SG2002

Technical Reference Manual

Version. 1.0-alpha

Release Date. 2023-11-22

© 2023 Calculus

1 revised record

Revision	Date	Description
1.0-alpha	2023/11/22	first draft (of writing)

CONTENTS

1 Revised records 2

2 System Overview 18

2.1 Presentation 18
2.2 System Framework Diagram 18
2.3 Characteristics 19
2.3.1 Processor Cores 19
2.3.2 TPU 19
2.3.3 Video Codec 19
2.3.4 Video Interface 20
2.3.5 ISP and Image Processing 20
2.3.6 Hardware Acceleration Engine 21
2.3.7 Audio Codecs 21
2.3.8 Network Interface 21
2.3.9 Safety System Module 21
2.3.10 Intelligent and Safe Operating Environments 21
2.3.11 Peripheral Interface 22
2.3.12 External Memory Interface 22
2.3.13 Chip Physical Specifications 23

3 System Architecture 24

3.1 Processor Subsystems 24
3.2 Interrupt Subsystem 25
3.3 Address Space Mapping 28
3.4 Packages and Pinouts 31
3.4.1 Package 31
3.4.2 Pin Information Description 33
3.4.3 Pin Default Status (QFN) 37

4 Startup and Upgrade 38

4.1 Overview 38
4.2 Correspondence between the startup mode and the corresponding signal latch value 38
4.3 Mirror Burn Mode 38
4.4 Secure Startup 38

5 reset (a dislocated joint) 39

5.1 Overview 39
5.2 Reset Control 39
5.2.1 Power-on reset 39

5.2.2 System Hard Reset 39
5.2.3 Soft Reset 40
5.2.4 Processor Subsystem Soft Reset 40
5.2.5 Function subsystem and function module soft reset 40
5.3 Reset Configuration Register 40
5.3.1 Reset Configuration Register Overview 40
5.3.2 Reset Configuration Register Description 40

6 Clock 44

6.1 Overview 44
6.2 Functional Block Diagram 44
6.3 Clock Resources and Crossover Structures 44
6.4 PLL Configuration 45
6.5 Integer Multiplier PLL 46
6.5.1 Fractional Multiplier PLL 47
6.6 CLK_DIV Clock Division Configuration 47
6.6.1 IP/SYS Clock Source, Clock Division Configuration 48
6.6.2 MCLK0/MCLK1 49
6.6.3 Clk_A24M 49
6.7 PLL CRG Register Overview 49
6.8 PLL CRG Register Description 50
6.8.1 PLL_G2 Register Description 50
6.8.2 PLL_G6 Register Description 56
6.9 CLK_DIV CRG Register Overview 59
6.10 CLK_DIV CRG Register Description 60

7 System Controller 78

7.1 Overview 78
7.2 Functional Description 78
7.2.1 Global reset enable 78
7.2.2 System DMA Channel Mapping 78
7.2.3 DDR AXI Urgent/Qos Configuration 79
7.3 System Control Register 79
7.3.1 System Control Registers Overview 79
7.3.2 System Control Register Description 81

8 Pin Multiplexing and Control 86

8.1 Pin Multiplexing PINMUX 86
8.1.1 Correspondence of Interface Functions to Signals/Pins/FMUX Registers 86
8.1.2 FMUX Register Description 107
8.2 Pin Control PINCTRL 120

8.2.1 Pin Control Register Overview 120
8.2.2 PINCTRL Register Description 122

9 DMA Controller 144
9.1 Overview 144
9.2 Features 145
9.3 Functional Description 145
9.3.1 Peripheral Request Line 145
9.3.2 Access to space 145
9.3.3 Basic Transmission 145
9.3.4 Chained Table Transmission 147
9.3.5 Interrupts and Status 147
9.3.6 Channel Safety Configuration 149
9.4 Working mode 149
9.4.1 Clock and Reset 149
9.4.2 Initialization 149
9.4.3 Basic Transmission 149
9.4.4 Chained Table Transmission 150
9.4.5 Interrupt Handling 150
9.5 DMAC Register 150

10 Timer 201
10.1 Overview 201
10.2 Features 201
10.3 Functional Description 201
10.4 Working Methods 201
10.4.1 Initialization 201
10.4.2 Interrupt Handling 202
10.4.3 Clock Selection 202
10.5 Timer Register Overview 202
10.6 Timer Register Description 202

11 Watchdog 205
11.1 Overview 205
11.2 Features 205
11.3 Functional Description 205
11.3.1 Application Block Diagram 205
11.3.2 Functional Principles 206
11.4 Working mode 206
11.4.1 Count Clock Frequency Configuration 206
11.4.2 System Initialization Configuration 206
11.4.3 Interrupt Handling Process 207

11.4.4 Close WatchDog 207
11.5 WDT Register Overview 207
11.6 WDT Register Description 208
12 Real Time Clock 211
12.1 Overview 211
12.2 Features 211
12.3 Functional Description 211
12.4 Working mode 212
12.4.1 Count Clock Frequency 212
12.4.2 RTC reset 212
12.4.3 RTC Initialization 213
12.4.4 Analog Second Clock Initialization 214
12.4.5 Interrupt Handling 214
12.4.6 Sleep and Wake 215
12.4.7 Power Off and Power On 215
12.5 RTC Register Overview 215
12.6 RTC Register Description 218
12.6.1 RTC_CORE_REG 218
12.6.2 RTC_MACRO_REG 229
12.6.3 RTC_CTRL 231
12.7 Power Management and Low Power Modes 242
12.8 Overview 242
12.8.1 MCU only 243
12.8.2 Active mode. 243
12.9 Clock Control 243
12.9.1 Turning Off Unwanted Clock Dividers 243
12.9.2 Adjusting the operating frequency of the module 243
12.9.3 Module-level low-power control 243
12.10 Turn off unused PLLs 244
12.11 DDR Low Power Control 244
12.12 Voltage regulation 244
13 8051 Subsystems 246
13.1 Overview 246
13.2 Features 246
13.3 Functional Description 247
13.4 Working Methods 247
13.4.1 Power domain control flow 247
13.4.2 8051 Initialization 248
13.4.3 Interrupt Handling 248
13.4.4 MAILBOX 248

13.5 8051 Subsystem Registers Overview 249
13.6 8051 Subsystem Registers 249

14 Memory Interface 262

14.1 DDR Controller 262
14.1.1 Overview 262
14.1.2 Features 262
14.1.3 Functional Description 262
14.1.4 Working Methods 266
14.1.5 AXI Register 267
14.1.6 DDRC Register 300
14.2 SPI NOR Flash Controller 302
14.2.1 Overview 302
14.2.2 Features 302
14.2.3 Functional Description 302
14.2.4 Workflow 306
14.2.5 Register Overview 307
14.2.6 Register Description 308
14.3 SPI NAND Flash Controller 311
14.3.1 Overview 311
14.3.2 Features 311
14.3.3 Functional Description 312
14.3.4 Workflow 315
14.3.5 Data Structure (NAND Flash/SPI NAND Flash) 316
14.3.6 Register Overview 316
14.3.7 Register Description 317

15 Network Interface 319

15.1 Ethernet MAC 319
15.1.1 Overview 319
15.1.2 Functional Description 319
15.1.3 Overall data flow 319
15.1.4 Single Port Functional Configuration Description 320
15.1.5 Ethernet Packet Receiving Interrupt Management Function 320
15.1.6 Configuring PHY Chip Operating Status 320
15.1.7 Operating mode switching 321
15.1.8 Typical Applications 321
15.1.9 Register Offset Address Description 321
15.1.10 GMAC Register Overview 321
15.1.11 GMAC Register Description 322
15.2 Ethernet PHY 325
15.2.1 Overview 325

algorithmic technology

15.2.2 Functional Description 325
15.2.3 Function Configuration Diagram Description 326

16 Video Interface 327
16.1 VI 327
16.2 Overview 327
16.2.1 Features 327
16.2.2 Mode Function Description 328
16.2.3 Image storage mode 333
16.2.4 VI Register Overview 333
16.2.5 VI Register Description 335
16.3 VDP (Video Display Processor) 348
16.3.1 Overview 348
16.3.2 Architecture Description 348
16.3.3 Working mode 349
16.3.4 Functional Description 351
16.3.5 VDP Register Overview 354
16.3.6 VDP Register Description 358
16.4 MIPI Rx 399
16.4.1 Overview 399
16.4.2 Features 399
16.4.3 Functional Description 400
16.4.4 MIPI Rx Register Overview 418
16.4.5 MIPI Rx Register Description 421
16.5 MIPI Tx 442
16.5.1 Overview 442
16.5.2 Features 442
16.5.3 Functional Description 442
16.5.4 MIPI Tx Register Overview 449
16.5.5 MIPI Tx Register Description 449
16.5.6 MIPI Tx PHY Register Overview 451
16.5.7 MIPI Tx PHY Register Description 452

17 Audio Interface 455
17.1 AIAO 455
17.1.1 Overview 455
17.1.2 Features 455
17.1.3 Functional Description 456
17.1.4 Operating mode 460
17.1.5 AIAO Register Overview 460
17.1.6 AIAO Register Description 461
17.2 Audio Codec 475

algorithmic technology

17.2.1 Overview 475
17.2.2 Features 475
17.2.3 Audio Codec Register Overview 475

18 Peripherals 482
18.1 I2C 482
18.1.1 Overview 482
18.1.2 Functional Description 482
18.1.3 Function Block Diagram 482
18.1.4 I2C Protocol Timing 483
18.1.5 Operating mode 483
18.1.6 I2C Register Overview 485
18.1.7 I2C Register Description 487
18.2 UART 494
18.2.1 Overview 494
18.2.2 Features 495
18.2.3 Functional Description 495
18.2.4 Operating mode 496
18.2.5 UART Register Overview 501
18.2.6 UART Register Description 502
18.3 SPI 507
18.3.1 OVERVIEW 507
18.3.2 Features 508
18.3.3 Functional Description 508
18.3.4 Working mode 509
18.3.5 Three Serial Peripheral Bus Timing Diagrams 513
18.3.6 Register Overview 516
18.3.7 Register Description 517
18.4 eMMC/SD/SDIO Controller 528
18.4.1 Functional Description 528
18.4.2 Application Notes 540
18.4.3 Register Overview 556
18.4.4 Register Description 556
18.5 GPIO 570
18.5.1 Overview 570
18.5.2 Features 571
18.5.3 Operating mode 571
18.5.4 GPIO Register Overview 572
18.5.5 GPIO Register Description 572
18.6 USB DRD 575
18.6.1 Overview 575
18.6.2 Functional Description 575

18.6.3 USBC Function and Register Description 577
18.6.4 Host Initialization Procedure Description 616
18.6.5 Host Register Description 616
18.6.6 Device Initialization Procedure 625
18.6.7 Device Register Description 625
18.7 SARADC 635
18.7.1 Overview 635
18.7.2 Features 635
18.7.3 Working mode 636
18.7.4 SARADC Register Overview 636
18.7.5 SARADC Register Description 636
18.8 Temperature sensor 638
18.8.1 Overview 638
18.8.2 Operating mode 638
18.8.3 Temperature sensor register overview 640
18.8.4 Temperature sensor register description 641
18.9 PWM 644
18.9.1 Overview 644
18.9.2 Features 645
18.9.3 Working mode 645
18.9.4 PWM Register Overview 647
18.9.5 PWM Register Description 648
18.10 Key scan 653
18.10.1 Overview 653
18.10.2 Operating mode 653
18.10.3 Basic Settings 654
18.10.4 Using FIFO Mode 654
18.10.5 Using snapshot array mode 655
18.10.6 Key scan register overview 655
18.10.7 Key scan register description 655
18.11 Wiegand 659
18.11.1 Overview 659
18.11.2 Operating mode 660
18.11.3 Wiegand Register Overview 662
18.11.4 Wiegand Register Description 662
18.12 IRRX Infrared Interface 667
18.12.1 Overview 667
18.12.2 Specials 667
18.12.3 Working mode 667
18.12.4 IRRX Register Overview 667
18.12.5 IRRX Register Description 669

19 Safety subsystem modules 678

19.1 CryptoDMA..... 678

19.1.1 Overview..... 678
19.1.2 Functional Characteristics..... 679
19.1.3 DMA Function Description..... 679
19.1.4 Symmetric Key Algorithm Group Encryption Mode Functional Description..... 679
19.1.5 CryptoDMA Register Overview..... 681
19.1.6 CryptoDMA Register Description..... 683

19.2 Security debugging firewall..... 693

19.2.1 Overview..... 693
19.2.2 Status Inquiry and Password Input Interface (I2C)..... 693
19.2.3 Status Inquiry and Password Entry Process..... 694

19.3 Efuse Controller..... 695

Table of Contents

Figure 2-1 Architecture Diagram 18
Figure 3-1 Package Appearance Dimensions, Top View 32
Figure 3-2. Package External Dimensions, Bottom View 32
Figure 3-3 Pinouts 33
Figure 5-1 Reset Management Module Block Diagram 39
Figure 6-1 Clock Management Block Diagram 44
Figure 6-2 Clock Source Frequency Division Diagram 45
Figure 9-1 DMAC Hardware Control Flow Diagram 144
Figure 9-2 DMA Transmission Class 146
Figure 9-3 Relative Addresses and Data Formats of Linked Tables 147
Figure 9-4 Diagram of Interrupt States and Sources 148
Figure 11-1 WatchDog Application Block Diagram 206
Fig. 12-1 Example of DCDC Voltage Control with PWM 245
Figure 13-1 8051 Subsystem Architecture 247
Figure 14-1 SoC/DRAM Interconnect Diagram 263
Figure 14-2 Standard SPI Interface Mode Write Operation Timing Chart 302
Figure 14-3 Standard SPI Interface Mode Read Operation Timing Chart 303
Figure 14-4 Dual-Input SPI Interface Timing 303
Figure 14-5 Dual-IO SPI Interface Timing 304
Figure 14-6 Quad-Input SPI Mode Timing Chart 304
Figure 14-7 Quad-IO SPI Mode Timing Chart 305
Figure 14-8 Standard SPI Interface Mode Write Operation Timing 312
Figure 14-9 Standard SPI Interface Mode Read Operation Timing 312
Figure 14-10 SPI Nand x2 Interface Mode Operation Timing 313
Figure 14-11 SPI Nand x4 Interface Mode Operation Timing 313
Figure 15-1 eMAC Overall Data Flow 319
Figure 15-2 Internal 10/100 Ethernet PHY Block Diagram 326
Figure 16-1 VI Function Block Diagram 327
Figure 16-2 BT.1120 Horizontal Interface Timing 328
Figure 16-3 BT.1120 Vertical Interface Timing 329
Figure 16-4 BT.656 Horizontal Interface Timings 330
Figure 16-5 BT.601 vhs Synchronization Mode 330
Figure 16-6 BT.601 vde Synchronization Mode 331
Figure 16-7 BT.601 vsde Synchronization Mode 331
Figure 16-8 BT.601 Y/C Separate 16bit Mode 331
Figure 16-9 BT.601 Y/C Combined 8bit Mode 331
Figure 16-10 DC Sync Code Mode 332
Figure 16-11 DC Synchronous Signal Mode -vhs mode 332
Figure 16-12DC Synchronous Signal Mode -VDE mode 332
Figure 16-13DC Synchronized Signal Mode -VSDE mode 332

Figure 16-14 Bayer 12 bit Image Storage Methods 333
Figure 16-15 YCbCr 8bit Image Storage Methods 333
Figure 16-16 VI Register Overview 333
Figure 16-17 VDP Function Block Diagram 348
Figure 16-18 VDP Clock Generator 349
Figure 16-19 LVDS JEIDA/VESA mode data format 354
Figure 16-20 MIPI Rx Functional Block Diagram and Location in the System 399
Figure 16-21 MIPI Rx Supported Interface Types 400
Figure 16-22 Packet Transmission Mechanism 401
Figure 16-23 CSI-2 Long Packet Format 402
Figure 16-24 CSI-2 Short Packet Format 402
Figure 16-25 YUV422 8-bit Data Transmission Sequence 403
Figure 16-26 YUV422-8bit Data Packet Transmission Correspondence 403
Figure 16-27 YUV422 8-bit Frame Format 404
Figure 16-28 YUV422-10bit Data Transfer Sequence 404
Figure 16-29 YUV422-10bit Data Packet Transmission Correspondence 405
Figure 16-30 YUV422-10bit Frame Format 405
Figure 16-31 RAW8 Data Transfer Sequence 406
Figure 16-32 RAW8 Frame Format 406
Figure 16-33 RAW10 Data Transfer Sequence 406
Figure 16-34 RAW10 Frame Format 407
Figure 16-35 RAW12 Data Transfer Sequence 407
Figure 16-36 RAW12 Frame Format 407
Figure 16-37 RAW16 Data Transfer Sequence 408
Figure 16-38 RAW16 Frame Format 408
Figure 16-39 MIPI Interface Image Formats 409
Figure 16-40 MIPI Interface Wide Dynamic Data Transfer (using DT) 410
Figure 16-41 MIPI Interface Wide Dynamic Data Transfer (Using ID) 411
Figure 16-42 MIPI Interface Wide Dynamic Data Transfer (Register Settings) 412
Figure 16-43 Sub-LVDS Interface Data Format 412
Figure 16-44 Sub-LVDS Synchronization Code Example 413
Figure 16-45 Sub-LVDS Multi Lane Transmission Mode 413
Figure 16-46 Sub-LVDS Individual Pixel Timing Chart 414
Figure 16-47 Sub-LVDS Linear Mode Timing Chart 414
Figure 16-48 Sub-LVDS Wide Dynamic Mode I 415
Figure 16-49 Sub-LVDS Wide Dynamic Mode II 415
Figure 16-50 HiSPi Packetized-SP Mode 416
Figure 16-51 HiSPi Streaming-SP Mode 417
Figure 16-52 HiSPi Sync Code Support Mode 417
Figure 16-53 HiSPi Packetized-SP Wide Dynamic Transmission 418
Figure 16-54 HiSPi Streaming-SP Wide Dynamic Transmission 418
Figure 16-55 MIPI Rx PHY Register Overview 419

Figure 16-56 MIPI Rx CSI Controller Register Overview 420
Figure 16-57 MIPI Rx Sub-LVDS Controller Registers Overview 421
Figure 16-58 MIPI TX Function Block Diagram 442
Figure 16-59 RGB16-bit format (Data type = 0x0E) 444
Chart 16-60 RGB 18-bit format (Data type = 0x1E) 445
Figure 16-61 RGB 24-bit format (Data type = 0x3E) 446
Figure 16-62 RGB 30-bit format (Data type = 0x0D) 447
Figure 16-63 MIPI TX Timing Markers 447
Figure 16-64 MIPI TX Video mode Burst Transmission Timings 448
Figure 17-1 AIAO Block Diagram 455
Figure 17-2 Connecting to the Built-in Audio Codec via the I2S Interface 457
Figure 17-3 Diagram of AIAO connecting with external Audio Codec in Master Mode via I2S Interface 457
Figure 17-4 Diagram of AIAO Connecting to External Audio Codec via I2S Interface in Slave Mode 458
Figure 17-5 I2S Interface Timing 459
Figure 17-6 PCM Interface Standard Mode Timing 459
Figure 17-7 PCM Interface Left Alignment Mode Timing 460
Figure 18-1 I2C Function Block Diagram 483
Figure 18-2 I2C Protocol Timing 483
Fig. 18-3 I2C Non-DMA Data Transfer Software Flow 484
Figure 18-4 Data Transfer Software Flow in I2C DMA Mode 485
Figure 18-5 UART Application Block Diagram 496
Figure 18-6 UART Transmission Data Structure 496
Figure 18-7 UART Data Transmission Flowchart 498
Figure 18-8 UART Data Receiving Flowchart 499
Figure 18-9 SPI Application Block Diagram 508
Figure 18-10 Data Transmission Flow when External SPI/SSP Slave is Connected 511
Figure 18-11 Data Transfer Flow When Mapping to an External Microwire Slave 512
Figure 18-12 Motorola SPI Single Frame Transmission Format (SCPH = 0) 513
Figure 18-13 Motorola SPI Sequential Frame Transfer Format (SCPH = 0) 514
Figure 18-14 Motorola SPI Single Frame Transmission Format (SCPH = 1) 514
Figure 18-15 Motorola SPI Sequential Frame Transfer Format (SCPH = 1) 514
Figure 18-16 TI SSP Single Frame Transmission Format 515
Figure 18-17 TI SSP Continuous Frame Transmission Format 515
Figure 18-18 NS Microwire Single Frame Transmission Format 516
Figure 18-19 NS Microwire Continuous Frame Transmission Format 516
Figure 18-20 Functional Block Diagram of SDMMC Controller 530
Figure 18-21 Typical Applications 531
Figure 18-22 eMMC/SD/SDIO Command Format 531
Figure 18-23 eMMC/SD/SDIO Response Formats 532
Figure 18-24 eMMC/SD/SDIO 1-bit Data Packet Format 532

Figure 18-25 eMMC/SD/SDIO 4-bit Data Packet Format 533
Figure 18-26 eMMC/SD/SDIO 8-bit Data Packet Format 534

Figure 18-27 4-Bit Dual Data Rate Data Packet Format 535
Figure 18-28 8-Bit Dual Data Rate Data Packet Format 536
Figure 18-29 Non-Data Transmission Commands: Command Transmission and Receive Response via Signal Line CMD 537
Figure 18-30 Single-Block and Multi-Block Read Operations 537
Figure 18-31 Single-Block and Multi-Block Write Operations 538
Figure 18-32 Clock Off Program 540
Figure 18-33 Clock Configuration Flowchart 542
Figure 18-34 Command Transfer Program 544
Figure 18-35 Instruction Completion Procedure 546
Figure 18-36 Abort Command Program 547
Figure 18-37 Programming Diagram for Unsynchronized Abort Instruction 548
Figure 18-38 Programming Diagram for the Synchronized Abort Instruction 549
Figure 18-39 Program for Non-DMA Data Transfer Mode 551
Figure 18-40 SDMA Data Transfer Mode Program 553
Figure 18-41 ADMA Data Transfer Mode Programs 555
Figure 18-42 USB DRD Internal System Block Diagram 576
Figure 18-43 Temperature Measurement Time, Counts and Interrupt Generation 640
Figure 18-44 PWM Continuous Mode 645
Figure 18-45 PWM Pulse count mode 646
Figure 18-46 PWM Continuous Shift Mode 647
Figure 18-47 Keyscan Architecture Block Diagram 653
Figure 18-48 wiegand signaling bus transfer 0/1 method 659
Figure 18-49 Wiegand 26 Format 659
Chart 18-50 Wiegand 34 format 660
Figure 18-51 Numerical Significance of Common Magnetic Cards and Magnetic Clasps 660
Figure 18-52 wiegand architecture block diagram 661
Figure 18-53 IRRX Operating Modes 667
Figure 19-1 Safety Subsystem Modules 678
Figure 19-2 ECB Model 680
Figure 19-3 CBC Model 680
Figure 19-4 CTR Mode 681
Figure 19-5 eFuse CTRL Module Architecture 696

Table of Contents

Table 3-1 ARM Cortex-A53 Processor Interrupt Number and Source Mapping Table 25
Table 3-2 Main Processor RISCV C906 @ 1.0Ghz Interrupt Number and Interrupt Source Mapping Table 26
Table 3-3 Co-Processor RISCV C906 @ 700Mhz Interrupt Number and Interrupt Source Mapping Table..... 27
Table 3-4 Pin Information Table..... 33
Table 3-5 Pin Default Status Table..... 37
Table 6-1 PLL Configuration Parameter Table..... 45
Table 6-2 Integer PLL Configuration Parameters 46
Table 6-3 Fractional PLL Parameter Configuration..... 47
Table 6-4 Clock Source and Preset Crossover Parameters Table..... 47
Table 8-1 Interface Function and Signal/Pin/FMUX Register Correspondence Table 86
Table 8-2 FMUX Register Configuration Information Table..... 107
Table 8-3 Pin and Pin Control Register Correspondence Information Table..... 120
Table 8-4 Pin Control Register Configuration Table 122
Table 9-1 DMAC Access Space Types..... 145
Table 10-1 Timer Register Overview (Base Address 0x030A0000) 202
Table 11-1 WDT Register Overview..... 207
Table 12-1 RTC_REG Register Overview (Base Address 0x05026000)..... 215
Table 12-2 RTC_MACRO_REG Register Overview (Base Address 0x05026400) 217
Table 12-3 RTC_CTRL Register Overview (Base Address 0x05025000)..... 217
Table 13-1 RTC_CTRL Register Overview (Base Address 0x05025000)..... 249
Table 14-1 DDR2/DDR3 Command Signal Comparison Table..... 263
Table 14-2 DDR2 Command Truth Table..... 264
Table 14-3 DDR3 Command Truth Table..... 264
Table 15-1 GMAC Register Overview 321
Table 17-1 AIAO Subsystem Registers Overview (Base Address 0x0410_8000) 460
Table 17-2 I2S_TDM_0/1/2/3 Register Overview (Base Address 0x0410_0000 + n*0x10000) 461
Table 18-1 I2C Clock Selection and Associated Register Configuration Relationship Table..... 483
Table 18-2 6-Group I2C Module Base Addresses for Chip..... 485
Table 18-3 Overview of I2C Registers for the Chip 486
Table 18-4 6-Group UART Module Base Addresses for Chip 501
Table 18-5 Overview of the Chip's UART Registers 501
Table 18-6 4-Group SPI Module Base Addresses for Chip 516
Table 18-7 SPI Register Overview..... 517
Table 18-8 SDMMC Controller Function Signal and Pin Correspondence Table 529
Table 18-9 SD3.0 Supported Speeds and Voltages..... 538

Table 18-10 eMMC Supported Speed Modes and Voltages 539
Table 18-11 SDMMC Register Overview 556
Table 18-12 Chip Base Addresses for 4 GPIO Modules 572
Table 18-13 GPIO Register Overview 572
Table 18-14 Tempsensor Interrupt Signal Description Table 639

Table 18-15 PWM Register Overview 647
Table 19-1 Status Query I2C Interface Register Addresses 693

2 System Overview

2.1 present (sb for a job etc)

SG2002 is a high-performance, low-power chip for AIot field, built-in multiple powerful cores: 2xC906, 1xArm Cortex A53, 1x8051, users can quickly switch cores according to needs. Meanwhile, it integrates hardware codecs: H.264 video compression codec, H.265 video compression encoder and ISP, and supports the configuration of professional-grade video image ISP: HDR Wide Dynamic, 3D Noise Reduction, Defogging, Lens Aberration Correction, and other image enhancement and correction algorithms.

The chip also integrates a self-developed TPU that delivers 1.0TOPS IN8 of arithmetic power. The specially designed TPU scheduling engine efficiently delivers extremely high bandwidth data streams to all tensor processor cores.

Meanwhile, it supports rich peripheral interfaces: 2xI2C, 2xSDIO3.0, 2xI2S, 15xPWM, 1xUSB2.0, etc., which can satisfy the needs of various scenarios.

2.2 System Framework Diagram

Figure *2-1* Architecture Diagram

algorithmic technology

2.3 Characteristics

2.3.1 processor core

- Main Processor RISCV C906 @ 1.0Ghz .
  - 32KB I-cache, 64KB D-Cache
  - Integrated Vector and Floating Point Unit (FPU)
- Main Processor ARM Cortex-A53 @ 1.0Ghz
  - 32KB I-cache, 32KB D-Cache
  - 128KB L2 cache
  - Neon and floating-point FPU support
- Co-processor RISCV C906 @ 700Mhz
  - Integrated Floating Point Unit (FPU)

2.3.2 TPU

- Built-in TPU, with ~1.0TOPS INT8.
- Support for major neural network architectures: Caffe, Pytorch, TensorFlow(Lite), ONNX and MXNet.
- It can realize Pedestrian Detection, Face Detection, Face recognition, Face anti-spoofing and other video structuring applications.

2.3.3 Video Codec

- H.264 Baseline/Main/High profile
- H.265 Main profile
- H.264/H.265 supports both I-frame and P-frame.
- MJPEG/JPEG baseline
- H.264 codec maximum resolution : 2880x1620 (5M)
- H.265 Encoding Maximum Resolution : 2880x1620 (5M)
- H.264 codec performance
  - 2880x1620@30fps+720x576@30fps
  - 1920x1080@30fps encoding + 1920x1080@30fps decoding
- H.265 Encoding Performance
  - 2880x1620@30fps+720x576@30fps
- JPEG Maximum Codec Performance
  - 2880x1620@30fps

- Supports various bit rate control modes such as CBR/VBR/FIXQP.

- Supports Region of Interest (ROI) coding

2.3.4 video interface

- importation
  - Supports two simultaneous video inputs (mipi 2L+1L)
  - Support MIPI, Sub-LVDS, HiSPI and other serial interfaces.
  - Support 8/10/12 bit RGB Bayer video input.
  - Support for BT.656
  - Supports the AHD multiplexed BT format.
  - Supports SONY, OnSemi, OmniVision and other HD CMOS sensors.
  - Provides a programmable frequency output for use by the sensor as a reference clock.
  - Supports a maximum width of 2880 and a maximum resolution of 5M (2688x1944, 2880x1620).
- exports
  - Supports a wide range of serial and parallel display specifications.
  - Supports serial interfaces such as MIPI
  - Supports BT656, BT601(8bit), BT1120, 8080 and other ❹ line interfaces.
  - Supports SPI output interface.

2.3.5 ISP and image processing

- Image video 90, 180, 270 degree rotation
- Image Video Mirror, Flip Function
- Video 2-layer OSD overlay
- Video 1/32 to 32x zoom function
- 3A (AE/AWB/AF) algorithm
- Fixed mode noise cancellation, bad dot correction
- Lens shading correction, lens distortion correction, violet edge correction
- Orientation Adaptive demosaic
- Gamma correction, (regional/full-area) dynamic contrast enhancement, color management and enhancement
- Zone Adaptive Defogging
- Bayer Noise Reduction, 3D Noise Reduction, Detail Enhancement and

Sharpening Enhancement

- Local Tone mapping
- Sensor Self Wide Dynamic and 2 Frame Wide Dynamic
- 2-axis digital image stabilization
- Lens distortion correction

algorithmic technology

- Provides PC-side ISP tuning tools

2.3.6 hardware acceleration engine

- Hardware/software mixed mode support for some OpenCV libraries.
- Hardware/Software Mixed Mode support for some IVE libraries.

2.3.7 Audio Codec

- Integrated Audio CODEC, supports 16 bit audio/voice input and output.
- Integrated mono microphone input.
- Integrated mono output. (Requires external amplifier to drive speakers)
- Another direct microphone output channel is integrated internally for easy AEC implementation.
- Software audio codec protocols (G.711, G.726, ADPCM)
- The software supports Audio 3A (AEC, ANR, AGC) functions.

2.3.8 network interface

- The Ethernet module provides one Ethernet MAC for receiving and sending network data.
- Ethernet MAC with built-in 10/100Mbps Fast Ethernet Transceiver operates at 10/100Mbps.
  Full-duplex or half-duplex mode.

2.3.9 Security system module

- Hardware implementation of AES/DES/SM4 encryption and decryption algorithms
- Hardware implementation of the HASH (SHA1/SHA256) hash algorithm
- Hardware implementation of a random number generator
- Internal integration of 2Kbit eFuse logic space

2.3.10 Intelligent and safe operating environment

- Support for Chain of Trust Establishment: Provides the foundation of a

secure environment and is fundamental to a trusted environment, such as hardware security settings, root of trust, etc.

- Supports secure boot, provides secure hardware and software protection features

- Support data encryption security: data encryption program, computing core encryption
- Support for software and firmware validation process: Diagnose the credibility and integrity of the software, including boot and load validation procedures
- Supports secure storage and transmission: protects external data storage and exchange
- Support for security updates

2.3.11 Peripheral Interface

- Integrated POR, Power sequence
- 4 single-ended ADCs (3 no die domain)
- 6 I2C (1 no die domain)
- 3 SPI
- 5 UART groups (1 no die domain)
- 4 groups (15 channels) PWM
- 2 SDIO interfaces.
  - One SD 3.0 Card supporting 3V connectivity (supports up to SDXC 2TB, supports UHS-I speed)
  - One supports 1.8V/3.0V connection to other SDIO 3.0 devices. (UHS-I speed support)
- 66 GPIO Interface (14 no die domain)
- Integration with keyscan and Wiegand
- Integrated MAC PHY supports 10/100Mbps full or half duplex mode.
- One USB Host / device interface

2.3.12 external memory interface

- Internal DRAM
  - DDR3 16bitx1, up to 1866Mbps, capacity 2Gbit (256MB)
- SPI NOR flash interface (1.8V / 3.0V)
  - Supports 1, 2, 4-wire modes.
  - Supports up to 256MByte.
- SPI Nand flash interface (1.8V / 3.0V)
  - Supports 1KB/2KB/4KB page (corresponding to the maximum capacity of 16GB/32GB/64GB).

- Use the ECC module built into the device.

- eMMC 4.5 interface (1.8V/3.0V) SD0 EMMC common power. Because of SD card default 3V, it is not suitable to connect 1.8V eMMC with SD card.
  - 4 bit interface.
  - Support HS200

- Maximum Supported Capacity 2TB

2.3.13 Chip Physical Specifications

- power wastage
  - 1080P + Video encode + AI : ~ 500mW
  - Remaining Scenarios : TBD
- operating voltage
  - Kernel voltage is 0.9V
  - IO voltage of 1.8V and 3.0V
  - The DDR voltage is shown in the table below.
    - ✓ 1.35V
- seal inside
  - QFN package with 9mmx9mmx0.9mm. 0.35mm pin spacing. 88 pins.

3 system architecture

3.1 processor subsystem

This chip has two main processors: RISC-V C906 and ARM Cortex-A53, which are switched by the peripheral circuit control of pin GPIO_RTX (refer to the datasheet for details) and there are two operation modes:

- RISC-V C906@1GHz + RISC-V C906@700MHz.
- ARM Cortex-A53@1GHz + RISC-V C906@700MHz.

It adopts ARM Cortex-A53 processor with the following features:

- The processor operates at up to 1.0 GHz.
- Support Neon.
- Floating point unit FPU support.
- The L1 Cache contains 32KB Instruction Cache and 32KB Data Cache.
- 128KB L2 cache.
- Supports MMU (Memory Management Unit)
- Integrated interrupt controller inside the processor
- Supports JTAG debugging interface.

Utilizing the RISCV C906 processor, it has the following features:

- The processor operates at up to 1.0 GHz.
- Integrated vector execution unit, floating point coprocessor.
- The L1 Cache contains 32KB Instruction Cache and 64KB Data Cache.
- Supports MMU (Memory Management Unit)
- Integrated interrupt controller inside the processor
- Supports JTAG debugging interface.

3.2 interruption subsystem

Table *3-1 ARM Cortex-A53* Processor Interrupt Number and Source Mapping Table

interruption number	source of interruption	interruption number	source of interruption	interruption number	source of interruption
32	TEMPSENS Interruptions	70	SPI1 Interrupt	108	TRNG Interrupt
33	RTC Alarm Interrupt	71	SPI2 Interrupt	109	ddr_axi_mon interrupt
34	RTC Longpress Interrupt	72	SPI3 Interrupt	110	ddr_pi_phy interrupt
35	VBAT DET Interrupt	73	SPI4 Interrupt	111	SPI_NOR Interrupt
36	JPEG Interrupt	74	Watch Dog0 Interrupt	112	EPHY Interruptions
37	H264 Interrupt	75	KEYSCAN Interrupt	113	IVE Interruptions
38	H265 Interruptions	76	GPIO0 Interrupt	114	reservations
39	VC SBM Interrupt	77	GPIO1 Interrupt	115	reservations
40	ISP interrupt	78	GPIO2 Interrupt	116	SARADC Interrupt
41	SC_TOP Interrupt	79	GPIO3 Interrupt	117	mbox Interrupt
42	CSI_MAC0 Interrupt	80	Wiegand0 Interrupt	151	npmuirq[0]
43	CSI_MAC1 Interrupt	81	Wiegand1 Interrupt	152	ctiirq[0]
44	LDC Interrupt	82	Wiegand2 Interrupt	159	nexterrirq
45	System DMA Interrupt	83	RTC MBOX Interrupt	116	SARADC Interrupt
46	USB interrupt	84	N/A		
47	Ethnet0 Interrupt	85	RTC IRRX Interrupt		
48	Ethnet0 Interrupt	86	RTC GPIO Interrupt		
49	EMMC Wakup Interrupt	87	RTC UART Interrupt		
50	EMMC Interrupt	88	RTC SPI_NOR interrupt		
51	SD0 Wakup Interrupt	89	RTC I2C Interrupt		
52	SD0 Interrupt	90	RTC WDG Interrupt		
53	SD1 Wakup Interrupt	91	TPU Interrupt		
54	SD1 Interrupt	92	TDMA interrupt		
55	SPI_NAND Interrupt	93	reservations		
56	I2S0 Interrupt	94	reservations		
57	I2S1 Interrupt	95	Timer0 Interrupt		
58	I2S2 Interrupt	96	Timer1 Interrupt		
59	I2S3 Interrupt	97	Timer2 Interrupt		
60	UART0 Interrupt	98	Timer3 Interrupt		
61	UART1 Interrupt	99	Timer4 Interrupt		
62	UART2 Interrupt	100	Timer5 Interrupt		
63	UART3 Interrupt	101	Timer6 Interrupt		
64	UART4 Interrupt	102	Timer7 Interrupt		
65	I2C0 Interrupt	103	peri_firewall		

			interrupt		
66	I2C1 Interrupt	104	hsperi_firewall interrupt		
67	I2C2 Interrupt	105	ddr_fw interrupt		
68	I2C3 Interrupt	106	rom_firewall interrupt		

69	I2C4 Interrupt	107	SPACC Interrupt		

Table *3-2* Main Processor *RISCV C906 @ 1.0Ghz* Interrupt Number and Interrupt Source Mapping Table

interruption number	source of interruption	interruption number	source of interruption	interruption number	source of interruption
16	TEMPSENS Interruptions	52	I2C3 Interrupt	88	hsperi_firewall interrupt
17	RTC Alarm Interrupt	53	I2C4 Interrupt	89	ddr_fw interrupt
18	RTC Longpress Interrupt	54	SPI1 Interrupt	90	rom_firewall interrupt
19	VBAT DET Interrupt	55	SPI2 Interrupt	91	SPACC Interrupt
20	JPEG Interrupt	56	SPI3 Interrupt	92	TRNG Interrupt
21	H264 Interrupt	57	SPI4 Interrupt	93	ddr_axi_mon interrupt
22	H265 Interruptions	58	Watch Dog1 Interrupt	94	ddr_pi_phy interrupt
23	VC SBM Interrupt	59	KEYSCAN Interrupt	95	SPI_NOR Interrupt
24	ISP interrupt	60	GPIO0 Interrupt	96	EPHY Interruptions
25	SC_TOP Interrupt	61	GPIO1 Interrupt	97	IVE Interruptions
26	CSI_MAC0 Interrupt	62	GPIO2 Interrupt	98	reservations
27	CSI_MAC1 Interrupt	63	GPIO3 Interrupt	99	reservations
28	LDC Interrupt	64	Wiegand0 Interrupt	100	SARADC Interrupt
29	System DMA Interrupt	65	Wiegand1 Interrupt	101	mbox Interrupt
30	USB interrupt	66	Wiegand2 Interrupt		
31	Ethnet0 Interrupt	67	RTC MBOX Interrupt		
32	Ethnet0 Interrupt	68	N/A		
33	EMMC Wakup Interrupt	69	RTC IRRX Interrupt		
34	EMMC Interrupt	70	RTC GPIO Interrupt		
35	SD0 Wakup Interrupt	71	RTC UART Interrupt		
36	SD0 Interrupt	72	RTC SPI_NOR interrupt		
37	SD1 Wakup Interrupt	73	RTC I2C Interrupt		
38	SD1 Interrupt	74	RTC WDG Interrupt		
39	SPI_NAND Interrupt	75	TPU Interrupt		
40	I2S0 Interrupt	76	TDMA interrupt		
41	I2S1 Interrupt	77	reservations		
42	I2S2 Interrupt	78	reservations		
43	I2S3 Interrupt	79	Timer0 Interrupt		
44	UART0 Interrupt	80	Timer1 Interrupt		
45	UART1 Interrupt	81	Timer2 Interrupt		
46	UART2 Interrupt	82	Timer3 Interrupt		
47	UART3 Interrupt	83	Timer4 Interrupt		
48	UART4 Interrupt	84	Timer5 Interrupt		

49	I2C0 Interrupt	85	Timer6 Interrupt		
50	I2C1 Interrupt	86	Timer7 Interrupt		
51	I2C2 Interrupt	87	peri_firewall interrupt		

Table *3-3* C o - p r o c e s s o r *RISCV C906 @ 700Mhz* Interrupt Number and Interrupt Source Mapping Table

interruption number	source of interruption	interruption number	source of interruption
16	JPEG Interrupt	54	reservations
17	H264 Interrupt	55	Timer4 Interrupt
18	H265 Interruptions	56	Timer5 Interrupt
19	VC SBM Interrupt	57	Timer6 Interrupt
20	ISP interrupt	58	Timer7 Interrupt
21	SC_TOP Interrupt	59	SPACC Interrupt
22	CSI_MAC0 Interrupt	60	IVE Interruptions
23	CSI_MAC1 Interrupt	61	mbox Interrupt
24	LDC Interrupt		
25	System DMA Interrupt		
26	I2S0 Interrupt		
27	I2S1 Interrupt		
28	I2S2 Interrupt		
29	I2S3 Interrupt		
30	UART0 Interrupt		
31	UART1 Interrupt		
32	I2C0 Interrupt		
33	I2C1 Interrupt		
34	I2C2 Interrupt		
35	I2C3 Interrupt		
36	I2C4 Interrupt		
37	SPI1 Interrupt		
38	SPI2 Interrupt		
39	Watch Dog2 Interrupt		
40	KEYSCAN Interrupt		
41	GPIO0 Interrupt		
42	GPIO1 Interrupt		
43	GPIO2 Interrupt		
44	GPIO3 Interrupt		
45	Wiegand0 Interrupt		
46	RTC MBOX Interrupt		
47	RTC IRRX Interrupt		
48	RTC GPIO Interrupt		
49	RTC UART Interrupt		
50	RTC I2C Interrupt		
51	RTC WDG Interrupt		

52	TDMA interrupt		
53	reservations		

3.3 address space mapping

Starting address [31:0]	End address [31:0]	Space Function	Space Size (Byte)
0x01000000	0x017FFFFF	reservations	8M
0x01800000	0x018FFFFF	reservations	
0x01900000	0x01900FFF	ap_mailbox	4K
0x01901000	0x01901FFF	ap_system_ctrl	4K
0x01902000	0x019EFFFF	reservations	
0x01F00000	0x01F0FFFF	reservations	64K
0x01F10000	0x01FFFFFF	reservations	
0x02000000	0x02FFFFFF	reservations	64K
0x03000000	0x03000FFF	TOP_MISC Control Register	4K
0x03001000	0x03001FFF	PINMUX Control Register	4K
0x03002000	0x03002FFF	CLKGEN/PLL Control Register	4K
0x03003000	0x03003FFF	RSTGEN Control Register	4K
0x03004000	0x03005FFF	reservations	
0x03006000	0x03006FFF	reservations	4K
0x03007000	0x03008FFF	reservations	
0x03009000	0x03009FFF	reservations	4K
0x0300A000	0x0300AFFF	reservations	4K
0x0300B000	0x0300FFFF	reservations	
0x03010000	0x03010FFF	WATCH DOG0 Control Registers	4K
0x03011000	0x03011FFF	WATCH DOG1 Control Registers	4K
0x03012000	0x03012FFF	WATCH DOG2 Control Registers	4K
0x03020000	0x03020FFF	GPIO0 Control Register	4K
0x03021000	0x03021FFF	GPIO1 Control Register	4K
0x03022000	0x03022FFF	GPIO2 Control Register	4K
0x03023000	0x03023FFF	GPIO3 Control Register	4K
0x03024000	0x0302FFFF	reservations	
0x03030000	0x03030FFF	WGN0 Control Register	4K
0x03031000	0x03031FFF	WGN1 Control Register	4K
0x03032000	0x03032FFF	WGN2 Control Register	4K
0x03033000	0x0303FFFF	reservations	
0x03040000	0x0304FFFF	KEYSCAN Control Register	64K
0x03050000	0x0305FFFF	EFUSE Control Register	64K
0x03060000	0x03060FFF	PWM0 Control Register	4K
0x03061000	0x03061FFF	PWM1 Control Register	4K
0x03062000	0x03062FFF	PWM2 Control Register	4K
0x03063000	0x03063FFF	PWM3 Control Register	4K
0x03064000	0x0309FFFF	reservations	

0x030A0000	0x030AFFFF	TIMER Control Register	64K
0x030C0000	0x030CFFFF	reservations	
0x030D0000	0x030D0FFF	reservations	4K
0x030D1000	0x030D1FFF	reservations	4K
0x030D2000	0x030D2FFF	reservations	4K
0x030D3000	0x030DFFFF	reservations	
0x030E0000	0x030EFFFF	TEMPSEN Control Register	64K
0x030F0000	0x030FFFFF	SARADC Control Register	64K
0x04000000	0x0400FFFF	I2C0 Control Register	64K
0x04010000	0x0401FFFF	I2C1 Control Register	64K
0x04020000	0x0402FFFF	I2C2 Control Register	64K
0x04030000	0x0403FFFF	I2C3 Control Register	64K
0x04040000	0x0404FFFF	I2C4 Control Register	64K
0x04050000	0x0405FFFF	reservations	
0x04060000	0x0406FFFF	SPI_NAND Control Register	64K
0x04070000	0x0407FFFF	ETH0 Control Register	
0x04080000	0x040FFFFF	reservations	
0x04100000	0x04107FFF	I2S0 Control Register	64K
0x04108000	0x0410FFFF	I2S Global Control Register	64K
0x04110000	0x0411FFFF	I2S1 Control Register	64K
0x04120000	0x0412FFFF	I2S2 Control Register	64K
0x04130000	0x0413FFFF	I2S3 Control Register	64K
0x04140000	0x0414FFFF	UART0 Control Register	64K
0x04150000	0x0415FFFF	UART1 Control Register	64K
0x04160000	0x0416FFFF	UART2 Control Register	64K
0x04170000	0x0417FFFF	UART3 Control Register	64K
0x04180000	0x0418FFFF	SPI0 Control Register	64K
0x04190000	0x0419FFFF	SPI1 Control Register	64K
0x041A0000	0x041AFFFF	SPI2 Control Register	64K
0x041B0000	0x041BFFFF	SPI3 Control Register	64K
0x041C0000	0x041CFFFF	UART4 Control Register	64K
0x041D0000	0x041DFFFF	AUDSRC Control Register	64K
0x041E0000	0x042FFFFF	reservations	
0x04300000	0x0430FFFF	eMMC Control Registers	64K
0x04310000	0x0431FFFF	SD0 Control Register	64K
0x04320000	0x0432FFFF	SD1 Control Register	
0x04330000	0x0433FFFF	DMA Control Register	64K
0x04340000	0x0434FFFF	USB Control Register	64K
0x04350000	0x043FFFFF	reservations	
0x04400000	0x0441FFFF	ROM memory space	128K
0x04420000	0x04FFFFFF	reservations	

0x05000000	0x05000FFF	reservations	4KB
0x05020000	0x05020FFF	RTCSYS_Timer Control Registers	4KB
0x05021000	0x05021FFF	RTCSYS_GPIO Control Registers	4KB
0x05022000	0x05022FFF	RTCSYS_UART Control Registers	4KB
0x05023000	0x05023FFF	RTCSYS_INTR Control Registers	4KB
0x05024000	0x05024FFF	RTCSYS_MBOX Control Registers	4KB
0x05025000	0x05025FFF	RTCSYS_CTRL Control Registers	4KB
0x05026000	0x05026FFF	RTCSYS_CORE	4KB
0x05027000	0x05027FFF	RTCSYS_IO Control Registers	4KB
0x05028000	0x05028FFF	RTCSYS_OSC Control Registers	4KB
0x05029000	0x05029FFF	reservations	4KB
0x0502A000	0x0502AFFF	RTCSYS_32kless Control Registers	4KB
0x0502B000	0x0502BFFF	RTCSYS_I2C Control Registers	4KB
0x0502C000	0x0502CFFF	RTCSYS_SAR Control Registers	4KB
0x0502D000	0x0502DFFF	RTCSYS_WDT Control Registers	4KB
0x0502E000	0x0502EFFF	RTCSYS_IRRX Control Registers	4KB
0x05200000	0x053FFFFF	RTCSYS_SRAM	8KB
0x05400000	0x057FFFFF	RTCSYS_SPINOR	4MB
0x08000000	0x08001FFF	reservations	8K
0x08004000	0x08005FFF	DDR Controler Control Register	8K
0x08006000	0x08007FFF	reservations	8K
0x08008000	0x08009FFF	DDR AXI Monitor Control Registers	8K
0x0800A000	0x0800BFFF	DDR Global Control Register	8K
0x08010000	0x08011FFF	reservations	8K
0x08012000	0x08013FFF	reservations	8K
0x08014000	0x09FFFFFF	reservations	
0x0A000000	0x0A07FFFF	ISP Control Register	512K
0x0A080000	0x0A0803FF	sc_top Control Register	1K
0x0A080400	0x0A080BFF	reservations	2K
0x0A080C00	0x0A080CFF	osd enc control register	256B
0x0A080D00	0x0A080FFF	reservations	768B
0x0A081000	0x0A081FFF	reservations	4K
0x0A082000	0x0A082FFF	img_v Control Register	4K
0x0A083000	0x0A083FFF	img_d control register	4K
0x0A084000	0x0A084FFF	sc_d Control Register	4K
0x0A085000	0x0A085FFF	sc_v1 Control register	4K
0x0A086000	0x0A086FFF	sc_v2 Control Register	4K
0x0A087000	0x0A087FFF	sc_v3 Control Register	4K
0x0A088000	0x0A088FFF	DISP Control Register	4K
0x0A089000	0x0A089FFF	reservations	4K
0x0A08A000	0x0A08AFFF	dsi_mac Control Register	4K

0x0A08B000	0x0A08BFFF	cmdq control register	4K
0x0A08C000	0x0A08CFFF	reservations	4K
0x0A08D000	0x0A08DFFF	reservations	4K
0x0A08E000	0x0A09FFFF	reservations	72K
0x0A0A0000	0x0A0AFFFF	IVE Control Register	64K
0x0A0A0000	0x0A0BFFFF	reservations	64K
0x0A0C0000	0x0A0C1FFF	ldc control register	8K
0x0A0C2000	0x0A0C3FFF	VI0/MIPI_RX0 Control Registers	8K
0x0A0C4000	0x0A0C5FFF	VI1/MIPI_RX1 Control Registers	8K
0x0A0C6000	0x0A0C7FFF	VI2/MIPI_RX2 Control Registers	8K
0x0A0C8000	0x0A0C9FFF	VIPSYS Control Registers	8K
0x0A0CA000	0x0A0CFFFF	reservations	24K
0x0A0D0000	0x0A0D0FFF	CSI_PHY Control Registers	4K
0x0A0D1000	0x0A0D1FFF	DSI_PHY Control Registers	4K
0x0A0D2000	0x0AFFFFFF	reservations	
0x0B000000	0x0B00FFFF	JPEG codec control register	64K
0x0B010000	0x0B01FFFF	H.264 codec control registers	64K
0x0B020000	0x0B02FFFF	H.265 codec control registers	64K
0x0B030000	0x0BFFFFFF	reservations	
0x0C000000	0x0FFFFFFF	reservations	
0x10000000	0x1FFFFFFF	SPI_NOR Memory Space	256M
0x30000000	0x7FFFFFFF	reservations	
0x80000000	0xFFFFFFFF	DDR memory space	2G

*Read and write operations to reserved address space may produce unintended results.

3.4 Packages and Pinouts

3.4.1 seal inside

QFN package size is 9mmx9mmx0.9mm with 0.35mm pin spacing. 88 pins. Please refer to the figure below for detailed package dimensions.

		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.8	0.85	0.9
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.7	---
L/F THICKNESS		A3	0.152 REF		
LEAD WIDTH		b	0.13	0.18	0.23
BODY SIZE	X	D	9 BSC		
	Y	E	9 BSC		
LEAD PITCH		e	0.35 BSC		
EP SIZE	X	J	7.6	7.7	7.8
	Y	K	7.6	7.7	7.8
LEAD LENGTH		L	0.35	0.4	0.45
PACKAGE EDGE TOLERANCE		aaa	0.1		
MOLD FLATNESS		bbb	0.1		
COPLANARITY		ccc	0.08		
LEAD OFFSET		ddd	0.1		
EXPOSED PAD OFFSET		eee	0.1		

Figure *3-1* Package Dimensions, Top View

		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.8	0.85	0.9
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.7	---
L/F THICKNESS		A3	0.152 REF		
LEAD WIDTH		b	0.13	0.18	0.23
BODY SIZE	X	D	9 BSC		
	Y	E	9 BSC		
LEAD PITCH		e	0.35 BSC		
EP SIZE	X	J	7.6	7.7	7.8
	Y	K	7.6	7.7	7.8
LEAD LENGTH		L	0.35	0.4	0.45
PACKAGE EDGE TOLERANCE		aaa	0.1		
MOLD FLATNESS		bbb	0.1		
COPLANARITY		ccc	0.08		
LEAD OFFSET		ddd	0.1		
EXPOSED PAD OFFSET		eee	0.1		

Figure *3-2*. Package Dimensions, Bottom View

Pinouts

Figure *3-3* Pinouts.

3.4.2 Pin Information Description

Table *3-4* Pin Information Table

Pin Num	Pin Name	IO Type	IOGroup	PowerDomain	Note
6	SD0_CLK	18OD33 IO	G10	VDDIO_SD0_EMMC	
7	SD0_CMD	18OD33 IO	G10	VDDIO_SD0_EMMC	
8	SD0_D0	18OD33 IO	G10	VDDIO_SD0_EMMC	

10	SD0_D1	18OD33 IO	G10	VDDIO_SD0_EMMC	
11	SD0_D2	18OD33 IO	G10	VDDIO_SD0_EMMC	

12	SD0_D3	18OD33 IO	G10	VDDIO_SD0_EMMC	
14	SD0_CD	18OD33 IO	G7	VDDIO_SD0_EMMC	
15	SD0_PWR_EN	18OD33 IO	G7	VDDIO_SD0_EMMC	
17	SPK_EN	18OD33 IO	G7	VDDIO_SD0_EMMC	
18	UART0_TX	18OD33 IO	G7	VDDIO_SD0_EMMC	
19	UART0_RX	18OD33 IO	G7	VDDIO_SD0_EMMC	
20	EMMC_DAT2	18OD33 IO	G7	VDDIO_SD0_EMMC	
21	EMMC_CLK	18OD33 IO	G7	VDDIO_SD0_EMMC	
22	EMMC_DAT0	18OD33 IO	G7	VDDIO_SD0_EMMC	
23	EMMC_DAT3	18OD33 IO	G7	VDDIO_SD0_EMMC	
24	EMMC_CMD	18OD33 IO	G7	VDDIO_SD0_EMMC	
25	EMMC_DAT1	18OD33 IO	G7	VDDIO_SD0_EMMC	
26	JTAG_CPU_TMS	18OD33 IO	G7	VDDIO_SD0_EMMC	
27	JTAG_CPU_TCK	18OD33 IO	G7	VDDIO_SD0_EMMC	
28	IIC0_SCL	18OD33 IO	G7	VDDIO_SD0_EMMC	
29	IIC0_SDA	18OD33 IO	G7	VDDIO_SD0_EMMC	
30	AUX0	18OD33 IO	G7	VDDIO_SD0_EMMC	
38	PWR_VBAT_DET	1.8V GPIO	GRTC	VDDIO_RTC	
39	PWR_RSTN	1.8V GPIO	GRTC	VDDIO_RTC	
40	PWR_SEQ1	1.8V GPIO	GRTC	VDDIO_RTC	
41	PWR_SEQ2	1.8V GPIO	GRTC	VDDIO_RTC	
43	PWR_WAKEUP0	1.8V GPIO	GRTC	VDDIO_RTC	
44	PWR_BUTTON1	1.8V GPIO	GRTC	VDDIO_RTC	
45	XTAL_XIN	Xtal	GRTC	VDDIO_RTC	
47	PWR_GPIO0	1.8V GPIO	GRTC	VDDIO_RTC	
48	PWR_GPIO1	1.8V GPIO	GRTC	VDDIO_RTC	

49	PWR_GPIO2	1.8V GPIO	GRTC	VDDIO_RTC	
51	SD1_D3	18OD33 IO	GRTC	VDDIO_SD1	SD1_D3 or VO[32] Check 0x0502_70E4
52	SD1_D2	18OD33 IO	GRTC	VDDIO_SD1	SD1_D2 or VO[33] Check 0x0502_70E4
53	SD1_D1	18OD33 IO	GRTC	VDDIO_SD1	SD1_D1 or VO[34] Check 0x0502_70E4
54	SD1_D0	18OD33 IO	GRTC	VDDIO_SD1	SD1_D0 or VO[35] Check 0x0502_70E4
55	SD1_CMD	18OD33 IO	GRTC	VDDIO_SD1	SD1_CMD or VO[36] Check 0x0502_70E4
56	SD1_CLK	18OD33 IO	GRTC	VDDIO_SD1	sd1_clk or vo[37] Check 0x0502_70E4
58	PWM0_BUCK	1.8V GPIO	G1	vdd18a_usb_pll_e th	
59	ADC1	1.8V GPIO	G1	vdd18a_usb_pll_e th	
60	USB_VBUS_DET	1.8V GPIO	G1	vdd18a_usb_pll_e th	
62	PAD_ETH_TXP EPHY_R XN	ETH GPIO (1.8V)		vdd18a_usb_pll_e th	
63	PAD_ETH_TXM EPHY_R XP	ETH GPIO (1.8V)		vdd18a_usb_pll_e th	
64	PAD_ETH_RXP EPHY_T XN	ETH GPIO (1.8V)		vdd18a_usb_pll_e th	
65	PAD_ETH_RXM EPHY_T XP	ETH GPIO (1.8V)		vdd18a_usb_pll_e th	
72	PAD_MIPIRX4N	1.8V GPIO	G12	VDD18A_MIPI	
73	PAD_MIPIRX4P	1.8V GPIO	G12	VDD18A_MIPI	
74	PAD_MIPIRX3N	1.8V GPIO	G12	VDD18A_MIPI	
75	PAD_MIPIRX3P	1.8V GPIO	G12	VDD18A_MIPI	

76	PAD_MIPIRX2N	1.8V GPIO	G12	VDD18A_MIPI	
77	PAD_MIPIRX2P	1.8V GPIO	G12	VDD18A_MIPI	
78	PAD_MIPIRX1N	1.8V GPIO	G12	VDD18A_MIPI	
79	PAD_MIPIRX1P	1.8V GPIO	G12	VDD18A_MIPI	
80	PAD_MIPIRX0N	1.8V GPIO	G12	VDD18A_MIPI	
81	PAD_MIPIRX0P	1.8V GPIO	G12	VDD18A_MIPI	
83	PAD_MIPI_TXM2	1.8V GPIO	G12	VDD18A_MIPI	
84	PAD_MIPI_TXP2	1.8V GPIO	G12	VDD18A_MIPI	
85	PAD_MIPI_TXM1	1.8V GPIO	G12	VDD18A_MIPI	
86	PAD_MIPI_TXP1	1.8V GPIO	G12	VDD18A_MIPI	
87	PAD_MIPI_TXM0	1.8V GPIO	G12	VDD18A_MIPI	
88	PAD_MIPI_TXP0	1.8V GPIO	G12	VDD18A_MIPI	
2	PAD_AUD_AINL_MIC	AUDIO GPIO (1.8V)		VDD18A_MIPI	
4	PAD_AUD_AOUTR	AUDIO GPIO (1.8V)		VDD18A_MIPI	
67	GPIO_RTX EPHY_RTX	1.8V GPIO	G12	vdd18a_usb_pll_e th	
35	GPIO_ZQ PAD_ZQ	1.8V GPIO	GRTC	VDDIO_RTC	
#N/A	PKG_TYPE0	1.8V GPIO	G1	vdd18a_usb_pll_e th	no pin out
#N/A	PKG_TYPE1	1.8V GPIO	G1	vdd18a_usb_pll_e th	no pin out
#N/A	PKG_TYPE2	1.8V GPIO	G1	vdd18a_usb_pll_e th	no pin out
#N/A	MUX_SPI1_MISO	NA	NA	NA	Result is feed to PAD_MIPIRX3N func7
#N/A	MUX_SPI1_MOSI	NA	NA	NA	Result is feed to PAD_MIPIRX3P func7
#N/A	MUX_SPI1_CS	NA	NA	NA	Result is feed to PAD_MIPIRX4P func7

#N/A	MUX_SPI1_SCK	NA	NA	NA	Result is feed to PAD_MIPIRX4N func7

3.4.3 Pin Default Status (QFN)

Table *3-5* Pin Default Status Table

PinNum	PinName	Default Function	Default Function Select	Default Pull (PU/PD/NO PULL)	Default Function Direction (During reset)	IO TRAP (become input during reset)	IO Type	PowerDomain	IOBLK_Group	Fail-safe
6	SD0_CLK	SDIO0_CLK	0	PD (61Kohm)->DRV_LO + PD	Output (Low)		18OD33	VDDIO_SD0_EMMC	G10	
7	SD0_CMD	SDIO0_CMD	0	PD (61Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G10	
8	SD0_D0	SDIO0_D[0]	0	PD (61Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G10	
10	SD0_D1	SDIO0_D[1]	0	PD (61Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G10	
11	SD0_D2	SDIO0_D[2]	0	PD (61Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G10	
12	SD0_D3	SDIO0_D[3]	0	PD (61Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G10	
14	SD0_CD	SDIO0_CD	0	PU (60Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G7	
15	SD0_PWR_EN	XGPIOA[14]	3	PD (61Kohm) -> DRV_HI/LO + PD	Input (Trap) -> Output	V	18OD33	VDDIO_SD0_EMMC	G7	
17	SPK_EN	XGPIOA[15]	3	PD (61Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G7	
18	UART0_TX	UART0_TX	0	PU (60Kohm) -> DRV_HI + PU	Input (Trap) -> Output	V	18OD33	VDDIO_SD0_EMMC	G7	
19	UART0_RX	UART0_RX	0	PU (60Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G7	
20	EMMC_DAT2	SPINOR_HOLD_X	1	PD (61Kohm) -> DRV_HI + PD	Input		18OD33	VDDIO_SD0_EMMC	G7	
21	EMMC_CLK	SPINOR_SCK	1	DRV_LO + PD (61Kohm)	Output (Low)		18OD33	VDDIO_SD0_EMMC	G7	
22	EMMC_DAT0	SPINOR_MOSI	1	PU (60Kohm)	Input (Trap)	V	18OD33	VDDIO_SD0_EMMC	G7	
23	EMMC_DAT3	SPINOR_WP_X	1	PU (60Kohm) -> DRV_HI + PU	Input (Trap)	V	18OD33	VDDIO_SD0_EMMC	G7	
24	EMMC_CMD	SPINOR_MISO	1	PU (60Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G7	
25	EMMC_DAT1	SPINOR_CS_X	1	DRV_HI + PD (61Kohm)	Output (High)		18OD33	VDDIO_SD0_EMMC	G7	
26	JTAG_CPU_TMS	CR_4WTMS	0	PD (61Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G7	
27	JTAG_CPU_TCK	CR_4WTCK	0	PD (61Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G7	
28	IIC0_SCL	CR_4WTDI	0	PU (60Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G7	
29	IIC0_SDA	CR_4WTDO	0	PU (60Kohm)	Output (CR)		18OD33	VDDIO_SD0_EMMC	G7	
30	AUX0	XGPIOA[30]	3	PD (61Kohm)	Input		18OD33	VDDIO_SD0_EMMC	G7	
38	PWR_VBAT_DET	PWR_VBAT_DET	0	NO PULL	Input		1.8V_IO	VDDIO_RTC	GRTC	
39	PWR_RSTN	PWR_RSTN	0	PU (60Kohm)	Input		1.8V_IO	VDDIO_RTC	GRTC	Fail-safe
40	PWR_SEQ1	PWR_SEQ1	0	PD (87Kohm) -> DRV + PD	Output (L->H)		1.8V_IO	VDDIO_RTC	GRTC	Fail-safe
41	PWR_SEQ2	PWR_SEQ2	0	PD (87Kohm) -> DRV + PD	Output (L->H)		1.8V_IO	VDDIO_RTC	GRTC	
43	PWR_WAKEUP0	PWR_WAKEUP0	0	PD (87Kohm)	Input		1.8V_IO	VDDIO_RTC	GRTC	Fail-safe
44	PWR_BUTTON1	PWR_BUTTON1	0	PU (60Kohm)	Input		1.8V_IO	VDDIO_RTC	GRTC	Fail-safe
47	PWR_GPIO0	PWR_GPIO[0]	0	PD (87Kohm)	Input		1.8V_IO	VDDIO_RTC	GRTC	Fail-safe
48	PWR_GPIO1	PWR_GPIO[1]	0	PD (87Kohm)	Input		1.8V_IO	VDDIO_RTC	GRTC	
49	PWR_GPIO2	PWR_GPIO[2]	0	PD (87Kohm)	Input		1.8V_IO	VDDIO_RTC	GRTC	
51	SD1_D3	PWR_SPINOR1_CS_X	6	DRV_HI + PD (61Kohm)	Output (High)		18OD33	VDDIO_SD1	GRTC	
52	SD1_D2	PWR_SPINOR1_HOLD_X	6	PD (61Kohm) -> DRV_HI + PD	Input		18OD33	VDDIO_SD1	GRTC	
53	SD1_D1	PWR_SPINOR1_WP_X	6	PD (61Kohm) -> DRV_HI + PD	Input		18OD33	VDDIO_SD1	GRTC	
54	SD1_D0	PWR_SPINOR1_MISO	6	PD (61Kohm)	Input		18OD33	VDDIO_SD1	GRTC	
55	SD1_CMD	PWR_SPINOR1_MOSI	6	PD (61Kohm)	Input		18OD33	VDDIO_SD1	GRTC	
56	SD1_CLK	PWR_SPINOR1_SCK	6	DRV_LO + PD (61Kohm)	Output (Low)		18OD33	VDDIO_SD1	GRTC	
58	PWM0_BUCK	XGPIOB[0]	3	NO PULL	Input		1.8V_IO	VDD18A_USB_PLL_ETH	G1	Fail-safe
59	ADC1	XGPIOB[3]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_USB_PLL_ETH	G1	Fail-safe
60	USB_VBUS_DET	USB_VBUS_DET	0	NO PULL	Input		1.8V_IO	VDD18A_USB_PLL_ETH	G1	
62	PAD_ETH_TXP___EPHY_RXN	XGPIOB[25]	3	NO PULL			ETH_GPIO	VDD18A_USB_PLL_ETH	G12	
63	PAD_ETH_TXN___EPHY_RXP	XGPIOB[24]	3	NO PULL			ETH_GPIO	VDD18A_USB_PLL_ETH	G12	
64	PAD_ETH_RXP___EPHY_TXN	XGPIOB[27]	3	NO PULL			ETH_GPIO	VDD18A_USB_PLL_ETH	G12	
65	PAD_ETH_RXN___EPHY_TXP	XGPIOB[26]	3	NO PULL			ETH_GPIO	VDD18A_USB_PLL_ETH	G12	
72	PAD_MIPIRX4N	CR_SCL0	0	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
73	PAD_MIPIRX4P	CR_SDA0	0	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
74	PAD_MIPIRX3N	XGPIOC[4]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
75	PAD_MIPIRX3P	XGPIOC[5]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
76	PAD_MIPIRX2N	XGPIOC[6]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
77	PAD_MIPIRX2P	XGPIOC[7]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
78	PAD_MIPIRX1N	XGPIOC[8]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
79	PAD_MIPIRX1P	XGPIOC[9]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
80	PAD_MIPIRX0N	XGPIOC[10]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
81	PAD_MIPIRX0P	XGPIOC[11]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
83	PAD_MIPI_TXM2	XGPIOC[16]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
84	PAD_MIPI_TXP2	XGPIOC[17]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
85	PAD_MIPI_TXM1	XGPIOC[14]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
86	PAD_MIPI_TXP1	XGPIOC[15]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
87	PAD_MIPI_TXM0	XGPIOC[12]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
88	PAD_MIPI_TXP0	XGPIOC[13]	3	PD (87Kohm)	Input		1.8V_IO	VDD18A_MIPI	G12	
2	PAD_AUD_AINL_MIC	XGPIOC[23]	3	NO PULL	Input		AUDIO_GPIO	VDD18A_MIPI	G11	
4	PAD_AUD_AOUTR	XGPIOC[24]	3	NO PULL	Output		AUDIO_GPIO	VDD18A_MIPI	G11	
67	GPIO_RTX___EPHY_RTX	XGPIOB[23]	3	PD (87Kohm)	Input	V	1.8V_IO	VDD18A_USB_PLL_ETH	G12	Fail-safe
35	GPIO_ZQ___PAD_ZQ	PWR_GPIO[24]	3	PD (87Kohm)	Input	V	1.8V_IO	VDDIO_RTC	GRTC	Fail-safe

4 Startup and Upgrade

4.1 summarize

The chip is booted from the internal ROM (BOOTROM). During reset, the chip detects the presence of weak pull-ups or pull-downs on two pins (EMMC_DAT3, EMMC_DAT0) to confirm the type of memory device currently selected for use.
For secure boot chips, a signature check is performed during boot-up and chip upgrades to ensure that the software being executed or upgraded is secure.

4.2 Correspondence between the startup mode and the corresponding signal latch value

- Supports booting from SPI Nor Flash (EMMC_DAT3 pull down, EMMC_DAT0 pull up)
- Supports booting from SPI Nand Flash (EMMC_DAT3 pull down, EMMC_DAT0 pull down).
- Supported by eMMC boot. (EMMC_DAT3 pull up, EMMC_DAT0 pull up)
  EMMC_DAT0 pull up)

Note. Because SD0 shares IO power with the eMMC domain. Because SD cards default 3.0V, while eMMC is more
1.8V. So basically no eMMC support unless SD0 is not connected to an SD card.

4.3 Mirror burn mode.

- Supports burning images via SD card.
- Supports burning images via USB device mode.
- If there is an image in the flash, the software support will be upgraded by the Internet.

4.4 Secure Start

- Supports secure startup and upgrades.
- AES/DES/SM4 hardware encryption and decryption.

- SHA/TRNG/Secure Efuse security hardware.

5 reset (a dislocated joint, an electronic device etc)

5.1 summarize

The reset management module provides unified management of the reset and timing of the entire chip, subsystems, and functional modules.

5.2 reset control

Figure *5-1* Reset Management Module Block Diagram

5.2.1 Power-on reset

The power-on reset (POR) is generated by the real-time clock module in conjunction with global power management and crystal timing. Refer to Section 12 for details.

5.2.2 System hard reset

System Hard Reset (System Hard Reset) is generated by Reset Ctrl Level2, which performs a hard reset of the chip's global including subsystems and each functional module, sourced from the following paths:

- Power-on reset
- watchdog reset
- Overheat protection reset

- External reset pin (RSTN)
  - Built-in de-jittering circuit, RSTN high and low valid signals must be up to 6.56ms.

5.2.3 soft reset (electronics)

Soft reset control is achieved by configuring the corresponding Reset Configuration Register (Reset CRG), implemented by Reset Ctrl Level 3, contains:

- System Soft Reset : resets the entire chip, except for a few circuits and internal RTC circuits
- Processor Subsystem Reset : Reset the processor and processor subsystem.
- Functional subsystem reset : Reset each functional subsystem and functional module
- Function Module Reset : Reset each function module

5.2.4 Processor subsystem soft reset

Operation register SOFT_AC_RSTN_0 can be used to soft reset the processor and subsystem. After writing 0 to the configuration register, the reset controller will wait for 24us delay time before triggering the reset of the corresponding processor. During this period, the processor should terminate the access to the bus in order to avoid the bus hanging after reset. After the reset is triggered, the corresponding reset signal will last for 8us, and the processor and processor subsystem will be reset and start to boot.

5.2.5 Function subsystem and function module soft reset

Operation registers SOFT_RSTN_0~3 can be used to soft reset each function module. The reset signal will not be cleared automatically if the reset is configured to be active low, therefore, after the software configures the corresponding register to be 0 to trigger the reset, it should also be configured to be 1 to release the reset. Before reset, make sure that the built-in DMA access to the bus and the processor access to the module are in idle state. Otherwise, the reset will fail and cause the system to hang.

5.3 Reset Configuration Register

5.3.1 Reset Configuration Register Overview

Base address 0x03003000

Name	Address Offset	Description
SOFT_RSTN_0	0x000	soft-reset ctrl register 0
SOFT_RSTN_1	0x004	soft-reset ctrl register 1
SOFT_RSTN_2	0x008	soft-reset ctrl register 2
SOFT_RSTN_3	0x00c	soft-reset ctrl register 3
SOFT_CPUAC_RSTN	0x020	CPU auto clear soft-reset ctrl register
SOFT_CPU_RSTN	0x024	CPU soft-reset ctrl register

5.3.2 Reset Configuration Register Description

SOFT_RSTN_0

Offset Address: 0x000

Bits	Name	Access	Description	Reset
1:0	Reserved			
2	reg_soft_reset_x_ddr	R/W	DDR system software reset (active low)	0x1
3	reg_soft_reset_x_h264c	R/W	H264 IP software reset (active low)	0x1
4	reg_soft_reset_x_jpeg	R/W	JPEG IP software reset (active low)	0x1
5	reg_soft_reset_x_h265c	R/W	H265 IP software reset (active low)	0x1
6	reg_soft_reset_x_vipsys	R/W	VIP system software reset (active low)	0x1
7	reg_soft_reset_x_tdma	R/W	TPU_DMA IP software reset (active low)	0x1
8	reg_soft_reset_x_tpu	R/W	TPU IP software reset (active low)	0x1
9	reg_soft_reset_x_tpusys	R/W	TPU system software reset (active low)	0x1
10	Reserved			
11	reg_soft_reset_x_usb	R/W	USB IP software reset (active low)	0x1
12	reg_soft_reset_x_eth0	R/W	ETH0 IP software reset (active low)	0x1
13	reg_soft_reset_x_eth1	R/W	ETH1 IP software reset (active low)	0x1
14	reg_soft_reset_x_nand	R/W	NAND IP software reset (active low)	0x1
15	reg_soft_reset_x_emmc	R/W	EMMC IP software reset (active low)	0x1
16	reg_soft_reset_x_sd0	R/W	SD0 IP software reset (active low)	0x1
17	Reserved			
18	reg_soft_reset_x_sdma	R/W	SDMA IP software reset (active low)	0x1
19	reg_soft_reset_x_i2s0	R/W	I2S0 IP software reset (active low)	0x1
20	reg_soft_reset_x_i2s1	R/W	I2S1 IP software reset (active low)	0x1
21	reg_soft_reset_x_i2s2	R/W	I2S2 IP software reset (active low)	0x1
22	reg_soft_reset_x_i2s3	R/W	I2S3 IP software reset (active low)	0x1
23	reg_soft_reset_x_uart0	R/W	UART0 IP software reset (active low)	0x1
24	reg_soft_reset_x_uart1	R/W	UART1 IP software reset (active low)	0x1
25	reg_soft_reset_x_uart2	R/W	UART2 IP software reset (active low)	0x1
26	reg_soft_reset_x_uart3	R/W	UART3 IP software reset (active low)	0x1
27	reg_soft_reset_x_i2c0	R/W	I2C0 IP software reset (active low)	0x1
28	reg_soft_reset_x_i2c1	R/W	I2C1 IP software reset (active low)	0x1
29	reg_soft_reset_x_i2c2	R/W	I2C2 IP software reset (active low)	0x1
30	reg_soft_reset_x_i2c3	R/W	I2C3 IP software reset (active low)	0x1
31	reg_soft_reset_x_i2c4	R/W	I2C4 IP software reset (active low)	0x1

SOFT_RSTN_1

Offset Address: 0x004

Bits	Name	Access	Description	Reset
0	reg_soft_reset_x_pwm0	R/W	PWM0 IP software reset (active low)	0x1
1	reg_soft_reset_x_pwm1	R/W	PWM1 IP software reset (active low)	0x1
2	reg_soft_reset_x_pwm2	R/W	PWM2 IP software reset (active low)	0x1
3	reg_soft_reset_x_pwm3	R/W	PWM3 IP software reset (active low)	0x1
7:4	Reserved			
8	reg_soft_reset_x_spi0	R/W	SPI0 IP software reset (active low)	0x1
9	reg_soft_reset_x_spi1	R/W	SPI1 IP software reset (active low)	0x1
10	reg_soft_reset_x_spi2	R/W	SPI2 IP software reset (active low)	0x1
11	reg_soft_reset_x_spi3	R/W	SPI3 IP software reset (active low)	0x1
12	reg_soft_reset_x_gpio0	R/W	GPIO0 IP software reset (active low)	0x1
13	reg_soft_reset_x_gpio1	R/W	GPIO1 IP software reset (active low)	0x1
14	reg_soft_reset_x_gpio2	R/W	GPIO2 IP software reset (active low)	0x1
15	reg_soft_reset_x_efuse	R/W	EFUSE IP software reset (active low)	0x1
16	reg_soft_reset_x_wdt	R/W	WDT0 IP software reset (active low)	0x1
17	reg_soft_reset_x_ahb_rom	R/W	ROM IP software reset (active low)	0x1
18	reg_soft_reset_x_spic	R/W	SPIC IP software reset (active low)	0x1
19	reg_soft_reset_x_tempsen	R/W	TEMPSEN IP software reset (active low)	0x1
20	reg_soft_reset_x_saradc	R/W	SARADC IP software reset (active low)	0x1
25:21	Reserved			

Bits	Name	Access	Description	Reset
26	reg_soft_reset_x_combo_phy0	R/W	USB_PHY IP software reset (active low)	0x1
28:27	Reserved			
29	reg_soft_reset_x_spi_nand	R/W	NAND IP software reset (active low)	0x1
30	reg_soft_reset_x_se	R/W	SE IP software reset (active low)	0x1
31	Reserved			

SOFT_RSTN_2

Offset Address: 0x008

Bits	Name	Access	Description	Reset
9:0	Reserved			
10	reg_soft_reset_x_uart4	R/W	UART4 IP software reset (active low)	0x1
11	reg_soft_reset_x_gpio3	R/W	GPIO3 IP software reset (active low)	0x1
12	reg_soft_reset_x_system	R/W	SYSTEM software reset (active low)	0x1
13	reg_soft_reset_x_timer	R/W	TIMER IP software reset (active low)	0x1
14	reg_soft_reset_x_timer0	R/W	TIMER0 IP software reset (active low)	0x1
15	reg_soft_reset_x_timer1	R/W	TIMER1 IP software reset (active low)	0x1
16	reg_soft_reset_x_timer2	R/W	TIMER2 IP software reset (active low)	0x1
17	reg_soft_reset_x_timer3	R/W	TIMER3 IP software reset (active low)	0x1
18	reg_soft_reset_x_timer4	R/W	TIMER4 IP software reset (active low)	0x1
19	reg_soft_reset_x_timer5	R/W	TIMER5 IP software reset (active low)	0x1
20	reg_soft_reset_x_timer6	R/W	TIMER6 IP software reset (active low)	0x1
21	reg_soft_reset_x_timer7	R/W	TIMER7 IP software reset (active low)	0x1
22	reg_soft_reset_x_wgn0	R/W	WGN0 IP software reset (active low)	0x1
23	reg_soft_reset_x_wgn1	R/W	WGN1 IP software reset (active low)	0x1
24	reg_soft_reset_x_wgn2	R/W	WGN2 IP software reset (active low)	0x1
25	reg_soft_reset_x_keyscan	R/W	KEYSCAN IP software reset (active low)	0x1
26	Reserved			
27	reg_soft_reset_x_auddac	R/W	AUDDAC IP software reset (active low)	0x1
28	reg_soft_reset_x_auddac_apb	R/W	AUDDAC APB software reset (active low)	0x1
29	reg_soft_reset_x_audadc	R/W	AUDADC IP software reset (active low)	0x1
30	Reserved			
31	reg_soft_reset_x_vcsys	R/W	VCSYS SYS software reset (active low)	0x1

SOFT_RSTN_3

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
0	reg_soft_reset_x_ethphy	R/W	ETHPHY IP software reset (active low)	0x1
1	reg_soft_reset_x_ethphy_apb	R/W	ETHPHY APB REG software reset (active) low)	0x1
2	reg_soft_reset_x_audsrc	R/W	AUDSRC IP software reset (active low)	0x1
3	reg_soft_reset_x_vip_cam0	R/W	VIP CAM0 IP software reset (active low)	0x1
4	reg_soft_reset_x_wdt1	R/W	WDT1 IP software reset (active low)	0x1
5	reg_soft_reset_x_wdt2	R/W	WDT2 IP software reset (active low)	0x1
31:6	Reserved			

SOFT_CPUAC_RSTN

Offset Address: 0x020

Write Lock: SOFT_CPUAC_RSTN_wr_lock

Bits	Name	Access	Description	Reset
0	reg_auto_clear_reset_x_cpucore0	R/W	CPUCORE0 auto_clear_reset (active low)	0x1
1	reg_auto_clear_reset_x_cpucore1	R/W	CPUCORE1 auto_clear_reset (active low)	0x1

Bits	Name	Access	Description	Reset
2	reg_auto_clear_reset_x_cpucore2	R/W	CPUCORE2 auto_clear_reset (active low)	0x1
3	reg_auto_clear_reset_x_cpucore3	R/W	CPUCORE3 auto_clear_reset (active low)	0x1
4	reg_auto_clear_reset_x_cpusys0	R/W	CPUSYS0 auto_clear_reset (active low)	0x1
5	reg_auto_clear_reset_x_cpusys1	R/W	CPUSYS1 auto_clear_reset (active low)	0x1
6	reg_auto_clear_reset_x_cpusys2	R/W	CPUSYS2 auto_clear_reset (active low)	0x1
31:7	Reserved			

SOFT_CPU_RSTN

Offset Address: 0x024

Bits	Name	Access	Description	Reset
0	reg_soft_reset_x_cpucore0	R/W	CPUCORE0 soft reset (active low)	0x0
1	reg_soft_reset_x_cpucore1	R/W	CPUCORE1 soft reset (active low)	0x0
2	reg_soft_reset_x_cpucore2	R/W	CPUCORE2 soft reset (active low)	0x0
3	reg_soft_reset_x_cpucore3	R/W	CPUCORE3 soft reset (active low)	0x0
4	reg_soft_reset_x_cpusys0	R/W	CPUSYS0 soft reset (active low)	0x0
5	reg_soft_reset_x_cpusys1	R/W	CPUSYS1 soft reset (active low)	0x0
6	reg_soft_reset_x_cpusys2	R/W	CPUSYS2 soft reset (active low)	0x0
31:7	Reserved			

6 clocks

6.1 summarize

The clock management module manages the chip clock, including:

- Management and control of clock inputs
- PLL clock source and associated doubling and dividing controls
- Clock division and control
- Generate operating clocks for each module
- Management and control of clock outputs

6.2 functional block diagram

Figure *6-1* Clock Management Block Diagram

XTAL_XIN is the PLL input clock with a fixed connection to the 25MHz crystal;
RTC_XIN is the RTC input clock with a fixed connection to the 32.768KHz crystal.

6.3 Clock resources and crossover structure

The system clock comes from external XTAL, PLL using XTAL multiplier, and external protocol input clock. As shown in the diagram

As shown in Figure 3-3, each IP generally has an XTAL, and clock feeds from one PLL or

algorithmic technology

more, each of which is passed through a frequency divider circuit.

to select as the IP or subsystem clock.

Figure *6-2* Clock Source Frequency Division Schematic.

6.4 PLL Configuration

As shown in Table 6-1, there are 9 built-in PLLs (not counting the built-in PLLs of Analog IP), which are categorized into two types of PLLs: integer multiplier and fractional multiplier.

Table *6-1 PLL* Configuration Parameters

PLL	configuration register	Power Down Control Register	preset frequency	PLL Patterns
FPLL	fpll_csr	fpll_pwd (default On)	1500MHz	integer frequency
MIPIMPLL	mipimpll_csr	mipimpll_pwd (default On)	900MHz	integer frequency

MPLL	mpll_csr mpll_ssc_syn_ctrl mpll_ssc_syn_set mpll_ssc_syn_span mpll_ssc_syn_setp	mpll_pwd (default On)	1000MHz	Integer/Fractional Multiplier
TPLL	tpll_csr tpll_ssc_syn_ctrl	tpll_pwd (default On)	1400MHz	Integer/Fractional Multiplier

PLL	configuration register	Power Down Control Register	preset frequency	PLL Patterns
	tpll_ssc_syn_set tpll_ssc_syn_span tpll_ssc_syn_setp			
APLL	apll_csr apll_ssc_syn_ctrl apll_ssc_syn_set apll_ssc_syn_span apll_ssc_syn_setp	apll_pwd(default On)	1050MHz	Integer/Fractional Multiplier
CAM0PLL	cam0pll_csr cam0pll_ssc_syn_ctrl cam0pll_ssc_syn_set cam0pll_ssc_syn_span cam0pll_ssc_syn_setp	cam0pll_pwd(default On)	1050MHz	Integer/Fractional Multiplier
CAM1PLL	cam1pll_csr cam1pll_ssc_syn_ctrl cam1pll_ssc_syn_set cam1pll_ssc_syn_span cam1pll_ssc_syn_setp	cam1pll_pwd(default On)	1025MHz	Integer/Fractional Multiplier
DISPPLL	disppll_csr disppll_ssc_syn_ctrl disppll_ssc_syn_set disppll_ssc_syn_span disppll_ssc_syn_setp	disppll_pwd(default On)	1200MHz	Integer/Fractional Multiplier

6.5 Integer Multiplier PLL

Adjust the integer PLL flow.

1. Adjustments must be made by turning off the relevant clock using this PLL or by selecting the clock source as XTAL or other PLLs.
2. Configure *_pll_csr register by integer PLL parameter table
3. Clear *_pll_pwd

Table 6-2 Integer *PLL* Configuration Parameters

PLL Parameters	realm	caveat
PLL_REF	25MHz~2500MHz	
PLL_VCO	800MHz~2500MHz	
Pre_div_sel	1~127	PLL_VCO = PLL_REF*Div_sel/Pre_div_sel
Div_sel	6~127	

Post_div_sel	1~127	PLL_FOUT = PLL_VCO/Post_div_sel
Ictrl	0~7	0.2< 1.84*(1+Mode)*(1+Ictrl)/2/Div_sel <=0.35

PLL Parameters	realm	caveat
Mode	0~3	

6.5.1 Fractional Multiplier PLL

Adjusting the fractional PLL flow.

1. When using this PLL, you must turn off the relevant IP clock or select XTAL or other stabilized clock source.
2. Configure *_ssc_syn_src_en to enable the synthesizer clock.
3. Configure *_ssc_syn_set according to the PLL frequency requirement , *_ssc_syn_set , *_ssc_syn_syn_set
4. Toggle *_ssc_syn_up ,make configuration effective
5. Configure *_pll_csr register by integer PLL parameter table
6. Clear *_pll_pwd

Table *6-3* Fractional *PLL* Parameter Configuration

PLL Parameters	realm	caveat
ssc_freq_in	DSIPLL: 1.2GHz DDRPLL: 1.5GHz Others 600MHz	Default clock gated , need enable
ssc_syn_set	> 4.x * 2^26	ssc_freq_in*div_sel*2^26/PLL_VCO
PLL_REF	100M~2500MHz	
PLL_VCO	800MHz~2500MHz	
Div_sel	6~127	PLL_VCO = PLL_REF*div_sel FOUT = PLL_VCO/Post_div_sel
Post_div_sel	1~127	
Ictrl	0~7	0.1< 1.84*(1+Mode)*(1+Ictrl)/2/Div_sel <=0.24
Mode	0~3	

6.6 CLK_DIV Clock Division Configuration

The table below shows the clock source and the preset clock frequency and division frequency of each clock that can be configured. The software can switch the clock source from XTAL to PLL and adjust the clock division frequency after boot.

Table *6-4* Clock Source and Preset Crossover Parameter Table

CLK_NAME	XTAL	DIV	SW	PLL SRC/DIV/FREQ	DIV_IN0 SRC	DIV_IN1 SRC

clk_a53	Y	Y	Y	fpll/(2)/750M	tpll	a0pll	mipimpll	mpll	fpll
clk_cpu_axi0	Y	Y		fpll/(3)/500M	fpll	disppll			
clk_cpu_gic	Y	Y		fpll/(5)/300M	fpll				

clk_tpu	Y	Y		fpll/(3)/500M	tpll	a0pll	mipimpll	fpll	
clk_emmc	Y	Y		fpll/(15)/100M	fpll	disppll			
clk_sd0	Y	Y		fpll/(15)/100M	fpll	disppll			
clk_sd1	Y	Y		fpll/(15)/100M	fpll	disppll			
clk_spi_nand	Y	Y		fpll/(8)/187.5M	fpll	disppll			
clk_sdma_aud0	Y	Y		a0pll/(18)/58.3M	a0pll	a24m			
clk_sdma_aud1	Y	Y		a0pll/(18)/58.3M	a0pll	a24m			
clk_sdma_aud2	Y	Y		a0pll/(18)/58.3M	a0pll	a24m			
clk_sdma_aud3	Y	Y		a0pll/(18)/58.3M	a0pll	a24m			
clk_pwm	Y	Y		fpll/(10)/150M	fpll	disppll			
clk_uart	Y	Y		xtal/(1)/25M	xtal	disppll			
clk_axi4	Y	Y		fpll/(5)/300M	fpll	disppll			
clk_axi6	Y	Y		fpll/(15)/100M	fpll				
clk_axi_vip	Y	Y		fpll/(6)/250M	mipimpll	cam0pll	disppll	fpll	
clk_src_vip_sys_0	Y	Y		fpll/(6)/250M	mipimpll	cam0pll	disppll	fpll	
clk_src_vip_sys_1	Y	Y		fpll/(5)/300M	mipimpll	cam0pll	disppll	fpll	
clk_axi_video_codec	Y	Y		mipimpll/(2)/450M	a0pll	mipimpll	cam1pll	fpll	
clk_vc_src0	Y	Y		mipimpll/(2)/450M	disppll	mipimpll	cam1pll	fpll	
clk_spi	Y	Y		fpll/(8)/187.5M	fpll				
clk_i2c	Y	Y		clk_axi6/(1)/100M	clk_axi6				
clk_src_vip_sys_2	Y	Y		disppll/(2)/600M	mipimpll	cam0pll	disppll	fpll	
clk_audsrc	Y	Y		a0pll/(18)/58.3M	a0pll	a24m			
clk_ap_debug	Y	Y		fpll/(5)/300M	fpll				
clk_src_rtc_sys_0	Y	Y		fpll/(5)/300M	fpll				
clk_c906_0	Y	Y	Y	fpll/(2)/750M	tpll	a0pll	mipimpll	mpll	fpll
clk_c906_1	Y	Y	Y	fpll/(3)/500M	tpll	a0pll	disppll	mpll	fpll
clk_src_vip_sys_3	Y	Y		mipimpll/(2)/450M	mipimpll	cam0pll	disppll	fpll	
clk_src_vip_sys_4	Y	Y		disppll/(3)/400M	mipimpll	cam0pll	disppll	fpll	

6.6.1 IP/SYS Clock source, clock division configuration

1. Turn off the IP clock; if the clock cannot be turned off, it should be configured to stabilize the crossover clock first. a. CPU Frequency: Configure clk_sel_0 to select the switch to SRC1 to avoid the frequency is too low. b. IP Frequency: Configure clk_byp_0/1 to switch the clock to XTAL. c. IP Frequency: Configure clk_byp_0/1 to switch the clock to XTAL.
2. Configure the clock source and divider configuration to be adjusted.

3. Divider register [3] needs to be configured for this clock divider configuration to take effect
4. Select the clock source to the configured clock divider.

6.6.2 MCLK0/MCLK1

1. MCLK0/MCLK1 is to provide an external sensor reference clock.
2. Configure CAM0PLL, clk_cam0_src_div, and clk_cam0_src_div to provide the proper MCLK0/MCLK1 Output frequency.

6.6.3 Clk_A24M

1. You can use clk_a24m as the audio clock if the performance is acceptable.
2. Output the desired audio clock source by configuring apll_frac_div_ctrl, apll_frac_div_m and apll_frac_div_n.
3. The frequency of clk_a24m is 900MHz * N/M/2.

6.7 PLL CRG Register Overview

PLL_G2 base address : 0x03002800

Name	Address Offset	Description
pll_g2_ctrl	0x000	Group2 PLL Ctrl register
pll_g2_status	0x004	Group2 PLL Status register
mipimpll_csr	0x008	MIPIMPLL Ctrl register
apll0_csr	0x00c	APLL0 Ctrl register
disppll_csr	0x010	DISPPLL Ctrl register
cam0pll_csr	0x014	CAM0PLL Ctrl register
cam1pll_csr	0x018	CAM1PLL Ctrl register
pll_g2_ssc_syn_ctrl	0x040	Group2 PLL Synthesizer ctrl register
apll_ssc_syn_ctrl	0x050	APLL synthesizer ctrl register
apll_ssc_syn_set	0x054	APLL synthesizer set register
disppll_ssc_syn_ctrl	0x060	DISPPLL synthesizer ctrl register
disppll_ssc_syn_set	0x064	DISPPLL synthesizer set register
cam0pll_ssc_syn_ctrl	0x070	CAM0PLL synthesizer ctrl register
cam0pll_ssc_syn_set	0x074	CAM0PLL synthesizer set register
cam1pll_ssc_syn_ctrl	0x080	CAM1PLL synthesizer ctrl register
cam1pll_ssc_syn_set	0x084	CAM1PLL synthesizer set register
apll_frac_div_ctrl	0x090	APLL frac divider ctrl register
apll_frac_div_m	0x094	APLL frac divider M parameter
apll_frac_div_n	0x098	APLL frac divider N parameter
mipimpll_clk_csr	0x0a0	MIPIMPLL clock Ctrl register
a0pll_clk_csr	0x0a4	a0pll clock Ctrl register
disppll_clk_csr	0x0a8	disppll clock Ctrl register
cam0pll_clk_csr	0x0ac	cam0pll clock Ctrl register
cam1pll_clk_csr	0x0b0	cam1pll clock Ctrl register
clk_cam0_src_div	0x0c0	clk_cam0_src_div
clk_cam1_src_div	0x0c4	clk_cam1_src_div

PLL_G6 base address : 0x03002900

Name	Address Offset	Description
pll_g6_ctrl	0x000	Group6 PLL Ctrl register
pll_g6_status	0x004	Group6 PLL Status register
mpll_csr	0x008	MPLL Ctrl register
tpll_csr	0x00c	TPLL Ctrl register
fpll_csr	0x010	FPLL Ctrl register
pll_g6_ssc_syn_ctrl	0x040	Group6 PLL Synthesizer ctrl register
dpll_ssc_syn_ctrl	0x050	dpll synthesizer ctrl register
dpll_ssc_syn_set	0x054	dpll synthesizer set register
dpll_ssc_syn_span	0x058	dpll synthesizer span register
dpll_ssc_syn_step	0x05c	dpll synthesizer step register
mpll_ssc_syn_ctrl	0x060	mpll synthesizer ctrl register
mpll_ssc_syn_set	0x064	mpll synthesizer set register
mpll_ssc_syn_span	0x068	mpll synthesizer span register
mpll_ssc_syn_step	0x06c	mpll synthesizer step register
tpll_ssc_syn_ctrl	0x070	tpll synthesizer ctrl register
tpll_ssc_syn_set	0x074	tpll synthesizer set register
tpll_ssc_syn_span	0x078	tpll synthesizer span register
tpll_ssc_syn_step	0x07c	tpll synthesizer step register

6.8 PLL CRG Register Description

6.8.1 PLL_G2 Register Description

pll_g2_ctrl

Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	mipimpll_pwd	R/W	pll power down	0x0
3:1	Reserved			
4	apll0_pwd	R/W	pll power down	0x0
7:5	Reserved			
8	disppll_pwd	R/W	pll power down	0x0
11:9	Reserved			
12	cam0pll_pwd	R/W	pll power down	0x0
15:13	Reserved			
16	cam1pll_pwd	R/W	pll power down	0x0
31:17	Reserved			

pll_g2_status

Offset Address: 0x004

Bits	Name	Access	Description	Reset
0	updating_mipimpll_val	RO	pll setting update status	
1	updating_apll0_val	RO	pll setting update status	
2	updating_disppll_val	RO	pll setting update status	
3	updating_cam0pll_val	RO	pll setting update status	

Bits	Name	Access	Description	Reset
4	updating_cam1pll_val	RO	pll setting update status	
15:5	Reserved			
16	mipimpll_lock	RO	pll lock status	
17	apll0_lock	RO	pll lock status	
18	disppll_lock	RO	pll lock status	
19	cam0pll_lock	RO	pll lock status	
20	cam1pll_lock	RO	pll lock status	
31:21	Reserved			

mipimpll_csr

Offset Address: 0x008

Bits	Name	Access	Description	Reset
6:0	mipimpll_pre_div_sel	R/W	pll pre_div_sel setting	0x0
7	Reserved			
14:8	mipimpll_post_div_sel	R/W	pll post_div_sel setting	0x0
16:15	mipimpll_sel_mode	R/W	pll mode setting	0x0
23:17	mipimpll_div_sel	R/W	pll div_sel setting	0x0
26:24	mipimpll_ictrl	R/W	pll ictrl setting	0x0
31:27	Reserved			

apll0_csr

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
6:0	apll0_pre_div_sel	R/W	pll pre_div_sel setting	0x0
7	Reserved			
14:8	apll0_post_div_sel	R/W	pll post_div_sel setting	0x0
16:15	apll0_sel_mode	R/W	pll mode setting	0x0
23:17	apll0_div_sel	R/W	pll div_sel setting	0x0
26:24	apll0_ictrl	R/W	pll ictrl setting	0x0
31:27	Reserved			

disppll_csr

Offset Address: 0x010

Bits	Name	Access	Description	Reset
6:0	disppll_pre_div_sel	R/W	pll pre_div_sel setting	0x0
7	Reserved			
14:8	disppll_post_div_sel	R/W	pll post_div_sel setting	0x0
16:15	disppll_sel_mode	R/W	pll mode setting	0x0
23:17	disppll_div_sel	R/W	pll div_sel setting	0x0
26:24	disppll_ictrl	R/W	pll ictrl setting	0x0
31:27	Reserved			

cam0pll_csr

Offset Address: 0x014

Bits	Name	Access	Description	Reset
6:0	cam0pll_pre_div_sel	R/W	pll pre_div_sel setting	0x0
7	Reserved			
14:8	cam0pll_post_div_sel	R/W	pll post_div_sel setting	0x0
16:15	cam0pll_sel_mode	R/W	pll mode setting	0x0
23:17	cam0pll_div_sel	R/W	pll div_sel setting	0x0

Bits	Name	Access	Description	Reset
26:24	cam0pll_ictrl	R/W	pll ictrl setting	0x0
31:27	Reserved			

cam1pll_csr

Offset Address: 0x018

Bits	Name	Access	Description	Reset
6:0	cam1pll_pre_div_sel	R/W	pll pre_div_sel setting	0x0
7	Reserved			
14:8	cam1pll_post_div_sel	R/W	pll post_div_sel setting	0x0
16:15	cam1pll_sel_mode	R/W	pll mode setting	0x0
23:17	cam1pll_div_sel	R/W	pll div_sel setting	0x0
26:24	cam1pll_ictrl	R/W	pll ictrl setting	0x0
31:27	Reserved			

pll_g2_ssc_syn_ctrl

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	reg_mipimpll_sel_syn_clk	R/W	mipimpll gen synthesizer clock source 0:450M 1:900M	0x1
1	reg_dsi_ssc_syn_src_en	R/W	pll synthesizer clock enable	0x0
2	reg_apll_ssc_syn_src_en	R/W	pll synthesizer clock enable	0x0
3	reg_disppll_ssc_syn_src_en	R/W	pll synthesizer clock enable	0x0
4	reg_cam0pll_ssc_syn_src_en	R/W	pll synthesizer clock enable	0x0
5	reg_cam1pll_ssc_syn_src_en	R/W	pll synthesizer clock enable	0x0
31:6	Reserved			

apll_ssc_syn_ctrl

Offset Address: 0x050

Bits	Name	Access	Description	Reset
0	reg_apll_ssc_syn_sw_up	W1T	pll synthesizer software update	
5:1	Reserved			
6	reg_apll_ssc_syn_fix_div	R/W		0x0
31:7	Reserved			

apll_ssc_syn_set

Offset Address: 0x054

Bits	Name	Access	Description	Reset
31:0	reg_apll_ssc_syn_set	R/W	pll synthesizer fraction setting: [31:26] integer 6 bits [26:0] decimal 26 bits	0x0

disppll_ssc_syn_ctrl

Offset Address: 0x060

Bits	Name	Access	Description	Reset
0	reg_disppll_ssc_syn_sw_up	W1T	pll synthesizer software update	
5:1	Reserved			
6	reg_disppll_ssc_syn_fix_div	R/W		0x0
31:7	Reserved			

disppll_ssc_syn_set

algorithmic technology

Offset Address: 0x064

Bits	Name	Access	Description	Reset
31:0	reg_disppll_ssc_syn_set	R/W	pll synthesizer fraction setting: [31:26] integer 6 bits [26:0] decimal 26 bits	0x0

cam0pll_ssc_syn_ctrl

Offset Address: 0x070

Bits	Name	Access	Description	Reset
0	reg_cam0pll_ssc_syn_sw_up	W1T	pll synthesizer software update	
5:1	Reserved			
6	reg_cam0pll_ssc_syn_fix_div	R/W		0x0
31:7	Reserved			

cam0pll_ssc_syn_set

Offset Address: 0x074

Bits	Name	Access	Description	Reset
31:0	reg_cam0pll_ssc_syn_set	R/W	pll synthesizer fraction setting: [31:26] integer 6 bits [26:0] decimal 26 bits	0x0

cam1pll_ssc_syn_ctrl

Offset Address: 0x080

Bits	Name	Access	Description	Reset
0	reg_cam1pll_ssc_syn_sw_up	W1T	pll synthesizer software update	
5:1	Reserved			
6	reg_cam1pll_ssc_syn_fix_div	R/W		0x0
31:7	Reserved			

cam1pll_ssc_syn_set

Offset Address: 0x084

Bits	Name	Access	Description	Reset
31:0	reg_cam1pll_ssc_syn_set	R/W	pll synthesizer fraction setting: [31:26] integer 6 bits [26:0] decimal 26 bits	0x0

apll_frac_div_ctrl

Offset Address: 0x090

Bits	Name	Access	Description	Reset
0	reg_apll_frac_div_clk_en	R/W	a24m clock src enable	0x0
1	reg_apll_frac_div_en	R/W	a24m clock div enable	0x0
2	reg_apll_frac_div_up	W1T		
3	reg_apll_frac_reg_out_en	R/W	a24m clock output enable	0x0
31:4	Reserved			

apll_frac_div_m

Offset Address: 0x094

Bits	Name	Access	Description	Reset
21:0	reg_apll_frac_div_m	R/W	a24m clock freq is 900*N/M/2 (MHz)	0x0
31:22	Reserved			

apll_frac_div_n

Offset Address: 0x098

Bits	Name	Access	Description	Reset
21:0	reg_apll_frac_div_n	R/W	a24m clock freq is 900*N/M/2 (MHz)	0x0
31:22	Reserved			

mipimpll_clk_csr

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
0	reg_mipimpll_pdiv_pd	R/W	pd post div	0x0
1	reg_mipimpll_d2_pd	R/W	pd div2 div	0x1
2	reg_mipimpll_d3_pd	R/W	pd div3 div	0x1
3	reg_mipimpll_d5_pd	R/W	pd div5 div	0x1
4	reg_mipimpll_d7_pd	R/W	pd div7 div	0x1
7:5	Reserved			
8	reg_mipimpll_pdiv_auto_pd	R/W	auto pd pdiv clk	0x0
9	reg_mipimpll_d2_auto_pd	R/W	auto pd div2 clk	0x1
10	reg_mipimpll_d3_auto_pd	R/W	auto pd div3 clk	0x1
11	reg_mipimpll_d5_auto_pd	R/W	auto pd div5 clk	0x1
12	reg_mipimpll_d7_auto_pd	R/W	auto pd div7 clk	0x1
31:13	Reserved			

a0pll_clk_csr

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
0	reg_a0pll_pdiv_pd	R/W	pd post div	0x0
1	reg_a0pll_d2_pd	R/W	pd div2 div	0x1
2	reg_a0pll_d3_pd	R/W	pd div3 div	0x1
3	reg_a0pll_d5_pd	R/W	pd div5 div	0x1
4	reg_a0pll_d7_pd	R/W	pd div7 div	0x1
7:5	Reserved			
8	reg_a0pll_pdiv_auto_pd	R/W	auto pd pdiv clk	0x0
9	reg_a0pll_d2_auto_pd	R/W	auto pd div2 clk	0x1
10	reg_a0pll_d3_auto_pd	R/W	auto pd div3 clk	0x1
11	reg_a0pll_d5_auto_pd	R/W	auto pd div5 clk	0x1
12	reg_a0pll_d7_auto_pd	R/W	auto pd div7 clk	0x1
31:13	Reserved			

disppll_clk_csr

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
0	reg_disppll_pdiv_pd	R/W	pd post div	0x0
1	reg_disppll_d2_pd	R/W	pd div2 div	0x1
2	reg_disppll_d3_pd	R/W	pd div3 div	0x1
3	reg_disppll_d5_pd	R/W	pd div5 div	0x1
4	reg_disppll_d7_pd	R/W	pd div7 div	0x1
7:5	Reserved			
8	reg_disppll_pdiv_auto_pd	R/W	auto pd pdiv clk	0x0
9	reg_disppll_d2_auto_pd	R/W	auto pd div2 clk	0x1
10	reg_disppll_d3_auto_pd	R/W	auto pd div3 clk	0x1
11	reg_disppll_d5_auto_pd	R/W	auto pd div5 clk	0x1
12	reg_disppll_d7_auto_pd	R/W	auto pd div7 clk	0x1
31:13	Reserved			

cam0pll_clk_csr

Offset Address: 0x0ac

Bits	Name	Access	Description	Reset
0	reg_cam0pll_pdiv_pd	R/W	pd post div	0x0
1	reg_cam0pll_d2_pd	R/W	pd div2 div	0x1
2	reg_cam0pll_d3_pd	R/W	pd div3 div	0x1
3	reg_cam0pll_d5_pd	R/W	pd div5 div	0x1
4	reg_cam0pll_d7_pd	R/W	pd div7 div	0x1
7:5	Reserved			
8	reg_cam0pll_pdiv_auto_pd	R/W	auto pd pdiv clk	0x0
9	reg_cam0pll_d2_auto_pd	R/W	auto pd div2 clk	0x1
10	reg_cam0pll_d3_auto_pd	R/W	auto pd div3 clk	0x1
11	reg_cam0pll_d5_auto_pd	R/W	auto pd div5 clk	0x1
12	reg_cam0pll_d7_auto_pd	R/W	auto pd div7 clk	0x1
31:13	Reserved			

cam1pll_clk_csr

Offset Address: 0x0b0

Bits	Name	Access	Description	Reset
0	reg_cam1pll_pdiv_pd	R/W	pd post div	0x0
1	reg_cam1pll_d2_pd	R/W	pd div2 div	0x1
2	reg_cam1pll_d3_pd	R/W	pd div3 div	0x1
3	reg_cam1pll_d5_pd	R/W	pd div5 div	0x1
4	reg_cam1pll_d7_pd	R/W	pd div7 div	0x1
7:5	Reserved			
8	reg_cam1pll_pdiv_auto_pd	R/W	auto pd pdiv clk	0x0
9	reg_cam1pll_d2_auto_pd	R/W	auto pd div2 clk	0x1
10	reg_cam1pll_d3_auto_pd	R/W	auto pd div3 clk	0x1
11	reg_cam1pll_d5_auto_pd	R/W	auto pd div5 clk	0x1
12	reg_cam1pll_d7_auto_pd	R/W	auto pd div7 clk	0x1
31:13	Reserved			

clk_cam0_src_div

Offset Address: 0x0c0

Bits	Name	Access	Description	Reset
0	reg_cam0_div_rstn	R/W	[0] Divider Reset Control 0: Assert Reset	0x1
3:1	Reserved			
4	reg_cam0_div_dis	R/W	[4] Divider Reset Control 0: Assert Reset	0x0
7:5	Reserved			
9:8	reg_cam0_src	R/W	[9:8] Clock source 0: cam0pll 1: cam0pll_d2 2: cam0pll_d3 3: mipimpll_d3	0x0
15:10	Reserved			
21:16	reg_cam0_div	R/W	[21:16] Clock Divider Factor	0x20
31:22	Reserved			

clk_cam1_src_div

Offset Address: 0x0c4

Bits	Name	Access	Description	Reset
0	reg_cam1_div_rstn	R/W	[0] Divider Reset Control 0: Assert Reset	0x1
3:1	Reserved			
4	reg_cam1_div_dis	R/W	[4] divider disable	0x0

Bits	Name	Access	Description	Reset
7:5	Reserved			
9:8	reg_cam1_src	R/W	[9:8] Clock source 0: cam0pll 1: cam0pll_d2 2: cam0pll_d3 3: mipimpll_d3	0x0
15:10	Reserved			
21:16	reg_cam1_div	R/W	[21:16] Clock Divider Factor	0x20
31:22	Reserved			

