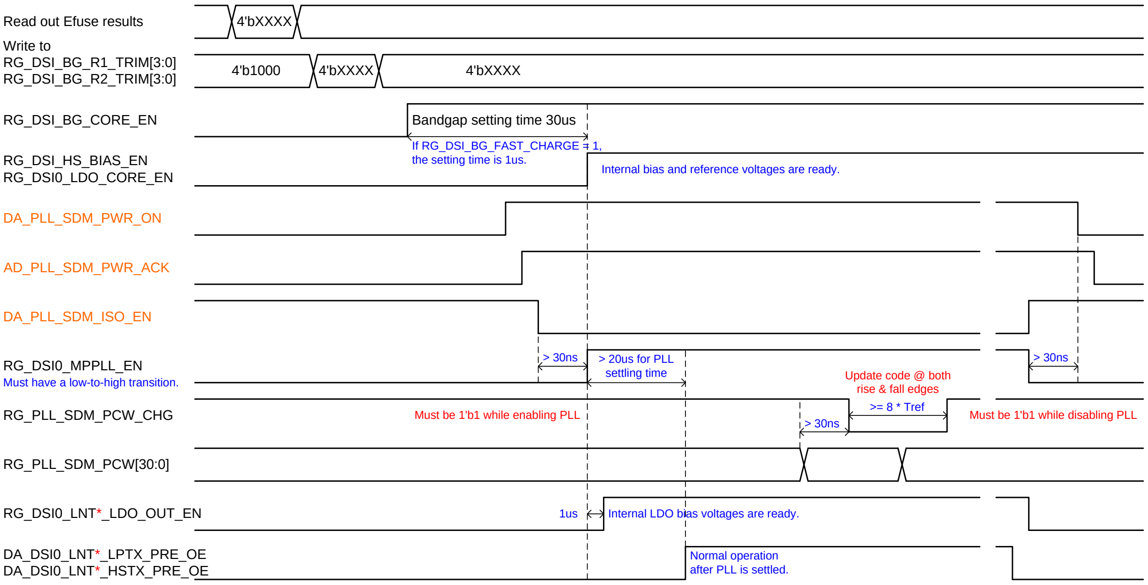


*Figure 6-69. Initial waveform of MIPI PLL*

*Table 6-25. MIPI PLL initial sequence*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Enable 26MHz reference clock | W | AP_PLL_CON0 [6]<br>(0x10209000) | 1 |
| 2 | Enable BG core power and clock | W | DSI_BG_CON [1:0] | 3 |
| 3 | Wait for 30 us | - | | |
| 4 | Enable HS bias | W | DSI_TOP_CON [1] | 1 |
| 5 | Enable LDO core power and CKG | W | DSI_CON [1:0] | 3 |
| 6 | Enable PLL power | W | DSI_PLL_PWR [0] | 1 |
| 7 | Disable PLL isolation | W | DSI_PLL_PWR [1] | 0 |
| 8 | Set divider control for PLL (no division) | W | DSI_PLL_CON0 [9:1] | 0 |
| 9 | Set SSC control for PLL (no SSC) | W | DSI_PLL_CON1 | 1 |
| 10 | Set PLL frequency (512MHz) | W | DSI_PLL_CON2 | 0x50000000 |
| 11 | Enable clock lane | W | DSI_CLOCK_LANE [0] | 1 |
| 12 | Enable data lane 0 | W | DSI_DATA_LANE_0 [0] | 1 |
| 13 | Enable data lane 1 | W | DSI_DATA_LANE_1 [0] | 1 |
| 14 | Enable data lane 2 | W | DSI_DATA_LANE_2 [0] | 1 |
| 15 | Enable data lane 3 | W | DSI_DATA_LANE_3 [0] | 1 |
| 16 | Enable PLL | W | DSI_PLL_CON0 [0] | 1 |
| 17 | Wait for 20 us | - | | |
| 18 | Disable pad tie low | W | DSI_TOP_CON [11] | 0 |

#### 6.21.4.2 MIPI PLL De-initial Sequence

For power-off of MIPI PLL for sleep-in or suspend purpose, follow the steps listed in Table 6-26. Be sure to make all registers to default settings to avoid unexpected power consumption.

*Table 6-26. MIPI PLL de-initial sequence*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Disable PLL | W | DSI_PLL_CON0 [0] | 0 |
| 2 | Enable pad tie low | W | DSI_TOP_CON [11] | 1 |
| 3 | Disable clock lane | W | DSI_CLOCK_LANE [0] | 0 |
| 4 | Disable data lane 0 | W | DSI_DATA_LANE_0 [0] | 0 |
| 5 | Disable data lane 1 | W | DSI_DATA_LANE_1 [0] | 0 |
| 6 | Disable data lane 2 | W | DSI_DATA_LANE_2 [0] | 0 |
| 7 | Disable data lane 3 (if it exists) | W | DSI_DATA_LANE_3 [0] | 0 |
| 8 | Enable PLL isolation | W | DSI_PLL_PWR [1] | 1 |
| 9 | Disable PLL power | W | DSI_PLL_PWR [0] | 0 |
| 10 | Disable HS bias | W | DSI_TOP_CON [1] | 0 |
| 11 | Disable LDO core power and CG | W | DSI_CON [1:0] | 0 |
| 12 | Disable BG core power and clock | W | DSI_BG_CON [1:0] | 0 |
| 13 | Reset divider control for PLL | W | DSI_PLL_CON0 [9:1] | 0 |
| 14 | Reset SSC control for PLL | W | DSI_PLL_CON1 | 0 |
| 15 | Reset PLL frequency (default value) | W | DSI_PLL_CON2 | 0x50000000 |
| 16 | Disable 26MHz reference clock | W | AP_PLL_CON0 [6]<br>(0x10209000) | 0 |

#### 6.21.4.3 Suspend/Resume Control

If the system enters suspend mode, DSI module should be operated to enter the ultra-low power state (ULPS) before execution of the sequence shown in Table 6-26. When the system needs to resume from suspend mode, the sequence shown in Table 6-23 should be performed first followed by the ULPS-exit procedure to return the normal output status.

#### 6.21.4.4 Software Control Mode

The software control mode allows user control output DP/DN pads directly. It is usually used for test or signal measurement. To enable output signals, it needs to setup PRE_OE and OE bits under this mode. Table 6-27 is an example of software control mode operation which sets all output pads to low voltage.

*Table 6-27. Software control mode example*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Set up clock lane PRE_OE/OE signal | W | DSI_SW_CTRL_CON0 [1:0] | 0x3 |

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 2 | Set up Data Lane 0 PRE_OE/OE signal | W | DSI_SW_CTRL_CON0 [8:7] | 0x3 |
| 3 | Set up Data Lane 1 PRE_OE/OE signal | W | DSI_SW_CTRL_CON0 [15:14] | 0x3 |
| 4 | Set up Data Lane 2 PRE_OE/OE signal | W | DSI_SW_CTRL_CON0 [21:20] | 0x3 |
| 5 | Set up Data Lane 3 PRE_OE/OE signal | W | DSI_SW_CTRL_CON0 [27:26] | 0x3 |
| 6 | Enable softwate control mode | W | DSI_SW_CTRL_EN | 1 |

#### 6.21.4.5 MIPI PLL SSC

MIPI PLL supports SSC (Spread-Spectrum Control) with fractional précised frequency multiplication. To enable SSC function, the user needs to set up the SSC_EN bit of register DSI_PLL_CON1 and other proper settings. Table 6-28 is an example of initial sequence with SSC for 0 ~ -4% and frequency of 208MHz.