6.8.2 PLL_G6 Register Description

pll_g6_ctrl

Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	mpll_pwd	R/W	pll power down	0x0
3:1	Reserved			
4	tpll_pwd	R/W	pll power down	0x0
7:5	Reserved			
8	fpll_pwd	R/W	pll power down	0x0
31:9	Reserved			

pll_g6_status

Offset Address: 0x004

Bits	Name	Access	Description	Reset
0	updating_mpll_val	RO	pll setting update status	
1	updating_tpll_val	RO	pll setting update status	
2	updating_fpll_val	RO	pll setting update status	
15:3	Reserved			
16	mpll_lock	RO	pll lock status	
17	tpll_lock	RO	pll lock status	
18	fpll_lock	RO	pll lock status	
31:19	Reserved			

mpll_csr

Offset Address: 0x008

Bits	Name	Access	Description	Reset
6:0	mpll_pre_div_sel	R/W	pll pre_div_sel setting	0x0
7	Reserved			
14:8	mpll_post_div_sel	R/W	pll post_div_sel setting	0x0
16:15	mpll_sel_mode	R/W	pll mode setting	0x0
23:17	mpll_div_sel	R/W	pll div_sel setting	0x0
26:24	mpll_ictrl	R/W	pll ictrl setting	0x0
31:27	Reserved			

tpll_csr

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
6:0	tpll_pre_div_sel	R/W	pll pre_div_sel setting	0x0
7	Reserved			
14:8	tpll_post_div_sel	R/W	pll post_div_sel setting	0x0
16:15	tpll_sel_mode	R/W	pll mode setting	0x0
23:17	tpll_div_sel	R/W	pll div_sel setting	0x0
26:24	tpll_ictrl	R/W	pll ictrl setting	0x0
31:27	Reserved			

fpll_csr

Offset Address: 0x010

Bits	Name	Access	Description	Reset
6:0	fpll_pre_div_sel	R/W	pll pre_div_sel setting	0x0
7	Reserved			
14:8	fpll_post_div_sel	R/W	pll post_div_sel setting	0x0
16:15	fpll_sel_mode	R/W	pll mode setting	0x0
23:17	fpll_div_sel	R/W	pll div_sel setting	0x0
26:24	fpll_ictrl	R/W	pll ictrl setting	0x0
31:27	Reserved			

pll_g6_ssc_syn_ctrl

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	reg_fpll_sel_syn_clk	R/W	fpll gen synthesizer clock source 0:750M 1:1.5G	0x1
1	reg_ddr_ssc_syn_src_en	R/W	ddr pll synthesizer clock enable	0x1
2	reg_mpll_ssc_syn_src_en	R/W	mpll synthesizer clock enable	0x0
3	reg_tpll_ssc_syn_src_en	R/W	tpll synthesizer clock enable	0x0
31:4	Reserved			

dpll_ssc_syn_ctrl

Offset Address: 0x050

Bits	Name	Access	Description	Reset
0	reg_dpll_ssc_syn_sw_up	W1T	pll synthesizer software update	
1	reg_dpll_ssc_syn_en_ssc	R/W	pll synthesizer ssc enable	0x0
3:2	reg_dpll_ssc_syn_ssc_mode	R/W		0x0
4	reg_dpll_ssc_syn_bypass	R/W		0x0
5	reg_dpll_ssc_syn_extpulse	R/W		0x0
6	reg_dpll_ssc_syn_fix_div	R/W		0x0
31:7	Reserved			

dpll_ssc_syn_set

Offset Address: 0x054

Bits	Name	Access	Description	Reset
31:0	reg_dpll_ssc_syn_set	R/W	pll synthesizer fraction setting: [31:26] integer 6 bits [26:0] decimal 26 bits	0x0

dpll_ssc_syn_span

Offset Address: 0x058

Bits	Name	Access	Description	Reset
15:0	reg_dpll_ssc_syn_span	R/W		0x0
31:16	Reserved			

dpll_ssc_syn_step

Offset Address: 0x05c

Bits	Name	Access	Description	Reset
23:0	reg_dpll_ssc_syn_step	R/W		0x0
31:24	Reserved			

mpll_ssc_syn_ctrl

Offset Address: 0x060

Bits	Name	Access	Description	Reset
0	reg_mpll_ssc_syn_sw_up	W1T	pll synthesizer software update	
1	reg_mpll_ssc_syn_en_ssc	R/W	pll synthesizer ssc enable	0x0
3:2	reg_mpll_ssc_syn_ssc_mode	R/W		0x0
4	reg_mpll_ssc_syn_bypass	R/W		0x1
5	reg_mpll_ssc_syn_extpulse	R/W		0x0
6	reg_mpll_ssc_syn_fix_div	R/W		0x0
31:7	Reserved			

mpll_ssc_syn_set

Offset Address: 0x064

Bits	Name	Access	Description	Reset
31:0	reg_mpll_ssc_syn_set	R/W	pll synthesizer fraction setting: [31:26] integer 6 bits [26:0] decimal 26 bits	0x0

mpll_ssc_syn_span

Offset Address: 0x068

Bits	Name	Access	Description	Reset
15:0	reg_mpll_ssc_syn_span	R/W		0x0
31:16	Reserved			

mpll_ssc_syn_step

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
23:0	reg_mpll_ssc_syn_step	R/W		0x0
31:24	Reserved			

tpll_ssc_syn_ctrl

Offset Address: 0x070

Bits	Name	Access	Description	Reset
0	reg_tpll_ssc_syn_sw_up	W1T	pll synthesizer software update	
1	reg_tpll_ssc_syn_en_ssc	R/W	pll synthesizer ssc enable	0x0
3:2	reg_tpll_ssc_syn_ssc_mode	R/W		0x0
4	reg_tpll_ssc_syn_bypass	R/W		0x1
5	reg_tpll_ssc_syn_extpulse	R/W		0x0
6	reg_tpll_ssc_syn_fix_div	R/W		0x0

Bits	Name	Access	Description	Reset
31:7	Reserved			

tpll_ssc_syn_set

Offset Address: 0x074

Bits	Name	Access	Description	Reset
31:0	reg_tpll_ssc_syn_set	R/W	pll synthesizer fraction setting: [31:26] integer 6 bits [26:0] decimal 26 bits	0x0

tpll_ssc_syn_span

Offset Address: 0x078

Bits	Name	Access	Description	Reset
15:0	reg_tpll_ssc_syn_span	R/W		0x0
31:16	Reserved			

tpll_ssc_syn_step

Offset Address: 0x07c

Bits	Name	Access	Description	Reset
23:0	reg_tpll_ssc_syn_step	R/W		0x0
31:24	Reserved			

6.9 CLK_DIV CRG Register Overview

Clock Gen base address : 0x03002000

Name	Address Offset	Description
clk_en_0	0x000	clock enable register 0
clk_en_1	0x004	clock enable register 1
clk_en_2	0x008	clock enable register 2
clk_en_3	0x00c	clock enable register 3
clk_en_4	0x010	clock enable register 4
clk_sel_0	0x020	Clock source selection register 0
clk_byp_0	0x030	clock bypass to xtal register 0
clk_byp_1	0x034	clock bypass to xtal register 1
div_clk_a53_0	0x040	divider register of clk_a53_0
div_clk_a53_1	0x044	divider register of clk_a53_1
div_clk_cpu_axi0	0x048	divider register of clk_cpu_axi0
div_clk_cpu_gic	0x050	divider register of clk_cpu_gic
div_clk_tpu	0x054	divider register of clk_tpu
div_clk_emmc	0x064	divider register of clk_emmc
div_clk_100k_emmc	0x06c	divider register of clk_100k_emmc
div_clk_sd0	0x070	divider register of clk_sd0
div_clk_100k_sd0	0x078	divider register of clk_100k_sd0
div_clk_sd1	0x07c	divider register of clk_sd1
div_clk_100k_sd1	0x084	divider register of clk_100k_sd1
div_clk_spi_nand	0x088	divider register of clk_spi_nand
div_clk_500m_eth0	0x08c	divider register of clk_500m_eth0
div_clk_gpio_db	0x094	divider register of clk_gpio_db

Name	Address Offset	Description
div_clk_sdma_aud0	0x098	divider register of clk_sdma_aud0
div_clk_sdma_aud1	0x09c	divider register of clk_sdma_aud1
div_clk_sdma_aud2	0x0a0	divider register of clk_sdma_aud2
div_clk_sdma_aud3	0x0a4	divider register of clk_sdma_aud3
div_clk_cam0_200	0x0a8	divider register of clk_cam0_200
div_clk_axi4	0x0b8	divider register of clk_axi4
div_clk_axi6	0x0bc	divider register of clk_axi6
div_clk_dsi_esc	0x0c4	divider register of clk_dsi_esc
div_clk_axi_vip	0x0c8	divider register of clk_axi_vip
div_clk_src_vip_sys_0	0x0d0	divider register of clk_src_vip_sys_0
div_clk_src_vip_sys_1	0x0d8	divider register of clk_src_vip_sys_1
div_clk_disp_src_vip	0x0e0	divider register of clk_disp_src_vip
div_clk_axi_video_codec	0x0e4	divider register of clk_axi_video_codec
div_clk_vc_src0	0x0ec	divider register of clk_vc_src0
div_clk_1m	0x0fc	divider register of clk_1m
div_clk_spi	0x100	divider register of clk_spi
div_clk_i2c	0x104	divider register of clk_i2c
div_clk_src_vip_sys_2	0x110	divider register of clk_src_vip_sys_2
div_clk_audsrc	0x118	divider register of clk_audsrc
div_clk_pwm_src_0	0x120	divider register of clk_pwm_src_0
div_clk_ap_debug	0x128	divider register of clk_ap_debug
div_clk_rtcsys_src_0	0x12c	divider register of clk_rtcsys_src_0
div_clk_c906_0_0	0x130	divider register of clk_c906_0_0
div_clk_c906_0_1	0x134	divider register of clk_c906_0_1
div_clk_c906_1_0	0x138	divider register of clk_c906_1_0
div_clk_c906_1_1	0x13c	divider register of clk_c906_1_1
div_clk_src_vip_sys_3	0x140	divider register of clk_src_vip_sys_3
div_clk_src_vip_sys_4	0x144	divider register of clk_src_vip_sys_4

6.10 CLK_DIV CRG Register Description

clk_en_0

Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	clk_en_0_0	R/W	Clock Enable for clk_a53 (1: Enable; 0. Gate)	0x1
1	clk_en_0_1	R/W	Clock Enable for clk_cpu_axi0 (1. Enable; 0: Gate)	0x1
2	clk_en_0_2	R/W	Clock Enable for clk_cpu_gic (1: Enable. 0: Gate)	0x1
3	clk_en_0_3	R/W	Clock Enable for clk_xtal_a53 (1: Enable. 0: Gate)	0x1
4	clk_en_0_4	R/W	Clock Enable for clk_tpu (1: Enable; 0: Gate)	0x1
5	Reserved			
6	clk_en_0_6	R/W	Clock Enable for clk_ahb_rom (1. Enable; 0: Gate)	0x1
7	clk_en_0_7	R/W	Clock Enable for clk_ddr_axi_reg (1. Enable; 0: Gate)	0x1
8	clk_en_0_8	R/W	Clock Enable for clk_rtc_25m (1: Enable; 0: Gate)	0x1

Bits	Name	Access	Description	Reset
9	clk_en_0_9	R/W	Clock Enable for clk_tempsen (1. Enable; 0: Gate)	0x1
10	clk_en_0_10	R/W	Clock Enable for clk_saradc (1: Enable. 0: Gate)	0x1
11	clk_en_0_11	R/W	Clock Enable for clk_efuse (1: Enable; 0: Gate)	0x1
12	clk_en_0_12	R/W	Clock Enable for clk_apb_efuse (1. Enable; 0: Gate)	0x1
13	Reserved			
14	clk_en_0_14	R/W	Clock Enable for clk_xtal_misc (1. Enable; 0: Gate)	0x1
15	clk_en_0_15	R/W	Clock Enable for clk_axi4_emmc (1. Enable; 0: Gate)	0x1
16	clk_en_0_16	R/W	Clock Enable for clk_emmc (1: Enable. 0: Gate)	0x1
17	clk_en_0_17	R/W	Clock Enable for clk_100k_emmc (1. Enable; 0: Gate)	0x1
18	clk_en_0_18	R/W	Clock Enable for clk_axi4_sd0 (1. Enable; 0: Gate)	0x1
19	clk_en_0_19	R/W	Clock Enable for clk_sd0 (1: Enable; 0: Gate)	0x1
20	clk_en_0_20	R/W	Clock Enable for clk_100k_sd0 (1. Enable; 0: Gate)	0x1
21	clk_en_0_21	R/W	Clock Enable for clk_axi4_sd1 (1. Enable; 0: Gate)	0x1
22	clk_en_0_22	R/W	Clock Enable for clk_sd1 (1: Enable; 0: Gate)	0x1
23	clk_en_0_23	R/W	Clock Enable for clk_100k_sd1 (1. Enable; 0: Gate)	0x1
24	clk_en_0_24	R/W	Clock Enable for clk_spi_nand (1. Enable; 0: Gate)	0x1
25	clk_en_0_25	R/W	Clock Enable for clk_500m_eth0 (1. Enable; 0: Gate)	0x1
26	clk_en_0_26	R/W	Clock Enable for clk_axi4_eth0 (1: Enable; 0: Gate)	0x1
28:27	Reserved			
29	clk_en_0_29	R/W	Clock Enable for clk_apb_gpio (1. Enable; 0: Gate)	0x1
30	clk_en_0_30	R/W	Clock Enable for clk_apb_gpio_intr (1. Enable; 0: Gate)	0x1
31	clk_en_0_31	R/W	Clock Enable for clk_gpio_db (1: Enable; 0: Gate)	0x1

clk_en_1

Offset Address: 0x004

Bits	Name	Access	Description	Reset
0	clk_en_1_0	R/W	Clock Enable for clk_ahb_sf (1: Enable. 0: Gate)	0x1
1	clk_en_1_1	R/W	Clock Enable for clk_sdma_axi (1: Enable; 0: Gate)	0x1
2	clk_en_1_2	R/W	Clock Enable for clk_sdma_aud0 (1. Enable; 0: Gate)	0x1
3	clk_en_1_3	R/W	Clock Enable for clk_sdma_aud1 (1. Enable; 0: Gate)	0x1
4	clk_en_1_4	R/W	Clock Enable for clk_sdma_aud2 (1. Enable; 0: Gate)	0x1

Bits	Name	Access	Description	Reset
5	clk_en_1_5	R/W	Clock Enable for clk_sdma_aud3 (1. Enable; 0: Gate)	0x1
6	clk_en_1_6	R/W	Clock Enable for clk_apb_i2c (1: Enable. 0: Gate)	0x1
7	clk_en_1_7	R/W	Clock Enable for clk_apb_wdt (1: Enable; 0: Gate)	0x1
8	clk_en_1_8	R/W	Clock Enable for clk_apb_pwm (1. Enable; 0: Gate)	0x1
9	clk_en_1_9	R/W	Clock Enable for clk_apb_spi0 (1. Enable; 0: Gate)	0x1
10	clk_en_1_10	R/W	Clock Enable for clk_apb_spi1 (1. Enable; 0: Gate)	0x1
11	clk_en_1_11	R/W	Clock Enable for clk_apb_spi2 (1: Enable; 0: Gate)	0x1
12	clk_en_1_12	R/W	Clock Enable for clk_apb_spi3 (1. Enable; 0: Gate)	0x1
13	clk_en_1_13	R/W	Clock Enable for clk_187p5m (1: Enable. 0: Gate)	0x1
14	clk_en_1_14	R/W	Clock Enable for clk_uart0 (1: Enable; 0: Gate)	0x1
15	clk_en_1_15	R/W	Clock Enable for clk_apb_uart0 (1. Enable; 0: Gate)	0x1
16	clk_en_1_16	R/W	Clock Enable for clk_uart1 (1: Enable; 0. Gate)	0x1
17	clk_en_1_17	R/W	Clock Enable for clk_apb_uart1 (1. Enable; 0: Gate)	0x1
18	clk_en_1_18	R/W	Clock Enable for clk_uart2 (1: Enable; 0. Gate)	0x1
19	clk_en_1_19	R/W	Clock Enable for clk_apb_uart2 (1. Enable; 0: Gate)	0x1
20	clk_en_1_20	R/W	Clock Enable for clk_uart3 (1: Enable; 0. Gate)	0x1
21	clk_en_1_21	R/W	Clock Enable for clk_apb_uart3 (1. Enable; 0: Gate)	0x1
22	clk_en_1_22	R/W	Clock Enable for clk_uart4 (1: Enable; 0: Gate)	0x1
23	clk_en_1_23	R/W	Clock Enable for clk_apb_uart4 (1. Enable; 0: Gate)	0x1
24	clk_en_1_24	R/W	Clock Enable for clk_apb_i2s0 (1. Enable; 0: Gate)	0x1
25	clk_en_1_25	R/W	Clock Enable for clk_apb_i2s1 (1: Enable; 0: Gate)	0x1
26	clk_en_1_26	R/W	Clock Enable for clk_apb_i2s2 (1. Enable; 0: Gate)	0x1
27	clk_en_1_27	R/W	Clock Enable for clk_apb_i2s3 (1. Enable; 0: Gate)	0x1
28	clk_en_1_28	R/W	Clock Enable for clk_axi4_usb (1. Enable; 0: Gate)	0x1
29	clk_en_1_29	R/W	Clock Enable for clk_apb_usb (1: Enable; 0: Gate)	0x1
31:30	Reserved			

clk_en_2

Offset Address: 0x008

Bits	Name	Access	Description	Reset
0	Reserved			

Bits	Name	Access	Description	Reset
1	clk_en_2_1	R/W	Clock Enable for clk_axi4 (1: Enable; 0. Gate)	0x1
2	clk_en_2_2	R/W	Clock Enable for clk_axi6 (1: Enable; 0. Gate)	0x1
3	clk_en_2_3	R/W	Clock Enable for clk_dsi_esc (1: Enable; 0: Gate)	0x1
4	clk_en_2_4	R/W	Clock Enable for clk_axi_vip (1: Enable. 0: Gate)	0x1
5	clk_en_2_5	R/W	Clock Enable for clk_src_vip_sys_0 (1. Enable; 0: Gate)	0x1
6	clk_en_2_6	R/W	Clock Enable for clk_src_vip_sys_1 (1. Enable; 0: Gate)	0x1
7	clk_en_2_7	R/W	Clock Enable for clk_disp_src_vip (1: Enable; 0: Gate)	0x1
8	clk_en_2_8	R/W	Clock Enable for clk_axi_video_codec (1: Enable; 0: Gate)	0x1
9	clk_en_2_9	R/W	Clock Enable for clk_vc_src0 (1: Enable. 0: Gate)	0x1
10	clk_en_2_10	R/W	Clock Enable for clk_h264c (1: Enable; 0: Gate)	0x1
11	clk_en_2_11	R/W	Clock Enable for clk_h265c (1: Enable; 0. Gate)	0x1
12	clk_en_2_12	R/W	Clock Enable for clk_jpeg (1: Enable; 0. Gate)	0x1
13	clk_en_2_13	R/W	Clock Enable for clk_apb_jpeg (1. Enable; 0: Gate)	0x1
14	clk_en_2_14	R/W	Clock Enable for clk_apb_h264c (1. Enable; 0: Gate)	0x1
15	clk_en_2_15	R/W	Clock Enable for clk_apb_h265c (1. Enable; 0: Gate)	0x1
16	clk_en_2_16	R/W	Clock Enable for clk_cam0 (1: Enable; 0. Gate)	0x1
17	clk_en_2_17	R/W	Clock Enable for clk_cam1 (1: Enable; 0. Gate)	0x1
18	clk_en_2_18	R/W	Clock Enable for clk_csi_mac0_vip (1: Enable; 0: Gate)	0x1
19	clk_en_2_19	R/W	Clock Enable for clk_csi_mac1_vip (1. Enable; 0: Gate)	0x1
20	clk_en_2_20	R/W	Clock Enable for clk_isp_top_vip (1. Enable; 0: Gate)	0x1
21	clk_en_2_21	R/W	Clock Enable for clk_img_d_vip (1: Enable; 0: Gate)	0x1
22	clk_en_2_22	R/W	Clock Enable for clk_img_v_vip (1. Enable; 0: Gate)	0x1
23	clk_en_2_23	R/W	Clock Enable for clk_sc_top_vip (1. Enable; 0: Gate)	0x1
24	clk_en_2_24	R/W	Clock Enable for clk_sc_d_vip (1. Enable; 0: Gate)	0x1
25	clk_en_2_25	R/W	Clock Enable for clk_sc_v1_vip (1: Enable; 0: Gate)	0x1
26	clk_en_2_26	R/W	Clock Enable for clk_sc_v2_vip (1. Enable; 0: Gate)	0x1
27	clk_en_2_27	R/W	Clock Enable for clk_sc_v3_vip (1. Enable; 0: Gate)	0x1
28	clk_en_2_28	R/W	Clock Enable for clk_ldc_vip (1: Enable. 0: Gate)	0x1
29	clk_en_2_29	R/W	Clock Enable for clk_bt_vip (1: Enable.	0x1

Bits	Name	Access	Description	Reset
			0: Gate)	
30	clk_en_2_30	R/W	Clock Enable for clk_disp_vip (1: Enable. 0: Gate)	0x1
31	clk_en_2_31	R/W	Clock Enable for clk_dsi_mac_vip (1. Enable; 0: Gate)	0x1

clk_en_3

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
0	clk_en_3_0	R/W	Clock Enable for clk_lvds0_vip (1. Enable; 0: Gate)	0x1
1	clk_en_3_1	R/W	Clock Enable for clk_lvds1_vip (1. Enable; 0: Gate)	0x1
2	clk_en_3_2	R/W	Clock Enable for clk_csi0_rx_vip (1. Enable; 0: Gate)	0x1
3	clk_en_3_3	R/W	Clock Enable for clk_csi1_rx_vip (1: Enable; 0: Gate)	0x1
4	clk_en_3_4	R/W	Clock Enable for clk_pad_vi_vip (1. Enable; 0: Gate)	0x1
5	clk_en_3_5	R/W	Clock Enable for clk_1m (1: Enable; 0. Gate)	0x1
6	clk_en_3_6	R/W	Clock Enable for clk_spi (1: Enable; 0. Gate)	0x1
7	clk_en_3_7	R/W	Clock Enable for clk_i2c (1: Enable; 0: Gate)	0x1
8	clk_en_3_8	R/W	Clock Enable for clk_pm (1: Enable; 0. Gate)	0x1
9	clk_en_3_9	R/W	Clock Enable for clk_timer0 (1: Enable. 0: Gate)	0x1
10	clk_en_3_10	R/W	Clock Enable for clk_timer1 (1: Enable. 0: Gate)	0x1
11	clk_en_3_11	R/W	Clock Enable for clk_timer2 (1: Enable; 0: Gate)	0x1
12	clk_en_3_12	R/W	Clock Enable for clk_timer3 (1: Enable. 0: Gate)	0x1
13	clk_en_3_13	R/W	Clock Enable for clk_timer4 (1: Enable. 0: Gate)	0x1
14	clk_en_3_14	R/W	Clock Enable for clk_timer5 (1: Enable. 0: Gate)	0x1
15	clk_en_3_15	R/W	Clock Enable for clk_timer6 (1: Enable; 0: Gate)	0x1
16	clk_en_3_16	R/W	Clock Enable for clk_timer7 (1: Enable. 0: Gate)	0x1
17	clk_en_3_17	R/W	Clock Enable for clk_apb_i2c0 (1. Enable; 0: Gate)	0x1
18	clk_en_3_18	R/W	Clock Enable for clk_apb_i2c1 (1: Enable; 0: Gate)	0x1
19	clk_en_3_19	R/W	Clock Enable for clk_apb_i2c2 (1. Enable; 0: Gate)	0x1
20	clk_en_3_20	R/W	Clock Enable for clk_apb_i2c3 (1. Enable; 0: Gate)	0x1
21	clk_en_3_21	R/W	Clock Enable for clk_apb_i2c4 (1. Enable; 0: Gate)	0x1
22	clk_en_3_22	R/W	Clock Enable for clk_wgn (1: Enable; 0: Gate)	0x1
23	clk_en_3_23	R/W	Clock Enable for clk_wgn0 (1: Enable; 0. Gate)	0x1

Bits	Name	Access	Description	Reset
24	clk_en_3_24	R/W	Clock Enable for clk_wgn1 (1: Enable; 0. Gate)	0x1
25	clk_en_3_25	R/W	Clock Enable for clk_wgn2 (1: Enable; 0. Gate)	0x1
26	clk_en_3_26	R/W	Clock Enable for clk_keyscan (1: Enable; 0: Gate)	0x1
27	clk_en_3_27	R/W	Clock Enable for clk_ahb_sf1 (1: Enable. 0: Gate)	0x1
28	Reserved			
29	clk_en_3_29	R/W	Clock Enable for clk_src_vip_sys_2 (1. Enable; 0: Gate	0x1
30	clk_en_3_30	R/W	Clock Enable for clk_pad_vi1_vip (1. Enable; 0: Gate)	0x1
31	clk_en_3_31	R/W	Clock Enable for clk_cfg_reg_vip (1. Enable; 0: Gate)	0x1

clk_en_4

Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	clk_en_4_0	R/W	Clock Enable for clk_cfg_reg_vc (1. Enable; 0: Gate)	0x1
1	clk_en_4_1	R/W	Clock Enable for clk_audsrc (1: Enable. 0: Gate)	0x1
2	clk_en_4_2	R/W	Clock Enable for clk_apb_audsrc (1. Enable; 0: Gate)	0x1
3	Reserved			
4	clk_en_4_4	R/W	Clock Enable for clk_pwm_src (1: Enable; 0: Gate)	0x1
5	clk_en_4_5	R/W	Clock Enable for clk_ap_debug(1. Enable; 0: Gate)	0x1
6	clk_en_4_6	R/W	Clock Enable for clk_rtcsys_src_0 (1. Enable; 0: Gate)	0x1
7	clk_en_4_7	R/W	Clock Enable for clk_pad_vi2_vip (1. Enable; 0: Gate)	0x1
8	clk_en_4_8	R/W	Clock Enable for clk_csi_be_vip (1: Enable; 0: Gate)	0x1
9	clk_en_4_9	R/W	Clock Enable for clk_vip_ip0_en	0x1
10	clk_en_4_10	R/W	Clock Enable for clk_vip_ip1_en	0x1
11	clk_en_4_11	R/W	Clock Enable for clk_vip_ip2_en	0x1
12	clk_en_4_12	R/W	Clock Enable for clk_vip_ip3_en	0x1
13	clk_en_4_13	R/W	Clock Enable for clk_c906_0_en	0x1
14	clk_en_4_14	R/W	Clock Enable for clk_c906_1_en	0x1
15	clk_en_4_15	R/W	Clock Enable for clk_src_vip_sys_3_en	0x1
16	clk_en_4_16	R/W	Clock Enable for clk_src_vip_sys_4_en	0x1
17	clk_en_4_17	R/W	Clock Enable for clk_ive_vip_en	0x1
18	clk_en_4_18	R/W	Clock Enable for clk_raw_vip_en	0x1
19	clk_en_4_19	R/W	Clock Enable for clk_osdc_vip_en	0x1
20	clk_en_4_20	R/W	Clock Enable for clk_fbc_vip_en	0x1
21	clk_en_4_21	R/W	Clock Enable for clk_cam0_vip_en	0x1
31:22	Reserved			

clk_sel_0

Offset Address: 0x020

Bits	Name	Access	Description	Reset
0	clk_sel_0_0	R/W	Clock Select for A53's clock clk_a53	0x0

Bits	Name	Access	Description	Reset
			1: Select div_clk_a53_0 as clock source 0: Select div_clk_a53_1 as clock source	
22:1	Reserved			
23	clk_sel_0_23	R/W	Clock Select for C906's clock clk_c906_0 1: Select div_clk_c906_0_0 as clock source 0: Select div_clk_c906_0_1 as clock source	0x0
24	clk_sel_0_24	R/W	Clock Select for C906's clock clk_c906_1 1: Select div_clk_c906_1_0 as clock source 0: Select div_clk_c906_1_1 as clock source	0x0
31:25	Reserved			

clk_byp_0

Offset Address: 0x030

Bits	Name	Access	Description	Reset
0	clk_byp_0_0	R/W	Clock Bypass to xtal for A53's clock clk_a53	0x1
1	clk_byp_0_1	R/W	Clock Bypass to xtal for A53's clock clk_cpu_axi0	0x1
2	clk_byp_0_2	R/W	Clock Bypass to xtal for A53's clock clk_cpu_gic	0x1
3	clk_byp_0_3	R/W	Clock Bypass to xtal for TPU's clock clk_tpu	0x1
4	Reserved			
5	clk_byp_0_5	R/W	Clock Bypass to xtal for EMMC's clock clk_emmc	0x1
6	clk_byp_0_6	R/W	Clock Bypass to xtal for SD's clock clk_sd0	0x1
7	clk_byp_0_7	R/W	Clock Bypass to xtal for SD's clock clk_sd1	0x1
8	clk_byp_0_8	R/W	Clock Bypass to xtal for SPI_NAND's clock clk_spi_nand	0x1
9	clk_byp_0_9	R/W	Clock Bypass to xtal for ETH0's clock clk_500m_eth0	0x1
10	Reserved			
11	clk_byp_0_11	R/W	Clock Bypass to xtal for AUDIO's clock clk_aud0	0x1
12	clk_byp_0_12	R/W	Clock Bypass to xtal for AUDIO's clock clk_aud1	0x1
13	clk_byp_0_13	R/W	Clock Bypass to xtal for AUDIO's clock clk_aud2	0x1
14	clk_byp_0_14	R/W	Clock Bypass to xtal for AUDIO's clock clk_aud3	0x1
15	clk_byp_0_15	R/W	Clock Bypass to xtal for PWM's clock clk_pwm_src	0x1
16	clk_byp_0_16	R/W	Clock Bypass to xtal for TOP's clock clk_cam0_200	0x1
18:17	Reserved			
19	clk_byp_0_19	R/W	Clock Bypass to xtal for FABRIC_AXI4's clock clk_axi4	0x1
20	clk_byp_0_20	R/W	Clock Bypass to xtal for FABRIC_AXI6's clock clk_axi6	0x1

Bits	Name	Access	Description	Reset
21	clk_byp_0_21	R/W	Clock Bypass to xtal for VIP_SYS's clock clk_dsi_esc	0x1
22	clk_byp_0_22	R/W	Clock Bypass to xtal for VIP_SYS's clock clk_axi_vip	0x1
23	clk_byp_0_23	R/W	Clock Bypass to xtal for VIP_SYS's clock clk_src_vip_sys_0	0x1
24	clk_byp_0_24	R/W	Clock Bypass to xtal for VIP_SYS's clock clk_src_vip_sys_1	0x1
25	clk_byp_0_25	R/W	Clock Bypass to xtal for VIP_SYS's clock clk_disp_src_vip	0x1
26	clk_byp_0_26	R/W	Clock Bypass to xtal for Video_subsys's clock clk_axi_video_codec	0x1
27	clk_byp_0_27	R/W	Clock Bypass to xtal for Video_subsys's clock clk_vc_src0	0x1
29:28	Reserved			
30	clk_byp_0_30	R/W	Clock Bypass to xtal for SPI's clock clk_spi	0x1
31	clk_byp_0_31	R/W	Clock Bypass to xtal for IIC's clock clk_i2c	0x1

clk_byp_1

Offset Address: 0x034

Bits	Name	Access	Description	Reset
0	Reserved			
1	clk_byp_1_1	R/W	Clock Bypass to xtal for VIP_SYS's clock clk_src_vip_sys_2	0x1
2	clk_byp_1_2	R/W	Clock Bypass to xtal for AUDSRC's clock clk_audsrc	0x1
3	clk_byp_1_3	R/W	Clock Bypass to xtal for Video_subsys's clock clk_vc_src2	0x1
4	clk_byp_1_4	R/W	Clock Bypass to xtal for clk_ap_debug	0x1
5	clk_byp_1_5	R/W	Clock Bypass to xtal for clk_src_rtc_sys_0	0x1
6	clk_byp_1_6	R/W	Clock Bypass to xtal for c906_0	0x1
7	clk_byp_1_7	R/W	Clock Bypass to xtal for c906_1	0x1
8	clk_byp_1_8	R/W	Clock Bypass to xtal for VIP_SYS's clock clk_src_vip_sys_3	0x1
9	clk_byp_1_9	R/W	Clock Bypass to xtal for VIP_SYS's clock clk_src_vip_sys_4	0x1
31:10	Reserved			

div_clk_a53_0

Offset Address: 0x040

Bits	Name	Access	Description	Reset
31:0	div_clk_a53_0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : tpll 1 : apll 2 : mipimpll 3 : mpll	0x00000201

div_clk_a53_1

Offset Address: 0x044

Bits	Name	Access	Description	Reset
31:0	div_clk_a53_1	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : fpll	0x00000001

div_clk_cpu_axi0

Offset Address: 0x048

Bits	Name	Access	Description	Reset
31:0	div_clk_cpu_axi0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : fpll 1 : disppll	0x00000001

div_clk_cpu_gic

Offset Address: 0x050

Bits	Name	Access	Description	Reset
31:0	div_clk_cpu_gic	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_tpu

Offset Address: 0x054

Bits	Name	Access	Description	Reset
31:0	div_clk_tpu	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : tpll 1 : apll 2 : mipimpll 3 : fpll	0x00000301

div_clk_emmc

Offset Address: 0x064

Bits	Name	Access	Description	Reset
31:0	div_clk_emmc	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0. Select initial value 1: Select Divide	0x00000001

Bits	Name	Access	Description	Reset
			Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : fpll 1 : disppll	

div_clk_100k_emmc

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
31:0	div_clk_100k_emmc	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_sd0

Offset Address: 0x070

Bits	Name	Access	Description	Reset
31:0	div_clk_sd0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : fpll 1 : disppll	0x00000001

div_clk_100k_sd0

Offset Address: 0x078

Bits	Name	Access	Description	Reset
31:0	div_clk_100k_sd0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_sd1

Offset Address: 0x07c

Bits	Name	Access	Description	Reset
31:0	div_clk_sd1	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : fpll 1 : disppll	0x00000001

div_clk_100k_sd1

Offset Address: 0x084

Bits	Name	Access	Description	Reset
31:0	div_clk_100k_sd1	R/W	[0] Divider Reset Control 0: Assert Reset	0x00000

Bits	Name	Access	Description	Reset
			1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	001

div_clk_spi_nand

Offset Address: 0x088

Bits	Name	Access	Description	Reset
31:0	div_clk_spi_nand	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : fpll 1 : disppll	0x00000001

div_clk_500m_eth0

Offset Address: 0x08c

Bits	Name	Access	Description	Reset
31:0	div_clk_500m_eth0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_gpio_db

Offset Address: 0x094

Bits	Name	Access	Description	Reset
31:0	div_clk_gpio_db	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_sdma_aud0

Offset Address: 0x098

Bits	Name	Access	Description	Reset
31:0	div_clk_sdma_aud0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : apll 1 : a24k	0x00000001

div_clk_sdma_aud1

Offset Address: 0x09c

Bits	Name	Access	Description	Reset
31:0	div_clk_sdma_aud1	R/W	[0] Divider Reset Control 0: Assert Reset	0x00000

Bits	Name	Access	Description	Reset
			1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : apll 1 : a24k	001

div_clk_sdma_aud2

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
31:0	div_clk_sdma_aud2	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : apll 1 : a24k	0x00000001

div_clk_sdma_aud3

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
31:0	div_clk_sdma_aud3	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : apll 1 : a24k	0x00000001

div_clk_cam0_200

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
31:0	div_clk_cam0_200	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : xtal 1 : disppll	0x00000001

div_clk_axi4

Offset Address: 0x0b8

Bits	Name	Access	Description	Reset
31:0	div_clk_axi4	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register	0x00000001

Bits	Name	Access	Description	Reset
			[20:16] Clock Divider Factor [9:8] clk_src 0 : fpll 1 : disppll	

div_clk_axi6

Offset Address: 0x0bc

Bits	Name	Access	Description	Reset
31:0	div_clk_axi6	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_dsi_esc

Offset Address: 0x0c4

Bits	Name	Access	Description	Reset
31:0	div_clk_dsi_esc	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_axi_vip

Offset Address: 0x0c8

Bits	Name	Access	Description	Reset
31:0	div_clk_axi_vip	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : mipimpll 1 : cam0pll 2 : disppll 3 : fpll	0x00000001

div_clk_src_vip_sys_0

Offset Address: 0x0d0

Bits	Name	Access	Description	Reset
31:0	div_clk_src_vip_sys_0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : mipimpll 1 : cam0pll 2 : disppll 3 : fpll	0x00000301

div_clk_src_vip_sys_1

Offset Address: 0x0d8

Bits	Name	Access	Description	Reset
31:0	div_clk_src_vip_sys_1	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : mipimpll 1 : cam0pll 2 : disppll 3 : fpll	0x00000301

div_clk_disp_src_vip

Offset Address: 0x0e0

Bits	Name	Access	Description	Reset
31:0	div_clk_disp_src_vip	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_axi_video_codec

Offset Address: 0x0e4

Bits	Name	Access	Description	Reset
31:0	div_clk_axi_video_codec	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : apll 1 : mipimpll 2 : cam1pll 3 : fpll	0x00000101

div_clk_vc_src0

Offset Address: 0x0ec

Bits	Name	Access	Description	Reset
31:0	div_clk_vc_src0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : apll 1 : mipimpll 2 : cam1pll 3 : fpll	0x00000101

div_clk_1m

Offset Address: 0x0fc

Bits	Name	Access	Description	Reset
31:0	div_clk_1m	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_spi

Offset Address: 0x100

Bits	Name	Access	Description	Reset
31:0	div_clk_spi	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_i2c

Offset Address: 0x104

Bits	Name	Access	Description	Reset
31:0	div_clk_i2c	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_src_vip_sys_2

Offset Address: 0x110

Bits	Name	Access	Description	Reset
31:0	div_clk_src_vip_sys_2	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : mipimpll 1 : cam0pll 2 : disppll 3 : fpll	0x00000201

div_clk_audsrc

Offset Address: 0x118

Bits	Name	Access	Description	Reset
31:0	div_clk_audsrc	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : apll 1 : a24k	0x00000001

div_clk_pwm_src_0

Offset Address: 0x120

Bits	Name	Access	Description	Reset
31:0	div_clk_pwm_src	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : fpll 1 : disppll	0x00000001

div_clk_ap_debug

Offset Address: 0x128

Bits	Name	Access	Description	Reset
31:0	div_clk_ap_debug	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_rtcsys_src_0

Offset Address: 0x12c

Bits	Name	Access	Description	Reset
31:0	div_clk_src_rtc_sys_0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor	0x00000001

div_clk_c906_0_0

Offset Address: 0x130

Bits	Name	Access	Description	Reset
31:0	div_clk_c906_0_0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [1] High Wide Control (when Divider Factor is odd) 0: Low level of the clock is wider 1: High level of the clock is wider [2] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [3] Select High Wide Control from Register 0: Select initial value 1: Select High Wide from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : tpll 1 : apll 2 : mipimpll 3 : mpll	0x00000201

div_clk_c906_0_1

Offset Address: 0x134

Bits	Name	Access	Description	Reset
31:0	div_clk_c906_0_1	R/W	[0] Divider Reset Control 0: Assert Reset	0x00000

Bits	Name	Access	Description	Reset
			1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : fpll	001

div_clk_c906_1_0

Offset Address: 0x138

Bits	Name	Access	Description	Reset
31:0	div_clk_c906_1_0	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [1] High Wide Control (when Divider Factor is odd) 0: Low level of the clock is wider 1: High level of the clock is wider [2] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [3] Select High Wide Control from Register 0: Select initial value 1: Select High Wide from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : tpll 1 : apll 2 : mipimpll 3 : mpll	0x00000001

div_clk_c906_1_1

Offset Address: 0x13c

Bits	Name	Access	Description	Reset
31:0	div_clk_c906_1_1	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : fpll	0x00000001

div_clk_src_vip_sys_3

Offset Address: 0x140

Bits	Name	Access	Description	Reset
31:0	div_clk_src_vip_sys_3	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : mipimpll 1 : cam0pll 2 : disppll 3 : fpll	0x00000001

div_clk_src_vip_sys_4

Offset Address: 0x144

Bits	Name	Access	Description	Reset
	div_clk_src_vip_sys_4	R/W	[0] Divider Reset Control 0: Assert Reset 1: De-assert Reset [3] Select Divide Factor from Register 0: Select initial value 1: Select Divide Factor from this register [20:16] Clock Divider Factor [9:8] clk_src 0 : mipimpll 1 : cam0pll 2 : disppll 3 : fpll	0x00000201

7 System Controller

7.1 summarize

The system controller realizes some system controls on the chip through registers, including system soft reset, clock control, and so on. Reset clock has been described in other chapters, this chapter describes some other system function module configuration and status registers.

7.2 Functional Description

7.2.1 Global reset enable

The system global soft reset, Debug reset and Watch Dog reset need to be enabled by configuring the reg_sw_root_reset_en register. The details are described in the reg_sw_root_reset_en register.

7.2.2 System DMA Channel Mapping

The system DMA of this chip has 8 built-in channels, each configured with 0~7 channel request interface. The 0~7 channel requests are mapped from the system control registers sdma_dma_ch_remap0, sdma_dma_ch_remap1 to the peripheral interfaces in the following table.

one. Note that you cannot configure multiple channels to the same peripheral interface.

Configuration Steps:

Configure the DMA channel image registers sdma_dma_ch_remap0, sdma_dma_ch_remap1, update_dma_remp_0_3, update_dma_remp_4_7

Write 1 to enable the mapping.

serial number	DMA interface	serial number	DMA interface
0	dma_rx_req_i2s0	24	dma_rx_req_i2c0
1	dma_tx_req_i2s0	25	dma_tx_req_i2c0
2	dma_rx_req_i2s1	26	dma_rx_req_i2c1
3	dma_tx_req_i2s1	27	dma_tx_req_i2c1
4	dma_rx_req_i2s2	28	dma_rx_req_i2c2

5	dma_tx_req_i2s2	29	dma_tx_req_i2c2
6	dma_rx_req_i2s3	30	dma_rx_req_i2c3
7	dma_tx_req_i2s3	31	dma_tx_req_i2c3
8	dma_rx_req_n_uart0	32	dma_rx_req_i2c4
9	dma_tx_req_n_uart0	33	dma_tx_req_i2c4

serial number	DMA interface	serial number	DMA interface
10	dma_rx_req_n_uart1	34	dma_rx_req_tdm0
11	dma_tx_req_n_uart1	35	dma_tx_req_tdm0
12	dma_rx_req_n_uart2	36	dma_rx_req_tdm1
13	dma_tx_req_n_uart2	37	dma_req_audsrc
14	dma_rx_req_n_uart3	38	dma_req_spi_nand
15	dma_tx_req_n_uart3	39	dma_req_spi_nor
16	dma_rx_req_spi0	40	dma_rx_req_n_uart4
17	dma_tx_req_spi0	41	dma_tx_req_n_uart4
18	dma_rx_req_spi1	42	dma_req_spi_nor1
19	dma_tx_req_spi1		
20	dma_rx_req_spi2		
21	dma_tx_req_spi2		
22	dma_rx_req_spi3		
23	dma_tx_req_spi3		

7.2.3 DDR AXI Urgent/Qos Configuration

The AXI transmission priority from the subsystem can be configured by ddr_axi_urgent_ow, ddr_axi_urgent, ddr_axi_qos_0, and ddr_axi_qos_1 of the control system controller, as described in detail in the section on DDR Controller.

7.3 System Control Register

7.3.1 System Control Registers Overview

Bass address 0x03000000

Name	Address Offset	Description
conf_info	0x004	conf_info
sys_ctrl_reg	0x008	sys_ctrl_reg
usb_phy_ctrl_reg	0x048	usb_phy_ctrl_reg
sdma_dma_ch_remap0	0x154	sdma_dma_ch_remap0
sdma_dma_ch_remap1	0x158	sdma_dma_ch_remap1
top_timer_clk_sel	0x1a0	top_timer_clk_sel
top_wdt_ctrl	0x1a8	top_timer_clk_sel
ddr_axi_urgent_ow	0x1b8	ddr_axi_urgent_ow
ddr_axi_urgent	0x1bc	ddr_axi_urgent
ddr_axi_qos_0	0x1d8	ddr_axi_qos_0
ddr_axi_qos_1	0x1dc	ddr_axi_qos_1
sd_pwrsw_ctrl	0x1f4	sd_pwrsw_ctrl
sd_pwrsw_time	0x1f8	sd_pwrsw_time

ddr_axi_qos_ow	0x23c	ddr_axi_qos_ow
sd_ctrl_opt	0x294	additional control register for sd

Name	Address Offset	Description
sdma_dma_int_mux	0x298	Mux sdma channel interrupt to different processors

7.3.2 System Control Register Description

conf_info

Offset Address: 0x004

Bits	Name	Access	Description	Reset
7:0	boot_sel	RO	[2:0] boot device selection 0: SPI_NAND 1: reserved 2: SPI_NOR 3: EMMC [7:3] : resreved	
8	io_sta_usbid	RO	IO status from USBID PAD	
9	io_sta_usbvbus	RO	IO status from USB_VBUS_DET PAD	
23:10	Reserved			
31:24	io_sta_trap	RO	io_sta_trap[0] : io_boot_rom_din io_sta_trap[1] : io_boot_dev0_din io_sta_trap[2] : io_boot_dev1_din io_sta_trap[3] : io_trap_sd0_pwr_din io_ sta_trap[4] : io_pkg_type0_din io_sta_trap[5] : io_pkg_type1_din io_sta_trap[6] : io_pkg_type2_din io_sta_trap[7] : io_trap_zq_din	

sys_ctrl_reg

Offset Address: 0x008

Bits	Name	Access	Description	Reset
1:0	Reserved			
5:2	reg_sw_root_reset_en	R/W	bit0 : wdt reset enable bit1 : cdbgrstreq enable bit2 : reserved bit3 : reg_soft_reset_x_system enable	0x0
31:6	Reserved			

usb_phy_ctrl_reg

Offset Address: 0x048

Bits	Name	Access	Description	Reset
0	reg_usb_phy_external_vbusvalid	R/W	external vbus status	0x0
1	reg_usb_drive_vbus	R/W	drive vbus power	0x0
4:2	Reserved			
5	toreg_usb_id_en	RO	usb id pullup status	
6	reg_usb_phy_idpad_c_ow	R/W	usb id overwrite enable	0x0
7	reg_usb_phy_idpad_c_sw	R/W	usb id overwrite value	0x0
8	io_usb_phy_idpad_c	RO	usb id external IO pin status	
9	toreg_usb_phy_idpad_c	RO	usb id pin status	
31:10	Reserved			

sdma_dma_ch_remap0

Offset Address: 0x154

Bits	Name	Access	Description	Reset
5:0	reg_dma_remap_ch0	R/W	dma channel 0 mapping	0x0
7:6	Reserved			

Bits	Name	Access	Description	Reset
13:8	reg_dma_remap_ch1	R/W	dma channel 1 mapping	0x0
15:14	Reserved			
21:16	reg_dma_remap_ch2	R/W	dma channel 2 mapping	0x0
23:22	Reserved			
29:24	reg_dma_remap_ch3	R/W	dma channel 3 mapping	0x0
30	Reserved			
31	update_dma_remp_0_3	W1T	write 1 to update dma channel0~3 mapping	

sdma_dma_ch_remap1

Offset Address: 0x158

Bits	Name	Access	Description	Reset
5:0	reg_dma_remap_ch4	R/W	dma channel 4 mapping	0x0
7:6	Reserved			
13:8	reg_dma_remap_ch5	R/W	dma channel 5 mapping	0x0
15:14	Reserved			
21:16	reg_dma_remap_ch6	R/W	dma channel 6 mapping	0x0
23:22	Reserved			
29:24	reg_dma_remap_ch7	R/W	dma channel 7 mapping	0x0
30	Reserved			
31	update_dma_remp_4_7	W1T	write 1 to update dma channel4~7 mapping	

top_timer_clk_sel

Offset Address: 0x1a0

Bits	Name	Access	Description	Reset
7:0	reg_timer_clk_sel	R/W	timer0~7 clock selection. 0: xtal clock,1:32k clock	0x0
31:8	Reserved			

top_wdt_ctrl

Offset Address: 0x1a8

Bits	Name	Access	Description	Reset
2:0	reg_wdt_rst_sys_en	R/W	enable wdt0~wdt2 to reset system	0x7
3	Reserved			
6:4	reg_wdt_rst_cpu_en	R/W	enable wdt0~wdt2 to reset cpu	0x0
7	Reserved			
10:8	reg_wdt_clk_sel	R/W	top_wdt clock selection. 0: xtal clock,1:32k clock	0x0
31:11	Reserved			

ddr_axi_urgent_ow

Offset Address: 0x1b8

Bits	Name	Access	Description	Reset
0	reg_awurgent_m1_ow	R/W	ddr axi port1 awurgent overwrite enable	0x1
1	reg_arurgent_m1_ow	R/W	ddr axi port1 arurgent overwrite enable	0x1
2	reg_awurgent_m2_ow	R/W	ddr axi port2 awurgent overwrite enable	0x1
3	reg_arurgent_m2_ow	R/W	ddr axi port2 arurgent overwrite enable	0x1
4	reg_awurgent_m3_ow	R/W	ddr axi port3 awurgent overwrite	0x1

Bits	Name	Access	Description	Reset
			enable	
5	reg_arurgent_m3_ow	R/W	ddr axi port3 arurgent overwrite enable	0x1
6	reg_awurgent_m4_ow	R/W	ddr axi port4 awurgent overwrite enable	0x1
7	reg_arurgent_m4_ow	R/W	ddr axi port4 arurgent overwrite enable	0x1
8	reg_awurgent_m5_ow	R/W	ddr axi port5 awurgent overwrite enable	0x1
9	reg_arurgent_m5_ow	R/W	ddr axi port5 arurgent overwrite enable	0x1
10	reg_awurgent_m6_ow	R/W	ddr axi port6 awurgent overwrite enable	0x1
11	reg_arurgent_m6_ow	R/W	ddr axi port6 arurgent overwrite enable	0x1
31:12	Reserved			

ddr_axi_urgent

Offset Address: 0x1bc

Bits	Name	Access	Description	Reset
0	reg_awurgent_m1	R/W	ddr axi port1 awurgent overwrite value	0x0
1	reg_arurgent_m1	R/W	ddr axi port1 arurgent overwrite value	0x0
2	reg_awurgent_m2	R/W	ddr axi port2 awurgent overwrite value	0x0
3	reg_arurgent_m2	R/W	ddr axi port2 arurgent overwrite value	0x0
4	reg_awurgent_m3	R/W	ddr axi port3 awurgent overwrite value	0x0
5	reg_arurgent_m3	R/W	ddr axi port3 arurgent overwrite value	0x0
6	reg_awurgent_m4	R/W	ddr axi port4 awurgent overwrite value	0x0
7	reg_arurgent_m4	R/W	ddr axi port4 arurgent overwrite value	0x0
8	reg_awurgent_m5	R/W	ddr axi port5 awurgent overwrite value	0x0
9	reg_arurgent_m5	R/W	ddr axi port5 arurgent overwrite value	0x0
10	reg_awurgent_m6	R/W	ddr axi port6 awurgent overwrite value	0x0
11	reg_arurgent_m6	R/W	ddr axi port6 arurgent overwrite value	0x0
31:12	Reserved			

ddr_axi_qos_0

Offset Address: 0x1d8

Bits	Name	Access	Description	Reset
3:0	reg_awqos_m1	R/W	ddr axi port1 awqos setting	0x0
7:4	reg_arqos_m1	R/W	ddr axi port1 arqos setting	0x0
11:8	reg_awqos_m2	R/W	ddr axi port2 awqos setting	0x0
15:12	reg_arqos_m2	R/W	ddr axi port2 arqos setting	0x0
19:16	reg_awqos_m3	R/W	ddr axi port3 awqos setting	0x0
23:20	reg_arqos_m3	R/W	ddr axi port3 arqos setting	0x0
27:24	reg_awqos_m4	R/W	ddr axi port4 awqos setting	0x0
31:28	reg_arqos_m4	R/W	ddr axi port4 arqos setting	0x0

ddr_axi_qos_1

Offset Address: 0x1dc

Bits	Name	Access	Description	Reset
3:0	reg_awqos_m5	R/W	ddr axi port5 awqos setting	0x0
7:4	reg_arqos_m5	R/W	ddr axi port5 arqos setting	0x0
11:8	reg_awqos_m6	R/W	ddr axi port6 awqos setting	0x0
15:12	reg_arqos_m6	R/W	ddr axi port6 arqos setting	0x0
31:16	Reserved			

sd_pwrsw_ctrl

Offset Address: 0x1f4

Bits	Name	Access	Description	Reset
0	reg_en_pwrsw	R/W	18/33 IO power switch enable	0x0
1	reg_pwrsw_vsel	R/W	18/33 IO power switch enable 0: 3.3v 1: 1.8v	0x1
2	reg_pwrsw_disc	R/W	18/33 IO power switch discharge enable	0x0
3	reg_pwrsw_auto	R/W	18/33 IO power switch auto protect enable	0x1
31:4	Reserved			

sd_pwrsw_time

Offset Address: 0x1f8

Bits	Name	Access	Description	Reset
15:0	reg_tpwrup	R/W	18/33 IO power switch, power up protection time is 500x40ns = 20us	0x1f4
31:16	reg_tpwrdn	R/W	18/33 IO power switch, power down protection time is 500x40ns = 20us	0x1f4

ddr_axi_qos_ow

Offset Address: 0x23c

Bits	Name	Access	Description	Reset
0	reg_awqos_m1_ow	R/W	ddr axi port1 awqos overwrite enable	0x1
1	reg_arqos_m1_ow	R/W	ddr axi port1 arqos overwrite enable	0x1
2	reg_awqos_m2_ow	R/W	ddr axi port2 awqos overwrite enable	0x1
3	reg_arqos_m2_ow	R/W	ddr axi port2 arqos overwrite enable	0x1
4	reg_awqos_m3_ow	R/W	ddr axi port3 awqos overwrite enable	0x1
5	reg_arqos_m3_ow	R/W	ddr axi port3 arqos overwrite enable	0x1
6	reg_awqos_m4_ow	R/W	ddr axi port4 awqos overwrite enable	0x1
7	reg_arqos_m4_ow	R/W	ddr axi port4 arqos overwrite enable	0x1
8	reg_awqos_m5_ow	R/W	ddr axi port5 awqos overwrite enable	0x1
9	reg_arqos_m5_ow	R/W	ddr axi port5 arqos overwrite enable	0x1
10	reg_awqos_m6_ow	R/W	ddr axi port6 awqos overwrite enable	0x1
11	reg_arqos_m6_ow	R/W	ddr axi port6 arqos overwrite enable	0x1
31:12	Reserved			

sd_ctrl_opt

Offset Address: 0x294

Bits	Name	Access	Description	Reset
0	reg_sd0_carddet_ow	R/W	sd0 card detect over write enable	0x0
1	reg_sd0_carddet_sw	R/W	sd0 card detect over write value	0x0
7:2	Reserved			
8	reg_sd1_carddet_ow	R/W	sd1 card detect over write enable	0x0
9	reg_sd1_carddet_sw	R/W	sd1 card detect over write value	0x0
10	reg_sd1_phy_sel	R/W	sd1 phy sel	0x0
15:11	Reserved			
16	reg_sd0_pwr_en_polarity	R/W	off chip sd0 pwr en polarity 0: SD_LDO power ctrl high is power on , low is power off 1: SD_LDO power ctrl high is power off , low is power on	0x0
31:17	Reserved			

sdma_dma_int_mux

Offset Address: 0x298

Bits	Name	Access	Description	Reset
8:0	reg_dma_int_mux_cpu0	R/W	This register is used to mux separate sdma channel interrupts to CPU0. These are enable bits corresponding to {intr_cmnreg,intr_ch[7:0]}	0x1FF
9	Reserved			
18:10	reg_dma_int_mux_cpu1	R/W	This register is used to mux separate sdma channel interrupts to CPU1. These are enable bits corresponding to {intr_cmnreg,intr_ch[7:0]}	0x0
19	Reserved			
28:20	reg_dma_int_mux_cpu2	R/W	This register is used to mux separate sdma channel interrupts to CPU2. These are enable bits corresponding to {intr_cmnreg,intr_ch[7:0]}	0x0
31:29	Reserved			

8 Pin Multiplexing and Control

8.1 Pin Multiplexing PINMUX

8.1.1 Correspondence of interface functions to signals/pins/FMUX registers

Table *8-1* Correspondence Table between Interface Functions and *Signals/Pins/FMUX* Registers

Interface	Signal Name	Direction	PinName	Function Number	Function select register
ADC	(ADC1)	I	ADC1	0	fmux_gpio_reg_ioctrl_adc1 0x0300_10f8
No die domain ADC	PWR_GPIO[2](PWRSAR.VIN1)	I	PWR_GPIO2	0	FMUX_GPIO_REG_IOCTRL_PWR_GPIO2 0x0300_10AC
	PWR_GPIO[1](PWRSAR1.VIN2)	I	PWR_GPIO1	0	FMUX_GPIO_REG_IOCTRL_PWR_GPIO1 0x0300_10A8
	PWR_VBAT_DET(PWRSAR.VIN3)	I	PWR_VBAT_DET	0	FMUX_GPIO_REG_IOCTRL_PWR_VBAT_DET 0x0300_107C
Audio	(PAD_AUD_AINL_MIC)	I	PAD_AUD_AINL_MIC	0	FMUX_GPIO_REG_IOCTRL_PAD_AUD_AINL_MIC 0x0300_11BC
	(PAD_AUD_AOUTR)	O	PAD_AUD_AOUTR	0	FMUX_GPIO_REG_IOCTRL_PAD_AUD_AOUTR 0x0300_11C8
Ethernet	(PAD_ETH_RXM)	I/O	PAD_ETH_RXM EPHY_T XP	0	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXM 0x0300_1130
	(PAD_ETH_RXP)	I/O	PAD_ETH_RXP EPHY_T XN	0	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXP 0x0300_112C
	(PAD_ETH_TXM)	I/O	PAD_ETH_TXM EPHY_R XP	0	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXM 0x0300_1128
	(PAD_ETH_TXP)	I/O	PAD_ETH_TXP EPHY_R XN	0	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXP 0x0300_1124
	EPHY_LNK_LED	O	SD0_CLK	6	fmux_gpio_reg_ioctrl_sd0_c lk 0x0300_101C
	EPHY_LNK_LED	O	PWR_WAKEUP0	6	fmux_gpio_reg_ioctrl_pwr_wakeup0 0x0300_1090
	EPHY_LNK_LED	o	SD1_CMD	5	FMUX_GPIO_REG_IOCTRL_SD1_CMD 0x0300_10E0

	EPHY_LNK_LED	0	PWR_GPIO1	3	FMUX_GPIO_REG_IOCTRL_PWR_GPIO1 0x0300_10A8

	EPHY_SPD_LED	O	SD0_CMD	6	FMUX_GPIO_REG_IOCTRL_SD0_CMD 0x0300_1020
	EPHY_SPD_LED	O	PWR_BUTTON1	6	FMUX_GPIO_REG_IOCTRL_PWR_BUTTON1 0x0300_1098
	EPHY_SPD_LED	O	SD1_CLK	5	fmux_gpio_reg_ioctrl_sd1_c lk 0x0300_10E4
	EPHY_SPD_LED	O	PWR_GPIO2	3	FMUX_GPIO_REG_IOCTRL_PWR_GPIO2 0x0300_10AC
DSI/LVDS	(PAD_MIPI_TXM0)	I/O	PAD_MIPI_TXM0	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM0 0x0300_11B4
	(PAD_MIPI_TXP0)	I/O	PAD_MIPI_TXP0	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP0 0x0300_11B8
	(PAD_MIPI_TXM1)	I/O	PAD_MIPI_TXM1	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM1 0x0300_11AC
	(PAD_MIPI_TXP1)	I/O	PAD_MIPI_TXP1	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP1 0x0300_11B0
	(PAD_MIPI_TXM2)	I/O	PAD_MIPI_TXM2	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM2 0x0300_11A4
	(PAD_MIPI_TXP2)	I/O	PAD_MIPI_TXP2	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP2 0x0300_11A8
CSI/sLVDS/HiSPI	(PAD_MIPIRX0N)	I/O	PAD_MIPIRX0N	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0N 0x0300_118C
	(PAD_MIPIRX0P)	I/O	PAD_MIPIRX0P	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0P 0x0300_1190
	(PAD_MIPIRX1N)	I/O	PAD_MIPIRX1N	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1N 0x0300_1184
	(PAD_MIPIRX1P)	I/O	PAD_MIPIRX1P	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1P 0x0300_1188
	(PAD_MIPIRX2N)	I/O	PAD_MIPIRX2N	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2N 0x0300_117C
	(PAD_MIPIRX2P)	I/O	PAD_MIPIRX2P	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2P 0x0300_1180
	(PAD_MIPIRX3N)	I/O	PAD_MIPIRX3N	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3N 0x0300_1174
	(PAD_MIPIRX3P)	I/O	PAD_MIPIRX3P	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3P 0x0300_1178
	(PAD_MIPIRX4N)	I/O	PAD_MIPIRX4N	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4N 0x0300_116C
	(PAD_MIPIRX4P)	I/O	PAD_MIPIRX4P	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4P 0x0300_1170

Aux clock out	AUX0	O	AUX0	0	FMUX_GPIO_REG_IOCTRL_AUX0 0x0300_1078
	AUX0	O	SD0_D1	2	fmux_gpio_reg_ioctrl_sd0_d 1 0x0300_1028
	AUX0	O	UART0_RX	5	FMUX_GPIO_REG_IOCTRL_UART0 _RX 0x0300_1044
	AUX0	O	JTAG_CPU_T MS	5	FMUX_GPIO_REG_IOCTRL_JTAG_ CPU_TMS 0x0300_1064
	AUX1	O	SD0_D2	2	fmux_gpio_reg_ioctrl_sd0_d 2 0x0300_102C
	AUX1	O	UART0_TX	5	FMUX_GPIO_REG_IOCTRL_UART0 _TX 0x0300_1040
	AUX1	O	JTAG_CPU_T CK	5	FMUX_GPIO_REG_IOCTRL_JTAG_ CPU_TCK 0x0300_1068
Camera Interface	CAM_HS0	O	SD1_CLK	4	fmux_gpio_reg_ioctrl_sd1_c lk 0x0300_10E4
	CAM_HS0	O	PAD_MIPI_T XP0	6	FMUX_GPIO_REG_IOCTRL_PAD_M IPI_TXP0 0x0300_11B8
	CAM_HS0	O	PAD_ETH_RX P EPHY_T XN	5	FMUX_GPIO_REG_IOCTRL_PAD_E TH_RXP 0x0300_112C
	CAM_MCLK0	O	PAD_MIPI_T XP0	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPI_TXP0 0x0300_11B8
	CAM_MCLK0	o	SD0_D3	1	fmux_gpio_reg_ioctrl_sd0_d 3 0x0300_1030
	CAM_MCLK0	o	UART0_RX	1	FMUX_GPIO_REG_IOCTRL_UART0 _RX 0x0300_1044
	CAM_MCLK0	o	JTAG_CPU_T MS	1	FMUX_GPIO_REG_IOCTRL_JTAG_ CPU_TMS 0x0300_1064
	CAM_MCLK0	o	SD1_D3	4	fmux_gpio_reg_ioctrl_sd1_d 3 0x0300_10D0
	CAM_MCLK0	O	SD1_D2	4	fmux_gpio_reg_ioctrl_sd1_d 2 0x0300_10D4
	CAM_MCLK0	O	USB_VBUS_D ET	4	fmux_gpio_reg_ioctrl_usb_v bus_det 0x0300_1108
	CAM_MCLK0	o	PAD_MIPIRX 3N	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX3N 0x0300_1174
	CAM_MCLK0	o	PAD_MIPIRX OP	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX0P 0x0300_1190
	CAM_MCLK0	O	MUX_SPI1_C S	2	fmux_gpio_reg_ioctrl_mux_s pi1_cs 0x0300_111C

	CAM_MCLK0	O	PAD_ETH_RX M EPHY_T XP	2	FMUX_GPIO_REG_IOCTRL_PAD_E TH_RXM 0x0300_1130
	CAM_MCLK0	O	PAD_ETH_TX P EPHY_R XN	5	FMUX_GPIO_REG_IOCTRL_PAD_E TH_TXP 0x0300_1124
	CAM_MCLK0	o	PAD_MIPIRX 4N	5	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX4N 0x0300_116C
	CAM_MCLK0	O	G P I O _ _ R T X _ _ _EPHY_RTX	5	FMUX_GPIO_REG_IOCTRL_GPIO_ RTX 0x0300_11CC
	CAM_MCLK1	O	SD1_D1	4	fmux_gpio_reg_ioctrl_sd1_d 1 0x0300_10D8
	CAM_MCLK1	O	SD1_D0	4	fmux_gpio_reg_ioctrl_sd1_d 0 0x0300_10DC
	CAM_MCLK1	O	PAD_MIPI_T XM0	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPI_TXM0 0x0300_11B4
	CAM_MCLK1	O	MUX_SPI1_S CK	2	fmux_gpio_reg_ioctrl_mux_s pi1_sck 0x0300_1120
	CAM_MCLK1	O	PAD_ETH_RX P EPHY_T XN	2	FMUX_GPIO_REG_IOCTRL_PAD_E TH_RXP 0x0300_112C
	CAM_MCLK1	O	USB_VBUS_D ET	5	fmux_gpio_reg_ioctrl_usb_v bus_det 0x0300_1108
	CAM_MCLK1	O	PAD_ETH_TX M EPHY_R XP	5	FMUX_GPIO_REG_IOCTRL_PAD_E TH_TXM 0x0300_1128
	CAM_MCLK1	O	PAD_MIPIRX 4P	5	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX4P 0x0300_1170
	CAM_MCLK1	O	PAD_MIPIRX 0N	5	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX0N 0x0300_118C
	CAM_MCLK1	O	SD0_D0	1	fmux_gpio_reg_ioctrl_sd0_d 0 0x0300_1024
	CAM_MCLK1	o	UART0_TX	1	FMUX_GPIO_REG_IOCTRL_UART0 _TX 0x0300_1040
	CAM_MCLK1	O	JTAG_CPU_T CK	1	FMUX_GPIO_REG_IOCTRL_JTAG_ CPU_TCK 0x0300_1068
	CAM_VS0	O	SD1_CMD	4	FMUX_GPIO_REG_IOCTRL_SD1_C MD 0x0300_10E0
	CAM_VS0	O	PAD_MIPI_T XM0	6	FMUX_GPIO_REG_IOCTRL_PAD_M IPI_TXM0 0x0300_11B4
	CAM_VS0	O	PAD_ETH_RX M EPHY_T XP	5	FMUX_GPIO_REG_IOCTRL_PAD_E TH_RXM 0x0300_1130
Parallel Video Out	VO_CLK0	O	PAD_MIPI_T XP2	2	FMUX_GPIO_REG_IOCTRL_PAD_M IPI_TXP2 0x0300_11A8

	VO_D[0]	O	PAD_MIPI_TXM2	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM2 0x0300_11A4
	VO_D[1]	O	PAD_MIPI_TXP1	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP1 0x0300_11B0
	VO_D[2]	O	PAD_MIPI_TXM1	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM1 0x0300_11AC
	VO_D[3]	o	PAD_MIPI_TXP0	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP0 0x0300_11B8
	VO_D[4]	o	PAD_MIPI_TXM0	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM0 0x0300_11B4
	VO_D[5]	o	PAD_MIPIRX0P	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0P 0x0300_1190
	VO_D[6]	o	PAD_MIPIRX0N	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0N 0x0300_118C
	VO_D[7]	O	PAD_MIPIRX1P	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1P 0x0300_1188
	VO_D[8]	O	PAD_MIPIRX1N	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1N 0x0300_1184
	VO_D[9]	O	PAD_MIPIRX2P	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2P 0x0300_1180
	VO_D[10]	o	PAD_MIPIRX2N	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2N 0x0300_117C
	VO_D[28]	O	JTAG_CPU_TMS	7	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TMS 0x0300_1064
	VO_D[29]	O	JTAG_CPU_TCK	7	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TCK 0x0300_1068
	VO_D[31]	O	AUX0	5	FMUX_GPIO_REG_IOCTRL_AUX0 0x0300_1078
	VO_D[32]	O	SD1_D3	0	fmux_gpio_reg_ioctrl_sd1_d 3 0x0300_10D0
	VO_D[33]	O	SD1_D2	0	fmux_gpio_reg_ioctrl_sd1_d 2 0x0300_10D4
	VO_D[34]	O	SD1_D1	0	fmux_gpio_reg_ioctrl_sd1_d 1 0x0300_10D8
	VO_D[35]	O	SD1_D0	0	fmux_gpio_reg_ioctrl_sd1_d 0 0x0300_10DC
	VO_D[36]	O	SD1_CMD	0	FMUX_GPIO_REG_IOCTRL_SD1_CMD 0x0300_10E0
	VO_D[37]	o	SD1_CLK	0	fmux_gpio_reg_ioctrl_sd1_c lk 0x0300_10E4

Parallel Video In	VI0_CLK	I	PAD_MIPIRX 4N	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4N 0x0300_116C
	VI0_D[0]	I	PAD_MIPIRX 4P	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4P 0x0300_1170
	VI0_D[1]	I	PAD_MIPIRX 3N	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3N 0x0300_1174
	VI0_D[10]	I	PAD_MIPI_TXP0	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP0 0x0300_11B8
	VI0_D[11]	I	PAD_MIPI_TXM1	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM1 0x0300_11AC
	VI0_D[12]	I	PAD_MIPI_TXP1	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP1 0x0300_11B0
	VI0_D[13]	I	PAD_MIPI_TXM2	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM2 0x0300_11A4
	VI0_D[14]	I	PAD_MIPI_TXP2	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP2 0x0300_11A8
	VI0_D[2]	I	PAD_MIPIRX 3P	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3P 0x0300_1178
	VI0_D[3]	I	PAD_MIPIRX 2N	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2N 0x0300_117C
	VI0_D[4]	I	PAD_MIPIRX 2P	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2P 0x0300_1180
	VI0_D[5]	I	PAD_MIPIRX 1N	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1N 0x0300_1184
	VI0_D[6]	I	PAD_MIPIRX 1P	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1P 0x0300_1188
	VI0_D[7]	I	PAD_MIPIRX 0N	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0N 0x0300_118C
	VI0_D[8]	I	PAD_MIPIRX 0P	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0P 0x0300_1190
	VI0_D[9]	I	PAD_MIPI_TXM0	1	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM0 0x0300_11B4
	VI1_D[13]	I	PAD_MIPIRX 4N	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4N 0x0300_116C
	VI1_D[14]	I	PAD_MIPIRX 4P	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4P 0x0300_1170
	VI1_D[15]	I	PAD_MIPIRX 3N	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3N 0x0300_1174
	VI1_D[16]	I	PAD_MIPIRX 3P	2	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3P 0x0300_1178

	VI1_D[17]	I	PAD_MIPIRX 2N	4	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2N 0x0300_117C
	VI1_D[18]	I	PAD_MIPIRX 2P	4	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2P 0x0300_1180
eMMC	EMMC_CLK	O	EMMC_CLK	0	fmux_gpio_reg_ioctrl_emmc_ clk 0x0300_1050
	EMMC_CMD	I/O	EMMC_CMD	0	fmux_gpio_reg_ioctrl_emmc_ cmd 0x0300_105C
	EMMC_DAT[0]	I/O	EMMC_DAT0	0	fmux_gpio_reg_ioctrl_emmc_ dat0 0x0300_1054
	EMMC_DAT[1]	I/O	EMMC_DAT1	0	fmux_gpio_reg_ioctrl_emmc_ dat1 0x0300_1060
	EMMC_DAT[2]	I/O	EMMC_DAT2	0	fmux_gpio_reg_ioctrl_emmc_ dat2 0x0300_104C
	EMMC_DAT[3]	I/O	EMMC_DAT3	0	fmux_gpio_reg_ioctrl_emmc_ dat3 0x0300_1058
SPI_NAND	SPINAND_CLK	O	EMMC_CLK	2	fmux_gpio_reg_ioctrl_emmc_ clk 0x0300_1050
	SPINAND_CS	O	EMMC_DAT1	2	fmux_gpio_reg_ioctrl_emmc_ dat1 0x0300_1060
	SPINAND_HOLD	I/O	EMMC_DAT2	2	fmux_gpio_reg_ioctrl_emmc_ dat2 0x0300_104C
	SPINAND_MISO	I/O	EMMC_CMD	2	fmux_gpio_reg_ioctrl_emmc_ cmd 0x0300_105C
	SPINAND_MOSI	I/O	EMMC_DAT0	2	fmux_gpio_reg_ioctrl_emmc_ dat0 0x0300_1054
	SPINAND_WP	I/O	EMMC_DAT3	2	fmux_gpio_reg_ioctrl_emmc_ dat3 0x0300_1058
SPI_NOR	SPINOR_CS_X	O	EMMC_DAT1	1	fmux_gpio_reg_ioctrl_emmc_ dat1 0x0300_1060
	SPINOR_HOLD_X	I/O	EMMC_DAT2	1	fmux_gpio_reg_ioctrl_emmc_ dat2 0x0300_104C
	SPINOR_MISO	I/O	EMMC_CMD	1	fmux_gpio_reg_ioctrl_emmc_ cmd 0x0300_105C
	SPINOR_MOSI	I/O	EMMC_DAT0	1	fmux_gpio_reg_ioctrl_emmc_ dat0 0x0300_1054
	SPINOR_SCK	O	EMMC_CLK	1	fmux_gpio_reg_ioctrl_emmc_ clk 0x0300_1050
	SPINOR_WP_X	I/O	EMMC_DAT3	1	fmux_gpio_reg_ioctrl_emmc_ dat3 0x0300_1058

I2C	IIC1_SCL	I/O	PAD_MIPIRX 4P	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX4P 0x0300_1170
	IIC1_SCL	I/O	PAD_MIPIRX 0N	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX0N 0x0300_118C
	IIC1_SCL	I/O	SD0_CMD	1	FMUX_GPIO_REG_IOCTRL_SD0_C MD 0x0300_1020
	IIC1_SCL	I/O	SD0_D2	1	fmux_gpio_reg_ioctrl_sd0_d 2 0x0300_102C
	IIC1_SCL	I/O	PAD_MIPI_T XP2	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPI_TXP2 0x0300_11A8
	IIC1_SCL	I/O	SD1_D3	2	fmux_gpio_reg_ioctrl_sd1_d 3 0x0300_10D0
	IIC1_SCL	I/O	MUX_SPI1_M OSI	2	fmux_gpio_reg_ioctrl_mux_s pi1_mosi 0x0300_1118
	IIC1_SCL	I/O	PAD_ETH_TX P EPHY_R XN	2	FMUX_GPIO_REG_IOCTRL_PAD_E TH_TXP 0x0300_1124
	IIC1_SCL	I/O	SD1_D2	1	fmux_gpio_reg_ioctrl_sd1_d 2 0x0300_10D4
	IIC1_SDA	I/O	PAD_MIPIRX 4N	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX4N 0x0300_116C
	IIC1_SDA	I/O	PAD_MIPIRX 1P	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX1P 0x0300_1188
	IIC1_SDA	I/O	SD0_CLK	1	fmux_gpio_reg_ioctrl_sd0_c lk 0x0300_101C
	IIC1_SDA	I/O	SD0_D1	1	fmux_gpio_reg_ioctrl_sd0_d 1 0x0300_1028
	IIC1_SDA	I/O	PAD_MIPI_T XM2	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPI_TXM2 0x0300_11A4
	IIC1_SDA	I/O	SD1_D0	2	fmux_gpio_reg_ioctrl_sd1_d 0 0x0300_10DC
	IIC1_SDA	I/O	MUX_SPI1_M ISO	2	FMUX_GPIO_REG_IOCTRL_MUX_S PI1_MISO 0x0300_1114
	IIC1_SDA	I/O	PAD_ETH_TX M EPHY_R XP	2	FMUX_GPIO_REG_IOCTRL_PAD_E TH_TXM 0x0300_1128
	IIC1_SDA	I/O	SD1_D1	1	fmux_gpio_reg_ioctrl_sd1_d 1 0x0300_10D8
	IIC2_SCL	I/O	PAD_MIPI_T XP1	4	FMUX_GPIO_REG_IOCTRL_PAD_M IPI_TXP1 0x0300_11B0
	IIC2_SCL	I/O	PWR_GPIO1	6	FMUX_GPIO_REG_IOCTRL_PWR_G PIO1 0x0300_10A8

	IIC2_SDA	I/O	PAD_MIPI_TXM1	4	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM1 0x0300_11AC
	IIC2_SDA	I/O	PWR_GPIO2	6	FMUX_GPIO_REG_IOCTRL_PWR_GPIO2 0x0300_10AC
	IIC3_SCL	I/O	SD1_CMD	2	FMUX_GPIO_REG_IOCTRL_SD1_CMD 0x0300_10E0
	IIC3_SDA	I/O	SD1_CLK	2	fmux_gpio_reg_ioctrl_sd1_c lk 0x0300_10E4
	IIC4_SCL	I/O	PWR_WAKEUP0	5	fmux_gpio_reg_ioctrl_pwr_wakeup0 0x0300_1090
	IIC4_SCL	I/O	PAD_MIPIRX2N	5	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2N 0x0300_117C
	IIC4_SDA	I/O	PWR_BUTTON1	5	FMUX_GPIO_REG_IOCTRL_PWR_BUTTON1 0x0300_1098
	IIC4_SDA	I/O	PAD_MIPIRX2P	5	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2P 0x0300_1180
no die I2C	PWR_IIC_SCL	I/O	PWR_GPIO1	5	FMUX_GPIO_REG_IOCTRL_PWR_GPIO1 0x0300_10A8
	PWR_IIC_SDA	I/O	PWR_GPIO2	5	FMUX_GPIO_REG_IOCTRL_PWR_GPIO2 0x0300_10AC
IIS	IIS1_BCLK	I/O	PAD_AUD_AINL_MIC	4	FMUX_GPIO_REG_IOCTRL_PAD_AUD_AINL_MIC 0x0300_11BC
	IIS1_DI	I	PAD_AUD_AOUTR	4	FMUX_GPIO_REG_IOCTRL_PAD_AUD_AOUTR 0x0300_11C8
	IIS1_DO	O	PAD_AUD_AOUTR	6	FMUX_GPIO_REG_IOCTRL_PAD_AUD_AOUTR 0x0300_11C8
	IIS1_MCLK	I/O	AUX0	4	FMUX_GPIO_REG_IOCTRL_AUX0 0x0300_1078
	IIS2_BCLK	I/O	PAD_AUD_AINL_MIC	5	FMUX_GPIO_REG_IOCTRL_PAD_AUD_AINL_MIC 0x0300_11BC
	IIS2_BCLK	I/O	PAD_ETH_TXM EPHY_R XP	7	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXM 0x0300_1128
	IIS2_DI	I	PAD_ETH_RXM EPHY_TXP	7	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXM 0x0300_1130
	IIS2_DO	O	PAD_AUD_AOUTR	5	FMUX_GPIO_REG_IOCTRL_PAD_AUD_AOUTR 0x0300_11C8
	IIS2_DO	O	PAD_ETH_RXP EPHY_TXN	7	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXP 0x0300_112C
	IIS2_LRCK	I/O	PAD_ETH_TXP EPHY_R XN	7	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXP 0x0300_1124

	KEY_COL0	I/O	MUX_SPI1_MOSI	5	fmux_gpio_reg_ioctrl_mux_spi1_mosi 0x0300_1118
	KEY_COL1	I/O	MUX_SPI1_MISO	5	FMUX_GPIO_REG_IOCTRL_MUX_SPI1_MISO 0x0300_1114
	KEY_COL2	I/O	ADC1	4	fmux_gpio_reg_ioctrl_adc1 0x0300_10f8
	KEY_ROW0	I/O	PAD_MIPIRX4N	6	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4N 0x0300_116C
	KEY_ROW1	I/O	PAD_MIPIRX4P	6	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4P 0x0300_1170
	KEY_ROW2	I/O	PAD_MIPIRX1P	6	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1P 0x0300_1188
	KEY_ROW2	I/O	MUX_SPI1_SCK	5	fmux_gpio_reg_ioctrl_mux_spi1_sck 0x0300_1120
	KEY_ROW3	I/O	PAD_MIPIRX1N	6	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1N 0x0300_1184
	KEY_ROW3	I/O	MUX_SPI1_CS	5	fmux_gpio_reg_ioctrl_mux_spi1_cs 0x0300_111C
PWM	PWM[0]	I/O	PWM0_BUCK	0	FMUX_GPIO_REG_IOCTRL_PWM0_BUCK 0x0300_10EC
	PWM[1]	I/O	GPIO_RTX_EPHY_RTX	4	FMUX_GPIO_REG_IOCTRL_GPIO_RTX 0x0300_11CC
	PWM[2]	I/O	GPIO_ZQ_PAD_ZQ	4	FMUX_GPIO_REG_IOCTRL_GPIO_ZQ 0x0300_11D0
	PWM[4]	I/O	UART0_TX	2	FMUX_GPIO_REG_IOCTRL_UART0_TX 0x0300_1040
	PWM[4]	I/O	SD1_D3	7	fmux_gpio_reg_ioctrl_sd1_d 3 0x0300_10D0
	PWM[5]	I/O	UART0_RX	2	FMUX_GPIO_REG_IOCTRL_UART0_RX 0x0300_1044
	PWM[5]	I/O	SD1_D2	7	fmux_gpio_reg_ioctrl_sd1_d 2 0x0300_10D4
	PWM[6]	I/O	JTAG_CPU_TCK	2	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TCK 0x0300_1068
	PWM[6]	I/O	SD1_D1	7	fmux_gpio_reg_ioctrl_sd1_d 1 0x0300_10D8
	PWM[7]	I/O	JTAG_CPU_TMS	2	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TMS 0x0300_1064
	PWM[7]	I/O	SD1_D0	7	fmux_gpio_reg_ioctrl_sd1_d 0 0x0300_10DC

	PWM[8]	I/O	PWR_GPIO0	4	FMUX_GPIO_REG_IOCTRL_PWR_GPIO0 0x0300_10A4
	PWM[8]	I/O	MUX_SPI1_MOSI	4	fmux_gpio_reg_ioctrl_mux_spi1_mosi 0x0300_1118
	PWM[8]	I/O	SD1_CMD	7	FMUX_GPIO_REG_IOCTRL_SD1_CMD 0x0300_10E0
	PWM[8]	I/O	PAD_MIPI_TXM2	5	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM2 0x0300_11A4
	PWM[9]	I/O	PWR_GPIO1	4	FMUX_GPIO_REG_IOCTRL_PWR_GPIO1 0x0300_10A8
	PWM[9]	I/O	MUX_SPI1_MISO	4	FMUX_GPIO_REG_IOCTRL_MUX_SPI1_MISO 0x0300_1114
	PWM[9]	I/O	SD1_CLK	7	fmux_gpio_reg_ioctrl_sd1_c lk 0x0300_10E4
	PWM[9]	I/O	PAD_MIPI_TXP2	5	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP2 0x0300_11A8
	PWM[10]	I/O	PWR_GPIO2	4	FMUX_GPIO_REG_IOCTRL_PWR_GPIO2 0x0300_10AC
	PWM[10]	I/O	MUX_SPI1_SCK	4	fmux_gpio_reg_ioctrl_mux_spi1_sck 0x0300_1120
	PWM[10]	I/O	SD0_D3	5	fmux_gpio_reg_ioctrl_sd0_d 3 0x0300_1030
	PWM[10]	I/O	PAD_MIPI_TXM1	5	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM1 0x0300_11AC
	PWM[11]	I/O	MUX_SPI1_CS	4	fmux_gpio_reg_ioctrl_mux_spi1_cs 0x0300_111C
	PWM[11]	I/O	SD0_D2	5	fmux_gpio_reg_ioctrl_sd0_d 2 0x0300_102C
	PWM[11]	I/O	PAD_MIPI_TXP1	5	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP1 0x0300_11B0
	PWM[12]	I/O	PAD_ETH_TXM EPHY_RXP	4	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXM 0x0300_1128
	PWM[12]	I/O	SD0_D1	5	fmux_gpio_reg_ioctrl_sd0_d 1 0x0300_1028
	PWM[13]	I/O	PAD_ETH_TXP EPHY_RXN	4	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXP 0x0300_1124
	PWM[13]	I/O	SD0_D0	5	fmux_gpio_reg_ioctrl_sd0_d 0 0x0300_1024
	PWM[14]	I/O	PAD_ETH_RXM EPHY_TXP	4	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXM 0x0300_1130

	PWM[14]	I/O	SD0_CMD	5	FMUX_GPIO_REG_IOCTRL_SD0_CMD 0x0300_1020
	PWM[14]	I/O	PAD_MIPI_TXM0	5	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM0 0x0300_11B4
	PWM[15]	I/O	PAD_ETH_RXP EPHY_TXN	4	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXP 0x0300_112C
	PWM[15]	I/O	SD0_CLK	5	fmux_gpio_reg_ioctrl_sd0_c lk 0x0300_101C
	PWM[15]	I/O	PAD_MIPI_TXP0	5	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP0 0x0300_11B8
CA53 JTAG (2W) RISCV JTAG (4W) I2C0	CR_4WTCK	I/O	JTAG_CPU_TCK	0	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TCK 0x0300_1068
	CR_4WTMS	I/O	JTAG_CPU_TMS	0	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TMS 0x0300_1064
	CR_2WTCK	I/O	PAD_MIPIRX3P	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3P 0x0300_1178
	CR_2WTCK	I/O	PAD_MIPI_TXP1	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP1 0x0300_11B0
	CR_2WTCK	I/O	PWR_GPIO2	7	FMUX_GPIO_REG_IOCTRL_PWR_GPIO2 0x0300_10AC
	CR_SCL0	I/O	PAD_MIPIRX4N	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4N 0x0300_116C
	CR_SCL0	I/O	PAD_MIPI_TXP2	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP2 0x0300_11A8
	CR_2WTMS	I/O	PAD_MIPIRX3N	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3N 0x0300_1174
	CR_2WTMS	I/O	PAD_MIPI_TXM1	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM1 0x0300_11AC
	CR_SDA0	I/O	PAD_MIPIRX4P	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4P 0x0300_1170
	CR_SDA0	I/O	PAD_MIPI_TXM2	0	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM2 0x0300_11A4
	CR_SDA0	I/O	PWR_GPIO1	7	FMUX_GPIO_REG_IOCTRL_PWR_GPIO1 0x0300_10A8
	CR_4WTDI	I/O	IIC0_SCL	0	FMUX_GPIO_REG_IOCTRL_IIC0_SCL 0x0300_1070
	CR_4WTDO	I/O	IIC0_SDA	0	FMUX_GPIO_REG_IOCTRL_IIC0_SDA 0x0300_1074
System	PWR_BUTTON1	I	PWR_BUTTON1	0	FMUX_GPIO_REG_IOCTRL_PWR_BUTTON1 0x0300_1098

	PWR_RSTN	I	PWR_RSTN	0	fmux_gpio_reg_ioctrl_pwr_rstn 0x0300_1080
	PWR_SEQ1	O	PWR_SEQ1	0	FMUX_GPIO_REG_IOCTRL_PWR_SEQ1 0x0300_1084
	PWR_SEQ2	O	PWR_SEQ2	0	FMUX_GPIO_REG_IOCTRL_PWR_SEQ2 0x0300_1088
	PWR_WAKEUP0	I	PWR_WAKEUP0	0	fmux_gpio_reg_ioctrl_pwr_wakeup0 0x0300_1090
	USB_VBUS_DET	I	USB_VBUS_DET	0	fmux_gpio_reg_ioctrl_usb_vbus_det 0x0300_1108
no die IR	PWR_IR0	I	PWR_WAKEUP0	1	fmux_gpio_reg_ioctrl_pwr_wakeup0 0x0300_1090
SPI_NOR1	PWR_SPINOR1_CS_X	O	SD1_D3	6	fmux_gpio_reg_ioctrl_sd1_d 3 0x0300_10D0
	PWR_SPINOR1_HOLD_X	I/O	SD1_D2	6	fmux_gpio_reg_ioctrl_sd1_d 2 0x0300_10D4
	PWR_SPINOR1_MISO	I/O	SD1_D0	6	fmux_gpio_reg_ioctrl_sd1_d 0 0x0300_10DC
	PWR_SPINOR1_MOSI	I/O	SD1_CMD	6	FMUX_GPIO_REG_IOCTRL_SD1_CMD 0x0300_10E0
	PWR_SPINOR1_SCK	O	SD1_CLK	6	fmux_gpio_reg_ioctrl_sd1_c lk 0x0300_10E4
	PWR_SPINOR1_WP_X	I/O	SD1_D1	6	fmux_gpio_reg_ioctrl_sd1_d 1 0x0300_10D8
SD1	PWR_SD1_CLK	O	SD1_CLK	0	fmux_gpio_reg_ioctrl_sd1_c lk 0x0300_10E4
	PWR_SD1_CMD	I/O	SD1_CMD	0	FMUX_GPIO_REG_IOCTRL_SD1_CMD 0x0300_10E0
	PWR_SD1_D0	I/O	SD1_D0	0	fmux_gpio_reg_ioctrl_sd1_d 0 0x0300_10DC
	PWR_SD1_D1	I/O	SD1_D1	0	fmux_gpio_reg_ioctrl_sd1_d 1 0x0300_10D8
	PWR_SD1_D2	I/O	SD1_D2	0	fmux_gpio_reg_ioctrl_sd1_d 2 0x0300_10D4
	PWR_SD1_D3	I/O	SD1_D3	0	fmux_gpio_reg_ioctrl_sd1_d 3 0x0300_10D0
SD0	SDIO0_CD	I	SD0_CD	0	fmux_gpio_reg_ioctrl_sd0_c d 0x0300_1034
	SDIO0_CLK	o	SD0_CLK	0	fmux_gpio_reg_ioctrl_sd0_c lk 0x0300_101C

	SDIO0_CMD	I/O	SD0_CMD	0	FMUX_GPIO_REG_IOCTRL_SD0_CMD 0x0300_1020
	SDIO0_D[0]	I/O	SD0_D0	0	fmux_gpio_reg_ioctrl_sd0_d 0 0x0300_1024
	SDIO0_D[1]	I/O	SD0_D1	0	fmux_gpio_reg_ioctrl_sd0_d 1 0x0300_1028
	SDIO0_D[2]	I/O	SD0_D2	0	fmux_gpio_reg_ioctrl_sd0_d 2 0x0300_102C
	SDIO0_D[3]	I/O	SD0_D3	0	fmux_gpio_reg_ioctrl_sd0_d 3 0x0300_1030
	SDIO0_PWR_EN	O	SD0_PWR_EN	0	fmux_gpio_reg_ioctrl_sd0_pwr_en 0x0300_1038
SPI	SPI0_CS_X	O	SD0_D3	2	fmux_gpio_reg_ioctrl_sd0_d 3 0x0300_1030
	SPI0_CS_X	O	PAD_MIPI_TXP2	6	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP2 0x0300_11A8
	SPI0_SCK	O	SD0_CLK	2	fmux_gpio_reg_ioctrl_sd0_c lk 0x0300_101C
	SPI0_SCK	O	PAD_MIPI_TXM2	6	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM2 0x0300_11A4
	SPI0_SDI	I	SD0_D0	2	fmux_gpio_reg_ioctrl_sd0_d 0 0x0300_1024
	SPI0_SDI	I	PAD_MIPI_TXP1	6	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP1 0x0300_11B0
	SPI0_SDO	I/O	SD0_CMD	2	FMUX_GPIO_REG_IOCTRL_SD0_CMD 0x0300_1020
	SPI0_SDO	I/O	PAD_MIPI_TXM1	6	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM1 0x0300_11AC
	SPI1_CS_X	O	MUX_SPI1_CS	6	fmux_gpio_reg_ioctrl_mux_spi1_cs 0x0300_111C
	SPI1_CS_X	O	PAD_ETH_RXM EPHY_TXP	6	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXM 0x0300_1130
	SPI1_SCK	O	MUX_SPI1_SCK	6	fmux_gpio_reg_ioctrl_mux_spi1_sck 0x0300_1120
	SPI1_SCK	O	PAD_ETH_RXP EPHY_TXN	6	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXP 0x0300_112C
	SPI1_SDI	I	MUX_SPI1_MISO	6	FMUX_GPIO_REG_IOCTRL_MUX_SPI1_MISO 0x0300_1114
	SPI1_SDI	I	PAD_ETH_TXM EPHY_RXP	6	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXM 0x0300_1128

	SPI1_SDO	I/O	MUX_SPI1_MOSI	6	fmux_gpio_reg_ioctrl_mux_spi1_mosi 0x0300_1118
	SPI1_SDO	I/O	PAD_ETH_TXP EPHY_RXN	6	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXP 0x0300_1124
	SPI2_CS_X	O	SD1_D3	1	fmux_gpio_reg_ioctrl_sd1_d 3 0x0300_10D0
	SPI2_SCK	O	SD1_CLK	1	fmux_gpio_reg_ioctrl_sd1_c lk 0x0300_10E4
	SPI2_SDI	I	SD1_D0	1	fmux_gpio_reg_ioctrl_sd1_d 0 0x0300_10DC
	SPI2_SDO	I/O	SD1_CMD	1	FMUX_GPIO_REG_IOCTRL_SD1_CMD 0x0300_10E0
UART	UART0_RX	I	UART0_RX	0	FMUX_GPIO_REG_IOCTRL_UART0_RX 0x0300_1044
	UART0_TX	O	UART0_TX	0	FMUX_GPIO_REG_IOCTRL_UART0_TX 0x0300_1040
	UART1_CTS	I	JTAG_CPU_TCK	4	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TCK 0x0300_1068
	UART1_RTS	O	JTAG_CPU_TMS	4	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TMS 0x0300_1064
	UART1_RX	I	SD0_D2	4	fmux_gpio_reg_ioctrl_sd0_d 2 0x0300_102C
	UART1_RX	I	UART0_RX	4	FMUX_GPIO_REG_IOCTRL_UART0_RX 0x0300_1044
	UART1_RX	I	PWR_BUTTON1	4	FMUX_GPIO_REG_IOCTRL_PWR_BUTTON1 0x0300_1098
	UART1_RX	I	JTAG_CPU_TCK	6	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TCK 0x0300_1068
	UART1_RX	I	IIC0_SDA	1	FMUX_GPIO_REG_IOCTRL_IIC0_SDA 0x0300_1074
	UART1_TX	O	SD0_D1	4	fmux_gpio_reg_ioctrl_sd0_d 1 0x0300_1028
	UART1_TX	o	UART0_TX	4	FMUX_GPIO_REG_IOCTRL_UART0_TX 0x0300_1040
	UART1_TX	O	PWR_WAKEUP0	4	fmux_gpio_reg_ioctrl_pwr_wakeup0 0x0300_1090
	UART1_TX	O	JTAG_CPU_TMS	6	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TMS 0x0300_1064
	UART1_TX	O	IIC0_SCL	1	FMUX_GPIO_REG_IOCTRL_IIC0_SCL 0x0300_1070

	UART2_RX	I	IIC0_SDA	2	FMUX_GPIO_REG_IOCTRL_IIC0_SDA 0x0300_1074
	UART2_RX	I	SD1_D1	2	fmux_gpio_reg_ioctrl_sd1_d 1 0x0300_10D8
	UART2_RX	I	PWR_GPIO1	1	FMUX_GPIO_REG_IOCTRL_PWR_GPIO1 0x0300_10A8
	UART2_TX	O	IIC0_SCL	2	FMUX_GPIO_REG_IOCTRL_IIC0_SCL 0x0300_1070
	UART2_TX	O	SD1_D2	2	fmux_gpio_reg_ioctrl_sd1_d 2 0x0300_10D4
	UART2_TX	O	PWR_GPIO0	1	FMUX_GPIO_REG_IOCTRL_PWR_GPIO0 0x0300_10A4
	UART3_CTS	I	MUX_SPI1_CS	1	fmux_gpio_reg_ioctrl_mux_spi1_cs 0x0300_111C
	UART3_CTS	I	PAD_ETH_RXM EPHY_TXP	1	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXM 0x0300_1130
	UART3_CTS	I	SD1_D3	5	fmux_gpio_reg_ioctrl_sd1_d 3 0x0300_10D0
	UART3_RTS	O	MUX_SPI1_MISO	1	FMUX_GPIO_REG_IOCTRL_MUX_SPI1_MISO 0x0300_1114
	UART3_RTS	O	PAD_ETH_TXM EPHY_RXP	1	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXM 0x0300_1128
	UART3_RTS	O	SD1_D0	5	fmux_gpio_reg_ioctrl_sd1_d 0 0x0300_10DC
	UART3_RX	I	SD0_D3	4	fmux_gpio_reg_ioctrl_sd0_d 3 0x0300_1030
	UART3_RX	I	MUX_SPI1_MOSI	1	fmux_gpio_reg_ioctrl_mux_spi1_mosi 0x0300_1118
	UART3_RX	I	PAD_ETH_TXP EPHY_RXN	1	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXP 0x0300_1124
	UART3_RX	I	SD1_D1	5	fmux_gpio_reg_ioctrl_sd1_d 1 0x0300_10D8
	UART3_TX	O	SD0_D0	4	fmux_gpio_reg_ioctrl_sd0_d 0 0x0300_1024
	UART3_TX	O	MUX_SPI1_SCK	1	fmux_gpio_reg_ioctrl_mux_spi1_sck 0x0300_1120
	UART3_TX	O	PAD_ETH_RXP EPHY_TXN	1	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXP 0x0300_112C
	UART3_TX	O	SD1_D2	5	fmux_gpio_reg_ioctrl_sd1_d 2 0x0300_10D4

no die UART	PWR_UART0_RX	I	PWR_GPIO0	2	FMUX_GPIO_REG_IOCTRL_PWR_GPIO0 0x0300_10A4
	PWR_UART0_TX	O	PWR_WAKEUP0	2	fmux_gpio_reg_ioctrl_pwr_wakeup0 0x0300_1090
Wiegand	WG0_D0	I/O	SD0_D0	6	fmux_gpio_reg_ioctrl_sd0_d 0 0x0300_1024
	WG0_D0	I/O	IIC0_SCL	5	FMUX_GPIO_REG_IOCTRL_IIC0_SCL 0x0300_1070
	WG0_D1	I/O	SD0_D1	6	fmux_gpio_reg_ioctrl_sd0_d 1 0x0300_1028
	WG0_D1	I/O	IIC0_SDA	5	FMUX_GPIO_REG_IOCTRL_IIC0_SDA 0x0300_1074
	WG1_D0	I/O	SD0_D2	6	fmux_gpio_reg_ioctrl_sd0_d 2 0x0300_102C
	WG1_D0	I/O	IIC0_SDA	6	FMUX_GPIO_REG_IOCTRL_IIC0_SDA 0x0300_1074
	WG1_D1	I/O	SD0_D3	6	fmux_gpio_reg_ioctrl_sd0_d 3 0x0300_1030
	WG1_D1	I/O	AUX0	6	FMUX_GPIO_REG_IOCTRL_AUX0 0x0300_1078
	WG2_D0	I/O	PWR_WAKEUP0	7	fmux_gpio_reg_ioctrl_pwr_wakeup0 0x0300_1090
	WG2_D1	I/O	PWR_BUTTON1	7	FMUX_GPIO_REG_IOCTRL_PWR_BUTTON1 0x0300_1098
GPIO	XGPIOA[7]	I/O	SD0_CLK	3	fmux_gpio_reg_ioctrl_sd0_c lk 0x0300_101C
	XGPIOA[8]	I/O	SD0_CMD	3	FMUX_GPIO_REG_IOCTRL_SD0_CMD 0x0300_1020
	XGPIOA[9]	I/O	SD0_D0	3	fmux_gpio_reg_ioctrl_sd0_d 0 0x0300_1024
	XGPIOA[10]	I/O	SD0_D1	3	fmux_gpio_reg_ioctrl_sd0_d 1 0x0300_1028
	XGPIOA[11]	I/O	SD0_D2	3	fmux_gpio_reg_ioctrl_sd0_d 2 0x0300_102C
	XGPIOA[12]	I/O	SD0_D3	3	fmux_gpio_reg_ioctrl_sd0_d 3 0x0300_1030
	XGPIOA[13]	I/O	SD0_CD	3	fmux_gpio_reg_ioctrl_sd0_c d 0x0300_1034
	XGPIOA[14]	I/O	SD0_PWR_EN	3	fmux_gpio_reg_ioctrl_sd0_pwr_en 0x0300_1038

	XGPIOA[15]	I/O	SPK_EN	3	fmux_gpio_reg_ioctrl_spk_en 0x0300_103C
	XGPIOA[16]	I/O	UART0_TX	3	FMUX_GPIO_REG_IOCTRL_UART0_TX 0x0300_1040
	XGPIOA[17]	I/O	UART0_RX	3	FMUX_GPIO_REG_IOCTRL_UART0_RX 0x0300_1044
	XGPIOA[18]	I/O	JTAG_CPU_TCK	3	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TCK 0x0300_1068
	XGPIOA[19]	I/O	JTAG_CPU_TMS	3	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TMS 0x0300_1064
	XGPIOA[22]	I/O	EMMC_CLK	3	fmux_gpio_reg_ioctrl_emmc_ clk 0x0300_1050
	XGPIOA[23]	I/O	EMMC_CMD	3	fmux_gpio_reg_ioctrl_emmc_ cmd 0x0300_105C
	XGPIOA[24]	I/O	EMMC_DAT1	3	fmux_gpio_reg_ioctrl_emmc_ dat1 0x0300_1060
	XGPIOA[25]	I/O	EMMC_DAT0	3	fmux_gpio_reg_ioctrl_emmc_ dat0 0x0300_1054
	XGPIOA[26]	I/O	EMMC_DAT2	3	fmux_gpio_reg_ioctrl_emmc_ dat2 0x0300_104C
	XGPIOA[27]	I/O	EMMC_DAT3	3	fmux_gpio_reg_ioctrl_emmc_ dat3 0x0300_1058
	XGPIOA[28]	I/O	IIC0_SCL	3	FMUX_GPIO_REG_IOCTRL_IIC0_SCL 0x0300_1070
	XGPIOA[29]	I/O	IIC0_SDA	3	FMUX_GPIO_REG_IOCTRL_IIC0_SDA 0x0300_1074
	XGPIOA[30]	I/O	AUX0	3	FMUX_GPIO_REG_IOCTRL_AUX0 0x0300_1078
	XGPIOB[0]	I/O	PWM0_BUCK	3	FMUX_GPIO_REG_IOCTRL_PWM0_BUCK 0x0300_10EC
	XGPIOB[10]	I/O	MUX_SPI1_CS	3	fmux_gpio_reg_ioctrl_mux_spi1_cs 0x0300_111C
	XGPIOB[23]	I/O	GPIO_RTX_EPHY_RTX	3	FMUX_GPIO_REG_IOCTRL_GPIO_RTX 0x0300_11CC
	XGPIOB[24]	I/O	PAD_ETH_TXM EPHY_RXP	3	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXM 0x0300_1128
	XGPIOB[25]	I/O	PAD_ETH_TXP EPHY_RXN	3	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXP 0x0300_1124
	XGPIOB[26]	I/O	PAD_ETH_RXM EPHY_TXP	3	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXM 0x0300_1130

	XGPIOB[27]	I/O	PAD_ETH_RXP EPHY_T XN	3	FMUX_GPIO_REG_IOCTRL_PAD_ETH_RXP 0x0300_112C
	XGPIOB[3]	I/O	ADC1	3	fmux_gpio_reg_ioctrl_adc1 0x0300_10f8
	XGPIOB[6]	I/O	USB_VBUS_DET	3	fmux_gpio_reg_ioctrl_usb_vbus_det 0x0300_1108
	XGPIOB[7]	I/O	MUX_SPI1_MOSI	3	fmux_gpio_reg_ioctrl_mux_spi1_mosi 0x0300_1118
	XGPIOB[8]	I/O	MUX_SPI1_MISO	3	FMUX_GPIO_REG_IOCTRL_MUX_SPI1_MISO 0x0300_1114
	XGPIOB[9]	I/O	MUX_SPI1_SCK	3	fmux_gpio_reg_ioctrl_mux_spi1_sck 0x0300_1120
	XGPIOC [2]	I/O	PAD_MIPIRX4N	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4N 0x0300_116C
	XGPIOC [3]	I/O	PAD_MIPIRX4P	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX4P 0x0300_1170
	XGPIOC [4]	I/O	PAD_MIPIRX3N	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3N 0x0300_1174
	XGPIOC [5]	I/O	PAD_MIPIRX3P	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3P 0x0300_1178
	XGPIOC [6]	I/O	PAD_MIPIRX2N	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2N 0x0300_117C
	XGPIOC [7]	I/O	PAD_MIPIRX2P	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2P 0x0300_1180
	XGPIOC [8]	I/O	PAD_MIPIRX1N	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1N 0x0300_1184
	XGPIOC [9]	I/O	PAD_MIPIRX1P	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1P 0x0300_1188
	XGPIOC [10]	I/O	PAD_MIPIRX0N	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0N 0x0300_118C
	XGPIOC [11]	I/O	PAD_MIPIRX0P	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0P 0x0300_1190
	XGPIOC [12]	I/O	PAD_MIPI_TXM0	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM0 0x0300_11B4
	XGPIOC [13]	I/O	PAD_MIPI_TXP0	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP0 0x0300_11B8
	XGPIOC [14]	I/O	PAD_MIPI_TXM1	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM1 0x0300_11AC
	XGPIOC [15]	I/O	PAD_MIPI_TXP1	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP1 0x0300_11B0

	XGPIOC [16]	I/O	PAD_MIPI_TXM2	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM2 0x0300_11A4
	XGPIOC [17]	I/O	PAD_MIPI_TXP2	3	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP2 0x0300_11A8
	XGPIOC[23]	I/O	PAD_AUD_AINL_MIC	3	FMUX_GPIO_REG_IOCTRL_PAD_AUD_AINL_MIC 0x0300_11BC
	XGPIOC [24]	I/O	PAD_AUD_AOUTR	3	FMUX_GPIO_REG_IOCTRL_PAD_AUD_AOUTR 0x0300_11C8
No die domain GPIO	PWR_GPIO[0]	I/O	PWR_GPIO0	0	FMUX_GPIO_REG_IOCTRL_PWR_GPIO0 0x0300_10A4
	PWR_GPIO[3]	I/O	PWR_SEQ1	3	FMUX_GPIO_REG_IOCTRL_PWR_SEQ1 0x0300_1084
	PWR_GPIO[4]	I/O	PWR_SEQ2	3	FMUX_GPIO_REG_IOCTRL_PWR_SEQ2 0x0300_1088
	PWR_GPIO[6](	I/O	PWR_WAKEUP0	3	fmux_gpio_reg_ioctrl_pwr_wakeup0 0x0300_1090
	PWR_GPIO[8]	I/O	PWR_BUTTON1	3	FMUX_GPIO_REG_IOCTRL_PWR_BUTTON1 0x0300_1098
	PWR_GPIO[18]	I/O	SD1_D3	3	fmux_gpio_reg_ioctrl_sd1_d 3 0x0300_10D0
	PWR_GPIO[19]	I/O	SD1_D2	3	fmux_gpio_reg_ioctrl_sd1_d 2 0x0300_10D4
	PWR_GPIO[20]	I/O	SD1_D1	3	fmux_gpio_reg_ioctrl_sd1_d 1 0x0300_10D8
	PWR_GPIO[21]	I/O	SD1_D0	3	fmux_gpio_reg_ioctrl_sd1_d 0 0x0300_10DC
	PWR_GPIO[22]	I/O	SD1_CMD	3	FMUX_GPIO_REG_IOCTRL_SD1_CMD 0x0300_10E0
	PWR_GPIO[23]	I/O	SD1_CLK	3	fmux_gpio_reg_ioctrl_sd1_c lk 0x0300_10E4
	PWR_GPIO[24]	I/O	GPIO_ZQ_____PAD_ZQ	3	FMUX_GPIO_REG_IOCTRL_GPIO_ZQ 0x0300_11D0
Debug	DBG[0]	O	SD0_CLK	7	fmux_gpio_reg_ioctrl_sd0_c lk 0x0300_101C
	DBG[1]	O	SD0_CMD	7	FMUX_GPIO_REG_IOCTRL_SD0_CMD 0x0300_1020
	DBG[10]	O	IIC0_SCL	7	FMUX_GPIO_REG_IOCTRL_IIC0_SCL 0x0300_1070
	DBG[10]	o	PAD_MIPIRX0N	7	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0N 0x0300_118C

	DBG[11]	o	IIC0_SDA	7	FMUX_GPIO_REG_IOCTRL_IIC0_SDA 0x0300_1074
	DBG[11]	o	PAD_MIPIRX0P	7	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0P 0x0300_1190
	DBG[12]	O	AUX0	7	FMUX_GPIO_REG_IOCTRL_AUX0 0x0300_1078
	DBG[12]	O	PAD_MIPI_TXM0	7	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM0 0x0300_11B4
	DBG[13]	O	MUX_SPI1_MOSI	7	fmux_gpio_reg_ioctrl_mux_spi1_mosi 0x0300_1118
	DBG[13]	o	PAD_MIPI_TXP0	7	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP0 0x0300_11B8
	DBG[14]	O	MUX_SPI1_MISO	7	FMUX_GPIO_REG_IOCTRL_MUX_SPI1_MISO 0x0300_1114
	DBG[14]	O	PAD_MIPI_TXM1	7	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM1 0x0300_11AC
	DBG[15]	O	MUX_SPI1_SCK	7	fmux_gpio_reg_ioctrl_mux_spi1_sck 0x0300_1120
	DBG[15]	o	PAD_MIPI_TXP1	7	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP1 0x0300_11B0
	DBG[16]	O	MUX_SPI1_CS	7	fmux_gpio_reg_ioctrl_mux_spi1_cs 0x0300_111C
	DBG[2]	O	SD0_D0	7	fmux_gpio_reg_ioctrl_sd0_d 0 0x0300_1024
	DBG[3]	O	SD0_D1	7	fmux_gpio_reg_ioctrl_sd0_d 1 0x0300_1028
	DBG[4]	O	SD0_D2	7	fmux_gpio_reg_ioctrl_sd0_d 2 0x0300_102C
	DBG[5]	O	SD0_D3	7	fmux_gpio_reg_ioctrl_sd0_d 3 0x0300_1030
	DBG[6]	O	UART0_TX	7	FMUX_GPIO_REG_IOCTRL_UART0_TX 0x0300_1040
	DBG[6]	O	PAD_MIPIRX2N	7	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2N 0x0300_117C
	DBG[7]	O	UART0_RX	7	FMUX_GPIO_REG_IOCTRL_UART0_RX 0x0300_1044
	DBG[7]	O	PAD_MIPIRX2P	7	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2P 0x0300_1180
	DBG[8]	O	PAD_MIPIRX1N	7	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1N 0x0300_1184

	DBG[9]	O	PAD_MIPIRX 1P	7	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX1P 0x0300_1188
Others	MUX_SPI1_C S	I/O	PAD_MIPIRX 4P	7	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX4P 0x0300_1170
	MUX_SPI1_M ISO	I/O	PAD_MIPIRX 3N	7	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX3N 0x0300_1174
	MUX_SPI1_M OSI	I/O	PAD_MIPIRX 3P	7	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX3P 0x0300_1178
	MUX_SPI1_S CK	I/O	PAD_MIPIRX 4N	7	FMUX_GPIO_REG_IOCTRL_PAD_M IPIRX4N 0x0300_116C
	PKG_TYPE0	I	PKG_TYPE0	0	FMUX_GPIO_REG_IOCTRL_PKG_T YPE0 0x0300_1104
	PKG_TYPE1	I	PKG_TYPE1	0	FMUX_GPIO_REG_IOCTRL_PKG_T YPE1 0x0300_110C
	PKG_TYPE2	I	PKG_TYPE2	0	FMUX_GPIO_REG_IOCTRL_PKG_T YPE2 0x0300_1110