*Table 6-28. MIPI PLL initial sequence with SSC enable*

| Step | Description | R/W | Register [bit] | Value |
|---|---|---|---|---|
| 1 | Enable BG core power and clock | W | DSI_BG_CON [1:0] | 3 |
| 2 | Wait for 30 us | - | | |
| 3 | Enable HS bias | W | DSI_TOP_CON [1] | 1 |
| 4 | Enable LDO core power and CG | W | DSI_CON [1:0] | 3 |
| 5 | Enable PLL power | W | DSI_PLL_PWR [0] | 1 |
| 6 | Disable PLL isolation | W | DSI_PLL_PWR [1] | 0 |
| 7 | Set up TXDIV0 divider (divided by 2) | W | DSI_PLL_CON0 [4:3] | 1 |
| 8 | Set up SSC fraction enable | W | DSI_PLL_CON1 [0] | 1 |
| 9 | Set up SSC phase initial (downward) | W | DSI_PLL_CON1 [1] | 1 |
| 10 | Set up SSC period | W | DSI_PLL_CON1 [31:16] | 0x01B1 |
| 11 | Set up PLL frequency | W | DSI_PLL_CON2 | 0x40000000 |
| 12 | Set up SSC amplitude | W | DSI_PLL_CON3 | 0x07900790 |
| 13 | Enable clock lane | W | DSI_CLOCK_LANE [0] | 1 |
| 14 | Enable Data Lane 0 | W | DSI_DATA_LANE_0 [0] | 1 |
| 15 | Enable Data Lane 1 | W | DSI_DATA_LANE_1 [0] | 1 |
| 16 | Enable Data Lane 2 | W | DSI_DATA_LANE_2 [0] | 1 |
| 17 | Enable Data Lane 3 | W | DSI_DATA_LANE_3 [0] | 1 |
| 18 | Enable PLL | W | DSI_PLL_CON0 [0] | 1 |
| 19 | Wait for 20 us | - | | |
| 20 | Enable SSC | W | DSI_PLL_CON1 [2] | 1 |
| 21 | Disable pad tie low | W | DSI_TOP_CON [11] | 0 |

## 6.22 JPEG Encoder

### 6.22.1 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 6-70 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.

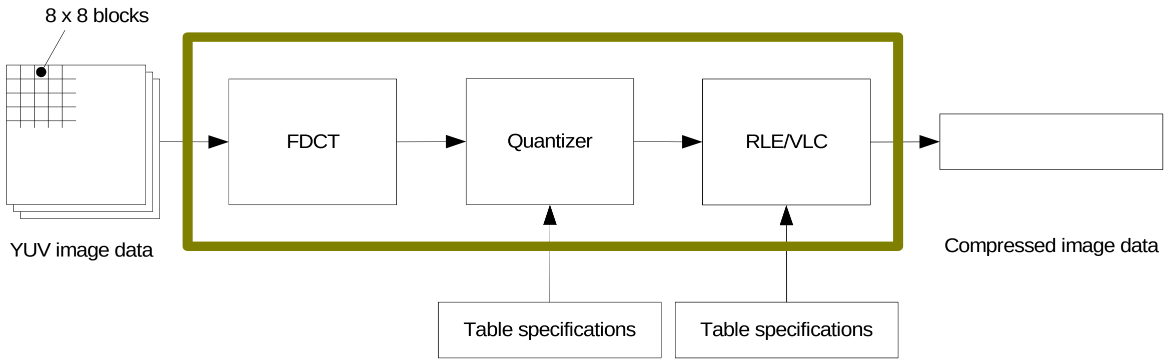


*Figure 6-70. Procedure of JPEG encoder*

### 6.22.2 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.

#### 6.22.2.1 Software Reset Mechanism

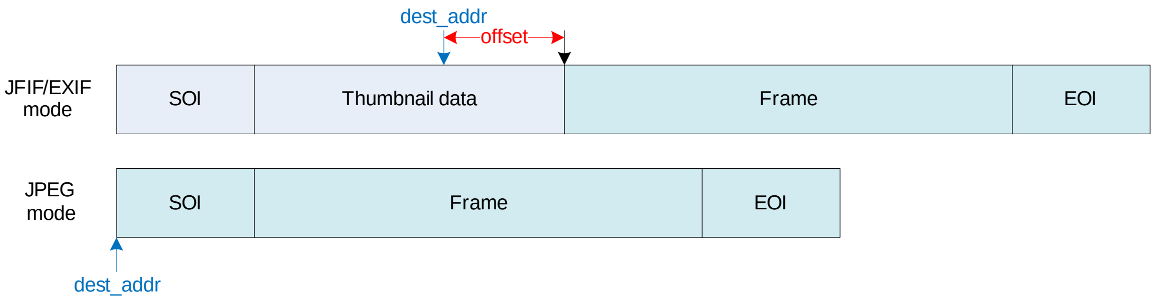
To avoid problem occurred with wrong SMI protocol, the software should do reset based on the following procedure.

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.

### 6.22.2.2 Byte Offset Address Setting

To align SMI bus bitwidth, the buffer address should be 16-bytes 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

## 6.22.3 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.

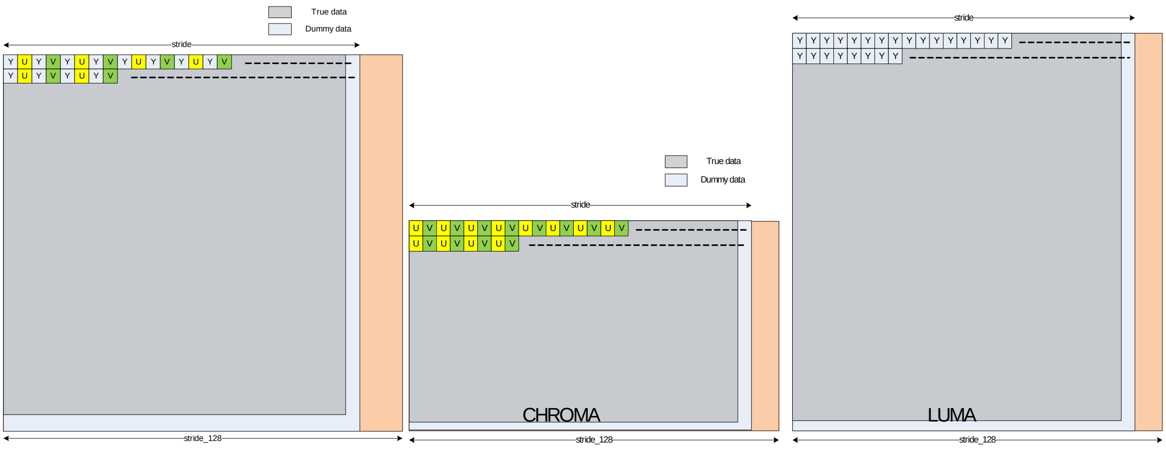


*Figure 6-71. Memory footprint of source frame buffer*

### 6.22.4 Register Definition

See chapater 4.22 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 6.23 Video Decoder

### 6.23.1 Introduction

MediaTek Video Decoder (VDEC) in project MT6737 supports multi-standard video compression format, 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 decoder Constraint Baseline Profile/Main/High Profile

VDEC supports full-HD 30fps under the limitation of picture size > full-HD, (i.e. does not support picture width > 1920 or picture height > 1088).

### 6.23.2 Block Diagram

The architecture and core blocks of VDEC are shown in Figure 6-72, 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.

### 6.23.3 Interface

The interface of VDEC is shown in Figure 6-73. 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 use DRAM as bitstream input, working buffer, reference buffer and output, and DRAM access process is achieved by using SMI interface. 7 sets of SMI interface with 128 bits 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 required.

Register settings are passed to VDEC by APB interface. There are 1 set of APB interface.

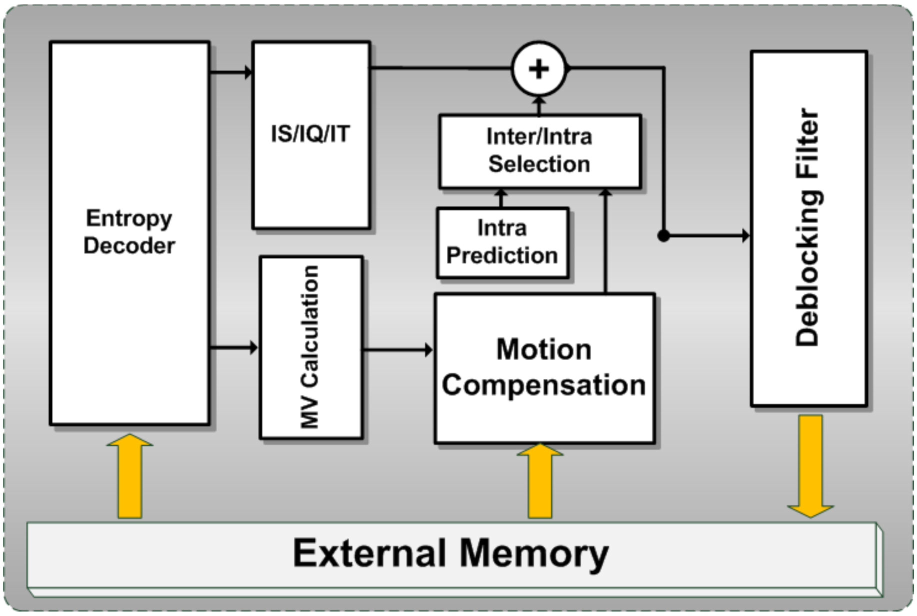


*Figure 6-72. Block diagram of video decoder*

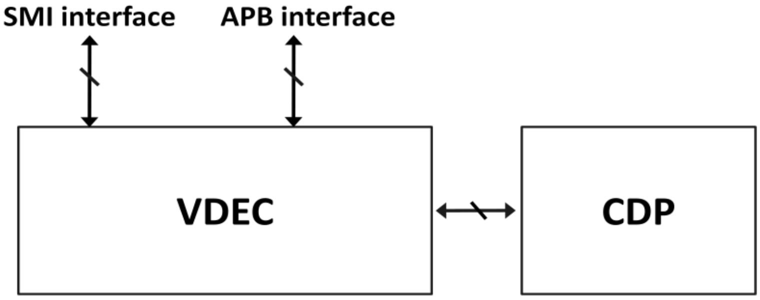


*Figure 6-73. Interface of VDEC*

### 6.23.4 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 under different video coding standards.

#### 6.23.4.1 Base Settings

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

#### 6.23.4.1.1 Clocks

Two clocks are required for VDEC.

- smi_clk: 300MHz
- vdec_clk: 273MHz

For more details, see the CKGEN register map.

#### 6.23.4.1.2 VDEC Register Base Address

Base address of each sub-module is described in Table 6-29.

*Table 6-29. 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 |

### 6.23.4.2 VDEC Hardware Architecture

Figure 6-74 is the hardware architecture; Figure 6-75 is the detail architecture of VLD. It is essential to know the architecture of VLD because most common settings are related to this module.