8.1.2 FMUX Register Description

Table *8-2 FMUX* Register Configuration Information Table

Pin Num	Pin Name	Function_select_register	fmux_default	Description
6	SD0_CLK	fmux_gpio_reg_ioctrl_sd0_clk 0x0300_101c	0x0	IO SD0_CLK function select . 0 : SDIO0_CLK (default) 1 : IIC1_SDA 2 : SPI0_SCK 3 :: XGPIOA[7] 5 : PWM [15] 6 : EPHY_LNK_LED 7 : DBG[0] Others : Reserved
7	SD0_CMD	fmux_gpio_reg_ioctrl_sd0_cmd 0x0300_1020	0x0	IO SD0_CMD function select . 0 : SDIO0_CMD (default) 1 : IIC1_SCL 2 : SPI0_SDO 3 :: XGPIOA[8] 5 : PWM [14] 6 : EPHY_SPD_LED 7 : DBG[1] Others : Reserved

8	SD0_D0	fmux_gpio_reg_ioctrl_sd0_d0 0x0300_1024	0x0	IO SD0_D0 function select . 0 : SDIO0_D[0] (default) 1 : CAM_MCLK1 2 : SPI0_SDI 3 :: XGPIOA[9] 4 : UART3_TX 5 :: PWM [13] 6 : WG0_D0 7 : DBG[2] Others : Reserved
10	SD0_D1	fmux_gpio_reg_ioctrl_sd0_d1 0x0300_1028	0x0	IO SD0_D1 function select . 0 : SDIO0_D[1] (default) 1 : IIC1_SDA 2 : AUX0 3 :: XGPIOA[10] 4 : UART1_TX 5 : PWM[12] 6 : WG0_D1 7 : DBG[3] Others : Reserved
11	SD0_D2	fmux_gpio_reg_ioctrl_sd0_d2 0x0300_102C	0x0	IO SD0_D2 function select . 0 : SDIO0_D[2] (default) 1 : IIC1_SCL 2 : AUX1 3 :: XGPIOA[11] 4 : UART1_RX 5 : PWM[11] 6 : WG1_D0 7 : DBG[4] Others : Reserved
12	SD0_D3	fmux_gpio_reg_ioctrl_sd0_d3 0x0300_1030	0x0	IO SD0_D3 function select . 0 : SDIO0_D[3] (default) 1 : CAM_MCLK0 2 : SPI0_CS_X 3 :: XGPIOA[12] 4 : UART3_RX 5 : PWM[10] 6 : WG1_D1 7 : DBG[5] Others : Reserved
14	SD0_CD	fmux_gpio_reg_ioctrl_sd0_cd 0x0300_1034	0x0	IO SD0_CD function select . 0 : SDIO0_CD (default) 3 : XGPIOA[13] Others : Reserved

15	SD0_PWR_EN	fmux_gpio_reg_ioctrl_sd0_pwr_e n 0x0300_1038	0x3	IO SD0_PWR_EN function select : 0 : SDIO0_PWR_EN 3 : XGPIOA[14] (default) Others : Reserved
17	SPK_EN	fmux_gpio_reg_ioctrl_spk_en 0x0300_103c	0x3	IO SPK_EN function select . 3 : XGPIOA[15] (default) Others : Reserved
18	UART0_TX	fmux_gpio_reg_ioctrl_uart0_tx 0x0300_1040	0x0	IO UART0_TX function select . 0 : UART0_TX (default) 1 : CAM_MCLK1 2 : PWM[4] 3 :: XGPIOA[16] 4 : UART1_TX 5 : AUX1 7 : DBG[6] Others : Reserved
19	UART0_RX	FMUX_GPIO_REG_IOCTRL_UART0_RX 0x0300_1044	0x0	IO UART0_RX function select . 0 : UART0_RX (default) 1 : CAM_MCLK0 2 : PWM[5] 3 :: XGPIOA [17] 4 : UART1_RX 5 : AUX0 7 : DBG[7] Others : Reserved
20	EMMC_DAT2	FMUX_GPIO_REG_IOCTRL_EMMC_DAT2 0x0300_104C	0x1	IO EMMC_DAT2 function select . 0 : EMMC_DAT[2] 1 : SPINOR_HOLD_X (default) 2 : SPINAND_HOLD 3 :: XGPIOA[26] Others : Reserved
21	EMMC_CLK	FMUX_GPIO_REG_IOCTRL_EMMC_CLK 0x0300_1050	0x1	IO EMMC_CLK function select . 0 : EMMC_CLK 1 : SPINOR_SCK (default) 2 : SPINAND_CLK 3 :: XGPIOA[22] Others : Reserved
22	EMMC_DAT0	FMUX_GPIO_REG_IOCTRL_EMMC_DAT0 0x0300_1054	0x1	IO EMMC_DAT0 function select : 0 : EMMC_DAT[0] 1 : SPINOR_MOSI (default) 2 : SPINAND_MOSI 3 :: XGPIOA[25] Others : Reserved

23	EMMC_DAT3	FMUX_GPIO_REG_IOCTRL_EMMC_DAT3 0x0300_1058	0x1	IO EMMC_DAT3 function select . 0 : EMMC_DAT[3] 1 : SPINOR_WP_X (default) 2 : SPINAND_WP 3 :: XGPIOA[27] Others : Reserved
24	EMMC_CMD	fmux_gpio_reg_ioctrl_emmc_cmd 0x0300_105c	0x1	IO EMMC_CMD function select . 0 : EMMC_CMD 1 : SPINOR_MISO (default) 2 : SPINAND_MISO 3 :: XGPIOA[23] Others : Reserved
25	EMMC_DAT1	FMUX_GPIO_REG_IOCTRL_EMMC_DAT1 0x0300_1060	0x1	IO EMMC_DAT1 function select . 0 : EMMC_DAT[1] 1 : SPINOR_CS_X (default) 2 : SPINAND_CS 3 :: XGPIOA[24] Others : Reserved
26	JTAG_CPU_TMS	FMUX_GPIO_REG_IOCTRL_JTAG_CPU_TMS 0x0300_1064	0x0	IO JTAG_CPU_TMS function select : 0 : CR_4WTMS (default) 1 : CAM_MCLK0 2 : PWM[7] 3 :: XGPIOA[19] 4 : UART1_RTS 5 : AUX0 6 : UART1_TX 7 : VO_D[28] Others : Reserved
27	JTAG_CPU_TCK	fmux_gpio_reg_ioctrl_jtag_cpu_ tck 0x0300_1068	0x0	IO JTAG_CPU_TCK function select : 0 : CR_4WTCK (default) 1 : CAM_MCLK1 2 : PWM[6] 3 :: XGPIOA[18] 4 : UART1_CTS 5 : AUX1 6 : UART1_RX 7 : VO_D[29] Others : Reserved
28	IIC0_SCL	FMUX_GPIO_REG_IOCTRL_IIC0_SCL 0x0300_1070	0x0	IO IIC0_SCL function select . 0 : CR_4WTDI (default) 1 : UART1_TX 2 : UART2_TX 3 :: XGPIOA [28] 5 : WG0_D0 7 : DBG[10] Others : Reserved

29	IIC0_SDA	FMUX_GPIO_REG_IOCTRL_IIC0_SDA 0x0300_1074	0x0	IO IIC0_SDA function select . 0 : CR_4WTDO (default) 1 : UART1_RX 2 : UART2_RX 3 : XGPIOA[29] 5 : WG0_D1 6 : WG1_D0 7 : DBG[11] Others : Reserved
30	AUX0	FMUX_GPIO_REG_IOCTRL_AUX0 0x0300_1078	0x3	IO AUX0 function select . 0 : AUX0 3 : XGPIOA[30] (default) 4 : IIS1_MCLK 5 : VO_D[31] 6 : WG1_D1 7 : DBG[12] Others : Reserved
38	PWR_VBAT_DET	fmux_gpio_reg_ioctrl_pwr_vbr_vbat_det 0x0300_107C	0x0	IO PWR_VBAT_DET function select : 0 : PWR_VBAT_DET (default) Others : Reserved
39	PWR_RSTN	FMUX_GPIO_REG_IOCTRL_PWR_RSTN 0x0300_1080	0x0	IO PWR_RSTN function select : 0 : PWR_RSTN (default) Others : Reserved
40	PWR_SEQ1	FMUX_GPIO_REG_IOCTRL_PWR_SEQ1 0x0300_1084	0x0	IO PWR_SEQ1 function select : 0 : PWR_SEQ1 (default) 3 : PWR_GPIO[3] Others : Reserved
41	PWR_SEQ2	FMUX_GPIO_REG_IOCTRL_PWR_SEQ2 0x0300_1088	0x0	IO PWR_SEQ2 function select . 0 : PWR_SEQ2 (default) 3 : PWR_GPIO[4] Others : Reserved
43	PWR_WAKEUP0	fmux_gpio_reg_ioctrl_pwr_wakeu p0 0x0300_1090	0x0	IO PWR_WAKEUP0 function select : 0 : PWR_WAKEUP0 (default) 1 : PWR_IR0 2 : PWR_UART0_TX 3 : PWR_GPIO[6]. 4 : UART1_TX 5 : IIC4_SCL 6 : EPHY_LNK_LED 7 : WG2_D0 Others : Reserved

44	PWR_BUTTON1	FMUX_GPIO_REG_IOCTRL_PWR_BUTTON1 0x0300_1098	0x0	IO PWR_BUTTON1 function select : 0 : PWR_BUTTON1 (default) 3 : PWR_GPIO[8]. 4 : UART1_RX 5 : IIC4_SDA 6 : EPHY_SPD_LED 7 : WG2_D1 Others : Reserved
45	XTAL_XIN	FMUX_GPIO_REG_IOCTRL_XTAL_XIN 0x0300_10A0	0x0	IO XTAL_XIN function select . 0 : PWR_XTAL_CLKIN (default) Others : Reserved
47	PWR_GPIO0	FMUX_GPIO_REG_IOCTRL_PWR_GPIO0 0x0300_10A4	0x0	IO PWR_GPIO0 function select . 0 : PWR_GPIO[0] (default) 1 : UART2_TX 2 : PWR_UART0_RX 4 : PWM[8] Others : Reserved
48	PWR_GPIO1	FMUX_GPIO_REG_IOCTRL_PWR_GPIO1 0x0300_10A8	0x0	IO PWR_GPIO1 function select . 0 : PWR_GPIO[1] (default) 1 : UART2_RX 3 : EPHY_LNK_LED 4 : PWM[9] 5 : PWR_IIC_SCL 6 : IIC2_SCL 7 : CR_SDA0 Others : Reserved
49	PWR_GPIO2	FMUX_GPIO_REG_IOCTRL_PWR_GPIO2 0x0300_10AC	0x0	IO PWR_GPIO2 function select . 0 : PWR_GPIO[2] (default) 2 : PWR_SECTICK 3 : EPHY_SPD_LED 4 : PWM[10] 5 : PWR_IIC_SDA 6 : IIC2_SDA 7 : CR_2WTCK Others : Reserved
51	SD1_D3	fmux_gpio_reg_ioctrl_sd1_d3 0x0300_10d0	0x6	IO SD1_D3 function select . 0 : PWR_SD1_D3_VO32 1 : SPI2_CS_X 2 : IIC1_SCL 3 : PWR_GPIO[18]. 4 : CAM_MCLK0 5 : UART3_CTS 6 : PWR_SPINOR1_CS_X (default) 7 : PWM[4] Others : Reserved

52	SD1_D2	fmux_gpio_reg_ioctrl_sd1_d2 0x0300_10d4	0x6	IO SD1_D2 function select . 0 : PWR_SD1_D2_VO33 1 : IIC1_SCL 2 : UART2_TX 3 : PWR_GPIO[19]. 4 : CAM_MCLK0 5 : UART3_TX 6 : PWR_SPINOR1_HOLD_X (default) 7 : PWM[5] Others : Reserved
53	SD1_D1	fmux_gpio_reg_ioctrl_sd1_d1 0x0300_10d8	0x6	IO SD1_D1 function select . 0 : PWR_SD1_D1_VO34 1 : IIC1_SDA 2 : UART2_RX 3 : PWR_GPIO[20]. 4 : CAM_MCLK1 5 : UART3_RX 6 : PWR_SPINOR1_WP_X (default) 7 : PWM[6] Others : Reserved
54	SD1_D0	FMUX_GPIO_REG_IOCTRL_SD1_D0 0x0300_10DC	0x6	IO SD1_D0 function select . 0 : PWR_SD1_D0_VO35 1 : SPI2_SDI 2 : IIC1_SDA 3 : PWR_GPIO[21]. 4 : CAM_MCLK1 5 : UART3_RTS 6 : PWR_SPINOR1_MISO (default) 7 : PWM[7] Others : Reserved
55	SD1_CMD	fmux_gpio_reg_ioctrl_sd1_cmd 0x0300_10e0	0x6	IO SD1_CMD function select . 0 : PWR_SD1_CMD_VO36 1 : SPI2_SDO 2 : IIC3_SCL 3 : PWR_GPIO[22]. 4 : CAM_VS0 5 : EPHY_LNK_LED 6 : PWR_SPINOR1_MOSI (default) 7 : PWM[8] Others : Reserved

56	SD1_CLK	fmux_gpio_reg_ioctrl_sd1_clk 0x0300_10e4	0x6	IO SD1_CLK function select . 0 : PWR_SD1_CLK_VO37 1 : SPI2_SCK 2 : IIC3_SDA 3 : PWR_GPIO[23]. 4 : CAM_HS0 5 : EPHY_SPD_LED 6 : PWR_SPINOR1_SCK (default) 7 : PWM[9] Others : Reserved
58	PWM0_BUCK	FMUX_GPIO_REG_IOCTRL_PWM0_BUCK 0x0300_10EC	0x3	IO PWM0_BUCK function select . 0 : PWM[0] 3 : XGPIOB[0] (default) Others : Reserved
59	ADC1	fmux_gpio_reg_ioctrl_adc1 0x0300_10f8	0x3	IO ADC1 function select : 3 : XGPIOB[3] (default) 4 : KEY_COL2 Others : Reserved
60	USB_VBUS_DET	fmux_gpio_reg_ioctrl_usb_vbus_ det 0x0300_1108	0x0	IO USB_VBUS_DET function select : 0 : USB_VBUS_DET (default) 3 : XGPIOB[6] 4 : CAM_MCLK0 5 : CAM_MCLK1 Others : Reserved
62	PAD_ETH_TXP_____EPHY_RXN	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXP 0x0300_1124	0x3	IO PAD_ETH_TXP function select : 1 : UART3_RX 2 : IIC1_SCL 3 : XGPIOB[25] (default) 4 :: PWM [13] 5 : CAM_MCLK0 6 : SPI1_SDO 7 : IIS2_LRCK Others : Reserved
63	PAD_ETH_TXM____EPHY_RXP	FMUX_GPIO_REG_IOCTRL_PAD_ETH_TXM 0x0300_1128	0x3	IO PAD_ETH_TXM function select : 1 : UART3_RTS 2 : IIC1_SDA 3 : XGPIOB[24] (default) 4 : PWM[12] 5 : CAM_MCLK1 6 : SPI1_SDI 7 : IIS2_BCLK Others : Reserved

64	PAD_ETH_RXP_____ EPHY_TX N	fmux_gpio_reg_ioctrl_pad_eth_r xp 0x0300_112C	0x3	IO PAD_ETH_RXP function select : 1 : UART3_TX 2 : CAM_MCLK1 3 : XGPIOB[27] (default) 4 : PWM[15] 5 : CAM_HS0 6 : SPI1_SCK 7 : IIS2_DO Others : Reserved
65	PAD_ETH_RXM____ EPHY_TX P	FMUX_GPIO_REG_IOCTRL_PAD_ETH_R XM 0x0300_1130	0x3	IO PAD_ETH_RXM function select : 1 : UART3_CTS 2 : CAM_MCLK0 3 : XGPIOB[26] (default) 4 : PWM[14] 5 : CAM_VS0 6 : SPI1_CS_X 7 : IIS2_DI Others : Reserved
72	PAD_MIP IRX4N	FMUX_GPIO_REG_IOCTRL_PAD_MIPIR X4N 0x0300_116C	0x0	IO PAD_MIPIRX4N function select : 0 : CR_SCL0 (default) 1 : VI0_CLK 2 : VI1_D[13] 3 :: XGPIOC [2] 4 : IIC1_SDA 5 : CAM_MCLK0 6 : KEY_ROW0 7 : MUX_SPI1_SCK Others : Reserved
73	PAD_MIP IRX4P	FMUX_GPIO_REG_IOCTRL_PAD_MIPIR X4P 0x0300_1170	0x0	IO PAD_MIPIRX4P function select : 0 : CR_SDA0 (default) 1 : VI0_D[0] 2 : VI1_D[14] 3 :: XGPIOC [3] 4 : IIC1_SCL 5 : CAM_MCLK1 6 : KEY_ROW1 7 : MUX_SPI1_CS Others : Reserved
74	PAD_MIP IRX3N	FMUX_GPIO_REG_IOCTRL_PAD_MIPIR X3N 0x0300_1174	0x3	IO PAD_MIPIRX3N function select : 0 : CR_2WTMS 1 : VI0_D[1] 2 : VI1_D[15] 3 : XGPIOC[4] (default) 4 : CAM_MCLK0 7 : MUX_SPI1_MISO Others : Reserved

75	PAD_MIPIRX3P	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX3P 0x0300_1178	0x3	IO PAD_MIPIRX3P function select : 0 : CR_2WTCK 1 : VI0_D[2] 2 : VI1_D[16] 3 : XGPIOC[5] (default) 7 : MUX_SPI1_MOSI Others : Reserved
76	PAD_MIPIRX2N	fmux_gpio_reg_ioctrl_pad_mipir x2n 0x0300_117C	0x3	IO PAD_MIPIRX2N function select : 1 : VI0_D[3] 2 : VO_D[10] 3 : XGPIOC[6] (default) 4 : VI1_D[17] 5 : IIC4_SCL 7 : DBG[6] Others : Reserved
77	PAD_MIPIRX2P	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX2P 0x0300_1180	0x3	IO PAD_MIPIRX2P function select : 1 : VI0_D[4] 2 : VO_D[9] 3 : XGPIOC[7] (default) 4 : VI1_D[18] 5 : IIC4_SDA 7 : DBG[7] Others : Reserved
78	PAD_MIPIRX1N	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX1N 0x0300_1184	0x3	IO PAD_MIPIRX1N function select : 1 : VI0_D[5] 2 : VO_D[8] 3 : XGPIOC[8] (default) 6 : KEY_ROW3 7 : DBG[8] Others : Reserved
79	PAD_MIPIRX1P	fmux_gpio_reg_ioctrl_pad_mipir x1p 0x0300_1188	0x3	IO PAD_MIPIRX1P function select : 1 : VI0_D[6] 2 : VO_D[7] 3 : XGPIOC[9] (default) 4 : IIC1_SDA 6 : KEY_ROW2 7 : DBG[9] Others : Reserved
80	PAD_MIPIRX0N	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0N 0x0300_118C	0x3	IO PAD_MIPIRX0N function select : 1 : VI0_D[7] 2 : VO_D[6] 3 : XGPIOC[10] (default) 4 : IIC1_SCL 5 : CAM_MCLK1 7 : DBG[10] Others : Reserved

81	PAD_MIPIRX0P	FMUX_GPIO_REG_IOCTRL_PAD_MIPIRX0P 0x0300_1190	0x3	IO PAD_MIPIRX0P function select : 1 : VI0_D[8] 2 : VO_D[5] 3 : XGPIOC[11] (default) 4 : CAM_MCLK0 7 : DBG[11] Others : Reserved
83	PAD_MIPI_TXM2	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM2 0x0300_11A4	0x3	IO PAD_MIPI_TXM2 function select . 0 : CR_SDA0 1 : VI0_D[13] 2 : VO_D[0] 3 : XGPIOC[16] (default) 4 : IIC1_SDA 5 : PWM[8] 6 : SPI0_SCK 7 : SD1_D2 Others : Reserved
84	PAD_MIPI_TXP2	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP2 0x0300_11A8	0x3	IO PAD_MIPI_TXP2 function select . 0 : CR_SCL0 1 : VI0_D[14] 2 : VO_CLK0 3 : XGPIOC[17] (default) 4 : IIC1_SCL 5 : PWM[9] 6 : SPI0_CS_X 7 : SD1_D3 Others : Reserved
85	PAD_MIPI_TXM1	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM1 0x0300_11AC	0x3	IO PAD_MIPI_TXM1 function select : 0 : CR_2WTMS 1 : VI0_D[11] 2 : VO_D[2] 3 : XGPIOC[14] (default) 4 : IIC2_SDA 5 : PWM[10] 6 : SPI0_SDO 7 : DBG[14] Others : Reserved
86	PAD_MIPI_TXP1	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP1 0x0300_11B0	0x3	IO PAD_MIPI_TXP1 function select . 0 : CR_2WTCK 1 : VI0_D[12] 2 : VO_D[1] 3 : XGPIOC[15] (default) 4 : IIC2_SCL 5 : PWM[11] 6 : SPI0_SDI 7 : DBG[15] Others : Reserved

87	PAD_MIPI_TXM0	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXM0 0x0300_11B4	0x3	IO PAD_MIPI_TXM0 function select : 1 : VI0_D[9] 2 : VO_D[4] 3 : XGPIOC[12] (default) 4 : CAM_MCLK1 5 : PWM[14] 6 : CAM_VS0 7 : DBG[12] Others : Reserved
88	PAD_MIPI_TXP0	FMUX_GPIO_REG_IOCTRL_PAD_MIPI_TXP0 0x0300_11B8	0x3	IO PAD_MIPI_TXP0 function select : 1 : VI0_D[10] 2 : VO_D[3] 3 : XGPIOC[13] (default) 4 : CAM_MCLK0 5 : PWM[15] 6 : CAM_HS0 7 : DBG[13] Others : Reserved
2	PAD_AUD_AINL_MIC	fmux_gpio_reg_ioctrl_pad_aud_ainl_mic 0x0300_11BC	0x3	IO PAD_AUD_AINL_MIC function select : 3 : XGPIOC[23] (default) 4 : IIS1_BCLK 5 : IIS2_BCLK Others : Reserved
4	PAD_AUD_AOUTR	fmux_gpio_reg_ioctrl_pad_aud_a outr 0x0300_11C8	0x3	IO PAD_AUD_AOUTR function select : 3 : XGPIOC[24] (default) 4 : IIS1_DI 5 : IIS2_DO 6 : IIS1_DO Others : Reserved
67	GPIO_RTX EPHY_RTX	FMUX_GPIO_REG_IOCTRL_GPIO_RTX 0x0300_11CC	0x3	IO GPIO_RTX function select . 3 : XGPIOB[23] (default) 4 : PWM[1] 5 : CAM_MCLK0 Others : Reserved
35	GPIO_ZQ PAD_ZQ	FMUX_GPIO_REG_IOCTRL_GPIO_ZQ 0x0300_11D0	0x3	IO GPIO_ZQ function select . 3 : PWR_GPIO[24] (default) 4 : PWM [2] Others : Reserved
#N/A	PKG_TYPE0	FMUX_GPIO_REG_IOCTRL_PKG_TYPE0 0x0300_1104	0x0	IO PKG_TYPE0 function select : 0 : PKG_TYPE0 (default) Others : Reserved
#N/A	PKG_TYPE1	FMUX_GPIO_REG_IOCTRL_PKG_TYPE1 0x0300_110C	0x0	IO PKG_TYPE1 function select : 0 : PKG_TYPE1 (default) Others : Reserved
#N/A	PKG_TYPE2	FMUX_GPIO_REG_IOCTRL_PKG_TYPE2 0x0300_1110	0x0	IO PKG_TYPE2 function select : 0 : PKG_TYPE2 (default) Others : Reserved

#N/A	MUX_SPI1_MISO	FMUX_GPIO_REG_IOCTRL_MUX_SPI1_MISO 0x0300_1114	0x3	IO MUX_SPI1_MISO function select . 1 : UART3_RTS 2 : IIC1_SDA 3 : XGPIOB[8] (default) 4 : PWM[9] 5 : KEY_COL1 6 : SPI1_SDI 7 : DBG[14] Others : Reserved
#N/A	MUX_SPI1_MOSI	FMUX_GPIO_REG_IOCTRL_MUX_SPI1_MOSI 0x0300_1118	0x3	IO MUX_SPI1_MOSI function select . 1 : UART3_RX 2 : IIC1_SCL 3 : XGPIOB[7] (default) 4 : PWM[8] 5 : KEY_COL0 6 : SPI1_SDO 7 : DBG[13] Others : Reserved
#N/A	MUX_SPI1_CS	fmux_gpio_reg_ioctrl_mux_spi1_ cs 0x0300_111C	0x3	IO MUX_SPI1_CS function select . 1 : UART3_CTS 2 : CAM_MCLK0 3 : XGPIOB[10] (default) 4 : PWM[11] 5 : KEY_ROW3 6 : SPI1_CS_X 7 : DBG[16] Others : Reserved
#N/A	MUX_SPI1_SCK	fmux_gpio_reg_ioctrl_mux_spi1_ sck 0x0300_1120	0x3	IO MUX_SPI1_SCK function select . 1 : UART3_TX 2 : CAM_MCLK1 3 : XGPIOB[9] (default) 4 : PWM[10] 5 : KEY_ROW2 6 : SPI1_SCK 7 : DBG[15] Others : Reserved

8.2 Pin Control PINCTRL

8.2.1 Pin Control Register Overview

Table *8-3 Pin* and Pin Control Register Correspondence Information Table

Pin Num	Pin Name	IO_cfg_register	Address
6	SD0_CLK	IOBLK_G10_REG_SD0_CLK	0x0300_1A00
7	SD0_CMD	IOBLK_G10_REG_SD0_CMD	0x0300_1A04
8	SD0_D0	IOBLK_G10_REG_SD0_D0	0x0300_1A08
10	SD0_D1	IOBLK_G10_REG_SD0_D1	0x0300_1A0C
11	SD0_D2	IOBLK_G10_REG_SD0_D2	0x0300_1A10
12	SD0_D3	IOBLK_G10_REG_SD0_D3	0x0300_1A14
14	SD0_CD	IOBLK_G7_REG_SD0_CD	0x0300_1900
15	SD0_PWR_EN	IOBLK_G7_REG_SD0_PWR_EN	0x0300_1904
17	SPK_EN	IOBLK_G7_REG_SPK_EN	0x0300_1908
18	UART0_TX	IOBLK_G7_REG_UART0_TX	0x0300_190C
19	UART0_RX	IOBLK_G7_REG_UART0_RX	0x0300_1910
20	EMMC_DAT2	IOBLK_G7_REG_EMMC_DAT2	0x0300_1918
21	EMMC_CLK	IOBLK_G7_REG_EMMC_CLK	0x0300_191C
22	EMMC_DAT0	IOBLK_G7_REG_EMMC_DAT0	0x0300_1920
23	EMMC_DAT3	IOBLK_G7_REG_EMMC_DAT3	0x0300_1924
24	EMMC_CMD	IOBLK_G7_REG_EMMC_CMD	0x0300_1928
25	EMMC_DAT1	IOBLK_G7_REG_EMMC_DAT1	0x0300_192C
26	JTAG_CPU_TMS	IOBLK_G7_REG_JTAG_CPU_TMS	0x0300_1930
27	JTAG_CPU_TCK	IOBLK_G7_REG_JTAG_CPU_TCK	0x0300_1934
28	IIC0_SCL	IOBLK_G7_REG_IIC0_SCL	0x0300_193C
29	IIC0_SDA	IOBLK_G7_REG_IIC0_SDA	0x0300_1940
30	AUX0	IOBLK_G7_REG_AUX0	0x0300_1944
38	PWR_VBAT_DET	IOBLK_GRTC_REG_PWR_VBAT_DET	0x0502_7000
39	PWR_RSTN	IOBLK_GRTC_REG_PWR_RSTN	0x0502_7004
40	PWR_SEQ1	IOBLK_GRTC_REG_PWR_SEQ1	0x0502_7008
41	PWR_SEQ2	IOBLK_GRTC_REG_PWR_SEQ2	0x0502_700C
43	PWR_WAKEUP0	IOBLK_GRTC_REG_PWR_WAKEUP0	0x0502_7018
44	PWR_BUTTON1	IOBLK_GRTC_REG_PWR_BUTTON1	0x0502_7020
45	XTAL_XIN	IOBLK_GRTC_REG_XTAL_XIN	0x0502_7028
47	PWR_GPIO0	IOBLK_GRTC_REG_PWR_GPIO0	0x0502_702C
48	PWR_GPIO1	IOBLK_GRTC_REG_PWR_GPIO1	0x0502_7030
49	PWR_GPIO2	IOBLK_GRTC_REG_PWR_GPIO2	0x0502_7034

51	SD1_D3	IOBLK_GRTC_REG_SD1_D3	0x0502_7058
52	SD1_D2	IOBLK_GRTC_REG_SD1_D2	0x0502_705C
53	SD1_D1	IOBLK_GRTC_REG_SD1_D1	0x0502_7060
54	SD1_D0	IOBLK_GRTC_REG_SD1_D0	0x0502_7064
55	SD1_CMD	IOBLK_GRTC_REG_SD1_CMD	0x0502_7068
56	SD1_CLK	IOBLK_GRTC_REG_SD1_CLK	0x0502_706C
58	PWM0_BUCK	IOBLK_G1_REG_PWM0_BUCK	0x0300_1804
59	ADC1	IOBLK_G1_REG_ADC1	0x0300_1810
60	USB_VBUS_DET	IOBLK_G1_REG_USB_VBUS_DET	0x0300_1820
62	PAD_ETH_TXP EPHY_RXN	#N/A	#N/A
63	PAD_ETH_TXM EPHY_RXP	#N/A	#N/A
64	PAD_ETH_RXP EPHY_TXN	#N/A	#N/A
65	PAD_ETH_RXM EPHY_TXP	#N/A	#N/A
72	PAD_MIPIRX4N	IOBLK_G12_REG_PAD_MIPIRX4N	0x0300_1C38
73	PAD_MIPIRX4P	IOBLK_G12_REG_PAD_MIPIRX4P	0x0300_1C3C
74	PAD_MIPIRX3N	IOBLK_G12_REG_PAD_MIPIRX3N	0x0300_1C40
75	PAD_MIPIRX3P	IOBLK_G12_REG_PAD_MIPIRX3P	0x0300_1C44
76	PAD_MIPIRX2N	IOBLK_G12_REG_PAD_MIPIRX2N	0x0300_1C48
77	PAD_MIPIRX2P	IOBLK_G12_REG_PAD_MIPIRX2P	0x0300_1C4C
78	PAD_MIPIRX1N	IOBLK_G12_REG_PAD_MIPIRX1N	0x0300_1C50
79	PAD_MIPIRX1P	IOBLK_G12_REG_PAD_MIPIRX1P	0x0300_1C54
80	PAD_MIPIRX0N	IOBLK_G12_REG_PAD_MIPIRX0N	0x0300_1C58
81	PAD_MIPIRX0P	IOBLK_G12_REG_PAD_MIPIRX0P	0x0300_1C5C
83	PAD_MIPI_TXM2	IOBLK_G12_REG_PAD_MIPI_TXM2	0x0300_1C70
84	PAD_MIPI_TXP2	IOBLK_G12_REG_PAD_MIPI_TXP2	0x0300_1C74
85	PAD_MIPI_TXM1	IOBLK_G12_REG_PAD_MIPI_TXM1	0x0300_1C78
86	PAD_MIPI_TXP1	IOBLK_G12_REG_PAD_MIPI_TXP1	0x0300_1C7C
87	PAD_MIPI_TXM0	IOBLK_G12_REG_PAD_MIPI_TXM0	0x0300_1C80
88	PAD_MIPI_TXP0	IOBLK_G12_REG_PAD_MIPI_TXP0	0x0300_1C84
2	PAD_AUD_AINL_MIC	#N/A	#N/A
4	PAD_AUD_AOUTR	#N/A	#N/A
67	GPIO_RTX EPHY_RTX	IOBLK_G12_REG_GPIO_RTX	0x0300_1C8C
35	GPIO_ZQ PAD_ZQ	IOBLK_GRTC_REG_GPIO_ZQ	0x0502_70E0
#N/A	PKG_TYPE0	IOBLK_G1_REG_PKG_TYPE0	0x0300_181C
#N/A	PKG_TYPE1	IOBLK_G1_REG_PKG_TYPE1	0x0300_1824
#N/A	PKG_TYPE2	IOBLK_G1_REG_PKG_TYPE2	0x0300_1828
#N/A	MUX_SPI1_MISO	#N/A	#N/A
#N/A	MUX_SPI1_MOSI	#N/A	#N/A
#N/A	MUX_SPI1_CS	#N/A	#N/A
#N/A	MUX_SPI1_SCK	#N/A	#N/A

8.2.2 PINCTRL Register Description

Table *8-4* Pin Control Register Configuration Table

Register Name	Field Name	Bit	Default	Field Description
IOBLK_G1_REG_PWM0_BUCK	IOBLK_G1_REG_PWM0_BUCK_PU	2	0x0	Pull-up resistor enable . 0=None Effective; 1=Effective
	IOBLK_G1_REG_PWM0_BUCK_PD	3	0x0	Pull-down resistor enable. 0=None Effective; 1=Effective
	IOBLK_G1_REG_PWM0_BUCK_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G1_REG_PWM0_BUCK_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G1_REG_PWM0_BUCK_ST0	8	0x0	Input Schmitt Trigger Strength Control. bit 0
	IOBLK_G1_REG_PWM0_BUCK_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G1_REG_PWM0_BUCK_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G1_REG_PWM0_BUCK_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G1_REG_ADC1	IOBLK_G1_REG_ADC1_PU	2	0x0	Pull-up resistor enable . 0=None Effective; 1=Effective
	IOBLK_G1_REG_ADC1_PD	3	0x1	Pull-down resistor enable. 0=None Effective; 1=Effective
	IOBLK_G1_REG_ADC1_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G1_REG_ADC1_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G1_REG_ADC1_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G1_REG_ADC1_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G1_REG_ADC1_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid

	IOBLK_G1_REG_ADC1_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G1_REG_PKG_TYPE0	IOBLK_G1_REG_PKG_TYPE0_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G1_REG_PKG_TYPE0_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G1_REG_PKG_TYPE0_DS0	5	0x0	Transmission drive capacity bit 0

	IOBLK_G1_REG_PKG_TYPE0_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G1_REG_PKG_TYPE0_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G1_REG_PKG_TYPE0_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G1_REG_PKG_TYPE0_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G1_REG_PKG_TYPE0_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G1_REG_USB_VBUS_DET	IOBLK_G1_REG_USB_VBUS_DET_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G1_REG_USB_VBUS_DET_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G1_REG_USB_VBUS_DET_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G1_REG_USB_VBUS_DET_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G1_REG_USB_VBUS_DET_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G1_REG_USB_VBUS_DET_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G1_REG_USB_VBUS_DET_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G1_REG_USB_VBUS_DET_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G1_REG_PKG_TYPE1	IOBLK_G1_REG_PKG_TYPE1_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G1_REG_PKG_TYPE1_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G1_REG_PKG_TYPE1_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G1_REG_PKG_TYPE1_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G1_REG_PKG_TYPE1_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0

	IOBLK_G1_REG_PKG_TYPE1_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G1_REG_PKG_TYPE1_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G1_REG_PKG_TYPE1_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)

IOBLK_G1_REG_PKG_TYPE2	IOBLK_G1_REG_PKG_TYPE2_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G1_REG_PKG_TYPE2_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G1_REG_PKG_TYPE2_DS0	5	0x0	Transmission 🔑 drive capacity bit 0
	IOBLK_G1_REG_PKG_TYPE2_DS1	6	0x1	Transmission 🔑 drive capacity bit 1
	IOBLK_G1_REG_PKG_TYPE2_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G1_REG_PKG_TYPE2_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G1_REG_PKG_TYPE2_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G1_REG_PKG_TYPE2_SL	11	0x0	Input 🔑 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_SD0_CD	IOBLK_G7_REG_SD0_CD_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_SD0_CD_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_SD0_CD_DS0	5	0x0	Transmission 🔑 drive capacity bit 0
	IOBLK_G7_REG_SD0_CD_DS1	6	0x1	Transmission 🔑 drive capacity bit 1
	IOBLK_G7_REG_SD0_CD_DS2	7	0x0	Transmission 🔑 drive capacity bit 2
	IOBLK_G7_REG_SD0_CD_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_SD0_CD_SL	11	0x0	Input 🔑 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_SD0_PWR_EN	IOBLK_G7_REG_SD0_PWR_EN_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_SD0_PWR_EN_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_SD0_PWR_EN_DS0	5	0x0	Transmission 🔑 drive capacity bit 0

	IOBLK_G7_REG_SD0_PWR_EN_DS1	6	0x1	Transmission ⚲ drive capacity bit 1
	IOBLK_G7_REG_SD0_PWR_EN_DS2	7	0x0	Transmission ⚲ drive capacity bit 2
	IOBLK_G7_REG_SD0_PWR_EN_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_SD0_PWR_EN_SL	11	0x0	Input ⚲ level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)

IOBLK_G7_REG_SPK_EN	IOBLK_G7_REG_SPK_EN_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_SPK_EN_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_SPK_EN_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G7_REG_SPK_EN_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G7_REG_SPK_EN_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G7_REG_SPK_EN_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_SPK_EN_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_UART0_TX	IOBLK_G7_REG_UART0_TX_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_UART0_TX_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_UART0_TX_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G7_REG_UART0_TX_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G7_REG_UART0_TX_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G7_REG_UART0_TX_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_UART0_TX_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_UART0_RX	IOBLK_G7_REG_UART0_RX_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_UART0_RX_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_UART0_RX_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G7_REG_UART0_RX_DS1	6	0x1	Transmission drive capacity bit 1

	IOBLK_G7_REG_UART0_RX_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G7_REG_UART0_RX_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_UART0_RX_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_EMMC_DAT2	IOBLK_G7_REG_EMMC_DAT2_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_DAT2_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid

	IOBLK_G7_REG_EMMC_DAT2_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G7_REG_EMMC_DAT2_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G7_REG_EMMC_DAT2_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G7_REG_EMMC_DAT2_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_EMMC_DAT2_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_EMMC_CLK	IOBLK_G7_REG_EMMC_CLK_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_CLK_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_CLK_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G7_REG_EMMC_CLK_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G7_REG_EMMC_CLK_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G7_REG_EMMC_CLK_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_EMMC_CLK_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_EMMC_DAT0	IOBLK_G7_REG_EMMC_DAT0_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_DAT0_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_DAT0_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G7_REG_EMMC_DAT0_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G7_REG_EMMC_DAT0_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G7_REG_EMMC_DAT0_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0

	IOBLK_G7_REG_EMMC_DAT0_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_EMMC_DAT3	IOBLK_G7_REG_EMMC_DAT3_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_DAT3_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_DAT3_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G7_REG_EMMC_DAT3_DS1	6	0x1	Transmission drive capacity bit 1

	IOBLK_G7_REG_EMMC_DAT3_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G7_REG_EMMC_DAT3_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_EMMC_DAT3_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_EMMC_CMD	IOBLK_G7_REG_EMMC_CMD_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_CMD_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_CMD_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G7_REG_EMMC_CMD_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G7_REG_EMMC_CMD_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G7_REG_EMMC_CMD_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_EMMC_CMD_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_EMMC_DAT1	IOBLK_G7_REG_EMMC_DAT1_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_DAT1_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_EMMC_DAT1_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G7_REG_EMMC_DAT1_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G7_REG_EMMC_DAT1_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G7_REG_EMMC_DAT1_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_EMMC_DAT1_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_JTAG_CPU_TMS	IOBLK_G7_REG_JTAG_CPU_TMS_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid

	IOBLK_G7_REG_JTAG_CPU_TMS_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_JTAG_CPU_TMS_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G7_REG_JTAG_CPU_TMS_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G7_REG_JTAG_CPU_TMS_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G7_REG_JTAG_CPU_TMS_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0

	IOBLK_G7_REG_JTAG_CPU_TMS_SL	11	0x0	Input 🔊 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_JTAG_CPU_TCK	IOBLK_G7_REG_JTAG_CPU_TCK_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_JTAG_CPU_TCK_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_JTAG_CPU_TCK_DS0	5	0x0	Transmission 🔊 drive capacity bit 0
	IOBLK_G7_REG_JTAG_CPU_TCK_DS1	6	0x1	Transmission 🔊 drive capacity bit 1
	IOBLK_G7_REG_JTAG_CPU_TCK_DS2	7	0x0	Transmission 🔊 drive capacity bit 2
	IOBLK_G7_REG_JTAG_CPU_TCK_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_JTAG_CPU_TCK_SL	11	0x0	Input 🔊 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_IIC0_SCL	IOBLK_G7_REG_IIC0_SCL_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_IIC0_SCL_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_IIC0_SCL_DS0	5	0x0	Transmission 🔊 drive capacity bit 0
	IOBLK_G7_REG_IIC0_SCL_DS1	6	0x1	Transmission 🔊 drive capacity bit 1
	IOBLK_G7_REG_IIC0_SCL_DS2	7	0x0	Transmission 🔊 drive capacity bit 2
	IOBLK_G7_REG_IIC0_SCL_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_IIC0_SCL_SL	11	0x0	Input 🔊 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G7_REG_IIC0_SDA	IOBLK_G7_REG_IIC0_SDA_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_IIC0_SDA_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_IIC0_SDA_DS0	5	0x0	Transmission 🔊 drive capacity bit 0

	IOBLK_G7_REG_IIC0_SDA_DS1	6	0x1	Transmission ꞔ drive capacity bit 1
	IOBLK_G7_REG_IIC0_SDA_DS2	7	0x0	Transmission ꞔ drive capacity bit 2
	IOBLK_G7_REG_IIC0_SDA_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_IIC0_SDA_SL	11	0x0	Input ꞔ level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)

IOBLK_G7_REG_AUX0	IOBLK_G7_REG_AUX0_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_AUX0_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G7_REG_AUX0_DS0	5	0x0	Transmission 🔑 drive capacity bit 0
	IOBLK_G7_REG_AUX0_DS1	6	0x1	Transmission 🔑 drive capacity bit 1
	IOBLK_G7_REG_AUX0_DS2	7	0x0	Transmission 🔑 drive capacity bit 2
	IOBLK_G7_REG_AUX0_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G7_REG_AUX0_SL	11	0x0	Input 🔑 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G10_REG_SD0_CLK	IOBLK_G10_REG_SD0_CLK_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_CLK_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_CLK_DS0	5	0x0	Transmission 🔑 drive capacity bit 0
	IOBLK_G10_REG_SD0_CLK_DS1	6	0x1	Transmission 🔑 drive capacity bit 1
	IOBLK_G10_REG_SD0_CLK_DS2	7	0x0	Transmission 🔑 drive capacity bit 2
	IOBLK_G10_REG_SD0_CLK_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G10_REG_SD0_CLK_SL	11	0x0	Input 🔑 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G10_REG_SD0_CMD	IOBLK_G10_REG_SD0_CMD_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_CMD_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_CMD_DS0	5	0x0	Transmission 🔑 drive capacity bit 0
	IOBLK_G10_REG_SD0_CMD_DS1	6	0x1	Transmission 🔑 drive capacity bit 1

	IOBLK_G10_REG_SD0_CMD_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G10_REG_SD0_CMD_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G10_REG_SD0_CMD_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G10_REG_SD0_D0	IOBLK_G10_REG_SD0_D0_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_D0_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid

	IOBLK_G10_REG_SD0_D0_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G10_REG_SD0_D0_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G10_REG_SD0_D0_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G10_REG_SD0_D0_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G10_REG_SD0_D0_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G10_REG_SD0_D1	IOBLK_G10_REG_SD0_D1_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_D1_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_D1_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G10_REG_SD0_D1_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G10_REG_SD0_D1_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G10_REG_SD0_D1_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G10_REG_SD0_D1_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G10_REG_SD0_D2	IOBLK_G10_REG_SD0_D2_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_D2_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_D2_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G10_REG_SD0_D2_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G10_REG_SD0_D2_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G10_REG_SD0_D2_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0

	IOBLK_G10_REG_SD0_D2_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G10_REG_SD0_D3	IOBLK_G10_REG_SD0_D3_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_D3_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G10_REG_SD0_D3_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G10_REG_SD0_D3_DS1	6	0x1	Transmission drive capacity bit 1

	IOBLK_G10_REG_SD0_D3_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_G10_REG_SD0_D3_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G10_REG_SD0_D3_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPI RX4N	IOBLK_G12_REG_PAD_MIPIRX4N _PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX4N _PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX4N _DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPIRX4N _DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPIRX4N _ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPIRX4N _ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPIRX4N _HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX4N _SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPI RX4P	IOBLK_G12_REG_PAD_MIPIRX4P _PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX4P _PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX4P _DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPIRX4P _DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPIRX4P _ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPIRX4P _ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPIRX4P _HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid

	IOBLK_G12_REG_PAD_MIPIRX4P_SL	11	0x0	Input ⊗ level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPIRX3N	IOBLK_G12_REG_PAD_MIPIRX3N_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX3N_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid

	IOBLK_G12_REG_PAD_MIPIRX3N_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPIRX3N_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPIRX3N_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPIRX3N_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPIRX3N_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX3N_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPIRX3P	IOBLK_G12_REG_PAD_MIPIRX3P_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX3P_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX3P_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPIRX3P_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPIRX3P_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPIRX3P_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPIRX3P_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX3P_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPIRX2N	IOBLK_G12_REG_PAD_MIPIRX2N_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX2N_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX2N_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPIRX2N_DS1	6	0x1	Transmission drive capacity bit 1

	IOBLK_G12_REG_PAD_MIPIRX2N_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPIRX2N_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPIRX2N_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid

	IOBLK_G12_REG_PAD_MIPIRX2N_SL	11	0x0	Input 🔊 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPIRX2P	IOBLK_G12_REG_PAD_MIPIRX2P_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX2P_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX2P_DS0	5	0x0	Transmission 🔊 drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPIRX2P_DS1	6	0x1	Transmission 🔊 drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPIRX2P_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPIRX2P_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPIRX2P_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX2P_SL	11	0x0	Input 🔊 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPIRX1N	IOBLK_G12_REG_PAD_MIPIRX1N_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX1N_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX1N_DS0	5	0x0	Transmission 🔊 drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPIRX1N_DS1	6	0x1	Transmission 🔊 drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPIRX1N_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPIRX1N_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPIRX1N_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX1N_SL	11	0x0	Input 🔊 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPIRX1P	IOBLK_G12_REG_PAD_MIPIRX1P_PU	2	0x0	Pull-up resistor enable. 0=invalid;

				1=valid
	IOBLK_G12_REG_PAD_MIPIRX1P_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX1P_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPIRX1P_DS1	6	0x1	Transmission drive capacity bit 1

	IOBLK_G12_REG_PAD_MIPIRX1P_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPIRX1P_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPIRX1P_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX1P_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPIRX0N	IOBLK_G12_REG_PAD_MIPIRX0N_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX0N_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX0N_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPIRX0N_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPIRX0N_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPIRX0N_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPIRX0N_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX0N_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPIRX0P	IOBLK_G12_REG_PAD_MIPIRX0P_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX0P_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX0P_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPIRX0P_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPIRX0P_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPIRX0P_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1

	IOBLK_G12_REG_PAD_MIPIRX0P_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPIRX0P_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPI_TXM2	IOBLK_G12_REG_PAD_MIPI_TXM2_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid

	IOBLK_G12_REG_PAD_MIPI_TXM2_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXM2_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPI_TXM2_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPI_TXM2_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPI_TXM2_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPI_TXM2_HE	10	0x0	Weak level maintainer (Bus holder) Enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXM2_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster); 1=valid (slower)
IOBLK_G12_REG_PAD_MIPI_TXP2	IOBLK_G12_REG_PAD_MIPI_TXP2_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXP2_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXP2_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPI_TXP2_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPI_TXP2_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPI_TXP2_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPI_TXP2_HE	10	0x0	Weak level maintainer (Bus holder) Enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXP2_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster); 1=valid (slower)
IOBLK_G12_REG_PAD_MIPI_TXM1	IOBLK_G12_REG_PAD_MIPI_TXM1_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXM1_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXM1_DS0	5	0x0	Transmission drive capacity bit 0

	IOBLK_G12_REG_PAD_MIPI_TXM1_DS1	6	0x1	Transmission 2 drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPI_TXM1_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPI_TXM1_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1

	IOBLK_G12_REG_PAD_MIPI_TXM1_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXM1_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPI_TXP1	IOBLK_G12_REG_PAD_MIPI_TXP1_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXP1_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXP1_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPI_TXP1_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPI_TXP1_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPI_TXP1_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPI_TXP1_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXP1_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPI_TXM0	IOBLK_G12_REG_PAD_MIPI_TXM0_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXM0_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXM0_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_PAD_MIPI_TXM0_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPI_TXM0_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPI_TXM0_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPI_TXM0_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid

	IOBLK_G12_REG_PAD_MIPI_TXM0_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_PAD_MIPI_TXP0	IOBLK_G12_REG_PAD_MIPI_TXP0_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXP0_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXP0_DS0	5	0x0	Transmission drive capacity bit 0

	IOBLK_G12_REG_PAD_MIPI_TXP0_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_PAD_MIPI_TXP0_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_PAD_MIPI_TXP0_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_PAD_MIPI_TXP0_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_PAD_MIPI_TXP0_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_G12_REG_GPIO_RTX	IOBLK_G12_REG_GPIO_RTX_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_GPIO_RTX_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_G12_REG_GPIO_RTX_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_G12_REG_GPIO_RTX_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_G12_REG_GPIO_RTX_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_G12_REG_GPIO_RTX_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_G12_REG_GPIO_RTX_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_G12_REG_GPIO_RTX_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_PWR_VBAT_DET	IOBLK_GRTC_REG_PWR_VBAT_DET_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_VBAT_DET_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_VBAT_DET_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_GRTC_REG_PWR_VBAT_DET_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_GRTC_REG_PWR_VBAT_DET_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0

	IOBLK_GRTC_REG_PWR_VBAT_DET_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_GRTC_REG_PWR_VBAT_DET_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_GRTC_REG_PWR_VBAT_DET_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)

IOBLK_GRTC_REG_PWR_RSTN	IOBLK_GRTC_REG_PWR_RSTN_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_RSTN_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_RSTN_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_GRTC_REG_PWR_RSTN_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_GRTC_REG_PWR_RSTN_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_PWR_RSTN_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_GRTC_REG_PWR_RSTN_HE	10	0x0	Weak level maintainer (Bus holder) Enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_RSTN_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster); 1=valid (slower)
IOBLK_GRTC_REG_PWR_SEQ1	IOBLK_GRTC_REG_PWR_SEQ1_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_SEQ1_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_SEQ1_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_GRTC_REG_PWR_SEQ1_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_GRTC_REG_PWR_SEQ1_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_PWR_SEQ1_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_GRTC_REG_PWR_SEQ1_HE	10	0x0	Weak level maintainer (Bus holder) Enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_SEQ1_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster); 1=valid (slower)
IOBLK_GRTC_REG_PWR_SEQ2	IOBLK_GRTC_REG_PWR_SEQ2_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_SEQ2_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid

	IOBLK_GRTC_REG_PWR_SEQ2_DS0	5	0x0	Transmission ꝏ drive capacity bit 0
	IOBLK_GRTC_REG_PWR_SEQ2_DS1	6	0x1	Transmission ꝏ drive capacity bit 1
	IOBLK_GRTC_REG_PWR_SEQ2_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_PWR_SEQ2_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1

	IOBLK_GRTC_REG_PWR_SEQ2_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_GRTC_REG_PWR_SEQ2_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_PTEST	IOBLK_GRTC_REG_PTEST_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PTEST_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PTEST_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_GRTC_REG_PTEST_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_GRTC_REG_PTEST_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_PTEST_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_GRTC_REG_PTEST_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_GRTC_REG_PTEST_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_PWR_WAKEUP0	IOBLK_GRTC_REG_PWR_WAKEUP0_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_WAKEUP0_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_WAKEUP0_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_GRTC_REG_PWR_WAKEUP0_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_GRTC_REG_PWR_WAKEUP0_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_PWR_WAKEUP0_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_GRTC_REG_PWR_WAKEUP0_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid

	IOBLK_GRTC_REG_PWR_WAKEUP0_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_PWR_BUTTON1	IOBLK_GRTC_REG_PWR_BUTTON1_PU	2	0x1	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_BUTTON1_PD	3	0x0	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_BUTTON1_DS0	5	0x0	Transmission drive capacity bit 0

	IOBLK_GRTC_REG_PWR_BUTTON1_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_GRTC_REG_PWR_BUTTON1_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_PWR_BUTTON1_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_GRTC_REG_PWR_BUTTON1_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_GRTC_REG_PWR_BUTTON1_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_XTAL_XIN	IOBLK_GRTC_REG_XTAL_XIN_XDS0	14	0x0	Crystal output drive capability gears bit 0
	IOBLK_GRTC_REG_XTAL_XIN_XDS1	15	0x0	Crystal output drive capability gears bit 1
	IOBLK_GRTC_REG_XTAL_XIN_XDS2	16	0x1	Crystal output drive capability gears bit 2
IOBLK_GRTC_REG_PWR_GPIO0	IOBLK_GRTC_REG_PWR_GPIO0_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_GPIO0_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_GPIO0_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_GRTC_REG_PWR_GPIO0_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_GRTC_REG_PWR_GPIO0_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_PWR_GPIO0_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_GRTC_REG_PWR_GPIO0_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_GRTC_REG_PWR_GPIO0_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_PWR_GPIO1	IOBLK_GRTC_REG_PWR_GPIO1_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_GPIO1_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid

	IOBLK_GRTC_REG_PWR_GPIO1_DS0	5	0x0	Transmission 2 drive capacity bit 0
	IOBLK_GRTC_REG_PWR_GPIO1_DS1	6	0x1	Transmission 2 drive capacity bit 1
	IOBLK_GRTC_REG_PWR_GPIO1_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_PWR_GPIO1_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1

	IOBLK_GRTC_REG_PWR_GPIO1_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_GRTC_REG_PWR_GPIO1_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_PWR_GPIO2	IOBLK_GRTC_REG_PWR_GPIO2_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_GPIO2_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_PWR_GPIO2_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_GRTC_REG_PWR_GPIO2_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_GRTC_REG_PWR_GPIO2_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_PWR_GPIO2_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_GRTC_REG_PWR_GPIO2_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_GRTC_REG_PWR_GPIO2_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_SD1_D3	IOBLK_GRTC_REG_SD1_D3_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_D3_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_D3_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_GRTC_REG_SD1_D3_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_GRTC_REG_SD1_D3_DS2	7	0x0	Transmission drive capacity bit 2
	IOBLK_GRTC_REG_SD1_D3_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_SD1_D3_SL	11	0x0	Input level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)

IOBLK_GRTC_REG_SD1_D2	IOBLK_GRTC_REG_SD1_D2_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_D2_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_D2_DS0	5	0x0	Transmission drive capacity bit 0
	IOBLK_GRTC_REG_SD1_D2_DS1	6	0x1	Transmission drive capacity bit 1
	IOBLK_GRTC_REG_SD1_D2_DS2	7	0x0	Transmission drive capacity bit 2

	IOBLK_GRTC_REG_SD1_D2_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_SD1_D2_SL	11	0x0	Input 电 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_SD1_D1	IOBLK_GRTC_REG_SD1_D1_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_D1_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_D1_DS0	5	0x0	Transmission 电 drive capacity bit 0
	IOBLK_GRTC_REG_SD1_D1_DS1	6	0x1	Transmission 电 drive capacity bit 1
	IOBLK_GRTC_REG_SD1_D1_DS2	7	0x0	Transmission 电 drive capacity bit 2
	IOBLK_GRTC_REG_SD1_D1_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_SD1_D1_SL	11	0x0	Input 电 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_SD1_D0	IOBLK_GRTC_REG_SD1_D0_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_D0_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_D0_DS0	5	0x0	Transmission 电 drive capacity bit 0
	IOBLK_GRTC_REG_SD1_D0_DS1	6	0x1	Transmission 电 drive capacity bit 1
	IOBLK_GRTC_REG_SD1_D0_DS2	7	0x0	Transmission 电 drive capacity bit 2
	IOBLK_GRTC_REG_SD1_D0_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_SD1_D0_SL	11	0x0	Input 电 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_SD1_CMD	IOBLK_GRTC_REG_SD1_CMD_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_CMD_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid

	IOBLK_GRTC_REG_SD1_CMD_DS0	5	0x0	Transmission ⚲ drive capacity bit 0
	IOBLK_GRTC_REG_SD1_CMD_DS1	6	0x1	Transmission ⚲ drive capacity bit 1
	IOBLK_GRTC_REG_SD1_CMD_DS2	7	0x0	Transmission ⚲ drive capacity bit 2
	IOBLK_GRTC_REG_SD1_CMD_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0

	IOBLK_GRTC_REG_SD1_CMD_SL	11	0x0	Input 电 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_SD1_CLK	IOBLK_GRTC_REG_SD1_CLK_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_CLK_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_SD1_CLK_DS0	5	0x0	Transmission 电 drive capacity bit 0
	IOBLK_GRTC_REG_SD1_CLK_DS1	6	0x1	Transmission 电 drive capacity bit 1
	IOBLK_GRTC_REG_SD1_CLK_DS2	7	0x0	Transmission 电 drive capacity bit 2
	IOBLK_GRTC_REG_SD1_CLK_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_SD1_CLK_SL	11	0x0	Input 电 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)
IOBLK_GRTC_REG_GPIO_ZQ	IOBLK_GRTC_REG_GPIO_ZQ_PU	2	0x0	Pull-up resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_GPIO_ZQ_PD	3	0x1	Pull-down resistor enable. 0=invalid; 1=valid
	IOBLK_GRTC_REG_GPIO_ZQ_DS0	5	0x0	Transmission 电 drive capacity bit 0
	IOBLK_GRTC_REG_GPIO_ZQ_DS1	6	0x1	Transmission 电 drive capacity bit 1
	IOBLK_GRTC_REG_GPIO_ZQ_ST0	8	0x0	Input Schmitt trigger intensity control. bit 0
	IOBLK_GRTC_REG_GPIO_ZQ_ST1	9	0x0	Input Schmitt trigger intensity control. bit 1
	IOBLK_GRTC_REG_GPIO_ZQ_HE	10	0x0	Weak level maintainer (Bus holder) Enable . 0=invalid ; 1=valid
	IOBLK_GRTC_REG_GPIO_ZQ_SL	11	0x0	Input 电 level conversion rate limit. 0=invalid (faster) ; 1=valid (slower)

9 DMA Controller

9.1 summarize

DMA (Direct Memory Access) does not require the CPU to intervene on a stroke-by-stroke basis, and data can be transferred directly between the memory and the device. This mechanism greatly reduces the CPU control time and improves the data transfer rate, which is very suitable for use in large amounts of data transfer. Chip operation often requires multiple channels of data transfer at the same time, a channel needs a DMA hardware support, DMAC (DMA controller) is responsible for the control of multiple channels. Figure 9-1 shows the DMAC hardware control flowchart, the source and destination devices can be converged in different AXIs.

Figure *9-1 DMAC* Hardware Control Flow Diagram

9.2 specificities

DMAC features the following.

a. Up to 8 DMA channels can be established simultaneously.
b. The data source and data destination of the transmission can be set to either memory or device.
c. Only one-way transmission configurations are allowed.
d. Provides DMA transfer pause, resume, and cancel.
e. DMA Burst length configuration is supported.
f. Provides DMA channel priority level configuration.
g. When channel data is transferred between devices, flow control can be left to the devices.
h. Supports hardware chain table functionality.
i. Channel locking is supported, and other channel requests are ignored until the locked channel is completed.

9.3 Functional Description

9.3.1 Peripheral request line

The DMA has eight built-in DMA channels, each of which is requested by the peripheral and needs to be configured to map to the peripheral. Please refer to 7.2.2 Description, configured before the DMA channel is enabled.

9.3.2 Access to space

Table *9-1 DMAC* Access Space Types

Type of space	descriptive
Memory	SRAM
	Non-secure DDR space
peripherals	UART0~UART3
	I2C0~I2C4
	SPI0~SPI3
	SPI_NAND
	I2S0~I2S3

9.3.3 basic transport

algorithmic technology

DMA data transfers are set up in blocks and completed by burst transfers, the length of which can be set.

However, it is often the case that the amount of data in a block is not a perfect integer multiple of the burst length, and the last transaction transmitted will be less than the set burst length, which will then need to be accomplished using a single transmission request.

The sources and purposes of the maximum 8 supported DMA channels can be in the following four combinations:

a. Memory Memory to Memory Memory.
b. memory memory to the device.
c. device to the memory cartridge.
d. Device to Device.

Figure *9-2 DMA* Transmission Class

The amount of data to be transferred individually can be calculated from the following values written by the staging device

- Amount of data transferred by the source (bytes).
  src_single_size_bytes = CHx_CTL.SRC_TR_WIDTH/8
- Source Burst Transmission Data Volume (bytes).
  src_burst_size_bytes = CHx_CTL.SRC_MSIZE * src_single_size_bytes
- Target transmission data amount (bytes).
  dst_single_size_bytes = CHx_CTL.DST_TR_WIDTH/8
- Target Burst Transmission Data Amount (bytes).
  dst_burst_size_bytes = CHx_CTL.DST_MSIZE * dst_single_size_bytes

The transfer process control can be controlled by the DMA controller or the source or destination device. When block data transfer is performed, the amount of transferred data is calculated as follows.

- The DMA controller controls the transfer process.
  blk_size_bytes_dma = CHx_BLOCK_TS.BLOCK_TS * src_single_size_bytes
- Control of the transmission process by the source device.

blk_size_bytes_src = (number of block burst transfers from source device * src_burst_size_bytes)
+ (number of separate transfers of source device blocks * src_single_size_bytes)

- Control of the transmission process by the destination device.

blk_size_bytes_dst = (number of times the destination device issued a block burst transmission * dst_burst_size_bytes) + (number of times the destination device block was transmitted individually * dst_single_size_bytes)

9.3.4 chained-list transport

Chain table transmission is used in the need to carry out a number of discontinuous addresses of the block transmission, in each block of information will be followed by a refining table information, stored in the next node's information, so that the data transfer does not require the intervention of the CPU, you can directly carry out the next discontinuous space of the block transmission. Figure 3-6 shows the configuration format of the chain table information, which must be in accordance with the information format in order to enter the chain table transmission work.

0x3C	31	Reserved	0
0x38	31	Reserved	0
0x34	31	CHx_LLP_STATUS [63:32]	0
0x30	31	CHx_LLP_STATUS [31:0]	0
0x2C	31	Write back for CHx_DSTAT	0
0x28	31	Write back for CHx_SSTAT	0
0x24	31	CHx_CTL [63:32]	0
0x20	31	CHx_CTL [31:0]	0
0x1C	31	CHx_LLP [63:32]	0
0x18	31	CHx_LLP [31:5] 6 | 5 Reserved	0
0x14	31	Reserved	0
0x10	31	CHx_BLOCK_TS [31:0]	0
0x0C	31	CHx_DAR [63:32]	0
0x08	31	CHx_DAR [31:0]	0
0x04	31	CHx_SAR [63:32]	0
0x00	31	CHx_SAR [31:0]	0

Figure *9-3* Relative Addresses and Data Formats of a Linked Table

9.3.5 Interrupts and status

DMAC interrupts are sourced as follows:

a. DMA transfer completion trigger.
b. Block transfer completion trigger.
c. Single transmission completion trigger.
d. Internal error trigger.
e. Channel abort trigger or channel cancel trigger.

Figure 9-4 Diagram of Interrupt States and Sources

Figure *9-4* Diagram of Interrupt States and Sources

algorithmic technology

9.3.6 Channel Safety Configuration

Channel security can be realized by using the awprot and arprot values of each channel. Following the AXI bus protocol, the arprot or awprot value needs to be 0x0 when the channel is a secure channel, and non-secure if it is any other value.

9.4 working method

9.4.1 Clock and Reset

The clock of DMAC is enabled by writing 0x1 to CLK_EN_1[1] to enable the clock to work normally. Write 0x0 to REG_SOFT_RESET_X_SDMA_INIT to reset the DMAC, and write 0x1 to leave the DMAC reset.

9.4.2 initialization

After reset, initialization can be performed as follows

1. Peripheral Configuration: The DMA Channel Mapping section describes how to configure the DMA peripheral request line, and how to configure the peripheral request line bindings according to the usage scenario.
2. Confirm channel shutdown: Write DMAC_ChEnReg of 8 channels supported by DMA to 0x0 to confirm channel shutdown.
3. Confirmation of the interrupt source: connects the register DMAC_COMMONREG_INTSIGNAL_ENABLEREG to the CHx_INSTATUS_ENABLEREG Write 0x0 to disable all interrupt sources, then write 0x1 to enable the demanded interrupt sources.
4. Configure Channel Priority: When a channel transmits data simultaneously, it will judge the order in which it passes according to its priority level.

   Channel Stagger CH_PRIOR The larger the value written, the higher the priority level and the higher the priority to pass.

9.4.3 basic transport

Maximum 8 channels can be supported for simultaneous transfer, DMAC channels need to be enabled after initialization to start data transfer, you can refer to the following steps for memory-to-memory data transfer.

- Read Staging DMAC_ChEnReg Get Idle Channel
- Write 0x0 to channel staging SRC_MULTBLK_TYPE, DST_MULTBLK_TYPE configured

for continuous block transfer, respectively

- Write 0x0 to staging TT_FC to configure the channel for memory-to-memory data transfer.
- Write the transferred information to the staging device CHx_SAR, CHx_ADR, CHx_BLOCK_TS, CHx_CTL
- Write 0x1 to staging DMAC_ChEnReg to enable the selected DMA channel
- The software can get the BLOCK_TFR_DONE status by interrupt or polling, when its value rises to 1, that is

Indicates that the data transfer has been completed, then write 0x0 to DMAC_ChEnReg to close the channel and restore it to an idle channel.

9.4.4 linked list transmission

There is no limit to the number of nodes in a chained table transfer, except for the end node, each node must have information pointing to the next node.

1. Read Staging DMAC_ChEnReg Get idle channel.
2. Configure 0x3 write channel stubs SRC_MULTBLK_TYPE, DST_MULTBLK_TYPE as a chained table transfer, respectively.
3. Configuration stager CHx_LLP, CHx_CTL.ShadowReg_Or_LLI_Valid, CHx_CTL.LLI_Last to write the information needed to point to the first node.
4. Write 0x1 to staging DMAC_ChEnReg to enable the selected DMA channel
5. Software can obtain the BLOCK_TFR_DONE status by interrupting or polling, when its value rises to
   When 1, it means the last node data transmission is completed, and then write 0x0 to DMAC_ChEnReg.

   Close the channel and return it to the idle channel.

9.4.5 interrupt processing

The processing steps after an interrupt is triggered are as follows

1. Find out the source of interrupt: Read CHx_INTSTATUS and DMAC_INSTATUSREG to find out the source of interrupt with value 0x1. When an interrupt occurs, it will be recorded as 0x1 on the corresponding selector element. If more than one interrupt occurs at the same time, the software can prioritize the interrupt to be served first.
2. **Clear Interrupt: Write 0x1 to the selected bit of CHx_INTCLEARREG or DMAC_INTCLEARREG,** the interrupt status **recorded in** CHx_INTSTATUS and DMAC_INSTATUSREG will be returned to 0x0, and the interrupt condition will be re-recorded for the next occurrence.

9.5 DMAC register

DMAC_IDREG

Offset Address: 0x000

Bits	Name	Access	Description	Reset
63:0	DMAC_IDREG	RO	DMAC ID Number	

DMAC_COMPVERREG

Offset Address: 0x008

Bits	Name	Access	Description	Reset
31:0	DMAC_COMPVER	RO	DMAC Component Version Number.	

DMAC_CFGREG

Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	DMAC_EN	R/W	This bit is used to enable the DW_axi_dmac. ■ 0: DW_axi_dmac disabled ■ 1: DW_axi_dmac enabled NOTE: If this bit DMAC_EN bit is cleared while any channel is still active, then this bit still returns 1 to indicate that there are channels still active until DW_axi_dmac hardware has terminated all activity on all channels, at which point this bit returns zero (0).	0x0
1	INT_EN	R/W	This bit is used to globally enable the interrupt generation. ■ 0: DW_axi_dmac Interrupts are disabled ■ 1: DW_axi_dmac Interrupt logic is enabled.	0x0
31:2	Reserved			

DMAC_CHENREG

Offset Address: 0x018

Bits	Name	Access	Description	Reset
0	CH1_EN	R/W	This bit is used to enable the DW_axi_dmac Channel-1. ■ 0: DW_axi_dmac Channel-1 is disabled ■ 1: DW_axi_dmac Channel-1 is enabled The bit 'DMAC_ChEnReg.CH1_EN' is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.	0x0
1	CH2_EN	R/W	This bit is used to enable the DW_axi_dmac Channel-2. ■ 0: DW_axi_dmac Channel-2 is disabled ■ 1: DW_axi_dmac Channel-2 is enabled The bit 'DMAC_ChEnReg.CH2_EN' is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.	0x0
2	CH3_EN	R/W	This bit is used to enable the	0x0

Bits	Name	Access	Description	Reset
			DW_axi_dmac Channel-3. ■ 0: DW_axi_dmac Channel-3 is disabled ■ 1: DW_axi_dmac Channel-3 is enabled The bit 'DMAC_ChEnReg.CH3_EN' is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.	
3	CH4_EN	R/W	This bit is used to enable the DW_axi_dmac Channel-4. ■ 0: DW_axi_dmac Channel-4 is disabled ■ 1: DW_axi_dmac Channel-4 is enabled The bit 'DMAC_ChEnReg.CH4_EN' is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.	0x0
4	CH5_EN	R/W	This bit is used to enable the DW_axi_dmac Channel-5. ■ 0: DW_axi_dmac Channel-5 is disabled ■ 1: DW_axi_dmac Channel-5 is enabled The bit 'DMAC_ChEnReg.CH5_EN' is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.	0x0
5	CH6_EN	R/W	This bit is used to enable the DW_axi_dmac Channel-6. ■ 0: DW_axi_dmac Channel-6 is disabled ■ 1: DW_axi_dmac Channel-6 is enabled The bit 'DMAC_ChEnReg.CH6_EN' is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the	0x0

Bits	Name	Access	Description	Reset
			destination has Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.	
6	CH7_EN	R/W	This bit is used to enable the DW_axi_dmac Channel-7. ■ 0: DW_axi_dmac Channel-7 is disabled ■ 1: DW_axi_dmac Channel-7 is enabled The bit 'DMAC_ChEnReg.CH7_EN' is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.	0x0
7	CH8_EN	R/W	This bit is used to enable the DW_axi_dmac Channel-8. ■ 0: DW_axi_dmac Channel-8 is disabled ■ 1: DW_axi_dmac Channel-8 is enabled The bit 'DMAC_ChEnReg.CH8_EN' is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.	0x0
8	CH1_EN_WE	WO	DW_axi_dmac Channel-1 Enable Write Enable bit. Read back value of this register bit is always '0'.	0x0
9	CH2_EN_WE	WO	DW_axi_dmac Channel-2 Enable Write Enable bit. Read back value of this register bit is always '0'.	0x0
10	CH3_EN_WE	WO	DW_axi_dmac Channel-3 Enable Write Enable bit. Read back value of this register bit is always '0'.	0x0
11	CH4_EN_WE	WO	DW_axi_dmac Channel-4 Enable Write Enable bit. Read back value of this register bit is always '0'.	0x0

Bits	Name	Access	Description	Reset
12	CH5_EN_WE	WO	DW_axi_dmac Channel-5 Enable Write Enable bit. Read back value of this register bit is always '0'.	0x0
13	CH6_EN_WE	WO	DW_axi_dmac Channel-6 Enable Write Enable bit. Read back value of this register bit is always '0'.	0x0
14	CH7_EN_WE	WO	DW_axi_dmac Channel-7 Enable Write Enable bit. Read back value of this register bit is always '0'.	0x0
15	CH8_EN_WE	WO	DW_axi_dmac Channel-8 Enable Write Enable bit. Read back value of this register bit is always '0'.	0x0
16	CH1_SUSP	R/W	Channel-1 Suspend Request. Software sets this bit to 1 to request channel suspend. if this bit is set to 1, DW_axi_dmac suspends all DMA data transfers from the source gracefully until this bit is cleared. There is no guarantee that the current dma transaction will complete. This bit can also be used in conjunction with CH1_Status.CH_SUSPENDED to cleanly disable the In this case, software first sets CH1_SUSP bit to 1 and polls CH1_Status.CH_SUSPENDED till it is set to 1. Software can then clear CH1_EN bit to 0 to disable the channel. ■ 0: No Channel Suspend Request. ■ 1: Request for Channel Suspend. Software can clear CH1_SUSP bit to 0, the after DW_axi_dmac sets CH1_Status.CH_SUSPENDED bit to 1, to exit the channel suspend mode. Note: CH_SUSP is cleared when channel is disabled.	0x0
17	CH2_SUSP	R/W	Channel-2 Suspend Request. Software sets this bit to 1 to request channel suspend. if this bit is set to 1, DW_axi_dmac suspends all DMA data transfers from the source gracefully	0x0

Bits	Name	Access	Description	Reset
			until this bit is cleared. There is no guarantee that the current dma transaction will complete. This bit can also be used in conjunction with CH2_Status.CH_SUSPENDED to cleanly disable the In this case, software first sets CH2_SUSP bit to 1 and polls CH2_Status.CH_SUSPENDED till it is set to 1. Software can then clear CH2_EN bit to 0 to disable the channel. ■ 0: No Channel Suspend Request. ■ 1: Request for Channel Suspend. Software can clear CH2_SUSP bit to 0, the after DW_axi_dmac sets CH2_Status.CH_SUSPENDED bit to 1, to exit the channel suspend mode. Note: CH_SUSP is cleared when channel is disabled.	
18	CH3_SUSP	R/W	Channel-3 Suspend Request. Software sets this bit to 1 to request channel suspend. if this bit is set to 1, DW_axi_dmac suspends all DMA data transfers from the source gracefully until this bit is cleared. There is no guarantee that the current dma transaction will complete. This bit can also be used in conjunction with CH3_Status.CH_SUSPENDED to cleanly disable the In this case, software first sets CH3_SUSP bit to 1 and polls CH3_Status.CH_SUSPENDED till it is set to 1. Software can then clear CH3_EN bit to 0 to disable the channel. ■ 0: No Channel Suspend Request. ■ 1: Request for Channel Suspend. Software can clear CH3_SUSP bit to 0, the after DW_axi_dmac sets CH3_Status.CH_SUSPENDED bit to 1, to exit the channel suspend mode. Note: CH_SUSP is cleared when channel is disabled.	0x0
19	CH4_SUSP	R/W	Channel-4 Suspend Request. Software sets this bit to 1 to request channel suspend. if this bit is set to 1, DW_axi_dmac suspends all DMA data transfers from the source gracefully	0x0

Bits	Name	Access	Description	Reset
			until this bit is cleared. There is no guarantee that the current dma transaction will complete. This bit can also be used in conjunction with CH4_Status.CH_SUSPENDED to cleanly disable the In this case, software first sets CH4_SUSP bit to 1 and polls CH4_Status.CH_SUSPENDED till it is set to 1. Software can then clear CH4_EN bit to 0 to disable the channel. ■ 0: No Channel Suspend Request. ■ 1: Request for Channel Suspend. Software can clear CH4_SUSP bit to 0, the after DW_axi_dmac sets CH4_Status.CH_SUSPENDED bit to 1, to exit the channel suspend mode. Note: CH_SUSP is cleared when channel is disabled.	
20	CH5_SUSP	R/W	Channel-5 Suspend Request. Software sets this bit to 1 to request channel suspend. if this bit is set to 1, DW_axi_dmac suspends all DMA data transfers from the source gracefully until this bit is cleared. There is no guarantee that the current dma transaction will complete. This bit can also be used in conjunction with CH5_Status.CH_SUSPENDED to cleanly disable the In this case, software first sets CH5_SUSP bit to 1 and polls CH5_Status.CH_SUSPENDED till it is set to 1. Software can then clear CH5_EN bit to 0 to disable the channel. ■ 0: No Channel Suspend Request. ■ 1: Request for Channel Suspend. Software can clear CH5_SUSP bit to 0, the after DW_axi_dmac sets CH5_Status.CH_SUSPENDED bit to 1, to exit the channel suspend mode. Note: CH_SUSP is cleared when channel is disabled.	0x0
21	CH6_SUSP	R/W	Channel-6 Suspend Request. Software sets this bit to 1 to request channel suspend. if this bit is set to 1, DW_axi_dmac suspends all DMA data transfers from the source gracefully	0x0

Bits	Name	Access	Description	Reset
			until this bit is cleared. There is no guarantee that the current dma transaction will complete. This bit can also be used in conjunction with CH6_Status.CH_SUSPENDED to cleanly disable the In this case, software first sets CH6_SUSP bit to 1 and polls CH6_Status.CH_SUSPENDED till it is set to 1. Software can then clear CH6_EN bit to 0 to disable the channel. ■ 0: No Channel Suspend Request. ■ 1: Request for Channel Suspend. Software can clear CH6_SUSP bit to 0, the after DW_axi_dmac sets CH6_Status.CH_SUSPENDED bit to 1, to exit the channel suspend mode. Note: CH_SUSP is cleared when channel is disabled.	
22	CH7_SUSP	R/W	Channel-7 Suspend Request. Software sets this bit to 1 to request channel suspend. if this bit is set to 1, DW_axi_dmac suspends all DMA data transfers from the source gracefully until this bit is cleared. There is no guarantee that the current dma transaction will complete. This bit can also be used in conjunction with CH7_Status.CH_SUSPENDED to cleanly disable the In this case, software first sets CH7_SUSP bit to 1 and polls CH7_Status.CH_SUSPENDED till it is set to 1. Software can then clear CH7_EN bit to 0 to disable the channel. ■ 0: No Channel Suspend Request. ■ 1: Request for Channel Suspend. Software can clear CH7_SUSP bit to 0, the after DW_axi_dmac sets CH7_Status.CH_SUSPENDED bit to 1, to exit the channel suspend mode. Note: CH_SUSP is cleared when channel is disabled.	0x0
23	CH8_SUSP	R/W	Channel-8 Suspend Request. Software sets this bit to 1 to request channel suspend. if this bit is set to 1, DW_axi_dmac suspends all DMA data transfers from the source gracefully	0x0

Bits	Name	Access	Description	Reset
			until this bit is cleared. There is no guarantee that the current dma transaction will complete. This bit can also be used in conjunction with CH8_Status.CH_SUSPENDED to cleanly disable the In this case, software first sets CH8_SUSP bit to 1 and polls CH8_Status.CH_SUSPENDED till it is set to 1. Software can then clear CH8_EN bit to 0 to disable the channel. ■ 0: No Channel Suspend Request. ■ 1: Request for Channel Suspend. Software can clear CH8_SUSP bit to 0, the after DW_axi_dmac sets CH8_Status.CH_SUSPENDED bit to 1, to exit the channel suspend mode. Note: CH_SUSP is cleared when channel is disabled.	
24	CH1_SUSP_WE	WO	This bit is used as a write enable to the Channel-1 Suspend The read back value of this register bit is always 0.	0x0
25	CH2_SUSP_WE	WO	This bit is used as a write enable to the Channel-2 Suspend The read back value of this register bit is always 0.	0x0
26	CH3_SUSP_WE	WO	This bit is used as a write enable to the Channel-3 Suspend The read back value of this register bit is always 0.	0x0
27	CH4_SUSP_WE	WO	This bit is used as a write enable to the Channel-4 Suspend The read back value of this register bit is always 0.	0x0
28	CH5_SUSP_WE	WO	This bit is used as a write enable to the Channel-5 Suspend The read back value of this register bit is always 0.	0x0
29	CH6_SUSP_WE	WO	This bit is used as a write enable to the Channel-6 Suspend The read back value of this register bit is always 0.	0x0
30	CH7_SUSP_WE	WO	This bit is used as a write enable to the Channel-7 Suspend The read back value of this register bit is always 0.	0x0
31	CH8_SUSP_WE	WO	This bit is used as a write enable to the Channel-8 Suspend The read back value of this register bit is always 0.	0x0
32	CH1_ABORT	R/W	Channel-1 Abort Request. Software sets this bit to 1 to request If this bit	0x0

Bits	Name	Access	Description	Reset
			is set to 1, DW_axi_dmac disables the channel immediately. Aborting the channel might result in AXI Protocol violation as DW_axi_dmac does not make sure that all AXI transfers initiated on the master interface are completed.Aborting the channel is not recommended and should be used only in situations where a particular channel hangs due to no response from the corresponding AXI slave interface and software wants to disable the channel without resetting the It is recommended to try channel disabling first and then only opt for channel aborting. ■ 0: No Channel Abort Request. ■ 1: Request for Channel Abort. DW_axi_dmac clears this bit to 0 once the channel is aborted (when it sets CH1_Status.CH_ABORTED bit to 1).	
33	CH2_ABORT	R/W	Channel-2 Abort Request. Software sets this bit to 1 to request channel abort. if this bit is set to 1, DW_axi_dmac disables the channel immediately. Aborting the channel might result in AXI Protocol violation as DW_axi_dmac does not make sure that all AXI transfers initiated on the master interface are completed.Aborting the channel is not recommended and should be used only in situations where a particular channel hangs due to no response from the corresponding AXI slave interface and software wants to disable the channel without resetting the It is recommended to try channel disabling first and then only opt for channel aborting. ■ 0: No Channel Abort Request. ■ 1: Request for Channel Abort. DW_axi_dmac clears this bit to 0 once the channel is aborted (when it sets CH2_Status.CH_ABORTED bit to 1).	0x0
34	CH3_ABORT	R/W	Channel-3 Abort Request. Software sets this bit to 1 to request If this bit	0x0

Bits	Name	Access	Description	Reset
			is set to 1, DW_axi_dmac disables the channel immediately. Aborting the channel might result in AXI Protocol violation as DW_axi_dmac does not make sure that all AXI transfers initiated on the master interface are completed.Aborting the channel is not recommended and should be used only in situations where a particular channel hangs due to no response from the corresponding AXI slave interface and software wants to disable the channel without resetting the It is recommended to try channel disabling first and then only opt for channel aborting. ■ 0: No Channel Abort Request. ■ 1: Request for Channel Abort. DW_axi_dmac clears this bit to 0 once the channel is aborted (when it sets CH3_Status.CH_ABORTED bit to 1).	
35	CH4_ABORT	R/W	Channel-4 Abort Request. Software sets this bit to 1 to request channel abort. if this bit is set to 1, DW_axi_dmac disables the channel immediately. Aborting the channel might result in AXI Protocol violation as DW_axi_dmac does not make sure that all AXI transfers initiated on the master interface are completed.Aborting the channel is not recommended and should be used only in situations where a particular channel hangs due to no response from the corresponding AXI slave interface and software wants to disable the channel without resetting the It is recommended to try channel disabling first and then only opt for channel aborting. ■ 0: No Channel Abort Request. ■ 1: Request for Channel Abort. DW_axi_dmac clears this bit to 0 once the channel is aborted (when it sets CH4_Status.CH_ABORTED bit to 1).	0x0
36	CH5_ABORT	R/W	Channel-5 Abort Request. Software sets this bit to 1 to request If this bit	0x0

Bits	Name	Access	Description	Reset
			is set to 1, DW_axi_dmac disables the channel immediately. Aborting the channel might result in AXI Protocol violation as DW_axi_dmac does not make sure that all AXI transfers initiated on the master interface are completed.Aborting the channel is not recommended and should be used only in situations where a particular channel hangs due to no response from the corresponding AXI slave interface and software wants to disable the channel without resetting the It is recommended to try channel disabling first and then only opt for channel aborting. ■ 0: No Channel Abort Request. ■ 1: Request for Channel Abort. DW_axi_dmac clears this bit to 0 once the channel is aborted (when it sets CH5_Status.CH_ABORTED bit to 1).	
37	CH6_ABORT	R/W	Channel-6 Abort Request. Software sets this bit to 1 to request channel abort. if this bit is set to 1, DW_axi_dmac disables the channel immediately. Aborting the channel might result in AXI Protocol violation as DW_axi_dmac does not make sure that all AXI transfers initiated on the master interface are completed.Aborting the channel is not recommended and should be used only in situations where a particular channel hangs due to no response from the corresponding AXI slave interface and software wants to disable the channel without resetting the It is recommended to try channel disabling first and then only opt for channel aborting. ■ 0: No Channel Abort Request. ■ 1: Request for Channel Abort. DW_axi_dmac clears this bit to 0 once the channel is aborted (when it sets CH6_Status.CH_ABORTED bit to 1).	0x0
38	CH7_ABORT	R/W	Channel-7 Abort Request. Software sets this bit to 1 to request If this bit	0x0

Bits	Name	Access	Description	Reset
			is set to 1, DW_axi_dmac disables the channel immediately. Aborting the channel might result in AXI Protocol violation as DW_axi_dmac does not make sure that all AXI transfers initiated on the master interface are completed.Aborting the channel is not recommended and should be used only in situations where a particular channel hangs due to no response from the corresponding AXI slave interface and software wants to disable the channel without resetting the It is recommended to try channel disabling first and then only opt for channel aborting. ■ 0: No Channel Abort Request. ■ 1: Request for Channel Abort. DW_axi_dmac clears this bit to 0 once the channel is aborted (when it sets CH7_Status.CH_ABORTED bit to 1).	
39	CH8_ABORT	R/W	Channel-8 Abort Request. Software sets this bit to 1 to request channel abort. if this bit is set to 1, DW_axi_dmac disables the channel immediately. Aborting the channel might result in AXI Protocol violation as DW_axi_dmac does not make sure that all AXI transfers initiated on the master interface are completed.Aborting the channel is not recommended and should be used only in situations where a particular channel hangs due to no response from the corresponding AXI slave interface and software wants to disable the channel without resetting the It is recommended to try channel disabling first and then only opt for channel aborting. ■ 0: No Channel Abort Request. ■ 1: Request for Channel Abort. DW_axi_dmac clears this bit to 0 once the channel is aborted (when it sets CH8_Status.CH_ABORTED bit to 1).	0x0
40	CH1_ABORT_WE	R/W	This bit is used to write enable the Channel-1 Abort bit. The read back value of this register bit is	0x0

Bits	Name	Access	Description	Reset
			always 0.	
41	CH2_ABORT_WE	R/W	This bit is used to write enable the Channel-2 Abort bit. The read back value of this register bit is always 0.	0x0
42	CH3_ABORT_WE	R/W	This bit is used to write enable the Channel-3 Abort bit. The read back value of this register bit is always 0.	0x0
43	CH4_ABORT_WE	R/W	This bit is used to write enable the Channel-4 Abort bit. The read back value of this register bit is always 0.	0x0
44	CH5_ABORT_WE	R/W	This bit is used to write enable the Channel-5 Abort bit. The read back value of this register bit is always 0.	0x0
45	CH6_ABORT_WE	R/W	This bit is used to write enable the Channel-6 Abort bit. The read back value of this register bit is always 0.	0x0
46	CH7_ABORT_WE	R/W	This bit is used to write enable the Channel-7 Abort bit. The read back value of this register bit is always 0.	0x0
47	CH8_ABORT_WE	R/W	This bit is used to write enable the Channel-8 Abort bit. The read back value of this register bit is always 0.	0x0
63:48	RSVD_DMAC_CHENREG	RO	DMAC_CHENREG Reserved bits	

DMAC_INTSTATUSREG

Offset Address: 0x030

Bits	Name	Access	Description	Reset
0	CH1_IntStat	RO	Channel 1 Interrupt Status Bit.	
1	CH2_IntStat	RO	Channel 2 Interrupt Status Bit.	
2	CH3_IntStat	RO	Channel 3 Interrupt Status Bit.	
3	CH4_IntStat	RO	Channel 4 Interrupt Status Bit.	
4	CH5_IntStat	RO	Channel 5 Interrupt Status Bit.	
5	CH6_IntStat	RO	Channel 6 Interrupt Status Bit.	
6	CH7_IntStat	RO	Channel 7 Interrupt Status Bit.	
7	CH8_IntStat	RO	Channel 8 Interrupt Status Bit.	
15:8	Reserved			
16	CommonReg_IntStat	RO	Common Register Interrupt Status Bit.	
31:17	Reserved			

DMAC_COMMONREG_INTCLEARREG

Offset Address: 0x038

Bits	Name	Access	Description	Reset
0	Clear_SLVIF_CommonReg_DEC_ERR_IntStat	WO	Slave Interface Common Register Decode Error Interrupt clear Bit.	0x0

Bits	Name	Access	Description	Reset
			This bit is used to clear the corresponding channel interrupt status bit (SLVIF_CommonReg_DEC_ERR_IntStat in DMAC_CommonReg_IntStatusReg.	
1	Clear_SLVIF_CommonReg_WR2RO_ERR_IntStat	WO	Slave Interface Common Register Write to Read only Error Interrupt clear Bit. This bit is used to clear the corresponding channel interrupt status bit(SLVIF_CommonReg_WR2RO_ERR_IntStat in DMAC_CommonReg_IntStatusReg.	0x0
2	Clear_SLVIF_CommonReg_RD2WO_ERR_IntStat	WO	Slave Interface Common Register Read to Write only Error Interrupt clear Bit. This bit is used to clear the corresponding channel interrupt status bit(SLVIF_CommonReg_RD2WO_ERR_IntStat in DMAC_CommonReg_IntStatusReg.	0x0
3	Clear_SLVIF_CommonReg_WrOnHold_ERR_IntStat	WO	Slave Interface Common Register Write On Hold Error Interrupt clear Bit. This bit is used to clear the corresponding channel interrupt status bit(SLVIF_CommonReg_WrOnHold_ERR_IntStat in DMAC_CommonReg_IntStatusReg.	0x0
7:4	Reserved			
8	Clear_SLVIF_UndefinedReg_DEC_ERR_IntStat	WO	Slave Interface Undefined register Decode Error Interrupt clear Bit. This bit is used to clear the corresponding channel interrupt status bit(SLVIF_UndefinedReg_DEC_ERR_IntStat in DMAC_CommonReg_IntStatusReg.	0x0
31:9	Reserved			

DMAC_COMMONREG_INTSTATUS_ENABLEREG

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	Enable_SLVIF_CommonReg_DEC_ERR_IntStat	R/W	Slave Interface Common Register Decode Error Interrupt Status Enable Bit. This bit is used to enable the corresponding channel interrupt status bit (SLVIF_CommonReg_DEC_ERR_IntStat	0x0

Bits	Name	Access	Description	Reset
			in DMAC_CommonReg_IntStatusReg.	
1	Enable_SLVIF_CommonReg_WR2RO_ERR_IntStat	R/W	Slave Interface Common Register Write to Read only Error Interrupt Status Enable Bit. This bit is used to enable the corresponding channel interrupt status bit (SLVIF_CommonReg_WR2RO_ERR_IntStat in DMAC_CommonReg_IntStatusReg.	0x0
2	Enable_SLVIF_CommonReg_RD2WO_ERR_IntStat	R/W	Slave Interface Common Register Read to Write only Error Interrupt Status Enable Bit. This bit is used to enable the corresponding channel interrupt status bit (SLVIF_CommonReg_RD2WO_ERR_IntStat in DMAC_CommonReg_IntStatusReg.	0x0
3	Enable_SLVIF_CommonReg_WrOnHold_ERR_IntStat	R/W	Slave Interface Common Register Write On Hold Error Interrupt Status Enable Bit. This bit is used to enable the corresponding channel interrupt status bit (SLVIF_CommonReg_WrOnHold_ERR_IntStat in DMAC_CommonReg_IntStatusReg.	0x0
7:4	Reserved			
8	Enable_SLVIF_UndefinedReg_DEC_ERR_IntStat	R/W	Slave Interface Undefined register Decode Error Interrupt Status enable Bit. This bit is used to enable the corresponding channel interrupt status bit (SLVIF_UndefinedReg_DEC_ERR_IntStat in DMAC_CommonReg_IntStatusReg.	0x0
31:9	Reserved			

DMAC_COMMONREG_INTSIGNAL_ENABLEREG

Offset Address: 0x048

Bits	Name	Access	Description	Reset
0	Enable_SLVIF_CommonReg_DEC_ERR_IntSignal	R/W	Slave Interface Common Register Decode Error Interrupt Signal Enable Bit. This bit is used to enable the propagation of corresponding channel interrupt status bit (SLVIF_CommonReg_DEC_ERR_IntStat in DMAC_CommonReg_IntStatusReg) to generate a port level interrupt.	0x0
1	Enable_SLVIF_CommonReg_WR2RO	R/W	Slave Interface Common Register Write	0x0

Bits	Name	Access	Description	Reset
	_ERR_IntSignal		to Read only Error Interrupt Signal Enable Bit. This bit is used to enable the propagation of corresponding channel interrupt status bit (SLVIF_CommonReg_WR2RO_ERR_IntStat in DMAC_CommonReg_IntStatusReg) to generate a port level interrupt.	
2	Enable_SLVIF_CommonReg_RD2WO_ERR_IntSignal	R/W	Slave Interface Common Register Read to Write only Error Interrupt Signal Enable Bit. This bit is used to enable the propagation of corresponding channel interrupt status bit (SLVIF_CommonReg_RD2WO_ERR_IntStat in DMAC_CommonReg_IntStatusReg) to generate a port level interrupt.	0x0
3	Enable_SLVIF_CommonReg_WrOnHold_ERR_IntSignal	R/W	Slave Interface Common Register Write On Hold Error Interrupt Signal Enable Bit. This bit is used to enable the propagation of corresponding channel interrupt status bit(SLVIF_CommonReg_WrOnHold_ERR_IntStat in DMAC_CommonReg_IntStatusReg) to generate a port level interrupt.	0x0
7:4	Reserved			
8	Enable_SLVIF_UndefinedReg_DEC_ERR_IntSignal	R/W	Slave Interface Undefined register Decode Error Interrupt Signal Enable Bit. This bit is used to enable the propagation of corresponding channel interrupt status bit(SLVIF_UndefinedReg_DEC_ERR_IntStat in DMAC_CommonReg_IntStatusReg) to generate a port level interrupt.	0x0
31:9	Reserved			

DMAC_COMMONREG_INTSTATUSREG

Offset Address: 0x050

Bits	Name	Access	Description	Reset
0	SLVIF_CommonReg_DEC_ERR_IntStat	RO	Slave Interface Common Register Decode Error Interrupt Status Bit. Decode Error generated by DW_axi_dmac during register This error occurs if the register access is to an	

			invalid address in the common register	

Bits	Name	Access	Description	Reset
			space (0x000 to 0x0FF) resulting in error response by DW_axi_dmac slave interface. ■ 0: No Slave Interface Decode Errors. ■ 1: Slave Interface Decode Error detected. The Error Interrupt status is generated if the corresponding Status Enable bit in DMAC_CommonReg_IntStatus_Enable register bit is set to 1. This bit is cleared to 0 on writing 1 to the corresponding channel interrupt clear bit in DMAC_COMMONREG_INTCLEARREG on enabling the channel (required when the interrupt is not enabled).	
1	SLVIF_CommonReg_WR2RO_ERR_IntStat	RO	Slave Interface Common Register Write to Read Only Error Interrupt Status bit. This error occurs if the write operation is performed to a Read Only register in the common register space (0x000 to 0x0FF). ■ 0: No Slave Interface Write to Read Only Errors. ■ 1: Slave Interface Write to Read Only Error detected. Error Interrupt status is generated if the corresponding Status Enable bit in DMAC_CommonReg_IntStatus_Enable register bit is set to 1. This bit is cleared to 0 on writing 1 to the corresponding channel interrupt clear bit in DMAC_COMMONREG_INTCLEARREG on enabling the channel (required when the interrupt is not enabled).	
2	SLVIF_CommonReg_RD2WO_ERR_IntStat	RO	Slave Interface Common Register Read to Write only Error Interrupt Status bit. This error occurs if Read operation is performed to a Write operation. Only register in the common register space (0x000 to 0x0FF). ■ 0: No Slave Interface Read to Write Only Errors. ■ 1: Slave Interface Read to Write Only Error detected. Error Interrupt status is generated if the corresponding Status Enable bit in	

Bits	Name	Access	Description	Reset
			DMAC_CommonReg_IntStatus_Enable register bit is set to 1. This bit is cleared to 0 on writing 1 to the corresponding channel interrupt clear bit in DMAC_COMMONREG_INTCLEARREG on enabling the channel (required when the interrupt is not enabled). Values. ■ 0x1 (Active_CommonReg_RD2WO_ERR). Slave Interface Read to Write Only Error detected ■ 0x0 (Inactive_CommonReg_RD2WO_ERR). No Slave Interface Read to Write Only Errors	
3	SLVIF_CommonReg_WrOnHold_ERR_IntStat	RO	Slave Interface Common Register Write On Hold Error Interrupt Status Bit. This error occurs if an illegal write operation is performed on a common register; this happens if a write operation is performed on a common register except DMAC_RESETREG with DMAC_RST field set to 1 when DW_axi_dmac is in Hold mode. ■ 0: No Slave Interface Common Register Write On Hold Errors. ■ 1: Slave Interface Common Register Write On Hold Error detected. Error Interrupt Status is generated if the corresponding Status Enable bit in DMAC_CommonReg_IntStatus_Enable register bit is set to 1. This bit is cleared to 0 on writing 1 to the corresponding channel interrupt clear bit in DMAC_COMMONREG_INTCLEARREG on enabling the channel (required when the interrupt is not enabled).	
7:4	Reserved			
8	SLVIF_UndefinedReg_DEC_ERR_IntStat	RO	Slave Interface Undefined register Decode Error Interrupt Signal Enable Bit. Decode Error generated by DW_axi_dmac during register This error occurs if the register access is to undefined address range (>0x8FF if 8	

Bits	Name	Access	Description	Reset
			channels are configured, >0x4FF if 4 channels are configured etc.) resulting in error response by DW_axi_dmac slave interface. ■ 0: No Slave Interface Decode Errors. ■ 1: Slave Interface Decode Error detected. Error Interrupt Status is generated if the corresponding Status Enable bit in DMAC_CommonReg_IntStatus_Enable register bit is set to 1. This bit is cleared to 0 on writing 1 to the corresponding channel interrupt clear bit in DMAC_COMMONREG_INTCLEARREG on enabling the channel (required when the interrupt is not enabled).	
31:9	Reserved			

DMAC_RESETREG

Offset Address: 0x058

Bits	Name	Access	Description	Reset
0	DMAC_RST	R/W	DMAC Reset Request bit Software writes 1 to this bit to reset the DW_axi_dmac and polls this bit to see it as 0. DW_axi_dmac resets all the modules except the slave bus interface module and clears This bit to 0. NOTE: Software is not allowed to write 0 to this bit.	0x0
31:1	Reserved			

CHx_SAR

Offset Address: 0x100

Bits	Name	Access	Description	Reset
63:0	SAR	R/W	Current Source Address of DMA transfer. Updated after each source transfer. The SINC fields in the CHx_CTL register determines whether the address increments or is left unchanged on every source transfer throughout the block transfer.	0x0

CHx_DAR

Offset Address: 0x108

Bits	Name	Access	Description	Reset
63:0	DAR	R/W	Current Destination Address of DMA transfer. Updated after each destination transfer.	0x0

Bits	Name	Access	Description	Reset
			The DINC fields in The CHx_CTL register determines whether the address increments or is left unchanged on every destination transfer throughout the block transfer.	

CHx_BLOCK_TS

Offset Address: 0x110

Bits	Name	Access	Description	Reset
21:0	BLOCK_TS	R/W	Block Transfer Size. The number programmed into BLOCK_TS field indicates the total number of data of width CHx_CTL.SRC_TR_WIDTH to be transferred in a DMA block transfer. Block Transfer Size = BLOCK_TS+1 When the transfer starts, the read-back value is the total number of data items already read from the source peripheral, regardless of who is the flow controller. When the source or destination peripheral is assigned as the flow controller, the value before the transfer starts saturates at DMAX_CHx_MAX_BLK_SIZE, but the actual block size can be greater.	0x0
31:22	Reserved			

CHx_CTL

Offset Address: 0x118

Bits	Name	Access	Description	Reset
0	SMS	R/W	Source Master Select. Identifies the Master Interface layer from which the source device (peripheral or memory) is accessed. ■ 0: AXI master 1 ■ 1: AXI Master 2	0x0
1	Reserved			
2	DMS	R/W	Destination Master Select. Identifies the Master Interface layer from which the destination device (peripheral or memory) is accessed. ■ 0: AXI master 1 ■ 1: AXI Master 2	0x0
3	Reserved			

4	SINC	R/W	Source Address Increment. Indicates whether to increment the source address on every If the device is fetching data from a source	0x0

Bits	Name	Access	Description	Reset
			peripheral FIFO with a fixed address. then set this field to 'No change'. ■ 0: Increment ■ 1: No Change	
5	Reserved			
6	DINC	R/W	Destination Address Increment. Indicates whether to increment the destination address on the If the device is writing data from a source peripheral FIFO with a fixed address, then set this field to 'No change'. ■ 0: Increment ■ 1: No Change	0x0
7	Reserved			
10:8	SRC_TR_WIDTH	R/W	Source Transfer Width. Mapped to AXI bus arsize, this value must be less than or equal to DMAX_M_DATA_WIDTH.	0x0
13:11	DST_TR_WIDTH	R/W	Destination Transfer Width. Mapped to AXI bus awsize, this value must be less than or equal to DMAX_M_DATA_WIDTH.	0x0
17:14	SRC_MSIZE	R/W	Source Burst Transaction Length. number of data items, each of width CHx_CTL.SRC_TR_WIDTH, to be read from the source every time a source burst transaction request is made from the corresponding hardware or software handshaking interface. The maximum value of DST_MSIZE is limited by DMAX_CHx_MAX_MULT_SIZE.	0x0
21:18	DST_MSIZE	R/W	Destination Burst Transaction Length. number of data items, each of width CHx_CTL.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from the corresponding hardware or software handshaking interface.Note: This Value is not related to the AXI awlen signal.	0x0
25:22	AR_CACHE	R/W	AXI 'ar_cache' signal	0x0
29:26	AW_CACHE	R/W	AXI 'aw_cache' signal	0x0
30	NonPosted_LastWrite_En	R/W	Non Posted Last Write Enable This bit decides whether posted writes can be used. throughout the block transfer. ■ 0: Posted writes may be used throughout the block transfer. ■ 1: Posted writes may be used till the	0x0

Bits	Name	Access	Description	Reset
			end of the block (inside a block) and the last write in the block must be This is to synchronize block completion interrupt generation to the last write data reaching the end memory/peripheral.	
31	Reserved			
34:32	AR_PROT	R/W	AXI 'ar_prot' signal	0x0
37:35	AW_PROT	R/W	AXI 'aw_prot' signal	0x0
38	ARLEN_EN	R/W	Source Burst Length Enable If this bit is set to 1, DW_axi_dmac uses the value of CHx_CTL.ARLEN as AXI Burst length for source data transfer till the extent possible; remaining transfers use maximum possible burst length. If this bit is set to 0, DW_axi_dmac uses any possible value that is less than or equal to DMAX_CHx_MAX_AMBA_BURST_LENGTH as AXI Burst length for source data transfer.	0x0
46:39	ARLEN	R/W	Source Burst Length AXI Burst length used for source data transfer. the specified burst length is used for source data transfer till the extent possible; remaining transfers use maximum possible value that is less than or equal to DMAX_CHx_MAX_AMBA_BURST_LENGTH. The maximum value of ARLEN is limited by DMAX_CHx_MAX_AMBA_BURST_LENGTH	0x0
47	AWLEN_EN	R/W	Destination Burst Length Enable If this bit is set to 1, DW_axi_dmac uses the value of CHx_CTL.AWLEN as AXI Burst length for destination data transfer till the extent possible; remaining transfers use maximum possible burst length. If this bit is set to 0, DW_axi_dmac uses any possible value which is less than or equal to DMAX_CHx_MAX_AMBA_BURST_LENGTH as AXI Burst length for destination data transfer.	0x0
55:48	AWLEN	RO	Destination Burst Length AXI Burst length used for destination data transfer. the specified burst length is used for destination data transfer till	

Bits	Name	Access	Description	Reset
			the extent possible; remaining transfers use maximum possible value that is less than or equal to DMAX_CHx_MAX_AMBA_BURST_LENGTH. The maximum value of AWLEN is limited by DMAX_CHx_MAX_AMBA_BURST_LENGTH.	
56	SRC_STAT_EN	R/W	Source Status Enable Enable the logic to fetch status from source peripheral of channel x pointed to by the content of CHx_SSTATAR register and stores it in CHx_SSTAT register. This value is written back to the CHx_SSTAT location of linked list at end of each block transfer if DMAX_CHx_LLI_WB_EN is set to 1 and if linked list based multi-block transfer is used by either source or destination peripheral.	0x0
57	DST_STAT_EN	R/W	Destination Status Enable Enable the logic to fetch status from destination peripheral of channel x pointed to by the content of CHx_DSTATAR register and stores it in CHx_DSTAT register. This value is written back to the CHx_DSTAT location of linked list at end of each block transfer if DMAX_CHx_LLI_WB_EN is set to 1 and if linked list based multi-block transfer is used by either source or destination peripheral.	0x0
58	IOC_BlkTfr	R/W	Interrupt On completion of Block Transfer This bit is used to control the block transfer completion interrupt generation on a block by block basis for shadow register or linked list based multi-block transfers. Writing 1 to This register field enables CHx_IntStatusReg.BLOCK_TFR_DONE_IntStat field if this interrupt generation is enabled in CHx_IntStatus_EnableReg register and the external interrupt output is is asserted if this interrupt generation is enabled in CHx_IntSignal_EnableReg register.	0x0
61:59	Reserved			
62	SHADOWREG_OR_LLI_LAST	R/W	Last Shadow Register/Linked List Item.	0x0

Bits	Name	Access	Description	Reset
			Indicates whether shadow register content or the linked list item fetched from the memory is the last one or not. ■ 0: Not last Shadow Register/LLI ■ 1: Last Shadow Register/LLI	
63	SHADOWREG_OR_LLI_VALID	R/W	Shadow Register content/Linked List Item valid. Indicates whether the content of shadow register or the linked list item fetched from the memory is valid. ■ 0: Shadow Register content/LLI is invalid. ■ 1: Last Shadow Register/LLI is valid.	0x0

CHx_CFG

Offset Address: 0x120

Bits	Name	Access	Description	Reset
1:0	SRC_MULTBLK_TYPE	RO	Source Multi Block Transfer Type. These bits define the type of multi-block transfer used for source peripheral. ■ 00: Contiguous ■ 01: Reload ■ 10: Shadow Register ■ 11: Linked List If the type selected is Contiguous, the CHx_SAR register is loaded with the value of the end source address of previous block + 1 at the end of every block for multi-block transfers. A new block transfer is then initiated. If the type selected is Reload, the CHx_SAR register is reloaded from the initial value of SAR at the end of every block for multi-block transfers. A new block transfer is then initiated. If the type selected is Shadow Register, the CHx_SAR register is loaded from the content of its shadow register if CHx_CTL.ShadowReg_Or_LLI_Valid bit is set to 1 at the end of every block for multi-block transfers. a new block The transfer is then initiated. If the type selected is Linked List, the CHx_SAR register is loaded from the Linked List if CTL.ShadowReg_Or_LLI_Valid bit is set to 1 at the end of every block for multi-block transfers. A	

Bits	Name	Access	Description	Reset
			new block transfer is then initiated. CHx_CTL and CHx_BLOCK_TS registers are loaded from their initial values or from the contents of their shadow registers (if CHx_CTL.ShadowReg_Or_LLI_Valid bit is set to 1) or from the linked list (if CTL.ShadowReg_Or_LLI_Valid bit is set to 1) at the end of every block for multi-block transfers based on the multi-block transfer type programmed for source and destination peripherals. Contiguous transfer on both source and destination peripheral is not a valid multi-block transfer configuration. This field does not exist if the configuration parameter DMAX_CHx_MULTI_BLK_EN is not selected; in that case, the The read-back value is always 0. Values. ■ 0x0 (CONTINGUOUS): Contiguous Multiblock Type used for Source Transfer ■ 0x1 (RELOAD): Reload Multiblock Type used for Source Transfer ■ 0x2 (SHADOW_REGISTER): Shadow Register based Multiblock Type used for Source Transfer ■ 0x3 (LINKED_LIST): Linked List based Multiblock Type used for Source Transfer	
3:2	DST_MULTBLK_TYPE	R/W	Destination Multi Block Transfer Type. These bits define the type of multi-block transfer used for destination peripheral. ■ 00: Contiguous ■ 01: Reload ■ 10: Shadow Register ■ 11: Linked List If the type selected is Contiguous, the CHx_DAR register is loaded with the value of the end source address of previous block + 1 at the end of every block for multi-block transfers. A new block transfer is then initiated. If the type selected is Reload, the CHx_DAR register is reloaded from the initial value of DAR at the end of every	0x0

Bits	Name	Access	Description	Reset
			block for multi-block transfers. a new block transfer is then initiated. If the type selected is Shadow Register, the CHx_DAR register is loaded from the content of its shadow register if CHx_CTL.ShadowReg_Or_LLI_Valid bit is set to 1 at the end of every block for multi-block transfers. a new block The transfer is then initiated. If the type selected is Linked List, the CHx_DAR register is loaded from the Linked List if CTL.ShadowReg_Or_LLI_Valid bit is set to 1 at the end of every block for multi-block transfers. A new block transfer is then initiated. CHx_CTL and CHx_BLOCK_TS registers are loaded from their initial values or from the contents of their shadow registers (if CHx_CTL.ShadowReg_Or_LLI_Valid bit is set to 1) or from the linked list (if CTL.ShadowReg_Or_LLI_Valid bit is set to 1) at the end of every block for multi-block transfers based on the multi-block transfer type programmed for source and destination peripherals. Contiguous transfer on both source and destination peripheral is not a valid multi-block transfer configuration. This field does not exist if the configuration parameter DMAX_CHx_MULTI_BLK_EN is not selected; in that case, the The read-back value is always 0. Values. ■ 0x0 (CONTINGUOUS): Contiguous Multiblock Type used for Destination Transfer ■ 0x1 (RELOAD): Reload Multiblock Type used for Destination Transfer ■ 0x2 (SHADOW_REGISTER): Shadow Register based Multiblock Type used for Destination Transfer ■ 0x3 (LINKED_LIST): Linked List based Multiblock Type used for Destination Transfer	
31:4	Reserved			

Bits	Name	Access	Description	Reset
34:32	TT_FC	R/W	Transfer Type and Flow Control. The following transfer types are supported. ■ Memory to Memory ■ Memory to Peripheral ■ Peripheral to Memory ■ Peripheral to Peripheral Flow Control can be assigned to the DW_axi_dmac, the source peripheral, or hte destination peripheral. Values. ■ 0x0 (MEM_TO_MEM_DMAC). Transfer Type is memory to memory and Flow Controller is DW_axi_dmac ■ 0x1 (MEM_TO_PER_DMAC): Transfer Type is memory to peripheral and Flow Controller is DW_axi_dmac ■ 0x2 (PER_TO_MEM_DMAC): Transfer Type is peripheral to memory and Flow Controller is DW_axi_dmac ■ 0x3 (PER_TO_PER_DMAC): Transfer Type is peripheral to peripheral and Flow Controller is DW_axi_dmac ■ 0x4 (PER_TO_MEM_SRC): Transfer Type is peripheral to Memory and Flow Controller is Source peripheral ■ 0x5 (PER_TO_PER_SRC): Transfer Type is peripheral to peripheral and Flow Controller is Source peripheral ■ 0x6 (MEM_TO_PER_DST): Transfer Type is memory to peripheral and Flow Controller is Destination peripheral ■ 0x7 (PER_TO_PER_DST): Transfer Type is peripheral to The peripheral and Flow Controller is Destination peripheral	0x0
35	HS_SEL_SRC	R/W	Source Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces (hardware or software) is active for source requests on this channel. ■ 0: Hardware handshaking interface. Software-initiated transaction requests are ignored. ■ 1: Software handshaking interface. Hardware-initiated transaction requests are ignored.	0x0

Bits	Name	Access	Description	Reset
			If the source peripheral is memory, then This bit is ignored.	
36	HS_SEL_DST	R/W	Destination Software or Hardware Handshaking Select. This register selects which of the handshaking interfaces (hardware or software) is active for destination requests on this channel. ■ 0: Hardware handshaking interface. Software-initiated transaction requests are ignored. ■ 1: Software handshaking interface. Hardware-initiated transaction requests are ignored. If the destination peripheral is memory, then this bit is ignored.	0x0
37	SRC_HWHS_POL	RO	Source Hardware Handshaking Interface Polarity. ■ 0: ACTIVE HIGH ■ 1: ACTIVE LOW	
38	DST_HWHS_POL	RO	Destination Hardware Handshaking Interface Polarity. ■ 0: ACTIVE HIGH ■ 1: ACTIVE LOW	
39	SRC_PER	R/W	Assigns a hardware handshaking interface (0 - DMAX_NUM_HS_IF-1) to the source of Channelx if the CHx_CFG.HS_SEL_SRC field is 0; otherwise, this field is The channel can then communicate with the source peripheral connected to that interface through the assigned hardware handshaking interface. reset Value = 1 Note: For correct DW_axi_dmac operation, only one peripheral (source or destination) should be assigned to the same handshaking interface.	0x0
43:40	Reserved			

44	DST_PER	R/W	Assigns a hardware handshaking interface (0 - DMAX_NUM_HS_IF-1) to the destination of Channelx if the CHx_CFG.HS_SEL_DST field is 0; otherwise, this field is The channel can then communicate with the destination peripheral connected to that interface through the assigned hardware handshaking interface. Reset Value = 1	0x0

Bits	Name	Access	Description	Reset
			Note: For correct DW_axi_dmac operation, only one peripheral (source or destination) should be assigned to the same handshaking interface.	
48:45	Reserved			
51:49	CH_PRIOR	R/W	Channel Priority A priority of 7 is the highest priority, and 0 is the lowest. This field must be programmed within the following range. 0: DMAX_NUM_CHANNELS-1 A programmed value outside this range will cause erroneous behavior.	0x0
52	LOCK_CH	R/W	Channel Lock bit When the channel is granted control of the master bus interface and if the CHx_CFG.LOCK_CH bit is asserted. then no other channels are granted control of the master bus interface for the duration specified in CHx_CFG.LOCK_CH_L. Indicates to the master bus interface arbiter that this channel wants exclusive access to the master bus interface for the duration specified in CHx_CFG.LOCK_CH_L. This field does not exist if the configuration parameter DMAX_CHx_LOCK_EN is set to False; in this case, the The read-back value is always 0. Locking the channel locks AXI Read Address, Write Address and Write Data channels on the corresponding master interface. Note: Channel locking feature is supported only for memoryto-memory transfer at Block Transfer and DMA Transfer Hardware does not check for the validity of channel locking setting, hence the software must take care of enabling the channel locking only for memory-to-memory transfers at Block Transfer or DMA Transfer levels. Illegal programming of channel locking might result in unpredictable behavior.	0x0

54:53	LOCK_CH_L	R/W	Channel Lock Level This bit indicates the duration over which	0x0

Bits	Name	Access	Description	Reset
			CHx_CFG.LOCK_CH bit applies. ■ 00: Over complete DMA transfer ■ 01: Over DMA block transfer ■ 1x: Reserved This field does not exist if the configuration parameter DMAX_CHx_LOCK_EN is set to False; in that case, the The read-back value is always 0.	
58:55	SRC_OSR_LMT	R/W	Source Outstanding Request Limit ■ Maximum outstanding request supported is 16. ■ Source Outstanding Request Limit = SRC_OSR_LMT + 1	0x0
62:59	DST_OSR_LMT	R/W	Destination Outstanding Request Limit ■ Maximum outstanding request supported is 16. ■ Source Outstanding Request Limit = DST_OSR_LMT + 1	0x0
63	Reserved			

CHx_LLP

Offset Address: 0x128

Bits	Name	Access	Description	Reset
0	LMS	R/W	LLI master Select This bit identifies the AXI layer/interface where the memory device that stores the next linked list item resides. - 0: AXI Master 1 ■ 1: AXI Master 2 This field does not exist if the configuration parameter DMAX_CHx_LMS is not set to NO_HARDCODE.	0x0
5:1	Reserved			
63:6	LOC	R/W	Starting Address Memory of LLI block Starting Address In Memory of next LLI if block chaining is The six LSBs of the starting address are not stored Because the address is assumed to be aligned to a 64-byte boundary. LLI access always uses the burst size (arsize/awsize) that is same as the data bus width and cannot be changed or Programmed to anything other than this. (awlen/arlen) is chosen based on the data bus width so that the access does not cross one complete LLI structure of 64 bytes. DW_axi_dmac will fetch the entire LLI (40 bytes) in	0x0

Bits	Name	Access	Description	Reset
			one AXI burst if the burst length is not limited by other settings.	