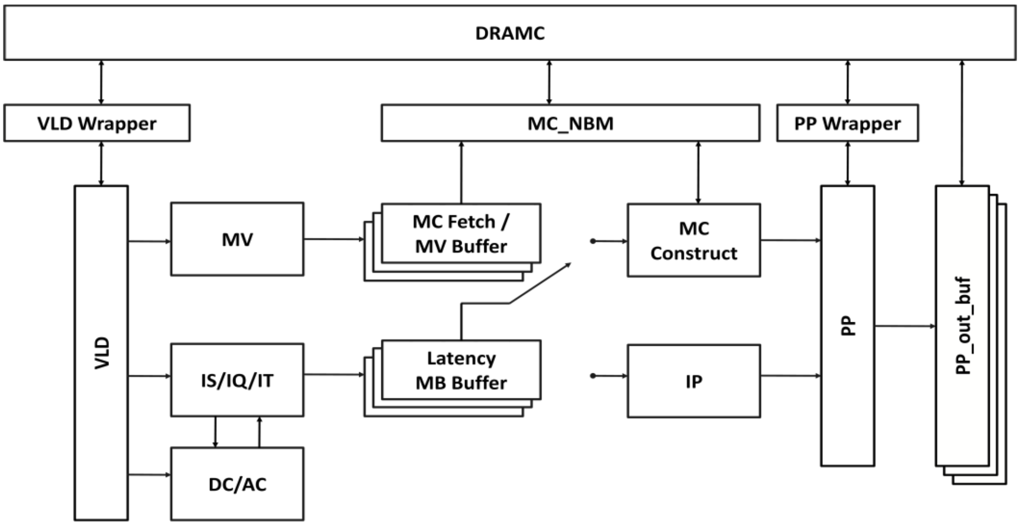


*Figure 6-74. VDEC hardware architecture*

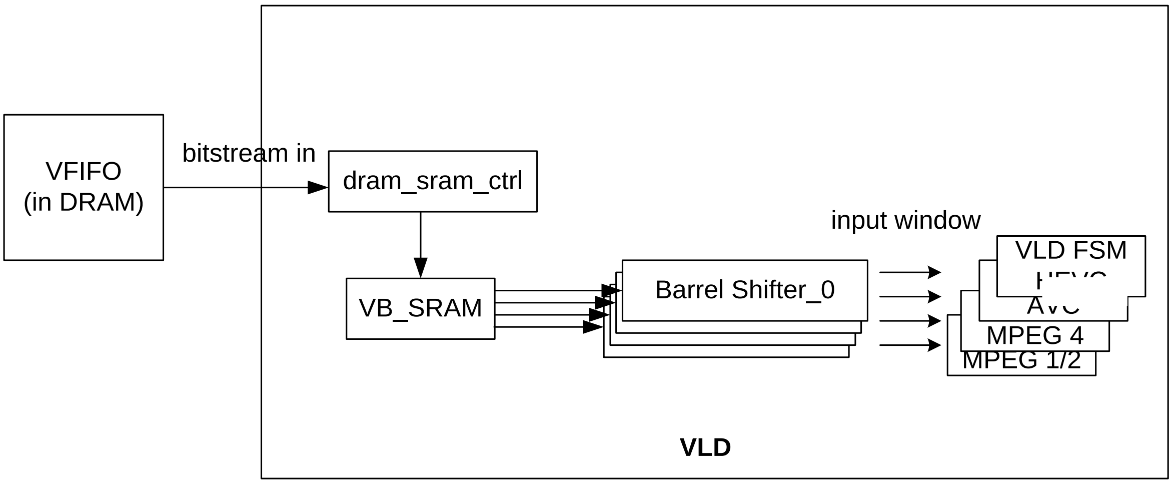


*Figure 6-75. VLD hardware architecture*

### 6.23.4.3 Firmware Hardware Partition

Figure 6-76 shows a typical decoding flow with FW/HW partition. SW takes charge of the 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 finishes decoding a frame and FW can reset HW and program settings for the next frame if necessary. Details of FW decoding are described in section 6.23.4.4.

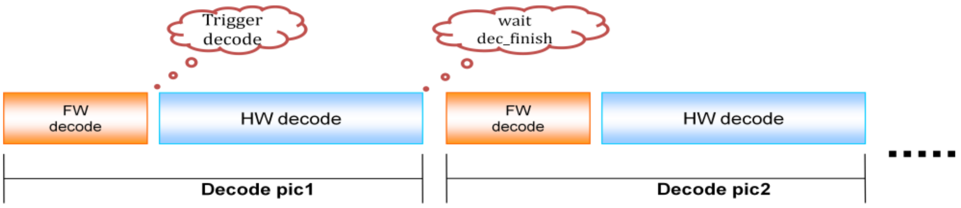


*Figure 6-76. VDEC decoding flow*

### 6.23.4.4 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. Initial 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).

#### 6.23.4.4.1 Software Reset and Power Control Flow

Figure 6-77 is the software reset and power control flow. Details of each step are described in the following sections.

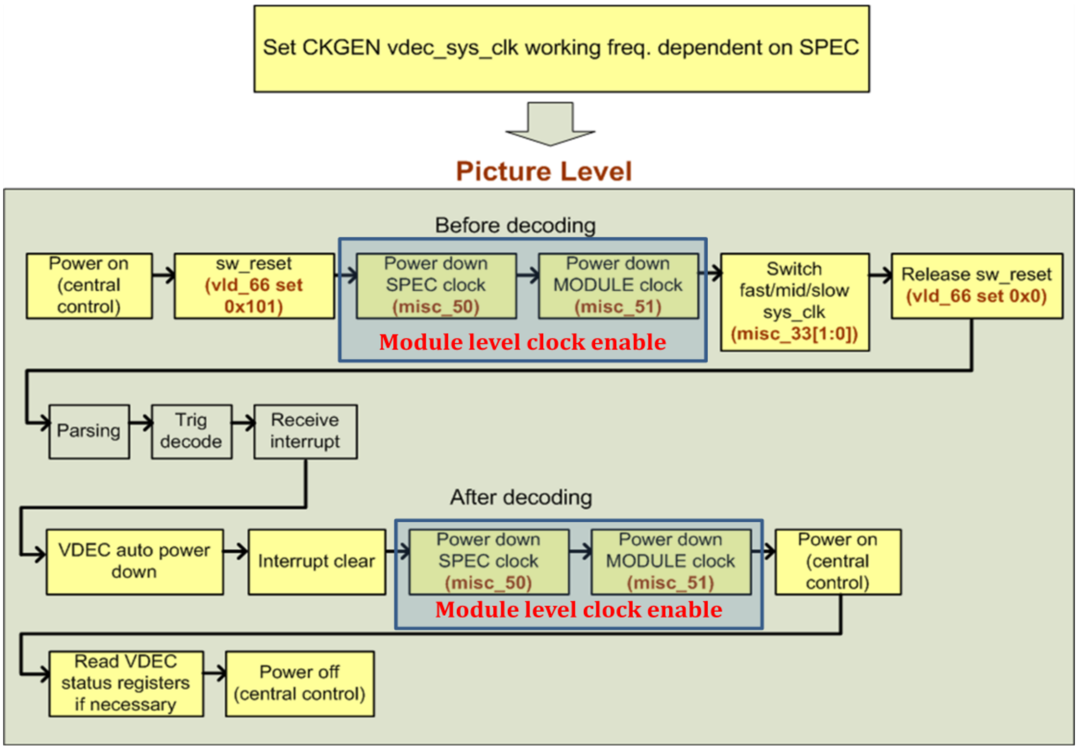

Set CKGEN vdec_sys_clk working freq. dependent on SPEC
Picture Level
Before decoding
Power on (central control)
sw_reset (vld_66 set 0x101)
Power down SPEC clock (misc_50)
Power down MODULE clock (misc_51)
Module level clock enable
Switch fast/mid/slow sys_clk (misc_33[1:0])
Release sw_reset (vld_66 set 0x0)
Parsing
Trig decode
Receive interrupt
After decoding
VDEC auto power down
Interrupt clear
Power down SPEC clock (misc_50)
Power down MODULE clock (misc_51)
Module level clock enable
Power on (central control)
Read VDEC status registers if necessary
Power off (central control)

*Figure 6-77. 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

#### 6.23.4.4.2 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 Figure 6-78.

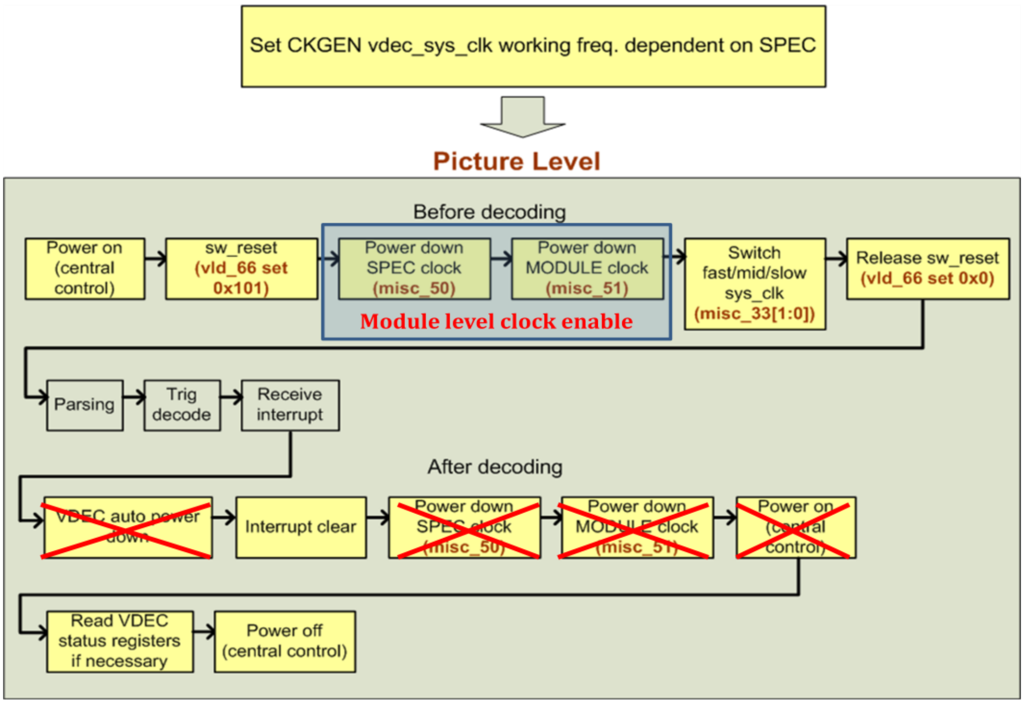

Set CKGEN vdec_sys_clk working freq. dependent on SPEC
Picture Level
Before decoding
Power on (central control)
sw_reset (vld_66 set 0x101)
Power down SPEC clock (misc_50)
Power down MODULE clock (misc_51)
Module level clock enable
Switch fast/mid/slow sys_clk (misc_33[1:0])
Release sw_reset (vld_66 set 0x0)
Parsing
Trig decode
Receive interrupt
After decoding
~~VDEC auto power down~~
Interrupt clear
~~Power down SPEC clock (misc_50)~~
~~Power down MODULE clock (misc_51)~~
~~Power on (central control)~~
Read VDEC status registers if necessary
Power off (central control)

*Figure 6-78. Software reset and power control (turn off VDEC auto power down)*

### 6.23.4.4.3 Other Relative Registers for 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".

### 6.23.4.4.4 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 up clock gating for each standard and modules.
   i. Software reset: Set VLD reg 66 = 0x101.
   ii. Set up **pdn_con_spec** & **pdn_con_module** setting for different specs.
      1. **pdn_con_spec** (**VDEC_MISC reg 50 (0xC8)**)
         A) **pdn_con_module** (**VDEC_MISC reg 51 (0xCC)**)
   iii. Set up vdec_sys_clk_sel for different specs:
      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 Module level power down control, depend on different specs. Details are described in Table 6-30.

*Table 6-30. Power down control and clock selection*

| SPEC | VDEC_MISC reg 50 (0xC8)<br>pdn_con_spec<br>(set 1: turn off clock ) | VDEC_MISC reg 51 (0xCC)<br>pdn_con_module<br>(set 1: turn off clock ) | VDEC_MISC reg 33(0x84)<br>vdec_sys_clk_sel<br>(0 = Slow; 1 = Mid; 2 = Fast) |
|---|---|---|---|
| MPEG1_2 | 0x1FE | 0x7F6A_15FD (w/o DBK)<br>0x7F6A_151D (w DBK) | 0x1 (mid) |
| MPEG4 | 0x1FD | 0x7F6A_11E8 (w/o DBK)<br>0x7F6A_1108 (w DBK) | 0x1 (mid) |
| H.264/MVC | 0x1F7 | 0x53E2_0180 | 0x2 (fast) |

#### 6.23.4.4.5 Interrupt Clear

In MT6737, 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

### 6.23.4.5 Initial Bitstream DMAs and Barrel-Shifters

Before triggering HW VDEC to decode a picture, load bitstream from DRAM to VDEC internal memory and set up correct read location of bitstream. These steps are known as the initial fetch and initial barrel-shifters. The concept is described in Figure 6-79, and the flow chart is in Figure 6-80.

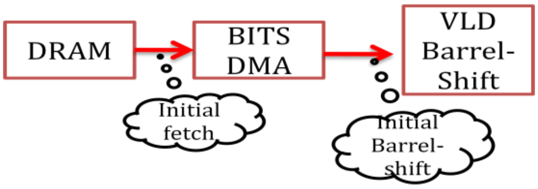
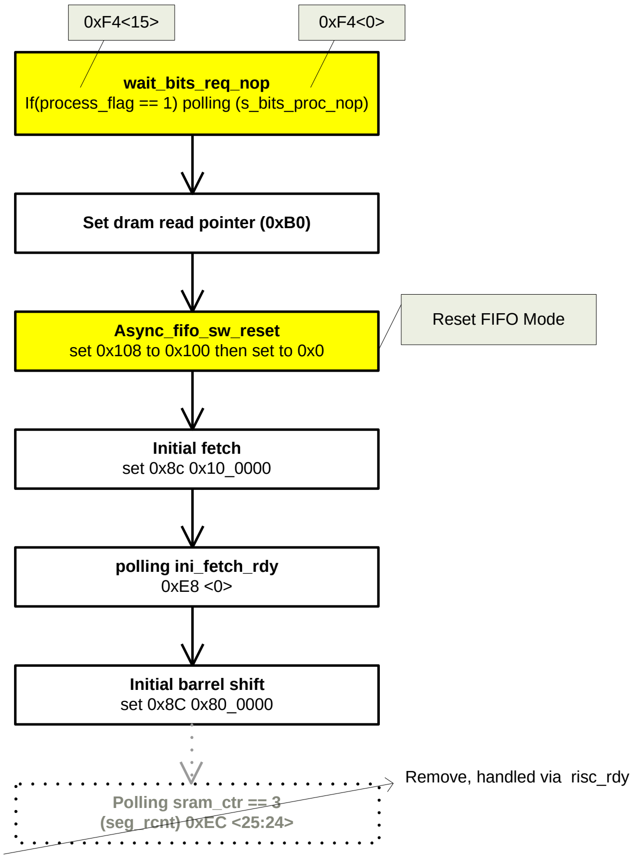


*Figure 6-79. Initial fetch and initial Barrel-shift*



ps. initial-fetch: sw_reset then initial fetch
re-initial-fetch: move r_ptr then initial fetch w/o sw_reset

*Figure 6-80. Flow chart of initial fetch and initial Barrel-shift*

#### 6.23.4.5.1 Bitstream FIFO (vb SRAM)

After initial fetch, the bitstream will be loaded in to bitstream FIFO (vb SRAM). When the amount of empty data is less than the threshold, the engine will auto fetch bitstream from DRAM to bitstream FIFO (see Figure 6-81).

- Read pointer of bitstream buffer: vb_sram_ra (0x0EC VB_SRAM_RA<4:0>)
- Write pointer of bitstream buffer: vb_sram_wa (0x0EC VB_SRAM_WA<12:8>)
- Read counter for each 128-bit data: seg_rcnt (0x0EC SEG_RCNT<25:24>)