CHx_STATUSREG

Offset Address: 0x130

Bits	Name	Access	Description	Reset
21:0	CMPLTD_BLK_TFR_SIZE	RO	Completed Block Transfer Size. This bit indicates the total number of data of width CHx_CTL.SRC_TR_WIDTH transferred for the previous block transfer.	
31:22	Reserved			
46:32	DATA_LEFT_IN_FIFO	RO	Data Left in FIFO. This bit indicates the total number of data left in DW_axi_dmac channel FIFO after completing the current block transfer.	
63:47	Reserved			

CHx_SWHSSRCREG

Offset Address: 0x138

Bits	Name	Access	Description	Reset
0	SWHS_REQ_SRC	R/W	Software Handshake Request for Channel Source. This bit is used to request dma source data transfer if software handshaking method is selected for the source of This bit is ignored if software handshaking is not enabled for the source of the Channelx. the functionality of this field depends on whether the peripheral is the flow controller or not.	0x0
1	SWHS_REQ_SRC_WE	WO	Write Enable bit for Software Handshake Request for Channel Source.	0x0
2	SWHS_SGLREQ_SRC	R/W	Software Handshake Single Request for Channel Source. This bit is used to request SINGLE (AXI burst length = 1) dma source data transfer if software handshaking method is selected for the source of the corresponding channel. this bit is ignored if software handshaking is not enabled for the source of the Channelx. The functionality of this field depends on whether the peripheral is the flow controller.	0x0

Bits	Name	Access	Description	Reset
3	SWHS_SGLREQ_SRC_WE	WO	Write Enable bit for Software Handshake Single Request for Channel Source.	0x0
4	SWHS_LST_SRC	R/W	Software Handshake Last Request for Channel Source. This bit is used to request LAST dma source data transfer if software handshaking method is selected for the source of This bit is ignored if software handshaking is not enabled for the corresponding channel. the source of the Channelx or if the source of Channelx is not the flow controller.	0x0
5	SWHS_LST_SRC_WE	WO	Write Enable bit for Software Handshake Last Request for Channel Source.	0x0
31:6	Reserved			

CHx_SWHSDSTREG

Offset Address: 0x140

Bits	Name	Access	Description	Reset
0	SWHS_REQ_DST	R/W	Software Handshake Request for Channel Destination. This bit is used to request dma destination data transfer if software handshaking method is selected for the destination of the corresponding channel.	0x0
1	SWHS_REQ_DST_WE	WO	Write Enable bit for Software Handshake Request for Channel Destination.	0x0
2	SWHS_SGLREQ_DST	R/W	Software Handshake Single Request for Channel Destination. This bit is used to request SINGLE (AXI burst length = 1) dma destination data transfer if software handshaking method is selected for the destination of the corresponding channel.	0x0
3	SWHS_SGLREQ_DST_WE	WO	Write Enable bit for Software Handshake Single Request for Channel Destination.	0x0
4	SWHS_LST_DST	R/W	Software Handshake Last Request for Channel Destination. This bit is used to request LAST dma destination data transfer if software handshaking method is selected for the destination of the corresponding channel.	0x0
5	SWHS_LST_DST_WE	WO	Write Enable bit for Software Handshake Last Request for Channel Destination.	0x0

Bits	Name	Access	Description	Reset
31:6	Reserved			

CHx_BLK_TFR_RESUMEREQREG

Offset Address: 0x148

Bits	Name	Access	Description	Reset
0	BLK_TFR_RESUMEREQ	WO	Block Transfer Resume Request during Linked-List or Shadow-Register-based multi-block transfer.	0x0
31:1	Reserved			

CHx_AXI_IDREG

Offset Address: 0x150

Bits	Name	Access	Description	Reset
14:0	AXI_READ_ID_SUFFIX	R/W	AXI Read ID Suffix These bits form part of the ARID output of AXI3/AXI4 master interface.	0x0
15	Reserved			
30:16	AXI_WRITE_ID_SUFFIX	R/W	AXI Write ID Suffix. These bits form part of the AWID output of AXI3/AXI4 master interface.	0x0
31	Reserved			

CHx_AXI_QOSREG

Offset Address: 0x158

Bits	Name	Access	Description	Reset
3:0	AXI_AWQOS	R/W	AXI AWQOS. These bits form the awqos output of AXI4 master interface.	0x0
7:4	AXI_ARQOS	R/W	AXI ARQOS. These bits form the arqos output of AXI4 master interface.	0x0
31:8	Reserved			

CHx_SSTAT

Offset Address: 0x160

Bits	Name	Access	Description	Reset
31:0	SSTAT	RO	Source Status Source status information retrieved by hardware from the address pointed to by the contents of the CHx_SSTATAR register.	

CHx_DSTAT

Offset Address: 0x168

Bits	Name	Access	Description	Reset
31:0	DSTAT	RO	Destination Status Destination status information retrieved by hardware from the address pointed to by the contents	

Bits	Name	Access	Description	Reset
			of the CHx_DSTATAR register.	

CHx_SSTATAR

Offset Address: 0x170

Bits	Name	Access	Description	Reset
63:0	SSTATAR	R/W	Source Status Fetch Address Pointer from where hardware can fetch the source status information, which is registered in the CHx_SSTAT register and written out to the CHx_SSTAT register location of the LLI before the start of the next block if DMAX_CHx_LLI_WB_EN = 1 and linked list based multi-block transfer is enabled for either source or destination peripheral of the channel. Source peripheral should update the source status information, if any, at the location pointed to by CHx_SSTATAR to utilize this feature.	0x0

CHx_DSTATAR

Offset Address: 0x178

Bits	Name	Access	Description	Reset
63:0	DSTATAR	R/W	Destination Status Fetch Address Pointer from where hardware can fetch the Destination status information, which is registered in the CHx_DSTAT register and written out to the CHx_DSTAT register location of the LLI before the start of the next block if DMAX_CHx_LLI_WB_EN = 1 and linked list based multiblock transfer is enabled for either source or destination peripheral of the channel. Destination peripheral should update the destination status information, if any, at the location pointed to by CHx_DSTATAR to utilize this feature.	0x0

CHx_INTSTATUS_ENABLEREG

Offset Address: 0x180

Bits	Name	Access	Description	Reset
0	Enable_BLOCK_TFR_DONE_IntStat	R/W	Block Transfer Done Interrupt Status Enable. ■ 0: Disable the generation of Block Transfer Done Interrupt in CHx_INTSTATUSREG	0x0

Bits	Name	Access	Description	Reset
			■ 1: Enable the generation of Block Transfer Done Interrupt in CHx_INTSTATUSREG	
1	Enable_DMA_TFR_DONE_IntStat	R/W	DMA Transfer Done Interrupt Status Enable. ■ 0: Disable the generation of DMA Transfer Done Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of DMA Transfer Done Interrupt in CHx_INTSTATUSREG	0x0
2	Reserved			
3	Enable_SRC_TRANSCOMP_IntStat	R/W	Source Transaction Completed Status Enable. ■ 0: Disable the generation of Source Transaction Complete Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Source Transaction Complete Interrupt in CHx_INTSTATUSREG	0x0
4	Enable_DST_TRANSCOMP_IntStat	R/W	Destination Transaction Completed Status Enable. ■ 0: Disable the generation of Destination Transaction complete Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Destination Transaction complete Interrupt in CHx_INTSTATUSREG	0x0
5	Enable_SRC_DEC_ERR_IntStat	R/W	Source Decode Error Status Enable. ■ 0: Disable the generation of Source Decode Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Source Decode Error Interrupt in CHx_INTSTATUSREG	0x0
6	Enable_DST_DEC_ERR_IntStat	R/W	Destination Decode Error Status Enable. ■ 0: Disable the generation of Destination Decode Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Destination Decode Error Interrupt in CHx_INTSTATUSREG	0x0
7	Enable_SRC_SLV_ERR_IntStat	R/W	Source Slave Error Status Enable. ■ 0: Disable the generation of Source Slave Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Source Slave Error Interrupt in CHx_INTSTATUSREG	0x0

8	Enable_DST_SLV_ERR_IntStat	R/W	Destination Slave Error Status Enable. ■ 0: Disable the generation of Destination Slave Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of	0x0

Bits	Name	Access	Description	Reset
			Destination Slave Error Interrupt in CHx_INTSTATUSREG	
9	Enable_LLI_RD_DEC_ERR_IntStat	R/W	LLI Read Decode Error Status Enable. ■ 0: Disable the generation of LLI Read Decode Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of LLI Read Decode Error Interrupt in CHx_INTSTATUSREG	0x0
10	Enable_LLI_WR_DEC_ERR_IntStat	R/W	LLI WRITE Decode Error Status Enable. ■ 0: Disable the generation of LLI WRITE Decode Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of LLI WRITE Decode Error Interrupt in CHx_INTSTATUSREG	0x0
11	Enable_LLI_RD_SLV_ERR_IntStat	R/W	LLI Read Slave Error Status Enable. ■ 0: Disable the generation of LLI Read Slave Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of LLI Read Slave Error Interrupt in CHx_INTSTATUSREG	0x0
12	Enable_LLI_WR_SLV_ERR_IntStat	R/W	LLI WRITE Slave Error Status Enable. ■ 0: Disable the generation of LLI WRITE Slave Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of LLI WRITE Slave Error Interrupt in CHx_INTSTATUSREG	0x0
13	Enable_SHADOWREG_OR_LLI_INVALID_ERR_IntStat	R/W	Shadow register or LLI Invalid Error Status Enable. ■ 0: Disable the generation of Shadow Register or LLI Invalid Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Shadow Register or LLI Invalid Error Interrupt in CHx_INTSTATUSREG	0x0
14	Enable_SLVIF_MULTIBLKTYPE_ERR_IntStat	R/W	Slave Interface Multi Block type Error Status Enable. ■ 0: Disable the generation of Slave Interface Multi Block type Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Slave Interface Multi Block type Error Interrupt in CHx_INTSTATUSREG	0x0
15	Reserved			
16	Enable_SLVIF_DEC_ERR_IntStat	R/W	Slave Interface Decode Error Status Enable. ■ 0: Disable the generation of Slave Interface Decode Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Slave	0x0

Bits	Name	Access	Description	Reset
			Interface Decode Error Interrupt in CHx_INTSTATUSREG	
17	Enable_SLVIF_WR2RO_ERR_IntStat	R/W	Slave Interface Write to Read Only Error Status Enable. ■ 0: Disable the generation of Slave Interface Write to Read only Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Slave Interface Write to Read Only Error Interrupt in CHx_INTSTATUSREG	0x0
18	Enable_SLVIF_RD2RWO_ERR_IntStat	R/W	Slave Interface Read to write Only Error Status Enable. ■ 0: Disable the generation of Slave Interface Read to Write only Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Slave Interface Read to Write Only Error Interrupt in CHx_INTSTATUSREG	0x0
19	Enable_SLVIF_WRONCHEN_ERR_IntStat	R/W	Slave Interface Write On Channel Enabled Error Status Enable. ■ 0: Disable the generation of Slave Interface Write On Channel enabled Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Slave Interface Write On Channel enabled Error Interrupt in CHx_INTSTATUSREG	0x0
20	Enable_SLVIF_SHADOWREG_WRON_VALID_ERR_IntStat	R/W	Shadow Register Write On Valid Error Status Enable. ■ 0: Disable the generation of Shadow Register Write On Valid Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Shadow register Write On Valid Error Interrupt in CHx_INTSTATUSREG	0x0
21	Enable_SLVIF_WRONHOLD_ERR_IntStat	R/W	Slave Interface Write On Hold Error Status Enable. ■ 0: Disable the generation of Slave Interface Write On Hold Error Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Slave Interface Write On Hold Error Interrupt in CHx_INTSTATUSREG	0x0
26:22	Reserved			
27	Enable_CH_LOCK_CLEARED_IntStat	R/W	Channel Lock Cleared Status Enable. ■ 0: Disable the generation of Channel LOCK CLEARED Interrupt in CHx_INTSTATUSREG	0x0

Bits	Name	Access	Description	Reset
			■ 1: Enable the generation of Channel LOCK CLEARED Interrupt in CHx_INTSTATUSREG	
28	Enable_CH_SRC_SUSPENDED_IntStat	R/W	Channel Source Suspended Status Enable. ■ 0: Disable the generation of Channel Source Suspended Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Channel Source Suspended Interrupt in CHx_INTSTATUSREG	0x0
29	Enable_CH_SUSPENDED_IntStat	R/W	Channel Suspended Status Enable. ■ 0: Disable the generation of Channel Suspended Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Channel Suspended Interrupt in CHx_INTSTATUSREG	0x0
30	Enable_CH_DISABLED_IntStat	R/W	Channel Disabled Status Enable. ■ 0: Disable the generation of Channel Disabled Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Channel Disabled Interrupt in CHx_INTSTATUSREG	0x0
31	Enable_CH_ABORTED_IntStat	R/W	Channel Aborted Status Enable. ■ 0: Disable the generation of Channel Aborted Interrupt in CHx_INTSTATUSREG ■ 1: Enable the generation of Channel Aborted Interrupt in CHx_INTSTATUSREG	0x0

CHx_INTSTATUS

Offset Address: 0x188

Bits	Name	Access	Description	Reset
0	BLOCK_TFR_DONE_IntStat	RO	Block Transfer Done. This indicates to the software that the DW_axi_dmac has completed the requested block transfer. The DW_axi_dmac sets this bit to 1 when the transfer is successfully completed. ■ 0: Block Transfer not completed. ■ 1: Block Transfer completed. This bit is cleared to 0 on writing 1 to the corresponding channel interrupt clear bit in CHx_IntClearReg register.	
1	DMA_TFR_DONE_IntStat	RO	DMA Transfer Done. This indicates to the software that the DW_axi_dmac has completed the requested DMA transfer. The DW_axi_dmac sets this bit to 1 along with setting CHx_INTSTATUS.BLOCK_TFR_DONE bit to 1 when the last	

Bits	Name	Access	Description	Reset
			block transfer is completed. ■ 0: DMA Transfer not completed. ■ 1: DMA Transfer Completed This bit is cleared to 0 on writing 1	
2	Reserved			
3	SRC_TRANSCOMP_IntStat	RO	Source Transaction Completed. This bit is cleared to 0 on writing 1 to the corresponding channel interrupt clear bit in CHx_IntClearReg register or on enabling the channel (needed when interrupt is not enabled.	
4	DST_TRANSCOMP_IntStat	RO	Destination Transaction Completed. this bit is cleared to 0 on writing 1 to the corresponding channel interrupt clear bit in CHx_IntClearReg register or on enabling the channel (needed when interrupt is not enabled.	
5	SRC_DEC_ERR_IntStat	RO	Source Decode Error. Decode Error detected by Master Interface during source This error occurs if the access is to invalid address and a Decode Error is returned from This error condition causes the DW_axi_dmac to disable the corresponding channel gracefully; the DMAC_ChEnReg.CH_EN bit corresponding to to the channel which received the error is set to 0. ■ 0: No Source Decode Errors. ■ 1: Source Decode Error detected.	
6	DST_DEC_ERR_IntStat	RO	Destination Decode Error. Decode Error detected by Master Interface during destination data transfer. This error occurs if the access is to invalid address and a Decode Error is returned from This error condition causes the DW_axi_dmac to disable the corresponding channel gracefully; the DMAC_ChEnReg.CH_EN bit corresponding to to the channel which received the error is set to 0. ■ 0: No destination Decode Errors. ■ 1: Destination Decode Error Detected	
7	SRC_SLV_ERR_IntStat	RO	Source Slave Error. Slave Error detected by Master Interface during source data This error occurs if the slave	

Bits	Name	Access	Description	Reset
			interface from which The data is read issues a Slave Error. This error condition causes the DW_axi_dmac to disable the corresponding channel gracefully; the DMAC_ChEnReg.CH_EN bit corresponding to the channel which received the error is set to 0. ■ 0: No Source Slave Errors ■ 1: Source Slave Error Detected	
8	DST_SLV_ERR_IntStat	RO	Destination Slave Error. Slave Error detected by Master Interface during destination This error occurs if the slave interface to which The data is written issues a Slave Error. This error condition causes the DW_axi_dmac to disable the corresponding channel gracefully; the DMAC_ChEnReg.CH_EN bit corresponding to the channel which received the error is set to 0. ■ 0: No Destination Slave Errors ■ 1: Destination Slave Errors Detected	
9	LLI_RD_DEC_ERR_IntStat	RO	LLI Read Decode Error. Decode Error detected by Master Interface during LLI read This error occurs if the access is to invalid address and a Decode Error is returned from This error condition causes the DW_axi_dmac to disable the corresponding channel gracefully; the DMAC_ChEnReg.CH_EN1 bit which received The error is set to 0. ■ 0: NO LLI Read Decode Errors. ■ 1: LLI Read Decode Error detected	
10	LLI_WR_DEC_ERR_IntStat	RO	LLI WRITE Decode Error. Decode Error detected by Master Interface during LLI writeback This error occurs if the access is to invalid address and a Decode Error is returned from This error condition causes the DW_axi_dmac to disable the corresponding channel gracefully; the DMAC_ChEnReg.CH_EN1 bit which received The error is set to 0.	

Bits	Name	Access	Description	Reset
			■ 0: NO LLI Write Decode Errors. ■ 1: LLI write Decode Error detected.	
11	LLI_RD_SLV_ERR_IntStat	RO	LLI Read Slave Error. Slave Error detected by Master Interface during LLI read This error occurs if the slave interface on which LLI resides issues a Slave Error. this error condition causes the DW_axi_dmac to disable the corresponding channel gracefully; the DMAC_ChEnReg.CH_EN1 bit which received The error is set to 0. ■ 0: No LLI Read Slave Errors. ■ 1: LLI read Slave Error detected.	
12	LLI_WR_SLV_ERR_IntStat	RO	LLI WRITE Slave Error. Slave Error detected by Master Interface during LLI writeback This error occurs if the slave interface on which LLI resides issues a Slave Error. This error condition causes the DW_axi_dmac to disable the corresponding channel gracefully; the DMAC_ChEnReg.CH_EN1 bit which received the error is set to 0. ■ 0: No LLI write Slave Errors. ■ 1: LLI Write SLAVE Error detected.	
13	SHADOWREG_OR_LLI_INVALID_ERR_IntStat	RO	Shadow register or LLI Invalid Error. This error occurs if CHx_CTL.ShadowReg_Or_LLI_Valid bit is seen to be 0 during DW_axi_dmac Shadow Register / LLI This error condition causes the DW_axi_dmac to halt the corresponding channel gracefully; Error Interrupt is generated if the corresponding channel error interrupt mask bit is set to 0 and the channel waits till software writes (any value) to CHx_BLK_TFR_ResumeReqReg to indicate Valid Shadow Register availability. In the case of LLI pre-fetching, ShadowReg_Or_LLI_Invalid_ERR Interrupt is not generated. even if ShadowReg_Or_LLI_Valid bit is seen to be 0 for the In this case, DW_axi_dmac re-attempts the LLI fetch operation after completing the current block transfer and generates ShadowReg_Or_LLI_Invalid_ERR	

Bits	Name	Access	Description	Reset
			interrupt only if ShadowReg_Or_LLI_Valid bit is still seen to be 0. ■ 0: No Shadow Register / LLI Invalid errors. ■ 1: Shadow Register / LLI Invalid error detected.	
14	SLVIF_MULTIBLKTYPE_ERR_IntStat	RO	Slave Interface Multi Block type Error. this error occurs if multi-block transfer type programmed in CHx_CFG register (SRC_MLTBLK_TYPE and DST_MLTBLK_TYPE) is invalid. this error condition causes the DW_axi_dmac to halt the corresponding channel gracefully; Error Interrupt is generated if the corresponding channel error interrupt mask bit is set to 0 and the channel waits till software writes (any value) to CHx_BLK_TFR_ResumeReqReg to indicate valid multiblock Transfer type availability. ■ 0: No Multi-block transfer type Errors. ■ 1: Multi-block transfer type Error detected.	
15	Reserved			
16	SLVIF_DEC_ERR_IntStat	RO	Slave Interface Decode Error. decode Error generated by DW_axi_dmac during register This error occurs if the register access is to invalid address in Channelx register space resulting in error response by DW_axi_dmac slave interface. ■ 0: No Slave Interface Decode errors. ■ 1: Slave Interface Decode Error detected.	
17	SLVIF_WR2RO_ERR_IntStat	RO	Slave Interface Write to Read Only Error. this error occurs if write operation is performed to a Read Only register. ■ 0: No Slave Interface Write to Read Only Errors. ■ 1: Slave Interface Write to Read Only Error detected.	

18	SLVIF_RD2RWO_ERR_IntStat	RO	Slave Interface Read to Write Only Error. This error occurs if read operation is performed to a Write operation. Only register. ■ 0: No Slave Interface Read to Write Only Errors. ■ 1: Slave Interface Read to Write Only Error detected.	

Bits	Name	Access	Description	Reset
19	SLVIF_WRONCHEN_ERR_IntStat	RO	Slave Interface Write On Channel Enabled Error. This error occurs if an illegal write operation is performed on a register; this happens if a write operation is performed on a register when the channel is enabled and if it is not allowed for the corresponding register as per the DW_axi_dmac specification. ■ 0: No Slave Interface Write On Channel Enabled Errors. ■ 1: Slave Interface Write On Channel Enabled Error detected.	
20	SLVIF_SHADOWREG_WRON_VALID_ERR_IntStat	RO	Shadow Register Write On Valid Error. This error occurs if shadow register based multi-block transfer is enabled and software tries to write to the shadow register when CHx_CTL.ShadowReg_Or_LLI_Valid bit is 1. ■ 0: No Slave Interface Shadow Register Write On Valid Errors. ■ 1: Slave Interface Shadow Register Write On Valid Error detected.	
21	SLVIF_WRONHOLD_ERR_IntStat	RO	Slave Interface Write On Hold Error. this error occurs if an illegal write operation is performed on the a register; this happens if a write operation is performed on a channel register when DW_axi_dmac is in Hold mode. ■ 0: No Slave Interface Write On Hold Errors. ■ 1: Slave Interface Write On Hold Error detected.	
26:22	Reserved			
27	CH_LOCK_CLEARED_IntStat	RO	Channel Lock Cleared. This indicates to the software that the locking of the corresponding channel in DW_axi_dmac is cleared. ■ 0: Channel locking is not cleared. ■ 1: Channel locking is cleared.	
28	CH_SRC_SUSPENDED_IntStat	RO	Channel Source Suspended. This indicates to the software that the corresponding channel source data transfer in DW_axi_dmac is suspended. ■ 0: Channel source is not suspended ■ 1: Channel Source is suspended.	

29	CH_SUSPENDED_IntStat	RO	Channel Suspended.	

Bits	Name	Access	Description	Reset
			This indicates to the software that the corresponding channel in DW_axi_dmac is suspended. ■ 0: Channel is not suspended. ■ 1: Channel is suspended.	
30	CH_DISABLED_IntStat	RO	Channel Disabled. This indicates to the software that the corresponding channel in DW_axi_dmac is disabled. ■ 0: Channel is not disabled. ■ 1: Channel is disabled. Interrupt is generated if the corresponding bit in CHx_INTSTATUS_ENABLEReg is enabled.	
31	CH_ABORTED_IntStat	RO	Channel Aborted. This indicates to the software that the corresponding channel in DW_axi_dmac is aborted. ■ 0: Channel is not aborted ■ 1: Channel is aborted	

CHx_INTSIGNAL_ENABLEREG

Offset Address: 0x190

Bits	Name	Access	Description	Reset
0	Enable_BLOCK_TFR_DONE_IntSignal	R/W	Block Transfer Done Interrupt Signal Enable. ■ 0: Disable the propagation of Block Transfer Done Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Block Transfer Done Interrupt to generate a port level interrupt	0x0
1	Enable_DMA_TFR_DONE_IntSignal	R/W	DMA Transfer Done Interrupt Signal Enable. ■ 0: Disable the propagation of DMA Transfer Done Interrupt to generate a port level interrupt ■ 1: Enable the propagation of DMA Transfer Done Interrupt to generate a port level interrupt	0x0
2	Reserved			
3	Enable_SRC_TRANSCOMP_IntSignal	R/W	Source Transaction Completed Signal Enable. ■ 0: Disable the propagation of Source Transaction Complete Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Source Transaction Complete Interrupt to generate a port level interrupt	0x0

Bits	Name	Access	Description	Reset
4	Enable_DST_TRANSCOMP_IntSignal	R/W	Destination Transaction Completed Signal Enable. ■ 0: Disable the propagation of Destination Transaction complete Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Destination Transaction complete Interrupt to generate a port level interrupt	0x0
5	Enable_SRC_DEC_ERR_IntSignal	R/W	Source Decode Error Signal Enable. ■ 0: Disable the propagation of Source Decode Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Source Decode Error Interrupt to generate a port level interrupt	0x0
6	Enable_DST_DEC_ERR_IntSignal	R/W	Destination Decode Error Signal Enable. ■ 0: Disable the propagation of Destination Decode Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Destination Decode Error Interrupt to generate a port level interrupt	0x0
7	Enable_SRC_SLV_ERR_IntSignal	R/W	Source Slave Error Signal Enable. ■ 0: Disable the propagation of Source Slave Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Source Slave Error Interrupt to generate a port level interrupt	0x0
8	Enable_DST_SLV_ERR_IntSignal	R/W	Destination Slave Error Signal Enable. ■ 0: Disable the propagation of Destination Slave Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Destination Slave Error Interrupt to generate a port level interrupt	0x0
9	Enable_LLI_RD_DEC_ERR_IntSignal	R/W	LLI Read Decode Error Signal Enable. ■ 0: Disable the propagation of LLI Read Decode Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of LLI Read Decode Error Interrupt to generate a port level interrupt	0x0
10	Enable_LLI_WR_DEC_ERR_IntSignal	R/W	LLI WRITE Decode Error Signal Enable. ■ 0: Disable the propagation of LLI WRITE Decode Error Interrupt to generate a port level	0x0

Bits	Name	Access	Description	Reset
			interrupt ■ 1: Enable the propagation of LLI WRITE Decode Error Interrupt to generate a port level interrupt	
11	Enable_LLI_RD_SLV_ERR_IntSignal	R/W	LLI Read Slave Error Signal Enable. ■ 0: Disable the propagation of LLI Read Slave Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of LLI Read Slave Error Interrupt to generate a port level interrupt	0x0
12	Enable_LLI_WR_SLV_ERR_IntSignal	R/W	LLI WRITE Slave Error Signal Enable. ■ 0: Disable the propagation of LLI WRITE Slave Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of LLI WRITE Slave Error Interrupt to generate a port level interrupt	0x0
13	Enable_SHADOWREG_OR_LLI_INVALID_ERR_IntSignal	R/W	Shadow register or LLI Invalid Error Signal Enable. ■ 0: Disable the propagation of Shadow Register or LLI Invalid Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Shadow Register or LLI Invalid Error Interrupt to generate a port level interrupt	0x0
14	Enable_SLVIF_MULTIBLKTYPE_ERR_IntSignal	R/W	Slave Interface Multi Block type Error Signal Enable. ■ 0: Disable the propagation of Slave Interface Multi Block type Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Slave Interface Multi Block type Error Interrupt to generate a port level interrupt	0x0
15	Reserved			
16	Enable_SLVIF_DEC_ERR_IntSignal	R/W	Slave Interface Decode Error Signal Enable. ■ 0: Disable the propagation of Slave Interface Decode Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Slave Interface Decode Error Interrupt to generate a port level interrupt	0x0
17	Enable_SLVIF_WR2RO_ERR_IntSignal	R/W	Slave Interface Write to Read Only Error Signal Enable. ■ 0: Disable the propagation of Slave	0x0

Bits	Name	Access	Description	Reset
			Interface Write to Read only Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Slave Interface Write to Read Only Error Interrupt to generate a port level interrupt	
18	Enable_SLVIF_RD2RWO_ERR_IntSignal	R/W	Slave Interface Read to write Only Error Signal Enable. ■ 0: Disable the propagation of Slave Interface Read to Write only Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Slave Interface Read to Write Only Error Interrupt to generate a port level interrupt	0x0
19	Enable_SLVIF_WRONCHEN_ERR_IntSignal	R/W	Slave Interface Write On Channel Enabled Error Signal Enable. ■ 0: Disable the propagation of Slave Interface Write On Channel enabled Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Slave Interface Write On Channel enabled Error Interrupt to generate a port level interrupt	0x0
20	Enable_SLVIF_SHADOWREG_WRON_VALID_ERR_IntSignal	R/W	Shadow Register Write On Valid Error Signal Enable. ■ 0: Disable the propagation of Shadow Register Write On Valid Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Shadow register Write On Valid Error Interrupt to generate a port level interrupt	0x0
21	Enable_SLVIF_WRONHOLD_ERR_IntSignal	R/W	Slave Interface Write On Hold Error Signal Enable. ■ 0: Disable the propagation of Slave Interface Write On Hold Error Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Slave Interface Write On Hold Error Interrupt to generate a port level interrupt	0x0
26:22	Reserved			
27	Enable_CH_LOCK_CLEARED_IntSignal	R/W	Channel Lock Cleared Signal Enable. ■ 0: Disable the propagation of Channel Lock Cleared Interrupt to generate a port level	0x0

Bits	Name	Access	Description	Reset
			interrupt ■ 1: Enable the propagation of Channel Lock Cleared Interrupt to generate a port level interrupt	
28	Enable_CH_SRC_SUSPENDED_IntSignal	R/W	Channel Source Suspended Signal Enable. ■ 0: Disable the propagation of Channel Source Suspended Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Channel Source Suspended Interrupt to generate a port level interrupt	0x0
29	Enable_CH_SUSPENDED_IntSignal	R/W	Channel Suspended Signal Enable. ■ 0: Disable the propagation of Channel Suspended Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Channel Suspended Interrupt to generate a port level interrupt	0x0
30	Enable_CH_DISABLED_IntSignal	R/W	Channel Disabled Signal Enable. ■ 0: Disable the propagation of Channel Disabled Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Channel Disabled Interrupt to generate a port level interrupt	0x0
31	Enable_CH_ABORTED_IntSignal	R/W	Channel Aborted Signal Enable. ■ 0: Disable the propagation of Channel Aborted Interrupt to generate a port level interrupt ■ 1: Enable the propagation of Channel Aborted Interrupt to generate a port level interrupt	0x0

CHx_INTCLEARREG

Offset Address: 0x198

Bits	Name	Access	Description	Reset
0	Clear_BLOCK_TFR_DONE_IntStat	WO	Block Transfer Done Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CH1_INTSTATUSREG	0x0
1	Clear_DMA_TFR_DONE_IntStat	WO	DMA Transfer Done Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
2	Reserved			
3	Clear_SRC_TRANSCOMP_IntStat	WO	Source Transaction Completed Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0

Bits	Name	Access	Description	Reset
4	Clear_DST_TRANSCOMP_IntStat	WO	Destination Transaction Completed Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
5	Clear_SRC_DEC_ERR_IntStat	WO	Source Decode Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
6	Clear_DST_DEC_ERR_IntStat	WO	Destination Decode Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
7	Clear_SRC_SLV_ERR_IntStat	WO	Source Slave Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
8	Clear_DST_SLV_ERR_IntStat	WO	Destination Slave Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
9	Clear_LLI_RD_DEC_ERR_IntStat	WO	LLI Read Decode Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
10	Clear_LLI_WR_DEC_ERR_IntStat	WO	LLI WRITE Decode Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
11	Clear_LLI_RD_SLV_ERR_IntStat	WO	LLI Read Slave Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
12	Clear_LLI_WR_SLV_ERR_IntStat	WO	LLI WRITE Slave Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
13	Clear_SHADOWREG_OR_LLI_INVALID_ERR_IntStat	WO	Shadow register or LLI Invalid Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
14	Clear_SLVIF_MULTIBLKTYPE_ERR_IntStat	WO	Slave Interface Multi Block type Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
15	Reserved			
16	Clear_SLVIF_DEC_ERR_IntStat	WO	Slave Interface Decode Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0

Bits	Name	Access	Description	Reset
17	Clear_SLVIF_WR2RO_ERR_IntStat	WO	Slave Interface Write to Read Only Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
18	Clear_SLVIF_RD2RWO_ERR_IntStat	WO	Slave Interface Read to write Only Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
19	Clear_SLVIF_WRONCHEN_ERR_IntStat	WO	Slave Interface Write On Channel Enabled Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
20	clear_SLVIF_SHADOWREG_WRON_VALID_ERR_IntStat	WO	Shadow Register Write On Valid Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
21	Clear_SLVIF_WRONHOLD_ERR_IntStat	WO	Slave Interface Write On Hold Error Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
26:22	Reserved			
27	Clear_CH_LOCK_CLEARED_IntStat	WO	Channel Lock Cleared Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
28	Clear_CH_SRC_SUSPENDED_IntStat	WO	Channel Source Suspended Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
29	Clear_CH_SUSPENDED_IntStat	WO	Channel Suspended Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
30	Clear_CH_DISABLED_IntStat	WO	Channel Disabled Interrupt Clear Bit. This bit is used to clear the corresponding channel interrupt status bit in CHx_INTSTATUSREG.	0x0
31	Reserved			

10 timers

10.1 summarize

The system is equipped with 8 timer modules. They can be used for timing and counting functions, and can be supplied to the application program to realize timing and counting, or to the operating system to realize the system clock.

10.2 specificities

The timer has the following features:

- 32bit subtractive timer/counter.
- Supports 2 counting modes: free-running mode and user-defined counting mode.
- The system can read the current count value.

When the count value decreases to 0, an interrupt is generated.

10.3 Functional Description

The Timer is based on a 32bit subtraction counter. The value of the counter is decremented by 1 on the rising edge of each counting clock, and the Timer generates an interrupt when the count value is decremented to zero.
The Timer has the following 2 counting modes:

- Free Run Mode: The timer keeps counting, and when the count value decreases to 0, it automatically returns to its maximum value and continues counting. The maximum count length is 0xFFFF_FFFF.
- User-defined counting mode: The timer keeps counting, and when the count value decreases to 0, the timer starts counting from 0.
  TimerNLoadCount(N=1~8) The initial value is loaded again in

the register and counting continues. The method of loading the initial value of the count for the timer is as follows:

The initial value of the count can be loaded to the timer by writing the TimerNLoadCount(N=1~8) register.

10.4 working method

10.4.1 initialization

Step 1 Write TimerNLoadCount(N=1~8) register to load the initial value of timer.

Step 2 Set TimerNControlReg[2:0](N=1~8) registers, select Timer counting mode, mask Timer interrupt, start the timer to start counting.

10.4.2 interrupt processing

When the timer generates an interrupt, proceed as follows:
Step 1 Read the TimerNEOI(N=1~8) register to clear the TimerN interrupt. Step 2 Execute the process waiting for this interrupt.
Step 3 Upon completion of the process, execution resumes to execute the program that was interrupted.

10.4.3 Clock selection

System Timer Selectable 25MHz/32KHz count clock. Use reg_timer_clk_sel to make the selection.

10.5 Timer Register Overview

Timer registers are accessed via the bus.

An overview of the timer registers is shown in Table 10-1.

Table *10-1 Timer* Register Overview *(*Base Address *0x030A0000)*

Name	Address Offset	Description
Timer1LoadCount	0x000	Value to be loaded into Timer1
Timer1CurrentValue	0x004	Current Value of Timer1
Timer1ControlReg	0x008	Control Register for Timer1
Timer1EOI	0x00c	Clears the interrupt from Timer1
Timer1IntStatus	0x010	Contains the interrupt status for Timer1
Timer2 Registers	0x014~ 0x024	There are 5 registers with the same content as Timer1.
Timer3 Registers	0x028~ 0x038	There are 5 registers with the same content as Timer1.
Timer4 Registers	0x03c~ 0x04c	There are 5 registers with the same content as Timer1.
Timer5 Registers	0x050~ 0x060	There are 5 registers with the same content as Timer1.
Timer6 Registers	0x064~ 0x074	There are 5 registers with the same content as Timer1.
Timer7 Registers	0x078~ 0x088	There are 5 registers with the same content as Timer1.
Timer8 Registers	0x08c~ 0x09c	There are 5 registers with the same content as Timer1.
TimersIntStatus	0x0a0	Contains the interrupt status of all timers in the component.
TimersEOI	0x0a4	Returns all zeroes (0) and clears all active interrupts.
TimersRawIntStatus	0x0a8	Contains the unmasked interrupt status of all timers in the component.

10.6 Timer Register Description

Timer1LoadCount

Offset Address: 0x000

Bits	Name	Access	Description	Reset
31:0	Timer1LoadCount	R/W	Timer1 Load Count Register Value to be loaded into Timer1. This is the value from which counting commences. Any value written to this register is loaded into the associated timer.	0x0

Timer1CurrentValue

Offset Address: 0x004

Bits	Name	Access	Description	Reset
31:0	Timer1CurrentValue	RO	Timer1 Current Value Current Value of Timer1. This register is supported only when timer_1_clk is synchronous to pclk. Reading this register when using independent clocks results in an undefined value.	

Timer1ControlReg

Offset Address: 0x008

Bits	Name	Access	Description	Reset
2:0	Timer1ControlReg	R/W	[2] Timer interrupt mask for Timer1 0 - not masked 1 - masked [1] Timer mode for Timer1 0 - free-running mode 1 - user-defined count mode [0] Timer enable bit for Timer1 0 - disable 1 - enable	0x0
31:3	Reserved			

Timer1EOI

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
0	Timer1EOI	RO	Reading from this register returns all zeroes (0) and clears the interrupt from Timer1.	
31:1	Reserved			

Timer1IntStatus

Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	Timer1IntStatus	RO	Contains the interrupt status for Timer1.	
31:1	Reserved			

TimersIntStatus

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
7:0	TimersIntStatus	RO	Contains the interrupt status of all timers. Reading from this register does not	

Bits	Name	Access	Description	Reset
			clear any active interrupts. 0 - either timer_intr or timer_intr_n is not active after masking 1 - either timer_intr or timer_intr_n is active after masking	
31:8	Reserved			

TimersEOI

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
7:0	TimersEOI	RO	Reading this register returns all zeroes (0) and clears all active interrupts.	
31:8	Reserved			

TimersRawIntStatus

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
7:0	TimersRawIntStatus	RO	The register contains the unmasked interrupt status of all timers. 0 - either timer_intr or timer_intr_n is not active prior to masking 1 - either timer_intr or timer_intr_n is active prior to masking	
31:8	Reserved			

11 watchdog

11.1 summarize

The system is equipped with 4 WatchDog modules. They are used to interrupt or reset the whole system by sending out interrupt or reset signals after a certain period of time in the event of system abnormalities.

11.2 specificities

WatchDog has the following features:

- Configure a 32bit decrementing counter.
- Support for counting initial values (i.e., time intervals) is configurable.
- Supports WatchDog restart protection to prevent WatchDog from being restarted due to misbehavior.
- Supports reset signal generation.
- Supports timeout interrupt generation.

11.3 Functional Description

The WatchDog is configured with register parameter values via the system bus, and the WatchDog sends out WDT_INTR interrupt request to the system at regular intervals, and sends out WDT_SYS_RST reset signal to reset the system if the system does not respond to the interrupt (e.g., when the system is crashed), so as to achieve the purpose of monitoring the operation of the system.

11.3.1 application block diagram

Figure *11-1 WatchDog* Application Block Diagram

11.3.2 Functional principle

The initial value of the WatchDog count is loaded in register WDT_TORR and the operation is based on a 32bit subtraction counter.
With the WatchDog clock enabled, the count value is decremented by 1 on the rising edge of each count clock, and WatchDog generates an interrupt when the count value decrements to 0. The counter then reloads the initial count value from register WDT_TORR on the next rising edge of the count clock to begin decrementing the count. Then on the next rising edge of the count clock, the counter reloads the initial count value from register WDT_TORR and starts counting down again.
If the CPU has not cleared the WatchDog interrupt by the time the counter count value is decremented to 0 for the second count, the
WatchDog will issue the reset signal WDT_SYS_RST and the counter will stop counting.
The user can set WDT_CR[1] to decide whether the first count of the counter will be decremented to 0 or not.
The reset signal WDT_SYS_RST is issued immediately.

11.4 working method

11.4.1 Count Clock Frequency Configuration

The WatchDog count clock is a 25MHz/32KHz clock. Use reg_wdt_clk_sel to make the selection.

11.4.2 System initialization configuration

The WatchDog counter is in a stopped counting state after system power-on reset, and during system initialization you need to set the

WatchDog initializes and starts its operation.The WatchDog initialization process is as follows:Step 1 Write register WDT_TORR to set the WatchDog count initial value.
Step 2 Write register WDT_CR[1] to set the WatchDog count timeout response mode. Step 3 Write register WDT_CR[0] to start WatchDog counting.

11.4.3 interrupt processing

When the system receives an interrupt from WatchDog, its interrupt status should be cleared in a timely manner.
WatchDog interrupt handling proceeds as follows:
Step 1 Read register WDT_EOI to clear the interrupt status of WatchDog. Step 2 Write 0x76 to register WDT_CRR to restart WatchDog.

11.4.4 Close WatchDog

Writing 0 or 1 to register WDT_CR[0] controls the status of the WatchDog:

- 0 : WDT off.
- 1 : WDT is on. Only a system reset can turn WDT off after startup.

11.5 WDT Register Overview

The WDT registers are accessed via the bus. The four WDT base addresses are.

- WDT0 : 0x03010000
- WDT1 : 0x03011000
- WDT2 : 0x03012000
- RTCSYS_WDT : 0x0502D000

An overview of the WDT registers is shown in Table 11-1.

Table *11-1 WDT* Register Overview

Name	Address Offset	Description
WDT_CR	0x000	Control register
WDT_TORR	0x004	Timeout range register
WDT_CCVR	0x008	Current counter value register
WDT_CRR	0x00c	Counter restart register
WDT_STAT	0x010	Interrupt status register
WDT_EOI	0x014	Interrupt clear register
WDT_TOC	0x01C	Time Out Count

11.6 WDT Register Description

WDT_CR

Offset Address: 0x000

Bits	Name	Access	Description	Reset
4:0	WDT_CR	R/W	[4:2] Reset pulse length. This is used to select the number of pclk cycles for which the system reset stays asserted. The range of values available is 2 to 256 pclk cycles. 000 - 2 pclk cycles 001 - 4 pclk cycles 010 - 8 pclk cycles 011 - 16 pclk cycles 100 - 32 pclk cycles 101 - 64 pclk cycles 110 - 128 pclk cycles 111 - 256 pclk cycles [1] Response mode. Selects the output response generated to a timeout. 0 = Generate a system reset. 1 = First generate an interrupt and if it is not cleared by the time a second timeout occurs then generate a system reset. [0] WDT enable. This bit is used to enable and disable the WatchDog. When disabled, the WatchDog will be disabled. The counter does not decrement. Thus, no interrupts or system resets are generated. Once this bit has been enabled, it can be cleared only by a system reset. 0 = WDT disabled. 1 = WDT enabled.	0x0
6	TOR_MODE	R/W	The Mode of Timeout Period	0x0
7	ITOR_MODE	R/W	The Mode of Timeout Period for initialization	0x0
31:5	Reserved			

WDT_TORR

Offset Address: 0x004

Bits	Name	Access	Description	Reset
3:0	WDT_TORR	R/W	[3:0] TOP(TimeOut Period). This field is used to select the timeout period from which the watchdog counter restarts. a change of the timeout period takes effect only after the next counter restart (kick). The range of values is limited by 32-bit width. If TOP is programmed to select a range that is greater than the counter	0x0

Bits	Name	Access	Description	Reset
			width, the timeout period is truncated This affects only the non-user specified values as users are limited to these boundaries during configuration. The range of values available for a 32-bit watchdog counter are. TOR_MODE = 0 T = 2^(16 + WDT_TORR) TOR_MODE = 1 T = WDT_TOC <<(WDT_TORR +1)	
7:4	WDT_ITORR		Initial TimeOut Period ITOR_MODE = 0 T = 2^(16 + WDT_ITORR) ITOR_MODE = 1 T = WDT_TOC <<(WDT_ITORR +1)	
31:4	Reserved			

WDT_CCVR

Offset Address: 0x008

Bits	Name	Access	Description	Reset
31:0	WDT_CCVR	RO	This register, when read, is the current value of the internal counter.	

WDT_CRR

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
7:0	WDT_CRR	R/W	[7:0] Counter Restart Register This register is used to restart the WDT counter. As a safety feature to prevent accidental restarts, the value 0x76 must be written. A restart also clears the WDT interrupt. This register is used to restart the WDT counter. returns zero.	0x0
31:8	Reserved			

WDT_STAT

Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	WDT_STAT	RO	[0] Interrupt Status Register This register shows the interrupt status of the WDT. 1 = Interrupt is active regardless of polarity. 0 = Interrupt is inactive.	
31:1	Reserved			

WDT_EOI

Offset Address: 0x014

Bits	Name	Access	Description	Reset
0	WDT_EOI	RO	[0] Interrupt Clear Register Clears the watchdog interrupt.	

Bits	Name	Access	Description	Reset
			be used to clear the interrupt without restarting the watchdog counter.	
31:1	Reserved			

WDT_TOC

Offset Address: 0x01C

Bits	Name	Access	Description	Reset
15:0	WDT_TOC	R/W	Time out counter	0x0

12 real time clock

12.1 summarize

The Real Time Clock (RTC) is an independently powered module within the chip that contains a 32KHz oscillator and Power-On-Reset/POR sub-module, which can be used for time display and alarm generation. In addition, a hardware state machine provides power-up, power-down and reset triggering and timing control.

12.2 specificities

The RTC has the following characteristics:

- Provides a system reset signal to the chip.
- Provides 32,768 Hz counting clock (error < ±1%)
- Provides 32-bit second counter and hardware calibration circuitry for 5ppm level of second accuracy
- Supports alarm clock configuration and generates alarm interrupts
- Provides 2KB of SRAM for software code or temporary data storage.
- Supports battery low voltage detection and generates an interrupt
- Supports keystroke triggered chip wake-up from hibernation
- Supports software-triggered chip sleep, reset or thermal restart
- Support Watchdog to trigger system-on-chip reset
- Supports alarm triggered chip wake-up from hibernation
- Configurable power-up/down timing and reset interval
- Provide 1 ultra-low power analog second counter (32 bit to count from 1970~2106 year)

12.3 Functional Description

RTC is an independent power supply module within the chip, when the RTC is first powered up, the internal POR module generates a low level pulse, and the 32KHz oscillator starts to oscillate, the POR maintains the low level for more than 13 32KHz clock cycles, and the RTC enters into the initial state. When the state machine detects that the battery voltage is in the normal state, it will start to complete the chip power-up process and release the system reset signal according to the default value. The software needs to initialize the RTC

and configure the count initial value after the first power-on. When the system wants to perform shutdown or enter hibernation, it can be configured by configuring the system control registers.

(RTC_CTRL) triggers the RTC state machine to complete the chip power-down process according to the configured timing.

When the chip is in power-down or hibernation state, it will continue to operate as long as the RTC supply is maintained to save necessary software code or user data in SRAM and information registers (RTC_INFO0 ~ RTC_INFO3, RTC_INFO4, RTC_INFO5, RTC_INFO6, RTC_INFO7, RTC_INFO8, RTC_INFO9). RTC_NOPOR_INFO0 ~ RTC_NOPOR_INFO3); the counter will keep counting and also detect whether the key is

Trigger the chip to power on or wake up from hibernation, and when the trigger is received, the state machine will complete the chip power-up process and release the system reset signal according to the configured timing. After the processor core is restarted, the software can determine the chip status by reading back the contents of the previously written information registers. The RTC also provides two status registers (RTC_ST_ON_REASON, RTC_ST_OFF_REASON), which record the trigger conditions of the previous chip power-down or power-up and reset, and provide the software with more detailed information, such as whether an unanticipated event has ever occurred: forced reset, chip overheating, or battery/power supply disengagement. This provides the software with more detailed information, such as whether an unintended event has occurred: forced reset, chip overheating, or battery/power supply disconnection. In addition, when the RTC receives a Watchdog, it will trigger the state machine to issue a system reset signal according to the configured timing to reboot the chip.

The counting clock of the RTC is a 32KHz clock, and the operation is based on a 32-bit additive counter providing seconds counting, the initial value of which is loaded in the register RTC_SET_SEC_CNTR_VALUE. The initial value of the count is loaded from register RTC_SET_SEC_CNTR_VALUE.

RTC_SEC_CNTR_VALUE Reads back the seconds value, converting it to a specific year, month, day, hour, and minute.

The 32KHz clock and second pulse period can be calibrated by a software process, or the hardware module can be turned on to perform automatic calibration periodically.

Software can enable Alarm by configuring the 32-bit register RTC_ALARM_TIME and writing RTC_ALARM_ENABLE to 1. The RTC will generate an alarm interrupt when the seconds count value, RTC_SEC_CNTR_VALUE, is incremented to equal the RTC_ALARM_TIME value. The interrupt state will be maintained until RTC_ALARM_ENABLE writes 0.

In addition, the RTC provides battery low voltage detection. When the battery voltage falls below a certain level, the RTC will generate an interrupt, and when the software receives the low voltage interrupt, it can instantly execute the shutdown program and trigger the RTC to complete the power-down process, in order to prevent the system from generating abnormal errors.

12.4 working method

12.4.1 Counting Clock Frequency

The RTC's second counter, which counts for a maximum time:

2^32 = 49710 days = 136 years

12.4.2 RTC reset

RTC, as the control unit of chip power-up and power-down, can not be soft reset by itself, except for the POR of the first power-up, it can support to force the whole chip to reset (including RTC) by pressing the RSTN button in case of abnormality, after the RSTN button is released, all the registers of RTC will be restored to the default value, and the state machine will be returned to the initial state. If the state machine detects that the battery voltage is in normal state, it will start to complete the chip power-up process and release the system reset signal according to the default value.

12.4.3 RTC Initialization

The RTC needs to be initialized by the system after the chip is powered up for the first time. However, before initialization, it is necessary to calibrate the 32KHz oscillator clock and the second time. The calibration module uses a 25MHz crystal clock to sample the 32KHz clock, and operates with the software process to sample a 32KHz oscillator clock with a 25MHz crystal clock in coarse tune mode.
32KHz clock cycle, the software adjusts the configuration register RTC_ANA_CALIB[8:0] according to the number of sampling pulses to speed up or slow down the 32KHz oscillating clock cycle time, in order to improve the accuracy of the 32KHz clock. After completing the coarse tuning, you can further enter the fine tuning mode, the default value is 25MHz crystal clock sampling 256 32KHz clock cycles, the software takes the average value of the number of sampled pulses to obtain the number of pulses needed to count the 32KHz clock in 1 second, and writes the value to the registers RTC_SEC_PULSE_GEN_INT, RTC_SEC_PULSE_GEN_F PULSE_GEN_FRAC to complete the second calibration.

The 32KHz clock coarse tuning calibration software flow is as follows:

1. Configuration register RTC_ANA_SEL_FTUNE is 0, and RTC_ANA_CALIB initial value is 0x100.
2. The following calibration is realized by using bindary search:
   FTUNE = RTC_ANA_CALIB; offset = 0x80
3. Configure register RTC_FC_COARSE_EN to 1 to start coarse tuning. Poll the value of RTC_FC_COARSE_TIME until it is greater than the value of the previous read, then configure RTC_FC_COARSE_EN to be 0
4. Read RTC_FC_COARSE_VALUE to get the count value of 25MHz clock sampling one 32KHz clock cycle.
   if (RTC_FC_COARSE_VALUE > 770) FTUNE = FTUNE + offset;
   if (RTC_FC_COARSE_VALUE < 755) FTUNE = FTUNE - offset.
   Write FTUNE value back to register
   RTC_ANA_CALIB offset = offset >> 1;
5. When the value of RTC_FC_COARSE_VALUE is between 755 and 770, the 32KHz clock precision has been reached.
   Within 32,768Hz ± 1%, finish coarse tuning. Otherwise, wait for 0.5ms and repeat steps 3~5 up to 8 times to finish.

The 32KHz clock fine-tuning calibration software process is as follows:

1. Configure register RTC_SEL_SEC_PULSE to 0. Configure RTC_FC_FINE_EN to 1 to start fine tuning.
2. Poll the value of RTC_FC_FINE_TIME until it is greater than the previous read

value

3. Read RTC_FC_FINE_VALUE to get the count of 256 32KHz clock cycles sampled at 25MHz.
4. The 32KHz clock frequency can be obtained from the following equation:
   Frequency = 256 / (RTC_FC_FINE_VALUE x 40ns)
   Example: 256 / (195310 x 40) = 32768.4194357
5. Take the integer part 32768, write to register RTC_SEC_PULSE_GEN_INT, and take the fractional part 8-bit =
   0.4194357 x 256 = 107, write to register RTC_SEC_PULSE_GEN_FRAC
6. Configure RTC_FC_FINE_EN to 0 to end fine tuning.

The clock calibration process can be performed by the software on a one-time basis or periodically, depending on the needs of the system. In addition to the software calibration process

In addition, the RTC also supports hardware-performed auto-calibration on a periodic basis.

After completing the clock calibration, the RTC should be further initialized, only the necessary initialization procedures are listed below, most of the remaining parameter registers are only necessary to be configured when optimizing the power-on and power-off timing of the chip, and it is generally recommended to use the default values.

1. Configure register RTC_POR_DB_MAGIC_KEY with value 0x5AF0 to enable POR debounce to prevent POR from being triggered incorrectly due to a transient drop in the supply voltage of the RTC module. The debounce time is about
   1ms
2. Configure RTC_SET_SEC_CNTR_VALUE to set the RTC time count initial value
3. Write 1 to RTC_SET_SEC_CNTR_TRIG to load the count initial value to the RTC second counter
4. Poll the RTC_SEC_CNTR_VALUE register value until the read value is equal to the RTC_SET_SEC_CNTR_VALUE value
5. Configure RTC_PWR_DET_COMP[0] to 1 to enable battery low voltage detection and configure the
   RTC_PWR_DET_SEL[0] is 1 to issue a low voltage detection interrupt, and the status signal goes low when the battery voltage falls below the threshold value. The threshold value can be adjusted by configuration register RTC_PWR_DET_COMP[12:8].
6. After the RTC is powered up for the first time, the register reg_rtcsys_rstn_src_sel must be reconfigured from the default value of 0 to 1, so that the RTC subsystem will remain in the working state after the chip is powered down (suspend or powerdown). If the value of this register is 0, the RTC subsystem will be reset when the chip is powered down.
7. RTC_EN_AUTO_POWER_UP must be reconfigured from the default value of 1 after the RTC is powered up for the first time.
   0, if the default value is maintained, when the chip enters the power down state, the RTC detects the power down state.

   PWR_VBAT_DET will automatically enter the power-up state if the level is high.

12.4.4 Analog second clock initialization

1. Configure RTC_MACRO_RG_DEFD = 16'hC80 (32000 32KHz clock cycles)
2. Configure RTC_MACRO_DA_SOC_READY to 1
3. Configure RTC_MACRO_DA_CLEAR_ALL to 1
4. Configure RTC_MACRO_DA_CLEAR_ALL to 0

5. Configure RTC_MACRO_RG_SET_T as a Demand Counter value.
6. Configure RTC_MACRO_DA_LATCH_PASS to 1.
7. Configure RTC_MACRO_DA_LATCH_PASS to 0
8. Configure RTC_MACRO_DA_SOC_READY to 0
9. Read the RTC_MACRO_RO_T to get the counter value.

12.4.5 interrupt processing

RTC will send out the status signal of alarm interrupt and low voltage detection interrupt. When the system receives the alarm interrupt, it means the timing time is up and the user can execute the corresponding customized operation. Setting register RTC_ALARM_ENABLE to 0 can

Clear the interrupt status. If you need to continue to set a new timing time, write the new value to register

After RTC_ALARM_TIME, set RTC_ALARM_ENABLE to 1 again.

12.4.6 Sleep and Wake

The system software can configure register req_suspend to 1 to enable the chip to power down and enter the sleep state (suspend), and configure register RTC_EN_PWR_WAKEUP to select the source that can trigger the chip to wake up. Note that before configuring req_suspend, it is necessary to write 0 to RTC_PG_REG to let the DDR IO enter a constant state.
(The DDR IOs are protected by a retent to prevent the DDR data from being corrupted during power down or power up. When the chip wakes up, the system software must write 1 to register RTC_PG_REG to unprotect the DDR IO before executing the DDR initialization process.

In addition, if you want to use key (PWR_ON, PWR_BUTTON, PWR_WAKEUP) to wake up, you have to configure the related IO PINMUX register to lock the IO as RTC input function before entering the hibernation state.

12.4.7 Power down and power up

The system software configures register req_shdn to 1 to enable the chip to include DDR to power down and enter the shutdown state.
(poweroff), configuration register RTC_EN_PWR_UP selects the source from which the chip can be triggered to power up and power on (powerup).

12.5 RTC Register Overview

The RTC registers contain multiple sections, RTC_CORE_REG, RTC_MACRO_REG, and RTC_CTRL_REG, with different register base addresses, all of which are accessed via the bus.

An overview of the RTC_CORE_REG register is shown in Table 12-1.

Table *12-1 RTC_REG* Register Overview *(*Base Address *0x05026000)*

Name	Address Offset	Description
RTC_ANA_CALIB	0x000	32K Oscillator Control

RTC_SEC_PULSE_GEN	0x004	Seconds pulse generator integer and decimal places
RTC_ALARM_TIME	0x008	Set timed alarm time
RTC_ALARM_ENABLE	0x00c	Enable Alarm
RTC_SET_SEC_CNTR_VALUE	0x010	Setting the second counter value
RTC_SET_SEC_CNTR_TRIG	0x014	Load seconds counter value
RTC_SEC_CNTR_VALUE	0x018	Read the current second counter value
RTC_INFO0	0x01c	Information register 0
RTC_INFO1	0x020	Information register 1

Name	Address Offset	Description
RTC_INFO2	0x024	Message register 2
RTC_INFO3	0x028	Information register 3
RTC_NOPOR_INFO0	0x02c	No reset information register 0
RTC_NOPOR_INFO1	0x030	No reset information register 1
RTC_NOPOR_INFO2	0x034	No reset information register 2
RTC_NOPOR_INFO3	0x038	No reset information register 3
rtc_db_pwr_vbat_det	0x040	PWR_VBAT_DET De-jitter time
RTC_DB_BUTTON1	0x048	PWR_BUTTON1 De-jitter time
RTC_DB_PWR_ON	0x04c	PWR_ON De-jitter time
RTC_7SEC_RESET	0x050	Set the number of seconds to press and hold PWR_BUTTON to force reset.
rtc_thm_shdn_auto_reboot	0x064	Select the REQ_THM_SHDN action.
RTC_POR_DB_MAGIC_KEY	0x068	Enable POR long time de-jittering
RTC_DB_SEL_PWR	0x06c	Select PWR_BUTTON de-dithering mode
RTC_UP_SEQ0	0x070	Power-up PWR_SEQ0 Output Timing
RTC_UP_SEQ1	0x074	Power-up PWR_SEQ1 Output Timing
RTC_UP_SEQ2	0x078	Power-up PWR_SEQ2 Output Timing
RTC_UP_SEQ3	0x07c	Power-up PWR_SEQ3 Output Timing
RTC_UP_IF_EN	0x080	Power-Up ISO Release Timing
RTC_UP_RSTN	0x084	Power-on system reset release timing
RTC_UP_MAX	0x088	Power-up process completion timing
RTC_DN_SEQ0	0x090	Power down PWR_SEQ0 Output Timing
RTC_DN_SEQ1	0x094	Power down PWR_SEQ1 Output Timing
RTC_DN_SEQ2	0x098	Power down PWR_SEQ2 Output Timing
RTC_DN_SEQ3	0x09c	Power down PWR_SEQ3 Output Timing
RTC_DN_IF_EN	0x0a0	Power-down ISO turn-on timing
RTC_DN_RSTN	0x0a4	Power-down system reset issuance timing
RTC_DN_MAX	0x0a8	Power-down process completion timing
RTC_PWR_CYC_MAX	0x0b0	Power-cycle Completion Timing
RTC_WARM_RST_MAX	0x0b4	Warm-reset Completion Timing
RTC_EN_7SEC_RST	0x0b8	Setting PWR_BUTTON1 7SEC reset mode
rtc_en_pwr_wakeup	0x0bc	Setting the hibernation wake-up source
RTC_EN_SHDN_REQ	0x0c0	Enable REQ_SHDN
RTC_EN_THM_SHDN	0x0c4	Enable REQ_THM_SHDN
rtc_en_pwr_cyc_req	0x0c8	Enabling REQ_PWR_CYC
RTC_EN_WARM_RST_REQ	0x0cc	Enabling REQ_WARM_RST
rtc_en_pwr_vbat_det	0x0d0	Enable state machine reference PWR_VBAT_DET
FSM_STATE	0x0d4	RTC state machine value
rtc_en_wdg_rst_req	0x0e0	Enable REQ_WDG_RST
RTC_EN_SUSPEND_REQ	0x0e4	Enable REQ_SUSPEND
rtc_db_req_wdg_rst	0x0e8	REQ_WDG_RST De-jitter time
RTC_DB_REQ_SUSPEND	0x0ec	REQ_SUSPEND De-jitter time
RTC_PG_REG	0x0f0	Power Good Register
RTC_ST_ON_REASON	0x0f8	Power-Up Status Register
rtc_st_off_reason	0x0fc	Power-down status register
rtc_en_wakeup_req	0x120	Enable REQ_WAKEUP
RTC_PWR_WAKEUP_POLARITY	0x128	Select PWR_WAKEUP Low level
RTC_DB_SEL_REQ	0x130	Selecting the de-jittering mode
RTC_PWR_DET_SEL	0x140	Selecting a Low Voltage Detection Signal Source

An overview of the RTC_MACRO_REG register is shown in Table 12-3.

Table *12-2 RTC_MACRO_REG* Register Overview *(*Base Address *0x05026400)*

Name	Address Offset	Description
RTC_PWR_DET_COMP	0x044	Low Voltage Detection Control
rtc_macro_da_clear_all	0x080	DA_CLEAR_ALL
rtc_macro_da_set_all	0x084	DA_SEL_ALL
rtc_macro_da_latch_pass	0x088	DA_LATCH_PASS
rtc_macro_da_soc_ready	0x08c	DA_SOC_READY
RTC_MACRO_PD_SLDO	0x090	PD_SLDO
rtc_macro_rg_defd	0x094	RG_DEFD
RTC_MACRO_RG_SET_T	0x098	RG_SET_T
rtc_macro_ro_clk_stop	0x0a0	RO_CLK_STOP
RTC_MACRO_RO_DEFQ	0x0a4	RO_DEFQ
RTC_MACRO_RO_T	0x0a8	RO_T

An overview of the RTC_CTRL register is shown in Table 12-3.

Table *12-3 RTC_CTRL* Register Overview *(*Base Address *0x05025000)*

Name	Address Offset	Description
rtc_ctrl_unlockkey	0x004	rtc_ctrl_unlockkey
rtc_ctrl0	0x008	rtc_ctrl0
rtc_ctrl_status0	0x00c	rtc_ctrl_status0
rtc_ctrl_status1	0x010	rtc_ctrl_status1
rtc_ctrl_status2gpio	0x014	rtc_ctrl_status2gpio
rtcsys_rst_ctrl	0x018	rtcsys_rst_ctrl
rtcsys_clkmux	0x01c	rtcsys_clkmux
rtcsys_mcu51_ctrl0	0x020	rtcsys_mcu51_ctrl0
rtcsys_mcu51_ctrl1	0x024	rtcsys_mcu51_ctrl1
rtcsys_pmu	0x028	rtcsys_pmu
rtcsys_status	0x02c	rtcsys_status
rtcsys_clkbyp	0x030	rtcsys_clkbyp
rtcsys_clk_en	0x034	rtcsys_clk_en
rtcsys_wkup_ctrl	0x038	rtcsys_wkup_ctrl
rtcsys_clkdiv	0x03c	rtcsys_clkdiv
fc_coarse_en	0x040	fc_coarse_en
fc_coarse_cal	0x044	fc_coarse_cal
fc_fine_en	0x048	fc_fine_en
fc_fine_period	0x04c	fc_fine_period
fc_fine_cal	0x050	fc_fine_cal
rtcsys_pmu2	0x054	rtcsys_pmu2
rtcsys_clkdiv1	0x058	rtcsys_clkdiv1
rtcsys_mcu51_dbg	0x05c	rtcsys_mcu51_dbg
sw_reg0	0x060	sw_reg0
sw_reg1_por	0x064	sw_reg1_por
fab_lp_ctrl	0x068	fab_lp_ctrl
fab_option	0x06c	fab_option
rtcsys_mcu51_ictrl1	0x07c	rtcsys_mcu51_ictrl1
rtc_ip_pwr_req	0x080	rtc_ip_pwr_req
rtc_ip_iso_ctrl	0x084	rtc_ip_iso_ctrl

Name	Address Offset	Description
rtcsys_spare_reg0	0x088	rtcsys_spare_reg0
rtcsys_spare_reg1	0x08c	rtcsys_spare_reg1
rtcsys_spare_ro	0x090	rtcsys_spare_ro
rtcsys_wkup_ctrl1	0x094	rtcsys_wkup_ctrl1
rtcsys_sram_ctrl	0x098	rtcsys_sram_ctrl
rtcsys_io_ctrl	0x09c	rtcsys_io_ctrl
rtcsys_wdt_ctrl	0x0a0	rtcsys_wdt_ctrl
rtcsys_irrx_clk_ctrl	0x0a4	rtcsys_irrx_clk_ctrl
rtcsys_rtc_wkup_ctrl	0x0a8	rtcsys_rtc_wkup_ctrl
rtcsys_por_rst_ctrl	0x0ac	rtcsys_por_rst_ctrl

12.6 RTC Register Description

12.6.1 RTC_CORE_REG

RTC_ANA_CALIB

Offset Address: 0x000

Bits	Name	Access	Description	Reset
15:0	RTC_ANA_CALIB	R/W	Adjusting the Analog Module 32K Oscillator Frequency	0x100
17:16	RTC_ANA_ISEL	R/W	Adjust Analog Module 32K XTAL Current 00 = 2uA, 01 = 1.5uA, 11 = 0.5uA	0x3
30:18	Reserved			
31	RTC_ANA_SEL_FTUNE	R/W	Selecting a 32K OSC Calibration Value Source 0 = controlled by register RTC_ANA_CALIB 1 = controlled by hardware circuitry	0x1

RTC_SEC_PULSE_GEN

Offset Address: 0x004

Bits	Name	Access	Description	Reset
7:0	RTC_SEC_PULSE_GEN_FRAC	R/W	Seconds pulse generator fractional part	0x0
23:8	rtc_sec_pulse_gen_int	R/W	Seconds pulse generator integer part When the counter increment value is greater than the integer bit value, the production Generates a second pulse signal to add 1 to the second counter.	0x8000
30:24	Reserved			

31	RTC_SEL_SEC_PULSE	R/W	Selecting the source of the second pulse signal 0 = second pulse signal generated internally 1 = second pulse signal generated by external hardware circuitry When set to 1, register RTC_SEL_PULSE_GEN_FRAC & RTC_SEL_PULSE_GEN_INT has no effect.	0x1

RTC_ALARM_TIME

Offset Address: 0x008

Bits	Name	Access	Description	Reset
31:0	RTC_ALARM_TIME	R/W	Set timed alarm time	0xffffffff

RTC_ALARM_ENABLE

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
0	RTC_ALARM_ENABLE	R/W	Alarm Enable Set 1 to enable alarms, 0 to disable or clear alarms. Alarm interrupt status.	0x0
31:1	Reserved			

RTC_SET_SEC_CNTR_VALUE

Offset Address: 0x010

Bits	Name	Access	Description	Reset
31:0	RTC_SET_SEC_CNTR_VALUE	R/W	Setting the second counter value	0x0

RTC_SET_SEC_CNTR_TRIG

Offset Address: 0x014

Bits	Name	Access	Description	Reset
0	RTC_SET_SEC_CNTR_TRIG	W1C	Load seconds counter value Set 1 for RTC_SET_SEC_CNTR_VALUE. The setting value takes effect, and the register will be cleared automatically after writing 1. is 0.	
31:1	Reserved			

RTC_SEC_CNTR_VALUE

Offset Address: 0x018

Bits	Name	Access	Description	Reset
31:0	RTC_SEC_CNTR_VALUE	RO	Read the current second counter value	

RTC_INFO0

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
31:0	RTC_INFO0	R/W	Information register 0	0xABCD 1234

RTC_INFO1

Offset Address: 0x020

Bits	Name	Access	Description	Reset
31:0	RTC_INFO1	R/W	Information register 1	0xDEAD BEEF

RTC_INFO2

Offset Address: 0x024

Bits	Name	Access	Description	Reset
31:0	RTC_INFO2	R/W	Message register 2	0xABCD 1234

RTC_INFO3

Offset Address: 0x028

Bits	Name	Access	Description	Reset

31:0	RTC_INFO3	R/W	Information register 3	0xDEAD BEEF

RTC_NOPOR_INFO0

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
31:0	RTC_NOPOR_INFO0	R/W	No reset information register 0	Random

RTC_NOPOR_INFO1

Offset Address: 0x030

Bits	Name	Access	Description	Reset
31:0	RTC_NOPOR_INFO1	R/W	No reset information register 1	Random

RTC_NOPOR_INFO2

Offset Address: 0x034

Bits	Name	Access	Description	Reset
31:0	RTC_NOPOR_INFO2	R/W	No reset information register 2	Random

RTC_NOPOR_INFO3

Offset Address: 0x038

Bits	Name	Access	Description	Reset
31:0	RTC_NOPOR_INFO3	R/W	No reset information register 3	Random

RTC_APB_BUSY_SEL

Offset Address: 0x03c

Bits	Name	Access	Description	Reset
3:0	Reserved	R/W		
4	rtc_apb_32k_busy_sel	R/W	Selects the rtc pclk busy signal source (when the busy (keep the pclk at full speed when you are) 0 = pclk busy signal generated by hardware circuitry 1 = pclk busy Signal by register rtc_apb_32k_force_busy control	0x0
7:5	Reserved			
8	rtc_apb_32k_force_busy	R/W	1 = pclk always operates at full speed 0 = pclk returns to full speed only at psel	0x0
31:9	Reserved			

rtc_db_pwr_vbat_det

Offset Address: 0x040

Bits	Name	Access	Description	Reset
15:0	rtc_db_pwr_vbat_det	R/W	PWR_VBAT_DET De-jittering time (unit: 0.5 seconds, 0.5 seconds, 0.5 seconds) 32K clock)	0x2
31:16	Reserved			

RTC_DB_BUTTON1

Offset Address: 0x048

Bits	Name	Access	Description	Reset
15:0	RTC_DB_BUTTON1	R/W	PWR_BUTTON1 De-dithering time (Unit: 32K) Clock) The default value of 0x100 is approximately 8ms.	0x100
31:16	Reserved			

RTC_DB_PWR_ON

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
15:0	RTC_DB_PWR_ON	R/W	PWR_ON De-jitter time (unit: 32K clocks)	0x100
31:16	Reserved			

RTC_7SEC_RESET

Offset Address: 0x050

Bits	Name	Access	Description	Reset
7:0	RTC_7SEC_RESET	R/W	Press and hold PWR_BUTTON1 to reset the jitter time. (Unit: seconds) This register will only be POR'd.	0x7
15:8	Reserved			
31:16	RTC_7SEC_UNLOCK_KEY	WO	Write 0xDC78 at the same time to release RTC_7SEC_RESET Anti-write	0x0

rtc_thm_shdn_auto_reboot

Offset Address: 0x064

Bits	Name	Access	Description	Reset
0	rtc_thm_shdn_auto_reboot	R/W	Select when REQ_THM_SHDN is received: 0 = Initiate power-down process 1 = Start Power-cycle process (power down then power up) Electricity)	0x0
31:1	Reserved			

RTC_POR_DB_MAGIC_KEY

Offset Address: 0x068

Bits	Name	Access	Description	Reset
15:0	RTC_POR_DB_MAGIC_KEY	R/W	Writing 0x5AF0 causes the POR to de-jitter (approximately). 1ms)	Random
31:16	Reserved			

RTC_DB_SEL_PWR

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
0	Reserved			
1	DB_SEL_PWR_BUTTON1	R/W	Select PWR_BUTTON1 de-dithering mode 0 = state machine triggered by falling edge of PWR_BUTTON1 de-jitter signal 1 = state machine uses PWR_BUTTON1 to de-dither Signal low level trigger	0x1
2	DB_SEL_PWR_ON	R/W	Select PWR_ON de-jitter mode 0 = state machine triggered by rising edge of PWR_ON de-jitter signal 1 = PWR_ON de-jittering signal is used by the state machine High Level Trigger	0x1

3	DB_SEL_PWR_WAKEUP0	R/W	Select PWR_WAKEUP0 de-jittering mode 0 = state machine triggered by PWR_WAKEUP0 de-jittering signal rising edge 1 = state machine uses PWR_WAKEUP0 for de-jittering Triggered by high level of dynamic signal	0x1
4	DB_SEL_PWR_WAKEUP1	R/W	Select PWR_WAKEUP1 de-jittering mode 0 = state machine triggered by PWR_WAKEUP1 de-jittering signal rising edge 1 = state machine uses PWR_WAKEUP1 for de-jittering. Triggered by high level of dynamic signal	0x1
31:5	Reserved			

RTC_UP_SEQ0

Offset Address: 0x070

Bits	Name	Access	Description	Reset
15:0	RTC_UP_SEQ0	R/W	Power-on process When PWR_SEQ0 output goes from 0 to 1 Time (Unit: 32K clocks)	0x0
31:16	Reserved			

RTC_UP_SEQ1

Offset Address: 0x074

Bits	Name	Access	Description	Reset
15:0	RTC_UP_SEQ1	R/W	Power-on process When PWR_SEQ1 output turns from 0 to 1 Time (unit: 32K clocks)	0x40
31:16	Reserved			

RTC_UP_SEQ2

Offset Address: 0x078

Bits	Name	Access	Description	Reset
15:0	RTC_UP_SEQ2	R/W	Power-on process When PWR_SEQ2 output turns from 0 to 1 Time (Unit: 32K clocks)	0x80
31:16	Reserved			

RTC_UP_SEQ3

Offset Address: 0x07c

Bits	Name	Access	Description	Reset
15:0	RTC_UP_SEQ3	R/W	Power-on process When PWR_SEQ3 output goes from 0 to 1 Time (unit: 32K clocks)	0xc0
31:16	Reserved			

RTC_UP_IF_EN

Offset Address: 0x080

Bits	Name	Access	Description	Reset
15:0	RTC_UP_IF_EN	R/W	Uplink process to release the downlink zone ISO time (unit. 32K clock)	0x100
31:16	Reserved			

RTC_UP_RSTN

Offset Address: 0x084

Bits	Name	Access	Description	Reset
15:0	RTC_UP_RSTN	R/W	Power-on process system reset release time (unit: 32K) Clock)	0x140
31:16	Reserved			

RTC_UP_MAX

Offset Address: 0x088

Bits	Name	Access	Description	Reset

15:0	RTC_UP_MAX	R/W	Complete power-up process completion time (unit: 32K clocks) RTC_UP_SEQ0~RTC_UP_MAX are the absolute timing of each stage of the power-up process, and it is recommended to use the default timing of RTC_UP_SEQ0~RTC_UP_MAX. Recognized Value.	0x180
31:16	Reserved			

RTC_DN_SEQ0

Offset Address: 0x090

Bits	Name	Access	Description	Reset
15:0	RTC_DN_SEQ0	R/W	Power-down process When PWR_SEQ0 output goes from 1 to 0 Time (Unit: 32K clocks)	0x140
31:16	Reserved			

RTC_DN_SEQ1

Offset Address: 0x094

Bits	Name	Access	Description	Reset
15:0	RTC_DN_SEQ1	R/W	Power-down process When PWR_SEQ1 output goes from 1 to 0 Time (Unit: 32K clocks)	0x100
31:16	Reserved			

RTC_DN_SEQ2

Offset Address: 0x098

Bits	Name	Access	Description	Reset
15:0	RTC_DN_SEQ2	R/W	Power-down process When PWR_SEQ2 output goes from 1 to 0 Time (Unit: 32K clocks)	0xc0
31:16	Reserved			

RTC_DN_SEQ3

Offset Address: 0x09c

Bits	Name	Access	Description	Reset
15:0	RTC_DN_SEQ3	R/W	Power-down process When PWR_SEQ3 output goes from 1 to 0 Time (Unit: 32K clocks)	0x80
31:16	Reserved			

RTC_DN_IF_EN

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
15:0	RTC_DN_IF_EN	R/W	Power-down process opens the power-down zone signal ISO time (single) Bit: 32K clock)	0x40
31:16	Reserved			

RTC_DN_RSTN

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
15:0	RTC_DN_RSTN	R/W	System reset time for power-down process (unit: 32K) Clock)	0x0
31:16	Reserved			

RTC_DN_MAX

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset

15:0	RTC_DN_MAX	R/W	Complete power-down or hibernation process completion time (unit: 32K clocks) RTC_DN_SEQ0~RTC_DN_MAX are the absolute timing of each stage of the power-down process, and are recommended to be used. Default Value.	0x180
31:16	Reserved			

RTC_PWR_CYC_MAX

Offset Address: 0x0b0

Bits	Name	Access	Description	Reset
15:0	RTC_PWR_CYC_MAX	R/W	Complete Power-cycle Process Completion Time (Unit: 32K clocks) Power-cycle time includes the complete power down process plus Power-up process.	0x4000
31:16	Reserved			

RTC_WARM_RST_MAX

Offset Address: 0x0b4

Bits	Name	Access	Description	Reset
15:0	RTC_WARM_RST_MAX	R/W	Full WARM_RESET process completion time (unit: 32K clocks) Equivalent to the system reset low time.	0x40
31:16	Reserved			

RTC_EN_7SEC_RST

Offset Address: 0x0b8

Bits	Name	Access	Description	Reset
0	RTC_EN_7SEC_RST	R/W	Enable long press PWR_BUTTON1 to trigger for 7 seconds. RTC forced reset	0x0
1	RTC_7SEC_RST_MODE	R/W	7 seconds forced reset mode 0 = low level mode, 1 = short pulse mode When 7 seconds of forced reset occurs, choose to generate a short-pulse reset signal or hold reset until released. PWR_BUTTON1 Press the button.	0x0
2	DB_SEL_PWR_BUTTON1_7SEC	R/W	0 = Reset the reset counter for 7 seconds if the PWR_BUTTON1 button has been triggered during the power-up process. 1 = NOP	0x0
3	SEL_7SEC_RST_RTCSYS	R/W	0 = 7 sec forced reset signal will reset RTC subsystem sec reset will not reset rtcsys 1 = RTC subsystem will not be reset	0x1
15:4	Reserved			
31:16	RTC_EN_7SEC_UNLOCK_KEY	WO	Write 0xDC78 at the same time to disable [3:0] write protection.	0x0

rtc_en_pwr_wakeup

Offset Address: 0x0bc

Bits	Name	Access	Description	Reset
6:0	rtc_en_pwr_wakeup	R/W	Setting the source that triggers a wake from sleep state 0 = Wakeup cannot be triggered 1 = Wake-up can be triggered [0] = PWR_WAKEUP0 [1] = PWR_WAKEUP1 [2] = PWR_ON [3] = REQ_POWERUP [4] = PWR_BUTTON1 [5] = Alarm [6] = REQ_WAKEUP	0x0
7	Reserved			

Bits	Name	Access	Description	Reset
14:8	RTC_EN_PWR_UP	R/W	Setting the source that triggers power-up and power-on 0 = No power-up can be triggered 1 = can trigger power-up [8] = PWR_WAKEUP0 [9] = PWR_WAKEUP1 [10] = PWR_ON [11] = REQ_POWERUP [12] = PWR_BUTTON1 [13] = Alarm [14] = REQ_WAKEUP	0x14
31:15	Reserved			

RTC_EN_SHDN_REQ

Offset Address: 0x0c0

Bits	Name	Access	Description	Reset
0	RTC_EN_SHDN_REQ	R/W	Enable software request for power down (REQ_SHDN) 0 = disable, 1 = enable	0x0
31:1	Reserved			

RTC_EN_THM_SHDN

Offset Address: 0x0c4

Bits	Name	Access	Description	Reset
0	RTC_EN_THM_SHDN	R/W	Enable to request an overheat power down or restart (REQ_THM_SHDN) 0 = disable, 1 = enable	0x0
31:1	Reserved			

rtc_en_pwr_cyc_req

Offset Address: 0x0c8

Bits	Name	Access	Description	Reset
0	rtc_en_pwr_cyc_req	R/W	Enable Request Power-cycle (REQ_PWR_CYC) 0 = disable, 1 = enable	0x0
31:1	Reserved			

RTC_EN_WARM_RST_REQ

Offset Address: 0x0cc

Bits	Name	Access	Description	Reset
0	RTC_EN_WARM_RST_REQ	R/W	Enable to request a soft reboot of the system (REQ_WARM_RST) 0 = disable, 1 = enable	0x0
31:1	Reserved			

rtc_en_pwr_vbat_det

Offset Address: 0x0d0

Bits	Name	Access	Description	Reset

0	rtc_en_pwr_vbat_det_up	R/W	Enable state machine reference battery low voltage detection state (PWR_VBAT_DET) 0 = disable, 1 = enable If this value is set to 1, when any key wants to trigger power on or wake up, the state machine will judge the value of the low voltage detection output, if the low voltage detection output is low level (power), the state machine will judge the value of the low voltage detection output. battery voltage is too low or there is no power supply), then the RTC state machine	0x1

Bits	Name	Access	Description	Reset
			Maintain the current status unchanged.	
1	rtc_en_pwr_vbat_det_dn	R/W	Enable battery low voltage state to trigger power down 0 = disable, 1 = enable If this value is set to 1, when the chip is in normal power-up or has entered hibernation, the RTC state machine will judge the value of the low-voltage detection output, and if the low-voltage detection output turns from a high level to a low level (the battery voltage is too low or the power supply is off), the state machine will be triggered to execute the next current Program.	0x1
2	RTC_EN_AUTO_POWER_UP	R/W	Enable the RTC status machine to automatically enter the power-up state. 1 = when entering power-down PWR_VBAT_DET is a high level, it will power up automatically. 0 = stay in this state when entering power-down State until any power source is triggered.	0x1
31:3	Reserved			

FSM_STATE

Offset Address: 0x0d4

Bits	Name	Access	Description	Reset
3:0	FSM_STATE	RO	RTC state machine value 4'h0 = ST_OFF (System power down complete) 4'h1 = ST_UP (power-up process in progress) 4'h2 = ST_DN (power down process in progress) 4'h3 = ST_ON (system power-up complete) 4'h4 = ST_PWR_CYC2 (end of power-cycle) 4'h6 = ST_PWR_CYC (Power-cycle down) 4'h7 = ST_WARM_RESET (system reset in progress) 4'h9 = ST_SUSP (System has been put to sleep) 4'hB = ST_PRE_SUSP (Hibernate power down process) (in rows)	
31:4	Reserved			

rtc_en_wdg_rst_req

Offset Address: 0x0e0

Bits	Name	Access	Description	Reset
0	rtc_en_wdg_rst_req	R/W	Enable Watchdog to request a system reset (REQ_WDG_RST) 0 = disable, 1 = enable	0x0
1	rtc_en_sus_wdg_rst_req	R/W	Enable Watchdog to request a system reset when in sleep state. 0 = disable, 1 = enable	0x1
31:2	Reserved			

RTC_EN_SUSPEND_REQ

Offset Address: 0x0e4

Bits	Name	Access	Description	Reset
0	RTC_EN_SUSPEND_REQ	R/W	Enable request to sleep (REQ_SUSPEND) 0 = disable, 1 = enable	0x0

Bits	Name	Access	Description	Reset
31:1	Reserved			

RTC_PG_REG

Offset Address: 0x0f0

Bits	Name	Access	Description	Reset
3:0	RTC_PG_REG	R/W	Chip Power Good Status 1 = Chip has power (Power Good), IO signal can be passed through 0 = the chip is powered down and the IO signals remain in the staging state (retent) [0] = Control DDR IO [3:1] = reserved This register signal is used to control whether the IO interface between the chip and DDR is normal or retent. When the system is going to sleep, the software must set this register to 0 to keep the DDR IO in a fixed state. When the system wakes up from hibernation, the value of this register must be set to 1 to let the DDR return to normal operation. When the system enters the power-down state, this register will be automatically cleared to the full value. 1.	0xF
31:4	Reserved			

RTC_ST_ON_REASON

Offset Address: 0x0f8

Bits	Name	Access	Description	Reset
3:0	st_on_reason_last_state	RO	The RTC state machine returns to power-up completion from the following states (ST_ON) 4'h0 = from power down (ST_OFF) back to power up 4'h3 = from power up to Power-cycle or Warm-reset and back to power up again 4'h9 = return from sleep to power-up When the system is rebooted, the software can read this mailbox by The memory determines the reason for the chip powering up.	
15:4	Reserved			

31:16	st_on_reason_last_input	RO	Trigger reason for the state machine to return to the power-up state (record signal state value) [0] = PWR_VBAT_DET (0: trigger power down) [1] = PWR_ON (1: key-triggered power-up) [2] = RTC_EN_AUTO_POWER_UP [3] = PWR_BUTTON1 (0: key-triggered power-up) [4] = PWR_BUTTON1_7SEC [5] = PWR_WAKEUP0 (1: key trigger recall) Wake up) [6] = PWR_WAKEUP1 (1: key trigger call) Wake up) [7] = Alarm (1: timed alarm occurs) [8] = REQ_PWR_CYC (1: software trigger) Power-cycle) [9] = REQ_THM_SHDN (1: under software trigger) (electricity/power-cycle) [10] = REQ_WARM_RST (1: software trigger) reset) [11] = REQ_WDG_RST (1: Watchdog contact)	

Bits	Name	Access	Description	Reset
			(reset) [12] = REQ_SHDN (1: software triggered power down) [13] = REQ_SUSPEND (1: software triggered rest) sleep) [14] = REQ_WAKEUP (1: event triggered wakeup) [15] = REQ_POWERUP	

rtc_st_off_reason

Offset Address: 0x0fc

Bits	Name	Access	Description	Reset
3:0	st_off_reason_last_state	RO	The RTC state machine had entered power-down completion (ST_OFF) from the following state 4'h3 = from power up (ST_ON) to power down 4'h9 = from hibernation (ST_SUSP) into power down other = 7 seconds forced reset occurs When the system is rebooted, the software can read this mailbox by The memory determines the cause of the chip's previous entry into the power down.	
15:4	Reserved			
31:16	st_off_reason_last_input	RO	Trigger reason for the state machine to have entered a power-down state (recording the signal state value) [14:0] with ST_ON_REASON_LAST_INPUT equal [15] = 0: 7 seconds of mandatory reset have occurred	

rtc_en_wakeup_req

Offset Address: 0x120

Bits	Name	Access	Description	Reset
0	rtc_en_wakeup_req	R/W	Enable event request to wake up from sleep state 0 = disable, 1 = enable	0x0
1	RTC_EN_POWERUP_REQ	R/W	Enable event request upload 0 = disable, 1 = enable	0x0
31:2	Reserved			

RTC_PWR_WAKEUP_POLARITY

Offset Address: 0x128

Bits	Name	Access	Description	Reset
0	PWR_WAKEUP0_POLARITY	R/W	Select PWR_WAKEUP0 Valid Polarity 1 = High level active 0 = Low level active	0x1

1	PWR_WAKEUP1_POLARITY	R/W	Select PWR_WAKEUP1 Valid Polarity 1 = High level active 0 = Low level active	0x1
31:2	Reserved			

RTC_DB_SEL_REQ

Offset Address: 0x130

Bits	Name	Access	Description	Reset
0	DB_SEL_REQ_SHDN	R/W	Select software signal REQ_SHDN De-jitter mode 0 = triggered by rising edge of register value 1 = triggered by register pulse signal	0x1
1	DB_SEL_REQ_THM_SHDN	R/W	Selection signal REQ_THM_SHDN De-jitter mode	0x1

Bits	Name	Access	Description	Reset
			style 0 = triggered by signal high 1 = untransmitted by the rising edge of the signal	
2	DB_SEL_REQ_PWR_CYC	R/W	Select software signal REQ_PWR_CYC De-jittering mode 0 = triggered by rising edge of register value 1 = triggered by register pulse signal	0x1
3	DB_SEL_REQ_WARM_RST	R/W	Select software signal REQ_WARM_RST De-jitter mode 0 = triggered by rising edge of register value 1 = triggered by register pulse signal	0x1
4	DB_SEL_REQ_WDG_RST	R/W	Selection signal REQ_WDG_RST De-jitter mode 0 = triggered by signal high 1 = untransmitted by the rising edge of the signal	0x1
5	DB_SEL_REQ_SUSPEND	R/W	Selects the software signal REQ_SUSPEND de-jittering mode 0 = triggered by rising edge of register value 1 = triggered by register pulse signal	0x1
6	DB_SEL_REQ_WAKEUP	R/W	Selection signal REQ_WAKEUP De-jitter mode 0 = triggered by signal high 1 = untransmitted by the rising edge of the signal	0x1
7	DB_SEL_REQ_POWERUP	R/W	Selection signal REQ_POWERUP De-jitter mode 0 = triggered by signal high 1 = untransmitted by the rising edge of the signal	0x1
31:8	Reserved			

RTC_PWR_DET_SEL

Offset Address: 0x140

Bits	Name	Access	Description	Reset
0	pwr_det_o_sel_comp	R/W	Selection of status signal output source for low voltage detection 0 = directly from IO PWR_VBAT_DET 1 = Output low voltage detection status value from analog low voltage detection circuit can be determined by register RTC_CTRL_STATUS0[0] readout	0x0
1	pwr_det_i_sel_comp	R/W	Select RTC state machine low voltage trigger down signal source 0 = directly from IO PWR_VBAT_DET 1 = from analog low voltage detection circuit outputs	0x0

31:2	Reserved			

12.6.2 RTC_MACRO_REG

RTC_PWR_DET_COMP

Offset Address: 0x44

Bits	Name	Access	Description	Reset
0	pwr_det_comp_enable	R/W	Enable analog module low voltage detection 1 = enable 0 = disable	0x0

Bits	Name	Access	Description	Reset
7:1	Reserved			
12:8	pwr_det_comp_sel	R/W	Setting the low voltage detection voltage comparison threshold Threshold = 1.20V + (pwr_det_comp_sel * 12.5mV)	0xf
31:13	Reserved			

rtc_macro_da_clear_all

Offset Address: 0x080

Bits	Name	Access	Description	Reset
0	DA_CLEAR_ALL	R/W		0x0
31:1	Reserved			

rtc_macro_da_set_all

Offset Address: 0x084

Bits	Name	Access	Description	Reset
0	DA_SEL_ALL	R/W		0x0
31:1	Reserved			

rtc_macro_da_latch_pass

Offset Address: 0x088

Bits	Name	Access	Description	Reset
0	DA_LATCH_PASS	R/W		0x0
31:1	Reserved			

rtc_macro_da_soc_ready

Offset Address: 0x08c

Bits	Name	Access	Description	Reset
0	DA_SOC_READY	R/W		0x0
31:1	Reserved			

RTC_MACRO_PD_SLDO

Offset Address: 0x090

Bits	Name	Access	Description	Reset
0	PD_SLDO	R/W		0x0
31:1	Reserved			

rtc_macro_rg_defd

Offset Address: 0x094

Bits	Name	Access	Description	Reset
15:0	RG_DEFD	R/W		0x7fff
31:16	Reserved			

RTC_MACRO_RG_SET_T

Offset Address: 0x098

Bits	Name	Access	Description	Reset
31:0	RG_SET_T	R/W		0x0

rtc_macro_ro_clk_stop

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
0	RO_CLK_STOP	RO		
31:1	Reserved			

RTC_MACRO_RO_DEFQ

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
15:0	RO_DEFQ	RO		
31:16	Reserved			

RTC_MACRO_RO_T

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
31:0	RO_T	RO		

12.6.3 RTC_CTRL

RTC_CTRL0_UNLOCKKEY

Offset Address: 0x004

Bits	Name	Access	Description	Reset
15:0	rtc_ctrl0_unlockkey	R/W	Before configuring register RTC_CTRL0, you must write 0xAB18 to this register to unlock write protection. If unlockkey_clear is set to 1, the value of this register will be cleared to 0 after a write action to RTC_CTRL0, and the value of RTC_CTRL0 will be returned to 1. Repeat anti-writing.	0x0000
31:16	Reserved			

RTC_CTRL0

Offset Address: 0x008

Bits	Name	Access	Description	Reset
0	req_shdn	R/W	Requesting a call. 0 = no action, 1 = request to RTC Register RTC_EN_SHDN_REQ must be set first. is 1 to be effective.	0x0
1	req_sw_thm_shdn	R/W	Software mode requests overheating power down or reboot 0 = no action, 1 = request to RTC Register RTC_EN_THM_SHDN must be set first. is 1 to be effective.	0x0
2	hw_thm_shdn_en	R/W	Enable hardware mode to request overheat power down or reboot 0 = diable, 1 = enable	0x0

3	req_pwr_cyc	R/W	Request Power-cycle 0 = no action, 1 = request to RTC Register RTC_EN_PWR_CYC_REQ must be first Set to 1 to be effective.	0x0
4	req_warm_rst	R/W	Request Warm-reset 0 = no action, 1 = request to RTC Register RTC_EN_WARM_RST_REQ must be Set it to 1 first to make it effective.	0x0

Bits	Name	Access	Description	Reset
5	req_sw_wdg_rst	R/W	Software Mode Request Watchdog Reset 0 = no action, 1 = request to RTC Register RTC_EN_WDG_RST_REQ Must be Set it to 1 first to make it effective.	0x0
6	hw_wdg_rst_en	R/W	Enable hardware mode request Watchdog reset 0 = diable, 1 = enable	0x0
7	req_suspend	R/W	Request to sleep 0 = no action, 1 = request to RTC Register RTC_EN_SUSPEND_REQ must be first Set to 1 to be effective.	0x0
8	unlockkey_clear	R/W	Enable auto clear register unlock	0x0
9	Reserved			
10	reg_rtc_mode	R/W	32K Clock Source 0 = OSC32K, 1 = XTAL32K	0x0
11	reg_clk32k_cg_en	R/W	32K Clock Switch 0 = off, 1 = on	0x1
31:12	Reserved			

RTC_CTRL_STATUS0

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
0	rtc_pwr_vbat_det_o	RO	Low voltage detection status signal output	
1	rtc_pwr_button0_o	RO	PWR_BUTTON0 IO signal output	
2	rtc_pwr_button1_o	RO	PWR_BUTTON1 IO Signal Output	
3	Reserved			
4	rtc_pwr_on_o	RO	PWR_ON IO signal output	
5	rtc_pwr_wakeup0_o	RO	PWR_WAKEUP0 IO signal output	
6	rtc_pwr_wakeup1_o	RO	PWR_WAKEUP1 IO signal output	
7	rtc_mode_o	RO	RTC_MODE IO Signal Output	
20:8	Reserved			
21	rtc_alarm_o	RO	Alarm status	
22	hw_thm_shdn_sta_i	RO	Thermal restart status signal	
23	hw_wdg_rst_sta_i	RO	Watchdog reset status signal	
24	sys_reset_x_i	RO		
25	cg_en_out_clk_32k	RO		
29:26	rtc_fsm_st	RO	RTC state machine value	
31:30	Reserved			

RTC_CTRL_STATUS1

Offset Address: 0x010

Bits	Name	Access	Description	Reset
31:0	rtc_sec_value_o	RO	RTC Second Counter Value	

rtc_ctrl_status2gpio

Offset Address: 0x014

Bits	Name	Access	Description	Reset
7:0	status2gpio_en	R/W		0x0

Bits	Name	Access	Description	Reset
31:8	Reserved			

rtcsys_rst_ctrl

Offset Address: 0x018

Bits	Name	Access	Description	Reset
0	Reserved			
1	reg_soft_rstn_mcu	R/W	0 : reest MCU	0x0
2	reg_soft_rstn_sdio	R/W	0 : reset SD1	0x1
3	reg_soft_rstn_uart	R/W	0 : reset Uart	0x1
4	reg_soft_rstn_spinor	R/W	0 : reset spinor1	0x1
5	reg_soft_rstn_ictl	R/W	0 : reset dw_ictl	0x1
6	reg_soft_rstn_mbox	R/W	0 : reset mbox	0x1
7	reg_soft_rstn_fab_hs2rtc	R/W	0 : reset hs2rtc	0x1
8	reg_soft_rstn_fab_rtc2ap	R/W	0 : reset rtc2ap	0x1
9	reg_soft_rstn_fab_sram	R/W	0 : reset ahb sram logic	0x1
10	reg_soft_rstn_apb	R/W	no load	0x1
11	reg_soft_rstn_apb_timer	R/W	0 : reset dw timer apb logic	0x1
12	reg_soft_rstn_timer0	R/W	0 : reset dw timer0	0x1
13	reg_soft_rstn_timer1	R/W	0 : reset dw timer1	0x1
14	reg_soft_rstn_osc	R/W	0 : reset osc	0x1
15	reg_soft_rstn_gpio	R/W	0 : reset gpio	0x1
16	reg_soft_rstn_i2c	R/W	0 : reset i2c	0x1
17	reg_soft_rstn_saradc	R/W	0 : reset saradc	0x1
18	reg_soft_rstn_wdt	R/W	0 : reset wdt	0x1
19	reg_soft_rstn_irrx	R/W	0 : reset irrx	0x1
20	reg_soft_rstn_f32kless	R/W	0: reset f32kless	0x1
31:21	Reserved			

rtcsys_clkmux

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
3:0	reg_sdio_clk_mux	R/W	clk_sd1_pre 0 : fpll/4 1: osc_div	0x0
7:4	reg_fab_clk_mux	R/W	clk_fab_pre 0 : 32K, 1: fpll/5, 2: osc_div	0x0
9:8	reg_timer0_clk_mux	R/W	0: xtal 1: 32K	0x0
11:10	reg_timer1_clk_mux	R/W	0: xtal 1: 32K	0x0
13:12	reg_apb_clk_mux	R/W	00 : cgdiv and refer to apbactive 01 : force clk_apb, clk_fab 1:1 (default) 10 : force clk_apb, clk_fab 1:2 11 : force clk_apb, clk_fab 1:4	0x1
15:14	Reserved			
17:16	reg_i2c_clk_mux	R/W	0: xtal 1: osc div	0x0
19:18	reg_sd_mclk_clk_mux	R/W	0: 100Khz from OSC, 1: 32K	0x0
20	reg_saradc_clk_mux	R/W	0 : XTAL, 1: OSC DIV	0x0
21	reg_irrx_clk_mux	R/W	0 : XTAL, 1: OSC DIV	0x0
31:22	Reserved			

rtcsys_mcu51_ctrl0

Offset Address: 0x020

Bits	Name	Access	Description	Reset
4:0	reg_51_rom_addr_size	R/W	Determines how many of the sixteen internal ROM address bits (irom_addr) are used. are used (0 = no internal ROM present).	0xc
5	reg_51_mem_ea_n	R/W	0 : external rom exist, 1: external rom not exist	0x0
6	reg_51_xdata_mode	R/W	0 : fetch xdata with clock gating 1 : fetch xdata wo clock gating (to support 51 timer and 51 uart)	0x0
7	reg_51_rom_addr_def	R/W	0: mercury define , max internal rom = 2^reg_51_rom_addr_size -1 internal rom offset = 4K*reg_51irom_ioffset 1: marsdefine max internal rom = 2K*reg_51_rom_addr_size -1 internal rom offset = 2K*reg_51irom_ioffset	0x0
10:8	Reserved			
31:11	reg_51xdata_ioffset0	R/W	Set offset address[31:12] to select mcu8051 boot device	0x05200

rtcsys_mcu51_ctrl1

Offset Address: 0x024

Bits	Name	Access	Description	Reset
4:0	reg_51irom_ioffset	R/W	boot rom offset to rtcsys_sram	0x0
5	Reserved			
9:6	reg_51_pf_mode	R/W	reg_51_pf_mode	0x0
10	Reserved			
31:11	reg_51xdata_doffset0	R/W	Set offset address[31:12] to select mcu8051 xdata	0x05200

rtcsys_pmu

Offset Address: 0x028

Bits	Name	Access	Description	Reset
3:0	Reserved			
4	reg_dis_pmu_ldo_ctrl	R/W	disable pmu ldo ctrl 0: enable pmu to ctrl RTC_LDO sleep mode 1: disable pmu to ctrl RTC_LDO sleep mode	0x0
5	reg_wdt_clkoff_by_pmu	R/W	wdt_clk gate by pmu when mcu into idle mode 1. wdt clock gate by pmu	0x0
6	reg_force_osc_off	R/W	1 : force osc off	0x0
7	reg_force_osc_on	R/W	1 : force osc on	0x0
8	reg_pmu_sleep_mode	R/W	pmu enter light sleep mode when mcu idle 1 : enable pmu light sleep mode when mcu idle (pmu control osc_req/ sram slp) 0 disable pmu light sleep mode	0x0
9	reg_pmu_lowpwr_mode	R/W	mcu_pmu into sleep state when rtc at suspend state & mcu idle & reg_pmu_sleep_mode enable 1 : enable mcu_pmu into sleep mode (trigger rtc ldo step down power) 0 disable mcu_pmu sleep mode	0x0

Bits	Name	Access	Description	Reset
13:10	reg_pmu_stable_cnt	R/W	Stable timer when mcu_pmu leave sleep state. clock unit : 31.25us (32khz), wait for 1~16 tick cycle	0x3
14	reg_xtal_off_by_pmu	R/W	pmu control xtal request 1: xtal request disable by pmu sleep mode	0x0
15	reg_rtcsys_clk25m_req	R/W	xtal request1 for rtcsys 0: disable 25m xtal request1(rtcsys) 1: enable 25m xtal request1 (rtcsys)	0x1
19:16	reg_rtc_vbat_det_db_cnt	R/W	vbat det int debounce time (cycle unit. 32K)	0x2
20	reg_rtc_vbat_det_db_en	R/W	0: disable vbat det int debounce 1: enable vbat det int debounce	0x1
21	reg_ahb_sram_auto_slp_en	R/W	1: enable ahb sram into slp md when bus idle	0x0
23:22	reg_ahb_sram_busy_sel	R/W	2'd0: cs | cs_d1 2'd1: cs | cs_d1 | cs_d2 2'd2: cs | cs_d1 | cs_d2 | cs_d3 3'd3: cs | cs_d1 | cs_d2 | cs_d3 | cs_d4	0x0
24	reg_rtc_stint_clr	W1P	clear rtc state change interrupt	
25	reg_vbat_det_int_clr	W1P	clear vbet det interrupt	
26	reg_rtcsys_clk25m_hw_req	R/W	xtal request1 for rtcsys from hw ip 0: disable 25m xtal request1 from hw ip(rtcsys) 1: enable 25m xtal request1 from hw ip(rtcsys)	0x0
27	Reserved			
28	reg_vbat_det_force_clk	R/W	1: when vbat det happen, change rtcsys bus clock to OSC	0x0
29	reg_mcu_clkoff_by_pmu	R/W	mcu_clk gate by pmu when into idle mode 1. mcu clock gate by pmu	0x1
30	reg_xtal_off_by_susp	R/W	ISO off control xtal request 1: xtal request disable by ISO_OFF	0x0
31	reg_osc_off_by_susp	R/W	ISO off control osc request 1: osc request disable by ISO_OFF	0x0

rtcsys_status

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
31:0	reg_rtcsys_status	RO	[0] enable rtc2apb ahb path 0: rtcsys ip can only access 0x05000000+16MB 1: rtcsys ip can access full range address [1] flag of vbat_det_force_clk	

rtcsys_clkbyp

Offset Address: 0x030

Bits	Name	Access	Description	Reset
31:0	reg_clk_byp	R/W	[0] : clk_fab , 0: clk_fab_pre, 1: xtal (default) [1] : clk_sdio, 1: clk_sd1_pre, 1: xtal (default) [31:2]: NA	0xffffffff

rtcsys_clk_en

Offset Address: 0x034

Bits	Name	Access	Description	Reset
31:0	reg_clk_en	R/W	[0] : NA [1] : clk_sd1 (sd1 card clock) [2] : clk_fab_sd1 (sd1 core clock) [3] : clk_mcu [4] : clk_hs2rtc_mst [5] : clk_rtc2ap_slv [6] : clk_spinor1 [7] : clk_fab_sram (AHB sram) [8] : NA [9] : clk_apb_timer [10] : clk_timer0 [11] : clk_timer1 [12] : clk_apb_uart [13] : clk_uart [14] : clk_apb_ictrl [15] : clk_apb_mbox [16] : clk_apb_gpio [17] : clk_apb_osc [18] : clk_gpio_db [19] : clk_apb_i2c [20] : clk_i2c [21] : NA [22] : clk_sd1_tmclk [23] : clk_apb_saradc [24] : clk_saradc [25] : clk_apb_wdt [26] : clk_wdt [27] : clk_irrx [31:28]: NA	0xffffffff

rtcsys_wkup_ctrl

Offset Address: 0x038

Bits	Name	Access	Description	Reset
14:0	reg_rtcsys_wkint_mask	R/W	mask int to RTC_CORE.REQ_WAKEUP/ MCU_PMU [0] : irrrx_intr [1] : gpio_int [2] : timer0_int [3] : timer1_int [4] : saradc_int [5] : rtcsys_ictrl_int [6] : wdt_int [7] : irrx_wakeup	0xff
15	reg_vbat_det_wkup_mask	R/W	1: mask vbat det int	0x1
16	reg_sw_wkint_req	R/W	mcu sw wakeup interrupt to RTC_CORE 1: interrupt active	0x0
23:17	Reserved			
24	reg_wkint2rtc_mask	R/W	1: mask wakeup int (rtcsys int) to RTC core	0x1
31:25	Reserved			

rtcsys_clkdiv

Offset Address: 0x03c

Bits	Name	Access	Description	Reset
3:0	reg_div_clk_osc_fab_div_val	R/W	Clock Divider Factor	0x1
4	reg_div_clk_osc_fab_dis	R/W	Clock gate	0x0
5	reg_div_clk_osc_fab_hwide	R/W	Select High Wide Control (when Divider Factor is odd) 0: Low level of the clock is wider 1: High level of the clock is wider	0x0
15:6	Reserved			
19:16	reg_div_clk_osc_i2c_div_val	R/W	Clock Divider Factor	0x1
20	reg_div_clk_osc_i2c_dis	R/W	Clock gate	0x0
21	reg_div_clk_osc_i2c_hwide	R/W	Select High Wide Control (when Divider Factor is odd) 0: Low level of the clock is wider 1: High level of the clock is wider	0x0
23:22	Reserved			
29:24	reg_div_clk_osc_saradc_div_val	R/W	Clock Divider Factor	0x1
30	reg_div_clk_osc_saradc_dis	R/W	Clock gate	0x0
31	reg_div_clk_osc_saradc_hwide	R/W	Select High Wide Control (when Divider Factor is odd) 0: Low level of the clock is wider 1: High level of the clock is wider	0x0

fc_coarse_en

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	fc_coarse_en	R/W	Enable 32K coarse tuning 0 = disable, 1 = enable	0x0
31:1	Reserved			

fc_coarse_cal

Offset Address: 0x044

Bits	Name	Access	Description	Reset
15:0	fc_coarse_value	RO	32K Coarse Counter Value (Unit: 25MHz clock) The 25MHz clock counts one 32K clock cycle.	
31:16	fc_coarse_time	RO	32K Number of rough adjustments completed	

fc_fine_en

Offset Address: 0x048

Bits	Name	Access	Description	Reset
0	fc_fine_en	R/W	Enable 32K fine tuning 0 = disable, 1 = enable	0x0
31:1	Reserved			

fc_fine_period

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
15:0	fc_fine_period	R/W	32K fine-tuned counting period (unit: 32K clocks) Sets the number of 32K counts to be counted each time the 25MHz clock is used. Clock period.	0x0100
31:16	Reserved			

fc_fine_cal

Offset Address: 0x050

Bits	Name	Access	Description	Reset
23:0	fc_fine_value	RO	32K fine-tuned counter value (unit: 25MHz clock) 25MHz clock count one fc_fine_period	

Bits	Name	Access	Description	Reset
			Cycle.	
31:24	fc_fine_time	RO	32K fine-tuning completions	

rtcsys_pmu2

Offset Address: 0x054

Bits	Name	Access	Description	Reset
0	reg_rtc_sys_wkint_db_en	R/W	PMU wakeup int debounce enable	0x1
4:1	reg_rtc_sys_wkint_db_cnt	R/W	PMU wakeup int debounce cycle (32K)	0x2
31:5	Reserved			

rtcsys_clkdiv1

Offset Address: 0x058

Bits	Name	Access	Description	Reset
15:0	Reserved			
21:16	reg_div_clk_osc_irrx_div_val	R/W	Clock Divider Factor	0x0
22	Reserved			
23	reg_div_clk_osc_irrx_dis	R/W	Clock gate	0x1
31:24	Reserved			

rtcsys_mcu51_dbg

Offset Address: 0x05c

Bits	Name	Access	Description	Reset
3:0	reg_51_dbg_sel	R/W	select mcu51 debug bus (check mcu design review ppt)	0x0
4	reg_51_dbg_snap_shot	W1P	snap shot mcu51 internal register to dbg register (reg_rtcsys_dbg)	
5	reg_51_dbg_step_en	R/W	0: disable mcu debug function 1: enable mcu debug function, and mcu stop at current PC	0x0
6	reg_51_dbg_step	W1P	1: mcu jump to next PC	
7	reg_51_dbg_jump	W1P	1: mcu jump to target pc value (reg_51_dbg_jump2pc)	
15:8	Reserved			
31:16	reg_51_dbg_jump2pc	R/W	16 bit mcu target pc value	0x0

sw_reg0

Offset Address: 0x060

Bits	Name	Access	Description	Reset
7:0	sw_reg0	R/W	reg for SW	0x0
31:8	Reserved			

sw_reg1_por

Offset Address: 0x064

Bits	Name	Access	Description	Reset
7:0	sw_reg1_por	R/W	reg for SW could only be reset by power reset	0x0
31:8	Reserved			

fab_lp_ctrl

Offset Address: 0x068

Bits	Name	Access	Description	Reset
7:0	rtcsys_fab_busy_sel	R/W	select signal to request sys_ctrl to speed up fab clock	0xDF
9:8	rtcsys_fab_busy_ctrl	R/W	rtcsys_fab_busy signal is combi or register out	0x0
11:10	apdbg_busy_ctrl	R/W	apdbg_busy signal is combi or register out	0x0
13:12	reg_apb_busy_ctrl	R/W	apb bridge_busy signal is combi or register out	0x3
15:14	reg_mcu_busy_ctrl	R/W	mcu_busy signal is combi or register out	0x3
31:16	Reserved			

rtcsys_mcu51_ictrl1

Offset Address: 0x07c

Bits	Name	Access	Description	Reset
15:0	reg_51_int1_src_mask	R/W	select rtcsys_int src to mcu int1_n 1: mask, 0: un-mask [0] : vbat_det [1] : mbox0_int [2] : NA [3] : irrx_int [4] : gpio_int [5] : uart_int [6] : spinor1_int [7] : timer0_int [8] : timer1_int [9] : irq_ap2rtc[0] [10] : irq_ap2rtc[1] [11] : i2c_int [12] : rtc_state_change_int [13] : hw_thm_shdn [14] : saradc [15] : wdt_int	0xffff
31:16	reg_51_int1_final_status	R0	mcu int1_n status [0]: vbat_det [1] : mbox0_int [2] : NA [3] : irrx_int [4] : gpio_int [5] : uart_int [6] : spinor1_int [7] : timer0_int [8] : timer1_int [9] : irq_ap2rtc[0] [10] : irq_ap2rtc[1] [11] : i2c_int [12] : rtc_state_change_int [13] : hw_thm_shdn [14] : saradc [15] : wdt_int	

rtc_ip_pwr_req

Offset Address: 0x080

Bits	Name	Access	Description	Reset
0	reg_sd1_pwr_req	R/W	power fence control	0x1

Bits	Name	Access	Description	Reset
			1: power on, 0: power off [0]: sd1	
1	reg_sd1_pwr_req_2nd	R/W	power fence control 1: power on, 0: power off [0]: sd1	0x1
2	reg_mcu_pwr_req	R/W	power fence control 1: power on, 0: power off [1]: mcu subsys	0x1
3	reg_mcu_pwr_req_2nd	R/W	power fence control 1: power on, 0: power off [1]: mcu subsys	0x1
15:4	Reserved			
16	reg_sd1_pwr_ack	R0	power fence power status 1: power on, 0: power off [0]: sd1	
17	reg_sd1_pwr_ack_2nd	R0	power fence power status 1: power on, 0: power off [0]: sd1	
18	reg_mcu_pwr_ack	R0	power fence power status 1: power on, 0: power off [1]: mcu subsys	
19	reg_mcu_pwr_ack_2nd	R0	power fence power status 1: power on, 0: power off [1]: mcu subsys	
31:20	Reserved			

rtc_ip_iso_ctrl

Offset Address: 0x084

Bits	Name	Access	Description	Reset
0	reg_sd1_iso_en	R/W	sd1 iso enablel 1: iso enable, 0: iso disable	0x0
1	reg_mcu_iso_en	R/W	mcu iso enablel 1: iso enable, 0: iso disable	0x0
15:2	Reserved			
17:16	reg_ip_por_en	R/W	1: pwr_island reset assert when power ack is 0	0x3
31:18	Reserved			

rtcsys_wkup_ctrl1

Offset Address: 0x094

Bits	Name	Access	Description	Reset
7:0	reg_rtcsys_wkint_final_status	RO	wkint final status [0] : sd1_wakeup_intr [1] : gpio_int [2] : timer0_int [3] : timer1_int [4] : saradc_int [5] : rtcsys_ictrl_int [6] : NA [7] : NA	
31:8	Reserved			

rtcsys_sram_ctrl

Offset Address: 0x098

Bits	Name	Access	Description	Reset
0	reg_ahb_sram_slp	R/W	1 : ahb sram into sleep mode	0x0
1	reg_ahb_sram_sd	R/W	1 : ahb sram into shut down mode	0x0
2	reg_ahb_sram_ctrl_ov	R/W	0 : ahb sram ctrl by PMU FSM and ahb sram busy 1: sram ctrol by register reg_ahb_sram_slp/reg_ahb_sram_sd	0x1
3	reg_sdio_sram_slp	R/W	1 : sdio sram into sleep mode	0x0
4	reg_sdio_sram_sd	R/W	1 : sdio sram into shut down mode	0x0
5	reg_sdio_sram_ctrl_ov	R/W	0 : sram's sd pin = 1'b0 1: sram ctrol by register reg_sdio_sram_sd	0x1
6	reg_mcu_sram_slp	R/W	1 : mcu iram sram into sleep mode	0x0
7	reg_mcu_sram_sd	R/W	1 : mcu iram sram into shut down mode	0x0
8	reg_mcu_sram_ctrl_ov	R/W	0 : mcu iram sram ctrl by PMU FSM 1: sram ctrol by register reg_ahb_sram_slp/reg_ahb_sram_sd	0x1
9	reg_rtc_sram_slp	R/W	1 : mcu iram sram into sleep mode	0x0
10	reg_rtc_sram_sd	R/W	1 : mcu iram sram into shut down mode	0x0
11	reg_rtc_sram_ctrl_ov	R/W	0 : mcu iram sram ctrl by PMU FSM 1: sram ctrol by register reg_ahb_sram_slp/reg_ahb_sram_sd	0x1
27:12	Reserved			
28	reg_mcu_sram_force_ce	R/W	1: force mcu_iram cs = 1	0x1
31:29	Reserved			

rtcsys_io_ctrl

Offset Address: 0x09c

Bits	Name	Access	Description	Reset
0	reg_i2c_mux_opt0	R/W	0: pwr_gpio6/8 control by dw_gpio 1: pwr_gpio6 is PWR_IIC_SDA pwr_gpio8 is PWR_IIC_SCL	0x0
31:1	Reserved			

rtcsys_wdt_ctrl

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
0	reg_rtc_hw_wdg_rst_en	R/W	0: disable rtc wdt trigger warm reset or pwrcyc reset 1: enable rtc wdt trigger warm reset or pwrcyc reset	0x0
1	reg_rtc_wdt_ctrl_mask_en	R/W	no load	0x1
31:2	Reserved			

rtcsys_irrx_clk_ctrl

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
0	reg_irrx_clk_sw_force_on	R/W	force on clk ctrl of irrx	0x1
1	reg_irrx_xtal_req_en	R/W	enable irrx clk ctrl requet XTAL	0x0
2	reg_irrx_osc_req_en	R/W	enable irrx clk ctrl requet OSC	0x0
3	reg_irrx_ldo_req_en	R/W	enable irrx clk ctrl requet LDO	0x0
7:4	Reserved			
15:8	reg_irrx_xtal_filter_cyc	R/W	irrx xtal filter cycle (default 2ms)	0x40
19:16	reg_irrx_clk_ctrl_st	RO	irrx clock ctrol state	

Bits	Name	Access	Description	Reset
31:20	Reserved			

rtcsys_rtc_wkup_ctrl

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
7:0	reg_rtc_wkint_mask	R/W	wakeup source mask int to RTC_CORE [0]: irrrx_intr [1] : gpio_int [2] : timer0_int [3] : timer1_int [4] : saradc_int [5] : rtcsys_ictrl_int [6] : wdt_int [7] : irrx_wakeup	0xff
15:8	Reserved			
23:16	reg_rtc_puint_mask	R/W	power-up source mask int to RTC_CORE [0]: irrrx_intr [1] : gpio_int [2] : timer0_int [3] : timer1_int [4] : saradc_int [5] : rtcsys_ictrl_int [6] : wdt_int [7] : irrx_wakeup	0xff
31:24	Reserved			

rtcsys_por_rst_ctrl

Offset Address: 0x0ac

Bits	Name	Access	Description	Reset
0	reg_rtcsys_reset_en	R/W	0: not allow rtcsys reset by pwr cyc/ wdt warm reset 1 : allow rtcsys reset by pwr cyc/ wdt warm reset	0x0
1	reg_rtcsys_rstn_src_sel	R/W	select rtcsys rstn src 0: rtc_core fsm (reset with die domain) 1: por_pwr_rstn	0x0
31:2	Reserved			

12.7 Power Management and Low Power Mode

12.8 summarize

The chip supports two main power modes.

a. MCU only (32k-less)
b. Active

12.8.1 MCU only

The chip has a built-in low power MCU. This requires a VDDIO_RTC power supply (VDDIO_SD1 as appropriate). In this No die domain there is a small MCU system with its own clock, timer, uart, i2c, gpio, adc and other peripherals. The MCU is able to replace the external MCU. The MCU is able to wake up the main system back to active after receiving and filtering external inputs.
mode.