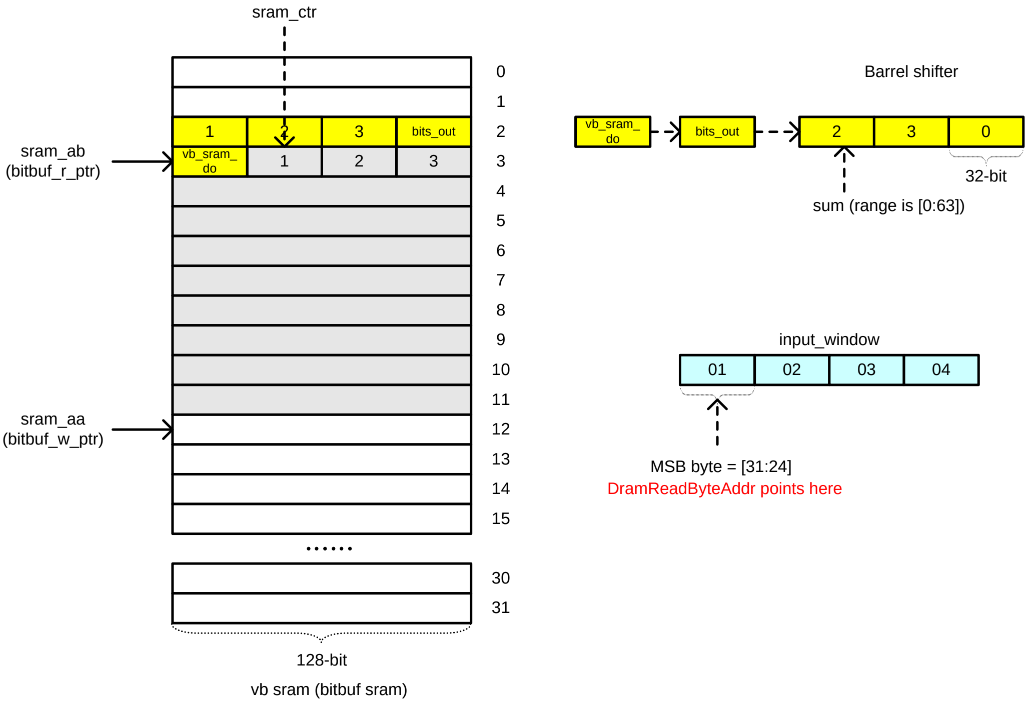


*Figure 6-81. Block diagram of bitstream buffer and barrel shifter*

When vb_sram_ra points to row 3 and seg_rcnt value is 1, the barrel shifter content is {row2{2}, row2{3}, row3{0}}.

### 6.23.4.5.2 Barrel Shifter

The barrel shifter has 3*32 bits. The sum is the read pointer of barrel shifter, i.e. the start address of input window. Figure 6-82 is the block diagram.

- Read pointer of barrel shifter: Sum (0x114 BSSR<5:0>); range [0, 63]
- Input window (0x0F0 BIW(#60) <31:0>)

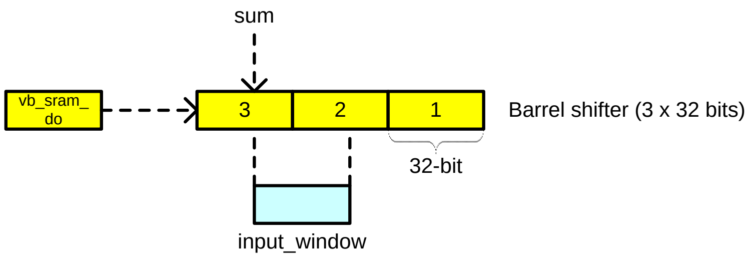


*Figure 6-82. Block diagram of barrel shifter and input window*

VDEC also provides checksum register for debugging. By checking the values of these registers with simulation result, you can confirm whether the above process is correct. The location of checksum registers are listed in Table 6-31.

*Table 6-31. Checksum register*

| Spec | Base | Reg offset |
|---|---|---|
| BITS_DMA | | |
| MPEG-2 | VLD | 69 (0x114) bit [5:0] |
| H.264/MVC | AVC_VLD | 33 (0xa4) bit [21:16] |

### 6.23.4.6 VDEC Output Format

VDEC supports two types of output formats:

- 16*32 Progressive Mode (NV12_BLK)
  - Original block mode (default)
- 16*32 Field Compact Mode (NV12_BLK_FCM)
  - PP, set PP reg 15 bit[28] = 0
  - MC, set MC reg 584 bit[24] = 0

The difference between progressive mode and field compact mode is shown in Figure 6-83. Besides MPEG2, MPEG4 and VP6, it is recommended that the output results are written to DRAM by PP for better efficiency.

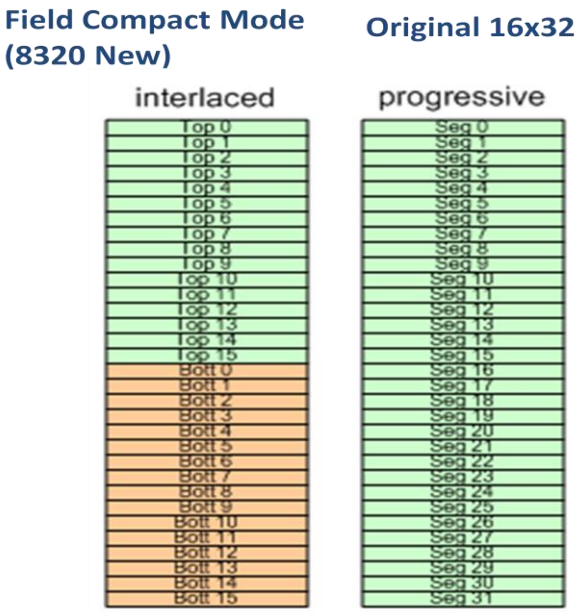


*Figure 6-83. Field compact mode and progressive mode*

#### 6.23.4.7 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.

### 6.23.5 Register Definition

See chapater 4.23 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 6.24 MPEG-4 Video Encoder

### 6.24.1 Introduction

MT6737 MPEG-4 VENC is a MPEG-4 simple profile 720p 30FPS @ 15Mbps encoder hardware design. It is composed of algorithm flexibility and guaranteed performance.

### 6.24.2 Features

- MPEG-4 simple profile @ level 3
- H.263 profile 0
- VOP type = I or P
- DC prediction
- Unrestricted motion compensation (with half-pel precision)
- Short header mode
- fcode = 1 ~ 7
- intra_dc_vlc_threshold = 0
- Maximum horizontal luma pixel resolution up to 1,280
- Maximum vertical luma pixel resolution up to 720

### 6.24.3 Block Diagram

MPEG-4 VENC adopts hybrid architecture. The hardware part processes texture coding and entropy coding, and other coding tools are processed by the ARM processor. VENC hardware and ARM processor transfer data through DRAM interface and ACE interface.

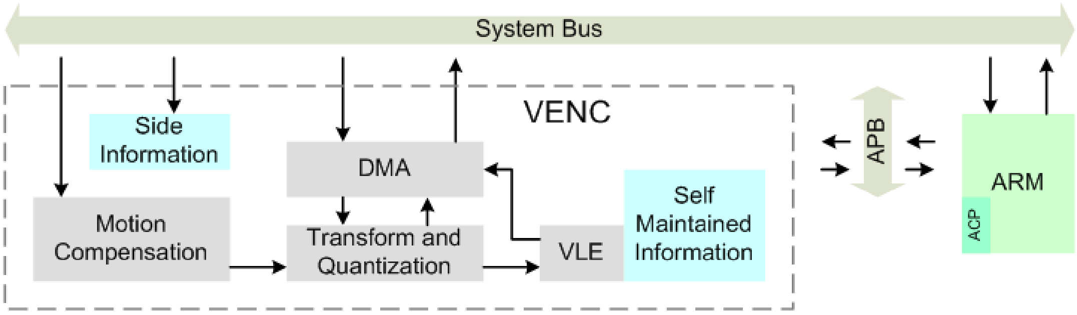


*Figure 6-84. Block diagram of MPEG-4 VENC*

### 6.24.4 Register Definition

See chapater 4.24 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

## 6.25 MFG

### 6.25.1 Introduction

MFG contains Mali-T720 MP1 GPU and clock/reset control logic. The Mali-T720 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.

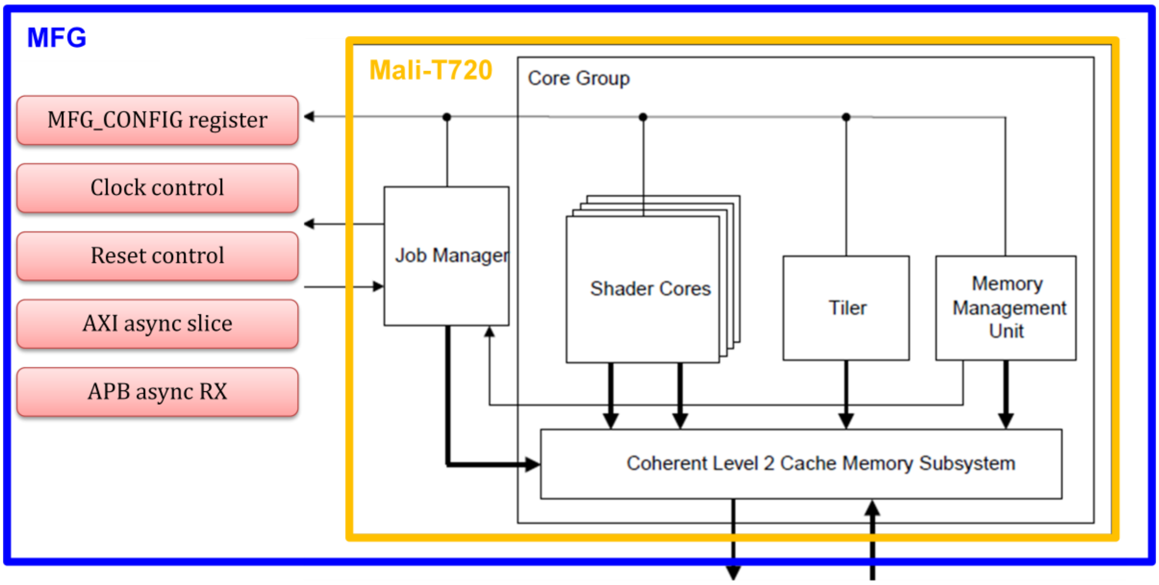


***Figure 6-85. Block diagram and interface of MFG***

### 6.25.2 Features

The Mali-T720 MP1 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*
  - *DirectX 9 Specification*
- Anti-aliasing capabilities
- An effective core for *General Purpose computing on GPU* (GPGPU) applications
- Image quality using double-precision FP64, and anti-aliasing
- Bus protocol
  - 128-bit AXI bus with up to 32 outstanding

  - 32-bit APB bus
- Level-2 cache
  - 64KB
  - 4-way set associative
- Performance
  - Triangle rate 72M Tri/sec
  - Fill rate 650M Pixels/sec
  - Shader rate 11 GFLOPS

### 6.25.3 Register Definition

See chapater 4.25 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.

## 6.26 Multimedia Memory Management Unit (M4U)

### 6.26.1 Introduction

In the smart phone system, the multimedia (MM) engine usually requires a certain amount of contiguous memory region. In order to avoid memory fragmentation, the Operating System (OS) needs to perform memory copy to align the requested ranges with one another. However, excessive memory copy may greatly degrade the system performance. For some performance-critical engines, a common method to guarantee sufficient memory available for them is to statically reserve the physical memory. This solution is straightforward but expensive since a large amount memory is reserved for a certain engine without sharing with others.

The Multimedia Memory Management Unit (M4U), which is also referred to as the IOMMU, is therefore designed to solve the fragmentation problem by paging the memory space and to reduce the cost of static memory reservation. Each page has an identical size of 4KB and can inherently be self-aligned without needing any memory copies. This means the system performance will be enhanced a lot due to exempting memory copies. With the unit of 4KB, the chance of occurring internal fragmentation is acceptable in our system and moreover the virtual memory efficiency is much better than that of using a larger unit of page size, e.g. 1MB. Therefore, with a smaller unit of page size, there will be ample page numbers available in the system. The engines will have even more chances to obtain the pages from the OS.

The major function of the M4U is to translate a virtual address into a physical one by performing page table look-up. As a result, the memory space of the MM engines is virtualized and contiguously seen by the MM engines. Along with the page table managed by the OS, there is no longer need to statically reserve any physical memory for the MM engines on system-up. The physical memory can be allocated dynamically in the unit of 4KB by the OS. From the system cost of view, the physical memory size is greatly reduced.

Since all the MM engines can use the M4U to perform address translation, M4U plays a critical role in MM performance, especially for hard real-time MM engines. The translation look-aside buffer (TLB), a major component in the M4U, caches the page numbers to accelerate the translation process. Besides, M4U performs auto-prefetching function to further lower the impact of page table walk caused by cache compulsory misses as well. In addition, a non-blocking TLB interface is designed to support miss-under-miss capability to achieve high performance translation.

### 6.26.2 Features

- One-level address translation with 4KB page size
- Two-level of TLB structure (main TLB and prefetch TLB):
  - Level 1 is used to cache the most-recently-used pages.
  - Level 2 performs table walk when TLB miss and auto-prefetching when TLB hit
  - Entry number: 48
- Entry lock in the main TLB for critical pages

- Supports TLB range invalidation and full range invalidation to free TLB entries
- Supports range sequential single-entry and multi-entry mode to minimize the entry usage
  - 16 sequential ranges available
- Supports range wrap mode in prefetch TLB for better address prediction
  - 4 wrap ranges available
- Each master port can be programmed to use M4U or not.
- Supports TrustZone security
- Supports high-performance bus prioritization for transactions with different priority levels

| Descriptor format | | | | |
|---|---|---|---|---|
| 31:12 | 11:4 | 3 | 2 | 1:0 |
| Page number | Reserved | Non-secure | Sharable | Access type |

*Figure 6-86. Descriptor attributes*

In Figure 6-86, bit[3]:NS is used to differentiate secure and non-secure memory region for TrustZone security support. bit[2]:sharable is used to indicate if the coherence access with the CPU caches is required or not. bit[1:0]:should be 2'b10 for a valid 4KB page.

### 6.26.3 Block Diagram

In the system architecture, M4U is placed between the Smart Multimedia Interface (SMI) and the External Memory Interface (EMI) (see Figure 6-87). In Figure 6-87, M4U is shared by all MM engines and performs virtual to physical address translation for MM engines.