The system has a calibrated oscillator that wakes up fast, and falls asleep fast. This further saves on power consumption each time it is awakened.

MCU power consumption is about 100uA (TBD) at wait time.

12.8.2 Active mode.

Active mode is when the chip is fully awake and working. There are other power saving techniques such as dynamic frequency scalling or dynamic clock gating.

12.9 Clock control

12.9.1 Turn off unneeded clock dividers

Refer to the Clock Configuration section to turn off the unused clock divider according to the clock source required by each module. The purpose is to save power consumption.

12.9.2 Adjusting the operating frequency of the module

Select a lower clock source according to the clock specifications required by each module. Configure more frequency divider to reduce the module operating frequency. Note that lowering the frequency of a single module does not necessarily reduce the overall power consumption.

12.9.3 Module-level low-power control

- Analog Modules: MIPI/USB/ETH/AUD related registers are set to disable the unused modules or enter the low power mode.
- Digital Modules: Turn off the clock of unneeded digital modules according

to hardware and specifications.

12.10 Turn off unused PLLs

Refer to the PLL configuration, you can power down the PLL that you don't need to use in order to save power consumption.

12.11 DDR Low Power Control

After a period of time when the bus has not been accessed, the DDR controller automatically enters the Self refresh and Power down states. This reduces the power consumption of the system.

In some scenarios where there are intermittent accesses and there is no large enough gap for self refresh, you can also consider using the built-in access statistics registers. In this case, you can also consider using the built-in data access statistics register to check if the bandwidth is too much to be considered for downclocking.
The DDR controller supports dynamic frequency adjustment. However, adjusting the frequency temporarily stops DDR access for a period of time. So to minimize the interrupt time may cause the real-time application buffer to underflow/overflow. It is limited to 50% and 100%.

12.12 Voltage regulation

The chip is pre-programmed to use PWM0 to control the VDDC voltage regulation.
The following is an example of using PWM to shadow the DCDC.
Example of output voltage.

VOUT = Vfb + R1(I2-I3)
= Vfb + R1(Vfb/R2-(Vpwm-Vfb)/R3)
= Vfb + R1(Vfb/R2-(VDDIOxduty-Vfb)/R3)

Vfb	R1	R2	R3	VDDIO
0.6	50,000	100,000	200,000	1.8

Duty	Vpwm	Vout
0.00%	0.00	1.05
5.00%	0.09	1.03
10.00%	0.18	1.01
15.00%	0.27	0.98
20.00%	0.36	0.96
25.00%	0.45	0.94
30.00%	0.54	0.92
35.00%	0.63	0.89
40.00%	0.72	0.87
45.00%	0.81	0.85
50.00%	0.90	0.83
55.00%	0.99	0.80
60.00%	1.08	0.78
65.00%	1.17	0.76
70.00%	1.26	0.74
75.00%	1.35	0.71
80.00%	1.44	0.69
85.00%	1.53	0.67
90.00%	1.62	0.65
95.00%	1.71	0.62
100.00%	1.80	0.60

5 RC = 5x1Kx2.2uF = 11ms
(Vpwm settle time)

Vpwm = VDDIO x PWM duty cycle

Figure *12-1 Example of DCDC* Voltage Control Using *PWM.*

138051 Subsystem

13.1 summarize

The 8051 subsystem is housed in a separate RTC-powered module and is configured with an 8051 and I2C/UART/SPI NOR/SD controllers with Timer/WDT/Interrupt Management/Mail box IP. The system software uses the 8051 to manage the wake-up conditions and wake-up the system while the system is asleep and to communicate with external devices through the peripheral controllers.

13.2 specificities

Configuration within the subsystem.

- The 8051 microprocessor has the following features
  - Supports standard 8051 instruction set
  - Frequency Range 25MHz ~ 300 MHz
  - Debugging function : single-step execution/ jump2pc/ snapshot PSW, DPTR, PC
  - Supports 32bit data access
  - The reset vector is configurable to the system AHB SRAM/ DRAM/ SPINOR
  - Supports WFI (Clock gating)
  - Support for code banking (up to 64x64 KB)
- Provides 8KB of AHB SRAM that can be used by the 8051 as an instruction TCM or to store data temporarily.
- Provides 2 sets of 32bit counters for timing and counting functions, which can be supplied to the program to realize timing and counting, or to the operating system to realize the system clock.
- Provides 1 set of WDTs for interrupting or resetting the whole system after a certain period of time in the event of a system abnormality.
- Provides 1 set of off-chip SPINOR controls
- Provides an interrupt controller to manage interrupt sources
- Provides 2 sets of Mailboxes for ACPU and 8051 to communicate with each other.
- Provides 1 set of I2C
- Provides 1 UART

13.3 Functional Description

Figure *13-1 8051* Subsystem Architecture

The subsystem is divided into two power domains: the AO domain and the MCU domain (green area). The system can choose to turn off the power domain of the MCU through the registers to achieve power saving.
When the system is asleep, it can handle interrupts through the MCU and wake up the system through configuration registers, and it can also communicate with external devices through I2C/UART.

13.4 working method

13.4.1 Power domain control flow

The subsystem is divided into three power domains, the RTC domain (Always on), and the MCU domain.

The power up and down process can be performed by configuring the registers with the following flow :

MCU power_off>

1. Soft reset register = 0
2. Configuration register reg_mcu_iso_en = 1
3. Configuration register reg_mcu_pwr_req = 0

MCU power_on=>

1. Configuration register reg_mcu_pwr_req = 1

2. Polling register reg_mcu_pwr_ack = 1
3. Configuration register reg_mcu_iso_en = 0
4. Soft reset register =1

13.4.2 8051 Initialization

The 8051 is initially in the reset state of the system. Using the 8051, the following software flow can be accomplished through the ACPU.

1. 8051 Reset state is in reset state (register reg_soft_rstn_mcu =0)
2. Configuration register reg_mcu_rom_addr_size determines the instruction TCM size.
3. Configuration register reg_51irom_ioffset determines the location of TCM execution on the AHB SRAM.
4. Configuration Register reg_soft_rstn_mcu =1 Unreset 8051

13.4.3 interrupt processing

The 8051 can receive external level-triggered interrupts through the int0_n, int1_n interfaces. int0_n/int1_n selects the output interrupt signal to the 8051 from the ictl (interrupt control) and the configuration register reg_51_int1_src_

Interruption	Interrupt name	disruption narrative
0	Vbat_det	System power-down interrupt
1	mbox_int0	Mailbox Interrupt
2	NA	reservations
3	irrx	Remote control receive interrupt
4	gpio_int	PWR GPIO Interrupt
5	uart_int	PWR UART Interrupt
6	spinor1_int	SPINOR1 Interrupt
7	timer_int0	TIMER0 Interrupt
8	timer_int1	TIMER1 Interrupt
9	Irq_ap2rtc[0]	system interruption
10	Irq_ap2rtc[1]	system interruption
11	I2c_int	PWR I2C Interrupt
12	st_change_int	RTC status change interrupt
13	13-12 8051 Subsystem Interrupt List	System overheating interruption
14	saradc	SARADC Interrupt
15	wdt_int	Watchdog interruptions

Figure 13-12 8051 Subsystem Interrupt List

13.4.4 MAILBOX

Mailbox provides 2 sets of spinlock function fields and 4 sets of 32bit mailbox fields for ACPU/8051 to transfer messages to each other.

13.5 8051 Subsystem Registers Overview

Table *13-1 RTC_CTRL* Register Overview *(*Base Address *0x05025000)*

Name	Address Offset	Description
rtc_ctrl_version	0x000	rtc_ctrl_version
rtc_ctrl_unlockkey	0x004	rtc_ctrl_unlockkey
rtc_ctrl0	0x008	rtc_ctrl0
rtc_ctrl_status0	0x00c	rtc_ctrl_status0
rtc_ctrl_status1	0x010	rtc_ctrl_status1
rtc_ctrl_status2gpio	0x014	rtc_ctrl_status2gpio
rtcsys_rst_ctrl	0x018	rtcsys_rst_ctrl
rtcsys_clkmux	0x01c	rtcsys_clkmux
rtcsys_mcu51_ctrl0	0x020	rtcsys_mcu51_ctrl0
rtcsys_mcu51_ctrl1	0x024	rtcsys_mcu51_ctrl1
rtcsys_pmu	0x028	rtcsys_pmu
rtcsys_status	0x02c	rtcsys_status
rtcsys_clkbyp	0x030	rtcsys_clkbyp
rtcsys_clk_en	0x034	rtcsys_clk_en
rtcsys_wkup_ctrl	0x038	rtcsys_wkup_ctrl
rtcsys_clkdiv	0x03c	rtcsys_clkdiv
fc_coarse_en	0x040	fc_coarse_en
fc_coarse_cal	0x044	fc_coarse_cal
fc_fine_en	0x048	fc_fine_en
fc_fine_period	0x04c	fc_fine_period
fc_fine_cal	0x050	fc_fine_cal
rtcsys_pmu2	0x054	rtcsys_pmu2
rtcsys_clkdiv1	0x058	rtcsys_clkdiv
rtcsys_mcu51_dbg	0x05c	rtcsys_mcu51_dbg
sw_reg0	0x060	sw_reg0
sw_reg1_por	0x064	sw_reg1_por
fab_lp_ctrl	0x068	fab_lp_ctrl
fab_option	0x06c	fab_option
rtcsys_mcu51_ictrl1	0x07c	rtcsys_mcu51_ictrl1
rtc_ip_pwr_req	0x080	rtc_ip_pwr_req
rtc_ip_iso_ctrl	0x084	rtc_ip_iso_ctrl
rtcsys_wkup_ctrl1	0x094	rtcsys_wkup_ctrl1
rtcsys_sram_ctrl	0x098	rtcsys_sram_ctrl
rtcsys_io_ctrl	0x09c	rtcsys_io_ctrl
rtcsys_wdt_ctrl	0x0a0	rtcsys_wdt_ctrl
rtcsys_irrx_clk_ctrl	0x0a4	rtcsys_irrx_clk_ctrl
rtcsys_rtc_wkup_ctrl	0x0a8	rtcsys_rtc_wkup_ctrl
rtcsys_por_rst_ctrl	0x0ac	rtcsys_por_rst_ctrl

13.6 8051 Subsystem Registers

rtc_ctrl_unlockkey

Offset Address: 0x004

Bits	Name	Access	Description	Reset
15:0	rtc_ctrl_unlockkey	R/W	"rtc_ctrl0" could be write when unlockkey is set to be 0xAB18.	0x0000

Bits	Name	Access	Description	Reset
			"ptest_adc2ram_ctrl" could be write when unlockkey is set to be 0x0423. If unlockkey_clear is set to 1, the rtc_ctrl0_unlockkey will be clear after an apb write to rtc_ctrl0 or ptest_adc2ram_ctrl	
31:16	Reserved			

rtc_ctrl0

Offset Address: 0x008
Write Lock: wr_lock_rtc_ctrl0

Bits	Name	Access	Description	Reset
0	req_shdn	W1P	Mask: Enabled	
1	req_sw_thm_shdn	R/W	Mask: Enabled	0x0
2	hw_thm_shdn_en	R/W	Mask: Enabled	0x0
3	req_pwr_cyc	W1P	Mask: Enabled	
4	req_warm_rst	W1P	Mask: Enabled	
5	req_sw_wdg_rst	R/W	Mask: Enabled	0x0
6	hw_wdg_rst_en	R/W	Mask: Enabled	0x0
7	req_suspend	W1P	Mask: Enabled	
8	unlockkey_clear	R/W	Mask: Enabled	0x0
9	Reserved			
10	reg_rtc_mode	R/W	Mask: Enabled	0x0
11	reg_clk32k_cg_en	R/W	Mask: Enabled	0x1
31:12	Reserved			

rtc_ctrl_status0

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
0	rtc_pwr_vbat_det_o	RO		
1	rtc_pwr_button0_o	RO		
2	rtc_pwr_button1_o	RO		
3	rtc_pwr_button1_7sec_o	RO		
4	rtc_pwr_on_o	RO		
5	rtc_pwr_wakeup0_o	RO		
6	rtc_pwr_wakeup1_o	RO		
7	rtc_mode_o	RO		
19:8	Reserved			
20	rtc_rstn_o	RO		
21	rtc_alarm_o	RO		

Bits	Name	Access	Description	Reset
22	hw_thm_shdn_sta_i	RO		
23	hw_wdg_rst_sta_i	RO		
24	sys_reset_x_i	RO		
25	cg_en_out_clk_32k	RO		
29:26	rtc_fsm_st	RO		
31:30	Reserved			

rtc_ctrl_status1

Offset Address: 0x010

Bits	Name	Access	Description	Reset
31:0	rtc_sec_value_o	RO		

rtc_ctrl_status2gpio

Offset Address: 0x014

Bits	Name	Access	Description	Reset
7:0	status2gpio_en	R/W		0x0
31:8	Reserved			

rtcsys_rst_ctrl

Offset Address: 0x018

Bits	Name	Access	Description	Reset
0	Reserved			
1	reg_soft_rstn_mcu	R/W	0 : reest MCU	0x0
2	reg_soft_rstn_sdio	R/W	0 : reset SD1	0x1
3	reg_soft_rstn_uart	R/W	0 : reset Uart	0x1
4	reg_soft_rstn_spinor	R/W	0 : reset spinor1	0x1
5	reg_soft_rstn_ictl	R/W	0 : reset dw_ictl	0x1
6	reg_soft_rstn_mbox	R/W	0 : reset mbox	0x1
7	reg_soft_rstn_fab_hs2rtc	R/W	0 : reset hs2rtc	0x1
8	reg_soft_rstn_fab_rtc2ap	R/W	0 : reset rtc2ap	0x1
9	reg_soft_rstn_fab_sram	R/W	0 : reset ahb sram logic	0x1
10	reg_soft_rstn_apb	R/W	no load	0x1
11	reg_soft_rstn_apb_timer	R/W	0 : reset dw timer apb logic	0x1
12	reg_soft_rstn_timer0	R/W	0 : reset dw timer0	0x1
13	reg_soft_rstn_timer1	R/W	0 : reset dw timer1	0x1
14	reg_soft_rstn_osc	R/W	0 : reset osc	0x1
15	reg_soft_rstn_gpio	R/W	0 : reset gpio	0x1
16	reg_soft_rstn_i2c	R/W	0 : reset i2c	0x1
17	reg_soft_rstn_saradc	R/W	0 : reset saradc	0x1
18	reg_soft_rstn_wdt	R/W	0 : reset wdt	0x1
19	reg_soft_rstn_irrx	R/W	0 : reset irrx	0x1
20	reg_soft_rstn_f32kless	R/W	0 : reset f32kless	0x1
31:21	Reserved			

rtcsys_clkmux

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
3:0	reg_sdio_clk_mux	R/W	clk_sd1_pre 0 : fpll/4 1: osc_div	0x0
7:4	reg_fab_clk_mux	R/W	clk_fab_pre 0 : 32K, 1: fpll/5, 2: osc_div	0x0

Bits	Name	Access	Description	Reset
9:8	reg_timer0_clk_mux	R/W	0: xtal 1: 32K	0x0
11:10	reg_timer1_clk_mux	R/W	0: xtal 1: 32K	0x0
13:12	reg_apb_clk_mux	R/W	00 : cgdiv and refer to apbactive 01 : force clk_apb, clk_fab 1:1 (default) 10 : force clk_apb, clk_fab 1:2 11 : force clk_apb, clk_fab 1:4	0x1
15:14	Reserved			
17:16	reg_i2c_clk_mux	R/W	0: xtal 1: osc div	0x0
19:18	reg_sd_mclk_clk_mux	R/W	0: 100Khz from OSC, 1: 32K	0x0
20	reg_saradc_clk_mux	R/W	0 : XTAL, 1: OSC DIV	0x0
21	reg_irrx_clk_mux	R/W	0 : XTAL, 1: OSC DIV	0x0
31:22	Reserved			

rtcsys_mcu51_ctrl0

Offset Address: 0x020

Bits	Name	Access	Description	Reset
4:0	reg_51_rom_addr_size	R/W	Determines how many of the sixteen internal ROM address bits (irom_addr) are used. are used (0 = no internal ROM present).	0xc
5	reg_51_mem_ea_n	R/W	0 : external rom exist, 1: external rom not exist	0x0
6	reg_51_xdata_mode	R/W	0 : fetch xdata with clock gating 1 : fetch xdata wo clock gating (to support 51 timer and 51 uart)	0x0
7	reg_51_rom_addr_def	R/W	0: mercury define , max internal rom = 2^reg_51_rom_addr_size -1 internal rom offset = 4K*reg_51irom_ioffset 1: marsdefine max internal rom = 2K*reg_51_rom_addr_size -1 internal rom offset = 2K*reg_51irom_ioffset	0x0
10:8	Reserved			
31:11	reg_51xdata_ioffset0	R/W	Set offset address[31:11] to select mcu8051 boot device	0x0A400

rtcsys_mcu51_ctrl1

Offset Address: 0x024

Bits	Name	Access	Description	Reset
4:0	reg_51irom_ioffset	R/W	boot rom offset to rtcsys_sram	0x0
5	Reserved			
9:6	reg_51_pf_mode	R/W	reg_51_pf_mode	0x0
10	Reserved			
31:11	reg_51xdata_doffset0	R/W	Set offset address[31:11] to select mcu8051 xdata	0x0A400

rtcsys_pmu

Offset Address: 0x028

Bits	Name	Access	Description	Reset
3:0	Reserved			

4	reg_dis_pmu_ldo_ctrl	R/W	disable pmu ldo ctrl	0x0

Bits	Name	Access	Description	Reset
			0: enable pmu to ctrl RTC_LDO sleep mode 1: disable pmu to ctrl RTC_LDO sleep mode	
5	reg_wdt_clkoff_by_pmu	R/W	wdt_clk gate by pmu when mcu into idle mode 1. wdt clock gate by pmu	0x0
6	reg_force_osc_off	R/W	1 : force osc off	0x0
7	reg_force_osc_on	R/W	1 : force osc on	0x0
8	reg_pmu_sleep_mode	R/W	pmu enter light sleep mode when mcu idle 1 : enable pmu light sleep mode when mcu idle (pmu control osc_req/ sram slp) 0 disable pmu light sleep mode	0x0
9	reg_pmu_lowpwr_mode	R/W	mcu_pmu into sleep state when rtc at suspend state & mcu idle & reg_pmu_sleep_mode enable 1 : enable mcu_pmu into sleep mode (trigger rtc ldo step down power) 0 disable mcu_pmu sleep mode	0x0
13:10	reg_pmu_stable_cnt	R/W	Stable timer when mcu_pmu leave sleep state. clock unit : 31.25us (32khz), wait for 1~16 tick cycle	0x3
14	reg_xtal_off_by_pmu	R/W	pmu control xtal request 1: xtal request disable by pmu sleep mode	0x0
15	reg_rtcsys_clk25m_req	R/W	xtal request1 for rtcsys 0: disable 25m xtal request1(rtcsys) 1: enable 25m xtal request1 (rtcsys)	0x1
19:16	reg_rtc_vbat_det_db_cnt	R/W	vbat det int debounce time (cycle unit : 32K)	0x2
20	reg_rtc_vbat_det_db_en	R/W	0: disable vbat det int debounce 1: enable vbat det int debounce	0x1
21	reg_ahb_sram_auto_slp_en	R/W	1: enable ahb sram into slp md when bus idle	0x0
23:22	reg_ahb_sram_busy_sel	R/W	2'd0: cs | cs_d1 2'd1: cs | cs_d1 | cs_d2 2'd2: cs | cs_d1 | cs_d2 | cs_d3 3'd3: cs | cs_d1 | cs_d2 | cs_d3 | cs_d4	0x0
24	reg_rtc_stint_clr	W1P	clear rtc state change interrupt	
25	reg_vbat_det_int_clr	W1P	clear vbet det interrupt	
26	reg_rtcsys_clk25m_hw_req	R/W	xtal request1 for rtcsys from hw ip 0: disable 25m xtal request1 from hw ip(rtcsys) 1: enable 25m xtal request1 from hw ip(rtcsys)	0x0
27	Reserved			
28	reg_vbat_det_force_clk	R/W	1: when vbat det happen, change rtcsys bus clock to OSC	0x0
29	reg_mcu_clkoff_by_pmu	R/W	mcu_clk gate by pmu when into idle mode 1. mcu clock gate by pmu	0x1
30	reg_xtal_off_by_susp	R/W	ISO off control xtal request 1: xtal request disable by ISO_OFF	0x0

Bits	Name	Access	Description	Reset
31	reg_osc_off_by_susp	R/W	ISO off control osc request 1: osc request disable by ISO_OFF	0x0

rtcsys_status

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
31:0	reg_rtcsys_status	RO	[0] enable rtc2apb ahb path 0: rtcsys ip can only access 0x05000000+16MB 1: rtcsys ip can access full range address [1] flag of vbat_det_force_clk	

rtcsys_clkbyp

Offset Address: 0x030

Bits	Name	Access	Description	Reset
31:0	reg_clk_byp	R/W	[0] : clk_fab , 0: clk_fab_pre, 1: xtal (default) [1] : clk_sdio, 1: clk_sd1_pre, 1: xtal (default) [31:2]: NA	0xffffffff

rtcsys_clk_en

Offset Address: 0x034

Bits	Name	Access	Description	Reset
31:0	reg_clk_en	R/W	[0] : NA [1] : clk_sd1 (sd1 card clock) [2] : clk_fab_sd1 (sd1 core clock) [3] : clk_mcu [4] : clk_hs2rtc_mst [5] : clk_rtc2ap_slv [6] : clk_spinor1 [7] : clk_fab_sram (AHB sram) [8] : NA [9] : clk_apb_timer [10] : clk_timer0 [11] : clk_timer1 [12] : clk_apb_uart [13] : clk_uart [14] : clk_apb_ictrl [15] : clk_apb_mbox [16] : clk_apb_gpio [17] : clk_apb_osc [18] : clk_gpio_db [19] : clk_apb_i2c [20] : clk_i2c [21] : NA [22] : clk_sd1_tmclk [23] : clk_apb_saradc [24] : clk_saradc [25] : clk_apb_wdt [26] : clk_wdt [27] : clk_irrx [31:28]: NA	0xffffffff

rtcsys_wkup_ctrl

Offset Address: 0x038

Bits	Name	Access	Description	Reset
14:0	reg_rtcsys_wkint_mask	R/W	mask int to RTC_CORE.REQ_WAKEUP/ MCU_PMU [0] : irrrx_intr [1] : gpio_int [2] : timer0_int [3] : timer1_int [4] : saradc_int [5] : rtcsys_ictrl_int [6] : wdt_int [7] : irrx_wakeup	0xff
15	reg_vbat_det_wkup_mask	R/W	1: mask vbat det int	0x1
16	reg_sw_wkint_req	R/W	mcu sw wakeup interrupt to RTC_CORE 1: interrupt active	0x0
23:17	Reserved			
24	reg_wkint2rtc_mask	R/W	1: mask wakeup int (rtcsys int) to RTC core	0x1
31:25	Reserved			

rtcsys_clkdiv

Offset Address: 0x03c

Bits	Name	Access	Description	Reset
3:0	reg_div_clk_osc_fab_div_val	R/W	Clock Divider Factor	0x1
4	reg_div_clk_osc_fab_dis	R/W	Clock gate	0x0
5	reg_div_clk_osc_fab_hwide	R/W	Select High Wide Control (when Divider Factor is odd) 0: Low level of the clock is wider 1: High level of the clock is wider	0x0
15:6	Reserved			
19:16	reg_div_clk_osc_i2c_div_val	R/W	Clock Divider Factor	0x1
20	reg_div_clk_osc_i2c_dis	R/W	Clock gate	0x0
21	reg_div_clk_osc_i2c_hwide	R/W	Select High Wide Control (when Divider Factor is odd) 0: Low level of the clock is wider 1: High level of the clock is wider	0x0
23:22	Reserved			
29:24	reg_div_clk_osc_saradc_div_val	R/W	Clock Divider Factor	0x1
30	reg_div_clk_osc_saradc_dis	R/W	Clock gate	0x0
31	reg_div_clk_osc_saradc_hwide	R/W	Select High Wide Control (when Divider Factor is odd) 0: Low level of the clock is wider 1: High level of the clock is wider	0x0

fc_coarse_en

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	fc_coarse_en	R/W	Enable 32K Coarse Emphasis 0 = disable, 1 = enable	0x0
31:1	Reserved			

fc_coarse_cal

Offset Address: 0x044

Bits	Name	Access	Description	Reset
15:0	fc_coarse_value	RO	32K Coarse Counter Value (Unit: 25MHz clock) 25MHz clock counts a 32K clock cycle.	

31:16	fc_coarse_time	RO	32K Coarse adjustment completion times	

fc_fine_en

Offset Address: 0x048

Bits	Name	Access	Description	Reset
0	fc_fine_en	R/W	Enable 32K fine tuning 0 = disable, 1 = enable	0x0
31:1	Reserved			

fc_fine_period

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
15:0	fc_fine_period	R/W	32K fine-tuned count period (Unit: 32K clock) Sets the 25MHz clock to count several 32K at a time. Clock cycle.	0x0100
31:16	Reserved			

fc_fine_cal

Offset Address: 0x050

Bits	Name	Access	Description	Reset
23:0	fc_fine_value	RO	32K fine tuning counter value (unit: 25MHz clock) 25MHz clock counter one fc_fine_period Period.	
31:24	fc_fine_time	RO	32K fine tuning completion times	

rtcsys_pmu2

Offset Address: 0x054

Bits	Name	Access	Description	Reset
0	reg_rtc_sys_wkint_db_en	R/W	PMU wakeup int debounce enable	0x1
4:1	reg_rtc_sys_wkint_db_cnt	R/W	PMU wakeup int debounce cycle (32K)	0x2
31:5	Reserved			

rtcsys_clkdiv1

Offset Address: 0x058

Bits	Name	Access	Description	Reset
15:0	Reserved			
21:16	reg_div_clk_osc_irrx_div_val	R/W	Clock Divider Factor	0x0
22	Reserved			
23	reg_div_clk_osc_irrx_dis	R/W	Clock gate	0x1
31:24	Reserved			

rtcsys_mcu51_dbg

Offset Address: 0x05c

Bits	Name	Access	Description	Reset
3:0	reg_51_dbg_sel	R/W	select mcu51 debug bus (check mcu design review ppt)	0x0
4	reg_51_dbg_snap_shot	W1P	snap shot mcu51 internal register to dbg register (reg_rtcsys_dbg)	
5	reg_51_dbg_step_en	R/W	0: disable mcu debug function 1: enable mcu debug function, and mcu stop at current PC	0x0
6	reg_51_dbg_step	W1P	1: mcu jump to next PC	

7	reg_51_dbg_jump	W1P	1: mcu jump to target pc value (reg_51_dbg_jump2pc)	

Bits	Name	Access	Description	Reset
15:8	Reserved			
31:16	reg_51_dbg_jump2pc	R/W	16 bit mcu target pc value	0x0

sw_reg0

Offset Address: 0x060

Bits	Name	Access	Description	Reset
7:0	sw_reg0	R/W	reg for SW	0x0
31:8	Reserved			

sw_reg1_por

G

Offset Address: 0x064

Bits	Name	Access	Description	Reset
7:0	sw_reg1_por	R/W	reg for SW could only be reset by power reset	0x0
31:8	Reserved			

fab_lp_ctrl

Offset Address: 0x068

Bits	Name	Access	Description	Reset
7:0	rtcsys_fab_busy_sel	R/W	select signal to request sys_ctrl to speed up fab clock	0xDF
9:8	rtcsys_fab_busy_ctrl	R/W	rtcsys_fab_busy signal is combi or register out	0x0
11:10	apdbg_busy_ctrl	R/W	apdbg_busy signal is combi or register out	0x0
13:12	reg_apb_busy_ctrl	R/W	apb bridge_busy signal is combi or register out	0x3
15:14	reg_mcu_busy_ctrl	R/W	mcu_busy signal is combi or register out	0x3
31:16	Reserved			

fab_option

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
1:0	rtcsys_fab_option	R/W	the ahb_h2h2 design option	0x0
31:2	Reserved			

rtcsys_mcu51_ictrl1

Offset Address: 0x07c

Bits	Name	Access	Description	Reset
15:0	reg_51_int1_src_mask	R/W	select rtcsys_int src to mcu int1_n 1: mask, 0: un-mask [0] : vbat_det [1] : mbox0_int [2] : NA [3] : irrx_int [4] : gpio_int [5] : uart_int [6] : spinor1_int [7] : timer0_int [8] : timer1_int [9] : irq_ap2rtc[0] [10] : irq_ap2rtc[1] [11] : i2c_int	0xffff

Bits	Name	Access	Description	Reset
			[12] : rtc_state_change_int [13] : hw_thm_shdn [14] : saradc [15] : wdt_int	
31:16	reg_51_int1_final_status	RO	mcu int1_n status [0]: vbat_det [1] : mbox0_int [2] : NA [3] : irrx_int [4] : gpio_int [5] : uart_int [6] : spinor1_int [7] : timer0_int [8] : timer1_int [9] : irq_ap2rtc[0] [10] : irq_ap2rtc[1] [11] : i2c_int [12] : rtc_state_change_int [13] : hw_thm_shdn [14] : saradc [15] : wdt_int	

rtc_ip_pwr_req

Offset Address: 0x080

Bits	Name	Access	Description	Reset
0	reg_sd1_pwr_req	R/W	power fence control 1: power on, 0: power off [0]: sd1	0x1
1	reg_sd1_pwr_req_2nd	R/W	power fence control 1: power on, 0: power off [0]: sd1	0x1
2	reg_mcu_pwr_req	R/W	power fence control 1: power on, 0: power off [1]: mcu subsys	0x1
3	reg_mcu_pwr_req_2nd	R/W	power fence control 1: power on, 0: power off [1]: mcu subsys	0x1
15:4	Reserved			
16	reg_sd1_pwr_ack	RO	power fence power status 1: power on, 0: power off [0]: sd1	
17	reg_sd1_pwr_ack_2nd	RO	power fence power status 1: power on, 0: power off [0]: sd1	
18	reg_mcu_pwr_ack	RO	power fence power status 1: power on, 0: power off [1]: mcu subsys	
19	reg_mcu_pwr_ack_2nd	RO	power fence power status 1: power on, 0: power off [1]: mcu subsys	
31:20	Reserved			

rtc_ip_iso_ctrl

Offset Address: 0x084

Bits	Name	Access	Description	Reset
0	reg_sd1_iso_en	R/W	sd1 iso enablel	0x0

Bits	Name	Access	Description	Reset
			1: iso enable, 0: iso disable	
1	reg_mcu_iso_en	R/W	mcu iso enablel 1: iso enable, 0: iso disable	0x0
15:2	Reserved			
17:16	reg_ip_por_en	R/W	1: pwr_island reset assert when power ack is 0	0x3
31:18	Reserved			

rtcsys_wkup_ctrl1

Offset Address: 0x094

Bits	Name	Access	Description	Reset
7:0	reg_rtcsys_wkint_final_status	RO	wkint final status [0] : sd1_wakeup_intr [1] : gpio_int [2] : timer0_int [3] : timer1_int [4] : saradc_int [5] : rtcsys_ictrl_int [6] : NA [7] : NA	
31:8	Reserved			

rtcsys_sram_ctrl

Offset Address: 0x098

Bits	Name	Access	Description	Reset
0	reg_ahb_sram_slp	R/W	1 : ahb sram into sleep mode	0x0
1	reg_ahb_sram_sd	R/W	1 : ahb sram into shut down mode	0x0
2	reg_ahb_sram_ctrl_ov	R/W	0 : ahb sram ctrl by PMU FSM and ahb sram busy 1: sram ctrol by register reg_ahb_sram_slp/reg_ahb_sram_sd	0x1
3	reg_sdio_sram_slp	R/W	1 : sdio sram into sleep mode	0x0
4	reg_sdio_sram_sd	R/W	1 : sdio sram into shut down mode	0x0
5	reg_sdio_sram_ctrl_ov	R/W	0 : sram's sd pin = 1'b0 1: sram ctrol by register reg_sdio_sram_sd	0x1
6	reg_mcu_sram_slp	R/W	1 : mcu iram sram into sleep mode	0x0
7	reg_mcu_sram_sd	R/W	1 : mcu iram sram into shut down mode	0x0
8	reg_mcu_sram_ctrl_ov	R/W	0 : mcu iram sram ctrl by PMU FSM 1: sram ctrol by register reg_ahb_sram_slp/reg_ahb_sram_sd	0x1
9	reg_rtc_sram_slp	R/W	1 : mcu iram sram into sleep mode	0x0
10	reg_rtc_sram_sd	R/W	1 : mcu iram sram into shut down mode	0x0
11	reg_rtc_sram_ctrl_ov	R/W	0 : mcu iram sram ctrl by PMU FSM 1: sram ctrol by register reg_ahb_sram_slp/reg_ahb_sram_sd	0x1
12	reg_ahb_sram_1_slp	R/W	1 : ahb sram into sleep mode	0x0
13	reg_ahb_sram_1_sd	R/W	1 : ahb sram into shut down mode	0x0
14	reg_ahb_sram_1_ctrl_ov	R/W	0 : ahb sram ctrl by PMU FSM and ahb sram busy 1: sram ctrol by register reg_ahb_sram_slp/reg_ahb_sram_sd	0x1
15	reg_ahb_sram_2_slp	R/W	1 : ahb sram into sleep mode	0x0
16	reg_ahb_sram_2_sd	R/W	1 : ahb sram into shut down mode	0x0
17	reg_ahb_sram_2_ctrl_ov	R/W	0 : ahb sram ctrl by PMU FSM and ahb	0x1

Bits	Name	Access	Description	Reset
			sram busy 1: sram ctrol by register reg_ahb_sram_slp/reg_ahb_sram_sd	
27:18	Reserved			
28	reg_mcu_sram_force_ce	R/W	1: force mcu_iram cs = 1	0x1
31:29	Reserved			

rtcsys_io_ctrl

Offset Address: 0x09c

Bits	Name	Access	Description	Reset
0	reg_i2c_mux_opt0	R/W	0: pwr_gpio6/8 control by dw_gpio 1: pwr_gpio6 is PWR_IIC_SDA pwr_gpio8 is PWR_IIC_SCL	0x0
31:1	Reserved			

rtcsys_wdt_ctrl

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
0	reg_rtc_hw_wdg_rst_en	R/W	0: disable rtc wdt trigger warm reset or pwrcyc reset 1: enable rtc wdt trigger warm reset or pwrcyc reset	0x0
1	reg_rtc_wdt_ctrl_mask_en	R/W	no load	0x1
31:2	Reserved			

rtcsys_irrx_clk_ctrl

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
0	reg_irrx_clk_sw_force_on	R/W	force on clk ctrl of irrx	0x1
1	reg_irrx_xtal_req_en	R/W	enable irrx clk ctrl requet XTAL	0x0
2	reg_irrx_osc_req_en	R/W	enable irrx clk ctrl requet OSC	0x0
3	reg_irrx_ldo_req_en	R/W	enable irrx clk ctrl requet LDO	0x0
7:4	Reserved			
15:8	reg_irrx_xtal_filter_cyc	R/W	irrx xtal filter cycle (default 2ms)	0x40
19:16	reg_irrx_clk_ctrl_st	RO	irrx clock ctrol state	
31:20	Reserved			

rtcsys_rtc_wkup_ctrl

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
7:0	reg_rtc_wkint_mask	R/W	wakeup source mask int to RTC_CORE [0]: irrrx_intr [1] : gpio_int [2] : timer0_int [3] : timer1_int [4] : saradc_int [5] : rtcsys_ictrl_int [6] : wdt_int [7] : irrx_wakeup	0xff
15:8	Reserved			
23:16	reg_rtc_puint_mask	R/W	power-up source mask int to RTC_CORE [0]: irrrx_intr [1] : gpio_int	0xff

Bits	Name	Access	Description	Reset
			[2] : timer0_int [3] : timer1_int [4] : saradc_int [5] : rtcsys_ictrl_int [6] : wdt_int [7] : irrx_wakeup	
31:24	Reserved			

rtcsys_por_rst_ctrl

Offset Address: 0x0ac

Bits	Name	Access	Description	Reset
0	reg_rtcsys_reset_en	R/W	0: not allow rtcsys reset by pwr cyc/ wdt warm reset 1 : allow rtcsys reset by pwr cyc/ wdt warm reset	0x0
1	reg_rtcsys_rstn_src_sel	R/W	select rtcsys rstn src 0: rtc_core fsm (reset with die domain) 1: por_pwr_rstn	0x0
31:2	Reserved			

14 memory interface

14.1 DDR Controller

14.1.1 summarize

The DDR controller implements dynamic memory (DRAM) data access by converting data access commands from each master device in the SoC into JEDEC-compliant DRAM commands, which are properly scheduled to improve the efficiency of the DRAM bus.

14.1.2 specificities

- Functional Features:
- Support.
  - DDR2 Maximum data rate 1333 Mbps.
  - DDR3 Maximum data rate 1866 Mbps.
- Supports interface data bit width of 16-bit.
- Supports single channel, single rank.
- Supports automatic refresh control.
- Supports priority control.
- Supports data traffic statistics.
- Supports low-power mode.
- Address mapping is supported.
- Pin multiplexing is supported.

14.1.3 Functional Description

14.1.3.1 application block diagram

The DRAM interface supports 16-bit data bit width and 32-bit bit width. Figure 14-1 illustrates the interconnection between the host chip and a single DRAM device:

Figure *14-1 SoC/DRAM* Interconnect Diagram

Command consists of several signals, which vary according to the DRAM type. Table 4-1 compares the DDR2 / DDR3 command signals:

Table *14-1 DDR2/DDR3* Command Signal Comparison Table

functionality	DDR2	DDR3
DDR CKE	D2x_CKE	D3x_CKE
DDR CLKN	D2x_CK_N	D3x_CK_N
DDR CLKP	D2x_CK_P	D3x_CK_P
DDR CSB	D2x_CS_N	D3x_CS_N
DDR RESETN	D2x_RESET_N	D3x_RESET_N
DDR RASN	D2x_RAS_N	D3x_RAS_N
DDR CASN	D2x_CAS_N	D3x_CAS_N
DDR WEN	D2x_WE_N	D3x_WE_N
DDR ACTN	N/A	N/A
DDR BA	D2x_BAn	D3x_BAn
DDR BG	N/A	N/A
DDR MA	D2x_An (n = 0 - 15)	D3x_An (n = 0 - 15)
DDR ODT	D2x_ODT	D3x_ODT
Description. Pin names may vary slightly depending on the package.		

14.1.3.2 Functional principle

Based on the storage characteristics of DRAMs, JEDEC has developed a set of standards that regulate the commands and timing needed to access DRAM data and control DRAM status. By appropriately configuring the DDR registers, the DDR controller is able to issue timings that satisfy the JEDEC standards and perform actions such as reads, writes, and low-power control.

14.1.3.2.1 Command Truth Table

The DDR interface timing meets the JDEDC standard, Table 14-2 and Table 14-3 show the truth table of DDR2 and DDR3 supported commands for user's reference, and the rest of the information can refer to the JEDEC standard.

Table *14-2 DDR2* Command Truth Table

Function	CKE Pre	CKE Cur	CS #	RAS#	CAS#	WE#	BA0 - BA2	A11 - A15	A10 / AP	A0 - A9
Mode Register Set	H	H	L	L	L	L	BA	OP		
Refresh	H	H	L	L	L	H	V	V	V	V
Self Refresh Entry	H	L	L	L	L	H	V	V	V	V
Self Refresh Exit	L	H	H	X	X	X	X	X	X	X
			L	H	H	H	V	V	V	V
Single Bank Precharge	H	H	L	L	H	L	BA	V	L	V
Precharge all Banks	H	H	L	L	H	L	V	V	H	V
Bank Activate	H	H	L	L	H	H	BA	RA		
Write	H	H	L	H	L	L	BA	RFU	L	CA
Write with Auto Precharge	H	H	L	H	L	L	BA	RFU	H	CA
Read	H	H	L	H	L	H	BA	RFU	L	CA
Read with Auto Precharge	H	H	L	H	L	H	BA	RFU	H	CA
Read with Auto Precharge	H	H	L	H	L	H	BA	RFU	H	CA
No Operation	H	H	L	H	H	H	V	V	V	V
Device Deselected	H	H	H	X	X	X	X	X	X	X
Power Down Entry	H	L	L	H	H	H	V	V	V	V
			H	X	X	X	X	X	X	X
Power Down Exit	L	H	L	H	H	H	V	V	V	V
			H	X	X	X	X	X	X	X
H: indicates high level; L: indicates low level; V: indicates valid; X: indicates do not care. RFU: Indicates reserved for future use.										

Table *14-3 DDR3* Command Truth Table

Function	CKE Pre	CKE Cur	CS #	RAS#	CAS#	WE#	BA0 - BA2	A13 - A15	A12 / BC #	A10 / AP	A0 - A9, 11
Mode Register Set	H	H	L	L	L	L	BA	OP			
Refresh	H	H	L	L	L	H	V	V	V	V	V
Self Refresh Entry	H	L	L	L	L	H	V	V	V	V	V
Self Refresh Exit	L	H	H	X	X	X	X	X	X	X	X
			L	H	H	H	V	V	V	V	V
Single Bank Precharge	H	H	L	L	H	L	BA	V	V	L	V
Precharge all Banks	H	H	L	L	H	L	V	V	V	H	V
Bank Activate	H	H	L	L	H	H	BA	RA			
Write (Fixed BL8 or BC4)	H	H	L	H	L	L	BA	RFU	V	L	CA

Function	CKE Pre	CKE Cur	CS #	RAS#	CAS#	WE#	BA0 - BA2	A13 - A15	A12 / BC #	A10 / AP	A0 - A9, 11
Write (BC4, on the Fly)	H	H	L	H	L	L	BA	RFU	L	L	CA
Write (BL8, on the Fly)	H	H	L	H	L	L	BA	RFU	H	L	CA
Write with Auto Precharge (Fixed BL8 or BC4)	H	H	L	H	L	L	BA	RFU	V	H	CA
Write with Auto Precharge (BC4, on the Fly)	H	H	L	H	L	L	BA	RFU	L	H	CA
Write with Auto Precharge (BL8, on the Fly)	H	H	L	H	L	L	BA	RFU	H	H	CA
Read (Fixed BL8 or BC4)	H	H	L	H	L	H	BA	RFU	V	L	CA
Read (BC4, on the Fly)	H	H	L	H	L	H	BA	RFU	L	L	CA
Read (BL8, on the Fly)	H	H	L	H	L	H	BA	RFU	H	L	CA
Read with Auto Precharge (Fixed BL8 or BC4)	H	H	L	H	L	H	BA	RFU	V	H	CA
Read with Auto Precharge (BC4, on the Fly)	H	H	L	H	L	H	BA	RFU	L	H	CA
Read with Auto Precharge (BL8, on the Fly)	H	H	L	H	L	H	BA	RFU	H	H	CA
No Operation	H	H	L	H	H	H	V	V	V	V	V
Device Deselected	H	H	H	X	X	X	X	X	X	X	X
Power Down Entry	H	L	L	H	H	H	V	V	V	V	V
			H	X	X	X	X	X	X	X	X
Power Down Exit	L	H	L	H	H	H	V	V	V	V	V
			H	X	X	X	X	X	X	X	X
ZQ Calibration Long	H	H	L	H	H	L	X	X	X	H	X
ZQ Calibration Short	H	H	L	H	H	L	X	X	X	L	X

H: indicates high level; L: indicates low level; V: indicates valid; X: indicates do not care.
RFU: Indicates reserved for future use.

14.1.3.2.2 auto-refresh

DDR controller has the ability to control the automatic refresh function, the purpose of controlling the automatic refresh is to reduce the delay in accessing data or to reduce the impact of the refresh command on the DRAM bandwidth, and try to send the refresh command when the DRAM is idle, the specific available means are:

- Equal interval refresh: send refresh command every tREFI time interval.
- Tactile Refresh: The DDR controller internally counts the number of tREFI expirations and then uses the idle time to send them out continuously.

14.1.3.2.3 Low Power Management

The DDR controller supports low-power mode:

- Normal Low Power Mode: Set an idle time through registers, when Normal Low Power Mode is enabled, and

  The DDR controller meets the idle time and automatically controls the DRAM to enter normal low consumption mode.

- Self-refresh mode: Self-refresh is a lower power consumption mode, set an idle time through registers, when the self-refresh mode is enabled and the DDR controller meets the idle time, it will automatically control the DRAM to enter the self-refresh mode.

14.1.3.2.4 Arbitration mechanism

The DDR controller optimizes the bandwidth usage of the system based on the DRAM control timing, and schedules the commands through the priority scheduling algorithm. In addition, DDRC also realizes timeout control and real-time control, which are two kinds of scheduling aids (these two kinds of controls can be enabled according to the business needs, and can be enabled at the same time or separately).
able) which controls the request for the command.

- contiguous address access restriction

  The limiting levels are: 0 to 15 DRAM read/write instructions, and each AXI port is configured independently.The DDR controller defaults to high priority for contiguous addresses to optimize DRAM utilization, a mechanism that limits the length of the maximum contiguous DRAM that can be accessed at each AXI port.
- prioritized scheduling

  The priority level is: 0~15, the higher value indicates higher priority, and the read/write priority configuration is independent for each AXI port.
- Timeout Control

  For read/write transfers to each AXI port, the timeout register can be configured to avoid excessively long waits, and to force blocking of AXI ports that have not yet reached the wait time, or that do not have the timeout attribute configured, after the wait time has been reached.
- Real-time control

  For real-time functions, hardware buffer thresholds can be configured to automatically raise the priority to the highest if the buffer is insufficient, and other AXI ports can be restricted from generating new transmissions.

14.1.3.2.5 Traffic statistics and command latency statistics

The DDR controller supports traffic counting: read/write traffic can be counted at each AXI port to gather current traffic information to determine if flow control is needed. Overall DRAM read/write traffic statistics can be counted.

The DDR controller supports the AXI latency statistics function, which supports cumulative latency statistics for specified/unspecified transmissions.

14.1.3.2.6 Address Mapping Method

The DDR controller implements the translation of the system bus access address to the DRAM access address. It implements RBC (row_bank_column),

BRC, and supports Bank interleave in row/column bit.

14.1.4 working method

14.1.4.1 soft reset (electronics)

Soft reset is not supported.

14.1.4.2 DDR Initialization Configuration Flow

This controller initialization process is provided as a software package.

14.1.5 AXI Register

14.1.5.1 AXI Register Overview

Base address 0x0800_4000

Name	Address Offset	Description
AXI_CTRL0_1	0x4b4	AXI1 read timeout control
AXI_CTRL1_1	0x4b8	AXI1 write timeout control
AXI_CTRL0_2	0x564	AXI2 read timeout control
AXI_CTRL1_2	0x568	AXI2 write timeout control
AXI_CTRL0_3	0x614	AXI3 read timeout control
AXI_CTRL1_3	0x618	AXI3 write timeout control

Base address 0x0800_8000

Name	Address Offset	Description
AXI_MON0_CTRL	0x000	AXI monitor 0 control
AXI_MON0_INPUT	0x004	AXI monitor 0 input selection
AXI_MON0_FILTER0	0x010	AXI monitor 0 filter settings
AXI_MON0_FILTER1	0x014	AXI monitor 0 filter settings
AXI_MON0_FILTER2	0x018	AXI monitor 0 filter settings
AXI_MON0_FILTER3	0x01c	AXI monitor 0 filter settings
AXI_MON0_FILTER4	0x020	AXI monitor 0 filter settings
AXI_MON0_FILTER5	0x024	AXI monitor 0 filter settings
AXI_MON0_FILTER6	0x028	AXI monitor 0 filter settings
AXI_MON0_FILTER7	0x02c	AXI monitor 0 filter settings
AXI_MON0_FILTER8	0x030	AXI monitor 0 filter settings
AXI_MON0_RPT0	0x040	AXI monitor 0 cycle count
AXI_MON0_RPT1	0x044	AXI monitor 0 hit count
AXI_MON0_RPT2	0x048	AXI monitor 0 byte count
AXI_MON0_RPT3	0x04c	AXI monitor 0 latency count
AXI_MON1_CTRL	0x080	AXI monitor 1 control
AXI_MON1_INPUT	0x084	AXI monitor 1 input selection
AXI_MON1_FILTER0	0x090	AXI monitor 1 filter settings
AXI_MON1_FILTER1	0x094	AXI monitor 1 filter settings
AXI_MON1_FILTER2	0x098	AXI monitor 1 filter settings
AXI_MON1_FILTER3	0x09c	AXI monitor 1 filter settings
AXI_MON1_FILTER4	0x0a0	AXI monitor 1 filter settings
AXI_MON1_FILTER5	0x0a4	AXI monitor 1 filter settings
AXI_MON1_FILTER6	0x0a8	AXI monitor 1 filter settings
AXI_MON1_FILTER7	0x0ac	AXI monitor 1 filter settings
AXI_MON1_FILTER8	0x0b0	AXI monitor 1 filter settings
AXI_MON1_RPT0	0x0c0	AXI monitor 1 cycle count
AXI_MON1_RPT1	0x0c4	AXI monitor 1 hit count
AXI_MON1_RPT2	0x0c8	AXI monitor 1 byte count
AXI_MON1_RPT3	0x0cc	AXI monitor 1 latency count
AXI_MON2_CTRL	0x100	AXI monitor 2 control

Name	Address Offset	Description
AXI_MON2_INPUT	0x104	AXI monitor 2 input selection
AXI_MON2_FILTER0	0x110	AXI monitor 2 filter settings
AXI_MON2_FILTER1	0x114	AXI monitor 2 filter settings
AXI_MON2_FILTER2	0x118	AXI monitor 2 filter settings
AXI_MON2_FILTER3	0x11c	AXI monitor 2 filter settings
AXI_MON2_FILTER4	0x120	AXI monitor 2 filter settings
AXI_MON2_FILTER5	0x124	AXI monitor 2 filter settings
AXI_MON2_FILTER6	0x128	AXI monitor 2 filter settings
AXI_MON2_FILTER7	0x12c	AXI monitor 2 filter settings
AXI_MON2_FILTER8	0x130	AXI monitor 2 filter settings
AXI_MON2_RPT0	0x140	AXI monitor 2 cycle count
AXI_MON2_RPT1	0x144	AXI monitor 2 hit count
AXI_MON2_RPT2	0x148	AXI monitor 2 byte count
AXI_MON2_RPT3	0x14c	AXI monitor 2 latency count
AXI_MON3_CTRL	0x180	AXI monitor 3 control
AXI_MON3_INPUT	0x184	AXI monitor 3 input selection
AXI_MON3_FILTER0	0x190	AXI monitor 3 filter settings
AXI_MON3_FILTER1	0x194	AXI monitor 3 filter settings
AXI_MON3_FILTER2	0x198	AXI monitor 3 filter settings
AXI_MON3_FILTER3	0x19c	AXI monitor 3 filter settings
AXI_MON3_FILTER4	0x1a0	AXI monitor 3 filter settings
AXI_MON3_FILTER5	0x1a4	AXI monitor 3 filter settings
AXI_MON3_FILTER6	0x1a8	AXI monitor 3 filter settings
AXI_MON3_FILTER7	0x1ac	AXI monitor 3 filter settings
AXI_MON3_FILTER8	0x1b0	AXI monitor 3 filter settings
AXI_MON3_RPT0	0x1c0	AXI monitor 3 cycle count
AXI_MON3_RPT1	0x1c4	AXI monitor 3 hit count
AXI_MON3_RPT2	0x1c8	AXI monitor 3 byte count
AXI_MON3_RPT3	0x1cc	AXI monitor 3 latency count
AXI_MON4_CTRL	0x200	AXI monitor 4 control
AXI_MON4_INPUT	0x204	AXI monitor 4 input selection
AXI_MON4_FILTER0	0x210	AXI monitor 4 filter settings
AXI_MON4_FILTER1	0x214	AXI monitor 4 filter settings
AXI_MON4_FILTER2	0x218	AXI monitor 4 filter settings
AXI_MON4_FILTER3	0x21c	AXI monitor 4 filter settings
AXI_MON4_FILTER4	0x220	AXI monitor 4 filter settings
AXI_MON4_FILTER5	0x224	AXI monitor 4 filter settings
AXI_MON4_FILTER6	0x228	AXI monitor 4 filter settings
AXI_MON4_FILTER7	0x22c	AXI monitor 4 filter settings
AXI_MON4_FILTER8	0x230	AXI monitor 4 filter settings
AXI_MON4_RPT0	0x240	AXI monitor 4 cycle count
AXI_MON4_RPT1	0x244	AXI monitor 4 hit count
AXI_MON4_RPT2	0x248	AXI monitor 4 byte count
AXI_MON4_RPT3	0x24c	AXI monitor 4 latency count
AXI_MON5_CTRL	0x280	AXI monitor 5 control
AXI_MON5_INPUT	0x284	AXI monitor 5 input selection
AXI_MON5_FILTER0	0x290	AXI monitor 5 filter settings
AXI_MON5_FILTER1	0x294	AXI monitor 5 filter settings
AXI_MON5_FILTER2	0x298	AXI monitor 5 filter settings
AXI_MON5_FILTER3	0x29c	AXI monitor 5 filter settings
AXI_MON5_FILTER4	0x2a0	AXI monitor 5 filter settings
AXI_MON5_FILTER5	0x2a4	AXI monitor 5 filter settings
AXI_MON5_FILTER6	0x2a8	AXI monitor 5 filter settings
AXI_MON5_FILTER7	0x2ac	AXI monitor 5 filter settings
AXI_MON5_FILTER8	0x2b0	AXI monitor 5 filter settings

Name	Address Offset	Description
AXI_MON5_RPT0	0x2c0	AXI monitor 5 cycle count
AXI_MON5_RPT1	0x2c4	AXI monitor 5 hit count
AXI_MON5_RPT2	0x2c8	AXI monitor 5 byte count
AXI_MON5_RPT3	0x2cc	AXI monitor 5 latency count
AXI_MON6_CTRL	0x300	AXI monitor 6 control
AXI_MON6_INPUT	0x304	AXI monitor 6 input selection
AXI_MON6_FILTER0	0x310	AXI monitor 6 filter settings
AXI_MON6_FILTER1	0x314	AXI monitor 6 filter settings
AXI_MON6_FILTER2	0x318	AXI monitor 6 filter settings
AXI_MON6_FILTER3	0x31c	AXI monitor 6 filter settings
AXI_MON6_FILTER4	0x320	AXI monitor 6 filter settings
AXI_MON6_FILTER5	0x324	AXI monitor 6 filter settings
AXI_MON6_FILTER6	0x328	AXI monitor 6 filter settings
AXI_MON6_FILTER7	0x32c	AXI monitor 6 filter settings
AXI_MON6_FILTER8	0x330	AXI monitor 6 filter settings
AXI_MON6_RPT0	0x340	AXI monitor 6 cycle count
AXI_MON6_RPT1	0x344	AXI monitor 6 hit count
AXI_MON6_RPT2	0x348	AXI monitor 6 byte count
AXI_MON6_RPT3	0x34c	AXI monitor 6 latency count
AXI_MON7_CTRL	0x380	AXI monitor 7 control
AXI_MON7_INPUT	0x384	AXI monitor 7 input selection
AXI_MON7_FILTER0	0x390	AXI monitor 7 filter settings
AXI_MON7_FILTER1	0x394	AXI monitor 7 filter settings
AXI_MON7_FILTER2	0x398	AXI monitor 7 filter settings
AXI_MON7_FILTER3	0x39c	AXI monitor 7 filter settings
AXI_MON7_FILTER4	0x3a0	AXI monitor 7 filter settings
AXI_MON7_FILTER5	0x3a4	AXI monitor 7 filter settings
AXI_MON7_FILTER6	0x3a8	AXI monitor 7 filter settings
AXI_MON7_FILTER7	0x3ac	AXI monitor 7 filter settings
AXI_MON7_FILTER8	0x3b0	AXI monitor 7 filter settings
AXI_MON7_RPT0	0x3c0	AXI monitor 7 cycle count
AXI_MON7_RPT1	0x3c4	AXI monitor 7 hit count
AXI_MON7_RPT2	0x3c8	AXI monitor 7 byte count
AXI_MON7_RPT3	0x3cc	AXI monitor 7 latency count
AXI_MON8_CTRL	0x400	AXI monitor 8 control
AXI_MON8_INPUT	0x404	AXI monitor 8 input selection
AXI_MON8_FILTER0	0x410	AXI monitor 8 filter settings
AXI_MON8_FILTER1	0x414	AXI monitor 8 filter settings
AXI_MON8_FILTER2	0x418	AXI monitor 8 filter settings
AXI_MON8_FILTER3	0x41c	AXI monitor 8 filter settings
AXI_MON8_FILTER4	0x420	AXI monitor 8 filter settings
AXI_MON8_FILTER5	0x424	AXI monitor 8 filter settings
AXI_MON8_FILTER6	0x428	AXI monitor 8 filter settings
AXI_MON8_FILTER7	0x42c	AXI monitor 8 filter settings
AXI_MON8_FILTER8	0x430	AXI monitor 8 filter settings
AXI_MON8_RPT0	0x440	AXI monitor 8 cycle count
AXI_MON8_RPT1	0x444	AXI monitor 8 hit count
AXI_MON8_RPT2	0x448	AXI monitor 8 byte count
AXI_MON8_RPT3	0x44c	AXI monitor 8 latency count
AXI_MON9_CTRL	0x480	AXI monitor 9 control
AXI_MON9_INPUT	0x484	AXI monitor 9 input selection
AXI_MON9_FILTER0	0x490	AXI monitor 9 filter settings
AXI_MON9_FILTER1	0x494	AXI monitor 9 filter settings
AXI_MON9_FILTER2	0x498	AXI monitor 9 filter settings
AXI_MON9_FILTER3	0x49c	AXI monitor 9 filter settings

Name	Address Offset	Description
AXI_MON9_FILTER4	0x4a0	AXI monitor 9 filter settings
AXI_MON9_FILTER5	0x4a4	AXI monitor 9 filter settings
AXI_MON9_FILTER6	0x4a8	AXI monitor 9 filter settings
AXI_MON9_FILTER7	0x4ac	AXI monitor 9 filter settings
AXI_MON9_FILTER8	0x4b0	AXI monitor 9 filter settings
AXI_MON9_RPT0	0x4c0	AXI monitor 9 cycle count
AXI_MON9_RPT1	0x4c4	AXI monitor 9 hit count
AXI_MON9_RPT2	0x4c8	AXI monitor 9 byte count
AXI_MON9_RPT3	0x4cc	AXI monitor 9 latency count
AXI_MON10_CTRL	0x500	AXI monitor 10 control
AXI_MON10_INPUT	0x504	AXI monitor 10 input selection
AXI_MON10_FILTER0	0x510	AXI monitor 10 filter settings
AXI_MON10_FILTER1	0x514	AXI monitor 10 filter settings
AXI_MON10_FILTER2	0x518	AXI monitor 10 filter settings
AXI_MON10_FILTER3	0x51c	AXI monitor 10 filter settings
AXI_MON10_FILTER4	0x520	AXI monitor 10 filter settings
AXI_MON10_FILTER5	0x524	AXI monitor 10 filter settings
AXI_MON10_FILTER6	0x528	AXI monitor 10 filter settings
AXI_MON10_FILTER7	0x52c	AXI monitor 10 filter settings
AXI_MON10_FILTER8	0x530	AXI monitor 10 filter settings
AXI_MON10_RPT0	0x540	AXI monitor 10 cycle count
AXI_MON10_RPT1	0x544	AXI monitor 10 hit count
AXI_MON10_RPT2	0x548	AXI monitor 10 byte count
AXI_MON10_RPT3	0x54c	AXI monitor 10 latency count
AXI_MON11_CTRL	0x580	AXI monitor 11 control
AXI_MON11_INPUT	0x584	AXI monitor 11 input selection
AXI_MON11_FILTER0	0x590	AXI monitor 11 filter settings
AXI_MON11_FILTER1	0x594	AXI monitor 11 filter settings
AXI_MON11_FILTER2	0x598	AXI monitor 11 filter settings
AXI_MON11_FILTER3	0x59c	AXI monitor 11 filter settings
AXI_MON11_FILTER4	0x5a0	AXI monitor 11 filter settings
AXI_MON11_FILTER5	0x5a4	AXI monitor 11 filter settings
AXI_MON11_FILTER6	0x5a8	AXI monitor 11 filter settings
AXI_MON11_FILTER7	0x5ac	AXI monitor 11 filter settings
AXI_MON11_FILTER8	0x5b0	AXI monitor 11 filter settings
AXI_MON11_RPT0	0x5c0	AXI monitor 11 cycle count
AXI_MON11_RPT1	0x5c4	AXI monitor 11 hit count
AXI_MON11_RPT2	0x5c8	AXI monitor 11 byte count
AXI_MON11_RPT3	0x5cc	AXI monitor 11 latency count

14.1.5.2 AXI register description

base address 0x0800_4000

AXI_CTRL0_1

Offset Address: 0x4b4

Bits	Name	Access	Description	Reset

9:0	axi1_rd_timeout_val	R/W	After an AXI read transaction is granted, a timeout counter starts to count. When it counts to axi<n>_rd_timeout_val, the corresponding channel has the highest priority.	0x0

Bits	Name	Access	Description	Reset
11:10	Reserved			
12	axi1_rd_timeout_en	R/W	If set to 1, enables the timeout function for the read channel of port n.	0x0
31:13	Reserved			

AXI_CTRL1_1

Offset Address: 0x4b8

Bits	Name	Access	Description	Reset
9:0	axi1_wr_timeout_val	R/W	After an AXI write transaction is granted, a timeout counter starts to count. When it counts to axi<n>_wr_timeout_val, the corresponding channel has the highest priority.	0x0
11:10	Reserved			
12	axi1_wr_timeout_en	R/W	If set to 1, enables the timeout function for the write channel of port n.	0x0
31:13	Reserved			

AXI_CTRL0_2

Offset Address: 0x564

Bits	Name	Access	Description	Reset
9:0	axi2_rd_timeout_val	R/W	After an AXI read transaction is granted, a timeout counter starts to count. When it counts to axi<n>_rd_timeout_val, the corresponding channel has the highest priority.	0x0
11:10	Reserved			
12	axi2_rd_timeout_en	R/W	If set to 1, enables the timeout function for the read channel of port n.	0x0
31:13	Reserved			

AXI_CTRL1_2

Offset Address: 0x568

Bits	Name	Access	Description	Reset
9:0	axi2_wr_timeout_val	R/W	After an AXI write transaction is granted, a timeout counter starts to count. When it counts to axi<n>_wr_timeout_val, the corresponding channel has the highest priority.	0x0
11:10	Reserved			
12	axi2_wr_timeout_en	R/W	If set to 1, enables the timeout function for the write channel of port n.	0x0
31:13	Reserved			

AXI_CTRL0_3

Offset Address: 0x614

Bits	Name	Access	Description	Reset

9:0	axi3_rd_timeout_val	R/W	After an AXI read transaction is granted, a timeout counter starts to count. When it counts to axi<n>_rd_timeout_val, the corresponding channel has the highest	0x0

Bits	Name	Access	Description	Reset
			priority.	
11:10	Reserved			
12	axi3_rd_timeout_en	R/W	If set to 1, enables the timeout function for the read channel of port n.	0x0
31:13	Reserved			

AXI_CTRL1_3

Offset Address: 0x618

Bits	Name	Access	Description	Reset
9:0	axi3_wr_timeout_val	R/W	After an AXI write transaction is granted, a timeout counter starts to count. When it counts to axi<n>_wr_timeout_val, the corresponding channel has the highest priority.	0x0
11:10	Reserved			
12	axi3_wr_timeout_en	R/W	If set to 1, enables the timeout function for the write channel of port n.	0x0
31:13	Reserved			

Base address 0x0800_8000

AXI_MON0_CTRL

Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	axi_mon0_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon0_clear	R/W	Clear all the counter.	0x0
2	axi_mon0_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXIs monitors	0x0
4	axi_mon0_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon0_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon0_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON0_INPUT

Offset Address: 0x004

Bits	Name	Access	Description	Reset
5:0	axi_mon0_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON0_FILTER0

Offset Address: 0x010

Bits	Name	Access	Description	Reset
9:0	axi_mon0_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask	0x0

Bits	Name	Access	Description	Reset
			bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	
31:10	Reserved			

AXI_MON0_FILTER1

Offset Address: 0x014

Bits	Name	Access	Description	Reset
31:0	axi_mon0_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON0_FILTER2

Offset Address: 0x018

Bits	Name	Access	Description	Reset
7:0	axi_mon0_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON0_FILTER3

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
31:0	axi_mon0_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON0_FILTER4

Offset Address: 0x020

Bits	Name	Access	Description	Reset
7:0	axi_mon0_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON0_FILTER5

Offset Address: 0x024

Bits	Name	Access	Description	Reset
23:0	axi_mon0_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON0_FILTER6

Offset Address: 0x028

Bits	Name	Access	Description	Reset
23:0	axi_mon0_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON0_FILTER7

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
7:0	axi_mon0_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon0_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon0_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON0_FILTER8

Offset Address: 0x030

Bits	Name	Access	Description	Reset
0	axi_mon0_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon0_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon0_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon0_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON0_RPT0

Offset Address: 0x040

Bits	Name	Access	Description	Reset
31:0	axi_mon0_cycle_count	RO	AXI monitor 0 cycle count, counting after func_en assert	

AXI_MON0_RPT1

Offset Address: 0x044

Bits	Name	Access	Description	Reset
31:0	axi_mon0_hit_count	RO	AXI monitor 0 hit count, counting after func_en assert	

AXI_MON0_RPT2

Offset Address: 0x048

Bits	Name	Access	Description	Reset
31:0	axi_mon0_byte_count	RO	AXI monitor 0 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON0_RPT3

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
31:0	axi_mon0_latency_count	RO	AXI monitor 0 latency count, counting after func_en assert, += oustanding	

AXI_MON1_CTRL

Offset Address: 0x080

Bits	Name	Access	Description	Reset
0	axi_mon1_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon1_clear	R/W	Clear all the counter.	0x0
2	axi_mon1_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXIs monitors	0x0
4	axi_mon1_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0

Bits	Name	Access	Description	Reset
5	axi_mon1_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon1_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON1_INPUT

Offset Address: 0x084

Bits	Name	Access	Description	Reset
5:0	axi_mon1_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON1_FILTER0

Offset Address: 0x090

Bits	Name	Access	Description	Reset
9:0	axi_mon1_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	0x0
31:10	Reserved			

AXI_MON1_FILTER1

Offset Address: 0x094

Bits	Name	Access	Description	Reset
31:0	axi_mon1_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON1_FILTER2

Offset Address: 0x098

Bits	Name	Access	Description	Reset
7:0	axi_mon1_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON1_FILTER3

Offset Address: 0x09c

Bits	Name	Access	Description	Reset
31:0	axi_mon1_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON1_FILTER4

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
7:0	axi_mon1_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON1_FILTER5

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
23:0	axi_mon1_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON1_FILTER6

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
23:0	axi_mon1_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON1_FILTER7

Offset Address: 0x0ac

Bits	Name	Access	Description	Reset
7:0	axi_mon1_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon1_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon1_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON1_FILTER8

Offset Address: 0x0b0

Bits	Name	Access	Description	Reset
0	axi_mon1_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon1_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon1_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon1_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON1_RPT0

Offset Address: 0x0c0

Bits	Name	Access	Description	Reset
31:0	axi_mon1_cycle_count	RO	AXI monitor 1 cycle count, counting after func_en assert	

AXI_MON1_RPT1

Offset Address: 0x0c4

Bits	Name	Access	Description	Reset
31:0	axi_mon1_hit_count	RO	AXI monitor 1 hit count, counting after func_en assert	

AXI_MON1_RPT2

Offset Address: 0x0c8

Bits	Name	Access	Description	Reset
31:0	axi_mon1_byte_count	RO	AXI monitor 1 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON1_RPT3

Offset Address: 0x0cc

Bits	Name	Access	Description	Reset
31:0	axi_mon1_latency_count	RO	AXI monitor 1 latency count, counting after func_en assert, += oustanding	

AXI_MON2_CTRL

Offset Address: 0x100

Bits	Name	Access	Description	Reset
0	axi_mon2_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon2_clear	R/W	Clear all the counter.	0x0
2	axi_mon2_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXI monitors	0x0
4	axi_mon2_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon2_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon2_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON2_INPUT

Offset Address: 0x104

Bits	Name	Access	Description	Reset
5:0	axi_mon2_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON2_FILTER0

Offset Address: 0x110

Bits	Name	Access	Description	Reset
9:0	axi_mon2_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	0x0
31:10	Reserved			

AXI_MON2_FILTER1

Offset Address: 0x114

Bits	Name	Access	Description	Reset
31:0	axi_mon2_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON2_FILTER2

Offset Address: 0x118

Bits	Name	Access	Description	Reset
7:0	axi_mon2_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON2_FILTER3

Offset Address: 0x11c

Bits	Name	Access	Description	Reset
31:0	axi_mon2_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON2_FILTER4

Offset Address: 0x120

Bits	Name	Access	Description	Reset
7:0	axi_mon2_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON2_FILTER5

Offset Address: 0x124

Bits	Name	Access	Description	Reset
23:0	axi_mon2_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON2_FILTER6

Offset Address: 0x128

Bits	Name	Access	Description	Reset
23:0	axi_mon2_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON2_FILTER7

Offset Address: 0x12c

Bits	Name	Access	Description	Reset
7:0	axi_mon2_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon2_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon2_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON2_FILTER8

Offset Address: 0x130

Bits	Name	Access	Description	Reset
0	axi_mon2_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon2_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0

Bits	Name	Access	Description	Reset
10:8	axi_mon2_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon2_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON2_RPT0

Offset Address: 0x140

Bits	Name	Access	Description	Reset
31:0	axi_mon2_cycle_count	RO	AXI monitor 2 cycle count, counting after func_en assert	

AXI_MON2_RPT1

Offset Address: 0x144

Bits	Name	Access	Description	Reset
31:0	axi_mon2_hit_count	RO	AXI monitor 2 hit count, counting after func_en assert	

AXI_MON2_RPT2

Offset Address: 0x148

Bits	Name	Access	Description	Reset
31:0	axi_mon2_byte_count	RO	AXI monitor 2 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON2_RPT3

Offset Address: 0x14c

Bits	Name	Access	Description	Reset
31:0	axi_mon2_latency_count	RO	AXI monitor 2 latency count, counting after func_en assert, += oustanding	

AXI_MON3_CTRL

Offset Address: 0x180

Bits	Name	Access	Description	Reset
0	axi_mon3_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon3_clear	R/W	Clear all the counter.	0x0
2	axi_mon3_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXIs monitors	0x0
4	axi_mon3_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon3_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon3_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON3_INPUT

Offset Address: 0x184

Bits	Name	Access	Description	Reset
5:0	axi_mon3_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON3_FILTER0

Offset Address: 0x190

Bits	Name	Access	Description	Reset
9:0	axi_mon3_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	0x0
31:10	Reserved			

AXI_MON3_FILTER1

Offset Address: 0x194

Bits	Name	Access	Description	Reset
31:0	axi_mon3_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON3_FILTER2

Offset Address: 0x198

Bits	Name	Access	Description	Reset
7:0	axi_mon3_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON3_FILTER3

Offset Address: 0x19c

Bits	Name	Access	Description	Reset
31:0	axi_mon3_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON3_FILTER4

Offset Address: 0x1a0

Bits	Name	Access	Description	Reset
7:0	axi_mon3_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON3_FILTER5

Offset Address: 0x1a4

Bits	Name	Access	Description	Reset
23:0	axi_mon3_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON3_FILTER6

Offset Address: 0x1a8

Bits	Name	Access	Description	Reset
23:0	axi_mon3_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON3_FILTER7

Offset Address: 0x1ac

Bits	Name	Access	Description	Reset
7:0	axi_mon3_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon3_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon3_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON3_FILTER8

Offset Address: 0x1b0

Bits	Name	Access	Description	Reset
0	axi_mon3_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon3_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon3_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon3_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON3_RPT0

Offset Address: 0x1c0

Bits	Name	Access	Description	Reset
31:0	axi_mon3_cycle_count	RO	AXI monitor 3 cycle count, counting after func_en assert	

AXI_MON3_RPT1

Offset Address: 0x1c4

Bits	Name	Access	Description	Reset
31:0	axi_mon3_hit_count	RO	AXI monitor 3 hit count, counting after func_en assert	

AXI_MON3_RPT2

Offset Address: 0x1c8

Bits	Name	Access	Description	Reset
31:0	axi_mon3_byte_count	RO	AXI monitor 3 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON3_RPT3

Offset Address: 0x1cc

Bits	Name	Access	Description	Reset
31:0	axi_mon3_latency_count	RO	AXI monitor 3 latency count, counting after func_en assert, += oustanding	

AXI_MON4_CTRL

Offset Address: 0x200

Bits	Name	Access	Description	Reset
0	axi_mon4_en	R/W	If set to 1, enables the AXI monitor function.	0x0

Bits	Name	Access	Description	Reset
1	axi_mon4_clear	R/W	Clear all the counter.	0x0
2	axi_mon4_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXI monitors	0x0
4	axi_mon4_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon4_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon4_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON4_INPUT

Offset Address: 0x204

Bits	Name	Access	Description	Reset
5:0	axi_mon4_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON4_FILTER0

Offset Address: 0x210

Bits	Name	Access	Description	Reset
9:0	axi_mon4_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	0x0
31:10	Reserved			

AXI_MON4_FILTER1

Offset Address: 0x214

Bits	Name	Access	Description	Reset
31:0	axi_mon4_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON4_FILTER2

Offset Address: 0x218

Bits	Name	Access	Description	Reset
7:0	axi_mon4_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON4_FILTER3

Offset Address: 0x21c

Bits	Name	Access	Description	Reset
31:0	axi_mon4_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON4_FILTER4

Offset Address: 0x220

Bits	Name	Access	Description	Reset
7:0	axi_mon4_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON4_FILTER5

Offset Address: 0x224

Bits	Name	Access	Description	Reset
23:0	axi_mon4_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON4_FILTER6

Offset Address: 0x228

Bits	Name	Access	Description	Reset
23:0	axi_mon4_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON4_FILTER7

Offset Address: 0x22c

Bits	Name	Access	Description	Reset
7:0	axi_mon4_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon4_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon4_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON4_FILTER8

Offset Address: 0x230

Bits	Name	Access	Description	Reset
0	axi_mon4_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon4_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon4_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon4_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON4_RPT0

Offset Address: 0x240

Bits	Name	Access	Description	Reset
31:0	axi_mon4_cycle_count	RO	AXI monitor 4 cycle count, counting after func_en assert	

AXI_MON4_RPT1

Offset Address: 0x244

Bits	Name	Access	Description	Reset
31:0	axi_mon4_hit_count	RO	AXI monitor 4 hit count, counting after	

Bits	Name	Access	Description	Reset
			func_en assert	

AXI_MON4_RPT2

Offset Address: 0x248

Bits	Name	Access	Description	Reset
31:0	axi_mon4_byte_count	RO	AXI monitor 4 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON4_RPT3

Offset Address: 0x24c

Bits	Name	Access	Description	Reset
31:0	axi_mon4_latency_count	RO	AXI monitor 4 latency count, counting after func_en assert, += oustanding	

AXI_MON5_CTRL

Offset Address: 0x280

Bits	Name	Access	Description	Reset
0	axi_mon5_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon5_clear	R/W	Clear all the counter.	0x0
2	axi_mon5_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXI monitors	0x0
4	axi_mon5_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon5_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon5_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON5_INPUT

Offset Address: 0x284

Bits	Name	Access	Description	Reset
5:0	axi_mon5_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON5_FILTER0

Offset Address: 0x290

Bits	Name	Access	Description	Reset
9:0	axi_mon5_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	0x0
31:10	Reserved			

AXI_MON5_FILTER1

Offset Address: 0x294

Bits	Name	Access	Description	Reset
31:0	axi_mon5_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON5_FILTER2

Offset Address: 0x298

Bits	Name	Access	Description	Reset
7:0	axi_mon5_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON5_FILTER3

Offset Address: 0x29c

Bits	Name	Access	Description	Reset
31:0	axi_mon5_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON5_FILTER4

Offset Address: 0x2a0

Bits	Name	Access	Description	Reset
7:0	axi_mon5_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON5_FILTER5

Offset Address: 0x2a4

Bits	Name	Access	Description	Reset
23:0	axi_mon5_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON5_FILTER6

Offset Address: 0x2a8

Bits	Name	Access	Description	Reset
23:0	axi_mon5_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON5_FILTER7

Offset Address: 0x2ac

Bits	Name	Access	Description	Reset
7:0	axi_mon5_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon5_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon5_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON5_FILTER8

Offset Address: 0x2b0

Bits	Name	Access	Description	Reset
0	axi_mon5_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0

Bits	Name	Access	Description	Reset
3:1	Reserved			
7:4	axi_mon5_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon5_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon5_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON5_RPT0

Offset Address: 0x2c0

Bits	Name	Access	Description	Reset
31:0	axi_mon5_cycle_count	RO	AXI monitor 5 cycle count, counting after func_en assert	

AXI_MON5_RPT1

Offset Address: 0x2c4

Bits	Name	Access	Description	Reset
31:0	axi_mon5_hit_count	RO	AXI monitor 5 hit count, counting after func_en assert	

AXI_MON5_RPT2

Offset Address: 0x2c8

Bits	Name	Access	Description	Reset
31:0	axi_mon5_byte_count	RO	AXI monitor 5 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON5_RPT3

Offset Address: 0x2cc

Bits	Name	Access	Description	Reset
31:0	axi_mon5_latency_count	RO	AXI monitor 5 latency count, counting after func_en assert, += oustanding	

AXI_MON6_CTRL

Offset Address: 0x300

Bits	Name	Access	Description	Reset
0	axi_mon6_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon6_clear	R/W	Clear all the counter.	0x0
2	axi_mon6_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXIs monitors	0x0
4	axi_mon6_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon6_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon6_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON6_INPUT

Offset Address: 0x304

Bits	Name	Access	Description	Reset
5:0	axi_mon6_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0

Bits	Name	Access	Description	Reset
31:6	Reserved			

AXI_MON6_FILTER0

Offset Address: 0x310

Bits	Name	Access	Description	Reset
9:0	axi_mon6_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	0x0
31:10	Reserved			

AXI_MON6_FILTER1

Offset Address: 0x314

Bits	Name	Access	Description	Reset
31:0	axi_mon6_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON6_FILTER2

Offset Address: 0x318

Bits	Name	Access	Description	Reset
7:0	axi_mon6_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON6_FILTER3

Offset Address: 0x31c

Bits	Name	Access	Description	Reset
31:0	axi_mon6_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON6_FILTER4

Offset Address: 0x320

Bits	Name	Access	Description	Reset
7:0	axi_mon6_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON6_FILTER5

Offset Address: 0x324

Bits	Name	Access	Description	Reset
23:0	axi_mon6_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON6_FILTER6

Offset Address: 0x328

Bits	Name	Access	Description	Reset
23:0	axi_mon6_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON6_FILTER7

Offset Address: 0x32c

Bits	Name	Access	Description	Reset
7:0	axi_mon6_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon6_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon6_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON6_FILTER8

Offset Address: 0x330

Bits	Name	Access	Description	Reset
0	axi_mon6_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon6_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon6_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon6_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON6_RPT0

Offset Address: 0x340

Bits	Name	Access	Description	Reset
31:0	axi_mon6_cycle_count	RO	AXI monitor 6 cycle count, counting after func_en assert	

AXI_MON6_RPT1

Offset Address: 0x344

Bits	Name	Access	Description	Reset
31:0	axi_mon6_hit_count	RO	AXI monitor 6 hit count, counting after func_en assert	

AXI_MON6_RPT2

Offset Address: 0x348

Bits	Name	Access	Description	Reset
31:0	axi_mon6_byte_count	RO	AXI monitor 6 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON6_RPT3

Offset Address: 0x34c

Bits	Name	Access	Description	Reset
31:0	axi_mon6_latency_count	RO	AXI monitor 6 latency count, counting after func_en assert, += oustanding	

AXI_MON7_CTRL

Offset Address: 0x380

Bits	Name	Access	Description	Reset
0	axi_mon7_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon7_clear	R/W	Clear all the counter.	0x0
2	axi_mon7_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXIs monitors	0x0
4	axi_mon7_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon7_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon7_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON7_INPUT

Offset Address: 0x384

Bits	Name	Access	Description	Reset
5:0	axi_mon7_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON7_FILTER0

Offset Address: 0x390

Bits	Name	Access	Description	Reset
9:0	axi_mon7_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	0x0
31:10	Reserved			

AXI_MON7_FILTER1

Offset Address: 0x394

Bits	Name	Access	Description	Reset
31:0	axi_mon7_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON7_FILTER2

Offset Address: 0x398

Bits	Name	Access	Description	Reset
7:0	axi_mon7_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON7_FILTER3

Offset Address: 0x39c

Bits	Name	Access	Description	Reset
31:0	axi_mon7_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON7_FILTER4

Offset Address: 0x3a0

Bits	Name	Access	Description	Reset
7:0	axi_mon7_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON7_FILTER5

Offset Address: 0x3a4

Bits	Name	Access	Description	Reset
23:0	axi_mon7_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON7_FILTER6

Offset Address: 0x3a8

Bits	Name	Access	Description	Reset
23:0	axi_mon7_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON7_FILTER7

Offset Address: 0x3ac

Bits	Name	Access	Description	Reset
7:0	axi_mon7_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon7_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon7_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON7_FILTER8

Offset Address: 0x3b0

Bits	Name	Access	Description	Reset
0	axi_mon7_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon7_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon7_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon7_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON7_RPT0

Offset Address: 0x3c0

Bits	Name	Access	Description	Reset
31:0	axi_mon7_cycle_count	RO	AXI monitor 7 cycle count, counting after func_en assert	

AXI_MON7_RPT1

Offset Address: 0x3c4

Bits	Name	Access	Description	Reset
31:0	axi_mon7_hit_count	RO	AXI monitor 7 hit count, counting after func_en assert	

AXI_MON7_RPT2

Offset Address: 0x3c8

Bits	Name	Access	Description	Reset
31:0	axi_mon7_byte_count	RO	AXI monitor 7 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON7_RPT3

Offset Address: 0x3cc

Bits	Name	Access	Description	Reset
31:0	axi_mon7_latency_count	RO	AXI monitor 7 latency count, counting after func_en assert, += oustanding	

AXI_MON8_CTRL

Offset Address: 0x400

Bits	Name	Access	Description	Reset
0	axi_mon8_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon8_clear	R/W	Clear all the counter.	0x0
2	axi_mon8_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXIs monitors	0x0
4	axi_mon8_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon8_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon8_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON8_INPUT

Offset Address: 0x404

Bits	Name	Access	Description	Reset
5:0	axi_mon8_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON8_FILTER0

Offset Address: 0x410

Bits	Name	Access	Description	Reset
9:0	axi_mon8_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	0x0

Bits	Name	Access	Description	Reset
31:10	Reserved			

AXI_MON8_FILTER1

Offset Address: 0x414

Bits	Name	Access	Description	Reset
31:0	axi_mon8_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON8_FILTER2

Offset Address: 0x418

Bits	Name	Access	Description	Reset
7:0	axi_mon8_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON8_FILTER3

Offset Address: 0x41c

Bits	Name	Access	Description	Reset
31:0	axi_mon8_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON8_FILTER4

Offset Address: 0x420

Bits	Name	Access	Description	Reset
7:0	axi_mon8_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON8_FILTER5

Offset Address: 0x424

Bits	Name	Access	Description	Reset
23:0	axi_mon8_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON8_FILTER6

Offset Address: 0x428

Bits	Name	Access	Description	Reset
23:0	axi_mon8_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON8_FILTER7

Offset Address: 0x42c

Bits	Name	Access	Description	Reset
7:0	axi_mon8_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon8_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon8_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON8_FILTER8

Offset Address: 0x430

Bits	Name	Access	Description	Reset
0	axi_mon8_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon8_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon8_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon8_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON8_RPT0

Offset Address: 0x440

Bits	Name	Access	Description	Reset
31:0	axi_mon8_cycle_count	RO	AXI monitor 8 cycle count, counting after func_en assert	

AXI_MON8_RPT1

Offset Address: 0x444

Bits	Name	Access	Description	Reset
31:0	axi_mon8_hit_count	RO	AXI monitor 8 hit count, counting after func_en assert	

AXI_MON8_RPT2

Offset Address: 0x448

Bits	Name	Access	Description	Reset
31:0	axi_mon8_byte_count	RO	AXI monitor 8 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON8_RPT3

Offset Address: 0x44c

Bits	Name	Access	Description	Reset
31:0	axi_mon8_latency_count	RO	AXI monitor 8 latency count, counting after func_en assert, += oustanding	

AXI_MON9_CTRL

Offset Address: 0x480

Bits	Name	Access	Description	Reset
0	axi_mon9_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon9_clear	R/W	Clear all the counter.	0x0
2	axi_mon9_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXIs monitors	0x0
4	axi_mon9_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon9_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon9_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON9_INPUT

Offset Address: 0x484

Bits	Name	Access	Description	Reset
5:0	axi_mon9_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON9_FILTER0

Offset Address: 0x490

Bits	Name	Access	Description	Reset
9:0	axi_mon9_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	0x0
31:10	Reserved			

AXI_MON9_FILTER1

Offset Address: 0x494

Bits	Name	Access	Description	Reset
31:0	axi_mon9_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON9_FILTER2

Offset Address: 0x498

Bits	Name	Access	Description	Reset
7:0	axi_mon9_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON9_FILTER3

Offset Address: 0x49c

Bits	Name	Access	Description	Reset
31:0	axi_mon9_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON9_FILTER4

Offset Address: 0x4a0

Bits	Name	Access	Description	Reset
7:0	axi_mon9_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON9_FILTER5

Offset Address: 0x4a4

Bits	Name	Access	Description	Reset
23:0	axi_mon9_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON9_FILTER6

Offset Address: 0x4a8

Bits	Name	Access	Description	Reset
23:0	axi_mon9_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON9_FILTER7

Offset Address: 0x4ac

Bits	Name	Access	Description	Reset
7:0	axi_mon9_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon9_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon9_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON9_FILTER8

Offset Address: 0x4b0

Bits	Name	Access	Description	Reset
0	axi_mon9_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon9_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon9_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon9_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON9_RPT0

Offset Address: 0x4c0

Bits	Name	Access	Description	Reset
31:0	axi_mon9_cycle_count	RO	AXI monitor 9 cycle count, counting after func_en assert	

AXI_MON9_RPT1

Offset Address: 0x4c4

Bits	Name	Access	Description	Reset
31:0	axi_mon9_hit_count	RO	AXI monitor 9 hit count, counting after func_en assert	

AXI_MON9_RPT2

Offset Address: 0x4c8

Bits	Name	Access	Description	Reset
31:0	axi_mon9_byte_count	RO	AXI monitor 9 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON9_RPT3

Offset Address: 0x4cc

Bits	Name	Access	Description	Reset
31:0	axi_mon9_latency_count	RO	AXI monitor 9 latency count, counting after func_en assert, += oustanding	

AXI_MON10_CTRL

Offset Address: 0x500

Bits	Name	Access	Description	Reset
0	axi_mon10_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon10_clear	R/W	Clear all the counter.	0x0
2	axi_mon10_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXIs monitors	0x0
4	axi_mon10_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon10_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon10_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON10_INPUT

Offset Address: 0x504

Bits	Name	Access	Description	Reset
5:0	axi_mon10_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON10_FILTER0

Offset Address: 0x510

Bits	Name	Access	Description	Reset
9:0	axi_mon10_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB boundary	0x0
31:10	Reserved			

AXI_MON10_FILTER1

Offset Address: 0x514

Bits	Name	Access	Description	Reset
31:0	axi_mon10_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON10_FILTER2

Offset Address: 0x518

Bits	Name	Access	Description	Reset
7:0	axi_mon10_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON10_FILTER3

Offset Address: 0x51c

Bits	Name	Access	Description	Reset
31:0	axi_mon10_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON10_FILTER4

Offset Address: 0x520

Bits	Name	Access	Description	Reset
7:0	axi_mon10_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON10_FILTER5

Offset Address: 0x524

Bits	Name	Access	Description	Reset
23:0	axi_mon10_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON10_FILTER6

Offset Address: 0x528

Bits	Name	Access	Description	Reset
23:0	axi_mon10_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON10_FILTER7

Offset Address: 0x52c

Bits	Name	Access	Description	Reset
7:0	axi_mon10_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon10_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon10_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON10_FILTER8

Offset Address: 0x530

Bits	Name	Access	Description	Reset
0	axi_mon10_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon10_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon10_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon10_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON10_RPT0

Offset Address: 0x540

Bits	Name	Access	Description	Reset
31:0	axi_mon10_cycle_count	RO	AXI monitor 10 cycle count, counting after func_en assert	

AXI_MON10_RPT1

Offset Address: 0x544

Bits	Name	Access	Description	Reset
31:0	axi_mon10_hit_count	RO	AXI monitor 10 hit count, counting after func_en assert	

AXI_MON10_RPT2

Offset Address: 0x548

Bits	Name	Access	Description	Reset
31:0	axi_mon10_byte_count	RO	AXI monitor 10 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON10_RPT3

Offset Address: 0x54c

Bits	Name	Access	Description	Reset
31:0	axi_mon10_latency_count	RO	AXI monitor 10 latency count, counting after func_en assert, += oustanding	

AXI_MON11_CTRL

Offset Address: 0x580

Bits	Name	Access	Description	Reset
0	axi_mon11_en	R/W	If set to 1, enables the AXI monitor function.	0x0
1	axi_mon11_clear	R/W	Clear all the counter.	0x0
2	axi_mon11_snapshot	R/W	Snapshot all the counter.	0x0
3	axi_mon0_snapshot_all	R/W	Snapshot all the counter of all AXIs monitors	0x0
4	axi_mon11_irq_en	R/W	If set to 1, enables the AXI monitor interrupt.	0x0
5	axi_mon11_irq_clear	R/W	If set to 1, clears the axi_mon_irq.	0x0
6	Reserved			
7	axi_mon11_irq	RO	Assert when all axi_mon<n>_hit_sel suscces.	
31:8	Reserved			

AXI_MON11_INPUT

Offset Address: 0x584

Bits	Name	Access	Description	Reset
5:0	axi_mon11_input_sel	R/W	Input/clk selection, 0 = No selection.	0x0
31:6	Reserved			

AXI_MON11_FILTER0

Offset Address: 0x590

Bits	Name	Access	Description	Reset
9:0	axi_mon11_hit_sel	R/W	Select which conditions are used to judege hit bit [0]: addr_st/addr_sp bit [1]: id/id_mask bit [2]: len bit [3]: size bit [4]: burst bit [5]: lock bit [6]: cache bit [7]: prot bit [8]: qos bit [9]: AXI transaction cross 4KB	0x0

Bits	Name	Access	Description	Reset
			boundary	
31:10	Reserved			

AXI_MON11_FILTER1

Offset Address: 0x594

Bits	Name	Access	Description	Reset
31:0	axi_mon11_hit_addr_st_lo	R/W	Hit start address[31:0] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0

AXI_MON11_FILTER2

Offset Address: 0x598

Bits	Name	Access	Description	Reset
7:0	axi_mon11_hit_addr_st_hi	R/W	Hit start address [39:32] hit = (Ax_addr >= hit_addr_st) && (Ax_addr < hit_addr_sp)	0x0
31:8	Reserved			

AXI_MON11_FILTER3

Offset Address: 0x59c

Bits	Name	Access	Description	Reset
31:0	axi_mon11_hit_addr_sp_lo	R/W	Hit end address[31:0]	0x0

AXI_MON11_FILTER4

Offset Address: 0x5a0

Bits	Name	Access	Description	Reset
7:0	axi_mon11_hit_addr_sp_hi	R/W	Hit end address[39:32]	0x0
31:8	Reserved			

AXI_MON11_FILTER5

Offset Address: 0x5a4

Bits	Name	Access	Description	Reset
23:0	axi_mon11_hit_id_mask	R/W	hit = (hit_id & hit_id_mask) == (Ax_id & hit_id_mask)	0x0
31:24	Reserved			

AXI_MON11_FILTER6

Offset Address: 0x5a8

Bits	Name	Access	Description	Reset
23:0	axi_mon11_hit_id	R/W	hit = Ax_id == hit_id	0x0
31:24	Reserved			

AXI_MON11_FILTER7

Offset Address: 0x5ac

Bits	Name	Access	Description	Reset
7:0	axi_mon11_hit_len	R/W	hit = Ax_len == hit_len	0x0
10:8	axi_mon11_hit_size	R/W	hit = Ax_size == hit_size	0x0
11	Reserved			
13:12	axi_mon11_hit_burst	R/W	hit = Ax_burst == hit_burst	0x0
31:14	Reserved			

AXI_MON11_FILTER8

Offset Address: 0x5b0

Bits	Name	Access	Description	Reset
0	axi_mon11_hit_lock	R/W	hit = Ax_lock == hit_lock	0x0
3:1	Reserved			
7:4	axi_mon11_hit_cache	R/W	hit = Ax_cache == hit_cache	0x0
10:8	axi_mon11_hit_prot	R/W	hit = Ax_prot == hit_prot	0x0
11	Reserved			
15:12	axi_mon11_hit_qos	R/W	hit = Ax_qos == hit_qos	0x0
31:16	Reserved			

AXI_MON11_RPT0

Offset Address: 0x5c0

Bits	Name	Access	Description	Reset
31:0	axi_mon11_cycle_count	RO	AXI monitor 11 cycle count, counting after func_en assert	

AXI_MON11_RPT1

Offset Address: 0x5c4

Bits	Name	Access	Description	Reset
31:0	axi_mon11_hit_count	RO	AXI monitor 11 hit count, counting after func_en assert	

AXI_MON11_RPT2

Offset Address: 0x5c8

Bits	Name	Access	Description	Reset
31:0	axi_mon11_byte_count	RO	AXI monitor 11 byte count, counting after func_en assert, (Ax_len + 1) << Ax_size	

AXI_MON11_RPT3

Offset Address: 0x5cc

Bits	Name	Access	Description	Reset
31:0	axi_mon11_latency_count	RO	AXI monitor 11 latency count, counting after func_en assert, += oustanding	

14.1.6 DDRC Register

14.1.6.1 DDRC Register Overview

Base address 0x0800_4000

Name	Address Offset	Description
DRAM_REF_CTRL	0x064	DRAM refresh parameter
DRAM_MRD0	0x0dc	DRAM MR value
DRAM_MRD1	0x0e0	DRAM MR value

14.1.6.2 DDRC Register Description

Base address 0x0800_4000

DRAM_REF_CTRL

Offset Address: 0x064

Bits	Name	Access	Description	Reset
9:0	t_rfc	R/W	Specify tRFC Unit: ddr core clock cycles	0x8c
15:10	Reserved			
27:16	t_refi	R/W	Specify tREFI Unit: 32 ddr core clocks	0x62
31:28	Reserved			

DRAM_MRD0

Offset Address: 0x0dc

Bits	Name	Access	Description	Reset
15:0	ddr_mr1	R/W	DDR3: Write value for MR1 register	0x510
31:16	ddr_mr0	R/W	DDR3: Write value for MR0 register	0x0

DRAM_MRD1

Offset Address: 0x0e0

Bits	Name	Access	Description	Reset
15:0	ddr_mr3	R/W	DDR3: Write value for MR3 register	0x0
31:16	ddr_mr2	R/W	DDR3: Write value for MR2 register	0x0

14.2 SPI NOR Flash Controller

14.2.1 summarize

Provides off-chip SPI NOR Flash data access.

14.2.2 specificities

- Supports external 1-chip selection.
- Supports Dual/Qual read and write operations.
- Supports a wide range of device specifications.
  - Supports 3Byte address devices and 4Byte address devices.
  - The maximum capacity of the device is 256MB at the time of support.
- Supports BOOT function.

14.2.3 Functional Description

14.2.3.1 interface description

The SPI NOR Flash controller can support three SPI NOR interface types, the types are Standard SPI, Dual SPI interface mode and Qual SPI interface mode.

- Standard SPI Interface Mode.

Standard SPI interface mode has 1bit data input line and 1bit data output line. The diagram shows the timing diagram for Standard SPI interface mode write operation and the timing diagram for Standard SPI interface mode read operation.

Figure *14-2 Standard SPI* Interface Mode Write Operation Timing Chart

Timing

Notes:

- ✓ command/address/dummy cycles are output on the DO line in single bit serial mode.
- ✓ Data is output on the DO line in single-bit serial mode.

Figure *14-3 Standard SPI* Interface Mode Read Operation Timing Chart

Timing Notes:

- ✓ command/address/dummy cycles are output on the DO line in single bit serial mode.
- ✓ Data is entered on the DI line in single bit serial mode.

- Dual-Input SPI Interface Mode.

Dual Input SPI interface mode with parallel 2bit data lines in the data input stage. Figure 4-4 shows the Dual Input SPI interface mode operation timing diagram.

Figure *14-4 Dual-Input SPI* Interface Timing

Timing Notes:

- ✓ Command/address/dummy cycles are output on the DO line in single bit serial mode.
- ✓ Data in 2 Bits on DO/DI lines or input (read)

- Dual-IO SPI Interface Mode.

Dual IO SPI interface mode with parallel 2bit data lines for address output and data input stages. Figure 4-4 shows the Dual IO SPI
Timing diagram for interface mode operation.

Figure *14-5 Dual-IO SPI* Interface Timing

Timing Notes:

- ✓ Command is output on the DO line in single bit serial mode.
- ✓ address/dummy cycles/Data are output (written) or input on DO/DI lines in 2 Bits.

  (Reading)

- Quad-Input SPI Interface Mode.

Quad Input SPI interface mode with parallel 4bit data lines in the data input stage. Figure 4-6 shows the Quad Input SPI interface mode operation timing diagram.

Figure *14-6 Quad-Input SPI* Mode Timing Chart

Timing Notes:

- ✓ Command/address/dummy cycles are output on the DO line in single bit serial mode.
- ✓ Data is input (read) at DO/DI/WPN/HOLDN in 4 Bits.

- Quad-IO SPI Interface Mode.

Quad IO SPI interface mode, address output, data input stage parallel 4bit data

lines. Figure 4-7 shows the Quad IO SPI interface mode operation timing diagram.

Figure *14-7 Quad-IO SPI* Mode Timing Chart

Timing Notes:

- ✓ Command is output on the DO line in single bit serial mode.
- ✓ address/dummy cycles/Data is output (write) or input (read) in DO/DI/WPN/HOLDN in 4 Bits.

14.2.3.2 Boot Function

SPI NOR Boot data is located at chip address 0x1000_0000~0x1FFF_FFFF, which is directly mapped to SPI NOR Flash contiguous address space 0x0000_0000~0x0FFF_FFFF. SPI_NOR Flash can support up to 256MB, in which if you need to use more than 16MB SPI_NOR Flash, you need to utilize 4 bytes address mode. NOR Flash, you need to utilize the 4 bytes address mode. SPI_NOR Flash can support up to 256MB, in which 4 bytes address mode is required to use SPI_NOR Flash larger than 16MB. The reset state of the chip is 3bytes address mode, and 4bytes address mode needs to be enabled through the configuration.

3bytes/4bytes Address Mode.

14.2.3.3 register operation

The software configures the operation related registers, such as operation command, address, etc. Finally, the reg_go_busy register is configured to send down the operation, and the controller sends down the operation to the device according to the software configuration value.

14.2.3.4 DMA operation

- DMMR Read Mode

When the SPI_NOR Flash Controller is in DMMR mode, the SPI_NOR Flash space is

directly mapped to the chip address space 0x1000_0000~0x1FFF_FFFF. The system DMA can be used in memory-to-memory mode by mapping the SPI_NOR Data is moved to DDR.

- Non-DMMR Read/Write Mode

Instructions, addresses and data must be sent and received via FF_PORT. You need to configure the controller registers to select the read instruction, write instruction, instruction length and data length, and then write FF_PORT through the CPU or DMA to send out the instruction and address, and then write FF_PORT through the CPU or DMA to send out the instruction and address.

algorithmic technology

Write and read FF_PORT to send and receive data.

14.2.4 workflow

14.2.4.1 initialization process

If you need to adjust the Timing parameters, configure the SPI Clock Divider according to the device. step 2. configure the Interrupt Control Register.

14.2.4.2 Device Status Register Operation

Step 1. configure the transmission data length. Step 2. transmission mode-related configuration. Step 3. configure reg_go_busy.
Step 4. Write the transmission to the cache.

Step 5. detect INT_STS and wait for the operation to complete.

14.2.4.3 SPI NOR Flash Address Mode Switching Flow

For SPI NOR Flash devices, both 3 Byte and 4 Byte Flash address modes are supported, and the address mode can be switched dynamically after the device is booted up through the configuration registers. To switch the Flash address mode after the device has booted up, proceed as follows.

Step 1. No Flash operation or ensure that the previous operation of the Flash device is complete.

Step 2. Configure the relevant registers of the device with register operations according to the device requirements and send specific commands to configure the Flash into 4 Byte mode.

Step 3. Configure the SPI NOR Flash controller [reg_byte4en] style to 4 Byte mode to complete the 3 Byte mode to 4 Byte mode switch.

14.2.4.4 DMA read operation flow

Step 1. Disable dmmr mode and disable dma_en.

Step 2. Write FF_PT to 1 to clear the FIFO and reset the read/write indicator. Step 3. Enable dmmr mode.

Step 4. Configure the system DMA for mem-to-mem relocation. DST_TR_WIDTH = 0x2 (transaction width is 32bit), DST_MSIZE = 0x0 (burst transaction length =1), BLOCK_TS = TRAN_NUM/4 -1 . In fact, BLOCK_TS/ DST_TR_WIDTH/ DST_MSIZE should be configured according to the actual transmission length.

Step 5. Enable the system DMA selected channel. Begin moving

Step 6. Wait for the DMA corresponding channel interrupt. Indicates that the DMA read is complete.

14.2.4.5 DMA Write Operation Flow

Step 1. Disable dmmr mode and disable dma_en.
Step 2. Write FF_PT to 1 to clear the FIFO and reset the read/write indicator.
Step 3. Configure the system DMA channel mapping to map the selected DMA channel to the 39:dma_req_spi_nor.
Step 4. Configure the system DMA for peri-to-mem relocation. DST_TR_WIDTH = 0x2 (transaction width is 32bit), DST_MSIZE = 0x0 (burst transaction length =1), BLOCK_TS = TRAN_NUM/4 -1 . In fact, BLOCK_TS/ DST_TR_WIDTH/ DST_MSIZE should be configured according to the actual transmission length.
Step 5. Enable the system DMA selected channel.
Step 6. Configure SPI_NOR register TRAN_NUM, without instruction and address
Step 7. Configure SPI_NOR registers TRAN_CSR, tran_mode = 0x2 (tx only), fast_mode, bus_width, addr_bn, dma_en=0, and reg_go_busy. Ex: TRAN_CSR = 0x0000BC2A
Step 8. Write command and address to SPI_NOR register FF_PORT.
Step 9. Query SPI_NOR register rdat_ff_pt to 0 , make sure command and address are sent.
Step 10. configure SPI_NOR register TRAN_CSR to enable dma_en.
Step 11. detect SPI_NOR register INT_STS and wait for the operation to complete, indicating that the buffer contents have been written to the device.

14.2.4.6 Other Precautions

- Do not change the related register configuration before the operation of the device is completed, or the operation may be abnormal.

14.2.5 Registers Overview

Base Address 0x10000000

Name	Address Offset	Description
SPI_CTRL	**0x000**	**SPI_NOR** Operation Control

CE_CTRL	0x004	**CE** Operational Control
DLY_CTRL	0x008	latency control
DMMR_CTRL	0x00c	**DMMR** Mode Control
TRAN_CSR	0x010	transmission control

Name	Address Offset	Description
TRAN_NUM	0x014	transmission frame rate
FF_PORT	0x018	**FIFO** Write Read Export
FF_PT	0x020	**FIFO** Indicator Status
INT_STS	0x028	interrupt status
INT_EN	0x02c	interrupt enable

14.2.6 register description

SPI_CTRL

Offset Address: 0x000

Bits	Name	Access	Description	Reset
10:0	sck_div	R/W	SPI Clock Divider SCK frequency = HCLK frequency / (2(SckDiv+ 1))	0x9
11	Reserved			
12	cpha	R/W	Clock Phase 0: Start sampling data at the first clock edge of SCK after chip select is active. 1: Open on the second clock edge of SCK after chip select is active Initial sampling data	0x0
13	cpol	R/W	Clock Polarity 0: SCK is low when idle 1: SCK is high when idle	0x0
14	hold_o	R/W	HOLD pin output level	0x1
15	wp_o	R/W	WP pin output level	0x1
19:16	frame_len	R/W	Transmit and receive frame lengths. 0 indicates the frame length. is 16 bits; a frame length of 1 is not supported.	0x8
20	lsb_first	R/W	LSBF: Least Significant Bit First 0: Frame MSB first 1: Frame LSB first	0x0
21	srst	R/W	Write 1 Reset each state machine and interrupt flag bit	0x0
31:22	Reserved			

CE_CTRL

Offset Address: 0x004

Bits	Name	Access	Description	Reset
0	ce_manual	R/W	CEManual controls the level of the CE pin	0x0

1	ce_manual_en	R/W	CE Manual Enable 0: The level of pin CE is controlled by the hardware state machine. 1: The level of pin CE is regulated by the CEManual register. machine control	0x0
31:2	Reserved			

DLY_CTRL

Offset Address: 0x008

Bits	Name	Access	Description	Reset
3:0	frame_interval	R/W	Controls the frame spacing between two adjacent frames of data: T = TSCK * FmIntvl (no SCK pulses within the frame spacing) Frame spacing between two adjacent frames of data, 0 No frame spacing when.	0x0
7:4	Reserved			
11:8	cet	R/W	CET controls the amount of time that CE is valid in advance relative to the first clock edge of SCK before the start of a transmission and the amount of time that CE remains valid relative to the last clock edge of SCK after the end of the transmission. The time is calculated as T = TSCK * (CET+1)	0x3
13:12	smp_en_dly	R/W	Receive sample delay option. After the positive edge of SCK Delay sample cycle (IP working clock) to enter the line sampling	0x0
14	rx_pipe_ctrl	R/W	Receive Sample Clock Edge option. 0: Normal sampling 1: SCK negative edge sampling for high speed transmission	0x0
31:15	Reserved			

DMMR_CTRL

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
0	dmmr_mode	R/W	When this bit is 1, the read address on the AHB will be directly mapped to the SPI Flash, and the controller will automatically read the data from the corresponding address of the SPI Flash without the need of software to set the relevant commands and addresses. In this case, the SPI Flash can be used as ROM. When 1, the registers in IP are writable but not readable.	0x1
31:1	Reserved			

TRAN_CSR

Offset Address: 0x010

Bits	Name	Access	Description	Reset

1:0	tran_mode	R/W	Transfer Mode 00: No Tx, No Rx 01: Rx only 10: Tx only 11: Tx and Rx TranMode Indicates transfers other than commands and addresses The mode of sending and receiving data.	0x0
2	Reserved			
3	fast_mode	R/W	FastMode. 0: Normal Mode 1: Fast Mode	0x0
5:4	bus_width	R/W	Bus Width 00: 1 bit bus 01: 2 bit bus 10: 4 bit bus 11: Reserved	0x0
6	dma_en	R/W	0: DMA Disable 1: DMA Enable When TranMode is 11 (transmit and receive at the same time) DMA transfer is not supported.	0x0
7	miso_cked	R/W	Level value of the miso_i pin	0x0

Bits	Name	Access	Description	Reset
10:8	addr_bn	R/W	Address Byte Number Indicates the byte of the current Flash Transmit Address field Number, 0 means no address field	0x3
11	with_cmd	R/W	With Command 0: Currently transmitting without commands 1: Current transmission band command	0x1
13:12	ff_trg_lvl	R/W	FFTrgLvl Controls the conditions under which the FIFO generates interrupts and DMA requests. 00: 1 Byte 01: 2 Bytes 10: 4 Bytes 11: 8 Bytes For Transmit, interrupts and DMA requests are generated when the number of free Bytes in the FIFO is greater than or equal to the number of Bytes defined by FFTrgLvl; For Receive, it is generated when the number of valid Bytes in the FIFO is greater than or equal to the number of Bytes defined by FFTrgLvl. The interrupts and DMA requests are generated.	0x3
14	Reserved			
15	go_busy	R/W	Writing a 0 to this bit has no effect; writing a 1 sets this bit to 0. 1 and initiates a transmission; this bit is automatically cleared to zero when the transmission is completed. Before initiating a new transmission, the software should query this register, which is 0 before it can Initiate a new transmission.	0x0
19:16	dummy_cyc	R/W	dummy cycle count	0x0
20	byte4en	R/W	4 bytes address cycle enable in dmmr_mode	0x0
21	byte4cmd	R/W	4 bytes address cmd enable in dmmr_mode	0x0
31:22	Reserved			

TRAN_NUM

Offset Address: 0x014

Bits	Name	Access	Description	Reset
15:0	rdat_tran_num	R/W	TRAN_NUM is one time in non-dmmr_mode. Number of frames sent and received in transmission	0x0
31:16	Reserved			

FF_PORT

Offset Address: 0x018

Bits	Name	Access	Description	Reset
31:0	rdat_ff_port	R/W	FIFO Write Read Address	0x0

FF_PT

Offset Address: 0x020

Bits	Name	Access	Description	Reset
3:0	rdat_ff_pt	R/W	Read is the number of valid data bytes in the FIFO, write is the number of FIFO Empty	0x0
7:4	Reserved			
9:8	wrcnt	R/W	Current fifo , write byte offset indicator status	0x0
12:10	rdpt	R/W	Current fifo ,read byte offset indicator status	0x0
31:13	Reserved			

INT_STS

Offset Address: 0x028

Bits	Name	Access	Description	Reset
0	tran_done_int	R/W	This interrupt is generated once for each successfully sent frame.	0x0
1	Reserved			
2	rdff_int	R/W	Generate this interrupt once for each successfully received frame.	0x0
3	wrff_int	R/W	The CPU writes frame data to the FIFO after receiving this interrupt.	0x0
4	rx_frame_int	R/W	The CPU receives this interrupt and reads the frame data from the FIFO.	0x0
5	tx_frame_int	R/W	This interrupt marks the completion of a transmission	0x0
31:6	Reserved			

INT_EN

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
0	tran_done_int_en	R/W	Enable interrupt tran_done_int	0x0
1	Reserved			
2	rdff_int_en	R/W	Enable interrupt rdff_int	0x0
3	wrff_int_en	R/W	Enable interrupt wrff_int	0x0
4	rx_frame_int_en	R/W	Enable interrupt rx_frame_int	0x0
5	tx_frame_int_en	R/W	Enable interrupt tx_frame_int	0x0
31:6	Reserved			

14.3 SPI NAND Flash Controller

14.3.1 summarize

Provides off-chip SPI NAND Flash data access.

14.3.2 specificities

- Supports external 1-chip selection.
- Supports SPI NAND Flash x1/x2/x4 read and write operations.
- Supports a wide range of SPI NAND Flash devices.
  - Supports page 2KB, 4KB devices.
  - Supports 64Pages/Block, 128Pages/Block devices.
- Supports the BOOT function of SPI NAND.

14.3.3 Functional Description

14.3.3.1 interface description

SPI NAND Flash controllers can support three SPI NAND interface types, the types are Standard SPI, X2
interface mode and X4 interface mode.

Standard SPI Interface Mode.
The Standard SPI interface mode has a 1-bit data input line and a 1-bit data output line. Figure 14-8 shows Standard SPI.
The interface mode write operation timing diagram, Figure 14-9 shows the

Standard SPI interface mode read operation timing diagram.

Figure *14-8* *Standard SPI* Interface Mode Write Operation Timing

Timing

Notes:

- ✓ command/address/dummy cycles are output on the DO line in single bit serial mode.
- ✓ Data is output on the DO line in single-bit serial mode.

Figure *14-9* *Standard SPI* Interface Mode Read Operation Timing

Timing Notes:

- ✓ command/address/dummy cycles are output on the DO line in single bit serial mode.

Data is entered on the DI line in single bit serial mode.

X2 Interface Mode.

The Standard SPI interface mode shares 2bit data input and output lines. Figure 14-10 shows the X2 interface mode operation timing diagram.

Figure *14-10 SPI Nand x2* Interface Mode Operation Timing

Timing Notes:

- ✓ command/address/dummy cycles are output on the DO line in single bit serial mode.
- ✓ Data is output (write) or input (read) on the DO/DI line in 2 Bits.

X4 interface mode.

The Standard SPI interface mode shares 4bit data input and output lines. Figure 14-11 shows the timing diagram for X4 interface mode operation.

Figure *14-11 SPI Nand x4* Interface Mode Operation Timing

Timing Notes:

- ✓ command/address/dummy cycles are output on the DO line in single bit serial mode.
- ✓ Data is output (write) or input (read) on DO/DI/WPN/HOLDN lines in 4 Bits.

14.3.3.2 SPI NAND FLASH Address Description

When placing a read or write operation on the SPI NAND Flash, the operation places the row and column addresses according to the specific operation.

- Write operations: configure the column address for LOAD operations and the row address for PROGRAM operations.
- Read operation: Configure the row address during a PAGE READ TO CACHE operation and the column address during a READ operation.
- Addressing is done by the controller, and the software needs to configure reg_trx_cmd_idx, and address configuration reg_trx_cmd_cnt0, reg_trx_cmd_cnt1 according to the operation instruction.

14.3.3.3 Boot Function

Boot data cannot be directly mapped to Flash because of the discontinuous SPI NAND Flash address space and the possibility of bad blocks.
Supports adaptive boot function, which can automatically adapt the information of the adapter according to the data of Block0. The controller requires that the physical Block0 must be a good block, and other blocks can be automatically skipped if they are bad.

14.3.3.4 register operation

The software configures the operation related registers, such as operation command, address, etc. Finally, the reg_trx_start register is configured to send down the operation, and the controller sends down the operation to the device according to the software configuration value. If there is also a need to transfer data to the device, then the internal DMA operation will be used to transfer the data.

14.3.3.5 Built-in DMA operation

Supports read and write operations using the built-in system DMA mode for faster access. In this way, on-chip or off-chip memory can be accessed directly through the bus.
Step 1. configure the DMA usage channel. Step 2. configure the source and destination addresses.

Step 3. configure the transmission format and data length.

14.3.3.6 TIMEOUT Function

The software sets up a maximum 1 second TIMEOUT mechanism to protect the device when it does not respond properly.

14.3.4 workflow

14.3.4.1 initialization process

Step 1. (If Timing parameters need to be adjusted) Configure timing registers reg_trx_sck_h and reg_trx_sck_h according to the device configuration.
reg_trx_sck_l.
Step 2. configure the interrupt control register reg_trx_done_int_en.

14.3.4.2 Device Register Operation Flow

Step 1: Configure the transmission data length reg_trx_cmd_cont_size and reg_trx_data_size. Step 2: Configure the device commands and their associated contents reg_trx_cmd_id, reg_trx_cmd_cont0 and reg_trx_cmd_cont0.
reg_trx_cmd_cont1.
Step 3. configure the reg_trx_start register to send down the operation.
Step 4. reg_trx_done_int is detected, indicating that the operation is complete.

14.3.4.3 Erase Procedure

For flash operation, erase must be performed before any programming operation, and the WREN operation must be completed before the erase operation.
Step 1. configure the transmission data length reg_trx_cmd_cont_size.
Step 2 Configure the device command and its related contents reg_trx_cmd_id, reg_trx_cmd_cont0. Step 3 Configure the reg_trx_start register to send down the operation.
Step 4. reg_trx_done_int is detected, indicating that the operation is complete.

14.3.4.4 Built-in DMA read operation flow

Step 1. Configure the system DMA registers, refer to 9.5.
Step 2 Configure the transmission data length reg_trx_cmd_cont_size, reg_trx_data_size Step 3 Configure the device commands and their associated contents reg_trx_cmd_id, reg_trx_cmd_cont0 and
reg_trx_cmd_cont1.
Step 4. configure the reg_trx_start register to send down the operation.
Step 5. reg_trx_done_int is detected, indicating that the device contents have been

read and written to the buffer.

14.3.4.5 Built-in DMA Write Flow

Step 1. Configure the system DMA registers, refer to 9.5.

Step 2. Configure the transmission data length reg_trx_cmd_cont_size , reg_trx_data_size.

Step 3: Configure the device commands and their related contents reg_trx_cmd_id, reg_trx_cmd_cont0, and reg_trx_cmd_cont0.
reg_trx_cmd_cont1.

Step 4. configure the reg_trx_start register to send down the operation.

Step 5. reg_trx_done_int is detected, indicating that the buffer contents have been written to the device cache.

14.3.4.6 Other Precautions

- Some SPI NAND Flash devices require a RESET operation before use or after an abnormal reset; therefore, for device compatibility, the first transfer command after start of use or abnormal reset should be RESET.
- Do not change the related register configuration before the operation of the device is completed, or the operation may not be normal.

14.3.5 Data Structure (NAND Flash/SPI NAND Flash)

14.3.5.1 2KB page_size

The common size of the redundancy area available to the software is 64Byte for a 2KB page_size configuration. data is stored in the Buffer and the

The structure in Flash is shown below. The size of the redundancy area depends on the actual device used.

user data	Data(2048)	OOB (64)

14.3.5.2 4KB page_size

For a 4KB page_size configuration, the common size of the redundancy area available to the software is 256Byte. data in the Buffer and the

The structure in Flash is shown below. The size of the redundancy area depends on the actual device used.

user data	Data(4096)	OOB(256)

14.3.6 Registers Overview

Name	Address Offset	Description

reg_ctrl	0x000	transmission control
reg_timing_ctrl	0x004	Timing control
reg_trx_size	0x008	number of content size
reg_int_en	0x010	interrupt enable
reg_int_clr	0x014	interrupt clear
reg_int_sts	0x01c	interrupt status
reg_cont0	0x030	content 0
reg_cont1	0x034	content 1
reg_cmplt_cnt	0x058	number of transferred bytes
reg_tx_data	0x060	tx data

Name	Address Offset	Description
reg_rx_data	0x064	rx data

14.3.7 register description

reg_ctrl

Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	reg_trx_start	W1T	trigger spi transmission start	
31:1	Reserved			

reg_timing_ctrl

Offset Address: 0x004

Bits	Name	Access	Description	Reset
1:0	reg_trx_time_start	R/W	time for cs assert to 1st command bit unit: sck period	0x0
3:2	Reserved			
7:4	reg_trx_time_end	R/W	time for last data bit to cs de-assert unit: sck period	0x0
15:8	Reserved			
19:16	reg_trx_sck_h	R/W	time for sck high unit: source clock period	0x0
23:20	reg_trx_sck_l	R/W	time for sck low unit: source clock period	0x1
31:24	Reserved			

reg_trx_size

Offset Address: 0x008

Bits	Name	Access	Description	Reset
2:0	reg_trx_cmd_cont_size	R/W	numbers of command content byte	0x0
3	Reserved			
5:4	reg_trx_dummy_size	R/W	numbers of dummy byte	0x0
15:6	Reserved			
28:16	reg_trx_data_size	R/W	numbers of data byte	0x0
31:29	Reserved			

reg_int_en

Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	reg_trx_done_int_en	R/W	trx_done interrupt enable	0x1
31:1	Reserved			

reg_int_clr

Offset Address: 0x014

Bits	Name	Access	Description	Reset
0	reg_trx_done_int_clr	W1T	trx_done interrupt clear	
31:1	Reserved			

reg_int_sts

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
0	reg_trx_done_int	RO	trx_done interrpt	
31:1	Reserved			

reg_cont0

Offset Address: 0x030

Bits	Name	Access	Description	Reset
7:0	reg_trx_cmd_idx	R/W	spi flash command value	0x0
31:8	reg_trx_cmd_cont0	R/W	spi flash address, or other contents	0x0

reg_cont1

Offset Address: 0x034

Bits	Name	Access	Description	Reset
31:0	reg_trx_cmd_cont1	R/W	spi flash address, or other contents	0x0

reg_cmplt_cnt

Offset Address: 0x058

Bits	Name	Access	Description	Reset
12:0	reg_cmplt_cnt	RO	number of transferred bytes	
31:13	Reserved			

reg_tx_data

Offset Address: 0x060

Bits	Name	Access	Description	Reset
31:0	reg_tx_data	RO	spi tx data	

reg_rx_data

Offset Address: 0x064

Bits	Name	Access	Description	Reset
31:0	reg_rx_data	RO	spi rx data	

15 network interface

15.1 Ethernet MAC

15.1.1 summarize

The chip supports 1 Ethernet MAC, realizing the reception and transmission of network data.
This Ethernet MAC with a built-in 10/100Mbps Fast Ethernet Transceiver operates in the 10/100Mbps full duplex or half duplex dual mode.

15.1.2 Functional Description

以太网模块有如下功能特点：

- Ethernet MAC0 with built-in 10/100Mbps Fast Ethernet Transceiver with built-in Ethernet PHY supports 10/100Mbit/s rate.
- Supports full-duplex or half-duplex operating modes.
- Supports CRC checksums on input frames.
- Supports adding CRC checksums to output frames.
- Supports short frame fill function.
- Supports inner loopback in port full-duplex mode.
- Supports statistical counting of received and transmitted frames.
- Supports sending and receiving packet caching.
- Supports COE (Checksum Offload Engine) checksum offload engine function.

15.1.3 Overall data flow

The overall data flow of the Ethernet switching interface is shown in Figure 15-1.

Figure *15-1 eMAC* Overall Data Flow

15.1.4 Description of Single Port Function Configuration

15.1.4.1 Ethernet send/receive frame management function

The CPU first configures the Ethernet MAC receive and transmit Descriptor List buffers and the Descriptor List contents, for example, the send/receive frame address and packet type and size parameters.

When receiving, Ethernet MAC receives various packets and stores the received packets in DDR according to the CPU's configuration in receiving Descriptor List messages, such as packet caching information, including packet caching start address, packet caching depth, etc. The CPU then notifies the CPU to do the subsequent processing. Then it notifies the CPU to do the subsequent processing.

When sending, the Ethernet MAC carries the packets stored in the DDR, assembles them into packets, and sends them to the network interface based on the packet cache information, such as the packet cache start address, packet length, and other packet information, that the CPU has configured in the Transmit Descriptor List. The CPU is then notified that the packet has been transmitted.

15.1.5 Ethernet Packetized Interrupt Management Function

15.1.5.1 disruption generation

Set receive direction interrupt, configure Re_Int_Enable bit[6] = 1, CPU query receive interrupt status Reg_Int_Status bit[6].

15.1.5.2 interrupt clearing

CPU query receive interrupt status Reg_Int_Status bit[6], write 1 to clear interrupt status.

15.1.6 Configure the PHY chip operating status

The Ethernet MAC provides the MDIO interface to interface with the PHY chip. the MDIO interface is divided into read and write operations, and the main

registers that control the MDIO interface are Reg_MdioAddr and Reg_MdioData.

- The configuration steps for the read operation are as follows:
  Configure the following settings for the MDIO control registers.
  Reg_MdioAddr bit[15:11] Set PHY chip address. Please refer to the PHY chip or board plan.
  Reg_MdioAddr bit[10:6] Sets the PHY internal register address to be read or written.
  Reg_MdioAddr bit[1] Write 0 (action command for read).
  Finally, set Reg_MdioAddr bit[0] = 1, to start the read action.

algorithmic technology

The MDIO interface receives the read back data into the Reg_MdioData bit[15:0], and changes the Reg_Mdi0Addr bit[0] to 0.

- The configuration steps for the write operation are as follows:
  Configure the following settings for the MDIO control registers.
  Reg_MdioAddr bit[15:11] Set PHY chip address. Please refer to the PHY chip or board plan.
  Reg_MdioAddr bit[10:6] Sets the PHY internal register address to be read or written.
  Reg_MdioAddr bit[1] Write 1 (write action command).
  Finally, set Reg_MdioAddr bit[0] = 1, to start the write action.
  The MDIO interface changes the Reg_Mdi0Addr bit[0] to 0 when the write action is completed.

15.1.7 Working Mode Switching

Working Mode of Ethernet MAC: Ethernet MAC0 supports the built-in EPHY function. The operating mode is RMII (10/100M).

The speed and mode switching registers are set as follows.

Configuration ETH0 Reg_MacConfig bit[14] = (100M:1, 10M:0);

Note: This configuration is not available during normal operation of the chip and is recommended to be done during initialization.

15.1.8 typical application

15.1.9 Register Offset Address Description

Ethernet MAC 0 Register Offset Address Space.

- ETH0_MAC : 0x0451_000~0x0451_FFFF

15.1.10 GMAC Register Overview

Table *15-1 GMAC* Register Overview

Name	Address Offset	Description
Reg_MacConfig	0x000	Local MAC operating status register
Reg_MdioAddr	0x010	MDIO operation register.

Reg_MdioData	0x014	MDIO Data Read/Write Register
Reg_MacAddr0_High	0x040	Local MAC address register #0 High 16bit

Name	Address Offset	Description
Reg_MacAddr0_Low	0x044	Local MAC Address Register #0 Low 32bit
Reg_MacAddr1_High	0x048	Local MAC address register #0 High 16bit
Reg_MacAddr1_Low	0x04c	Local MAC Address Register #0 Low 32bit
Reg_Tx_Packet_Num_Good_Bad	0x118	Register for counting the number of good and wrong frames sent successfully.
Reg_Tx_Bcast_Packets_Good	0x11c	Broadcast frame count register for successfully sent good packets
Reg_Tx_Mcast_Packets_Good	0x120	Multicast frame count register for successfully sent good packets
Reg_Tx_Ucast_Packets_Good_Bad	0x13c	Unicast frame count register for good and wrong packets sent successfully
Reg_Tx_Mcast_Packets_Good_Bad	0x140	Multicast frame count register for good and bad packets sent successfully
Reg_Tx_Bcast_Packets_Good_Bad	0x144	Statistics register for the number of broadcast frames for successfully sent good and wrong packets
Reg_Rx_Packets_Num_Good_Bad	0x180	Statistics register for the number of successfully received good and wrong packet frames
Reg_Rx_Bcast_Packets_Good	0x18c	Broadcast frame count register for successfully received good packets
Reg_Rx_Mcast_Packets_Good	0x190	Multicast frame count register for successfully received good packets
Reg_Rx_CRC_Error_Packets	0x194	Receive CRC Detection Error Frames Count Register
Reg_Rx_Ucast_Packets_Good	0x1c4	Unicast frame count register for successfully received good packets
Reg_Int_Enable	0x101c	Interrupt Enable Register
Reg_Int_Status	0x1014	Interrupt Status Register

15.1.11 GMAC Register Description

Reg_MacConfig

Offset Address: 0x000

Bits	Name	Access	Description	Reset
1:0	Reserved			
2	RX_EN	R/W	MAC Receive Enable Register	0x0
3	TX_EN	R/W	MAC Transmit Enable Register	0x0
6:4	Reserved			

algorithmic technology

7	APCS_EN	R/W	Auto Pad or CRC Stripping Control Enable Registers	0x0
9:8	Reserved			
10	CHKS_EN	R/W	IP checksum Auxiliary enable register	0x0
11	DUPLEX_MODE	R/W	Full/Half Duplex Mode Register, 1 (enable full duplex) Mode)	0x0
12	LPBK_MODE	R/W	Loopback Mode Control Register	0x0
13	Reserved			
14	SPEED_MODE	R/W	Speed Mode Register 1'b1:100M, 1'b0:10M	0x0
16:15	Reserved			
19:17	IPG_VAL	R/W	Send Packet Pitch Control Register.	0x0

Bits	Name	Access	Description	Reset
22:20	Reserved			
23	WD_DISABLE	R/W	Watchdog disable register	0x0
24	Reserved			
25	CRC_STRIP_EN	R/W	CRC Stripping Type State Packet Control Enable Hosting tool	0x0
31:26	Reserved			

Reg_MdioAddr

Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	GO	R/W	MDIO operation completion indication. 1:Operation started 0:Operation completed.	0x0
1	CMD	R/W	MDIO Operation Command Type Write(1'b1), Read(1'b0)	0x0
5:2	Reserved			
10:6	RegAddr	R/W	External PHY Address Configuration Register	0x0
15:11	PhyAddr	R/W	PHY Device Internal Register Address Registers	0x0
31:16	Reserved			

Reg_MdioData

Offset Address: 0x014

Bits	Name	Access	Description	Reset
15:0	MdioData	R/W	MDIO Write or Read Back Data Register from PHY	0x0
31:16	Reserved			

Reg_MacAddr0_High

Offset Address: 0x040

Bits	Name	Access	Description	Reset
15:0	Addr0_High	R/W	MAC address register #0 bit[47:32]	0x0
30:16	Reserved			
31	Addr0_EN	R/W	Addr0 Enable	0x0

Reg_MacAddr0_Low

Offset Address: 0x044

Bits	Name	Access	Description	Reset
31:0	Addr0_Low	R/W	MAC address register #0 bit[31:0]	0x0

Reg_MacAddr1_High

Offset Address: 0x048

Bits	Name	Access	Description	Reset
15:0	Addr1_High	R/W	MAC address register #1 bit[47:32]	0x0
23:16	Reserved			
29:24	Addr1_MASK	R/W	Addr1 Mask Byte	0x0
30	Addr1_TYPE	R/W	1'b1: compare SA 1'b0: compare DA	0x0
31	Addr1_EN	R/W	Addr1 Enable	0x0

Reg_MacAddr1_Low

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
31:0	Addr1_Low	R/W	MAC address register #1 bit[31:0]	0x0

Reg_Tx_Packet_Num_Good_Bad

Offset Address: 0x118

Bits	Name	Access	Description	Reset
31:0	TxPktNumGB	RO	Hosting of statistics on the number of good and bad packets sent successfully tool	

Reg_Tx_Bcast_Packets_Good

Offset Address: 0x11c

Bits	Name	Access	Description	Reset
31:0	TxBcG	RO	Broadcast frame counts for successfully sent good packets are hosted. tool	

Reg_Tx_Mcast_Packets_Good

Offset Address: 0x120

Bits	Name	Access	Description	Reset
31:0	TxMcG	RO	Multicast frame count statistics for successfully sent good packets are hosted tool	

Reg_Tx_Ucast_Packets_Good_Bad

Offset Address: 0x13c

Bits	Name	Access	Description	Reset
31:0	TxUcGB	RO	Unicast frame count of successful good and wrong packets sent. registers	

Reg_Tx_Mcast_Packets_Good_Bad

Offset Address: 0x140

Bits	Name	Access	Description	Reset
31:0	TxMcGB	RO	Multicast frame count of successful good and wrong packets sent. registers	

Reg_Tx_Bcast_Packets_Good_Bad

Offset Address: 0x144

Bits	Name	Access	Description	Reset
31:0	TxBcGB	RO	The number of broadcast frames in which good and bad packets were successfully sent. registers	

Reg_Rx_Packets_Num_Good_Bad

Offset Address: 0x180

Bits	Name	Access	Description	Reset
31:0	RxPktGB	RO	Hosting of statistics on the number of successful good and bad packets received tool	

Reg_Rx_Bcast_Packets_Good

Offset Address: 0x18c

Bits	Name	Access	Description	Reset
31:0	RxBcG	RO	Statistics of broadcast frames of successfully received good packets are hosted tool	

Reg_Rx_Mcast_Packets_Good

Offset Address: 0x190

Bits	Name	Access	Description	Reset
31:0	RxMcG	RO	Multicast frame count statistics for successfully received good packets are hosted tool	

Reg_Rx_CRC_Error_Packets

Offset Address: 0x194

Bits	Name	Access	Description	Reset
31:0	RxCrcERR	RO	Receive CRC Detection Error Frame Count Register	

Reg_Rx_Ucast_Packets_Good

Offset Address: 0x1c4

Bits	Name	Access	Description	Reset
31:0	RxUcG	RO	Unicast frame count statistics for successfully received good packets are hosted tool	

Reg_Int_Enable

Offset Address: 0x101c

Bits	Name	Access	Description	Reset
0	TxInt_EN0	R/W	Transmit Interrupt Enable Register	0x0
5:1	Reserved			
6	RxInt_EN0	R/W	Receive Interrupt Enable Register	0x0
31:7	Reserved			

Reg_Int_Status

Offset Address: 0x1014

Bits	Name	Access	Description	Reset
0	TxInt_ST0	RO	Transmit Interrupt Status Register	
5:1	Reserved			
6	RxInt_ST0	RO	Receive Interrupt Status Register	
31:7	Reserved			

15.2 Ethernet PHY

15.2.1 summarize

The die provides a set of built-in Ethernet 10/100 Base-TX compliant PHY interfaces.

15.2.2 Functional Description

- Supports IEEE 802.3 10/100 Base-TX compliant.
- Supports auto-negotiation between transmission and reception speeds and full and half-duplex operation modes.
- Supports automatic movement flip function for cross or straight Ethernet lines.

- Supports Ethernet WOL (Wake on Lan) function.

15.2.3 Functional Configuration Diagram Description

Figure *15-2* Internal *10/100 Ethernet PHY* Block Diagram

10/100Mbps transmit and receive functions over standard category5 (CAT5) twisted pair cable, transmitting and receiving signals via the

The transformer is connected to a standard RJ45 connector.

16 video interface

16.1 VI

16.2 summarize

Video Input Unit VI (Video Input) is the module for receiving camera video data on the chip, which can support receiving video data through MIPI Rx (including MIPI, Sub-LVDS, HiSPi) interface or BT.656, BT.601, BT.1120 interface and DC (Digital Camera), and then send video data to the next level of video processing module (ISP) after processing. The functional block diagram of VI is shown in Figure 16-1.

VI is divided into two physical sub-modules, consisting of the Mobile Industry Processor Interface Receiver (MIPI Rx) module, which receives and processes different video data, and the Video Input Processing (VI Proc) module, which consolidates video signals of different formats into a single video signal for the ISP module.

Figure *16-1 VI* Function Block Diagram

16.2.1 specificities

- MIPI Rx supports up to two sensor data inputs simultaneously.
  - Single sensor supports up to 5M (2688x1944, 2880x1620) @60fps HDR input or

@30fps linear input.

- Dual sensor supports up to FHD(1920x1080) @60fps HDR or linear inputs.

- The MIPI Rx input supports a maximum data bit width of 12 bits.
- Supports BT.656, BT.601, BT.1120 (progressive mode only)
- Supports BT.656/BT.1120 multi-channel fusion inputs (1,2,4 channels, progressive mode only)

DC Interface

- Supports MIPI CSI-2, Sub-LVDS, HiSPi interfaces.
- Supports YUV422 format for MIPI interface input.
- Supports two-frame High Dynamic Range (HDR) image input.

16.2.2 Mode Function Description

16.2.2.1 typical application

VI can support a variety of timing inputs and different interfaces, and for different encoding methods to do video input capture, the system can use the register to configure different functional modes to adapt to different video interfaces.

The VI module can support up to three inputs, typical inputs are as follows

- 1 x 5M (2688x1944, 2880x1620) @60fps HDR input or linear @30fps input
- 2 FHD(1920x1080) @60fps HDR or Linear Inputs + 1 BT Input (BT.656, BT.601 or BT.1120)
- 1 x 5M(2688x1944, 2880x1620) @60fps HDR or @30fps Linear Input + 1 x BT Input

16.2.2.2 Functional principle

16.2.2.3 BT.1120 Interface Timing

The VI supports the BT.1120 interface timing with separate Y/C inputs. Before transmitting the video signal, it transmits a sync code, which is a special byte SAV and EAV in the data stream to indicate the beginning and end of the valid line of data, respectively. The sync code is followed by a 16bit video signal, of which 8bit is used for luminance and 8bit for chrominance, as shown in Figure 16-2 and Figure 16-3.

Figure *16-2 BT.1120* Horizontal Interface Timings

Figure *16-3 BT.1120* Vertical Interface Timings

Synchronization code format is a combination of 4 bytes, in the order of {0xFF, 0x00, 0x00, EAV/SAV}, the fourth byte is explained as follows, SG2002 only supports Progressive format, so bit 6 value is 0.

	SAV/EAV bit			Protection Bit				
7 (Fixed)	**6 (F)**	**5 (V)**	**4 (H)**	**3 (P3)**	**2 (P2)**	**1 (P1)**	**0 (P0)**	annotate
1	0	0	0	0	0	0	0	SAV_VLD
1	0	0	1	1	1	0	1	EAV_VLD
1	0	1	0	1	0	1	1	SAV_BLK
1	0	1	1	0	1	1	0	EAV_BLK

SAV_VLD : Valid line area, end of line sync signal, start of valid pixel. EVA_VLD : Valid line area, start of line sync signal, end of valid pixel. SAV_BLK : Fade line area, end of line sync signal.

EAV_BLK : Fade line area, line synchronization signal start.

16.2.2.4 BT.656 Interface Timing

The VI also supports the BT.656 interface timing for combined Y/C inputs, and

uses the sync codes SAV and EAV to indicate the beginning and end of valid lines of data, but uses only 8-bit to transmit the video signals, and utilizes a time-sharing method to transmit the brightness.

and chroma, as shown in Figure 16-4.

Figure *16-4 BT.656* Horizontal Interface Timings

The difference between BT.656 and BT.1120 is only in the 16 bit (BT.1120) and 8 bit (BT.656) used for image transmission, the rest of the vertical timing and synchronization code formats are identical.

16.2.2.5 BT.601 Interface Timing

In addition to the sync codes BT.1120 and BT.656, the VI supports the BT.601 interface timing which utilizes a variety of different sync signals for transmission. The actual video data can be set to Y/C separate input 16bit mode or Y/C combined input 8bit mode, while the synchronization mode can be selected from vhs, vde, or vsde by using the registers, the detailed timings are shown below.

Figure *16-5 BT.601 vhs* Synchronization Mode

vhs mode input synchronization signals are frame synchronization signal (vs), line synchronization signal (hs), the system must set the number of lines to be blanked after the frame.
(vs_back_porch), image height (img_ht), number of pixels after line fade (hs_back_porch) and image width
(img_wd).

Figure *16-6 BT.601 vde* Synchronization Mode

In vde mode, the synchronization signals are line valid signal (vde) and line valid signal (hde). In this mode, the system does not need to set the parameters related to timing and phase sequence, and the VI module will receive the data according to the hde/vde signals and update the frame according to the vde signal.

Figure *16-7 BT.601 vsde* Synchronization Mode

In vsde mode, the synchronization signals are frame synchronization signal (vs) and valid pixel flag (de). In this mode, the system does not need to set the parameters related to timing and phase sequence, and the VI module will receive the data according to the de signal and update the frame according to the vs signal.

Figure *16-8 BT.601 Y/C* Separate *16bit* Mode

Figure *16-9 BT.601 Y/C* Combined *8bit* Mode

16.2.2.6 Digital Camera (DC) Interface Timing

VI supports Digital Camera (DC) interface timings for transmitting RAW format emulating BT transmissions, which can be supported on the DC interface. There are four different modes: 8bit, 10bit, 12bit, and 16bit, and the registers can also be set to receive video signals using the receive sync code or three different sync modes similar to BT.601.

Figure *16-10 DC* Sync Code Mode

Figure *16-11 DC* Synchronized Signal *Mode-vhs mode*

Figure *16-12DC* Synchronous Signal Mode - *VDE mode*

Figure *16-13DC* Synchronized Signal *Mode-VSDE mode*

16.2.3 Image storage mode

The images stored in DRAM are categorized into Bayer 12 bit and YCbCr 8 bit formats. Y/Cb/Cr is stored separately in three different DRAM locations. The arrangement of the two (12bit/8bit) formats in DRAM is shown in the figure below.

Figure *16-14 Bayer 12 bit* Image Storage Methods

Bayer 12bit		Pixel 0	Pixel 1	Pixel 2	Pixel 3				
		Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
DRAM Address	Addr N+0	4	5	6	7	8	9	10	11
	Addr N+1	4	5	6	7	8	9	10	11
	Addr N+2	0	1	2	3	0	1	2	3
	Addr N+3	4	5	6	7	8	9	10	11
	Addr N+4	4	5	6	7	8	9	10	11
	Addr N+5	0	1	2	3	0	1	2	3

Figure *16-15 YCbCr 8bit* Image Storage Methods

YCbCr 8bit		Pixel 0	Pixel 1	Pixel 2	Pixel 3				
		Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
DRAM Address	Addr N+0	0	1	2	3	4	5	6	7
	Addr N+1	0	1	2	3	4	5	6	7
	Addr N+2	0	1	2	3	4	5	6	7
	Addr N+3	0	1	2	3	4	5	6	7

16.2.4 VI Register Overview

There are two identical VI modules on the chip, both with the same internal register offset addresses and base addresses of 0x0A0C2000 and 0x0A0C4000. there is also a set of VI modules that support only the BT interface with base address 0x0A0C6000.

Figure *16-16 VI* Register Overview

Name	Address Offset	Description
REG_00	0x000	MODE
REG_10	0x010	TTL_MODE_0
REG_14	0x014	TTL_MODE_1

REG_18	0x018	TTL_MODE_2
REG_1C	0x01c	TTL_MODE_3

Name	Address Offset	Description
REG_20	0x020	TTL_MODE_4
REG_24	0x024	TTL_MODE_5
REG_28	0x028	TTL_MODE_6
REG_30	0x030	TTL_MODE_7
REG_40	0x040	HDR_MODE_0
REG_44	0x044	HDR_MODE_1
REG_48	0x048	HDR_MODE_2
REG_50	0x050	BLC_MODE
REG_54	0x054	BLC_MODE_0
REG_58	0x058	BLC_MODE_1
REG_60	0x060	VI_PINMUX_0
REG_64	0x064	VI_PINMUX_1
REG_68	0x068	VI_PINMUX_2
REG_6C	0x06c	VI_PINMUX_3
REG_70	0x070	VI_PINMUX_4
REG_74	0x074	VI_PINMUX_5
REG_80	0x080	BT_PATH_0
REG_88	0x088	BT_PATH_2
REG_8C	0x08c	BT_PATH_3
REG_90	0x090	BT_PATH_4
REG_94	0x094	BT_PATH_5
REG_98	0x098	BT_PATH_6
REG_9C	0x09c	BT_PATH_7
REG_A0	0x0a0	BT_PATH_8
REG_A4	0x0a4	BT_PATH_A
REG_B0	0x0b0	CROP_0
REG_B4	0x0b4	CROP_1
REG_D0	0x0d0	MODE_CTRL
REG_D4	0x0d4	SYNC_CODE_0
REG_D8	0x0d8	SYNC_CODE_1
REG_DC	0x0dc	SYNC_CODE_2
REG_E0	0x0e0	SYNC_CODE_3
REG_E4	0x0e4	SYNC_CODE_4
REG_E8	0x0e8	SYNC_CODE_5
REG_EC	0x0ec	SYNC_CODE_6
REG_F0	0x0f0	SYNC_CODE_7
REG_F4	0x0f4	SYNC_CODE_8
REG_F8	0x0f8	SYNC_CODE_9
REG_FC	0x0fc	VS_GEN
REG_100	0x100	SYNC_CODE_A
REG_104	0x104	SYNC_CODE_B
REG_108	0x108	HDR_PATTEN_2
REG_110	0x110	HISPI_MODE_CTRL_0
REG_114	0x114	HISPI_MODE_CTRL_1
REG_118	0x118	HISPI_MODE_CTRL_2
REG_11C	0x11c	HISPI_MODE_CTRL_3
REG_120	0x120	HISPI_MODE_CTRL_4
REG_124	0x124	HISPI_MODE_CTRL_5

16.2.5 VI Register Description

REG_00

Offset Address: 0x000

Bits	Name	Access	Description	Reset
2:0	reg_sensor_mac_mode	R/W	Sensor mode 3'b000: Disable 3'b001: CSI 3'b010: Sub-LVDS 3'b011: TTL	0x0
3	reg_bt_demux_enable	R/W	BT Demux enable	0x0
4	reg_csi_ctrl_enable	R/W	CSI controller enable	0x0
5	reg_csi_vs_inv	R/W	CSI VS inverse	0x1
6	reg_csi_hs_inv	R/W	CSI HS inverse	0x1
7	Reserved			
8	reg_sublvds_ctrl_enable	R/W	Sub-LVDS controller enable	0x0
9	reg_sublvds_vs_inv	R/W	Sub-LVDS VS inverse	0x1
10	reg_sublvds_hs_inv	R/W	Sub-LVDS HS inverse	0x1
11	reg_sublvds_hdr_inv	R/W	Sub-LVDS HDR inverse	0x1
31:12	Reserved			

REG_10

Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	reg_ttl_ip_en	R/W	TTL enable	0x0
2:1	reg_ttl_sensor_bit	R/W	TTL bit mode 2'b00: 8-bit 2'b01: 10-bit 2'b10: 12-bit 2'b11: 16-bit	0x0
3	Reserved			
5:4	reg_ttl_bt_fmt_out	R/W	TTL BT output format 2'b00: {Cb,Y},{Cr,Y} 2'b01: {Cr,Y},{Cb,Y} 2'b10: {Y,Cb},{Y,Cr} 2'b11: {Y,Cr},{Y,Cb}	0x2
7:6	Reserved			
11:8	reg_ttl_fmt_in	R/W	TTL input format 4'b0000: bt_2x with sync pattern, 9-bit BT656 4'b0001: bt_1x with sync pattern, 17-bit BT1120 4'b0010: bt_2x without sync pattern, 11-bit BT601 (vhs mode) 4'b0011: bt_1x without sync pattern, 19-bit BT601 (vhs mode) 4'b0100: bt_2x without sync pattern, 11-bit BT601 (vde mode) 4'b0101: bt_1x without sync pattern, 19-bit BT601 (vde mode) 4'b0110: bt_2x without sync pattern, 11-bit BT601 (vsde mode) 4'b0111: bt_1x without sync pattern.	0x0

Bits	Name	Access	Description	Reset
			19-bit BT601 (vsde mode) 4'b100x: sensor with sync pattern 4'b101x: sensor without sync pattern, use vs + hs (vhs mode) 4'b110x: sensor without sync pattern, use vde + hde (vde mode) 4'b111x: sensor without sync pattern. use vs + hde (vsde mode)	
13:12	reg_ttl_bt_data_seq	R/W	TTL bt data sequence 2'b00: Cb0-Y0-Cr0-Y1 2'b01: Cr0-Y0-Cb0-Y1 2'b10: Y0-Cb0-Y1-Cr0 2'b11: Y0-Cr0-Y1-Cb0	0x0
14	reg_ttl_vs_inv	R/W	TTL vs inverse	0x0
15	reg_ttl_hs_inv	R/W	TTL hs inverse	0x0
31:16	Reserved			

REG_14

Offset Address: 0x014

Bits	Name	Access	Description	Reset
11:0	reg_ttl_vs_bp	R/W	TTL vsync back porch setting	0x0
15:12	Reserved			
27:16	reg_ttl_hs_bp	R/W	TTL hsync back porch setting	0x0
31:28	Reserved			

REG_18

Offset Address: 0x018

Bits	Name	Access	Description	Reset
11:0	reg_ttl_img_wd	R/W	TTL image width setting	0x0
15:12	Reserved			
27:16	reg_ttl_img_ht	R/W	TTL image height setting	0x0
31:28	Reserved			

REG_1C

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
15:0	reg_ttl_sync_0	R/W	TTL sync code 0	0x0
31:16	reg_ttl_sync_1	R/W	TTL sync code 1	0x0

REG_20

Offset Address: 0x020

Bits	Name	Access	Description	Reset
15:0	reg_ttl_sync_2	R/W	TTL sync code 2	0x0
31:16	Reserved			

REG_24

Offset Address: 0x024

Bits	Name	Access	Description	Reset
15:0	reg_ttl_sav_vld	R/W	TTL valid line SAV	0x0
31:16	reg_ttl_sav_blk	R/W	TTL blanking line SAV	0x0

REG_28

Offset Address: 0x028

Bits	Name	Access	Description	Reset
15:0	reg_ttl_eav_vld	R/W	TTL valid line EAV	0x0
31:16	reg_ttl_eav_blk	R/W	TTL blanking line EAV	0x0

REG_30

Offset Address: 0x030

Bits	Name	Access	Description	Reset
2:0	reg_vi_sel	R/W	VI input mode select 3'h1: RAW 3'h2: BT601 3'h3: BT656 3'h4: BT1120 else: reserved	0x0
3	reg_vi_from	R/W	VI input from VI0 or VI1 1'b0: from VI0 1'b1: from VI1	0x0
4	reg_vi_clk_inv	R/W	VI clock inverse	0x0
5	reg_vi_v_sel_vs	R/W	1'b1: vs_in signal as vs 1'b0: vs_in signal as vde	0x1
6	reg_vi_vs_dbg	R/W	vsync source select	0x0
7	Reserved			
8	reg_pad_vi0_clk_inv	R/W	vi0 clk inverse	0x0
9	reg_pad_vi1_clk_inv	R/W	vi1 clk inverse	0x0
10	reg_pad_vi2_clk_inv	R/W	vi2 clk inverse	0x0
31:11	Reserved			

REG_40

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	reg_sensor_mac_hdr_en	R/W	Sensor mac hdr manual mode enable Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
1	reg_sensor_mac_hdr_vsinv	R/W	Sensor mac vsync output inverse shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
2	reg_sensor_mac_hdr_hsinv	R/W	Sensor mac hsync output inverse Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
3	reg_sensor_mac_hdr_deinv	R/W	Sensor mac de output inverse Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
4	reg_sensor_mac_hdr_hdr0inv	R/W	Sensor mac hdr[0] output inverse Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
5	reg_sensor_mac_hdr_hdr1inv	R/W	Sensor mac hdr[1] output inverse Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
6	reg_sensor_mac_hdr_blcinv	R/W	Sensor mac blc output inverse Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

Bits	Name	Access	Description	Reset
7	Reserved			
8	reg_sensor_mac_hdr_mode	R/W	Sensor mac hdr mode 1'b1 stands for HiSPi S-SP HDR mode, remove HDR blanking line Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:9	Reserved			

REG_44

Offset Address: 0x044

Bits	Name	Access	Description	Reset
12:0	reg_sensor_mac_hdr_shift	R/W	Sensor mac hdr long exposure shift (long exposure lines before 1st short exposure line) Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:13	Reserved			
28:16	reg_sensor_mac_hdr_vsize	R/W	Sensor mac hdr vsize Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:29	Reserved			

REG_48

Offset Address: 0x048

Bits	Name	Access	Description	Reset
12:0	reg_sensor_mac_info_line_num	R/W	Info line number Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x1
15:13	Reserved			
16	reg_sensor_mac_rm_info_line	R/W	Remove info line Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:17	Reserved			

REG_50

Offset Address: 0x050

Bits	Name	Access	Description	Reset
0	reg_sensor_mac_blc0_en	R/W	BLC0 mode enable	0x0
1	reg_sensor_mac_blc1_en	R/W	BLC1 mode enable	0x0
31:2	Reserved			

REG_54

Offset Address: 0x054

Bits	Name	Access	Description	Reset
12:0	reg_sensor_mac_blc0_start	R/W	BLC0 start line number	0x0
15:13	Reserved			
28:16	reg_sensor_mac_blc0_size	R/W	BLC0 line size	0x4
31:29	Reserved			

REG_58

Offset Address: 0x058

Bits	Name	Access	Description	Reset
12:0	reg_sensor_mac_blc1_start	R/W	BLC1 start line number	0x0
15:13	Reserved			
28:16	reg_sensor_mac_blc1_size	R/W	BLC1 line size	0x4
31:29	Reserved			

REG_60

Offset Address: 0x060

Bits	Name	Access	Description	Reset
5:0	reg_vi_vs_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
7:6	Reserved			
13:8	reg_vi_hs_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
15:14	Reserved			
21:16	reg_vi_vde_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
23:22	Reserved			
29:24	reg_vi_hde_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
31:30	Reserved			

REG_64

Offset Address: 0x064

Bits	Name	Access	Description	Reset
5:0	reg_vi_d0_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
7:6	Reserved			
13:8	reg_vi_d1_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
15:14	Reserved			
21:16	reg_vi_d2_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
23:22	Reserved			
29:24	reg_vi_d3_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
31:30	Reserved			

REG_68

Offset Address: 0x068

Bits	Name	Access	Description	Reset
5:0	reg_vi_d4_sel	R/W	vi pin select [5]: from VI1 or VI0	0x0

Bits	Name	Access	Description	Reset
			[4:0]: from which VI pad count	
7:6	Reserved			
13:8	reg_vi_d5_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
15:14	Reserved			
21:16	reg_vi_d6_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
23:22	Reserved			
29:24	reg_vi_d7_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
31:30	Reserved			

REG_6C

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
5:0	reg_vi_d8_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
7:6	Reserved			
13:8	reg_vi_d9_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
15:14	Reserved			
21:16	reg_vi_d10_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
23:22	Reserved			
29:24	reg_vi_d11_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
31:30	Reserved			

REG_70

Offset Address: 0x070

Bits	Name	Access	Description	Reset
5:0	reg_vi_d12_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
7:6	Reserved			
13:8	reg_vi_d13_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
15:14	Reserved			
21:16	reg_vi_d14_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0
23:22	Reserved			
29:24	reg_vi_d15_sel	R/W	vi pin select [5]: from VI1 or VI0 [4:0]: from which VI pad count	0x0

Bits	Name	Access	Description	Reset
31:30	Reserved			

REG_74

Offset Address: 0x074

Bits	Name	Access	Description	Reset
2:0	reg_vi_bt_d0_sel	R/W	vi bt pin select from which VI2 pad count	0x0
3	Reserved			
6:4	reg_vi_bt_d1_sel	R/W	vi bt pin select from which VI2 pad count	0x1
7	Reserved			
10:8	reg_vi_bt_d2_sel	R/W	vi bt pin select from which VI2 pad count	0x2
11	Reserved			
14:12	reg_vi_bt_d3_sel	R/W	vi bt pin select from which VI2 pad count	0x3
15	Reserved			
18:16	reg_vi_bt_d4_sel	R/W	vi bt pin select from which VI2 pad count	0x4
19	Reserved			
22:20	reg_vi_bt_d5_sel	R/W	vi bt pin select from which VI2 pad count	0x5
23	Reserved			
26:24	reg_vi_bt_d6_sel	R/W	vi bt pin select from which VI2 pad count	0x6
27	Reserved			
30:28	reg_vi_bt_d7_sel	R/W	vi bt pin select from which VI2 pad count	0x7
31	Reserved			

REG_80

Offset Address: 0x080

Bits	Name	Access	Description	Reset
0	reg_bt_clr_sync_lost_1t	R/W	Clear sync_lost signal	0x0
1	reg_bt_ip_en	R/W	BT path enable	0x0
2	reg_bt_ddr_mode	R/W	BT DDR mode	0x0
3	reg_bt_hs_gate_by_vde	R/W	HS gating by VDE	0x0
4	reg_bt_vs_inv	R/W	vsync inverse	0x0
5	reg_bt_hs_inv	R/W	hsync inverse	0x0
6	reg_bt_vs_as_vde	R/W	input vsync as vde	0x0
7	reg_bt_hs_as_hde	R/W	input hsync as hde	0x0
14:8	reg_bt_sw_en_clk	R/W	Clock gating software enable [0]: delay control clock enable [1] : timing demultiplexer clock enable [2] : timing gen clock enable [3] : rx decode 0 clock enable [4] : rx decode 1 clock enable [5] : rx decode 2 clock enable [6] : rx decode 3 clock enable	0x0
15	Reserved			
17:16	reg_bt_demux_ch	R/W	Demux setting 2'h0: No demux 2'h1: Demux 2	0x0

Bits	Name	Access	Description	Reset
			2'h2: Demux 3 2'h3: Demux 4	
19:18	Reserved			
22:20	reg_bt_fmt_sel	R/W	3'b000 : bt_2x with sync pattern, 9-bit BT656 (clock + 8-bit data) 3'b001 : bt_1x with sync pattern, 17-bit BT1120 (clock + 16-bit data) 3'b010 : bt_2x without sync pattern, 11-bit BT601 (clock + 8-bit data + vs + hs) (vhs_mode) 3'b011 : bt_1x without sync pattern, 19- bit BT601 (clock + 16-bit data + vs + hs) (vhs_mode) 3'b100 : bt_2x without sync pattern, 11-bit BT601 (clock + 8-bit data + vde + hde) (vde_mode) 3'b101 : bt_1x without sync pattern, 19-bit BT601 (clock + 16-bit data + vde + hde) (vde_mode) 3'b110 : bt_2x without sync pattern, 11-bit BT601 (clock + 8-bit data + vs + hde) (vsde_mode) 3'b111 : bt_1x without sync pattern, 19- bit BT601 (clock + 16-bit data + vs + hde) (vsde_mode)	0x0
31:23	Reserved			

REG_88

Offset Address: 0x088

Bits	Name	Access	Description	Reset
11:0	reg_bt_img_wd_m1	R/W	BT image width	0x0
15:12	Reserved			
27:16	reg_bt_img_ht_m1	R/W	BT image height	0x0
31:28	Reserved			

REG_8C

Offset Address: 0x08c

Bits	Name	Access	Description	Reset
11:0	reg_bt_vs_bp_m1	R/W	BT vsync back porch	0x0
15:12	Reserved			
27:16	reg_bt_hs_bp_m1	R/W	BT hsync back porch	0x0
31:28	Reserved			

REG_90

Offset Address: 0x090

Bits	Name	Access	Description	Reset
7:0	reg_bt_vs_fp_m1	R/W	BT vsync front porch	0x0
15:8	reg_bt_hs_fp_m1	R/W	BT hsync front porch	0x0
31:16	Reserved			

REG_94

Offset Address: 0x094

Bits	Name	Access	Description	Reset
7:0	reg_bt_sync_0	R/W	BT sync code byte 0	0x0

Bits	Name	Access	Description	Reset
15:8	reg_bt_sync_1	R/W	BT sync code byte 1	0x0
23:16	reg_bt_sync_2	R/W	BT sync code byte 2	0x0
31:24	Reserved			

REG_98

Offset Address: 0x098

Bits	Name	Access	Description	Reset
7:0	reg_bt_sav_vld_0	R/W	BT valid SAV sync code for demux 0	0x0
15:8	reg_bt_sav_blk_0	R/W	BT blank SAV sync code for demux 0	0x0
23:16	reg_bt_eav_vld_0	R/W	BT valid EAV sync code for demux 0	0x0
31:24	reg_bt_eav_blk_0	R/W	BT blank EAV sync code for demux 0	0x0

REG_9C

Offset Address: 0x09c

Bits	Name	Access	Description	Reset
7:0	reg_bt_sav_vld_1	R/W	BT valid SAV sync code for demux 1	0x0
15:8	reg_bt_sav_blk_1	R/W	BT blank SAV sync code for demux 1	0x0
23:16	reg_bt_eav_vld_1	R/W	BT valid EAV sync code for demux 1	0x0
31:24	reg_bt_eav_blk_1	R/W	BT blank EAV sync code for demux 1	0x0

REG_A0

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
7:0	reg_bt_sav_vld_2	R/W	BT valid SAV sync code for demux 2	0x0
15:8	reg_bt_sav_blk_2	R/W	BT blank SAV sync code for demux 2	0x0
23:16	reg_bt_eav_vld_2	R/W	BT valid EAV sync code for demux 2	0x0
31:24	reg_bt_eav_blk_2	R/W	BT blank EAV sync code for demux 2	0x0

REG_A4

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
7:0	reg_bt_sav_vld_3	R/W	BT valid SAV sync code for demux 3	0x0
15:8	reg_bt_sav_blk_3	R/W	BT blank SAV sync code for demux 3	0x0
23:16	reg_bt_eav_vld_3	R/W	BT valid EAV sync code for demux 3	0x0
31:24	reg_bt_eav_blk_3	R/W	BT blank EAV sync code for demux 3	0x0

REG_B0

Offset Address: 0x0b0

Bits	Name	Access	Description	Reset
12:0	reg_sensor_mac_crop_start_x	R/W	Pixels before reg_sensor_mac_crop_start_x will be cropped in each line if enable reg_sensor_mac_crop_en.	0xFFF
15:13	Reserved			
28:16	reg_sensor_mac_crop_end_x	R/W	Pixels after reg_sensor_mac_crop_end_x will be cropped in each line if enable reg_sensor_mac_crop_en.	0xFFF
30:29	Reserved			
31	reg_sensor_mac_crop_en	R/W	enable crop function	0x0

REG_B4

Offset Address: 0x0b4

Bits	Name	Access	Description	Reset
12:0	reg_sensor_mac_crop_start_y	R/W	Lines before reg_sensor_mac_crop_start_y will be cropped in each frame if enable reg_sensor_mac_crop_en.	0xFFF
15:13	Reserved			
28:16	reg_sensor_mac_crop_end_y	R/W	Lines after reg_sensor_mac_crop_end_y will be cropped in each frame if enable reg_sensor_mac_crop_en.	0xFFF
31:29	Reserved			

REG_D0

Offset Address: 0x0d0

Bits	Name	Access	Description	Reset
0	reg_ttl_as_slvds_enable	R/W	Sub-LVDS lane enable for each lane	0x0
7:1	Reserved			
9:8	reg_ttl_as_slvds_bit_mode	R/W	Sub-LVDS bit mode 2'b00: 8-bit 2'b01: 10-bit 2'b10: 12-bit	0x2
10	reg_ttl_as_slvds_data_reverse	R/W	Sub-LVDS data packet bit inverse	0x0
11	Reserved			
12	reg_ttl_as_slvds_hdr_mode	R/W	Sub-LVDS HDR mode enable	0x0
13	reg_ttl_as_slvds_hdr_pattern	R/W	Sub-LVDS HDR pattern mode 1'b0: pattern 1 1'b1: pattern 2	0x0
31:14	Reserved			

REG_D4

Offset Address: 0x0d4

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_sync_1st	R/W	Sub-LVDS SYNC code 1st word	0xFFF
15:12	Reserved			
27:16	reg_ttl_as_slvds_sync_2nd	R/W	Sub-LVDS SYNC code 2nd word	0x000
31:28	Reserved			

REG_D8

Offset Address: 0x0d8

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_sync_3rd	R/W	Sub-LVDS SYNC code 3rd word	0x000
15:12	Reserved			
27:16	reg_ttl_as_slvds_norm_bk_sav	R/W	Normal mode blanking SAV	0xAB0
31:28	Reserved			

REG_DC

Offset Address: 0x0dc

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_norm_bk_eav	R/W	Normal mode blanking EAV	0xB60
15:12	Reserved			
27:16	reg_ttl_as_slvds_norm_sav	R/W	Normal mode active SAV	0x800
31:28	Reserved			

REG_E0

Offset Address: 0x0e0

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_norm_eav	R/W	Normal mode active EAV	0x9D0
15:12	Reserved			
27:16	reg_ttl_as_slvds_n0_bk_sav	R/W	HDR mode n0 blanking SAV	0x2B0
31:28	Reserved			

REG_E4

Offset Address: 0x0e4

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_n0_bk_eav	R/W	HDR mode n0 blanking EAV	0x360
15:12	Reserved			
27:16	reg_ttl_as_slvds_n1_bk_sav	R/W	HDR mode n1 blanking SAV	0x6B0
31:28	Reserved			

REG_E8

Offset Address: 0x0e8

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_n1_bk_eav	R/W	HDR mode n1 blanking EAV	0x760
15:12	Reserved			
27:16	reg_ttl_as_slvds_n0_lef_sav	R/W	Sub-LVDS mode: n0 long exposure sav Sub-LVDS 12-bit LEF SAV n0 (801) Sub-LVDS 10-bit LEF SAV n0 (004) HiSPi P-SP mode: SOL T1 (800)	0x801
31:28	Reserved			

REG_EC

Offset Address: 0x0ec

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_n0_lef_eav	R/W	Sub-LVDS mode: n0 long exposure eav Sub-LVDS 12-bit LEF EAV n0 (9D1) Sub-LVDS 10-bit LEF EAV n0 (1D4) HiSPi P-SP mode: EOL T1 (A00)	0x9D1
15:12	Reserved			
27:16	reg_ttl_as_slvds_n0_sef_sav	R/W	Sub-LVDS mode: n0 short exposure sav Sub-LVDS 12-bit SEF SAV n0 (802) Sub-LVDS 10-bit SEF SAV n0 (008) HiSPi P-SP mode: SOL T2 (820)	0x802
31:28	Reserved			

REG_F0

Offset Address: 0x0f0

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_n0_sef_eav	R/W	Sub-LVDS mode: n0 short exposure eav Sub-LVDS 12-bit SEF EAV n0 (9D2) Sub-LVDS 10-bit SEF EAV n0 (1d8) HiSPi P-SP mode: EOL T2 (A20)	0x9D2
15:12	Reserved			
27:16	reg_ttl_as_slvds_n1_lef_sav	R/W	Sub-LVDS mode: n1 long exposure sav Sub-LVDS 12-bit LEF SAV n1 (C01) Sub-LVDS 10-bit LEF SAV n1 (404) HiSPi P-SP mode: SOF T1 (C00)	0xC01

Bits	Name	Access	Description	Reset
31:28	Reserved			

REG_F4

Offset Address: 0x0f4

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_n1_lef_eav	R/W	Sub-LVDS mode: n1 long exposure eav Sub-LVDS 12-bit LEF EAV n1 (DD1) Sub-LVDS 10-bit LEF EAV n1 (5D4) HiSPi P-SP mode: EOF T1 (E00)	0xDD1
15:12	Reserved			
27:16	reg_ttl_as_slvds_n1_sef_sav	R/W	Sub-LVDS mode: n1 short exposure sav Sub-LVDS 12-bit SEF SAV n1 (C02) Sub-LVDS 10-bit SEF SAV n1 (408) HiSPi P-SP mode: SOF T2 (C20)	0xC02
31:28	Reserved			

REG_F8

Offset Address: 0x0f8

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_n1_sef_eav	R/W	Sub-LVDS mode: n1 short exposure eav Sub-LVDS 12-bit SEF EAV n1 (DD2) Sub-LVDS 10-bit SEF EAV n1 (5D8) HiSPi P-SP mode: EOF T2 (E20)	0xDD2
31:12	Reserved			

REG_FC

Offset Address: 0x0fc

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_vs_gen_sync_code	R/W	vs generate sync code value using scenario: HiSPi P-SP HDR	0xC00
12	reg_ttl_as_slvds_vs_gen_by_sync_code	R/W	vs generate by identical sync code using scenario: HiSPi P-SP HDR	0x0
31:13	Reserved			

REG_100

Offset Address: 0x100

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_n0_lsef_sav	R/W	SAV for n0 long & short exposure both exist line only used for pattern 2	0x803
15:12	Reserved			
27:16	reg_ttl_as_slvds_n0_lsef_eav	R/W	EAV for n0 long & short exposure both exist line only used for pattern 2	0x9D3
31:28	Reserved			

REG_104

Offset Address: 0x104

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_slvds_n1_lsef_sav	R/W	SAV for n1 long & short exposure both exist line only used for pattern 2	0xC03
15:12	Reserved			

Bits	Name	Access	Description	Reset
27:16	reg_ttl_as_slvds_n1_lsef_eav	R/W	EAV for n1 long & short exposure both exist line only used for pattern 2	0xDD3
31:28	Reserved			

REG_108

Offset Address: 0x108

Bits	Name	Access	Description	Reset
13:0	reg_ttl_as_slvds_hdr_p2_hsize	R/W	Hsize for pattern 2	0xF0
15:14	Reserved			
29:16	reg_ttl_as_slvds_hdr_p2_hblank	R/W	Hblank size for pattern 2	0x14
31:30	Reserved			

REG_110

Offset Address: 0x110

Bits	Name	Access	Description	Reset
0	reg_ttl_as_hispi_mode	R/W	HiSPi mode enable 1'b0: Sub-LVDS 1'b1: HiSPi	0x0
1	reg_ttl_as_hispi_use_hsize	R/W	HiSPi DE de-assert by register count	0x0
3:2	Reserved			
4	reg_ttl_as_hispi_hdr_psp_mode	R/W	HiSPi P-SP HDR mode enable	0x0
31:5	Reserved			

REG_114

Offset Address: 0x114

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_hispi_norm_sof	R/W	HiSPi SOF sync code	0xC00
15:12	Reserved			
27:16	reg_ttl_as_hispi_norm_eof	R/W	HiSPi EOF sync code	0xE00
31:28	Reserved			

REG_118

Offset Address: 0x118

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_hispi_hdr_t1_sof	R/W	HiSPi HDR T1 SOF	0xC00
15:12	Reserved			
27:16	reg_ttl_as_hispi_hdr_t1_eof	R/W	HiSPi HDR T1 EOF	0xE00
31:28	Reserved			

REG_11C

Offset Address: 0x11c

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_hispi_hdr_t1_sol	R/W	HiSPi HDR T1 SOL	0x800
15:12	Reserved			
27:16	reg_ttl_as_hispi_hdr_t1_eol	R/W	HiSPi HDR T1 EOL	0xA00
31:28	Reserved			

REG_120

Offset Address: 0x120

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_hispi_hdr_t2_sof	R/W	HiSPi HDR T2 SOF	0xC20
15:12	Reserved			
27:16	reg_ttl_as_hispi_hdr_t2_eof	R/W	HiSPi HDR T2 EOF	0xE20
31:28	Reserved			

REG_124

Offset Address: 0x124

Bits	Name	Access	Description	Reset
11:0	reg_ttl_as_hispi_hdr_t2_sol	R/W	HiSPi HDR T2 SOL	0x820
15:12	Reserved			
27:16	reg_ttl_as_hispi_hdr_t2_eol	R/W	HiSPi HDR T2 EOL	0xA20
31:28	Reserved			

16.3 VDP (Video Display Processor)

16.3.1 summarize

The VDP module can iterate graphic data over video data and output it through different display channels. The video data can be read from memory or received from the VPSS module. Graphics data must be read from memory.

16.3.2 Architecture Description

The overall architecture of VDP is shown in Figure

Figure *16-17 VDP* Function Block Diagram

algorithmic technology

SG2002 supports only one way BT.656/BT.601/BT.1120/MCU-I80/MCU-M68/LVDS/MIPI DSI/serial TTL.

exports

- Bus data reading, data processing, including video layer V, graphics layer G.
- The video layer can receive different image formats (YUV422,YUV420,YUV444,RBG). packet,NV12/NV21,YUYV-packet), and converts them to the timing and format required for the display channel.
- The graphics layer supports ARGB8888,ARGB4444, ARGB1555, 8-bit/4-bit LUT, font base and other formats. OSD.
- Support for OSD anaerobic format
- Blend : Single Graphics Layer G Video Graphics Iterator.

VDP is characterized by the following.

- MIPI Tx Output Interface : Maximum support 1080P@60fps RGB24-bit output
- LVDS Output : Supports up to 720P@60fps RGB24-bit Output
- Digital Output Connector : (BT format only support progressive mode)
  - Supports ITU-R BT.1120 outputs
  - Supports ITU-R BT.656 outputs
  - Supports ITU-R BT.601 outputs
  - MCU I80/M68 8-bit outputs
  - Supports serial TTL parallel output
- Video Layer : V-Layer
- Graphics Layer : G Layer
- Iteration characteristics: V/G 256th order linear iteration with Gamma correction.
- The VDP contains a display channel, a timing interrupt, and a low-bandwidth interrupt.

16.3.3 working method

16.3.3.1 Clock Configuration

VDP is paired with a proprietary clock generator with the following architecture

Figure *16-18 VDP* Clock Generator

algorithmic technology

- clk_syn : 1GHz or 1.2Ghz

- FREQ_SYN : 6-bit integer, 26-bit fractional frequency generator
- PLL: Generates the VDP clock and the clock required for MIPI Tx/ LVDS serial.

16.3.3.2 reset (a dislocated joint, an electronic device etc)

The VDP reset consists of a hardware reset and a software reset. Before performing an AXI bus reset.

- Turns all AXI access off.
- Confirm that the AXI access action has ended, and then configure the bus reset.

16.3.3.3 output interface

VDP supports the following three interface outputs

- MIPI DSI
- LVDS
- ITU-R BT.1120/ITU-R BT.656/ITU-R BT.601/MCU I80/MCU M68/Serial TTL outputs

16.3.3.4 disruptions

VDP supports two types of interrupts

- Vertical Timing Interrupt
- low-bandwidth disruption

16.3.3.5 Vertical Timing Interrupt

The VDP contains one display channel, corresponding to a vertical timing interrupt.

- Supports frame start/end interrupt.
- Support for interrupt shielding can be matched.
- Each interrupt can be written 1 clear.

16.3.3.6 low-bandwidth disruption

VDP supports polling to provide low-bandwidth state.

- Interrupt shielding can be matched.
- Interrupt Write 1 Clear.

16.3.4 Functional Description

16.3.4.1 Video Layer Functions

16.3.4.2 Video Layer V Characteristics

- Support input image format : 400, planar-420,planar-422,planar-444, RGB packet.
- Minimum input resolution is 64x64, maximum input resolution is 1920x1080.
- The minimum output resolution is 64x64 and the maximum output resolution is 1920x1080.
- Supported input data bit width: 8-bit.
- YUV420 Horizontal/vertical resolution is a multiple of 2.
- YUV422 Horizontal resolution is a multiple of 2.
- YUV400/Planar-444(RGB or YUV)/RGB packet No resolution limitation.
- The source start address is assignable and the addresses are 32-byte aligned.
  - Source stride assignable, 32-byte aligned
- Supports color space conversion and contrast/brightness adjustment.
- Support video layer BT.601, BT.709 color gamut conversion.
- Support the display position/size can be matched, in any position of the screen display.

16.3.4.3 Graphics Layer G Characteristics

- Support input pixel format: ARGB8888,ARGB4444,ARGB1555,LUT8,LUT4,LUT1.
- Minimum input resolution is 64x64, maximum input resolution is 1920x1080.
  - The minimum output resolution is 64x64 and the maximum output resolution is 1920x1080.
- The source start address is assignable and the addresses are 32-byte aligned.
  - Source stride mappable, 32-byte aligned.
  - Supports color space conversion.
  - Support the display position can be assigned, in any position of the screen display.
  - Support 0~255 alpha.
  - Support for colorkey processing.

16.3.4.4 overlay processing

VDP supports only one video layer and one graphics layer iteratively.

16.3.4.5 superimposed property

Supports 8 window iterations.

16.3.4.6 Display Channel Characteristics

- It can be used as an output channel for HD and SD.
  - Only one output connector can be selected for output.
  - Supports typical 1080P@60fps, 720P@60fps output timing.

16.3.4.7 timing configuration

The VDP output interface can be configured with various typical and atypical timings depending on the interfaces of the interfacing chips. All timing parameters must be configured before the interface is turned on.

16.3.4.8 HD Output Interface MIPI Tx

- Support RGB666,RGB565,RGB10-10-10 output.
- Supports 1080P@60fps 4-channel display.
- Support 720P@60fps 2/4-channel display

16.3.4.9 HD Output Interface BT.1120 Features

- Supports YUV422 8bit output.
- The clip that supports data clamping, according to the interface protocol, the Y clamp range is 16~235 and the C clamp range is 16~240.
- Supports the following typical output timings: 720P, 1080P.
- 16bit data, default Y is in high position, C is in low position, YC position is interchangeable.
- Supports color signal Cb/Cr timing interchange.
- Supports inversion of the output follow-on clock
- Only Progressive timings are supported.

16.3.4.10 SD Output Interface BT.656 Features

- Supports YUV422 8bit output.
- The clip that supports data clamping, according to the interface protocol, the Y clamp range is 16~235 and the C clamp range is 16~240.
- Supports Y/C timing interchange and Cb/Cr timing interchange for color signals.
- Progressive timing only

16.3.4.11 BT.601 Characteristics

16.3.4.12

- Supports YUV422 8-bit output.
- YC Data range 0 ~ 255.
- Supports VS/HS synchronized signal output
- Supports Y/C timing interchange and Cb/Cr timing interchange for color signals.
- Only Progressive timings are supported.

16.3.4.13 MCU Characteristics

- Supports 8-bit data, 4-bit control output.
- Supports I80/M68 format output.

16.3.4.14 LCD LVDS Output Interface

Supports 1-link LVDS output. Supports 6-bit/8-bit RGB serial output. Supports VESA/JEIDA format output.

Figure ***16-19 LVDS JEIDA/VESA mode data format***

16.3.5 VDP Register Overview

VDP DISP register location 0X0A08_8000 ~0x0A08_83FF

Name	Address Offset	Description
REG_00	0x000	
REG_01	0x004	
REG_02	0x008	
REG_03	0x00c	
REG_04	0x010	
REG_05	0x014	
REG_06	0x018	
REG_07	0x01c	
REG_13	0x034	
REG_14	0x038	
REG_15	0x03c	
REG_16	0x040	
REG_17	0x044	

Name	Address Offset	Description
REG_18	0x048	
REG_19	0x04c	
REG_20	0x050	
REG_21	0x054	
REG_22	0x058	
REG_23	0x05c	
REG_24	0x060	
REG_25	0x064	
REG_26	0x068	
REG_27	0x06c	
REG_28	0x070	
REG_29	0x074	
REG_30	0x078	
REG_31	0x07c	
REG_32	0x080	
REG_33	0x084	
REG_34	0x088	
REG_35	0x08c	
REG_36	0x090	
REG_37	0x094	
REG_39	0x09c	
REG_40	0x0a0	
REG_41	0x0a4	
REG_42	0x0a8	
REG_43	0x0ac	
REG_CATCH	0x0c0	
REG_OSD_FIFO_M0	0x130	
REG_OSD_FIFO_M1	0x134	
REG_OSD_FIFO_CNT0	0x138	
REG_OSD_FIFO_CNT1	0x13c	
REG_OSD_FIFO_CNT2	0x140	
REG_OSD_FIFO_CNT3	0x144	
REG_GAMMA_CTRL	0x180	
REG_GAMMA_WR_LUT	0x184	
REG_MCU_IF_CTRL	0x200	
REG_HW_MCU_AUTO	0x210	
REG_HW_MCU_CMD	0x214	
REG_HW_MCU_CMD_0	0x218	
REG_HW_MCU_CMD_1	0x21c	
REG_HW_MCU_CMD_2	0x220	
REG_HW_MCU_CMD_3	0x224	

Name	Address Offset	Description
REG_HW_MCU_CMD_4	0x228	
REG_HW_MCU_CMD_5	0x22c	
REG_HW_MCU_CMD_6	0x230	
REG_HW_MCU_CMD_7	0x234	
REG_HW_MCU_OV	0x238	
REG_SRGB_CTRL	0x240	
COV_W0_CFG	0x280	
COV_W0_SIZE	0x284	
COV_W0_COLOR	0x288	
COV_W1_CFG	0x28c	
COV_W1_SIZE	0x290	
COV_W1_COLOR	0x294	
COV_W2_CFG	0x298	
COV_W2_SIZE	0x29c	
COV_W2_COLOR	0x2a0	
COV_W3_CFG	0x2a4	
COV_W3_SIZE	0x2a8	
COV_W3_COLOR	0x2ac	
REG_TGEN_LITE_SIZE	0x304	
REG_TGEN_LITE_VS	0x308	
REG_TGEN_LITE_HS	0x314	

VDP OSD register locations 0X0A08_8800 ~0x0A08_89FF

Name	Address Offset	Description
VGOP_REG_0	0x000	OW0_Setting_0
VGOP_REG_1	0x004	OW0_Setting_1
VGOP_REG_2	0x008	OW0_Setting_2
VGOP_REG_3	0x00c	OW0_Setting_3
VGOP_REG_4	0x010	OW0_Setting_3_1
VGOP_REG_5	0x014	OW0_Setting_4
VGOP_REG_6	0x018	OW0_Setting_5
VGOP_REG_10	0x020	OW1_Setting_0
VGOP_REG_11	0x024	OW1_Setting_1
VGOP_REG_12	0x028	OW1_Setting_2
VGOP_REG_13	0x02c	OW1_Setting_3
VGOP_REG_14	0x030	OW1_Setting_3_1
VGOP_REG_15	0x034	OW1_Setting_4
VGOP_REG_16	0x038	OW1_Setting_5
VGOP_REG_20	0x040	OW2_Setting_0
VGOP_REG_21	0x044	OW2_Setting_1
VGOP_REG_22	0x048	OW2_Setting_2
VGOP_REG_23	0x04c	OW2_Setting_3
VGOP_REG_24	0x050	OW2_Setting_3_1
VGOP_REG_25	0x054	OW2_Setting_4
VGOP_REG_26	0x058	OW2_Setting_5
VGOP_REG_30	0x060	OW3_Setting_0

Name	Address Offset	Description
VGOP_REG_31	0x064	OW3_Setting_1
VGOP_REG_32	0x068	OW3_Setting_2
VGOP_REG_33	0x06c	OW3_Setting_3
VGOP_REG_34	0x070	OW3_Setting_3_1
VGOP_REG_35	0x074	OW3_Setting_4
VGOP_REG_36	0x078	OW3_Setting_5
VGOP_REG_40	0x080	OW4_Setting_0
VGOP_REG_41	0x084	OW4_Setting_1
VGOP_REG_42	0x088	OW4_Setting_2
VGOP_REG_43	0x08c	OW4_Setting_3
VGOP_REG_44	0x090	OW4_Setting_3_1
VGOP_REG_45	0x094	OW4_Setting_4
VGOP_REG_46	0x098	OW4_Setting_5
VGOP_REG_50	0x0a0	OW5_Setting_0
VGOP_REG_51	0x0a4	OW5_Setting_1
VGOP_REG_52	0x0a8	OW5_Setting_2
VGOP_REG_53	0x0ac	OW5_Setting_3
VGOP_REG_54	0x0b0	OW5_Setting_3_1
VGOP_REG_55	0x0b4	OW5_Setting_4
VGOP_REG_56	0x0b8	OW5_Setting_5
VGOP_REG_60	0x0c0	OW6_Setting_0
VGOP_REG_61	0x0c4	OW6_Setting_1
VGOP_REG_62	0x0c8	OW6_Setting_2
VGOP_REG_63	0x0cc	OW6_Setting_3
VGOP_REG_64	0x0d0	OW6_Setting_3_1
VGOP_REG_65	0x0d4	OW6_Setting_4
VGOP_REG_66	0x0d8	OW6_Setting_5
VGOP_REG_70	0x0e0	OW7_Setting_0
VGOP_REG_71	0x0e4	OW7_Setting_1
VGOP_REG_72	0x0e8	OW7_Setting_2
VGOP_REG_73	0x0ec	OW7_Setting_3
VGOP_REG_74	0x0f0	OW7_Setting_3_1
VGOP_REG_75	0x0f4	OW7_Setting_4
VGOP_REG_76	0x0f8	OW7_Setting_5
VGOP_REG_80	0x100	Common
VGOP_REG_81	0x104	idx_sram_ctl
VGOP_REG_82	0x108	idx_sram_ctl
VGOP_REG_83	0x10c	clr_key_content
VGOP_REG_84	0x110	const_clr
VGOP_REG_85	0x114	debug
VGOP_REG_86	0x120	fb_thr
VGOP_REG_87	0x124	fb0_setting
VGOP_REG_88	0x128	fb0_init_st
VGOP_REG_89	0x12c	fb0_st_ro
VGOP_REG_90	0x134	fb1_setting
VGOP_REG_91	0x138	fb1_init_st
VGOP_REG_92	0x13c	fb1_st_ro
BW_LIMIT	0x140	
VGOP_DEC_00	0x150	vgop_dec_ctrl
VGOP_DEC_01	0x154	vgop_dec_debug
VGOP_LUT16_0	0x160	vgop_lut16_0_1
VGOP_LUT16_1	0x164	vgop_lut16_2_3
VGOP_LUT16_2	0x168	vgop_lut16_4_5
VGOP_LUT16_3	0x16c	vgop_lut16_6_7

Name	Address Offset	Description
VGOP_LUT16_4	0x170	vgop_lut16_8_9
VGOP_LUT16_5	0x174	vgop_lut16_10_11
VGOP_LUT16_6	0x178	vgop_lut16_12_13
VGOP_LUT16_7	0x17c	vgop_lut16_14_15

16.3.6 VDP Register Description

REG_00

Offset Address: 0x000

Bits	Name	Access	Description	Reset
2:0	reg_disp_sel	R/W	disp soruce selection 0 : from DMA 1 : from scaler_disp Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
8:3	Reserved			
9	reg_vs_pol_lite	R/W	vsync polarity for tgen_lite 0 : high active 1 : low active	0x0
10	reg_hs_pol_lite	R/W	hsync polarity for tgen lite 0 : high active 1 : low active	0x0
11	reg_tgen_en_lite	R/W	tgen lite enable signal 0 : disable 1 : enable	0x0
15:12	reg_fmt_sel	R/W	disp source from DMA 4'h0 : YUV420 planar 4'h1 : YUV422 planar 4'h2 : RGB888 planar 4'h3 : RGB packed {R,G,B} 4'h4 : RGB packed {B,G,R} 4'h5 : Y only 4'h6 : reserved 4'h7 : reserved 4'h8 : NV21 {V,U} 4'h9 : NV12 {U,V} 4'ha : YUV422-SP1{V,U} 4'hb : YUV422-SP2{U,V} 4'hc : YUV2-1 {U,Y,V,Y} 4'hd : YUV2-2 {V,Y,U,Y} 4'he : YUV2-3 {Y,U,Y,V} 4'hf : YUV2-4 {Y,V,Y,U} Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:16	Reserved			

REG_01

Offset Address: 0x004

Bits	Name	Access	Description	Reset
13:0	reg_vtt	R/W	vtotal Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x167
15:14	Reserved			
29:16	reg_htt	R/W	htotal Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x1df
31:30	Reserved			

REG_02

Offset Address: 0x008

Bits	Name	Access	Description	Reset
13:0	reg_vs_str	R/W	vsync start Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x1
15:14	Reserved			
29:16	reg_vs_stp	R/W	vsync end Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x3
31:30	Reserved			

REG_03

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
13:0	reg_vfde_str	R/W	vfde start Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x6e
15:14	Reserved			
29:16	reg_vfde_stp	R/W	vfde end Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x15d
31:30	Reserved			

REG_04

Offset Address: 0x010

Bits	Name	Access	Description	Reset
13:0	reg_vmde_str	R/W	vmde start Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x6e
15:14	Reserved			
29:16	reg_vmde_stp	R/W	vmde end Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x15d
31:30	Reserved			

REG_05

Offset Address: 0x014

Bits	Name	Access	Description	Reset
13:0	reg_hs_str	R/W	hsync start Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x1
15:14	Reserved			
29:16	reg_hs_stp	R/W	hsync end Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0xa
31:30	Reserved			

REG_06

Offset Address: 0x018

Bits	Name	Access	Description	Reset
13:0	reg_hfde_str	R/W	hfde start Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x14
15:14	Reserved			
29:16	reg_hfde_stp	R/W	hfde end Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x153
31:30	Reserved			

REG_07

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
13:0	reg_hmde_str	R/W	hmde start Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x14
15:14	Reserved			
29:16	reg_hmde_stp	R/W	hmde end Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x153
31:30	Reserved			

REG_13

Offset Address: 0x034

Bits	Name	Access	Description	Reset
31:0	reg_src_y_base_0	R/W	source Y/R/packet DRAM address Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

REG_14

Offset Address: 0x038

Bits	Name	Access	Description	Reset
7:0	reg_src_y_base_1	R/W	Shadow: Yes	0x0

Bits	Name	Access	Description	Reset
			Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	
31:8	Reserved			

REG_15

Offset Address: 0x03c

Bits	Name	Access	Description	Reset
31:0	reg_src_u_base_0	R/W	source U/G DRAM address Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

REG_16

Offset Address: 0x040

Bits	Name	Access	Description	Reset
7:0	reg_src_u_base_1	R/W	Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:8	Reserved			

REG_17

Offset Address: 0x044

Bits	Name	Access	Description	Reset
31:0	reg_src_v_base_0	R/W	source V/B DRAM address Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

REG_18

Offset Address: 0x048

Bits	Name	Access	Description	Reset
7:0	reg_src_v_base_1	R/W	Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:8	Reserved			

REG_19

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
23:0	reg_src_y_pitch	R/W	source Y/R/packet line pitch, must be 32-byte aligned Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:24	Reserved			

REG_20

Offset Address: 0x050

Bits	Name	Access	Description	Reset
23:0	reg_src_c_pitch	R/W	source UV/GB line pitch, must be 32- byte aligned Shadow: Yes	0x0

Bits	Name	Access	Description	Reset
			Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	
24	reg_64b_align	R/W	read address 64-byte align enable, only effective when reg_burst_ln se to 3,7,11 or 15	0x0
31:25	Reserved			

REG_21

Offset Address: 0x054

Bits	Name	Access	Description	Reset
15:0	reg_src_x_str	R/W	source crop x start position Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:16	reg_src_y_str	R/W	source crop y start position Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

REG_22

Offset Address: 0x058

Bits	Name	Access	Description	Reset
15:0	reg_src_wd	R/W	effect when source is from DRAM Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:16	reg_src_ht	R/W	effect when source is from DRAM Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

REG_23

Offset Address: 0x05c

Bits	Name	Access	Description	Reset
13:0	reg_out_csc_c00	R/W	display output csc coef C00, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x400
15:14	Reserved			
29:16	reg_out_csc_c01	R/W	display output csc coef C01, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
30	Reserved			
31	reg_out_csc_en	R/W	display output Color Space Convert Enable Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

REG_24

Offset Address: 0x060

Bits	Name	Access	Description	Reset
13:0	reg_out_csc_c02	R/W	display output csc coef C02, s.3.10 Shadow: Yes	0x59c

Bits	Name	Access	Description	Reset
			Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	
15:14	Reserved			
29:16	reg_out_csc_c10	R/W	display output csc coef C10, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x400
31:30	Reserved			

REG_25

Offset Address: 0x064

Bits	Name	Access	Description	Reset
13:0	reg_out_csc_c11	R/W	display output csc coef C11, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x2160
15:14	Reserved			
29:16	reg_out_csc_c12	R/W	display output csc coef C12, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x22db
31:30	Reserved			

REG_26

Offset Address: 0x068

Bits	Name	Access	Description	Reset
13:0	reg_out_csc_c20	R/W	display output csc coef C20, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x400
15:14	Reserved			
29:16	reg_out_csc_c21	R/W	display output csc coef C21, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x717
31:30	Reserved			

REG_27

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
13:0	reg_out_csc_c22	R/W	display output csc coef C22, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:14	Reserved			

REG_28

Offset Address: 0x070

Bits	Name	Access	Description	Reset
7:0	reg_out_csc_sub_0	R/W	display output csc sub value, 0~255 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:8	reg_out_csc_sub_1	R/W	display output csc sub value, 0~255	0x80

Bits	Name	Access	Description	Reset
			Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	
23:16	reg_out_csc_sub_2	R/W	display output csc sub value, 0~255 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x80
31:24	Reserved			

REG_29

Offset Address: 0x074

Bits	Name	Access	Description	Reset
7:0	reg_out_csc_add_0	R/W	display output csc add value, 0~255 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:8	reg_out_csc_add_1	R/W	display output csc add value, 0~255 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
23:16	reg_out_csc_add_2	R/W	display output csc add value, 0~255 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:24	Reserved			

REG_30

Offset Address: 0x078

Bits	Name	Access	Description	Reset
13:0	reg_in_csc_c00	R/W	display input csc coef C00, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x400
15:14	Reserved			
29:16	reg_in_csc_c01	R/W	display input csc coef C01, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
30	Reserved			
31	reg_in_csc_en	R/W	display input Color Space Convert Enable Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

REG_31

Offset Address: 0x07c

Bits	Name	Access	Description	Reset
13:0	reg_in_csc_c02	R/W	display input csc coef C02, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x59c
15:14	Reserved			
29:16	reg_in_csc_c10	R/W	display input csc coef C10, s.3.10 Shadow: Yes	0x400

Bits	Name	Access	Description	Reset
			Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	
31:30	Reserved			

REG_32

Offset Address: 0x080

Bits	Name	Access	Description	Reset
13:0	reg_in_csc_c11	R/W	display input csc coef C11, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x2160
15:14	Reserved			
29:16	reg_in_csc_c12	R/W	display input csc coef C12, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x22db
31:30	Reserved			

REG_33

Offset Address: 0x084

Bits	Name	Access	Description	Reset
13:0	reg_in_csc_c20	R/W	display input csc coef C20, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x400
15:14	Reserved			
29:16	reg_in_csc_c21	R/W	display input csc coef C21, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x717
31:30	Reserved			

REG_34

Offset Address: 0x088

Bits	Name	Access	Description	Reset
13:0	reg_in_csc_c22	R/W	display input csc coef C22, s.3.10 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:14	Reserved			

REG_35

Offset Address: 0x08c

Bits	Name	Access	Description	Reset
7:0	reg_in_csc_sub_0	R/W	display input csc sub value, 0~255 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:8	reg_in_csc_sub_1	R/W	display input csc sub value, 0~255 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x80
23:16	reg_in_csc_sub_2	R/W	display input csc sub value, 0~255 Shadow: Yes	0x80

Bits	Name	Access	Description	Reset
			Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	
31:24	Reserved			

REG_36

Offset Address: 0x090

Bits	Name	Access	Description	Reset
7:0	reg_in_csc_add_0	R/W	display input csc add value, 0~255 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:8	reg_in_csc_add_1	R/W	display input csc add value, 0~255 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
23:16	reg_in_csc_add_2	R/W	display input csc add value, 0~255 Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:24	Reserved			

REG_37

Offset Address: 0x094

Bits	Name	Access	Description	Reset
4:0	Reserved			
5	reg_fix_mc	R/W	mde window output fix color enable Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:6	Reserved			

REG_39

Offset Address: 0x09c

Bits	Name	Access	Description	Reset
15:0	Reserved			
25:16	reg_fde_r	R/W	fde window r color	0x0
31:26	Reserved			

REG_40

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
9:0	reg_fde_g	R/W	fde window g color	0x0
15:10	Reserved			
25:16	reg_fde_b	R/W	fde window b color	0x3ff
31:26	Reserved			

REG_41

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
9:0	reg_mde_r	R/W	mde window r color	0x3ff
15:10	Reserved			
25:16	reg_mde_g	R/W	mde window g color	0x0

Bits	Name	Access	Description	Reset
31:26	Reserved			

REG_42

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
9:0	reg_mde_b	R/W	mde window b color	0x0
15:10	Reserved			
17:16	reg_out_bit	R/W	2 : 8-bit 3 : 6-bit others : 10-bit	0x2
19:18	reg_drop_md	R/W	2 : rounding 3 : direct drop others : noise dither	0x1
31:20	Reserved			

REG_43

Offset Address: 0x0ac

Bits	Name	Access	Description	Reset
0	reg_disp_bw_fail	RO	data not ready for display, Low Bandwidth interrupt	
1	reg_clr_disp_bw_fail	W1T	clear reg_disp_bw_fail	
2	reg_osd_bw_fail	RO	OSD data not ready for display, Low Bandwidth interrupt	
3	reg_clr_osd_bw_fail	W1T	clear reg_osd_bw_fail	
31:4	Reserved			

REG_CATCH

Offset Address: 0x0c0

Bits	Name	Access	Description	Reset
1:0	Reserved			
2	reg_qos_sel_rr	R/W	qos criteria selection 0 : data in buffer + outstanding 1 : data in buffer only	0x1
31:3	Reserved			

REG_OSD_FIFO_M0

Offset Address: 0x130

Bits	Name	Access	Description	Reset
11:0	reg_osd_thres_0	R/W	osd fifo monitor threshold value 0	0x10
15:12	Reserved			
27:16	reg_osd_thres_1	R/W	osd fifo monitor threshold value 1	0x18
31:28	Reserved			

REG_OSD_FIFO_M1

Offset Address: 0x134

Bits	Name	Access	Description	Reset
11:0	reg_osd_thres_2	R/W	osd fifo monitor threshold value 2	0x20
15:12	Reserved			
27:16	reg_osd_thres_3	R/W	osd fifo monitor threshold value 3	0x28
31:28	Reserved			

REG_OSD_FIFO_CNT0

Offset Address: 0x138

Bits	Name	Access	Description	Reset
31:0	reg_osd_rd_cnt_0	RO	osd fifo data less than threshold value 0 count	

REG_OSD_FIFO_CNT1

Offset Address: 0x13c

Bits	Name	Access	Description	Reset
31:0	reg_osd_rd_cnt_1	RO	osd fifo data less than threshold value 1 count	

REG_OSD_FIFO_CNT2

Offset Address: 0x140

Bits	Name	Access	Description	Reset
31:0	reg_osd_rd_cnt_2	RO	osd fifo data less than threshold value 2 count	

REG_OSD_FIFO_CNT3

Offset Address: 0x144

Bits	Name	Access	Description	Reset
31:0	reg_osd_rd_cnt_3	RO	osd fifo data less than threshold value 3 count	

REG_GAMMA_CTRL

Offset Address: 0x180

Bits	Name	Access	Description	Reset
0	reg_gamma_acc_lut	R/W	apb access gamma lut table	0x0
1	reg_gamma_acc_wr	R/W	0 : read , 1 : write	0x0
2	reg_gamma_en	R/W	gamma enable Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
3	reg_gamma_pre_osd	R/W	1 : gamma -> osd 0 : osd -> gamma Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:4	Reserved			

REG_GAMMA_WR_LUT

Offset Address: 0x184

Bits	Name	Access	Description	Reset
7:0	reg_gamma_b_wdat	R/W	B gamma Lut write data	0x0
15:8	reg_gamma_g_wdat	R/W	G gamma Lut write data	0x0
23:16	reg_gamma_r_wdat	R/W	R gamma Lut write data	0x0
30:24	reg_gamma_addr	R/W	gamma lut address	0x0
31	reg_gamma_acc_w1t	W1T	write 1 to access lut (wrtie)	

REG_MCU_IF_CTRL

Offset Address: 0x200

Bits	Name	Access	Description	Reset
0	reg_i80_if_en	R/W	display S.W. mcu interface enable (read prepared data from DRAM)	0x0
1	reg_i80_sw_mode_en	R/W	mcu sw step by step mode enable	0x0

Bits	Name	Access	Description	Reset
2	reg_i80_hw_if_en	R/W	display H.W. mcu interface enable (read RGB data fromDRAM) This mode is mutually exclusive with reg_i80_if_en, and the control is in REG_HW_MCU related register group	0x0
3	Reserved			
7:4	reg_i80_ctrl_ini	R/W	Control signal initial value	0xf
9:8	Reserved			
10	reg_i80_ip_clr_w1t	W1T	write 1 to clear mcu interface IP	
11	reg_i80_run_w1t	W1T	write 1 to start mcu interface IP	
31:12	Reserved			

REG_HW_MCU_AUTO

Offset Address: 0x210

Bits	Name	Access	Description	Reset
0	reg_mcu_hw_trig	R/W	rising edge to start H.W MCU	0x0
1	reg_mcu_hw_stop	R/W	rising edge to stop H.W. MCU	0x0
3:2	reg_cs_h_hw_blk	R/W	csx signal state in video blanking duration 2'b00 : csx high in h-blanking & v-blanking 2'b01 : csx high in v-blanking 2'b1x : csx high when not sending video data	0x2
4	reg_mcu_565	R/W	H.W. MCU data out format 0 : RGB565, 2 TX cycle per pix 1 : RGB888, 3 TX cycle per pix (RGB666) (also with this mode)	0x1
5	reg_pre_cmd_en	R/W	1 : insert cmd before H.W. MCU data send use reg_sw_tx related config, if this bit set to high a. please set reg_sw_tx related config b. set hfde_str number > reg_sw_tx_num	0x0
6	reg_hw_mcu_start_flag	RWC	H.W. MCU interface starts to send data, the write 1 to clear	
7	reg_hw_mcu_stop_flag	RWC	H.W. MCU interace stop to send data. write 1 to clear	
31:8	Reserved			

REG_HW_MCU_CMD

Offset Address: 0x214

Bits	Name	Access	Description	Reset
0	reg_mcu_sw_trig	R/W	rising edge to trig S.W. command command number is set in reg_mcu_sw_tx_num (max 16 commands per trig) command queue is set in reg_sw_tx0 ~ reg_sw_txf	0x0
1	reg_cs_h_sw_idle	R/W	csx signal state after S.W. trig finish 1'b0 : keep low 1'b1 : go high	0x0
2	Reserved			

Bits	Name	Access	Description	Reset
3	reg_mcu_sw_tx_done	RWC	S.W. TX done flag, write 1 to clear	
7:4	reg_mcu_sw_tx_num	R/W	H.W. MCU engine send s.w. command number(0 ~ 15) -> (1~16)	0x0
11:8	reg_hw_mcu_state	RO	H.W. MCU status 0 : IDLE 1 : SW_TX 2 : WAIT_VS 3 : BLK 4 : HW_TX0 5 : HW_TX1 6 : HW_TX2	
15:12	reg_sw_tx_cnt_ro	RO	S.W. TX number by H.W. when finish, and This value is set to 0	
31:16	Reserved			

REG_HW_MCU_CMD_0

Offset Address: 0x218

Bits	Name	Access	Description	Reset
9:0	reg_sw_tx_0	R/W	S.W. TX data [9:8] : OP code 2'b00 : command write 2'b01 : data write 2'b10 : command read 2'b11 : data read [7:0] TX data	0x0
15:10	Reserved			
25:16	reg_sw_tx_1	R/W		0x0
31:26	Reserved			

REG_HW_MCU_CMD_1

Offset Address: 0x21c

Bits	Name	Access	Description	Reset
9:0	reg_sw_tx_2	R/W		0x0
15:10	Reserved			
25:16	reg_sw_tx_3	R/W		0x0
31:26	Reserved			

REG_HW_MCU_CMD_2

Offset Address: 0x220

Bits	Name	Access	Description	Reset
9:0	reg_sw_tx_4	R/W		0x0
15:10	Reserved			
25:16	reg_sw_tx_5	R/W		0x0
31:26	Reserved			

REG_HW_MCU_CMD_3

Offset Address: 0x224

Bits	Name	Access	Description	Reset
9:0	reg_sw_tx_6	R/W		0x0
15:10	Reserved			

Bits	Name	Access	Description	Reset
25:16	reg_sw_tx_7	R/W		0x0
31:26	Reserved			

REG_HW_MCU_CMD_4

Offset Address: 0x228

Bits	Name	Access	Description	Reset
9:0	reg_sw_tx_8	R/W		0x0
15:10	Reserved			
25:16	reg_sw_tx_9	R/W		0x0
31:26	Reserved			

REG_HW_MCU_CMD_5

Offset Address: 0x22c

Bits	Name	Access	Description	Reset
9:0	reg_sw_tx_a	R/W		0x0
15:10	Reserved			
25:16	reg_sw_tx_b	R/W		0x0
31:26	Reserved			

REG_HW_MCU_CMD_6

Offset Address: 0x230

Bits	Name	Access	Description	Reset
9:0	reg_sw_tx_c	R/W		0x0
15:10	Reserved			
25:16	reg_sw_tx_d	R/W		0x0
31:26	Reserved			

REG_HW_MCU_CMD_7

Offset Address: 0x234

Bits	Name	Access	Description	Reset
9:0	reg_sw_tx_e	R/W		0x0
15:10	Reserved			
25:16	reg_sw_tx_f	R/W		0x0
31:26	Reserved			

REG_HW_MCU_OV

Offset Address: 0x238

Bits	Name	Access	Description	Reset
0	reg_i80_wrx_sw_ov	R/W	S.W. over-write i80 WRX enable	0x0
1	reg_i80_rdx_sw_ov	R/W		0x0
2	reg_i80_cdx_sw_ov	R/W		0x0
3	reg_i80_csx_sw_ov	R/W		0x0
4	reg_i80_dat_sw_ov	R/W		0x0
7:5	Reserved			
8	reg_i80_wrx_sw_dat	R/W		0x0
9	reg_i80_rdx_sw_dat	R/W		0x0

Bits	Name	Access	Description	Reset
10	reg_i80_cdx_sw_dat	R/W		0x0
11	reg_i80_csx_sw_dat	R/W		0x0
15:12	Reserved			
23:16	reg_i80_dat_sw_dat	R/W		0x0
31:24	Reserved			

REG_SRGB_CTRL

Offset Address: 0x240

Bits	Name	Access	Description	Reset
0	reg_srgb_ttl_en	R/W	serial RGB TTL interface enable	0x0
1	reg_srgb_ttl_4t	R/W	serial RGB output cycle per pixel 0 : 3T 1 : 4T	0x0
3:2	Reserved			
5:4	reg_srgb_ttl_g_num	R/W	sw overite serial RGB out G pixel idx. effective when reg_srgb_ttl_sw_seq = 1	0x0
7:6	reg_srgb_ttl_b_num	R/W	sw overite serial RGB out B pixel idx, effective when reg_srgb_ttl_sw_seq = 1	0x0
8	reg_srgb_ttl_sw_seq	R/W	sw redefine serial RGB output squence	0x0
31:9	Reserved			

COV_W0_CFG

Offset Address: 0x280

Bits	Name	Access	Description	Reset
11:0	reg_cover_w0_x_str	R/W	cover window 0 horizontal start (include this point) Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:12	Reserved			
27:16	reg_cover_w0_y_str	R/W	cover window 0 verticlal start (include this point) Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
30:28	Reserved			
31	reg_cover_w0_enable	R/W	cover window 0 enable Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

COV_W0_SIZE

Offset Address: 0x284

Bits	Name	Access	Description	Reset
11:0	reg_cover_w0_wd_m1	R/W	cover window 0 widh minus 1 value Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:12	Reserved			
27:16	reg_cover_w0_ht_m1	R/W	cover window 0 height minus 1 value Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

Bits	Name	Access	Description	Reset
31:28	Reserved			

COV_W0_COLOR

Offset Address: 0x288

Bits	Name	Access	Description	Reset
7:0	reg_cover_w0_r	R/W	cover window 0 r color	0x0
15:8	reg_cover_w0_g	R/W	cover window 0 g color	0x0
23:16	reg_cover_w0_b	R/W	cover window 0 b color	0x0
31:24	Reserved			

COV_W1_CFG

Offset Address: 0x28c

Bits	Name	Access	Description	Reset
11:0	reg_cover_w1_x_str	R/W	cover window1 horizontal start (include this point) Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:12	Reserved			
27:16	reg_cover_w1_y_str	R/W	cover window1 verticlal start (include this point) Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
30:28	Reserved			
31	reg_cover_w1_enable	R/W	cover window 1 enable Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

COV_W1_SIZE

Offset Address: 0x290

Bits	Name	Access	Description	Reset
11:0	reg_cover_w1_wd_m1	R/W	cover window 1 widh minus 1 value Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:12	Reserved			
27:16	reg_cover_w1_ht_m1	R/W	cover window 1 height minus 1 value Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:28	Reserved			

COV_W1_COLOR

Offset Address: 0x294

Bits	Name	Access	Description	Reset
7:0	reg_cover_w1_r	R/W	cover window 1 r color	0x0
15:8	reg_cover_w1_g	R/W	cover window 1 g color	0x0
23:16	reg_cover_w1_b	R/W	cover window 1 b color	0x0
31:24	Reserved			

COV_W2_CFG

Offset Address: 0x298

Bits	Name	Access	Description	Reset
11:0	reg_cover_w2_x_str	R/W	cover window 2 horizontal start (include this point) Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:12	Reserved			
27:16	reg_cover_w2_y_str	R/W	cover window 2 verticlal start (include this point) Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
30:28	Reserved			
31	reg_cover_w2_enable	R/W	cover window 2 enable Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

COV_W2_SIZE

Offset Address: 0x29c

Bits	Name	Access	Description	Reset
11:0	reg_cover_w2_wd_m1	R/W	cover window 2 widh minus 1 value Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:12	Reserved			
27:16	reg_cover_w2_ht_m1	R/W	cover window 2 height minus 1 value Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:28	Reserved			