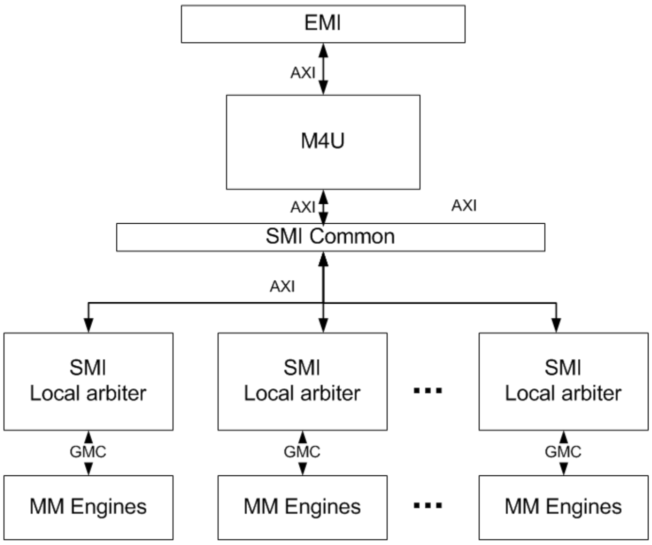


*Figure 6-87. MM bus architecture*

When a virtual address comes from the master port, M4U will perform page table look-up to translate the virtual address into a physical one. main_tlb and pre_fetch_tlb in (see Figure 6-88) are two levels of caches for the page table. They are used to accelerate the translation process. If a page number is a miss in tlbs, prefetch_tlb will perform "table walk" to obtain the page number in the DRAM. prefetch_tlb contains auto-prefetching logic as well to increase the cache hit-rate. The non-blocking TLB interface is designed to support miss-under-miss capability to achieve high performance translation.

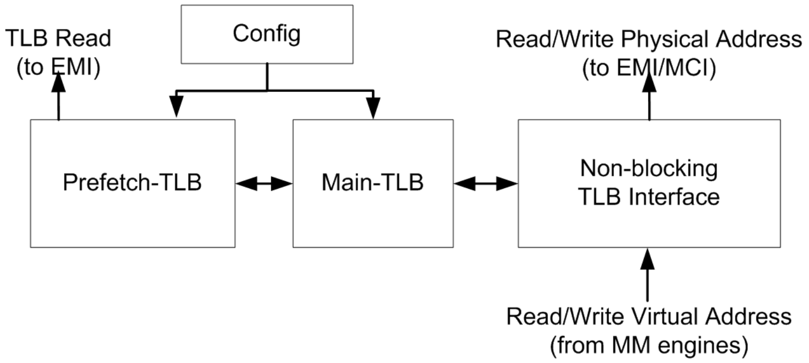


***Figure 6-88. Block diagram of M4U***

In Figure 6-88, M4U has a non-blocking interface to support miss-under-miss behavior and TLBs to cache the recently-used pages.

### 6.26.4 Register Definition

See chapater 4.26 of *"MT6737 LTE Smartphone Application Processor Software Register Table"*.

### 6.26.5 Programming Guide

1. Enable M4U for the corresponding ports (refer to the SMI_LARB/SMI_COMMON functional spec).
2. Request for a page table from the OS for the working address space.
3. Confirm the memory access behavior of each master.
4. Program the registers based on the master's memory-access behavior, e.g. set the range sequential single-entry for the scan-line based masters then the range sequential multiple-entry and wrap mode for the rotators.
5. Set the pre-fetch distance to either "forward" or "backward" direction for the ports (default: forward 1).
6. Set up the page table base address.
7. Launch the engines.

# 7 Analog Baseband

## 7.1 AP Mixedsys

### 7.1.1 Block Diagram

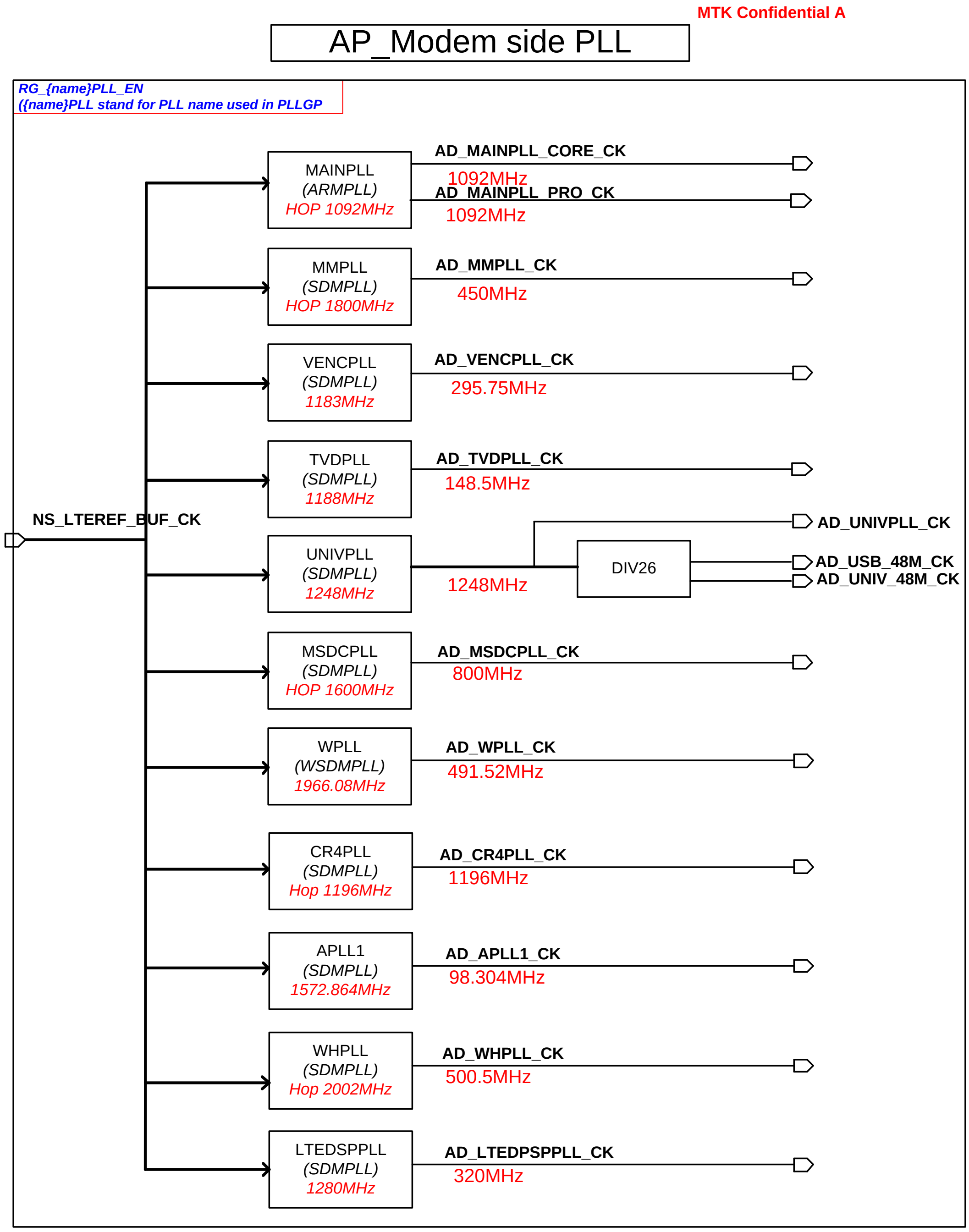


*Figure 7-1. PLL block diagram*

### 7.1.2 Register Definition

See chapater 5.1 of *“MT6737 LTE Smartphone Application Processor Software Register Table”*.