COV_W2_COLOR

Offset Address: 0x2a0

Bits	Name	Access	Description	Reset
7:0	reg_cover_w2_r	R/W	cover window 2 r color	0x0
15:8	reg_cover_w2_g	R/W	cover window 2 g color	0x0
23:16	reg_cover_w2_b	R/W	cover window 2 b color	0x0
31:24	Reserved			

COV_W3_CFG

Offset Address: 0x2a4

Bits	Name	Access	Description	Reset
11:0	reg_cover_w3_x_str	R/W	cover window 3 horizontal start (include this point) Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:12	Reserved			
27:16	reg_cover_w3_y_str	R/W	cover window 3 verticlal start (include this point) Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

Bits	Name	Access	Description	Reset
30:28	Reserved			
31	reg_cover_w3_enable	R/W	cover window 3 enable Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0

COV_W3_SIZE

Offset Address: 0x2a8

Bits	Name	Access	Description	Reset
11:0	reg_cover_w3_wd_m1	R/W	cover window 3 widh minus 1 value Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
15:12	Reserved			
27:16	reg_cover_w3_ht_m1	R/W	cover window 3 height minus 1 value Shadow: Yes Shadow Ctrl: up_1t Shadow Read Select: shrd_sel	0x0
31:28	Reserved			

COV_W3_COLOR

Offset Address: 0x2ac

Bits	Name	Access	Description	Reset
7:0	reg_cover_w3_r	R/W	cover window 3 r color	0x0
15:8	reg_cover_w3_g	R/W	cover window 3 g color	0x0
23:16	reg_cover_w3_b	R/W	cover window 3 b color	0x0
31:24	Reserved			

REG_TGEN_LITE_SIZE

Offset Address: 0x304

Bits	Name	Access	Description	Reset
13:0	reg_vtt_lite	R/W	vtotal Shadow: Yes Shadow Ctrl: up_1t_lite Shadow Read Select: shrd_sel	0x167
15:14	Reserved			
29:16	reg_htt_lite	R/W	htotal Shadow: Yes Shadow Ctrl: up_1t_lite Shadow Read Select: shrd_sel	0x1df
31:30	Reserved			

REG_TGEN_LITE_VS

Offset Address: 0x308

Bits	Name	Access	Description	Reset
13:0	reg_vs_str_lite	R/W	vsync start Shadow: Yes Shadow Ctrl: up_1t_lite Shadow Read Select: shrd_sel	0x1
15:14	Reserved			
29:16	reg_vs_stp_lite	R/W	vsync end Shadow: Yes Shadow Ctrl: up_1t_lite Shadow Read Select: shrd_sel	0x3

Bits	Name	Access	Description	Reset
31:30	Reserved			

REG_TGEN_LITE_HS

Offset Address: 0x314

Bits	Name	Access	Description	Reset
13:0	reg_hs_str_lite	R/W	hsync start Shadow: Yes Shadow Ctrl: up_1t_lite Shadow Read Select: shrd_sel	0x1
15:14	Reserved			
29:16	reg_hs_stp_lite	R/W	hsync end Shadow: Yes Shadow Ctrl: up_1t_lite Shadow Read Select: shrd_sel	0xa
31:30	Reserved			

VDP_OSD Register Description

VGOP_REG_0

Offset Address: 0x000

Bits	Name	Access	Description	Reset
3:0	reg_ow0_format	R/W	OW0 Format 4'b0000: ARGB8888 4'b0100: ARGB4444 4'b0101: ARGB1555 4'b1000: 256LUT-ARGB4444 4'b1010: 16-LUT-ARGB4444 4'b1100: Font-base Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:4	Reserved			

VGOP_REG_1

Offset Address: 0x004

Bits	Name	Access	Description	Reset
11:0	reg_ow0_h_start	R/W	OW0 H start pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow0_h_end	R/W	OW0 H end pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_2

Offset Address: 0x008

Bits	Name	Access	Description	Reset
11:0	reg_ow0_v_start	R/W	OW0 V start pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update	0x0

Bits	Name	Access	Description	Reset
			Shadow Read Select: shdw_read_sel	
15:12	Reserved			
27:16	reg_ow0_v_end	R/W	OW0 V end pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_3

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
3:0	Reserved			
31:4	reg_ow0_dram_str_adr_l	R/W	OW0 DRAM Start address[31:0] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_4

Offset Address: 0x010

Bits	Name	Access	Description	Reset
7:0	reg_ow0_dram_str_adr_h	R/W	OW0 DRAM Start address[39:32] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:8	Reserved			

VGOP_REG_5

Offset Address: 0x014

Bits	Name	Access	Description	Reset
3:0	Reserved			
15:4	reg_ow0_dram_strip	R/W	OW0 DRAM Strip Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
22:16	reg_ow0_crop_pixels	R/W	OW0 crop pixels (per line) Format ARGB8888, crop pixels valid value: 0~3 Format ARGB4444/ARGB1555, crop pixels valid value: 0~7 Format 256LUT, crop pixels valid value: 0~15 Format 16-LUT, crop pixels valid value: 0~31 Format Font-base, crop pixels valid value: 0~127 Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:23	Reserved			

VGOP_REG_6

Offset Address: 0x018

Bits	Name	Access	Description	Reset

3:0	Reserved			

Bits	Name	Access	Description	Reset
13:4	reg_ow0_dram_hsize	R/W	OW0 DRAM Hsize ** while reg_odec_en=1, odec_stream_size = {reg_ow_dram_vsize,reg_ow_dram_hsize} Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:14	Reserved			
27:16	reg_ow0_dram_vsize	R/W	OW0 DRAM Vsize Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_10

Offset Address: 0x020

Bits	Name	Access	Description	Reset
3:0	reg_ow1_format	R/W	OW1 Format 4'b0000: ARGB8888 4'b0100: ARGB4444 4'b0101: ARGB1555 4'b1000: 256LUT-ARGB4444 4'b1010: 16-LUT-ARGB4444 4'b1100: Font-base Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:4	Reserved			

VGOP_REG_11

Offset Address: 0x024

Bits	Name	Access	Description	Reset
11:0	reg_ow1_h_start	R/W	OW1 H start pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow1_h_end	R/W	OW1 H end pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_12

Offset Address: 0x028

Bits	Name	Access	Description	Reset
11:0	reg_ow1_v_start	R/W	OW1 V start pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow1_v_end	R/W	OW1 V end pixel, unit: 1line Shadow: Yes	0x0

Bits	Name	Access	Description	Reset
			Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	
31:28	Reserved			

VGOP_REG_13

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
3:0	Reserved			
31:4	reg_ow1_dram_str_adr_l	R/W	OW1 DRAM Start address[31:0] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_14

Offset Address: 0x030

Bits	Name	Access	Description	Reset
7:0	reg_ow1_dram_str_adr_h	R/W	OW1 DRAM Start address[39:32] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:8	Reserved			

VGOP_REG_15

Offset Address: 0x034

Bits	Name	Access	Description	Reset
3:0	Reserved			
15:4	reg_ow1_dram_strip	R/W	OW1 DRAM Strip Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
22:16	reg_ow1_crop_pixels	R/W	OW1 crop pixels (per line) Format ARGB8888, crop pixels valid value: 0~3 Format ARGB4444/ARGB1555, crop pixels valid value: 0~7 Format 256LUT, crop pixels valid value: 0~15 Format 16-LUT, crop pixels valid value: 0~31 Format Font-base, crop pixels valid value: 0~127 Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:23	Reserved			

VGOP_REG_16

Offset Address: 0x038

Bits	Name	Access	Description	Reset
3:0	Reserved			

13:4	reg_ow1_dram_hsize	R/W	OW1 DRAM Hsize ** while reg_odec_en=1, odec_stream_size = {reg_ow_dram_vsize,reg_ow_dram_hsiz	0x0

Bits	Name	Access	Description	Reset
			e} Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	
15:14	Reserved			
27:16	reg_ow1_dram_vsize	R/W	OW1 DRAM Vsize Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_20

Offset Address: 0x040

Bits	Name	Access	Description	Reset
3:0	reg_ow2_format	R/W	OW2 Format 4'b0000: ARGB8888 4'b0100: ARGB4444 4'b0101: ARGB1555 4'b1000: 256LUT-ARGB4444 4'b1010: 16-LUT-ARGB4444 4'b1100: Font-base Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:4	Reserved			

VGOP_REG_21

Offset Address: 0x044

Bits	Name	Access	Description	Reset
11:0	reg_ow2_h_start	R/W	OW2 H start pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow2_h_end	R/W	OW2 H end pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_22

Offset Address: 0x048

Bits	Name	Access	Description	Reset
11:0	reg_ow2_v_start	R/W	OW2 V start pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow2_v_end	R/W	OW2 V end pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_23

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
3:0	Reserved			
31:4	reg_ow2_dram_str_adr_l	R/W	OW2 DRAM Start address[31:0] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_24

Offset Address: 0x050

Bits	Name	Access	Description	Reset
7:0	reg_ow2_dram_str_adr_h	R/W	OW2 DRAM Start address[39:32] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:8	Reserved			

VGOP_REG_25

Offset Address: 0x054

Bits	Name	Access	Description	Reset
3:0	Reserved			
15:4	reg_ow2_dram_strip	R/W	OW2 DRAM Strip Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
22:16	reg_ow2_crop_pixels	R/W	OW2 crop pixels (per line) Format ARGB8888, crop pixels valid value: 0~3 Format ARGB4444/ARGB1555, crop pixels valid value: 0~7 Format 256LUT, crop pixels valid value: 0~15 Format 16-LUT, crop pixels valid value: 0~31 Format Font-base, crop pixels valid value: 0~127 Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:23	Reserved			

VGOP_REG_26

Offset Address: 0x058

Bits	Name	Access	Description	Reset
3:0	Reserved			
13:4	reg_ow2_dram_hsize	R/W	OW2 DRAM Hsize ** while reg_odec_en=1, odec_stream_size = {reg_ow_dram_vsize,reg_ow_dram_hsize} Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

15:14	Reserved		

Bits	Name	Access	Description	Reset
27:16	reg_ow2_dram_vsize	R/W	OW2 DRAM Vsize Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_30

Offset Address: 0x060

Bits	Name	Access	Description	Reset
3:0	reg_ow3_format	R/W	OW3 Format 4'b0000: ARGB8888 4'b0100: ARGB4444 4'b0101: ARGB1555 4'b1000: 256LUT-ARGB4444 4'b1010: 16-LUT-ARGB4444 4'b1100: Font-base Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:4	Reserved			

VGOP_REG_31

Offset Address: 0x064

Bits	Name	Access	Description	Reset
11:0	reg_ow3_h_start	R/W	OW3 H start pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow3_h_end	R/W	OW3 H end pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_32

Offset Address: 0x068

Bits	Name	Access	Description	Reset
11:0	reg_ow3_v_start	R/W	OW3 V start pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow3_v_end	R/W	OW3 V end pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_33

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
3:0	Reserved			
31:4	reg_ow3_dram_str_adr_l	R/W	OW3 DRAM Start address[31:0]	0x0

Bits	Name	Access	Description	Reset
			Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	

VGOP_REG_34

Offset Address: 0x070

Bits	Name	Access	Description	Reset
7:0	reg_ow3_dram_str_adr_h	R/W	OW3 DRAM Start address[39:32] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:8	Reserved			

VGOP_REG_35

Offset Address: 0x074

Bits	Name	Access	Description	Reset
3:0	Reserved			
15:4	reg_ow3_dram_strip	R/W	OW3 DRAM Strip Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
22:16	reg_ow3_crop_pixels	R/W	OW3 crop pixels (per line) Format ARGB8888, crop pixels valid value: 0~3 Format ARGB4444/ARGB1555, crop pixels valid value: 0~7 Format 256LUT, crop pixels valid value: 0~15 Format 16-LUT, crop pixels valid value: 0~31 Format Font-base, crop pixels valid value: 0~127 Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:23	Reserved			

VGOP_REG_36

Offset Address: 0x078

Bits	Name	Access	Description	Reset
3:0	Reserved			
13:4	reg_ow3_dram_hsize	R/W	OW3 DRAM Hsize ** while reg_odec_en=1, odec_stream_size = {reg_ow_dram_vsize,reg_ow_dram_hsize} Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:14	Reserved			

27:16	reg_ow3_dram_vsize	R/W	OW3 DRAM Vsize Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

Bits	Name	Access	Description	Reset
31:28	Reserved			

VGOP_REG_40

Offset Address: 0x080

Bits	Name	Access	Description	Reset
3:0	reg_ow4_format	R/W	OW4 Format 4'b0000: ARGB8888 4'b0100: ARGB4444 4'b0101: ARGB1555 4'b1000: 256LUT-ARGB4444 4'b1010: 16-LUT-ARGB4444 4'b1100: Font-base Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:4	Reserved			

VGOP_REG_41

Offset Address: 0x084

Bits	Name	Access	Description	Reset
11:0	reg_ow4_h_start	R/W	OW4 H start pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow4_h_end	R/W	OW4 H end pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_42

Offset Address: 0x088

Bits	Name	Access	Description	Reset
11:0	reg_ow4_v_start	R/W	OW4 V start pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow4_v_end	R/W	OW4 V end pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_43

Offset Address: 0x08c

Bits	Name	Access	Description	Reset
3:0	Reserved			
31:4	reg_ow4_dram_str_adr_l	R/W	OW4 DRAM Start address[31:0] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_44

Offset Address: 0x090

Bits	Name	Access	Description	Reset
7:0	reg_ow4_dram_str_adr_h	R/W	OW4 DRAM Start address[39:32] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:8	Reserved			

VGOP_REG_45

Offset Address: 0x094

Bits	Name	Access	Description	Reset
3:0	Reserved			
15:4	reg_ow4_dram_strip	R/W	OW4 DRAM Strip Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
22:16	reg_ow4_crop_pixels	R/W	OW4 crop pixels (per line) Format ARGB8888, crop pixels valid value: 0~3 Format ARGB4444/ARGB1555, crop pixels valid value: 0~7 Format 256LUT, crop pixels valid value: 0~15 Format 16-LUT, crop pixels valid value: 0~31 Format Font-base, crop pixels valid value: 0~127 Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:23	Reserved			

VGOP_REG_46

Offset Address: 0x098

Bits	Name	Access	Description	Reset
3:0	Reserved			
13:4	reg_ow4_dram_hsize	R/W	OW4 DRAM Hsize ** while reg_odec_en=1, odec_stream_size = {reg_ow_dram_vsize,reg_ow_dram_hsize} Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:14	Reserved			
27:16	reg_ow4_dram_vsize	R/W	OW4 DRAM Vsize Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_50

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
3:0	reg_ow5_format	R/W	OW5 Format 4'b0000: ARGB8888 4'b0100: ARGB4444 4'b0101: ARGB1555 4'b1000: 256LUT-ARGB4444 4'b1010: 16-LUT-ARGB4444 4'b1100: Font-base Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:4	Reserved			

VGOP_REG_51

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
11:0	reg_ow5_h_start	R/W	OW5 H start pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow5_h_end	R/W	OW5 H end pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_52

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
11:0	reg_ow5_v_start	R/W	OW5 V start pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow5_v_end	R/W	OW5 V end pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_53

Offset Address: 0x0ac

Bits	Name	Access	Description	Reset
3:0	Reserved			
31:4	reg_ow5_dram_str_adr_l	R/W	OW5 DRAM Start address[31:0] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_54

Offset Address: 0x0b0

Bits	Name	Access	Description	Reset
7:0	reg_ow5_dram_str_adr_h	R/W	OW5 DRAM Start address [39:32] Shadow: Yes	0x0

Bits	Name	Access	Description	Reset
			Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	
31:8	Reserved			

VGOP_REG_55

Offset Address: 0x0b4

Bits	Name	Access	Description	Reset
3:0	Reserved			
15:4	reg_ow5_dram_strip	R/W	OW5 DRAM Strip Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
22:16	reg_ow5_crop_pixels	R/W	OW5 crop pixels (per line) Format ARGB8888, crop pixels valid value: 0~3 Format ARGB4444/ARGB1555, crop pixels valid value: 0~7 Format 256LUT, crop pixels valid value: 0~15 Format 16-LUT, crop pixels valid value: 0~31 Format Font-base, crop pixels valid value: 0~127 Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:23	Reserved			

VGOP_REG_56

Offset Address: 0x0b8

Bits	Name	Access	Description	Reset
3:0	Reserved			
13:4	reg_ow5_dram_hsize	R/W	OW5 DRAM Hsize ** while reg_odec_en=1, odec_stream_size = {reg_ow_dram_vsize,reg_ow_dram_hsize} Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:14	Reserved			
27:16	reg_ow5_dram_vsize	R/W	OW5 DRAM Vsize Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_60

Offset Address: 0x0c0

Bits	Name	Access	Description	Reset

3:0	reg_ow6_format	R/W	OW6 Format 4'b0000: ARGB8888 4'b0100: ARGB4444 4'b0101: ARGB1555	0x0

Bits	Name	Access	Description	Reset
			4'b1000: 256LUT-ARGB4444 4'b1010: 16-LUT-ARGB4444 4'b1100: Font-base Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	
31:4	Reserved			

VGOP_REG_61

Offset Address: 0x0c4

Bits	Name	Access	Description	Reset
11:0	reg_ow6_h_start	R/W	OW6 H start pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow6_h_end	R/W	OW6 H end pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_62

Offset Address: 0x0c8

Bits	Name	Access	Description	Reset
11:0	reg_ow6_v_start	R/W	OW6 V start pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow6_v_end	R/W	OW6 V end pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_63

Offset Address: 0x0cc

Bits	Name	Access	Description	Reset
3:0	Reserved			
31:4	reg_ow6_dram_str_adr_l	R/W	OW6 DRAM Start address[31:0] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_64

Offset Address: 0x0d0

Bits	Name	Access	Description	Reset
7:0	reg_ow6_dram_str_adr_h	R/W	OW6 DRAM Start address[39:32] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:8	Reserved			

VGOP_REG_65

Offset Address: 0x0d4

Bits	Name	Access	Description	Reset
3:0	Reserved			
15:4	reg_ow6_dram_strip	R/W	OW6 DRAM Strip Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
22:16	reg_ow6_crop_pixels	R/W	OW6 crop pixels (per line) Format ARGB8888, crop pixels valid value: 0~3 Format ARGB4444/ARGB1555, crop pixels valid value: 0~7 Format 256LUT, crop pixels valid value: 0~15 Format 16-LUT, crop pixels valid value: 0~31 Format Font-base, crop pixels valid value: 0~127 Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:23	Reserved			

VGOP_REG_66

Offset Address: 0x0d8

Bits	Name	Access	Description	Reset
3:0	Reserved			
13:4	reg_ow6_dram_hsize	R/W	OW6 DRAM Hsize ** while reg_odec_en=1, odec_stream_size = {reg_ow_dram_vsize,reg_ow_dram_hsize} Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:14	Reserved			
27:16	reg_ow6_dram_vsize	R/W	OW6 DRAM Vsize Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_70

Offset Address: 0x0e0

Bits	Name	Access	Description	Reset

3:0	reg_ow7_format	R/W	OW7 Format 4'b0000: ARGB8888 4'b0100: ARGB4444 4'b0101: ARGB1555 4'b1000: 256LUT-ARGB4444 4'b1010: 16-LUT-ARGB4444 4'b1100: Font-base Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

Bits	Name	Access	Description	Reset
31:4	Reserved			

VGOP_REG_71

Offset Address: 0x0e4

Bits	Name	Access	Description	Reset
11:0	reg_ow7_h_start	R/W	OW7 H start pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow7_h_end	R/W	OW7 H end pixel, unit: 1pixel Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_72

Offset Address: 0x0e8

Bits	Name	Access	Description	Reset
11:0	reg_ow7_v_start	R/W	OW7 V start pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:12	Reserved			
27:16	reg_ow7_v_end	R/W	OW7 V end pixel, unit: 1line Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_73

Offset Address: 0x0ec

Bits	Name	Access	Description	Reset
3:0	Reserved			
31:4	reg_ow7_dram_str_adr_l	R/W	OW7 DRAM Start address[31:0] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_74

Offset Address: 0x0f0

Bits	Name	Access	Description	Reset
7:0	reg_ow7_dram_str_adr_h	R/W	OW7 DRAM Start address[39:32] Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:8	Reserved			

VGOP_REG_75

Offset Address: 0x0f4

Bits	Name	Access	Description	Reset
3:0	Reserved			
15:4	reg_ow7_dram_strip	R/W	OW7 DRAM Strip	0x0

Bits	Name	Access	Description	Reset
			Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	
22:16	reg_ow7_crop_pixels	R/W	OW7 crop pixels (per line) Format ARGB8888, crop pixels valid value: 0~3 Format ARGB4444/ARGB1555, crop pixels valid value: 0~7 Format 256LUT, crop pixels valid value: 0~15 Format 16-LUT, crop pixels valid value: 0~31 Format Font-base, crop pixels valid value: 0~127 Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:23	Reserved			

VGOP_REG_76

Offset Address: 0x0f8

Bits	Name	Access	Description	Reset
3:0	Reserved			
13:4	reg_ow7_dram_hsize	R/W	OW7 DRAM Hsize ** while reg_odec_en=1, odec_stream_size = {reg_ow_dram_vsize,reg_ow_dram_hsize} Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15:14	Reserved			
27:16	reg_ow7_dram_vsize	R/W	OW7 DRAM Vsize Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:28	Reserved			

VGOP_REG_80

osd_common_ctrl
Offset Address:
0x100

Bits	Name	Access	Description	Reset
0	reg_ow0_en	R/W	OSD Window0 enable Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
1	reg_ow1_en	R/W	OSD Window1 enable Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
2	reg_ow2_en	R/W	OSD Window2 enable Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

3	reg_ow3_en	R/W	OSD Window3 enable Shadow: Yes	0x0

Bits	Name	Access	Description	Reset
			Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	
4	reg_ow4_en	R/W	OSD Window4 enable Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
5	reg_ow5_en	R/W	OSD Window5 enable Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
6	reg_ow6_en	R/W	OSD Window6 enable Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
7	reg_ow7_en	R/W	OSD Window7 enable Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
8	reg_vgop_hscal	R/W	VGOP H scale up (x2) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
9	reg_vgop_vscal	R/W	VGOP V scale up (x2) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
10	reg_clr_key_en	R/W	Color Key enable Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
11	Reserved			
15:12	reg_vgop_arlen	R/W	AXI-R burst length (INCR mode) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
30:16	Reserved			
31	reg_vgop_sw_rst	R/W	vgop software reset Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_81

LUT256 SRAM

Offset Address: 0x104

Bits	Name	Access	Description	Reset
15:0	reg_idx_wdata	R/W	Idx sram wdata (ARGB4444)	0x0
23:16	reg_idx_adr	R/W	Idx sram address	0x0
31:24	Reserved			

VGOP_REG_82

LUT256 SRAM

Offset Address: 0x108

Bits	Name	Access	Description	Reset
15:0	reg_idx_rdata	R	idx sram rdata	
16	reg_idx_wr	R/W	idx sram write	0x0

Bits	Name	Access	Description	Reset
17	reg_idx_rd	R/W	idx sram read	0x0
18	reg_vgop_db_clr	R/W	vgop debug flag clear	0x0
31:19	Reserved			

VGOP_REG_83

color key content
Offset Address:
0x10c

Bits	Name	Access	Description	Reset
23:0	reg_clr_key	R/W	Color Key (RGB888) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:24	Reserved			

VGOP_REG_84

constant color
Offset Address: 0x110

Bits	Name	Access	Description	Reset
15:0	reg_const_argb0	R/W	Constant0 ARGB4444 for Font- base Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:16	reg_const_argb1	R/W	Constant1 ARGB4444 for Font- base Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_85

debug
Offset Address: 0x114

Bits	Name	Access	Description	Reset
31:0	reg_vgop_debug	R	debug	

VGOP_REG_86

fb_thr
Offset Address: 0x120

Bits	Name	Access	Description	Reset
5:0	reg_fb_clr_hi_thr	R/W	font_box brightness strong threshold , unit: 4 levels Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x30
7:6	Reserved			
13:8	reg_fb_clr_lo_thr	R/W	font_box brightness wrak threshold , unit: 4 levels Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x20
15:14	Reserved			
16	reg_fb_init	R/W	fb strong status init Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

17	reg_fb_font_is_dark	R/W	fb inv control	0x0

Bits	Name	Access	Description	Reset
			1 : font color is dark 0 : font color is light Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	
19:18	Reserved			
25:20	reg_fb_diff_fnum	R/W	font_box brightness detect interval 0: stop detection update 1: always detection update 2~63: detection update if n-1 continous frames detection value is all difference Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x1
31:26	Reserved			

VGOP_REG_87

fb0_setting

Offset Address: 0x124

Bits	Name	Access	Description	Reset
6:0	reg_fb0_width	R/W	font_box_0 width , unit: 1 pixel, 1~128 pixels (reg_fb_width+1) * (reg_fb_num+1) must be equal to attached ow width Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0f
7	Reserved			
12:8	reg_fb0_pix_thr	R/W	font_box_0 strong pixel pixel threshold, unit: 8 pixels Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x10
14:13	reg_fb0_sample_rate	R/W	font_box_0 sample rqtio 0: per pixel 1: 2 pixels 2: 4 pixels 3: 8 pixels Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x1
15	Reserved			
20:16	reg_fb0_num	R/W	font_box_0 box numbers 1~32 boxes Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x1f
23:21	Reserved			
26:24	reg_fb0_attached_idx	R/W	font_box_0 attached ow number Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
27	Reserved			
28	reg_fb0_en	R/W	font_box_0 enable (active while attached ow is alive and	0x0

Bits	Name	Access	Description	Reset
			set as font base fromat) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	
31:29	Reserved			

VGOP_REG_88

fb0_init_st
Offset Address: 0x128

Bits	Name	Access	Description	Reset
31:0	reg_fb0_init_st	R/W	font_box_0 init strong value Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_89

fb0_st_ro
Offset Address: 0x12c

Bits	Name	Access	Description	Reset
31:0	reg_fb0_record	R	font_box_0 strong value record	

VGOP_REG_90

fb1_setting
Offset Address: 0x134

Bits	Name	Access	Description	Reset
6:0	reg_fb1_width	R/W	font_box_1 width , unit: 1 pixel, 1~128 pixels (reg_fb_width+1) * (reg_fb_num+1) must be equal to attached ow width Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0f
7	Reserved			
12:8	reg_fb1_pix_thr	R/W	font_box_1 strong pixel pixel threshold, unit: 8 pixels Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x10
14:13	reg_fb1_sample_rate	R/W	font_box_1 sample rqtio 0: per pixel 1: 2 pixels 2: 4 pixels 3: 8 pixels Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x1
15	Reserved			
20:16	reg_fb1_num	R/W	font_box_1 box numbers 1~32 boxes Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x1f
23:21	Reserved			
26:24	reg_fb1_attached_idx	R/W	font_box_1 attached ow number Shadow: Yes	0x7

Bits	Name	Access	Description	Reset
			Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	
27	Reserved			
28	reg_fb1_en	R/W	font_box_1 enable (active while attached ow is alive and set as font base fromat) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:29	Reserved			

VGOP_REG_91

fb1_init_st

Offset Address: 0x138

Bits	Name	Access	Description	Reset
31:0	reg_fb1_init_st	R/W	font_box_1 init strong value Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_REG_92

fb1_st_ro

Offset Address: 0x13c

Bits	Name	Access	Description	Reset
31:0	reg_fb1_record	R	font_box_1 strong value record	

BW_LIMIT

Offset Address: 0x140

Bits	Name	Access	Description	Reset
9:0	reg_bwl_win	R/W	B.W. limit window period, unit : clk_axi cycle	0x0
15:10	Reserved			
25:16	reg_bwl_vld	R/W	B.W. limit valid number in reg_bwl_win, unit : Byte	0x0
30:26	Reserved			
31	reg_bwl_en	R/W	B.W. limit enable	0x0

VGOP_DEC_00

vgop_dec_ctrl

Offset Address: 0x150

Bits	Name	Access	Description	Reset
0	reg_odec_en	R/W	vgop decoder enable 0: bypass 1: decoder mode Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
1	reg_odec_int_en	R/W	vgop_decoder INT enable	0x0
2	reg_odec_int_clr	R/W	vgop_decoder INT clear	0x0
3	reg_odec_wdt_en	R/W	vgop wdt enable	0x0
7:4	reg_odec_dbg_ridx	R/W	vgop_decoder debug idx	0x0
8	reg_odec_done	R	vgop_decoder done	
11:9	Reserved			

Bits	Name	Access	Description	Reset
14:12	reg_odec_attached_idx	R/W	vgop decoder attached ow number Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
15	Reserved			
18:16	reg_odec_wdt_fdiv_bit	R/W	wdt precision	0x0
23:19	Reserved			
31:24	reg_odec_int_vec	R	vgop_decoder INT vector 1: odec done 2: Watch Dog time out 4: detect reduntant data input	

VGOP_DEC_01

vgop_dec_debug
Offset Address:
0x154

Bits	Name	Access	Description	Reset
31:0	reg_odec_dbg_rdata	R	vgop decoder debug rdata	

VGOP_LUT16_0

vgop_lut16_0_1
Offset Address:
0x160

Bits	Name	Access	Description	Reset
15:0	reg_lut16_cnt0	R/W	LUT 16 content 0 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:16	reg_lut16_cnt1	R/W	LUT 16 content 1 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_LUT16_1

vgop_lut16_2_3
Offset Address:
0x164

Bits	Name	Access	Description	Reset
15:0	reg_lut16_cnt2	R/W	LUT 16 content 2 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:16	reg_lut16_cnt3	R/W	LUT 16 content 3(ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_LUT16_2

vgop_lut16_4_5
Offset Address:
0x168

Bits	Name	Access	Description	Reset

15:0	reg_lut16_cnt4	R/W	LUT 16 content 4 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:16	reg_lut16_cnt5	R/W	LUT 16 content 5 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update	0x0

Bits	Name	Access	Description	Reset
			Shadow Read Select: shdw_read_sel	

VGOP_LUT16_3

vgop_lut16_6_7
Offset Address:
0x16c

Bits	Name	Access	Description	Reset
15:0	reg_lut16_cnt6	R/W	LUT 16 content 6 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:16	reg_lut16_cnt7	R/W	LUT 16 content 7 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_LUT16_4

vgop_lut16_8_9
Offset Address:
0x170

Bits	Name	Access	Description	Reset
15:0	reg_lut16_cnt8	R/W	LUT 16 content 8 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:16	reg_lut16_cnt9	R/W	LUT 16 content 9 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_LUT16_5

vgop_lut16_10_11
Offset Address:
0x174

Bits	Name	Access	Description	Reset
15:0	reg_lut16_cnt10	R/W	LUT 16 content 10 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:16	reg_lut16_cnt11	R/W	LUT 16 content 11 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_LUT16_6

vgop_lut16_12_13
Offset Address:
0x178

Bits	Name	Access	Description	Reset
15:0	reg_lut16_cnt12	R/W	LUT 16 content 12 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:16	reg_lut16_cnt13	R/W	LUT 16 content 13 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

VGOP_LUT16_7
vgop_lut16_14_15

Offset Address: 0x17c

Bits	Name	Access	Description	Reset
15:0	reg_lut16_cnt14	R/W	LUT 16 content 14 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0
31:16	reg_lut16_cnt15	R/W	LUT 16 content 15 (ARGB4444) Shadow: Yes Shadow Ctrl: shdw_update Shadow Read Select: shdw_read_sel	0x0

16.4 MIPI Rx

16.4.1 summarize

The MIPI Rx (Mobile Industry Processor Interface Receiver) module receives video data from CMOS sensors and supports MIPI D-PHY, sub-LVDS (Low-Voltage Differential), and MIPI Rx (Mobile Industry Processor Interface Receiver).
It processes different serial video signal inputs such as Signal, HiSPi (High-Speed Serial Pixel Interface), etc., and converts them into internal video timings that are passed on to the next level of the video processing module (ISP).
The MIPI Rx module can be subdivided into two parts: the PHY module integrates analog and digital components and mainly converts serial signals into parallel signals, while the Controller module is responsible for decoding different video data formats and transmitting them to the back-end video processing module (ISP). The functional block diagram and its location in the system are shown in Figure 16-20.

Figure *16-20 MIPI Rx* Functional Block Diagram and Location in the System

16.4.2 specificities

- Support for MIPI DPHY-ver2.1

- Supports 2 sensor inputs simultaneously
- Single sensor supports up to 5M (2688x1944, 2880x1620) @30fps HDR or @60fps linear inputs.
- Dual sensor supports up to FHD(1920x1080) @60fps HDR or linear inputs.
- Supports up to 4-Lane MIPI D-PHY interfaces up to 1.5Gbps/Lane on a single channel.
- Supports up to 4-Lane sub-LVDS/ HiSPi interfaces up to 1.5Gbps/Lane on a single channel.
- Supports parsing of RAW8/ RAW10/ RAW12/RAW16 data types.
- Supports parsing of YUV422 8-bit/ YUV422 10-bit data types.
- Supports up to 2 frames of WDR and multiple WDR timings
- Support sub-LVDS/ HiSPi mode pixel/sync code size end configuration
- Supports configurable number of lanes and order of lanes

16.4.3 Functional Description

16.4.3.1 typical application

For applications using image sensors, the MIPI Rx module registers can be set according to different interface options (MIPI/Sub-LVDS/ HiSPi), while the MIPI Rx also supports transmission requirements at different speeds and resolutions, and is compatible with a wide range of image sensor formats.

The MIPI Rx module contains one D-PHY with six differential pairs each. One D-PHY can support one differential clock pair with up to four differential data pairs, or two differential clock pairs with two differential data pairs, so the MIPI Rx can support two Sensor inputs at the same time. In addition, the MIPI Rx supports different differential pair ordering and clock differential pair positions, and the source of the clock and the differential pair ordering can be configured via registers.

MIPI Rx is only for the timing conversion and decoding of the interface, and does not deal with the image processing part, so it can support any resolution and frame rate as long as the bandwidth is satisfied. Therefore, it can support any resolution and frame rate under the premise of satisfying the bandwidth.The bandwidth of MIPI Rx is limited by two parts: the interface data rate of PHY and the internal processing speed. The input interface supports up to 1.5Gbps/Lane and the internal processing speed is up to 600M*1pixels/s. The MIPI Rx can support any resolution and frame rate as long as the bandwidth is satisfied.

	Common mode voltage	Differential mode voltage	Maximum clock frequency	Maximum data rate per lane
MIPI DPHY	200mV	200mV	750MHz	1.5Gbps
Sub-LVDS	900mV	150mV	750MHz	1.5Gbps

HiSPi (HiVCM)	900mV	280mV	750MHz	1.5Gbps
HiSPi (SLVDS)	200mV	200mV	750MHz	1.5Gbps

Figure *16-21 MIPI Rx* Supported Interface Types

16.4.3.2 MIPI Interface Data Format

The MIPI specification is developed and maintained by different working groups for different areas of application.MIPI Rx supports D-PHY and CSI-2 (Camera Serial Interface).D-PHY specifies the physical layer transmission specifications, CSI-2 specifies the Camera Output packet format and protocol.

- D-PHY

  D-PHY is a high-speed physical layer standard published by the MIPI Alliance that specifies the physical characteristics and transport protocols of the interface layer. D-PHY employs low-voltage differential signaling technology with 200mV source synchronization and supports a data rate range of up to 2500Mbps per lane.D-PHY can operate in both Low Power (LP) and High Speed (HS) modes.
- CSI-2

  CSI-2 is a data protocol for cameras that specifies the packet format for host-peripheral communication.

  CSI-2 can support image applications with different pixel formats, and the minimum granularity of data transmission is bytes. To increase the performance of CSI-2, the number of data lanes can be selected, and the CSI-2 protocol specifies a mechanism for the sender to pack pixel data into bytes, and specifies how multiple data lanes are allocated and managed. The byte data is organized in the form of packets, which are transmitted between the SoT and the EoT. The receiver parses the corresponding packets according to the protocol and recovers the original pixel data.

  MIPI Rx supports parsing of RAW8/ RAW10/ RAW12/RAW16/ YUV422-8bit/ YUV422-10bit data types.

  CSI-2 packets are categorized into long and short packets, contain checksums, and are capable of bit error correction and error detection.

Both long and short packets are transmitted between the SoT and the EoT, and the D-PHY is in LP mode during the data transfer gap.

The transmission mechanism of CSI-2 packet is shown in Fig. PH and PF represent Packet Header and Packet Footer respectively.

Figure *16-22* Packet Transmission Mechanism

Long packets are used to transmit valid pixel data and are divided into five parts: Data ID, Word Count, ECC, PAYLOAD, and CHECKSUM.

The Data ID consists of a Virtual Channel and a Data Type, the Virtual Channel controls

the channel used for the transfer, and can be used to transfer different data on different channels, while the Data Type specifies the type of data being transferred.

Word Count indicates the amount of data to be received by the receiver.

ECC is an error correction code that corrects or detects Data Type and Word Count errors.

PAYLOAD is the actual pixel data to be transmitted.

CHECKSUM is a checksum generated using a linear feedback shift register and is used for checksumming PAYLOAD data. The structure of a long packet is shown in Figure 16-23.

Figure *16-23 CSI-2* Long Packet Format

The function of the short packet is to synchronize the transmission of information, including the Data ID, Data Field and ECC three parts, the format is as shown in the figure.

16-24 shown.

Figure *16-24 CSI-2* Short Packet Format

The MIPI Rx supports the transmission of six video formats, including YUV422-

8bit, YUV422-10bit, RAW8, RAW10, RAW12, and RAW16, and the transmission methods for each format are described below.

The transmission mode for YUV422-8bit is in the form of UYVY, as shown in Figure 16-25.

Figure *16-25 YUV422 8-bit* Data Transmission Sequence

The correspondence of the packet to the video signal is shown in Figure 16-26.

Figure 109 YUV422 8-bit Transmission

Figure *16-26 YUV422-8bit* Data Packet Transmission Correspondence

The transmission format of the entire Frame becomes as shown in Figure 16-27.

FS	Packet Header, PH											Packet Footer, PF	FE
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		
	U	Y	*V*	Y	U	....	Y	U	Y	*V*	Y		

Figure *16-27 YUV422 8-bit Frame* Format

The transmission mode for YUV422-10bit is also UYVY and the transmission sequence is shown in Figure 16-28.

Line Start	Packet Header	**U1[9:2]**	Y1[9:2]	***V1[9:2]***	Y2[9:2]	LSB's	
Line End		**U639[9:2]**	Y639[9:2]	***V639[9:2]***	Y640[9:2]	LSB's	Packet Footer

Figure *16-28 YUV422-10bit* Data Transmission Sequence

The correspondence of the packet to the video signal is shown in Figure 16-29.

Figure *16-29 YUV422-10bit* Data Packet Transmission Correspondence

The transmission format of the entire Frame becomes as shown in Figure 16-30.

Figure *16-30 YUV422-10bit Frame* Format

The transmission sequence of RAW8 is shown in Figure 16-31.

Figure *16-31 RAW8* Data Transfer Sequence

The transmission format of the entire Frame becomes as shown in Figure 16-32.

Figure *16-32 RAW8 Frame* Format

The transmission sequence for RAW10 is shown in Figure 16-33.

Figure *16-33 RAW10* Data Transfer Sequence

The transmission format of the entire Frame becomes as shown in Figure 16-34.

FS	Packer Header, PH													Packer Footer, PF	
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			
	P1	P2	*P3*	P4	LSBs	P5	....	P637	P638	*P639*	P640	LSBs			FE

Figure *16-34 RAW10 Frame* Format

The RAW12 transmission sequence is shown in Figure 16-35.

Figure *16-35 RAW12* Data Transfer Sequence

The transmission format of the entire Frame becomes as shown in Figure 16-36.

FS	Packer Header, PH													Packer Footer, PF	
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			
	P1	P2	LSBs	*P3*	P4	LSBs	....	P638	LSBs	*P639*	P640	LSBs			FE

Figure *16-36 RAW12 Frame* Format

The transmission sequence for RAW16 is shown in Figure 16-37.

Line Start	Packet Header	**P1[15:8]**	**P1[7:0]**	P2[15:8]	P2[7:0]	***P3[15:8]***	***P3[7:0]***	P4[15:8]	
Line End		***P637[7:0]***	P638[15:8]	P638[7:0]	***P639[15:8]***	***P639[7:0]***	P640[15:8]	P640[7:0]	Packet Footer

Figure *16-37 RAW16* Data Transfer Sequence

The transmission format of the entire Frame becomes as shown in Figure 16-38.

FS												
	Packer Header, PH	**P1**	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	Packer Footer, PF
		P1	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	
		P1	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	
		P1	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	
		P1	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	
		P1	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	
		P1	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	
		P1	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	
		P1	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	
		P1	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	
		P1	**P1**	P2	P2	***P3***	....	***P639***	***P639***	P640	P640	FE

Figure *16-38 RAW16 Frame* Format

16.4.3.3 MIPI Interface Linear Mode

The linear mode transmission format of the MIPI interface is shown in Figure 16-39. Each figure is transmitted as a short packet Frame Start (FS) as the starting formula and the short packet Frame End (FE) as the ending. The intermediate video content is in lines, and each long packet transmits one complete line of video. The long packet format, as standardized in the MIPI standard, has a 32-bit packet header (PH, Pecket Header) for each line, which contains information such as the Virtual Channel and Data Type of the current line.

Figure *16-39 MIPI* Interface Image Formats

16.4.3.4 MIPI Interface Wide Dynamic Mode

The MIPI Rx supports four Wide Dynamic (WDR) modes for the MIPI interface, namely:

1. Use DT (Data Type) to distinguish between short and long exposure data.
2. Distinguish between short and long exposure data using Identification Code
3. Setting the long and short exposure data delay intervals using registers

The WDR transmission method using DT is shown in Figure 16-40. Different exposure lengths share a set of FS/FE short packets, and the long packets contain DT information in the packet header, which can be utilized to differentiate between short and long exposure data, in which the real data format DT and the two sets of DTs representing the short and long exposure data can be set using the register, and then the MIPI Rx is able to parse the correct WDR timing for transmission to the rear video processing module. The real data format DT and the two sets of DT representing the short and long exposure data can be set by registers, and then the MIPI Rx can parse out the correct wide dynamic timing and transmit it to the rear video processing module.

Figure *16-40 MIPI* Interface Wide Dynamic Data Transfer *(using DT)*

The WDR transmission method using ID is shown in Figure 16-41. Different exposure lengths share a set of FS/FE short packets, and the first four pixels of each long packet of the transmitted data are used to transmit IDs (Identification Code) representing the different exposure lengths, which can be set using registers. The MIPI Rx will use the IDs to separate the different exposure video signals and remove the first four pixels before transmitting them to the video processing module.

Figure *16-41 MIPI* Interface Wide Dynamic Data Transfer *(Using ID)*

The last supported WDR transmission method does not have any DT or ID to indicate whether the transmitted packet is a long or short exposure. Users have to set up a register to indicate the difference in the number of lines of exposure between the long and short exposures, and the MIPI Rx will parse out the corresponding timings for the video processing module, and the actual transmission timings are shown in Figure 16-42.

Figure *16-42 MIPI* Interface Wide Dynamic Data Transfer *(Register* Setting*)*

16.4.3.5 Sub-LVDS Interface Data Format

The sub-LVDS (Low-Voltage Differential Signal) is commonly used in front-end cameras to differentiate between zones of valid video signals and long and short exposures of wide dynamic modes through synchronization codes.
The PHY of the MIPI Rx converts the differential serial data into parallel data, and then the controller of the MIPI Rx decodes the parallel data into pixel data based on different patterns and synchronization codes.
MIPI Rx supports Sub-LVDS transmission modes of 8bit, 10bit and 12bit bit widths. The interface data format is shown in Figure 16-43, all valid video signals will be in the middle of the SAV and EAV sync codes, where the sync codes are composed of four fields, and the bit width of each field is the same as that of the pixel at the back. The first three fields are the fixed base code words, and the fourth field can be used to distinguish the beginning or end of the valid interval, the format of the Sub-LVDS sync code is shown in Figure 16-44, and the sync code uses different values according to different vendors, Figure 16-44 is just one of the implementations, different values can be set in the The different values can be set in the registers.

Figure *16-43 Sub-LVDS* Interface Data Format

12-bit					
		1st word	2nd word	3rd word	4th word
Blanking line	Start sync code (SAV)	FFFh	000h	000h	AB0h
	End sync code (EAV)				B60h
Effective line	Start sync code (SAV)				800h
	End sync code (EAV)				9D0h
10-bit					
		1st word	2nd word	3rd word	4th word
Blanking line	Start sync code (SAV)	3FFh	000h	000h	2ACh
	End sync code (EAV)				2D8h
Effective line	Start sync code (SAV)				200h
	End sync code (EAV)				274h
8-bit					
		1st word	2nd word	3rd word	4th word
Blanking line	Start sync code (SAV)	FFh	00h	00h	ABh
	End sync code (EAV)				B6h
Effective line	Start sync code (SAV)				80h
	End sync code (EAV)				9Dh

Figure *16-44* Sub-LVDS Synchronization Code Example

As shown in Figure 16-45, the transmission pattern of sync codes and pixel information in different lanes of Sub-LVDS is the same for each lane, followed by the pixel data, which is sorted by pixel bit width according to the number of lanes used.

Figure *16-45 Sub-LVDS Multi Lane* Transmission Mode

The sync code and pixel data are serial in Sub-LVDS, while MIPI Rx supports that the large and small ends of the data can be set using the registers, and the timing of outputting a single pixel point in the large-end mode, for example, is shown in Figure 16-46.

Figure *16-46 Sub-LVDS* Single Pixel Timing Chart

16.4.3.6 Sub-LVDS Interface Linear Mode

Sub-LVDS uses sync codes in linear mode to indicate the beginning and end of each line in a piece of image data, and anything outside the sync codes SAV and EAV is not valid video data, as shown in Figure 16-47.

SAV blanking line	V-blanking	EAV blanking line	H-blanking
	V-blanking		H-blanking
	V-blanking		H-blanking
	V-blanking		H-blanking
SAV effective line	Effective Pixel	EAV effective line	H-blanking
	Effective Pixel		H-blanking
	⋮		⋮
	Effective Pixel		H-blanking
	Effective Pixel		H-blanking
SAV blanking line	V-blanking	EAV blanking line	H-blanking
	⋮		⋮
	V-blanking		H-blanking

Figure *16-47 Sub-LVDS* Linear Mode Timing Chart

16.4.3.7 Sub-LVDS Interface Wide Dynamic Mode

The MIPI Rx supports two Sub-LVDS interface wide-motion modes. In the first mode, as shown in Figure 16-48, the long and short exposure video signals are encapsulated with SAV and EAV sync codes, respectively, and the MIPI Rx can use the different sync codes to determine whether the video signal is a long or short exposure. In the second mode, as shown in Figure 16-49, both long and short exposures are wrapped with the same SAV and EAV synchronization codes.

EAV wrapping, the width and blanking length of each line must be set in the registers, and the MIPI Rx must use these register settings and synchronization codes to parse out the timing of the long and short exposures, respectively, before transmitting them to the video processing module.

Figure *16-48 Sub-LVDS* Wide Dynamic Mode I

Figure *16-49 Sub-LVDS* Wide Dynamic Mode II

16.4.3.8 HiSPi Interface Data Format

The High-Speed Serial Pixel (HiSPi) interface is also used in some cameras, and similar to Sub-LVDS, uses sync codes to differentiate between valid video information and long and short exposures in wide-motion modes.4 Four different packetization modes are defined in the HiSPi specification, Packetized-SP, Streaming-SP, Streaming-S, and ActiveStart-SP8 . SP, Streaming-S, Streaming-S, and ActiveStart-SP8 four different transmission modes are defined in the HiSPi specification.

MIPI Rx supports two of the more common transport methods, Packetized-SP and Streaming-SP.

16.4.3.9 HiSPi Interface Linear Mode

The MIPI Rx supports two different HiSPi modes. In Packetized-SP mode, as shown in Figure 16-50, the image sensor uses SOF to indicate the first line of a valid video signal and EOF to indicate the end of the last line of a valid video signal. Other valid video signals use SOL and EOL as the beginning and end of a line.

<table>
<tr><td>SOF</td><td>Effective Pixel</td><td rowspan="5">EOL</td><td>H-blanking</td></tr>
<tr><td rowspan="4">SOL</td><td>Effective Pixel</td><td>H-blanking</td></tr>
<tr><td>⋮</td><td>⋮</td></tr>
<tr><td>Effective Pixel</td><td>H-blanking</td></tr>
<tr><td>Effective Pixel</td><td>EOF</td><td>H-blanking</td></tr>
<tr><td colspan="4">V-blanking</td></tr>
</table>

Figure *16-50 HiSPi Packetized-SP* Mode

In Streaming-SP mode, as shown in Figure 16-51, the image sensor does not transmit EOL or EOF for end of line, so the MIPI Rx controller has to know the number of valid video signals by using the register settings in order to parse out the correct video signals for transmission to the video processing module (ISP). In addition, Streaming-SP mode also supports SAV signals that indicate the number of blank lines. The synchronization codes supported by the two different transmission methods are shown in Figure 16-52.

Figure *16-51 HiSPi Streaming-SP* Mode

synchronization code	Packetized-SP	Streaming-SP
SOF	Required	Required
SOL	Required	Required
EOF	Required	Unsupported
EOL	Required	Unsupported
SAV	Unsupported	Required

Figure *16-52* HiSPi Sync Code Support Mode

16.4.3.10 HiSPi Interface Wide Dynamic Mode

HiSPi interface wide dynamic mode is also divided into two different modes, the first Packetized-SP as shown in Figure 16-53, long exposure and short exposure will use different synchronization codes to distinguish between the SOF_L and EOF_L is a valid long exposure video signal, the same in the SOF_S and EOF_S is a valid short exposure video signal. Short Exposure Video Signal. The last rows of the long exposure and the first rows of the short exposure are not valid pixel areas, but are filled with fixed values.

Figure *16-53 HiSPi Packetized-SP* Wide Dynamic Transmission

The second type of Streaming-SP wide-motion transmission is shown in Figure 16-54, in which the sync code and linear mode are the same for both long and short exposures, so it is necessary to use registers to set the difference in the number of lines of exposure between the long and short exposures in order for the MIPI Rx to be able to parse out the correct wide-motion video signal.

Figure *16-54 HiSPi Streaming-SP* Wide Dynamic Transmission

16.4.4 MIPI Rx Register Overview

Up to two MIPI Rx modules can be used simultaneously in the chip, which is

divided into three main groups of registers, the first part is the register controlling the PHY module with the base address 0x0A0D0000, and the second part is the register controlling the CSI module.

The base addresses are 0x0A0C2400 and 0x0A0C4400, while the third section contains registers controlling the Sub-LVDS and HiSPi modules with base addresses 0x0A0C2200 and 0x0A0C4200.

Figure *16-55 MIPI Rx PHY* Register Overview

Name	Address Offset	Description
REG_00	0x000	PD_CTRL
REG_04	0x004	ANALOG_CTRL
REG_30	0x030	SENSOR_MODE_CTRL
REG_34	0x034	ANALOG_CAL_0
REG_38	0x038	ANALOG_CAL_1
REG_3C	0x03c	ANALOG_CAL_2
REG_40	0x040	ANALOG_CAL_3
REG_44	0x044	ANALOG_CAL_4
REG_48	0x048	ANALOG_CAL_5
REG_80	0x080	CLOCK_INVERCE_CTL
REG_A0	0x0a0	
REG_A4	0x0a4	
REG_A8	0x0a8	
REG_AC	0x0ac	

Offset = 0x0A0D0300

Name	Address Offset	Description
REG_00	0x000	SENSOR_MODE_CTRL
REG_04	0x004	LANE_SWAP_0
REG_08	0x008	LANE_SWAP_1
REG_0C	0x00c	CSI_GLB_CTL_0
REG_20	0x020	SLVDS_CTRL_0
REG_24	0x024	SLVDS_CTRL_1
REG_D0_0	0x100	D0_REG_CTRL_CALIB_0
REG_D0_1	0x104	D0_REG_CTRL_CALIB_1
REG_D0_3	0x10c	D0_CALIB_RESULT_0
REG_D0_4	0x110	D0_CALIB_RESULT_1
REG_D0_5	0x114	D0_CALIB_RESULT_2
REG_D0_6	0x118	D0_CALIB_RESULT_3
REG_D0_7	0x11c	D0_CALIB_RESULT_4
REG_D0_8	0x120	D0_CALIB_RESULT_5
REG_D0_9	0x124	D0_CALIB_RESULT_6
REG_D0_A	0x128	D0_CALIB_RESULT_7
REG_D1_0	0x140	D1_REG_CTRL_CALIB_0
REG_D1_1	0x144	D1_REG_CTRL_CALIB_1
REG_D1_3	0x14c	D1_CALIB_RESULT_0
REG_D1_4	0x150	D1_CALIB_RESULT_1
REG_D1_5	0x154	D1_CALIB_RESULT_2
REG_D1_6	0x158	D1_CALIB_RESULT_3
REG_D1_7	0x15c	D1_CALIB_RESULT_4
REG_D1_8	0x160	D1_CALIB_RESULT_5
REG_D1_9	0x164	D1_CALIB_RESULT_6
REG_D1_A	0x168	D1_CALIB_RESULT_7
REG_D2_0	0x180	D2_REG_CTRL_CALIB_0
REG_D2_1	0x184	D2_REG_CTRL_CALIB_1

Name	Address Offset	Description
REG_D2_3	0x18c	D2_CALIB_RESULT_0
REG_D2_4	0x190	D2_CALIB_RESULT_1
REG_D2_5	0x194	D2_CALIB_RESULT_2
REG_D2_6	0x198	D2_CALIB_RESULT_3
REG_D2_7	0x19c	D2_CALIB_RESULT_4
REG_D2_8	0x1a0	D2_CALIB_RESULT_5
REG_D2_9	0x1a4	D2_CALIB_RESULT_6
REG_D2_A	0x1a8	D2_CALIB_RESULT_7
REG_D3_0	0x1c0	D3_REG_CTRL_CALIB_0
REG_D3_1	0x1c4	D3_REG_CTRL_CALIB_1
REG_D3_3	0x1cc	D3_CALIB_RESULT_0
REG_D3_4	0x1d0	D3_CALIB_RESULT_1
REG_D3_5	0x1d4	D3_CALIB_RESULT_2
REG_D3_6	0x1d8	D3_CALIB_RESULT_3
REG_D3_7	0x1dc	D3_CALIB_RESULT_4
REG_D3_8	0x1e0	D3_CALIB_RESULT_5
REG_D3_9	0x1e4	D3_CALIB_RESULT_6
REG_D3_A	0x1e8	D3_CALIB_RESULT_7

Offset = 0x0A0D0600

Name	Address Offset	Description
REG_00	0x000	SENSOR_MODE_CTRL
REG_04	0x004	LANE_SWAP_0
REG_08	0x008	LANE_SWAP_1
REG_0C	0x00c	CSI_GLB_CTL_0
REG_20	0x020	SLVDS_CTRL_0
REG_D0_0	0x100	D0_REG_CTRL_CALIB_0
REG_D0_1	0x104	D0_REG_CTRL_CALIB_1
REG_D0_3	0x10c	D0_CALIB_RESULT_0
REG_D0_4	0x110	D0_CALIB_RESULT_1
REG_D0_5	0x114	D0_CALIB_RESULT_2
REG_D0_6	0x118	D0_CALIB_RESULT_3
REG_D0_7	0x11c	D0_CALIB_RESULT_4
REG_D0_8	0x120	D0_CALIB_RESULT_5
REG_D0_9	0x124	D0_CALIB_RESULT_6
REG_D0_A	0x128	D0_CALIB_RESULT_7
REG_D1_0	0x140	D1_REG_CTRL_CALIB_0
REG_D1_1	0x144	D1_REG_CTRL_CALIB_1
REG_D1_3	0x14c	D1_CALIB_RESULT_0
REG_D1_4	0x150	D1_CALIB_RESULT_1
REG_D1_5	0x154	D1_CALIB_RESULT_2
REG_D1_6	0x158	D1_CALIB_RESULT_3
REG_D1_7	0x15c	D1_CALIB_RESULT_4
REG_D1_8	0x160	D1_CALIB_RESULT_5
REG_D1_9	0x164	D1_CALIB_RESULT_6
REG_D1_A	0x168	D1_CALIB_RESULT_7

Figure *16-56 MIPI Rx CSI* Controller Register Overview

Name	Address Offset	Description
REG_00	0x000	MODE_CTRL

Name	Address Offset	Description
REG_04	0x004	INTR_CTRL
REG_08	0x008	HDR_CTRL_0
REG_0C	0x00c	HDR_CTRL_1
REG_10	0x010	HDR_CTRL_2
REG_14	0x014	BLC_CTRL
REG_18	0x018	HDR_CTRL_3
REG_1C	0x01c	HDR_CTRL_4
REG_20	0x020	HDR_CTRL_5
REG_24	0x024	HDR_CTRL_6
REG_40	0x040	CSI_STATUS
REG_60	0x060	
REG_70	0x070	CSI_VS_GEN
REG_74	0x074	HDR_DT_CTRL

Figure *16-57 MIPI Rx Sub-LVDS* Controller Register Overview

Name	Address Offset	Description
REG_00	0x000	MODE_CTRL
REG_04	0x004	SYNC_CODE_0
REG_08	0x008	SYNC_CODE_1
REG_0C	0x00c	SYNC_CODE_2
REG_10	0x010	SYNC_CODE_3
REG_14	0x014	SYNC_CODE_4
REG_18	0x018	SYNC_CODE_5
REG_1C	0x01c	SYNC_CODE_6
REG_20	0x020	SYNC_CODE_7
REG_24	0x024	SYNC_CODE_8
REG_28	0x028	SYNC_CODE_9
REG_2C	0x02c	VS_GEN
REG_30	0x030	LANE_MODE
REG_50	0x050	SYNC_CODE_A
REG_54	0x054	SYNC_CODE_B
REG_58	0x058	HDR_PATTEN_2
REG_60	0x060	HISPI_MODE_CTRL_0
REG_64	0x064	HISPI_MODE_CTRL_1
REG_68	0x068	HISPI_MODE_CTRL_2
REG_6C	0x06c	HISPI_MODE_CTRL_3
REG_70	0x070	HISPI_MODE_CTRL_4
REG_74	0x074	HISPI_MODE_CTRL_5
REG_80	0x080	DBG_SEL

16.4.5 MIPI Rx Register

Description Part I MIPI Rx PHY

Register Description **REG_00**

Offset Address: 0x000

Bits	Name	Access	Description	Reset
13:0	Reserved			

Bits	Name	Access	Description	Reset
14	reg_mipirx_pd_ibias	R/W	Power down analog ibias	0x1
15	Reserved			
21:16	reg_mipirx_pd_rxlp	R/W	Power down analog RXLP	0x3f
31:22	Reserved			

REG_04

Offset Address: 0x004

Bits	Name	Access	Description	Reset
15:0	Reserved			
21:16	reg_mipirx_sel_clk_channel	R/W	Analog macro clock lane select	0x0
30:22	Reserved			
31	reg_mipimpll_clk_csi_en	R/W	Gating test clock from mipimpll	0x0

REG_30

Offset Address: 0x030

Bits	Name	Access	Description	Reset
2:0	reg_sensor_phy_mode	R/W	Sensor PHY mode enable select 0: 1C4D 1: 1C2D + 1C2D else: reserved	0x0
31:3	Reserved			

REG_34

Offset Address: 0x034

Bits	Name	Access	Description	Reset
31:0	reg_mipirx_ro_cal0	RO	Analog lane 0 calibration result	

REG_38

Offset Address: 0x038

Bits	Name	Access	Description	Reset
31:0	reg_mipirx_ro_cal1	RO	Analog lane 1 calibration result	

REG_3C

Offset Address: 0x03c

Bits	Name	Access	Description	Reset
31:0	reg_mipirx_ro_cal2	RO	Analog lane 2 calibration result	

REG_40

Offset Address: 0x040

Bits	Name	Access	Description	Reset
31:0	reg_mipirx_ro_cal3	RO	Analog lane 3 calibration result	

REG_44

Offset Address: 0x044

Bits	Name	Access	Description	Reset
31:0	reg_mipirx_ro_cal4	RO	Analog lane 4 calibration result	

REG_48

Offset Address: 0x048

Bits	Name	Access	Description	Reset
31:0	reg_mipirx_ro_cal5	RO	Analog lane 5 calibration result	

REG_80

Offset Address: 0x080

Bits	Name	Access	Description	Reset
0	reg_ad_d0_clk_inv	R/W	AD clock lane0 inverse	0x0
1	reg_ad_d1_clk_inv	R/W	AD clock lane1 inverse	0x0
2	reg_ad_d2_clk_inv	R/W	AD clock lane2 inverse	0x0
3	reg_ad_d3_clk_inv	R/W	AD clock lane3 inverse	0x0
4	reg_ad_d4_clk_inv	R/W	AD clock lane4 inverse	0x0
5	reg_ad_d5_clk_inv	R/W	AD clock lane5 inverse	0x0
31:6	Reserved			

REG_A0

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
13:0	reg_cam0_vtt	R/W		0x0
15:14	Reserved			
29:16	reg_cam0_vs_str	R/W		0x0
31:30	Reserved			

REG_A4

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
13:0	reg_cam0_vs_stp	R/W		0x0
15:14	Reserved			
29:16	reg_cam0_htt	R/W		0x0
31:30	Reserved			

REG_A8

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
13:0	reg_cam0_hs_str	R/W		0x0
15:14	Reserved			
29:16	reg_cam0_hs_stp	R/W		0x0
31:30	Reserved			

REG_AC

Offset Address: 0x0ac

Bits	Name	Access	Description	Reset
0	reg_cam0_vs_pol	R/W		0x0
1	reg_cam0_hs_pol	R/W		0x0
2	reg_cam0_tgen_en	R/W		0x0
31:3	Reserved			

Offset = 0x0A0D0300

REG_00

Offset Address: 0x000

Bits	Name	Access	Description	Reset
1:0	reg_sensor_mode	R/W	Sensor mode select 2'b00: CSI 2'b01: Sub-LVDS & HiSPi 2'b10: SLVSEC	0x0
31:2	Reserved			

REG_04

Offset Address: 0x004

Bits	Name	Access	Description	Reset
2:0	reg_csi_lane_d0_sel	R/W	Data lane 0 select	0x1
3	Reserved			
6:4	reg_csi_lane_d1_sel	R/W	Data lane 1 select	0x2
7	Reserved			
10:8	reg_csi_lane_d2_sel	R/W	Data lane 2 select	0x3
11	Reserved			
14:12	reg_csi_lane_d3_sel	R/W	Data lane 3 select	0x4
31:15	Reserved			

REG_08

Offset Address: 0x008

Bits	Name	Access	Description	Reset
2:0	reg_csi_lane_ck_sel	R/W	Clock lane select	0x0
3	Reserved			
4	reg_csi_lane_ck_pnswap	R/W	Clock lane pn swap	0x0
7:5	Reserved			
8	reg_csi_lane_d0_pnswap	R/W	Data lane 0 pn swap	0x0
9	reg_csi_lane_d1_pnswap	R/W	Data lane 1 pn swap	0x0
10	reg_csi_lane_d2_pnswap	R/W	Data lane 2 pn swap	0x0
11	reg_csi_lane_d3_pnswap	R/W	Data lane 3 pn swap	0x0
15:12	Reserved			
23:16	reg_csi_ck_phase	R/W	Clock lane phase	0x0
31:24	Reserved			

REG_0C

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
3:0	reg_deskew_lane_en	R/W	Deskew lane enable 4'h0: No lane 4'h1: 1-lane 4'h3: 2-lane 8'hf: 4-lane	0x0
31:4	Reserved			

REG_20

Offset Address: 0x020

Bits	Name	Access	Description	Reset
0	reg_slvds_inv_en	R/W	Sub-LVDS bit reverse 1'b0: LSB first 1'b1: MSB first	0x1
1	Reserved			

3:2	reg_slvds_bit_mode	R/W	Sub-LVDS bit mode	0x2

Bits	Name	Access	Description	Reset
			2'b00: 8-bit 2'b01: 10-bit 2'b10: 12-bit	
7:4	reg_slvds_lane_en	R/W	Sub-LVDS lane enable Set this register to start finding sync code	0x0
15:8	Reserved			
27:16	reg_slvds_sav_1st	R/W	Sub-LVDS sync code 1st symbol	0xfff
31:28	Reserved			

REG_24

Offset Address: 0x024

Bits	Name	Access	Description	Reset
11:0	reg_slvds_sav_2nd	R/W	Sub-LVDS sync code 2nd symbol	0x0
15:12	Reserved			
27:16	reg_slvds_sav_3rd	R/W	Sub-LVDS sync code 3rd symbol	0x0
31:28	Reserved			

REG_D0_0

Offset Address: 0x100

Bits	Name	Access	Description	Reset
0	reg_d0_prbs9_en	R/W	Manual PRBS9 enable	0x0
1	reg_d0_prbs9_clr_err	R/W	PRBS9 clear error	0x0
2	reg_d0_prbs9_source	R/W	PRBS9 source select 1'b0: after sync code shift 1'b1: direct from input	0x0
3	reg_d0_prbs9_stop_when_done	R/W	PRBS9 error count accumulation 1'b0: still count after test time done 1'b1: do not count after test time done	0x0
7:4	Reserved			
15:8	reg_d0_calib_max	R/W	Calibration max step	0x1f
23:16	reg_d0_calib_step	R/W	Calibration one step value	0x1
31:24	reg_d0_calib_pattern	R/W	Calibration golden pattern	0xaa

REG_D0_1

Offset Address: 0x104

Bits	Name	Access	Description	Reset
0	reg_d0_calib_en	R/W	Calibration software enable	0x0
1	reg_d0_calib_source	R/W	Calibration source 1'b0: normal position 1'b1: direct from analog	0x0
2	reg_d0_calib_mode	R/W	Calibration software mode 1'b0: use identical calibration pattern 1'b1: use PRBS9 pattern	0x0
3	reg_d0_calib_ignore	R/W	Ignore calibration command	0x0
31:4	Reserved			

REG_D0_3

Offset Address: 0x10c

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_0	RO	Calibration result phase 0~31	

REG_D0_4

Offset Address: 0x110

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_1	RO	Calibration result phase 32~63	

REG_D0_5

Offset Address: 0x114

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_2	RO	Calibration result phase 64~95	

REG_D0_6

Offset Address: 0x118

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_3	RO	Calibration result phase 96~127	

REG_D0_7

Offset Address: 0x11c

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_4	RO	Calibration result phase 128~159	

REG_D0_8

Offset Address: 0x120

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_5	RO	Calibration result phase 160~191	

REG_D0_9

Offset Address: 0x124

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_6	RO	Calibration result phase 192~223	

REG_D0_A

Offset Address: 0x128

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_7	RO	Calibration result phase 224~255	

REG_D1_0

Offset Address: 0x140

Bits	Name	Access	Description	Reset
0	reg_d1_prbs9_en	R/W	Manual PRBS9 enable	0x0
1	reg_d1_prbs9_clr_err	R/W	PRBS9 clear error	0x0
2	reg_d1_prbs9_source	R/W	PRBS9 source select 1'b0: after sync code shift 1'b1: direct from input	0x0
3	reg_d1_prbs9_stop_when_done	R/W	PRBS9 error count accumulation 1'b0: still count after test time done 1'b1: do not count after test time done	0x0
7:4	Reserved			
15:8	reg_d1_calib_max	R/W	Calibration max step	0x1f
23:16	reg_d1_calib_step	R/W	Calibration one step value	0x1
31:24	reg_d1_calib_pattern	R/W	Calibration golden pattern	0xaa

REG_D1_1

Offset Address: 0x144

Bits	Name	Access	Description	Reset
0	reg_d1_calib_en	R/W	Calibration software enable	0x0
1	reg_d1_calib_source	R/W	Calibration source 1'b0: normal position 1'b1: direct from analog	0x0
2	reg_d1_calib_mode	R/W	Calibration software mode 1'b0: use identical calibration pattern 1'b1: use PRBS9 pattern	0x0
3	reg_d1_calib_ignore	R/W	Ignore calibration command	0x0
31:4	Reserved			

REG_D1_3

Offset Address: 0x14c

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_0	RO	Calibration result phase 0~31	

REG_D1_4

Offset Address: 0x150

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_1	RO	Calibration result phase 32~63	

REG_D1_5

Offset Address: 0x154

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_2	RO	Calibration result phase 64~95	

REG_D1_6

Offset Address: 0x158

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_3	RO	Calibration result phase 96~127	

REG_D1_7

Offset Address: 0x15c

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_4	RO	Calibration result phase 128~159	

REG_D1_8

Offset Address: 0x160

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_5	RO	Calibration result phase 160~191	

REG_D1_9

Offset Address: 0x164

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_6	RO	Calibration result phase 192~223	

REG_D1_A

Offset Address: 0x168

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_7	RO	Calibration result phase 224~255	

REG_D2_0

Offset Address: 0x180

Bits	Name	Access	Description	Reset
0	reg_d2_prbs9_en	R/W	Manual PRBS9 enable	0x0
1	reg_d2_prbs9_clr_err	R/W	PRBS9 clear error	0x0
2	reg_d2_prbs9_source	R/W	PRBS9 source select 1'b0: after sync code shift 1'b1: direct from input	0x0
3	reg_d2_prbs9_stop_when_done	R/W	PRBS9 error count accumulation 1'b0: still count after test time done 1'b1: do not count after test time done	0x0
7:4	Reserved			
15:8	reg_d2_calib_max	R/W	Calibration max step	0x1f
23:16	reg_d2_calib_step	R/W	Calibration one step value	0x1
31:24	reg_d2_calib_pattern	R/W	Calibration golden pattern	0xaa

REG_D2_1

Offset Address: 0x184

Bits	Name	Access	Description	Reset
0	reg_d2_calib_en	R/W	Calibration software enable	0x0
1	reg_d2_calib_source	R/W	Calibration source 1'b0: normal position 1'b1: direct from analog	0x0
2	reg_d2_calib_mode	R/W	Calibration software mode 1'b0: use identical calibration pattern 1'b1: use PRBS9 pattern	0x0
3	reg_d2_calib_ignore	R/W	Ignore calibration command	0x0
31:4	Reserved			

REG_D2_3

Offset Address: 0x18c

Bits	Name	Access	Description	Reset
31:0	reg_d2_skew_calib_result_0	RO	Calibration result phase 0~31	

REG_D2_4

Offset Address: 0x190

Bits	Name	Access	Description	Reset
31:0	reg_d2_skew_calib_result_1	RO	Calibration result phase 32~63	

REG_D2_5

Offset Address: 0x194

Bits	Name	Access	Description	Reset
31:0	reg_d2_skew_calib_result_2	RO	Calibration result phase 64~95	

REG_D2_6

Offset Address: 0x198

Bits	Name	Access	Description	Reset
31:0	reg_d2_skew_calib_result_3	RO	Calibration result phase 96~127	

REG_D2_7

Offset Address: 0x19c

Bits	Name	Access	Description	Reset
31:0	reg_d2_skew_calib_result_4	RO	Calibration result phase 128~159	

REG_D2_8

Offset Address: 0x1a0

Bits	Name	Access	Description	Reset
31:0	reg_d2_skew_calib_result_5	RO	Calibration result phase 160~191	

REG_D2_9

Offset Address: 0x1a4

Bits	Name	Access	Description	Reset
31:0	reg_d2_skew_calib_result_6	RO	Calibration result phase 192~223	

REG_D2_A

Offset Address: 0x1a8

Bits	Name	Access	Description	Reset
31:0	reg_d2_skew_calib_result_7	RO	Calibration result phase 224~255	

REG_D3_0

Offset Address: 0x1c0

Bits	Name	Access	Description	Reset
0	reg_d3_prbs9_en	R/W	Manual PRBS9 enable	0x0
1	reg_d3_prbs9_clr_err	R/W	PRBS9 clear error	0x0
2	reg_d3_prbs9_source	R/W	PRBS9 source select 1'b0: after sync code shift 1'b1: direct from input	0x0
3	reg_d3_prbs9_stop_when_done	R/W	PRBS9 error count accumulation 1'b0: still count after test time done 1'b1: do not count after test time done	0x0
7:4	Reserved			
15:8	reg_d3_calib_max	R/W	Calibration max step	0x1f
23:16	reg_d3_calib_step	R/W	Calibration one step value	0x1
31:24	reg_d3_calib_pattern	R/W	Calibration golden pattern	0xaa

REG_D3_1

Offset Address: 0x1c4

Bits	Name	Access	Description	Reset
0	reg_d3_calib_en	R/W	Calibration software enable	0x0
1	reg_d3_calib_source	R/W	Calibration source 1'b0: normal position 1'b1: direct from analog	0x0
2	reg_d3_calib_mode	R/W	Calibration software mode 1'b0: use identical calibration pattern 1'b1: use PRBS9 pattern	0x0
3	reg_d3_calib_ignore	R/W	Ignore calibration command	0x0
31:4	Reserved			

REG_D3_3

Offset Address: 0x1cc

Bits	Name	Access	Description	Reset
31:0	reg_d3_skew_calib_result_0	RO	Calibration result phase 0~31	

REG_D3_4

Offset Address: 0x1d0

Bits	Name	Access	Description	Reset
31:0	reg_d3_skew_calib_result_1	RO	Calibration result phase 32~63	

REG_D3_5

Offset Address: 0x1d4

Bits	Name	Access	Description	Reset
31:0	reg_d3_skew_calib_result_2	RO	Calibration result phase 64~95	

REG_D3_6

Offset Address: 0x1d8

Bits	Name	Access	Description	Reset
31:0	reg_d3_skew_calib_result_3	RO	Calibration result phase 96~127	

REG_D3_7

Offset Address: 0x1dc

Bits	Name	Access	Description	Reset
31:0	reg_d3_skew_calib_result_4	RO	Calibration result phase 128~159	

REG_D3_8

Offset Address: 0x1e0

Bits	Name	Access	Description	Reset
31:0	reg_d3_skew_calib_result_5	RO	Calibration result phase 160~191	

REG_D3_9

Offset Address: 0x1e4

Bits	Name	Access	Description	Reset
31:0	reg_d3_skew_calib_result_6	RO	Calibration result phase 192~223	

REG_D3_A

Offset Address: 0x1e8

Bits	Name	Access	Description	Reset
31:0	reg_d3_skew_calib_result_7	RO	Calibration result phase 224~255	

Offset = 0x0A0D0600

REG_00

Offset Address: 0x000

Bits	Name	Access	Description	Reset
1:0	reg_sensor_mode	R/W	Sensor mode select 2'b00: CSI 2'b01: Sub-LVDS & HiSPi 2'b10: SLVSEC	0x0
31:2	Reserved			

REG_04

Offset Address: 0x004

Bits	Name	Access	Description	Reset
1:0	reg_csi_lane_d0_sel	R/W	Data lane 0 select	0x1
3:2	Reserved			
5:4	reg_csi_lane_d1_sel	R/W	Data lane 1 select	0x2
31:6	Reserved			

REG_08

Offset Address: 0x008

Bits	Name	Access	Description	Reset
1:0	reg_csi_lane_ck_sel	R/W	Clock lane select	0x0
3:2	Reserved			
4	reg_csi_lane_ck_pnswap	R/W	Clock lane pn swap	0x0
7:5	Reserved			
8	reg_csi_lane_d0_pnswap	R/W	Data lane 0 pn swap	0x0
9	reg_csi_lane_d1_pnswap	R/W	Data lane 1 pn swap	0x0
15:10	Reserved			
23:16	reg_csi_ck_phase	R/W	Clock lane phase	0x0
31:24	Reserved			

REG_0C

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
1:0	reg_deskew_lane_en	R/W	Deskew lane enable 2'h0: No lane 2'h1: 1-lane 2'h3: 2-lane	0x0
31:2	Reserved			

REG_20

Offset Address: 0x020

Bits	Name	Access	Description	Reset
0	reg_slvds_inv_en	R/W	Sub-LVDS bit reverse 1'b0: LSB first 1'b1: MSB first	0x1
1	Reserved			
3:2	reg_slvds_bit_mode	R/W	Sub-LVDS bit mode 2'b00: 8-bit 2'b01: 10-bit 2'b10: 12-bit	0x2
5:4	reg_slvds_lane_en	R/W	Sub-LVDS lane enable Set this register to start finding sync code	0x0
31:6	Reserved			

REG_D0_0

Offset Address: 0x100

Bits	Name	Access	Description	Reset
0	reg_d0_prbs9_en	R/W	Manual PRBS9 enable	0x0
1	reg_d0_prbs9_clr_err	R/W	PRBS9 clear error	0x0
2	reg_d0_prbs9_source	R/W	PRBS9 source select 1'b0: after sync code shift 1'b1: direct from input	0x0
3	reg_d0_prbs9_stop_when_done	R/W	PRBS9 error count accumulation 1'b0: still count after test time done 1'b1: do not count after test time done	0x0
7:4	Reserved			
15:8	reg_d0_calib_max	R/W	Calibration max step	0x1f
23:16	reg_d0_calib_step	R/W	Calibration one step value	0x1
31:24	reg_d0_calib_pattern	R/W	Calibration golden pattern	0xaa

REG_D0_1

Offset Address: 0x104

Bits	Name	Access	Description	Reset
0	reg_d0_calib_en	R/W	Calibration software enable	0x0
1	reg_d0_calib_source	R/W	Calibration source 1'b0: normal position 1'b1: direct from analog	0x0
2	reg_d0_calib_mode	R/W	Calibration software mode 1'b0: use identical calibration pattern 1'b1: use PRBS9 pattern	0x0
3	reg_d0_calib_ignore	R/W	Ignore calibration command	0x0
31:4	Reserved			

REG_D0_3

Offset Address: 0x10c

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_0	RO	Calibration result phase 0~31	

REG_D0_4

Offset Address: 0x110

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_1	RO	Calibration result phase 32~63	

REG_D0_5

Offset Address: 0x114

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_2	RO	Calibration result phase 64~95	

REG_D0_6

Offset Address: 0x118

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_3	RO	Calibration result phase 96~127	

REG_D0_7

Offset Address: 0x11c

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_4	RO	Calibration result phase 128~159	

REG_D0_8

Offset Address: 0x120

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_5	RO	Calibration result phase 160~191	

REG_D0_9

Offset Address: 0x124

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_6	RO	Calibration result phase 192~223	

REG_D0_A

Offset Address: 0x128

Bits	Name	Access	Description	Reset
31:0	reg_d0_skew_calib_result_7	RO	Calibration result phase 224~255	

REG_D1_0

Offset Address: 0x140

Bits	Name	Access	Description	Reset
0	reg_d1_prbs9_en	R/W	Manual PRBS9 enable	0x0
1	reg_d1_prbs9_clr_err	R/W	PRBS9 clear error	0x0
2	reg_d1_prbs9_source	R/W	PRBS9 source select 1'b0: after sync code shift 1'b1: direct from input	0x0
3	reg_d1_prbs9_stop_when_done	R/W	PRBS9 error count accumulation 1'b0: still count after test time done 1'b1: do not count after test time done	0x0
7:4	Reserved			
15:8	reg_d1_calib_max	R/W	Calibration max step	0x1f
23:16	reg_d1_calib_step	R/W	Calibration one step value	0x1
31:24	reg_d1_calib_pattern	R/W	Calibration golden pattern	0xaa

REG_D1_1

Offset Address: 0x144

Bits	Name	Access	Description	Reset
0	reg_d1_calib_en	R/W	Calibration software enable	0x0
1	reg_d1_calib_source	R/W	Calibration source 1'b0: normal position 1'b1: direct from analog	0x0
2	reg_d1_calib_mode	R/W	Calibration software mode 1'b0: use identical calibration pattern 1'b1: use PRBS9 pattern	0x0
3	reg_d1_calib_ignore	R/W	Ignore calibration command	0x0
31:4	Reserved			

REG_D1_3

Offset Address: 0x14c

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_0	RO	Calibration result phase 0~31	

REG_D1_4

Offset Address: 0x150

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_1	RO	Calibration result phase 32~63	

REG_D1_5

Offset Address: 0x154

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_2	RO	Calibration result phase 64~95	

REG_D1_6

Offset Address: 0x158

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_3	RO	Calibration result phase 96~127	

REG_D1_7

Offset Address: 0x15c

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_4	RO	Calibration result phase 128~159	

REG_D1_8

Offset Address: 0x160

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_5	RO	Calibration result phase 160~191	

REG_D1_9

Offset Address: 0x164

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_6	RO	Calibration result phase 192~223	

REG_D1_A

Offset Address: 0x168

Bits	Name	Access	Description	Reset
31:0	reg_d1_skew_calib_result_7	RO	Calibration result phase 224~255	

The second part is the MIPI Rx CSI controller registers.

REG_00

Offset Address: 0x000

Bits	Name	Access	Description	Reset
2:0	reg_csi_lane_mode	R/W	Lane mode 3'b000: 1-lane 3'b001: 2-lane 3'b011: 4-lane 3'b111: 8-lane	0x0
3	reg_csi_ignore_ecc	R/W	Ignore ecc result 1'b0: normal 1'b1: still processing even ecc error	0x0
4	reg_csi_vc_check	R/W	VC check enable 1'b0: do not check VC 1'b1: only process packets that meet vc_set[3:0]	0x0
7:5	Reserved			
11:8	reg_csi_vc_set	R/W	VC set only use when reg_csi_vc_check assert	0x0
12	reg_csi_line_start_sent	R/W	LS and LE packet sent 1'b0: create hsync signal by controller 1'b1: use LS and LE to create hsync signal	0x0
31:13	Reserved			

REG_04

Offset Address: 0x004

Bits	Name	Access	Description	Reset
7:0	reg_csi_intr_mask	R/W	Interrupt mask control	0x0
15:8	reg_csi_intr_clr	W1T	Interrupt clear	
16	reg_csi_hdr_en	R/W	HDR mode enable	0x0
17	reg_csi_hdr_mode	R/W	HDR mode selection 1'b0: HDR VC mode 1'b1: HDR ID mode	0x0

18	reg_csi_id_rm_else	R/W	Remove non reconized ID line 1'b0: dont remove 1'b1: remove	0x1

Bits	Name	Access	Description	Reset
19	reg_csi_id_rm_ob	R/W	Remove ob line 1'b0: don't remove 1'b1: remove	0x1
31:20	Reserved			

REG_08

Offset Address: 0x008

Bits	Name	Access	Description	Reset
15:0	reg_csi_n0_ob_lef	R/W	ID for LEF ob n0	0x221
31:16	reg_csi_n0_ob_sef	R/W	ID for SEF ob n0	0x222

REG_0C

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
15:0	reg_csi_n0_lef	R/W	ID for LEF active n0	0x241
31:16	reg_csi_n1_ob_lef	R/W	ID for LEF ob n1	0x231

REG_10

Offset Address: 0x010

Bits	Name	Access	Description	Reset
15:0	reg_csi_n1_ob_sef	R/W	ID for SEF ob n1	0x232
31:16	reg_csi_n1_lef	R/W	ID for LEF active n1	0x251

REG_14

Offset Address: 0x014

Bits	Name	Access	Description	Reset
5:0	reg_csi_blc_dt	R/W	Data type for optical black line	0x37
7:6	Reserved			
8	reg_csi_blc_en	R/W	Optical black line mode enable	0x0
11:9	Reserved			
14:12	reg_csi_blc_format_set	R/W	Optical black line data format set 3'd0: YUV422 8bit 3'd1: YUV422 10bit 3'd2: RAW8 3'd3: RAW10 3'd4: RAW12 3'd5: RAW16 else: reserved	0x2
31:15	Reserved			

REG_18

Offset Address: 0x018

Bits	Name	Access	Description	Reset
3:0	reg_csi_vc_map_ch00	R/W	VC mapping to ISP channel 00	0x0
7:4	reg_csi_vc_map_ch01	R/W	VC mapping to ISP channel 01	0x1
11:8	reg_csi_vc_map_ch10	R/W	VC mapping to ISP channel 10	0x2
15:12	reg_csi_vc_map_ch11	R/W	VC mapping to ISP channel 11	0x3
31:16	Reserved			

REG_1C

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
15:0	reg_csi_n0_sef	R/W	ID for SEF active n0	0x242

Bits	Name	Access	Description	Reset
31:16	reg_csi_n1_sef	R/W	ID for SEF active n1	0x252

REG_20

Offset Address: 0x020

Bits	Name	Access	Description	Reset
15:0	reg_csi_n0_sef2	R/W	ID for SEF2 active n0	0x244
31:16	reg_csi_n1_sef2	R/W	ID for SEF2 active n1	0x254

REG_24

Offset Address: 0x024

Bits	Name	Access	Description	Reset
15:0	reg_csi_n0_ob_sef2	R/W	ID for SEF2 ob n0	0x224
31:16	reg_csi_n1_ob_sef2	R/W	ID for SEF2 ob n1	0x234

REG_40

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	reg_csi_ecc_no_error	RO	ECC no error	
1	reg_csi_ecc_corrected_error	RO	ECC corrected error	
2	reg_csi_ecc_error	RO	ECC error	
3	Reserved			
4	reg_csi_crc_error	RO	CRC error	
5	reg_csi_wc_error	RO	WC error	
7:6	Reserved			
8	reg_csi_fifo_full	RO	CSI FIFO full	
15:9	Reserved			
21:16	reg_csi_decode_format	RO	CSI decode format from header bit[0]: YUV422 8bit bit[1]: YUV422 10bit bit[2]: RAW8 bit[3]: RAW10 bit[4]: RAW12 bit[5]: RAW16	
31:22	Reserved			

REG_60

Offset Address: 0x060

Bits	Name	Access	Description	Reset
7:0	reg_csi_intr_status	RO	Interrupt status bit[0]: ecc error bit[1]: crc error bit[2]: hdr id error bit[3]: word count error bit[4]: fifo full	
31:8	Reserved			

REG_70

Offset Address: 0x070

Bits	Name	Access	Description	Reset
1:0	reg_csi_vs_gen_mode	R/W	2'b00: vs gen by FS 2'b01: vs gen by FE else: vs gen by FS & FE	0x2

Bits	Name	Access	Description	Reset
3:2	Reserved			
4	reg_csi_vs_gen_by_vcset	R/W	Vsync generation setting 1'b0: generated by all vc short packet 1'b1: only generated by indicated vc short packet	0x0
31:5	Reserved			

REG_74

Offset Address: 0x074

Bits	Name	Access	Description	Reset
0	reg_csi_hdr_dt_mode	R/W	CSI HDR DT mode enable	0x0
3:1	Reserved			
9:4	reg_csi_hdr_dt_format	R/W	CSI HDR DT mode video format data type	0x0
11:10	Reserved			
17:12	reg_csi_hdr_dt_lef	R/W	CSI HDR DT mode LEF data type	0x0
19:18	Reserved			
25:20	reg_csi_hdr_dt_sef	R/W	CSI HDR DT mode SEF data type	0x0
31:26	Reserved			

The third part is the MIPI Rx Sub-LVDS control register.

REG_00

Offset Address: 0x000

Bits	Name	Access	Description	Reset
7:0	reg_slvds_enable	R/W	Sub-LVDS lane enable for each lane	0x0
9:8	reg_slvds_bit_mode	R/W	Sub-LVDS bit mode 2'b00: 8-bit 2'b01: 10-bit 2'b10: 12-bit	0x2
10	reg_slvds_data_reverse	R/W	Sub-LVDS data packet bit inverse	0x0
11	Reserved			
12	reg_slvds_hdr_mode	R/W	Sub-LVDS HDR mode enable	0x0
13	reg_slvds_hdr_pattern	R/W	Sub-LVDS HDR pattern mode 1'b0: pattern 1 1'b1: pattern 2	0x0
31:14	Reserved			

REG_04

Offset Address: 0x004

Bits	Name	Access	Description	Reset
11:0	reg_slvds_sync_1st	R/W	Sub-LVDS SYNC code 1st word	0xFFF
15:12	Reserved			
27:16	reg_slvds_sync_2nd	R/W	Sub-LVDS SYNC code 2nd word	0x000
31:28	Reserved			

REG_08

Offset Address: 0x008

Bits	Name	Access	Description	Reset
11:0	reg_slvds_sync_3rd	R/W	Sub-LVDS SYNC code 3rd word	0x000

Bits	Name	Access	Description	Reset
15:12	Reserved			
27:16	reg_slvds_norm_bk_sav	R/W	Normal mode blanking SAV	0xAB0
31:28	Reserved			

REG_0C

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
11:0	reg_slvds_norm_bk_eav	R/W	Normal mode blanking EAV	0xB60
15:12	Reserved			
27:16	reg_slvds_norm_sav	R/W	Normal mode active SAV	0x800
31:28	Reserved			

REG_10

Offset Address: 0x010

Bits	Name	Access	Description	Reset
11:0	reg_slvds_norm_eav	R/W	Normal mode active EAV	0x9D0
15:12	Reserved			
27:16	reg_slvds_n0_bk_sav	R/W	HDR mode n0 blanking SAV	0x2B0
31:28	Reserved			

REG_14

Offset Address: 0x014

Bits	Name	Access	Description	Reset
11:0	reg_slvds_n0_bk_eav	R/W	HDR mode n0 blanking EAV	0x360
15:12	Reserved			
27:16	reg_slvds_n1_bk_sav	R/W	HDR mode n1 blanking SAV	0x6B0
31:28	Reserved			

REG_18

Offset Address: 0x018

Bits	Name	Access	Description	Reset
11:0	reg_slvds_n1_bk_eav	R/W	HDR mode n1 blanking EAV	0x760
15:12	Reserved			
27:16	reg_slvds_n0_lef_sav	R/W	Sub-LVDS mode: n0 long exposure sav Sub-LVDS 12-bit LEF SAV n0 (801) Sub-LVDS 10-bit LEF SAV n0 (004) HiSPi P-SP mode: SOL T1 (800)	0x801
31:28	Reserved			

REG_1C

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
11:0	reg_slvds_n0_lef_eav	R/W	Sub-LVDS mode: n0 long exposure eav Sub-LVDS 12-bit LEF EAV n0 (9D1) Sub-LVDS 10-bit LEF EAV n0 (1D4) HiSPi P-SP mode: EOL T1 (A00)	0x9D1
15:12	Reserved			
27:16	reg_slvds_n0_sef_sav	R/W	Sub-LVDS mode: n0 short exposure sav Sub-LVDS 12-bit SEF SAV n0 (802) Sub-LVDS 10-bit SEF SAV n0 (008) HiSPi P-SP mode: SOL T2 (820)	0x802

Bits	Name	Access	Description	Reset
31:28	Reserved			

REG_20

Offset Address: 0x020

Bits	Name	Access	Description	Reset
11:0	reg_slvds_n0_sef_eav	R/W	Sub-LVDS mode: n0 short exposure eav Sub-LVDS 12-bit SEF EAV n0 (9D2) Sub-LVDS 10-bit SEF EAV n0 (1d8) HiSPi P-SP mode: EOL T2 (A20)	0x9D2
15:12	Reserved			
27:16	reg_slvds_n1_lef_sav	R/W	Sub-LVDS mode: n1 long exposure sav Sub-LVDS 12-bit LEF SAV n1 (C01) Sub-LVDS 10-bit LEF SAV n1 (404) HiSPi P-SP mode: SOF T1 (C00)	0xC01
31:28	Reserved			

REG_24

Offset Address: 0x024

Bits	Name	Access	Description	Reset
11:0	reg_slvds_n1_lef_eav	R/W	Sub-LVDS mode: n1 long exposure eav Sub-LVDS 12-bit LEF EAV n1 (DD1) Sub-LVDS 10-bit LEF EAV n1 (5D4) HiSPi P-SP mode: EOF T1 (E00)	0xDD1
15:12	Reserved			
27:16	reg_slvds_n1_sef_sav	R/W	Sub-LVDS mode: n1 short exposure sav Sub-LVDS 12-bit SEF SAV n1 (C02) Sub-LVDS 10-bit SEF SAV n1 (408) HiSPi P-SP mode: SOF T2 (C20)	0xC02
31:28	Reserved			

REG_28

Offset Address: 0x028

Bits	Name	Access	Description	Reset
11:0	reg_slvds_n1_sef_eav	R/W	Sub-LVDS mode: n1 short exposure eav Sub-LVDS 12-bit SEF EAV n1 (DD2) Sub-LVDS 10-bit SEF EAV n1 (5D8) HiSPi P-SP mode: EOF T2 (E20)	0xDD2
31:12	Reserved			

REG_2C

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
11:0	reg_vs_gen_sync_code	R/W	vs generate sync code value using scenario: HiSPi P-SP HDR	0xC00
12	reg_vs_gen_by_sync_code	R/W	vs generate by identical sync code using scenario: HiSPi P-SP HDR	0x0
31:13	Reserved			

REG_30

Offset Address: 0x030

Bits	Name	Access	Description	Reset
2:0	reg_slvds_lane_mode	R/W	Sub-LVDS lane mode 2'b0: 1-lane	0x3

Bits	Name	Access	Description	Reset
			2'b1: 2-lane 2'b3: 4-lane 2'b7: 8-lane	
3	Reserved			
11:4	reg_slvds_sync_source	R/W	Sub-LVDS output sync source select	0x1
31:12	Reserved			

REG_50

Offset Address: 0x050

Bits	Name	Access	Description	Reset
11:0	reg_slvds_n0_lsef_sav	R/W	SAV for n0 long & short exposure both exist line only used for pattern 2	0x803
15:12	Reserved			
27:16	reg_slvds_n0_lsef_eav	R/W	EAV for n0 long & short exposure both exist line only used for pattern 2	0x9D3
31:28	Reserved			

REG_54

Offset Address: 0x054

Bits	Name	Access	Description	Reset
11:0	reg_slvds_n1_lsef_sav	R/W	SAV for n1 long & short exposure both exist line only used for pattern 2	0xC03
15:12	Reserved			
27:16	reg_slvds_n1_lsef_eav	R/W	EAV for n1 long & short exposure both exist line only used for pattern 2	0xDD3
31:28	Reserved			

REG_58

Offset Address: 0x058

Bits	Name	Access	Description	Reset
13:0	reg_slvds_hdr_p2_hsize	R/W	Hsize for pattern 2	0xF0
15:14	Reserved			
29:16	reg_slvds_hdr_p2_hblank	R/W	Hblank size for pattern 2	0x14
31:30	Reserved			

REG_60

Offset Address: 0x060

Bits	Name	Access	Description	Reset
0	reg_hispi_mode	R/W	HiSPi mode enable 1'b0: Sub-LVDS 1'b1: HiSPi	0x0
1	reg_hispi_use_hsize	R/W	HiSPi DE de-assert by register count	0x0
3:2	Reserved			
4	reg_hispi_hdr_psp_mode	R/W	HiSPi P-SP HDR mode enable	0x0
31:5	Reserved			

REG_64

Offset Address: 0x064

Bits	Name	Access	Description	Reset
11:0	reg_hispi_norm_sof	R/W	HiSPi SOF sync code	0xC00
15:12	Reserved			
27:16	reg_hispi_norm_eof	R/W	HiSPi EOF sync code	0xE00
31:28	Reserved			

REG_68

Offset Address: 0x068

Bits	Name	Access	Description	Reset
11:0	reg_hispi_hdr_t1_sof	R/W	HiSPi HDR T1 SOF	0xC00
15:12	Reserved			
27:16	reg_hispi_hdr_t1_eof	R/W	HiSPi HDR T1 EOF	0xE00
31:28	Reserved			

REG_6C

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
11:0	reg_hispi_hdr_t1_sol	R/W	HiSPi HDR T1 SOL	0x800
15:12	Reserved			
27:16	reg_hispi_hdr_t1_eol	R/W	HiSPi HDR T1 EOL	0xA00
31:28	Reserved			

REG_70

Offset Address: 0x070

Bits	Name	Access	Description	Reset
11:0	reg_hispi_hdr_t2_sof	R/W	HiSPi HDR T2 SOF	0xC20
15:12	Reserved			
27:16	reg_hispi_hdr_t2_eof	R/W	HiSPi HDR T2 EOF	0xE20
31:28	Reserved			

REG_74

Offset Address: 0x074

Bits	Name	Access	Description	Reset
11:0	reg_hispi_hdr_t2_sol	R/W	HiSPi HDR T2 SOL	0x820
15:12	Reserved			
27:16	reg_hispi_hdr_t2_eol	R/W	HiSPi HDR T2 EOL	0xA20
31:28	Reserved			

REG_80

Offset Address: 0x080

Bits	Name	Access	Description	Reset
7:0	reg_dbg_sel	R/W	Debug signal select	0x0
31:8	Reserved			

16.5 MIPI Tx

16.5.1 summarize

The Display Serial Interface (DSI) interface is a high-speed serial interface defined by the Mobile Industry Processor Interface alliance (MIPI alliance) and is primarily used for processor and

The connection between the display modules. the MIPI Tx interface implements the DSI interface and supports MIPI D-PHY V1.0 serial signal output.

The MIPI Tx consists of two parts, the analog PHY and the digital controller, and the system architecture and functional block diagram are shown in Figure.

Figure *16-58 MIPI TX* Function Block Diagram

16.5.2 specificities

The MIPI Tx has the following features:

- Supports 1/2/4 Data Lane MIPI D-PHY interface, sequential, PN poles configurable.
- High-speed mode supports up to 2500Mbps/Lane.
- Only Data Lane0 supports low-speed transmit and receive, BTA (Bus Turn-Around) function, and up to 10Mbps in low-speed mode.
- Supports DSI RGB 16/18/24/30 bit data class shape output.
- Supports DSI video mode and command mode, video mode supports Burst mode, Non-burst mode with Sync Events and Non-burst mode with Sync pulses.

16.5.3 Functional Description

MIPI Tx consists of the Tx D-PHY and the Tx Controller.

- The Tx D-PHY supports the MIPI D-PHY ver2.1 protocol, which mainly implements the transmission specification of the physical layer.
- The Tx Controller encapsulates the data format according to the MIPI DSI protocol.

16.5.3.1 Tx D-PHY

The Tx D-PHY has two modes of operation, High Speed (HS) and Low Power (LP):

- Data in Video mode is transmitted in High Speed mode.
- Command mode data is transmitted through the low-speed mode.

The rate range of each lane in high speed mode is 80~2500Mbps, and the maximum rate of low speed mode is
10Mbps.
High-speed mode supports a maximum of 4 data lanes, the actual number of data lanes used can be 1/2/4, the order and polarity can be configured. Low-speed mode transmit, receive and Bus Turn-Around (BTA) are only supported by the configured data lane0.

16.5.3.2 Tx Controller

16.5.3.3 packet delivery

When there are multiple high-speed packets to be transmitted, the Tx D-PHY will automatically switch between HS and LP modes according to the high-speed data transmission requirements of the Tx Controller, and the Tx Controller supports whether or not to send EoT packets at the end of HS transmission.

16.5.3.4 data type

The controller supports DSI RGB16/18/24/30 bit transmissions, and the composition format of the various data types is shown in the figure.

Figure *16-59* RGB16-bit format (Data type = 0x0E)

Chart *16-60* RGB 18-bit format (Data type = 0x1E)

Note: RGB 18-bit only supports data type = 0x0E, loosely Packet mode (data type = 0x2E) is not supported.

Figure *16-61* RGB 24-bit Format (Data type = 0x3E)

Figure *16-62* RGB 30-bit format (Data type = 0x0D)

16.5.3.5 interface timing

The timing marks are as follows

Figure *16-63 MIPI TX* Timing Markers

VSA: number of frame synchronization

algorithmic technology

lines VBP: number of post-frame blanking lines
VACT: valid data lines of the frame

VFP: frame pre-fade line VSS: frame synchronization signal

HSS: Line Synchronization Signal HBP: Post Line Blanking Area RGB: Line Valid Data HFP: Pre Line Blanking Area

BLLP: Blanking or Low-Power Interval

Video The vertical timing is
Frame Synchronization Signal (VSS), Frame Backward Blanking Zone (VBP), Valid Row Area (VACT), Frame Pre-Fade Zone (VFP), Frame Synchronization Signal NO.
The horizontal timing is
Line Synchronization Signal (HSS), Line Post Fade Zone (HBP), Active Pixel (HACT), Line Pre Fade Zone (HFP), Line Synchronization Signal.
The valid synchronization signals are VSS, HSS, and the valid pixel is the intersection of the valid line area (VACT) and the valid pixel (HACT).

Video mode Burst Transmission timing is as follows

algorithmic technology

Figure *16-64 MIPI TX Video mode Burst Transmission* Timings

The MIPI Tx transmits only valid synchronization signals and data, and enters the BLLP area the rest of the time to reduce power consumption.

16.5.4 MIPI Tx Register Overview

MIPI Tx Control register location is 0x0A08A000

Name	Address Offset	Description
DSI_MAC_REG_00	0x000	DSI_MAC_EN
DSI_MAC_REG_01	0x004	HS_REG_00
DSI_MAC_REG_02	0x008	HS_REG_01
DSI_MAC_REG_03	0x00c	ESC_REG
DSI_MAC_REG_04	0x010	ESC_REG_TX0
DSI_MAC_REG_05	0x014	ESC_REG_TX1
DSI_MAC_REG_06	0x018	ESC_REG_TX2
DSI_MAC_REG_07	0x01c	ESC_REG_TX3
DSI_MAC_REG_08	0x020	ESC_REG_RX0
DSI_MAC_REG_09	0x024	ESC_REG_RX1

16.5.5 MIPI Tx Register Description

DSI_MAC_REG_00

Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	reg_sw_spkt_en	RWS	trigger soft short packet transmission (only work while reg_video_mode=0)	
1	reg_esc_en	RWS	Escape mode enable	
2	reg_video_mode	RWS	HS video mode enable 0: HS command mode or Escape mode 1: HS Video mode	
3	Reserved			
4	reg_sw_spkt_done	R	soft Short packet transmission finish , and clear after reg_sw_spkt_en deassert	
5	reg_esc_done_d0	R	Escape mode finish	
31:6	Reserved			

DSI_MAC_REG_01

Offset Address: 0x004

Bits	Name	Access	Description	Reset
23:0	reg_sw_spkt	R/W	Soft Short packet for HS mode [7:0] Data ID [15:8]: Data 0 [23:16] Data 1 (* ECC is generated by HW)	0x0
25:24	reg_lane_mode	R/W	lane active setting while HS mode 0: D0 Lane active 1: D0/D1 Lane Active 2: D0/D1/D2/D3 Lane Active 3: reserved	0x1
26	reg_eotp_en	R/W	enable insert eopt short packet	0x1

Bits	Name	Access	Description	Reset
			followed each transmission	
27	reg_skew_cal	R/W	skew calibration	0x0
28	reg_skew_cal_ini	R/W	skew calibration initialize	0x0
29	reg_hs_c_conti	R/W	c lane continous mode	0x1
31:30	reg_rgb_clr_fmt	R/W	RGB Packed format 0: RGB888 1: RGB666 2: RGB565 3: RGB10-10-10	0x0

DSI_MAC_REG_02

Offset Address: 0x008

Bits	Name	Access	Description	Reset
15:0	reg_lpkt_wc	R/W	Video line data paket byte counts (packed data must be byte alignment)	0x0
26:16	reg_event_delay	R/W	time shift for each video packet event (unit: pixel time)	0x80
31:27	Reserved			

DSI_MAC_REG_03

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
2:0	reg_esc_mode	R/W	Escape mode 0: TX trigger 1: TX LPDT 2: ULPS 3: ULPS EXIT 4: RX 5: reserved 6: C ULPS 7: C ULPS EXIT	0x1
3	Reserved			
7:4	reg_esc_trig	R/W	Trigger command for Escape TX trigger mode	0x0
11:8	reg_tx_bc	R/W	TX LPDT byte counts 0: 1 byte 1: 2 bytes ..... 15: 16 bytes	0x0
15:12	reg_bta_rx_bc	R	Byte counts for RX received 0: no received 1: 1 byte ..... 8: 8 bytes	
31:16	Reserved			

DSI_MAC_REG_04

Offset Address: 0x010

Bits	Name	Access	Description	Reset
7:0	reg_tx_byte0	R/W	TX byte 0	0x0
15:8	reg_tx_byte1	R/W	TX byte 1	0x0
23:16	reg_tx_byte2	R/W	TX byte 2	0x0

Bits	Name	Access	Description	Reset
31:24	reg_tx_byte3	R/W	TX byte 3	0x0

DSI_MAC_REG_05

Offset Address: 0x014

Bits	Name	Access	Description	Reset
7:0	reg_tx_byte4	R/W	TX byte 4	0x0
15:8	reg_tx_byte5	R/W	TX byte 5	0x0
23:16	reg_tx_byte6	R/W	TX byte 6	0x0
31:24	reg_tx_byte7	R/W	TX byte 7	0x0

DSI_MAC_REG_06

Offset Address: 0x018

Bits	Name	Access	Description	Reset
7:0	reg_tx_byte8	R/W	TX byte 8	0x0
15:8	reg_tx_byte9	R/W	TX byte 9	0x0
23:16	reg_tx_bytea	R/W	TX byte 10	0x0
31:24	reg_tx_byteb	R/W	TX byte 11	0x0

DSI_MAC_REG_07

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
7:0	reg_tx_bytec	R/W	TX byte 12	0x0
15:8	reg_tx_byted	R/W	TX byte 13	0x0
23:16	reg_tx_bytee	R/W	TX byte 14	0x0
31:24	reg_tx_bytef	R/W	TX byte 15	0x0

DSI_MAC_REG_08

Offset Address: 0x020

Bits	Name	Access	Description	Reset
7:0	reg_bta_rx_byte0	R	RX byte 0	
15:8	reg_bta_rx_byte1	R	RX byte 1	
23:16	reg_bta_rx_byte2	R	RX byte 2	
31:24	reg_bta_rx_byte3	R	RX byte 3	

DSI_MAC_REG_09

Offset Address: 0x024

Bits	Name	Access	Description	Reset
7:0	reg_bta_rx_byte4	R	RX byte 4	
15:8	reg_bta_rx_byte5	R	RX byte 5	
23:16	reg_bta_rx_byte6	R	RX byte 6	
31:24	reg_bta_rx_byte7	R	RX byte 7	

16.5.6 MIPI Tx PHY Register Overview

MIPI Tx PHY register location 0x0A0D_1000

Name	Address Offset	Description
REG_00	0x000	DSI PHY REG_00
REG_01	0x004	DSI PHY REG_01
REG_02	0x008	DSI PHY REG_02
REG_03	0x00c	DSI PHY REG_03
REG_04	0x010	DSI PHY REG_04
REG_05	0x014	DSI PHY REG_05
REG_23	0x08c	DSI PHY REG_23
REG_24	0x090	DSI PHY REG_24
REG_25	0x094	DSI PHY REG_25
REG_26	0x098	DSI PHY REG_26
REG_27	0x09c	DSI PHY REG_27
REG_28	0x0a0	DSI PHY REG_28
REG_2D	0x0b4	DSI PHY REG_2D

16.5.7 MIPI Tx PHY Register Description

REG_00

Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	reg_clk_lane_en	R/W	MIPI Tx Clock Lane Enable	0x0
1	reg_dat_0_lane_en	R/W	MIPI Tx data Lane 0 enable	0x0
2	reg_dat_1_lane_en	R/W	MIPI Tx data Lane 1 enable	0x0
3	reg_dat_2_lane_en	R/W	MIPI Tx data Lane 2 enable	0x0
4	reg_dat_3_lane_en	R/W	MIPI Tx data Lane 3 enable	0x0
31:5	Reserved			

REG_01

Offset Address: 0x004

Bits	Name	Access	Description	Reset
7:0	reg_t_clk_prepare	R/W	CLK-PREPARE timing parameter, unit . 8UI	0x5
15:8	reg_t_clk_zero	R/W	CLK-ZERO timing parameter, unit : 8UI	0x24
23:16	reg_t_clk_pre	R/W	CLK-PRE timing parameter, unit : 8UI	0x8
31:24	reg_t_clk_post	R/W	CLK-POST timing parameter, unit : 8UI	0x20

REG_02

Offset Address: 0x008

Bits	Name	Access	Description	Reset
7:0	reg_t_clk_trail	R/W	CLK-TRAIL timing parameter, unit : 8UI	0x1
31:8	Reserved			

REG_03

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
19:0	reg_t_esc_init	R/W	ESC-INIT timing parameter, unit : T_LPX	0x1000
31:20	Reserved			

REG_04

Offset Address: 0x010

Bits	Name	Access	Description	Reset
19:0	reg_t_esc_wakeup	R/W	ESC-WAKEUP timing parameter, unit : T_LPX	0x4e20
31:20	Reserved			

REG_05

Offset Address: 0x014

Bits	Name	Access	Description	Reset
7:0	reg_t_hs_pre_on	R/W	HS-PRE-ON timing parameter, unit : 8UI	0x1
15:8	reg_t_hs_prepare	R/W	HS-PREPARE timing parameter, unit. 8UI	0x6
23:16	reg_t_hs_zero	R/W	HS-ZERO timing parameter, unit : 8UI	0x20
31:24	reg_t_hs_trail	R/W	HS-TRAIL timing parameter, unit : 8UI	0x1

REG_23

Offset Address: 0x08c

Bits	Name	Access	Description	Reset
0	Reserved			
1	reg_en_ssc	R/W	MIPI Tx Spred Spectrum Enable	0x0
3:2	reg_ssc_mode	R/W	Spread Spectrum Mode	0x0
31:4	Reserved			

REG_24

Offset Address: 0x090

Bits	Name	Access	Description	Reset
31:0	reg_set	R/W	Frequency Synthesizer divider, format 6.26	0x11F51 4F9

REG_25

Offset Address: 0x094

Bits	Name	Access	Description	Reset
15:0	reg_span	R/W	SSC Span parameter	0x0
31:16	Reserved			

REG_26

Offset Address: 0x098

Bits	Name	Access	Description	Reset
23:0	reg_step	R/W	SSC Step parameter	0x0
31:24	Reserved			

REG_27

Offset Address: 0x09c

Bits	Name	Access	Description	Reset
2:0	reg_dsi_lane_0_sel	R/W	MIPI Tx PAD0 Config 0 : clock lane 1 : Data Lane 0 2 : Data Lane 1 3 :: Data Lane 2 4 :: Data Lane 3	0x1
3	Reserved			
6:4	reg_dsi_lane_1_sel	R/W	MIPI Tx PAD1 Config 0 : clock lane 1 : Data Lane 0	0x2

Bits	Name	Access	Description	Reset
			2 : Data Lane 1 3 :: Data Lane 2 4 :: Data Lane 3	
7	Reserved			
10:8	reg_dsi_lane_2_sel	R/W	MIPI Tx PAD2 Config 0 : clock lane 1 : Data Lane 0 2 : Data Lane 1 3 :: Data Lane 2 4 :: Data Lane 3	0x0
11	Reserved			
14:12	reg_dsi_lane_3_sel	R/W	MIPI Tx PAD3 Config 0 : clock lane 1 : Data Lane 0 2 : Data Lane 1 3 :: Data Lane 2 4 :: Data Lane 3	0x3
15	Reserved			
18:16	reg_dsi_lane_4_sel	R/W	MIPI Tx PAD4 Config 0 : clock lane 1 : Data Lane 0 2 : Data Lane 1 3 :: Data Lane 2 4 :: Data Lane 3	0x4
23:19	Reserved			
28:24	reg_sel_clk_lane	R/W	MIPI Tx Clock lane Config 5'h01 : PAD0 as MIPI Tx Clock Lane 5'h02 : PAD1 as MIPI Tx Clock Lane 5'h04 : PAD2 as MIPI Tx Clock Lane 5'h08 : PAD3 as MIPI Tx Clock Lane 5'h10 : PAD4 as MIPI Tx Clock Lane	0x4
31:29	Reserved			

REG_28

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
0	reg_dsi_lane_0_pn_swap	R/W	MIPI Tx PAD0 PN swap enable	0x0
1	reg_dsi_lane_1_pn_swap	R/W	MIPI Tx PAD1 PN swap enable	0x0
2	reg_dsi_lane_2_pn_swap	R/W	MIPI Tx PAD2 PN swap enable	0x0
3	reg_dsi_lane_3_pn_swap	R/W	MIPI Tx PAD3 PN swap enable	0x0
31:4	Reserved			

REG_2D

Offset Address: 0x0b4

Bits	Name	Access	Description	Reset
0	reg_lvds_en	R/W	MIPI Tx modue in LVDS Tx mode	0x0
31:1	Reserved			

17 audio interface

17.1 AIAO

17.1.1 summarize

Audio Input/Audio Output interface (Audio Input/Audio Output) is used to interface with the built-in Audio Codec of the chip or the external Audio Codec of the chip and the digital microphone, to complete the transmission and reception of audio data, and to realize the recording and playback.
Audio, intercom and other functions. The chip integrates AIAO related modules into one subsystem, and the built-in Audio Codec ADC/DAC supports stereo input and output. It supports two I2S interfaces externally and four I2S TX/RX modules internally, which can receive and send audio data simultaneously and support multi-channel data transmission and reception at the same time. The basic module block diagram is shown below:

Figure *17-1 AIAO* Block Diagram

17.1.2 specificities

algorithmic technology

The AIAO interface supports Master-mode, Slave-mode I2S and PCM modes, and multi-channel TDM mode. Receiving and transmitting audio data accesses the DDR space through DMA. Specific features are listed below:

- Highly flexible and configurable timing parameters, frame period, frame sync signal duration and polarity are all configurable
- Configurable clock edge for signaling and sampling
- Supports master and slave mode stereo I2S mode audio data transmission and reception
- Supports sending and receiving audio data in both mono and stereo PCM modes in both master and slave modes.
- Supports transmission and reception of audio data in both master and slave multi-channel TDM modes
- Receive and transmit can be enabled separately or simultaneously
- Data is DMA operated and can be accessed cyclically through a software-opened buffer.

17.1.2.1 PCM Interface

- Supports 16-bit linear PCM encoding for transmit and receive
- The PCM interface frame synchronization signals support short pulses (1 clock cycle) and long pulses (configurable number of clock cycles).
- Interface timing supports standard and left-aligned modes

17.1.2.2 I2S Interface

- Supports 16/24-bit data transmission and reception
- Support 8kHz ~ 192kHz sampling rate
- The I2S interface frame synchronization signal can support low-level left channel or high-level left channel
- Interface timing supports standard and left-aligned modes

17.1.3 Functional Description

The AIAO subsystem connects the built-in audio codec, I2S pins, and TX/RX modules through the internal PINMUX, and the registers can be configured by the software according to the application to achieve different connections.

17.1.3.1 Typical

Applications

Typical applications are as follows:

- Supports I2S slave mode to interface with built-in Audio Codec ADC, or I2S/PCM/TDM master/slave mode to interface with external ADC for audio capture.
- Supports I2S master mode to interface with built-in Audio Codec DAC, or I2S/PCM/TDM master/slave mode to interface with external DAC for audio playback.

Fig. *17-2* Diagram of connection with built-in *Audio Codec* via *I2S* interface

Fig. *17-3* Diagram of *AIAO* connecting with external *Audio Codec* in Master Mode via *I2S* Interface

algorithmic technology

Figure *17-4* Diagram of *AIAO* connecting to external *Audio Codec* in slave mode via *I2S* interface

17.1.3.2 Functional principle

The audio source is converted from Analog-to-Digital by the built-in or external Audio Codec ADC to audio data, which is received by the docked RX module through the I2S or PCM interface, stored in the circular buffer through DMA, and then taken away by the CPU for storage to complete the recording function, while the TX module reads the audio data from the circular buffer through DMA, and transmits the audio data to the docked built-in or external Audio Codec DAC for Digital-to-Digital through the I2S or PCM interface. The TX module reads the audio data from the circular buffer via DMA, and transmits the audio data via I2S or PCM interface to the docked internal or external Audio Codec DAC for Digital-to-Digital recording.

Analog conversion for audio playback.

When interfacing to an external I2S interface, the supported I2S timing is shown in Figure 17-5.

(*fs_offset* = 1, *fs_idef* = 1, *fs_polarity* = 0, *slot_num* = 1, *slot_size* = 31, *data_size* = 23, *fb_offset* = 0, *slot_en* = 16'h3)
(*tx_sample_edge* = 0, *rx_sample_edge* = 1)

Figure *17-5 I2S* Interface Timing

Fig. 17-5 Using the audio data width of 24-bit as an example, the data is transmitted in MSB First mode, where the MSB is delayed by one BCLK cycle with respect to the LRCK signal, and the data and LRCK signals are sent out using the falling edge of the BCLK, and are sampled at the rising edge of the BCLK (tx_sample_edge = 0, rx_sample_edge = 1). sample_edge = 0, rx_sample_edge = 1).

When connecting to external PCM interface, it supports PCM standard timing and data left-aligned timing, and the standard mode timing is shown in the figure. 17-6, and the left-aligned mode timing is shown in Figure 17-7.

Figure *17-6 PCM* Interface Standard Mode Timings

Figure 11-6 shows an example of 16-bit stereo audio data width, where the data is transmitted in MSB First mode, where the MSB is delayed by one BCLK cycle with respect to the LRCK signal, and the data and LRCK signals are sent out using the falling edge of the BCLK, and sampled at the rising edge of the BCLK (tx_sample_edge = 0, rx_sample_edge = 1). 1).

Figure *17-7 PCM* Interface Left-Aligned Mode Timing

In left-aligned mode, data and LRCK signals are sent at the beginning of the same beat.

17.1.4 working method

The internal Audio Codec of AIAO and the external Audio Codec via I2S interface can be configured by software to work simultaneously as shown in Figures 11-2 to 11-4, and the software needs to be configured as priority before enabling the data transfer.
AIAO subsystem registers i2s_tdm_sclk_in_sel, i2s_tdm_fs_in_sel, i2s_tdm_sdi_in_sel, i2s_tdm_sdo_out_sel, to connect each interface to the TX/RX module of the corresponding operation (I2S_TDM_0~I2S_TDM_3).

17.1.4.1 Clock gating and clock configuration

If AIAO operation is in master mode, the TX/RX module register maste_mode, which is the source of the MCLK/BCLK clock, must be set to 1, and depending on the sample rate, the crossover register I2S_CLK_CTRL1(mclk_div, bclk_div) must be configured, and then register I2S_CLK_CTRL0(aud_en) must be set to 1 to turn on clock gating.

17.1.4.2 soft reset (electronics)

Each of the four TX/RX modules integrated in the AIAO has its own soft reset. Before enabling the TX/RX modules for data transfer, the registers FIFO_RESET and I2S_RESET must be configured for soft reset.

17.1.5 AIAO Register Overview

An overview of the AIAO subsystem registers is shown in Table 17-1.

Table *17-1 AIAO* Subsystem Registers Overview *(*Base Address *0x0410_8000)*

Name	Address Offset	Description
i2s_tdm_sclk_in_sel	0x000	Select Slave Mode TX/RX Module SCLK Source
i2s_tdm_fs_in_sel	0x004	Select Slave Mode TX/RX Module Frame Synchronization Signal Source
i2s_tdm_sdi_in_sel	0x008	Select RX Module SDI Signal Source

i2s_tdm_sdo_out_sel	0x00c	Select the source of the subsystem SDO output
i2s_bclk_oen_sel	0x030	BCLK IO output control
audio_pdm_ctrl	0x040	Enable PDM mode
i2s_sys_int_en	0x060	Enable I2S Subsystem Interrupt Signal
i2s_sys_ints	0x064	I2S Subsystem Interrupt Signal Status

An overview of the I2S_TDM_0~I2S_TDM_3 module registers is shown in Table 17-2.

Table *17-2 I2S_TDM_0/1/2/3* Register Overview *(*Base Address *0x0410_0000 + n*0x10000)*

Name	Address Offset	Description
BLK_MODE_SETTING	0x000	TX/RX Module Operational Control
FRAME_SETTING	0x004	Audio frame timing control
SLOT_SETTING1	0x008	Channel and Data Control
SLOT_SETTING2	0x00c	Enable channel
DATA_FORMAT	0x010	Setting the storage data format
BLK_CFG	0x014	TX/RX Module Function Configuration
I2S_ENABLE	0x018	Enable TX/RX Module
I2S_RESET	0x01c	TX/RX Module Reset
I2S_INT_EN	0x020	disable interrupt (computing)
I2S_INT	0x024	interrupt status
FIFO_THRESHOLD	0x028	FIFO threshold
I2S_LRCK_MASTER	0x02c	Enable LRCK Master Mode Continuous Output Mode
FIFO_RESET	0x030	FIFO reset
RX_STATUS	0x040	RX Module Internal Status Register
TX_STATUS	0x048	TX module internal status register
I2S_CLK_CTRL0	0x060	Clock control
I2S_CLK_CTRL1	0x064	Master Mode Clock Divider
I2S_PCM_SYNTH	0x068	PCM Frame Synchronization Synthesis Mode Control
RX_RD_PORT	0x080	RX FIFO read port
TX_WR_PORT	0x0c0	TX FIFO write port

17.1.6 AIAO Register Description

17.1.6.1 AIAO Subsystem Register Description

i2s_tdm_sclk_in_sel

Select sclk source

Offset Address:

0x000

Bits	Name	Access	Description	Reset

2:0	i2s_tdm_0_sclk_in_sel	R/W	When i2s_tdm_0 is operating in slave mode, select the sclk Input source 3'b000 = reserved 3'b001 = from i2s_tdm_1 bclk_out 3'b010 = from i2s_tdm_2 bclk_out 3'b011 = from i2s_tdm_3 bclk_out 3'b100 = from the built-in Audio Codec ADC BCLK 3'b101 = from IO I2S1_BCLK 3'b110 = from IO I2S2_BCLK	0x4

Bits	Name	Access	Description	Reset
			3'b111 = reserved	
3	Reserved			
6:4	i2s_tdm_1_sclk_in_sel	R/W	When i2s_tdm_1 operates in slave mode, select the sclk Input source 3'b000 = from i2s_tdm_0 bclk_out 3'b001 = from audio_pdm module sclk_out 3'b010 = from i2s_tdm_2 bclk_out 3'b011 = from i2s_tdm_3 bclk_out 3'b100 = from the built-in Audio Codec ADC BCLK 3'b101 = from IO I2S1_BCLK 3'b110 = from IO I2S2_BCLK 3'b111 = reserved	0x5
7	Reserved			
10:8	i2s_tdm_2_sclk_in_sel	R/W	When i2s_tdm_2 is operating in slave mode, select the sclk Input source 3'b000 = from i2s_tdm_0 bclk_out 3'b001 = from i2s_tdm_1 bclk_out 3'b010 = reserved 3'b011 = from i2s_tdm_3 bclk_out 3'b100 = from the built-in Audio Codec ADC BCLK 3'b101 = from IO I2S1_BCLK 3'b110 = from IO I2S2_BCLK 3'b111 = reserved	0x6
11	Reserved			
14:12	i2s_tdm_3_sclk_in_sel	R/W	When i2s_tdm_3 is operating in slave mode, select the sclk Input source 3'b000 = from i2s_tdm_0 bclk_out 3'b001 = from i2s_tdm_1 bclk_out 3'b010 = from i2s_tdm_2 bclk_out 3'b011 = reserved 3'b100 = from built-in Audio Codec ADC BCLK 3'b101 = from IO I2S1_BCLK 3'b110 = from IO I2S2_BCLK 3'b111 = reserved	0x7
31:15	Reserved			

i2s_tdm_fs_in_sel

Offset Address: 0x004

Bits	Name	Access	Description	Reset

2:0	i2s_tdm_0_fs_in_sel	R/W	When i2s_tdm_0 is operating in slave mode, select the LRCK/FS Input source 3'b000 = reserved 3'b001 = from i2s_tdm_1 lrck_out 3'b010 = from i2s_tdm_2 lrck_out 3'b011 = from i2s_tdm_3 lrck_out 3'b100 = from the built-in Audio Codec ADC LRCK 3'b101 = from IO I2S1_LRCK 3'b110 = from IO I2S2_LRCK 3'b111 = reserved	0x4

Bits	Name	Access	Description	Reset
3	Reserved			
6:4	i2s_tdm_1_fs_in_sel	R/W	When i2s_tdm_1 operates in slave mode, select the LRCK/FS Input source 3'b000 = from i2s_tdm_0 lrck_out 3'b001 = from audio_pdm module lrck_out 3'b010 = from i2s_tdm_2 lrck_out 3'b011 = from i2s_tdm_3 lrck_out 3'b100 = from the built-in Audio Codec ADC LRCK 3'b101 = from IO I2S1_LRCK 3'b110 = from IO I2S2_LRCK 3'b111 = reserved	0x5
7	Reserved			
10:8	i2s_tdm_2_fs_in_sel	R/W	When i2s_tdm_2 is operating in slave mode, select the LRCK/FS Input source 3'b000 = from i2s_tdm_0 lrck_out 3'b001 = from i2s_tdm_1 lrck_out 3'b010 = reserved 3'b011 = from i2s_tdm_3 lrck_out 3'b100 = from the built-in Audio Codec ADC LRCK 3'b101 = from IO I2S1_LRCK 3'b110 = from IO I2S2_LRCK 3'b111 = reserved	0x6
11	Reserved			
14:12	i2s_tdm_3_fs_in_sel	R/W	When i2s_tdm_3 is operating in slave mode, select the LRCK/FS Input source 3'b000 = from i2s_tdm_0 lrck_out 3'b001 = from i2s_tdm_1 lrck_out 3'b010 = from i2s_tdm_2 lrck_out 3'b011 = reserved 3'b100 = from built-in Audio Codec ADC LRCK 3'b101 = from IO I2S1_LRCK 3'b110 = from IO I2S2_LRCK 3'b111 = reserved	0x7
31:15	Reserved			

i2s_tdm_sdi_in_sel

Offset Address: 0x008

Bits	Name	Access	Description	Reset

2:0	i2s_tdm_0_sdi_in_sel	R/W	Select i2s_tdm_0 RX Module SDI Signal Source 3'b000 = reserved 3'b001 = from i2s_tdm_1 sdo 3'b010 = from i2s_tdm_2 sdo 3'b011 = from i2s_tdm_3 sdo 3'b100 = from the built-in Audio Codec ADC SDO 3'b101 = from IO I2S1_SDI 3'b110 = from IO I2S2_SDI 3'b111 = reserved	0x4
3	Reserved			
6:4	i2s_tdm_1_sdi_in_sel	R/W	Select i2s_tdm_1 RX Module SDI Signal Source	0x5

Bits	Name	Access	Description	Reset
			3'b000 = from i2s_tdm_0 sdo 3'b001 = from audio_pdm module sdo 3'b010 = from i2s_tdm_2 sdo 3'b011 = from i2s_tdm_3 sdo 3'b100 = from the built-in Audio Codec ADC SDO 3'b101 = from IO I2S1_SDI 3'b110 = from IO I2S2_SDI 3'b111 = reserved	
7	Reserved			
10:8	i2s_tdm_2_sdi_in_sel	R/W	Select i2s_tdm_2 RX Module SDI Signal Source 3'b000 = from i2s_tdm_0 sdo 3'b001 = from i2s_tdm_1 sdo 3'b010 = reserved 3'b011 = from i2s_tdm_3 sdo 3'b100 = from the built-in Audio Codec ADC SDO 3'b101 = from IO I2S1_SDI 3'b110 = from IO I2S2_SDI 3'b111 = reserved	0x6
11	Reserved			
14:12	i2s_tdm_3_sdi_in_sel	R/W	Select i2s_tdm_3 RX Module SDI Signal Source 3'b000 = from i2s_tdm_0 sdo 3'b001 = from i2s_tdm_1 sdo 3'b010 = from i2s_tdm_2 sdo 3'b011 = reserved 3'b100 = from built-in Audio Codec ADC SDO 3'b101 = from IO I2S1_SDI 3'b110 = from IO I2S2_SDI 3'b111 = reserved	0x7
31:15	Reserved			

i2s_tdm_sdo_out_sel

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
2:0	i2s_tdm_0_sdo_out_sel	R/W	Only allowed to be configured to the default value 0x4	0x4
3	Reserved			
6:4	i2s_tdm_1_sdo_out_sel	R/W	Select I2S1_SDO signal source 3'b100 = I2S1_SDO from i2s_tdm_0 TX Module sdo 3'b101 = I2S1_SDO from i2s_tdm_1 TX Module sdo 3'b110 = I2S1_SDO from i2s_tdm_2 TX Module sdo 3'b111 = I2S1_SDO from i2s_tdm_3 TX Module sdo Other values = reserved	0x5

7	Reserved			
10:8	i2s_tdm_2_sdo_out_sel	R/W	Select I2S2_SDO signal source 3'b100 = I2S2_SDO from i2s_tdm_0 TX Module sdo 3'b101 = I2S2_SDO from i2s_tdm_1 TX	0x6

Bits	Name	Access	Description	Reset
			Module sdo 3'b110 = I2S2_SDO from i2s_tdm_2 TX Module sdo 3'b111 = I2S2_SDO from i2s_tdm_3 TX Module sdo Other values = reserved	
11	Reserved			
14:12	i2s_tdm_3_sdo_out_sel	R/W	Allowed to be configured as default only 0x7 SDI fixed with built-in Audio Codec DAC from i2s_tdm_3 TX module sdo	0x7
31:15	Reserved			

i2s_bclk_oen_sel

Offset Address: 0x030

Bits	Name	Access	Description	Reset
0	i2s0_bclk_oen_sel	R/W	Reserved	0x0
1	i2s1_bclk_oen_sel	R/W	Selects the oen control method for I2S1_BCLK IOs 0 = controlled by i2s_tdm_1 TX module 1 = unified by register i2s1_bclk_oen_ext containment	0x0
2	i2s2_bclk_oen_sel	R/W	Selects the oen control method for I2S2_BCLK IOs 0 = controlled by i2s_tdm_2 TX module 1 = unified by register i2s2_bclk_oen_ext containment	0x0
3	i2s3_bclk_oen_sel	R/W	Allowed to be configured as default only 0x0	0x0
7:4	Reserved			
8	i2s0_bclk_oen_ext	R/W	Reserved	0x0
9	i2s1_bclk_oen_ext	R/W	I2S1_BCLK IO's controlled by subsystem registers oen 0 = disable IO output 1 = Enable IO output	0x0
10	i2s2_bclk_oen_ext	R/W	I2S2_BCLK IO's controlled by subsystem registers oen 0 = disable IO output 1 = Enable IO output	0x0
11	i2s3_bclk_oen_ext	R/W	Allowed to be configured as default only 0x0	0x0
15:12	Reserved			
16	i2s_bclk_oen_no_delay	R/W	Allowed to be configured as default only 0x0	0x0
31:17	Reserved			

audio_pdm_ctrl

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	Reserved			
1	audio_pdm_sel_i2s1	R/W	Enable PDM mode 0 = general operating mode 1 = I2S2 IO operates in PDM mode using the i2s_tdm_1 RX module receiving data When this value is set to 1, the oen of the I2S2_BCLK IO is set to 1, and the oen of the I2S2_BCLK IO is set to 1. Fixed by register i2s2_bclk_oen_ext regulate	0x0
31:2	Reserved			

i2s_sys_int_en

Offset Address: 0x060

Bits	Name	Access	Description	Reset
0	i2s0_int_en	R/W	Enable I2S0 interrupt	0x1
1	i2s1_int_en	R/W	Enable I2S1 interrupt	0x1
2	i2s2_int_en	R/W	Enable I2S2 interrupt	0x1
3	i2s3_int_en	R/W	Enable I2S3 interrupt	0x1
7:4	Reserved			
8	i2s_subsys_int_en	R/W	Enables the I2S_SUBSYS interrupt.	0x1
31:9	Reserved			

i2s_sys_ints

Offset Address: 0x064

Bits	Name	Access	Description	Reset
0	i2s0_int	RO	I2S0 interrupt status When an I2S0 interrupt occurs, I2S0 can be further read. Register I2S_INT value to determine the interrupt status.	
1	i2s1_int	RO	I2S1 interrupt status When an I2S1 interruption occurs, I2S1 can be further read. Register I2S_INT value to determine the interrupt status.	
2	i2s2_int	RO	I2S2 interrupt status When an I2S2 interruption occurs, the I2S2 can be further read. Register I2S_INT value to determine the interrupt status.	
3	i2s3_int	RO	I2S3 interrupt status When an I2S3 interruption occurs, the I2S3 can be further read. Register I2S_INT value to determine the interrupt status.	
31:4	Reserved			

17.1.6.2 I2S_TDM Module Register Description

BLK_MODE_SETTING

Offset Address: 0x000

Bits	Name	Access	Description	Reset

algorithmic technology

0	tx_mode	R/W	Selecting the transmission mode 0 = receive mode (RX), 1 = transmit mode (TX)	0x0
1	master_mode	R/W	Select Master or Slave mode 0 = slave mode 1 = Master mode, BCLK and LRCK are generated internally	0x0
2	rx_sample_edge	R/W	Selects the BCLK clock sampling margin for the received signal 0 = Falling edge sampling 1 = rising edge sampling	0x1
3	tx_sample_edge	R/W	Selects the BCLK clock margin for sending data SDOs. 0 = Falling edge change	0x0

Bits	Name	Access	Description	Reset
			1 = Rising edge change	
6:4	Reserved			
7	dma_mode	R/W	Data migration model 0 = software mode, 1 = DMA mode	0x1
8	Reserved	R/W	Used for internal debugging, only allowed to be configured to the default value. 0x0	0x0
9	Reserved			
10	slave_tx_fs_direct_in	R/W	Selects the source of the LRCK when sending data in slave mode. 0 = LRCK signal is generated internally by the TX module 1 = LRCK signal directly from master This register is for debugging purposes only.	0x0
11	Reserved			
12	pcm_synth_mode	R/W	Master Mode PCM Frame Synchronization Signal Using Frequency Synthesis Mode 0 = PCM Frame synchronization signal period is defined by frame_length control 1 = PCM frame synchronization signal period is defined by ck_coef_n/ck_coef_m control	0x0
15:13	Reserved			
31:16	reg_dummy	R/W	Dummy register	0xff00

FRAME_SETTING

Offset Address: 0x004

Bits	Name	Access	Description	Reset
8:0	frame_length	R/W	Audio frame width (BCLK clock) 0~511 = 1~512 bits	0x1F
11:9	Reserved			
12	fs_polarity	R/W	LRCK Signal Polarity 0 = start low (active) 1 = start high (active) I2S mode, set 0 for left channel priority. LRCK output low, set 1 for left channel priority, LRCK output high. pcm mode. Setting 1 indicates that the frame synchronization pulse signal is high.	0x0
13	fs_offset	R/W	Frame Audio Data MSB Delay 0 = same beat as LRCK signal 1 = one beat delay from LRCK signal	0x1
14	fs_idef	R/W	Selects the LRCK signaling mode 0 = as PCM/TDM frame synchronization 1 = as I2S left and right channels	0x1

15	Reserved			
23:16	fs_active_length	R/W	LRCK Signal Effective Width 0~255 = 1~256 bits.	0x0F
31:24	Reserved			

SLOT_SETTING1

Offset Address: 0x008

Bits	Name	Access	Description	Reset
3:0	slot_num	R/W	Number of channels within each audio frame 0~15 = 1~16 channels Configuration 1 indicates two channels (left/right).	0x1

Bits	Name	Access	Description	Reset
7:4	Reserved			
13:8	slot_size	R/W	Channel Width 0~63 = 1~64 bits	0x0F
15:14	Reserved			
20:16	data_size	R/W	Effective data width in the sound channel 0~31 = 1~32 bits In the general case (e.g., I2S and PCM modes), the mated Set the data width and channel width to the same value.	0x0F
23:21	Reserved			
28:24	fb_offset	R/W	Data in channel MSB delay 0 = data and channel timings are on the same beat. 1~31 = Delay 1~31 bits This value controls the value of the first valid bit in each channel. Timing.	0x0
31:29	Reserved			

SLOT_SETTING2

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
15:0	slot_en	R/W	Enable valid channel data Configure slot_en[n] to 1 for the nth channel data to be valid and 0 for invalid. In TX mode, if a channel is set to invalid, the data sent from that channel will be 0. In RX mode, if a channel is set to invalid, the data from that channel will be 0. When received, it is replaced by all zeros.	0x0003
31:16	Reserved			

DATA_FORMAT

Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	data_format	R/W	Allowed to be configured as default only 0x0	0x0
2:1	word_length	R/W	Selecting Audio Data File Memory Width 2'b00 = 8-bit. 2'b01 = 16-bit, 2'b10 = 32-bit. 2'b11 = reserved If the number of valid bits of the received and sent data is less than this value, the Align the data to the right, MSB by 0.	0x1

3	pad_slot_no	R/W	Fill channel notes into the RX data. 0 = None 1 = 4-bit channel sequential padding to audio data MSB When 24-bit data is received, bits 31 to 28 can be set. Fill in the channel order and store it as 32-bit. Provides software post-processing applications.	0x0
4	skip_rx_inactive_slot	R/W	Ignore channel data set to invalid in RX mode. 0 = when receiving channel data that is set to invalid If you want to store the whole 0, it will be replaced with 0. 1 = when receiving channel data that is set to invalid If it is ignored, it will not be stored in the RX FIFO.	0x0
5	skip_tx_inactive_slot	R/W	Ignore the number of channels set to invalid in TX mode.	0x0

Bits	Name	Access	Description	Reset
			act in accordance with 0 = W h e n sending channel data that is set to invalid, the data is read from the TX FIFO, but is replaced with all zeros and sent. 1 = W h e n sending channel data that is set to invalid, it will not be read from the TX FIFO but will be sent directly to the TX FIFO. Send all 0 bits.	
6	tx_source_left_align	R/W	TX mode memory data is left-aligned 0 = data right-aligned 1 = data left-aligned If the memory width is greater than the audio data width, e.g., if the memory width is 32-bit and the data width is 24-bit, then the audio data width is greater than the audio data width. If this value is set to 0, it will take [23:0] and send it out. Send, when this value is configured to 1, it will take [31:8] and send it out. Send.	0x0
31:7	Reserved			

BLK_CFG

Offset Address: 0x014

Bits	Name	Access	Description	Reset
0	force_complete	R/W	compulsory termination mode 0 = when i2s_enable is set to 0 from 1, it does not end until a complete frame of audio data has been received or sent 1 = Force end of data reception or transmission when i2s_enable is set to 0 from 1. The default value of 0x0 is recommended.	0x0
1	dma_req_force_stop	R/W	dma_req Force stop mode 0 = When i2s_enable is set from 1 to 0, the dma_req that has been sent will remain high until the dma_ack is received back from the DMA. set from 1 to 0 1 = cancel sent dma_req when i2s_enable is set to 0 from 1 The default value of 0x1 is recommended.	0x1
3:2	Reserved			

4	auto_disable_with_ch_en	R/W	I2S FIFO transfer auto-stop 0 = normal operation 1 = FIFO operation linked to system DMA When the DMA channel ends before the TX/RX module (i2s_enable set to 0) has ended the transmission, the TX/RX FIFO will be stopped continuously, in RX mode, data will continue to be received but will not be written to the FIFO, in TX mode, it will send the full number of zeros. The value will not be read by the FIFO.	0x0
5	Reserved			
6	rx_start_wait_dma_en	R/W	RX mode wait DMA enable 0 = RX mode starts after i2s_enable is set from 0 to 1. 1 = RX mode in i2s_enable set from 0 to 1 After that, it will wait for DMA to be enabled before starting operation. writings or works	0x0
7	Reserved			

Bits	Name	Access	Description	Reset
8	rx_blk_clk_force_en	R/W	RX Module Clock Gated Constant On 0 = Enable clock gating auto-off 1 = Clock hold constant on	0x0
9	rx_fifo_dma_clk_force_en	R/W	RX FIFO Control Module Clock Gated Constant On 0 = Enable clock gating auto-off 1 = Clock hold constant on	0x1
15:10	Reserved			
16	tx_blk_clk_force_en	R/W	TX Module Clock Gated Constant On 0 = Enable clock gating auto-off 1 = Clock hold constant on	0x0
17	tx_fifo_dma_clk_force_en	R/W	TX FIFO Control Module Clock Gated Constant On 0 = Enable clock gating auto-off 1 = Clock hold constant on	0x1
31:18	Reserved			

I2S_ENABLE

block enable

Offset Address: 0x018

Bits	Name	Access	Description	Reset
0	i2s_enable	R/W	Enable TX/RX Module 0 = off TX/RX 1 = Enable TX/RX module Before you can configure this value to 1, you must configure the tx_fifo_reset/rx_fifo_reset and i2s_reset_tx/i2s_reset_rx.	0x0
31:1	Reserved			

I2S_RESET

sw reset

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
0	i2s_reset_rx	R/W	RX Module Soft Reset 0 = no reset, 1 = reset Write 1 and then 0 to this value to reset the RX module. Because of the cross-clock domain, you must wait for a certain amount of time after writing 1 to reset the RX module. Time to write 0 again to allow the reset signal to take effect.	0x0
1	i2s_reset_tx	R/W	TX Module Soft Reset 0 = no reset, 1 = reset Write 1 and then 0 to this value to reset the TX module. Because of the cross-clock domain, you must wait for a certain amount of time after writing 1 to reset the TX module. Time to write 0 again to allow the reset signal to take effect.	0x0
31:2	Reserved			

I2S_INT_EN

interrupt enable
Offset Address:
0x020

Bits	Name	Access	Description	Reset
0	rx_fifo_avail_int_en	R/W	Enable RX FIFO data available interrupt.	0x0
1	rx_fifo_overflow_int_en	R/W	Enable RX FIFO overflow interrupt.	0x1
2	rx_fifo_underflow_int_en	R/W	Enable RX FIFO underflow interrupt.	0x1
3	Reserved			
4	tx_fifo_avail_int_en	R/W	Enable TX FIFO data available interrupt.	0x0

Bits	Name	Access	Description	Reset
5	tx_fifo_overflow_int_en	R/W	Enable TX FIFO overflow interrupt	0x1
6	tx_fifo_underflow_int_en	R/W	Enable TX FIFO underflow interrupt.	0x1
7	Reserved			
8	i2s_int_en	R/W	Enable I2S IP interrupt All I2S interrupt signals are combined into 1-bit, ❹ inverted. should be in the I2S subsystem register i2s_sys_ints	0x1
31:9	Reserved			

I2S_INT

interrupt status
Offset Address:
0x024

Bits	Name	Access	Description	Reset
0	rx_fifo_avail_int	RO	RX FIFO data valid interrupt status When RX FIFO fullness is greater than rx_fifo_threshold Write 1 to clear the interrupt status. State. (rx_fifo_avail_int_en needs to be set to 1)	
1	rx_fifo_overflow_int	RO	RX FIFO Overflow Interrupt Status Write 1 to clear the interrupt status. (rx_fifo_overflow_int_en needs to be set to 1)	
2	rx_fifo_underflow_int	RO	RX FIFO Underflow Interrupt Status Write 1 to clear the interrupt status. (rx_fifo_underflow_int_en needs to be set to 1)	
3	Reserved			
4	tx_fifo_avail_int	RO	TX FIFO has an air-disconnect state When TX FIFO null is greater than tx_fifo_threshold Write 1 to clear the interrupt status. State. (tx_fifo_avail_int_en needs to be set to 1)	
5	tx_fifo_overflow_int	RO	TX FIFO Overflow Interrupt Status Write 1 to clear the interrupt status. (tx_fifo_overflow_int_en needs to be set to 1)	
6	tx_fifo_underflow_int	RO	TX FIFO Underflow Interrupt Status Write 1 to clear the interrupt status. (tx_fifo_underflow_int_en needs to be set to 1)	
7	Reserved			

8	rx_fifo_avail_int_raw	RO	RX FIFO data valid interrupt status When RX FIFO fullness is greater than rx_fifo_threshold Write 1 to clear the interrupt status. State. (rx_fifo_avail_int_en needs to be set to 1)	
9	rx_fifo_overflow_int_raw	RO	RX FIFO overflow raw interrupt status Write 1 to clear the interrupt status.	
10	rx_fifo_underflow_int_raw	RO	RX FIFO Underflow Raw Interrupt Status Write 1 to clear the interrupt status.	
11	Reserved			
12	tx_fifo_avail_int_raw	RO	TX FIFO empty raw interrupt status When TX FIFO null is greater than tx_fifo_threshold Write 1 to clear the interrupt status. State.	
13	tx_fifo_overflow_int_raw	RO	TX FIFO overflow raw interrupt status Write 1 to clear the interrupt status.	
14	tx_fifo_underflow_int_raw	RO	TX FIFO underflow raw interrupt status Write 1 to clear the interrupt status.	
31:15	Reserved			

FIFO_THRESHOLD

Offset Address: 0x028

Bits	Name	Access	Description	Reset
4:0	rx_fifo_threshold	R/W	RX FIFO threshold In DMA transfer mode, when the RX FIFO fullness is greater than or equal to this value, a read request is issued to the DMA Signal.	0x7
15:5	Reserved			
20:16	tx_fifo_threshold	R/W	TX FIFO threshold In DMA transfer mode, a write request is made to the DMA when the TX FIFO empty level is greater than or equal to this value Signal.	0x7
23:21	Reserved			
28:24	tx_fifo_high_threshold	R/W	TX FIFO high threshold After enabling the TX module, the TX FIFO will issue a write request to the DMA until the FIFO is full to a value greater than or equal to this value, and then the TX module will begin to send the number of external According to.	0x1F
31:29	Reserved			

I2S_LRCK_MASTER

block enable

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
0	i2s_lrck_master_enable	R/W	Enable i2s_tdm module as BCLK/LRCK master mode output generator 0 = When operating in master mode (master_mode set to 1), BCLK and LRCK signals will be output only after TX/RX is enabled (i2s_enable set to 1). 1 = When operation is in master mode and aud_en is set to 1, after setting this value to 1, BCLK and LRCK will start to output, at this time, the i2s_tdm module is used as single Pure master mode BCLK/LRCK signal generator.	0x0
31:1	Reserved			

FIFO_RESET

Offset Address: 0x030

Bits	Name	Access	Description	Reset
0	rx_fifo_reset	R/W	RX channel FIFO soft reset Write 1 followed by 0 to soft reset the RX FIFO.	0x0

15:1	Reserved			
16	tx_fifo_reset	R/W	TX channel FIFO soft reset Write 1 followed by 0 to soft reset the TX FIFO.	0x0
31:17	Reserved			

RX_STATUS

Offset Address: 0x040

Bits	Name	Access	Description	Reset
8:0	rx_frame_size_cnt	RO	RX module internal frame width counter value	
9	rx_i2s_disable_req	RO	internal signal	
10	i2s_rx_start_wait	RO	internal signal	
16:11	rx_data_size_cnt	RO	RX Module Internal Valid Data Width Counter Value	

Bits	Name	Access	Description	Reset
22:17	rx_slot_size_cnt	RO	RX module internal channel width counter value	
23	i2s_reset_rx_sclk	RO	The i2s_reset_rx signal in the clock domain of the RX module. attitude	
28:24	rx_slot_num_cnt	RO	RX module internal channel count counter value	
29	receive_start_sclk	RO	Status of the i2s_enable signal in the clock domain of the RX module	
30	rx_blk_active	RO	RX module in operation	
31	rx_dma_req	RO	RX module sends dma_req	

TX_STATUS

Offset Address: 0x048

Bits	Name	Access	Description	Reset
8:0	tx_frame_size_cnt	RO	TX module internal frame width counter value	
9	tx_i2s_disable_req	RO	internal signal	
10	i2s_tx_start_wait	RO	internal signal	
16:11	tx_data_size_cnt	RO	TX module internal valid data width counter value	
22:17	tx_slot_size_cnt	RO	TX module internal channel width counter value	
23	i2s_reset_tx_sclk	RO	i2s_reset_tx signaling in the clock domain of the TX module. attitude	
28:24	tx_slot_num_cnt	RO	TX module internal channel count counter value	
29	transmit_start_sclk	RO	Status of the i2s_enable signal in the clock domain of the TX module	
30	tx_blk_active	RO	TX module in operation	
31	tx_dma_req	RO	TX module sends dma_req	

I2S_CLK_CTRL0

Offset Address: 0x060

Bits	Name	Access	Description	Reset
0	aud_clk_sel	R/W	Select audio clock source (aud_clk) 0 = from internal audio PLL 1 = from external mclk_in	0x0
1	Reserved			
2	mclk_out_inv	R/W	Master Mode mclk_out Clock Signal External Interface Inverted 0 = mclk_out no inversion 1 = mclk_out inverted	0x0
3	bclk_out_inv	R/W	Main Mode bclk_out Clock Signal External Interface Inverted 0 = bclk_out without inverse 1 = bclk_out inverted	0x0

4	bclk_in_inv	R/W	Inverted from mode bclk_in clock signal 0 = bclk_in no inverse 1 = bclk_in inverted	0x0
5	Reserved			
6	bclk_out_clk_force_en	R/W	Main mode bclk_out Clock constant output 1 = constant output 0 = when TX/RX mode is enabled and the data is started output only when transmitting	0x1
7	mclk_out_en	R/W	Enable mclk_out IO Outputs 0 = disable 1 = enable	0x0

Bits	Name	Access	Description	Reset
8	aud_en	R/W	Enable clock generator and master mode clock outputs 0 = disable clock generator 1 = Enable the clock generator and output clock signal When in slave mode, configure this value to 0. When in master mode, you must configure the divider for the data sampling rate before configuring this value to 1. mclk_div/bclk_div.	0x0
31:9	Reserved			

I2S_CLK_CTRL1

Offset Address: 0x064

Bits	Name	Access	Description	Reset
15:0	mclk_div	R/W	mclk Clock from Audio Clock Divider 1 = divide by 1, 2 = divide by 2, 3 = divide by 3, ...	0x1
31:16	bclk_div	R/W	bclk clock divides from mclk clock (only available in master mode) 1 = divide by 1, 2 = divide by 2, 3 = divide by 3, ...	0x2

I2S_PCM_SYNTH

Offset Address: 0x068

Bits	Name	Access	Description	Reset
11:0	ck_coef_n	R/W	PCM Frame Synchronization Signal Period Crossover Coefficient N When pcm_synth_mode is set to 1, the frame synchronization signal period = BCLK * (N/M), and the value of N must be small. at the M-value.	0x1
15:12	Reserved			
31:16	ck_coef_m	R/W	PCM Frame Synchronization Signal Period Crossover Coefficient M When pcm_synth_mode is set to 1, the frame synchronization signal period = BCLK * (N/M), and the value of N must be small. at the M-value.	0x40

RX_RD_PORT

Offset Address: 0x080

Bits	Name	Access	Description	Reset
31:0	rx_rd_port	RO	RX FIFO Read Address The DMA needs to configure this address to be used as the Read the address.	

TX_WR_PORT

Offset Address: 0x0c0

Bits	Name	Access	Description	Reset

31:0	tx_wr_port	WO	TX FIFO Write Address The DMA needs to configure this address to be used as the address for sending data. Write Address.	0x0

17.2 Audio Codec

17.2.1 summarize

The chip integrates high-performance Audio Codec, including stereo playback DAC (90dB DR A-Weighted) supporting two single-ended lineout outputs; stereo recording ADC (90dB DR A-Weighted) supporting stereo single-ended inputs.

17.2.2 specificities

- 90dB DR A-Weighted Stereo DAC
- Stereo Single-Ended Lineout Output
- DAC digital volume control range: -24dB to 0dB
- DAC supports 8kHz~48kHz sampling rate.
- 90dB DR A-Weighted Stereo ADCs
- ADC PGA gain range: 0db ~48dB
- ADC support 8kHz~48kHz sampling rate
- ADC supports Mic in stereo single-ended input or Line in stereo single-ended input.

17.2.3 Audio Codec Register Overview

17.2.3.1 Audio DAC/ADC Register Overview

Name	Address Offset	Description
txdac_ctrl0	0x000	
txdac_ctrl1	0x004	
txdac_status	0x008	
txdac_afe0	0x00c	
txdac_afe1	0x010	
txdac_ana0	0x020	
txdac_ana1	0x024	
rxadc_ctrl0	0x100	
rxadcc_ctrl1	0x104	
rxadc_status	0x108	
rxadc_ana0	0x110	
rxadc_ana1	0x114	
rxadc_ana2	0x118	
rxadc_ana3	0x11c	
rxadc_ana4	0x120	

17.2.3.2 Audio ADC Register Description

rxadc_ctrl0

Offset Address: 0x100

Bits	Name	Access	Description	Reset
0	reg_rxadc_en	R/W		0x0
1	reg_i2s_tx_en	R/W		0x0
31:2	Reserved			

rxadcc_ctrl1

Offset Address: 0x104

Bits	Name	Access	Description	Reset
1:0	reg_rxadc_cic_opt	R/W	CIC decimation filter option 0: downsample ratio 64 1: downsample ratio 128 2: downsample ratio 256 3: downsample ratio 512	0x0
2	reg_rxadc_chn_swap	R/W	L/R input data channel swap	0x0
3	reg_rxadc_single	R/W	Only single channel supported, used when ANALOG in differential input mode	0x0
6:4	reg_rxadc_dcb_opt	R/W	DC blocking filter option 3'b000: bypass 3'b001: 1-2^(-8) 3'b010: 1-2^(-9) 3'b011: 1-2^(-10) 3'b100: 1-2^(-11) 3'b101: 1-2^(-12) other: bypass	0x5
7	Reserved			
8	reg_rxadc_igr_init	R/W	i2s keep silence when filter with initial value	0x0
9	reg_rxadc_clk_force_en	R/W	force clock enable	0x0
10	reg_rxadc_fsm_src_sel	R/W	FSM trigger source 0: adc0_vld 1: adc1_vld	0x0
31:11	Reserved			

rxadc_status

Offset Address: 0x108

Bits	Name	Access	Description	Reset
0	reg_rxadc_cic0_init_done	RO	cic_d_0 init done	
1	reg_rxadc_fir1_0_init_done	RO	fir1_0 init done	
2	reg_rxadc_fir2_0_init_done	RO	fir2_0 init done	
3	Reserved			
4	reg_rxadc_cic1_init_done	RO	cic_d_1 init done	
5	reg_rxadc_fir1_1_init_done	RO	fir1_1 init done	
6	reg_rxadc_fir2_1_init_done	RO	fir2_1 init done	
7	Reserved			
10:8	reg_rxadc_fsm	RO	ADC FSM state	
31:11	Reserved			

rxadc_ana0

Offset Address: 0x110

Bits	Name	Access	Description	Reset
12:0	reg_gstepl_rxpga	R/W	PGA feedback resistance selection 2dB/step default:20Kohm	0x0001
13	reg_g6dbl_rxpga	R/W	PGA input resistance selection 0:20Kohm 1:10Kohm	0x0
15:14	reg_gainl_rxadc	R/W	PGA input resistance selection 0:20Kohm(0dB) 1:10Kohm(6dB) 2: 5Kohm(12dB) 3: 2.5Kohm (18dB)	0x0
28:16	reg_gstepr_rxpga	R/W	PGA feedback resistance selection 2dB/step default:20Kohm	0x0001
29	reg_g6dbr_rxpga	R/W	PGA input resistance selection 0:20Kohm 1:10Kohm	0x0
31:30	reg_gainr_rxadc	R/W	PGA input resistance selection 0:20Kohm(0dB) 1:10Kohm(6dB) 2: 5Kohm(12dB) 3: 2.5Kohm (18dB)	0x0

rxadc_ana1

Offset Address: 0x114

Bits	Name	Access	Description	Reset
15:0	reg_gainl_status	RO	[12:0]: gstep1 [13]: g6dbl [15:14]: gainl	
31:16	reg_gainr_status	RO	[28:16]: gstepr [29]: g6dbr [31:30]: gainr	

rxadc_ana2

Offset Address: 0x118

Bits	Name	Access	Description	Reset
0	reg_mutel_rxpga	R/W	Enable pin of mute left channel, active high	0x0
1	reg_muter_rxpga	R/W	Enable pin of mute right channel, active high	0x0
15:2	Reserved			

Bits	Name	Access	Description	Reset
16	reg_diff_en_rxpga	R/W	Enable pin of differential mode(Left channel only. VIN/VINB=PAD_VINL/PAD_VINR	0x0
17	reg_tristate_rxpga	R/W	?	0x0
31:18	Reserved			

rxadc_ana3

Offset Address: 0x11c

Bits	Name	Access	Description	Reset
0	reg_addi_rxadc	R/W	ADC opamp current +50%, active High	0x0
1	reg_cksel_rxadc	R/W	PGA enable control input, active High	0x0
2	reg_en_asar_i_rxadc	R/W	Enable pin of L channel quantizer	0x1
3	reg_en_asar_q_rxadc	R/W	Enable pin of Q channel quantizer	0x1
5:4	reg_dem_type_rxadc	R/W	DEM TYPE 0:rotation 1: min cell switching 2/3: NA	0x1
7:6	Reserved			
11:8	reg_ctune_rxadc	R/W	RXADC integrator CFB selection 12*135fF + (8/4/2/1)*135fF	0xc
12	reg_en_dither_rxadc	R/W	Enable pin of dithering	0x1
13	reg_rstsdm_rxadc	R/W	Enable pin of resetting integrator	0x0
14	reg_en_vcmt_rxadc	R/W	?	0x0
15	Reserved			
17:16	reg_vldo0p9_rxadc	R/W	0.9V LDO output selection 00:0.85V 01:0.9V 10:0.95V 11:1.0V	0x1
19:18	reg_vldo12_rxadc	R/W	1.2V LDO output selection 00:1.1V 01:1.15V 10:1.2V 11:1.25V	0x1
21:20	reg_rnlvl_rxadc	R/W	Dither option	0x0
31:22	Reserved			

rxadc_ana4

Offset Address: 0x120

Bits	Name	Access	Description	Reset
0	reg_da_en_rxpga_status	RO	DA_EN_RXPGA status	
1	reg_da_end2us_rxpga_status	RO	DA_END2US_RXPGA status	
2	reg_da_en_rxadc_status	RO	DA_EN_RXADC status	
3	reg_da_en_audbias_status	RO	DA_EN_AUDBIAS status	
15:4	Reserved			
18:16	reg_ad_dol_rxadc	RO	Left channel 3-bit output	
19	Reserved			
22:20	reg_ad_dor_rxadc	RO	Right channel 3-bit output	
31:23	Reserved			

17.2.3.3 Audio DAC Register Description

txdac_ctrl0

Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	reg_txdac_en	R/W	audio dac enable	0x0
1	reg_i2s_rx_en	R/W	audio dac i2s output enable	0x0
31:2	Reserved			

txdac_ctrl1

Offset Address: 0x004

Bits	Name	Access	Description	Reset
1:0	reg_txdac_cic_opt	R/W	CIC decimation filter option 0: upsample ratio 64 1: upsample ratio 128 2: upsample ratio 256 3: upsample ratio 512	0x0
3:2	Reserved			
5:4	reg_txdac_dem_type	R/W	DEM TYPE 0:rotation 1: min cell switching 2/3: thermal code	0x1
7:6	Reserved			
8	reg_txdac_dsm_opt	R/W	DSM order option 0: order2 1: order1	0x0
9	reg_txdac_zcd_en	R/W	enable zero corssing function move from reg_spare0	0x0
10	reg_txdac_fsm_src_sel	R/W	FSM trigger source 0: i2s1_vld 1: i2s0_vld	0x1
11	Reserved			
14:12	reg_txdac_dither_opt	R/W	Dither option 0: disable others: weight = LSB/(2^(n-1))	0x0
31:15	Reserved			

txdac_status

Offset Address: 0x008

Bits	Name	Access	Description	Reset
2:0	reg_txdac_fsm	RO	DAC main fsm state	
3	Reserved			
6:4	reg_txdac_afe_fsm	RO	DAC AFE fsm state	
7	Reserved			
16:8	reg_txdac_gain0	RO	DAC L-channel gain	
19:17	Reserved			
28:20	reg_txdac_gain1	RO	DAC R-channel gain	
31:29	Reserved			

txdac_afe0

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
5:0	reg_txdac_init_dly_cnt	R/W	DAC AFE initialize delay, min>2us	0x28
7:6	Reserved			
15:8	reg_txdac_value_tick	R/W	DAC AFE tick value for initialize/stop value ramp	0x0
27:16	reg_txdac_gain_tick	R/W	DAC AFE tick value for gain ramp	0x800
31:28	Reserved			

txdac_afe1

Offset Address: 0x010

Bits	Name	Access	Description	Reset
8:0	reg_txdac_gain_ub_0	R/W	channel 0, L-channel, DAC gain upper bound 0x00: gain=0 0xff: gain=1-2^(-8), default value to prevent DSM overflow 0x100: gain=1	0xff
15:9	Reserved			
24:16	reg_txdac_gain_ub_1	R/W	channel 1, R-channel, DAC gain upper bound 0x00: gain=0 0xff: gain=1-2^(-8), default value to prevent DSM overflow 0x100: gain=1	0xff
27:25	Reserved			
28	reg_txdac_ramp_bp	R/W	bypass initial ramp procedure	0x0
31:29	Reserved			

txdac_ana0

Offset Address: 0x020

Bits	Name	Access	Description	Reset
0	reg_addi_txdac	R/W	na	0x0
3:1	Reserved			
5:4	reg_tsel_txdac	R/W	2'b00: NA 2'b01: VCM 2'b10: VDD15A	0x0

Bits	Name	Access	Description	Reset
			2'b11: undefined	
7:6	Reserved			
9:8	reg_vsel_txdac	R/W	1.5V LDO output selection 2'b00:1.35V 2'b01:1.4V 2'b10:1.45V 2'b11:1.5V	0x3
31:10	Reserved			

txdac_ana1

Offset Address: 0x024

Bits	Name	Access	Description	Reset
0	reg_da_en_txdac_ow_val	R/W	DA_EN_TXDAC overwrite value	0x0
1	reg_da_end2us_txdac_ow_val	R/W	DA_END2US_TXDAC overwrite value	0x0
15:2	Reserved			
16	reg_da_en_txdac_ow_en	R/W	DA_EN_TXDAC overwrite enable	0x0
17	reg_da_end2us_txdac_ow_en	R/W	DA_END2US_TXDAC overwrite enable	0x0
31:18	Reserved			

18 peripheral equipment

18.1 I2C

18.1.1 summarize

The chip is configured with 5 I2C controllers, which can be individually configured as Master/Slave. IO configuration should refer to Function pin mux for configuration.

18.1.2 Functional Description

The I2C controller has the following functional characteristics:

- Supports both standard (7bit) and extended (10bit)
- The transmission rate supports standard mode (100kbit/s) and fast mode (400kbit/s)
- General Call and Start Byte functions are supported.
- CBUS devices are not supported.
- DMA operation is supported.
- Supports 64 x 8bit TX FIFO and 64 x 8bit RX FIFO.

18.1.3 functional block diagram

Figure 12-1 shows the functional block diagram of the I2C module.
The CPU selects I2C modes and timings through the APB bus configuration registers, writes TXFIFO, reads
RXFIFO and trigger FSM to send and receive SDA/SCL related IO signals. system DMA can also be used with I2C DMA_IF and write TXFIFO and read RXFIFO to send and receive I2C signals through APB bus.

algorithmic technology

Figure *18-1 I2C* Function Block Diagram

18.1.4 I2C Protocol Timing

The chip I2C supports the general standard I2C protocol timing as shown in Figure 12-2.

Figure *18-2 I2C* Protocol Timing

18.1.5 working method

18.1.5.1 Configuring I2C Clock and Timing Parameters

1. CLK_DIV CRG register section, configuring clk_byp_0_31 selects IIC_CLK as either a 25MHz preset clock source or a 100MHz clock source.
2. Configuring the relevant timing configuration requires the module to be in an unenabled state. You need to set IC_ENABLE to 0 and query the IC_ENABLE_STATUS[0] is determined to be 0.
3. Refer to Table 18-1 and configure the I2C Timing Counter Register by I2C Clock Selection.

Table *18-1* Relationship between *I2C* Clock Selection and Related Register Configurations

processor register	25M IIC_CLK	100M IIC_CLK	descriptive
IC_SS_SCL_HCNT	115	460	SCL High Time Count for Standard Speed Mode
IC_SS_SCL_LCNT	135	540	SCL Low Time Count for Standard Speed Mode
IC_FS_SCL_HCNT	21	90	SCL High Time Count for Fast Speed Mode
IC_FS_SCL_LCNT	42	160	SCL Low Time Count for Fast Speed Mode
IC_SDA_HOLD	1	1	SDA Hold Time Count, Negative Edge to SCL
IC_SDA_SETUP	6	25	SDA set time count, positive edge relative to SCL

processor register	25M IIC_CLK	100M IIC_CLK	descriptive
IC_FS_SPKLEN	2	5	I2C Burr Suppression Time Count

18.1.5.2 Data transfer in non-DMA mode

Figure *18-3* Data Transfer Software Flow in *I2C Non-DMA* Mode

algorithmic technology

18.1.5.3 Data transfer in DMA mode

Figure *18-4* Data Transfer Software Flow in *I2C DMA* Method

18.1.6 I2C Register Overview

Table *18-2* 6-Group *I2C* Module Base Addresses for Chip

GPIO Module	base address
I2C0	0x04000000

GPIO Module	base address
I2C1	0x04010000
I2C2	0x04020000
I2C3	0x04030000
I2C4	0x04040000
RTCSYS_I2C	0x0502B000

Table *18-3* Overview of *I2C* Registers for the Chip

Name	Address Offset	Description
IC_CON	0x000	I2C Control
IC_TAR	0x004	I2C Target Address
IC_SAR	0x008	I2C Slave Address
IC_DATA_CMD	0x010	I2C Rx/Tx Data Buffer and Command
IC_SS_SCL_HCNT	0x014	Standard speed I2C Clock SCL High Count
IC_SS_SCL_LCNT	0x018	Standard speed I2C Clock SCL Low Count
IC_FS_SCL_HCNT	0x01c	Fast speed I2C Clock SCL High Count
IC_FS_SCL_LCNT	0x020	Fast speed I2C Clock SCL Low Count
IC_INTR_STAT	0x02c	I2C Interrupt Status
IC_INTR_MASK	0x030	I2C Interrupt Mask
IC_RAW_INTR_STAT	0x034	I2C Raw Interrupt Status
IC_RX_TL	0x038	I2C Receive FIFO Threshold
IC_TX_TL	0x03c	I2C Transmit FIFO Threshold
IC_CLR_INTR	0x040	Clear Combined and Individual Interrupts
IC_CLR_RX_UNDER	0x044	Clear RX_UNDER Interrupt
IC_CLR_RX_OVER	0x048	Clear RX_OVER Interrupt
IC_CLR_TX_OVER	0x04c	Clear TX_OVER Interrupt
IC_CLR_RD_REQ	0x050	Clear RD_REQ Interrupt
IC_CLR_TX_ABRT	0x054	Clear TX_ABRT Interrupt
IC_CLR_RX_DONE	0x058	Clear RX_DONE Interrupt
IC_CLR_ACTIVITY	0x05c	Clear ACTIVITY Interrupt
IC_CLR_STOP_DET	0x060	Clear STOP_DET Interrupt
IC_CLR_START_DET	0x064	Clear START_DET Interrupt
IC_CLR_GEN_CALL	0x068	Clear GEN_CALL Interrupt
IC_ENABLE	0x06c	I2C Enable
IC_STATUS	0x070	I2C Status register
IC_TXFLR	0x074	Transmit FIFO Level Register
IC_RXFLR	0x078	Receive FIFO Level Register
IC_SDA_HOLD	0x07c	SDA hold time length register
IC_TX_ABRT_SOURCE	0x080	I2C Transmit Abort Status Register
IC_SLV_DATA_NACK_ONLY	0x084	Generate SLV_DATA_NACK Register
IC_DMA_CR	0x088	DMA Control Register for transmit and receive handshaking interface
IC_DMA_TDLR	0x08c	DMA Transmit Data Level
IC_DMA_RDLR	0x090	DMA Receive Data Level
IC_SDA_SETUP	0x094	I2C SDA Setup Register
IC_ACK_GENERAL_CALL	0x098	I2C ACK General Call Register
IC_ENABLE_STATUS	0x09c	I2C Enable Status Register
IC_FS_SPKLEN	0x0a0	ISS and FS spike suppression limit
IC_HS_SPKLEN	0x0a4	HS spike suppression limit

18.1.7 I2C Register Description

IC_CON

Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	MASTER_MODE	R/W	enable master mode	0x1
2:1	SPEED	R/W	1: standard mode (~100 kbit/s) 2: fast mode (~400 kbit/s)	0x3
3	IC_10BITADDR_SLAVE	R/W	enable 10bit slave address mode	0x1
4	IC_10BITADDR_MASTER	R/W	enable 10bit master address mode	0x1
5	IC_RESTART_EN	R/W	enable I2C master to be able generate restart	0x1
6	IC_SLAVE_DISABLE	R/W	0: slave is enabled 1: slave is disabled	0x1
31:7	Reserved			

IC_TAR

Offset Address: 0x004

Bits	Name	Access	Description	Reset
9:0	IC_TAR	R/W	I2C Target Address Register	0x55
10	GC_OR_START	R/W	If bit SPECIAL is set to 1, then this bit indicates whether a General Call or START byte command is being used. 0: general call 1: start byte	0x0
11	SPECIAL	R/W	Used to issue General Call or START BYTE	0x0
31:12	Reserved			

IC_SAR

Offset Address: 0x008

Bits	Name	Access	Description	Reset
9:0	IC_SAR	R/W	I2C Slave Address Register	0x55
31:10	Reserved			

IC_DATA_CMD

Offset Address: 0x010

Bits	Name	Access	Description	Reset
7:0	DAT	R/W	Transmitted or received data port	0x0
8	CMD	R/W	0: Write 1: Read	0x0
9	STOP	R/W	issue stop	0x0
10	RESTART	R/W	issue restart	0x0
31:11	Reserved			

IC_SS_SCL_HCNT

Offset Address: 0x014

Bits	Name	Access	Description	Reset
15:0	IC_SS_SCL_HCNT	R/W	Standard Speed I2C Clock SCL High Count Register	0x0190
31:16	Reserved			

IC_SS_SCL_LCNT

Offset Address: 0x018

Bits	Name	Access	Description	Reset
15:0	IC_SS_SCL_LCNT	R/W	Standard Speed I2C Clock SCL Low Count Register	0x01d6
31:16	Reserved			

IC_FS_SCL_HCNT

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
15:0	IC_FS_SCL_HCNT	R/W	Fast Speed I2C Clock SCL High Count Register	0x003C
31:16	Reserved			

IC_FS_SCL_LCNT

Offset Address: 0x020

Bits	Name	Access	Description	Reset
15:0	IC_FS_SCL_LCNT	R/W	Fast Speed I2C Clock SCL Low Count Register	0x0082
31:16	Reserved			

IC_INTR_STAT

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
0	R_RX_UNDER	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
1	R_RX_OVER	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
2	R_RX_FULL	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
3	R_TX_OVER	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
4	R_TX_EMPTY	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
5	R_RD_REQ	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
6	R_TX_ABRT	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
7	R_RX_DONE	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
8	R_ACTIVITY	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
9	R_STOP_DET	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
10	R_START_DET	RO	corresponding masked interrupt staus.	

Bits	Name	Access	Description	Reset
			please reference I2C Raw Interrupt Status	
11	R_GEN_CALL	RO	corresponding masked interrupt staus, please reference I2C Raw Interrupt Status	
31:12	Reserved			

IC_INTR_MASK

Offset Address: 0x030

Bits	Name	Access	Description	Reset
0	M_RX_UNDER	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x1
1	M_RX_OVER	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x1
2	M_RX_FULL	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x1
3	M_TX_OVER	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x1
4	M_TX_EMPTY	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x1
5	M_RD_REQ	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x1
6	M_TX_ABRT	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x1
7	M_RX_DONE	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x1
8	M_ACTIVITY	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x0
9	M_STOP_DET	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x0
10	M_START_DET	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x0
11	M_GEN_CALL	R/W	corresponding interrupt staus mask, please reference I2C Raw Interrupt Status	0x1

31:12	Reserved			

IC_RAW_INTR_STAT

Offset Address: 0x034

Bits	Name	Access	Description	Reset
0	IST_RX_UNDER	RO	when receive buffer is empty by reading from the IC_DATA_CMD register	
1	IST_RX_OVER	RO	receive buffer is oveflow (64Bytes)	
2	IST_RX_FULL	RO	receive buffer reaches or goes above	

Bits	Name	Access	Description	Reset
			the RX_TL threshold	
3	IST_TX_OVER	RO	transmit buffer is oveflow (64Bytes)	
4	IST_TX_EMPTY	RO	transmit buffer is at or below the TX_TL threshold	
5	IST_RD_REQ	RO	In slave mode, I2C hold SCL and wait for the response from processor	
6	IST_TX_ABRT	RO	In master or slave mode, when The transmitter is unable to complete the action	
7	IST_RX_DONE	RO	In slave-transmitter mode, a NACK is received	
8	IST_ACTIVITY	RO	I2C activity is detected	
9	IST_STOP_DET	RO	STOP occurred	
10	IST_START_DET	RO	START or RESTART occurred	
11	IST_GEN_CALL	RO	General Call address is received	
31:12	Reserved			

IC_RX_TL

Offset Address: 0x038

Bits	Name	Access	Description	Reset
7:0	RX_TL	R/W	Receive FIFO Threshold Level	0x0
31:8	Reserved			

IC_TX_TL

Offset Address: 0x03c

Bits	Name	Access	Description	Reset
7:0	TX_TL	R/W	Transmit FIFO Threshold Level	0x0
31:8	Reserved			

IC_CLR_INTR

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	CLR_INTR	RO	read to clear corresponding all raw staus	
31:1	Reserved			

IC_CLR_RX_UNDER

Offset Address: 0x044

Bits	Name	Access	Description	Reset
0	CLR_RX_UNDER	RO	read to clear corresponding interupt raw staus, please reference I2C Raw Interrupt Status	
31:1	Reserved			

IC_CLR_RX_OVER

Offset Address: 0x048

Bits	Name	Access	Description	Reset
0	CLR_RX_OVER	RO	read to clear corresponding interupt raw staus, please reference I2C Raw Interrupt Status	
31:1	Reserved			

IC_CLR_TX_OVER

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
0	CLR_TX_OVER	RO	read to clear corresponding interupt raw staus, please reference I2C Raw Interrupt Status	
31:1	Reserved			

IC_CLR_RD_REQ

Offset Address: 0x050

Bits	Name	Access	Description	Reset
0	CLR_RD_REQ	RO	read to clear corresponding interupt raw staus, please reference I2C Raw Interrupt Status	
31:1	Reserved			

IC_CLR_TX_ABRT

Offset Address: 0x054

Bits	Name	Access	Description	Reset
0	CLR_TX_ABRT	RO	read to clear corresponding interupt raw staus, please reference I2C Raw Interrupt Status	
31:1	Reserved			

IC_CLR_RX_DONE

Offset Address: 0x058

Bits	Name	Access	Description	Reset
0	CLR_RX_DONE	RO	read to clear corresponding interupt raw staus, please reference I2C Raw Interrupt Status	
31:1	Reserved			

IC_CLR_ACTIVITY

Offset Address: 0x05c

Bits	Name	Access	Description	Reset
0	CLR_ACTIVITY	RO	read to clear corresponding interupt raw staus, please reference I2C Raw Interrupt Status	
31:1	Reserved			

IC_CLR_STOP_DET

Offset Address: 0x060

Bits	Name	Access	Description	Reset
0	CLR_STOP_DET	RO	read to clear corresponding interupt raw staus, please reference I2C Raw Interrupt Status	
31:1	Reserved			

IC_CLR_START_DET

Offset Address: 0x064

Bits	Name	Access	Description	Reset
0	CLR_START_DET	RO	read to clear corresponding interupt raw staus, please reference I2C Raw	

Bits	Name	Access	Description	Reset
			Interrupt Status	
31:1	Reserved			

IC_CLR_GEN_CALL

Offset Address: 0x068

Bits	Name	Access	Description	Reset
0	CLR_GEN_CALL	RO	read to clear corresponding interupt raw staus, please reference I2C Raw Interrupt Status	
31:1	Reserved			

IC_ENABLE

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
0	ENABLE	R/W	Enables I2C controller	0x0
31:1	Reserved			

IC_STATUS

Offset Address: 0x070

Bits	Name	Access	Description	Reset
0	ST_ACTIVITY	RO	I2C Activity Status.	
1	ST_TFNF	RO	Transmit FIFO Not Full	
2	ST_TFE	RO	Transmit FIFO Completely Empty	
3	ST_RFNE	RO	Receive FIFO Not Empty	
4	ST_RFF	RO	Receive FIFO Completely Full	
5	ST_MST_ACTIVITY	RO	Master FSM Activity Status	
6	ST_SLV_ACTIVITY	RO	Slave FSM Activity Status	
31:7	Reserved			

IC_TXFLR

Offset Address: 0x074

Bits	Name	Access	Description	Reset
6:0	TXFLR	RO	I2C Transmit FIFO Level	
31:7	Reserved			

IC_RXFLR

Offset Address: 0x078

Bits	Name	Access	Description	Reset
6:0	RXFLR	RO	I2C Receive FIFO Level Register	
31:7	Reserved			

IC_SDA_HOLD

Offset Address: 0x07c

Bits	Name	Access	Description	Reset
15:0	IC_SDA_HOLD	R/W	Sets the required SDA hold time in units of IP clock.	0x1
31:16	Reserved			

IC_TX_ABRT_SOURCE

Offset Address: 0x080

Bits	Name	Access	Description	Reset
15:0	TX_ABRT_SOURCE	RO	I2C Transmit Abort Source Register	
31:16	Reserved			

IC_SLV_DATA_NACK_ONLY

Offset Address: 0x084

Bits	Name	Access	Description	Reset
0	NACK	R/W	generate a NACK in slave-receiver mode	0x0
31:1	Reserved			

IC_DMA_CR

Offset Address: 0x088

Bits	Name	Access	Description	Reset
0	RDMAE	R/W	Receive DMA Enable	0x0
1	TDMAE	R/W	Transmit DMA Enable	0x0
31:2	Reserved			

IC_DMA_TDLR

Offset Address: 0x08c

Bits	Name	Access	Description	Reset
5:0	DMATDL	R/W	the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value	0x0
31:6	Reserved			

IC_DMA_RDLR

Offset Address: 0x090

Bits	Name	Access	Description	Reset
5:0	DMARDL	R/W	dma_rx_req is generated when the number of valid data entries in the the receive FIFO is equal to or more than this field value + 1	0x0
31:6	Reserved			

IC_SDA_SETUP

Offset Address: 0x094

Bits	Name	Access	Description	Reset
0	SDA_SETUP	R/W	SDA Setup time config register	0x64
31:1	Reserved			

IC_ACK_GENERAL_CALL

Offset Address: 0x098

Bits	Name	Access	Description	Reset
0	ACK_GEN_CALL	R/W	When set to 1, DW_apb_i2c responds with an ACK when it receives a General Call. When set to 0, the IP does not generate General Call interrupts	0x1
31:1	Reserved			

IC_ENABLE_STATUS

Offset Address: 0x09c

Bits	Name	Access	Description	Reset
0	IC_EN	RO	I2C Enable Status Register	
1	SLV_DISABLED_WHILE_BUSY	RO	Slave Disabled While Busy (Transmit, Receive)	
2	SLV_RX_DATA_LOST	RO	Slave Received Data Lost.	
31:3	Reserved			

IC_FS_SPKLEN

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
7:0	IC_FS_SPKLEN	R/W	I2C SS and FS Spike Suppression Limit register	0x5
31:8	Reserved			

IC_HS_SPKLEN

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
7:0	IC_HS_SPKLEN	R/W	I2C HS Spike Suppression Limit Register	0x1
31:8	Reserved			

18.2 UART

18.2.1 summarize

UART (Universal Asynchronous Receiver Transmitter) is a non-synchronous serial communication interface, the main function of which is to transfer the data from peripheral devices into the internal bus after serial-parallel conversion, and to output the data to external devices after parallel-serial conversion. The main function of UART is to interface with the UART of the external chip to realize the communication between the two chips.

The chip provides 5 UART controllers, which are summarized in the following table. For IO related configuration, please refer to the section of pin information description.

controllers	Support Mode	IO Pin
UART0	Two-wire UART	UART0_TX/UART0_RX
UART1	Two/four wire UART	uart1_tx/uart1_rx/uart1_cts/uart1_rts
		xgpioa[20]/ xgpioa[21]/ xgpioa[22]/ xgpioa[26]

UART2	Two/four wire UART	uart2_tx/uart2_rx/uart2_cts/uart2_rts
		xgpioa[20]/ xgpioa[21]/ xgpioa[22]/ xgpioa[26]
		IIC2_SDA/IIC2_SCL

controllers	Support Mode	IO Pin
UART3	Two/four wire UART	spi0_cs_x/spi0_sck/spi0_sdi/spi0_sdo
		vi_data22/vi_data21/vi_data24/vi_data23
		PWM3/PWM2
UART4	Two-wire UART	xgpioa[22]/ xgpioa[26]
		UART1_RTS/UART1_CTS

18.2.2 specificities

The UART module has the following features:

- Supports 64 x 8bit transmit FIFO and 64 x 8bit receive FIFO.
- Supports programmable bit widths for data bits and stop bits. Data bits can be programmed to 5/6/7/8 bits;
- The stop bit can be programmed as 1bit, 1.5bit or 2bit.
- Supports odd or even checksum method or no checksum.
- Supports transmission rate programmability.
- Supports receive FIFO interrupt, transmit FIFO interrupt, and error interrupt.
- Supports initial interrupt status query and post-mask interrupt status query.
- DMA operation is supported.

18.2.3 Functional Description

18.2.3.1 application block diagram

UART is a universal point-to-point physical layer transmission protocol that can be used to interface with a variety of systems, including PCs and various peripheral chips, and can be used as a chip-to-chip interface.

algorithmic technology

Figure *18-5 UART* Application Block Diagram

18.2.3.2 Functional principle

- Baud rate (buad rate)

Since the UART interface does not have a reference clock, it is an asynchronous transmission method that requires both parties to use the same transmission speed, i.e., the same baud rate, in order to communicate. If there is an error, the error rate must be small enough to prevent miscommunication.1 The bit rate is called the baud rate. Typical baud rates are 300, 1200, 2400, 9600, 19200, 38400, 115200 bps and so on.

- Frame structure

The UART transmission data structure is organized in frames. The frame structure contains a start signal, an information signal, a parity bit, and an end signal.

Figure *18-6 UART* Transmission Data Structure

- Start signal (start bit)

The start signal signals the beginning of a frame. The very beginning of a frame transmission is initiated by sending a low level on TXD
Signal bit. On RXD, if a low signal bit is received in the idle state, it is judged to be the start of a transmission.

- Data signal (data bit)

The data bit width can be adjusted to 5/6/7/8 bit data bit widths depending on the application requirements. Typically 8 bits of data bit width.

- Parity bit

The parity bit is a 1-bit error correction signal. The parity bit of UART has odd parity check, even parity and fixed parity bit, and also supports enable and disable of parity bit, please refer to LCR register for detailed description.

- End signal (stop bit)

The end signal is the stop bit of the frame and supports 1 bit, 1.5 bit and 2 bit stop bits. The end signal for sending a frame is to send TXD high to complete the transmission and enter the idle state. To receive a frame, the end signal is received after the parity bit has been counted.

18.2.4 working method

18.2.4.1 baud rate configuration

- UART Working Clock (UART_SCLK) Configuration

You can refer to CLK_DIV CRG register description to configure clk_sel_0_9~clk_sel_0_13 to select uart0~uart4 operating clock. The preset is 1:XTAL 25MHz, and the configuration is 0 to select UART PLL divided clock source.

The source is preset to 187.5 MHz, and the divider register div_clk_187p5m can be configured to adjust the PLL divider clock to 1500/N MHz up to 187.5 MHz if required.

- UART Baud Rate Configuration

DLL and DLH are baud rate divider control registers inside the UART controller, DLH is the high 8 bits and DLL is the low 8 bits. Before configuring DLH and DLL, LCR[7] must be configured to 1. At this time, you can configure registers RBR_THR_DLL (DLL) and IER_DLH (DLH).

After the configuration is completed, the baud rate is set, and the formula is:

$$Baud\ rate = \frac{UART_SCLK}{16 * (256 * DLH + DLL)}$$

- Using the UART SCLK 25MHz example and configuring the 115200 baud rate, the math is:

$$(256 * DLH + DLL) = \frac{25M}{16 * 115200} = 13.5$$

If you choose to configure DLL as 14 and DLH as 0, the actual baud rate is:

$$Baud\ rate = \frac{25M}{16 * 14} = 111607$$

The one-bit time error is:

$$Bit\ Error = \frac{(115200 - 114286)}{115200} = 3.12\ percent$$

Cumulative one frame time error is:

$$Frame\ Error = 3.12\% * 10 = 31.2\ percent$$

- Using UART SCLK 187.5MHz as an example and configuring 115200 baud rate, the math is:

$$(256 * DLH + DLL) = \frac{187.5M}{16 * 115200} = 101.7$$

If you choose to configure DLL as 102 and DLH as 0, the actual baud rate is:

$$Baud\ rate = \frac{187.5M}{16 * 102} = 114890$$

The one-bit time error is:

$$Bit\ Error = \frac{(115200 - 114890)}{115200} = 0.27\ percent$$

Cumulative one frame time error is:

$$Frame\ Error = 3.12\% * 10 = 2.7\ percent$$

18.2.4.2 Flowchart of data sending

Figure *18-7 UART* Data Transmission Flowchart

18.2.4.3 Data Receiving Flowchart

Figure *18-8 UART* Data Receiving Flowchart

18.2.4.4 Data transfer in interrupt or query mode

18.2.4.4.1 initialization step

1. Write 1 to LCR[7]. enable configuration Divisor Latch Access

2. Write the corresponding configuration values to RBR_THR_DLL, IER_DLH registers to configure the transmission baud rate.
3. Write 0 to LCR[7].
4. Configure the LCR to set the appropriate UART operating mode
5. Configure the FCR to set the corresponding transmit and receive FIFO thresholds.
6. If the interrupt method is used, set IER to enable the corresponding interrupt signal;

18.2.4.4.2 data transmission

1. In the state where LCR[7] is 0, write the transmit data to RBR_THR_DLL (Transmit Holding Register) to start data transmission.
2. If the query method is used, the TX_FIFO status is detected by reading USR[1] (Transmit FIFO not full) and TFL (Transmit FIFO Level) and the TX_FIFO status is used to decide whether to continue to send a query to the TX_FIFO.
   Write data in RBR_THR_DLL;
3. If the interrupt method is used, it is detected according to the corresponding interrupt status bit; decide whether to continue to the Write data in RBR_THR_DLL.
4. Determine whether the UART has completed sending all data by detecting USR[2] (Transmit FIFO Empty).

18.2.4.4.3 data reception

1. If the query method is used, the RX_FIFO status is detected by reading USR[3] (Receive FIFO Not Empty) and RFL (Receive FIFO Level), and it is decided whether or not to read based on the status of the RX_FIFO.
   RBR_THR_DLL (Receive Buffer Register) to get the data.
2. If the interrupt method is used, the decision to read or not is based on the detection of the corresponding interrupt status bit.
   RBR_THR_DLL (Receive Buffer Register) to get the data.

18.2.4.5 Data transfer in DMA mode

18.2.4.5.1 initialization step

1. Write 1 to LCR[7]. enable configuration Divisor Latch Access
2. Write the corresponding configuration values to RBR_THR_DLL, IER_DLH registers to configure the transmission baud rate.
3. Write 0 to LCR[7].
4. Configure the LCR to set the appropriate UART operating mode
5. Configure the FCR to set the corresponding transmit and receive FIFO thresholds.
6. Turn off ETBEI/ERBFI in the IER;

18.2.4.5.2 data transmission

1. Configure the system DMA channel mapping. Configure the selected UART controller TX/RX request line number to the corresponding system DMA channel by referring to the system DMA channel mapping. For example, if UART0 TX is configured for system DMA channel 3, sdma_dma_ch_remap0[29:24]=9. To complete the configuration, you need to configure update_dma_remp_0_3 to make the configuration take effect.
2. Configure the system DMA data channel, including parameters such as data transfer source and destination addresses, number of data transfers, and transfer type. See the DMA Controller section for specific configuration.
3. The system DMA interrupt is reported to determine whether the data has been sent or not.

18.2.4.5.3 data reception

1. Configure the system DMA channel mapping. Configure the selected UART controller TX/RX request line number to the corresponding system DMA channel by referring to the DMA channel mapping. For example, if UART0 RX is configured as system DMA channel 1, then set sdma_dma_ch_remap0[13:8] = 8. To complete the configuration, you need to configure update_dma_remp_0_3 to make the configuration take effect.
2. Configure the system DMA data channel, including parameters such as data transfer source and destination addresses, data receiving area address, number of data transfers, and transfer type. See the DMA Controller section for specific configuration.
3. The system DMA interrupt is reported to determine whether the data reception is completed or not.

18.2.5 UART Register Overview

Table *18-4* 6-Group *UART* Module Base Addresses for Chip

GPIO Module	base address
UART0	0x04140000

UART1	0x04150000
UART2	0x04160000
UART3	0x04170000
UART4	0x041C0000
RTCSYS_UART	0x05022000

Table *18-5* Overview of the Chip's *UART* Registers

Name	Address Offset	Description
RBR_THR_DLL	0x000	Receive Buffer,Transmit Holding or Divisor Latch Low byte register
IER_DLH	0x004	Interrupt Enable or Divisor Latch high byte Register
FCR_IIR	0x008	FIFO Control or Interrupt Identification Register
LCR	0x00c	Line Control Register
MCR	0x010	Modem Control Register

Name	Address Offset	Description
LSR	0x014	Line Status Register
MSR	0x018	Modem Status Register
LPDLL	0x020	Low Power Divisor Latch (Low) Register
LPDLH	0x024	Low Power Divisor Latch (High) Register
SRBR_STHR	0x030	Shadow Receive/Trasnmit Buffer Register
FAR	0x070	FIFO Access Register
TFR	0x074	Transmit FIFO Read
RFW	0x078	Receive FIFO Write
USR	0x07c	UART Status Register
TFL	0x080	Transmit FIFO Level
RFL	0x084	Receive FIFO Level
SRR	0x088	Software Reset Register
SRTS	0x08c	Shadow Request to Send
SBCR	0x090	Shadow Break Control Register
SDMAM	0x094	Shadow DMA Mode
SFE	0x098	Shadow FIFO Enable
SRT	0x09c	Shadow RCVR Trigger
STET	0x0a0	Shadow TX Empty Trigger
HTX	0x0a4	Halt TX
DMASA	0x0a8	DMA Software Acknowledge

18.2.6 UART Register Description

RBR_THR_DLL

Offset Address: 0x000

Bits	Name	Access	Description	Reset
7:0	RBR_THR_DLL	R/W	LCR[7] bit = 0 : (R) Receive Buffer Register ,Data byte received on the serial input port (W)Transmit Holding Register,Data to be transmitted on the serial output port LCR[7] bit = 1 : Lower 8 bits of a 16-bit Divisor Latch register that contains the baud rate divisor for the UART	0x0
31:8	Reserved			

IER_DLH

Offset Address: 0x004

Bits	Name	Access	Description	Reset
7:0	IER_DLH	R/W	LCR[7] bit = 0 : IER[0] : Enable Received Data Available Interrupt. IER[1] : Enable Transmit Holding Register Empty Interrupt. IER[2] : Enable Receiver Line Status Interrupt. IER[3] : Enable Modem Status Interrupt. IER[7] : Programmable THRE	0x0

			Interrupt Mode Enable	

Bits	Name	Access	Description	Reset
			LCR[7] bit = 1 : Upper 8-bits of a 16-bit Divisor Latch register that contains the baud rate divisor for the UART.	
31:8	Reserved			

FCR_IIR

Offset Address: 0x008

Bits	Name	Access	Description	Reset
7:0	FCR_IIR	R/W	(R) interrupt Identification Register [3:0] Interrupt ID 0000 : modem status 0001 : no interrupt pending 0010 : THR empty 0100 : received data available 0110 : receiver line status 0111 : busy detect 1100 : character timeout [7:6] FIFOs Enabled 00 - disabled 11 - enable (W) FIFO Control Register [0] FIFO Enable [1] RCVR FIFO Reset [2] XMIT FIFO Reset [3] DMA Mode 0 - mode 0, single DMA data transfers at a time 1 - mode 1, multi DMA data transfers are made continuously [5:4] TX Empty 00 - FIFO empty 01 - 2 characters in the FIFO 10 - FIFO ¼ full 11 - FIFO ½ full [7:6] RCVR Trigger 00 - 1 character in the FIFO 01 - FIFO ¼ full 10 - FIFO ½ full 11 - FIFO 2 less than full	0x1
31:8	Reserved			

LCR

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
7:0	LCR	R/W	Line Control Register [1:0] Data Length Select. (00:5 bits,01:6 bits,10:7 bits,11:8 bits) [2] Number of stop bits. (0:1 stop bit,1:1.5 stop bits when Data Length Select is 0, else 2 stop bits) [3] Parity Enable [4] Even Parity Select [5] Stick Parity [6] Break Control Bit	0x0

			[7] Divisor Latch Access Bit	

Bits	Name	Access	Description	Reset
31:8	Reserved			

MCR

Offset Address: 0x010

Bits	Name	Access	Description	Reset
7:0	MCR	R/W	Modem Control Register [0] reserved [1] Request to Send. This is used to directly control the Request to Send (rts_n) output [2] reserved [3] reserved [4] reserved [5] Auto Flow Control Enable. [6] reserved	0x0
31:8	Reserved			

LSR

Offset Address: 0x014

Bits	Name	Access	Description	Reset
7:0	LSR	RO	Line Status Register [0] Data Ready bit. there is at least one character in the RBR or the receiver FIFO. [1] This is used to indicate the occurrence of an overrun error. [2] Parity Error bit. [3] Framing Error bit... [4] Break Interrupt bit. [5] Transmit Holding Register Empty bit. [6] Transmitter Empty bit. [7] Receiver FIFO Error bit.	
31:8	Reserved			

MSR

Offset Address: 0x018

Bits	Name	Access	Description	Reset
7:0	MSR	RO	Modem Status Register [0] Delta Clear to Send. [1] reserved [2] reserved [3] reserved [4] CTS [5] reserved [6] reserved [7] reserved	
31:8	Reserved			

LPDLL

Offset Address: 0x020

Bits	Name	Access	Description	Reset
7:0	LPDLL	R/W	LCR[7] bit = 1 : Low Power Divisor Latch	0x0

Bits	Name	Access	Description	Reset
			(Low) Register	
31:8	Reserved			

LPDLH

Offset Address: 0x024

Bits	Name	Access	Description	Reset
7:0	LPDLH	R/W	LCR[7] bit = 1 : Low Power Divisor Latch (High) Register	0x0
31:8	Reserved			

SRBR_STHR

Offset Address: 0x030

Bits	Name	Access	Description	Reset
7:0	SRBR_STHR	R/W	LCR[7] bit = 0 : (R) Shadow Receive Buffer Register (W) Shadow Transmit Holding Register	0x0
31:8	Reserved			

FAR

Offset Address: 0x070

Bits	Name	Access	Description	Reset
0	FAR	R/W	FIFO Access Register,This register is use to enable a FIFO access mode for testing	0x0
31:1	Reserved			

TFR

Offset Address: 0x074

Bits	Name	Access	Description	Reset
7:0	TFR	R/W	Transmit FIFO Read. these bits are only valid when FIFO access mode is enabled	0x0
31:8	Reserved			

RFW

Offset Address: 0x078

Bits	Name	Access	Description	Reset
9:0	RFW	R/W	Receive FIFO Write. These bits are only valid when FIFO access mode is enabled. [7:0] Receive FIFO Write Data. [8] Receive FIFO Parity Error. [9] Receive FIFO Framing Error.	0x0
31:10	Reserved			

USR

Offset Address: 0x07c

Bits	Name	Access	Description	Reset
4:0	USR	RO	UART Status Register [0] UART Busy. [1] Transmit FIFO Not Full. [2] Transmit FIFO Empty. [3] Receive FIFO Not Empty. [4] Receive FIFO Full.	

Bits	Name	Access	Description	Reset
31:5	Reserved			

TFL

Offset Address: 0x080

Bits	Name	Access	Description	Reset
5:0	TFL	RO	Transmit FIFO Level. This is indicates the number of data entries in the transmit FIFO.	
31:6	Reserved			

RFL

Offset Address: 0x084

Bits	Name	Access	Description	Reset
5:0	RFL	RO	Receive FIFO Level. This is indicates the number of data entries in the receive FIFO.	
31:6	Reserved			

SRR

Offset Address: 0x088

Bits	Name	Access	Description	Reset
2:0	SRR	R/W	Software Reset Register [0] UART Reset. [1] RCVR FIFO Reset. [2] XMIT FIFO Reset.	0x0
31:3	Reserved			

SRTS

Offset Address: 0x08c

Bits	Name	Access	Description	Reset
0	SRTS	R/W	Shadow Request to Send. This is a shadow register for the RTS bit (MCR[1])	0x0
31:1	Reserved			

SBCR

Offset Address: 0x090

Bits	Name	Access	Description	Reset
0	SBCR	R/W	Shadow Break Control Bit This is a shadow register for the Break bit (LCR[6]).	0x0
31:1	Reserved			

SDMAM

Offset Address: 0x094

Bits	Name	Access	Description	Reset
0	SDMAM	R/W	Shadow DMA Mode. This is a shadow register for the DMA mode bit (FCR[3]).	0x0
31:1	Reserved			

SFE

Offset Address: 0x098

Bits	Name	Access	Description	Reset
0	SFE	R/W	Shadow FIFO Enable. this is a shadow register for the FIFO enable bit (FCR[0]).	0x0
31:1	Reserved			

SRT

Offset Address: 0x09c

Bits	Name	Access	Description	Reset
1:0	SRT	R/W	Shadow RCVR Trigger. this is a shadow register for the RCVR trigger bits (FCR[7:6]).	0x0
31:2	Reserved			

STET

Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
1:0	STET	R/W	Shadow TX Empty Trigger. This is a shadow register for the TX empty trigger bits (FCR[5:4]).	0x0
31:2	Reserved			

HTX

Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
0	HTX	R/W	This register is use to halt transmissions for testing.	0x0
31:1	Reserved			

DMASA

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
0	DMASA	R/W	This register is use to perform a DMA software acknowledge if a transfer needs to be terminated due to an error condition.	0x0
31:1	Reserved			

18.3 SPI

18.3.1 summarize

The system is configured with 4 SPI controller modules, which can act as masters to synchronize serial communication with external devices and realize serial-parallel and parallel-serial conversion of data.

18.3.2 specificities

The SPI controller module is characterized as follows.

- Supports Motorola SPI (full-duplex), TI SSP (full-duplex), NS MicroWire (half-duplex) serial peripheral interface protocols.
- Separate transmit/receive FIFOs
- Programmable data frame length: 4 to 16 bits
- SPI interface clock frequency programmable
- Supports DMA mode of operation
- Supports internal loopback test mode
- The operating reference clock can be set to 187.5MHz or 100MHz, and the output SPI_SCK supports a maximum of 46.875MHz.

18.3.3 Functional Description

18.3.3.1 typical application

The application block diagram for SPI master to external slave is shown in Figure 18-9.

Figure *18-9 SPI* Application Block Diagram

18.3.4 working method

18.3.4.1 operating mode

The operating modes of SPI are categorized into.

- Data transfer in interrupt or query mode
- Data transfer in DMA mode.

18.3.4.2 clocks

The SPI controller module reference clock can be set to 187.5MHz or 100MHz. the output SPI_SCK supports a maximum of 46.875MHz.
The calculations are as follows:
Output SPI_SCK = SPI operating reference clock / BAUDR
SPI working reference clock: 187.5MHz or 100MHz. BAUDR register: Set an even number between 2 and 65534. Calculation Example.
SPI operating reference clock = 187.5MHz with BAUDR =4
Output SPI_SCK = 187.5MHz/4 = 46.875MHz

18.3.4.3 interrupt processing

The SPI controller module has six interrupts, the first five of which are maskable independent interrupt sources active high.

- RXFINTR
  Receive FIFO interrupt request. This interrupt is set when there is RXFTLR+1 or more valid data in the receive FIFO.
- RXOINTR
  This interrupt is set when the receive FIFO is full and new data needs to be written to the FIFO, causing a FIFO Overflow. At this time the data is written to the receive shift register instead of the FIFO.
- RXUINTR
  When the receive FIFO is read empty and no new data is written to the receive FIFO before a new read request occurs, it will cause a FIFO Underflow and this interrupt will be set. At this time, all read values are 0. The interrupt can be cleared by reading register RXUICR.
- TXOINTR
  This interrupt is set when the transmit FIFO is full and new data needs to be written to the FIFO, causing a FIFO Overflow.
- TXEINTR

Send FIFO Interrupt Request. This interrupt is set when there are TXFTLR or fewer valid data in the transmit FIFO.

- SPI_INTR

  Combined interrupt is the result of the above 5 interrupts after the "or" operation.

To mask this interrupt, you must set register IMR to mask the above five interrupts. If any of the above five independent interrupts is set and enabled, the interrupt is set.

18.3.4.4 initialization

The SPI controller module initialization procedure is as follows:

Step 1 Set SPIENR register to "0" to stop the SPI module.

Step 2 Configure register BAUDR to set the output clock division divisor, the setting value must be even. Step 3 Configure register CTRLR0 to configure parameters such as transmission data bit width and transmission frame format.

Step 4 In DMA operation mode, configure register DMACR to enable the DMA function of the SPI. When operating in DMA mode, set the interrupt related registers to disable the generation of interrupt signals.

Step 5 In interrupt operation mode, set register IMR to generate the corresponding interrupt signal. Step 6 Set register SPIENR to "1" to enable the SPI module.

18.3.4.5 SPI Data Transfer Flow

- The flow when the SPI master interfaces with an external SPI/SSP slave is shown in Figure 18-10.

Fig. *18-10* Data Transfer Flow for External *SPI/SSP Slave* Connection

- The flow when the SPI master interfaces with an external Microwire slave is shown in Figure 18-11.

Figure *18-11* Data Transfer Flow When Connecting to an External *Microwire Slave*

18.3.4.6 Data transfer in DMA mode

The SPI module uses two DMA channels, one for transmit and one for receive. The

SPI DMA mode sets the phase

The off registers are DMACR, DMATDLR, and DMARDLR.
To enable SPI DMA mode,
proceed as follows: Step 1
Acquire two DMA channels.
Step 2 Set register DMACR[1:0] to enable SPI DMA
transmit and receive. Step 3 Set register SPIENR to "1"
to enable SPI.
Step 4 Send data

1. Configure the control registers associated with the transmit DMA channel.
2. Starts the DMA controller in response to a request from the SPI to send a FIFO.
3. Interrupt reporting through the DMA controller to determine whether the transmission is complete, and if so, close the The transmit DMA

function of SPI. Step 5 Receive Data

1. Configure the control registers associated with the receive DMA channel.
2. Starts the DMA controller in response to a DMA request from the SPI receive FIFO.
3. The DMA controller interrupt is reported to determine whether the data reception is completed, and if it is completed, the receive DMA function of SPI is disabled.

Step 6 Set the register SPIENR to "0" to stop SPI.

18.3.5 Three Serial Peripheral Bus Timing Diagrams

18.3.5.1 Motorola SPI Interface

The following diagrams represent the various data transfer formats of Motorola SPI. SCPH stands for SPI_SCK phase and SCPOL stands for SPI_SCK polarity, which are set by registers CTRLR0[7:6].

(A) SCPH = 0

In this mode, SPI_CS_X is set to high level when it is in idle state and low level when it is transmitting, SPI_SCK is different by SCPOL setting, SCPOL = 0, set to low level when it is in idle state, and grab the data with the rising edge of the clock when it is transmitting, and SCPOL = 1, set to high level when it is in idle state, and grab the data with the falling edge of the clock when it is transmitting. SCPOL = 1, which is set to high when idle, and the data is captured on the falling edge of the clock during transmission.

- The single-frame transmission format is shown in Figure 18-12.

algorithmic technology

Figure *18-12 Motorola SPI* Single Frame Transmission Format *(SCPH = 0)*

- The continuous frame transmission format is shown in Figure 18-13.

Figure *18-13 Motorola SPI* Sequential Frame Transfer Format *(SCPH = 0)*

(B) SCPH = 1

In this mode, SPI_CS_X is set to high level when it is in idle state and low level when it is transmitting, SPI_SCK is different by SCPOL setting, SCPOL = 0, set to low level when it is in idle state, and the data is captured with the falling edge of the clock when it is transmitting, SCPOL = 1, set to high level when it is in idle state, and the data is captured with the rising edge of the clock when it is transmitting. SCPOL = 1, which is set to high level when in idle state, and the data is captured on the rising edge of the clock during transmission.

- The single-frame transmission format is shown in Figure 18-14.

Figure *18-14 Motorola SPI* Single Frame Transmission Format *(SCPH = 1)*

- The continuous frame transmission format is shown in Figure 18-15.

Figure *18-15 Motorola SPI* Sequential Frame Transfer Format *(SCPH = 1)*

18.3.5.2 TI Synchronous Serial Interface

In SSP mode, SPI_CS_X is set high when it is in idle state and low when it is transmitting. SPI_SCK is set low when it is in idle state and data is grabbed on the falling edge of the clock during transmission. The following diagrams represent the TI SSP data transfer format.

- The single-frame transmission format is shown in Figure 18-16.
-

Figure *18-16 TI SSP* Single Frame Transmission Format

- The continuous frame transmission format is shown in Figure 18-17.

Figure *18-17 TI SSP* Continuous Frame Transmission Format

18.3.5.3 National Semiconductor Microwire Interface

In Microwire mode, SPI_CS_X is set high when idle and low when transmitting. SPI_SCK is set low when it is idle, and data is grabbed on the rising edge of the clock during transmission.
In this mode, when transferring data, the control word must be added first, and then the external chip will respond to the data word demanded by the Master according to the control word. The control word length can be set by register CTRLR0[15:12], and other related parameters can be set by register MWCR.

The following diagrams represent the NS Microwire data transfer format.

- The single-frame transmission format is shown in Figure 18-18.

Figure *18-18 NS Microwire* Single Frame Transmission Format

- The continuous frame transmission format is shown in Figure 18-19.

Figure *18-19 NS Microwire* Continuous Frame Transmission Format

18.3.6 Registers Overview

The four SPI module base addresses of the chip are shown in Table 18-6.

Table *18-6* Chip Base Addresses for *4 SPI* Modules

GPIO Module	base address
SPI0	0x04180000
SPI1	0x04190000
SPI2	0x041A0000
SPI3	0x041B0000

Table 18-7 shows the offset addresses and definitions of the Group 1 SPI module (SPI0) registers, SPI0 through SPI3 have the same register definitions.

Table *18-7 SPI* Register Overview

Name	Address Offset	Description
CTRLR0	0x000	Control Register 0
CTRLR1	0x004	Control Register 1
SPIENR	0x008	SPI Enable Register
MWCR	0x00c	Microwire Control Register
SER	0x010	Slave Enable Register
BAUDR	0x014	Baud Rate Select
TXFTLR	0x018	Transmit FIFO Threshold Level
RXFTLR	0x01c	Receive FIFO Threshold Level
TXFLR	0x020	Transmit FIFO Level Register
RXFLR	0x024	Receive FIFO Level Register
SR	0x028	Status Register
IMR	0x02c	Interrupt Mask Register
ISR	0x030	Interrupt Status Register
RISR	0x034	Raw Interrupt Status Register
TXOICR	0x038	Transmit FIFO Overflow Interrupt Clear Register
RXOICR	0x03c	Receive FIFO Overflow Interrupt Clear Register
RXUICR	0x040	Receive FIFO Underflow Interrupt Clear Register
MSTICR	0x044	Multi-Master Interrupt Clear Register
ICR	0x048	Interrupt Clear Register
DMACR	0x04c	DMA Control Register
DMATDLR	0x050	DMA Transmit Data Level
DMARDLR	0x054	DMA Receive Data Level
DR (36 groups)	0x060	Data Register
RX_SAMPLE_DLY	0x0f0	Rx Sample Delay Register

18.3.7 register description

CTRLR0

Offset Address: 0x000

Bits	Name	Access	Description	Reset
15:0	CTRLR0	R/W	[15:12] Control Frame Size. Selects the length of the control word for the Microwire frame format. 0000 1-bit control word 0001 2-bit control word 0010 3-bit control word 0011 4-bit control word 0100 5-bit control word 0101 6-bit control word 0110 7-bit control word 0111 8-bit control word 1000 9-bit control word 1001 10-bit control word 1010 11-bit control word 1011 12-bit control word 1100 13-bit 13-bit control word 1101 14-bit control word 1110 15-bit control word	0x7

			1111 16-bit control word	

Bits	Name	Access	Description	Reset
			[11] Shift Register Loop. Used for testing purposes only. When internally active, connects the transmit shift register output to the receive shift register input. can be used in both both serialslave and serial-master modes. 0 - Normal Mode Operation 1 - Test Mode Operation [10] only for slave mode. [9:8] Transfer Mode. Selects the mode of transfer for serial This field does not affect the transfer duplicity. only indicates whether the receive or Transmit data are valid. In transmit-only mode, data received from the external device is not valid and is not stored in the receive FIFO memory; it is overwritten on the next transfer. In receive-only mode, transmitted data are not valid. After the first write to the transmit FIFO, the same word is retransmitted for the duration of the transfer. In transmit-and-receive mode, both The transfer continues until the transmit FIFO is empty. Data received from the external device are stored into the receive FIFO memory, where it can be used to store the data. receive FIFO memory, where it can be accessed by the host processor. In eeprom-read mode, receive data is not valid while control data is being transmitted. When all control data is sent to the EEPROM, receive data becomes valid and transmit data becomes invalid. All data in the transmit FIFO is considered All data in the transmit FIFO is considered control data in this mode. 00 -- Transmit & Receive 01 -- Transmit Only 10 -- Receive Only 11 -- EEPROM Read [7] Serial Clock Polarity. Valid when the frame format (FRF) is set to Motorola SPI. used to select the polarity of the inactive serial clock, which is held inactive when the SPI master is not actively transferring data on the serial bus.	

			0 - Inactive state of serial clock is low 1 - Inactive state of serial clock is high	

Bits	Name	Access	Description	Reset
			[6] Serial Clock Phase. Valid when the frame format (FRF) is set to Motorola SPI. The serial clock phase selects the relationship of the serial clock with the slave select signal. When SCPH = 0, data are captured on the first edge of the serial clock. When SCPH = 1, the serial clock starts toggling one cycle after the slave select line is When SCPH = 1, the serial clock starts toggling one cycle after the slave select line is activated, and data are captured on the second edge of the serial clock. 0: Serial clock toggles in middle of first data bit 1: Serial clock toggles at start of first data bit [5:4] Frame Format. Selects which serial protocol transfers the data. 00 -- Motorola SPI 01 -- Texas Instruments SSP 10 -- National Semiconductors Microwire 11 -- Reserved [3:0] Data Frame Size. Selects the data frame length. When the data frame size is programmed to be less than 16 bits, the receive data are automatically right-justified by the When the data frame size is programmed to be less than 16 bits, the receive data are automatically right-justified by the receive logic, with the upper bits of the receive FIFO zero-padded. right-justify transmit data before writing into the transmit FIFO. The transmit logic ignores the upper unused bits when transmitting the data. 0000 Reserved - undefined operation 0001 Reserved - undefined operation 0010 Reserved - undefined operation 0011 4-bit serial data transfer 0100 5-bit serial data transfer 0101 6-bit serial data transfer 0110 7-bit serial data transfer 0111 8-bit serial data transfer 1000 9-bit serial data transfer 1001 10-bit serial data transfer 1010 11-bit serial data transfer 1011 12-bit serial data transfer 1100 13-bit serial data transfer 1101 14-bit serial data transfer 1101	

			14-bit serial data transfer 1110 15-bit serial data transfer 1111 16-bit serial data transfer	
31:16	Reserved			

CTRLR1

Offset Address: 0x004

Bits	Name	Access	Description	Reset
15:0	CTRLR1	R/W	Number of Data Frames. When TMOD = 10 or TMOD = 11, this register field sets the number of data frames to be continuously received by the SPI. The SPI continues to receive serial data until the number of data frames is reached. The SPI continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables you to receive up to 64 KB of data in a continuous transfer.	0x0
31:16	Reserved			

SPIENR

Offset Address: 0x008

Bits	Name	Access	Description	Reset
0	SPIENR	R/W	SPI Enable. Enables and disables all SPI operations. When disabled, all serial transfers are halted immediately. It is impossible to program some of the SPI control registers when enabled. When disabled, the spi_sleep output is set (after delay) to inform the system that it is safe to remove the spi_sleep output.) to inform the system that it is safe to remove the spi_clk, thus saving power consumption in the system.	0x0
31:1	Reserved			

MWCR

Offset Address: 0x00c

Bits	Name	Access	Description	Reset

2:0	MWCR	R/W	[2] Microwire Handshaking. Relevant only when the SPI is Used to enable and disable the "busy/ready" handshaking interface for the Microwire protocol. "busy/ready" handshaking interface for the Microwire protocol. When enabled, the SPI checks for a ready status from the target slave, after the transfer of the last data/control bit, before clearing the BUSY status in the SR register. 0: handshaking interface is disabled 1: handshaking interface is enabled [1] Microwire Control. Defines the direction of the data word when the Microwire serial protocol is used. When this bit is set to 0, the data word is received by the SPI from the external serial device. When this bit is set to 0, the data word is received by the SPI from the external serial device. from the SPI to the external serial device.	0x0

Bits	Name	Access	Description	Reset
			[0] Microwire Transfer Mode. Defines whether the Microwire transfer is sequential or non-sequential. When sequential mode is used, only one control word is needed to transmit or receive a block of data words. When non-sequential mode is used, there must be a control word for each data word that is transmitted or received. When non-sequential mode is used, there must be a control word for each data word that is transmitted or received. 0 - non-sequential transfer 1 - sequential transfer	
31:3	Reserved			

SER

Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	SER	R/W	Slave Select Enable Flag. This register corresponds to a slave select line (ss_x_n]) from the SPI master. When this register is set (1), the slave select line from the master is This register corresponds to a slave select line (ss_x_n] from the SPI master. corresponding slave select outputs until a transfer is started. Before beginning a transfer, you should enable this register that corresponds to the Before beginning a transfer, you should enable this register that corresponds to the slave device with which the master wants to communicate. 1: Selected 0: Not Selected	0x0
31:1	Reserved			

BAUDR

Offset Address: 0x014

Bits	Name	Access	Description	Reset
15:0	BAUDR	R/W	SPI Clock Divider(SCKDV). The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register. If the value is 0, the serial output clock (sclk_out) is disabled. _out is derived from the following equation. Fsclk_out/SCKDV = Fssi_clk where SCKDV is any even value between 2 and 65534. For example. for Fssi_clk = 3.6864MHz and SCKDV =2 Fsclk_out = 3.6864/2 = 1.8432MHz	0x0

31:16	Reserved		

TXFTLR

Offset Address: 0x018

Bits	Name	Access	Description	Reset
2:0	TXFTLR	R/W	Transmit FIFO Threshold. Controls the level of entries (or below) at which the transmit FIFO controller triggers an interrupt. The FIFO depth is 8; If you attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and If you attempt to set this value greater than or equal to the depth of the FIFO, this field is not written and retains its current value. than or equal to this value, the transmit FIFO empty interrupt is triggered.	0x0
31:3	Reserved			

RXFTLR

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
2:0	RXFTLR	R/W	Receive FIFO Threshold. Controls the level of entries (or above) at which the receive FIFO controller triggers an interrupt. The FIFO depth is 8. If you attempt to set this value greater than the depth of the FIFO, this field is not written and retains its current value. When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered.	0x0
31:3	Reserved			

TXFLR

Offset Address: 0x020

Bits	Name	Access	Description	Reset
3:0	TXFLR	RO	Transmit FIFO Level. Contains the number of valid data entries in the transmit FIFO.	
31:4	Reserved			

RXFLR

Offset Address: 0x024

Bits	Name	Access	Description	Reset
3:0	RXFLR	RO	Receive FIFO Level. Contains the number of valid data entries in the receive FIFO.	
31:4	Reserved			

SR

Offset Address: 0x028

Bits	Name	Access	Description	Reset

6:0	SR	RO	[6] Data Collision Error. This bit is set if the SPI master is actively transmitting when another master selects this device as a slave. This informs the processor that the last data transfer was	

Bits	Name	Access	Description	Reset
			This bit is cleared when read. 0 - No error 1 - Transmit data collision error [5] Transmission Error. Set if the transmit FIFO is empty when a transfer is started. Data from the previous transmission is resent on the txd line. This bit is cleared when read. 0 - No error 1 - Transmission error [4] Receive FIFO Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0 - Receive FIFO is not full 1 - Receive FIFO is full [3] Receive FIFO Not Empty. Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO. Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. 0 - Receive FIFO is empty 1 - Receive FIFO is not empty [2] Transmit FIFO Empty. When the transmit FIFO is completely empty, this bit is set. When the transmit FIFO contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. This bit field does not request an interrupt. 0 - Transmit FIFO is not empty 1 - Transmit FIFO is empty [1] Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0 - Transmit FIFO is full 1 - Transmit FIFO is not full [0] SPI Busy Flag. When set, indicates that a serial transfer is in progress; when cleared indicates that the SPI is idle or disabled. 0 - SPI is idle or disabled 1 - SPI is actively transferring data	

31:7	Reserved		

IMR

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
5:0	IMR	R/W	[5] Multi-Master Contention Interrupt Mask. 0 - spi_mst_intr interrupt is masked 1 - spi_mst_intr interrupt is not masked [4] Receive FIFO Full Interrupt Mask 0 - spi_rxf_intr interrupt is masked 1 - spi_rxf_intr interrupt is not masked [3] Receive FIFO Overflow Interrupt Mask 0 - spi_rxo_intr interrupt is masked 1 - spi_rxo_intr interrupt is not masked [2] Receive FIFO Underflow Interrupt Mask 0 - spi_rxu_intr interrupt is masked 1 - spi_rxu_intr interrupt is not masked [1] Transmit FIFO Overflow Interrupt Mask 0 - spi_txo_intr interrupt is masked 1 - spi_txo_intr interrupt is not masked [0] Transmit FIFO Empty Interrupt Mask 0 - spi_txe_intr interrupt is masked 1 - spi_txe_intr interrupt is not masked	0x3F
31:6	Reserved			

ISR

Offset Address: 0x030

Bits	Name	Access	Description	Reset
5:0	ISR	RO	[5] Multi-Master Contention Interrupt Status. 0 = spi_mst_intr interrupt not active after masking 1 = spi_mst_intr interrupt is active after masking [4] Receive FIFO Full Interrupt Status 0 = spi_rxf_intr interrupt is not active after masking 1 = spi_rxf_intr interrupt is full after masking [3] Receive FIFO Overflow Interrupt Status 0 = spi_rxo_intr interrupt is not active after masking 1 = spi_rxo_intr interrupt is active after masking [2] Receive FIFO Underflow Interrupt Status	

Bits	Name	Access	Description	Reset
			0 = spi_rxu_intr interrupt is not active after masking 1 = spi_rxu_intr interrupt is active after masking [1] Transmit FIFO Overflow Interrupt Status 0 = spi_txo_intr interrupt is not active after masking 1 = spi_txo_intr interrupt is active after masking [0] Transmit FIFO Empty Interrupt Status 0 = spi_txe_intr interrupt is not active after masking 1 = spi_txe_intr interrupt is active after masking	
31:6	Reserved			

RISR

Offset Address: 0x034

Bits	Name	Access	Description	Reset
5:0	RISR	RO	[5] Multi-Master Contention Raw Interrupt Status. 0 = spi_mst_intr interrupt is not active prior to masking 1 = spi_mst_intr interrupt is active prior masking [4] Receive FIFO Full Raw Interrupt Status 0 = spi_rxf_intr interrupt is not active prior to masking 1 = spi_rxf_intr interrupt is active prior to masking [3] Receive FIFO Overflow Raw Interrupt Status 0 = spi_rxo_intr interrupt is not active prior to masking 1 = spi_rxo_intr interrupt is active prior masking [2] Receive FIFO Underflow Raw Interrupt Status 0 = spi_rxu_intr interrupt is not active prior to masking 1 = spi_rxu_intr interrupt is active prior to masking [1] Transmit FIFO Overflow Raw Interrupt Status 0 = spi_txo_intr interrupt is not active prior to masking 1 = spi_txo_intr interrupt is active prior	

Bits	Name	Access	Description	Reset
			masking [0] Transmit FIFO Empty Raw Interrupt Status 0 = spi_txe_intr interrupt is not active prior to masking 1 = spi_txe_intr interrupt is active prior masking	
31:6	Reserved			

TXOICR

Offset Address: 0x038

Bits	Name	Access	Description	Reset
0	TXOICR	RO	Clear Transmit FIFO Overflow Interrupt. this register reflects the status of the interrupt. A read from this register is not allowed. clears the spi_txo_intr interrupt; writing has no effect.	
31:1	Reserved			

RXOICR

Offset Address: 0x03c

Bits	Name	Access	Description	Reset
0	RXOICR	RO	Clear Receive FIFO Overflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the spi_rxo_intr interrupt; writing has no effect.	
31:1	Reserved			

RXUICR

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	RXUICR	RO	Clear Receive FIFO Underflow Interrupt. This register reflects the status of the interrupt. A read from this register clears the spi_rxu_intr interrupt; writing has no effect.	
31:1	Reserved			

MSTICR

Offset Address: 0x044

Bits	Name	Access	Description	Reset
0	MSTICR	RO	Clear Multi-Master Contention Interrupt. This register reflects the status of the interrupt. A read from this register clears the spi_mst_intr interrupt; writing has no effect.	
31:1	Reserved			

ICR

Offset Address: 0x048

Bits	Name	Access	Description	Reset
0	ICR	RO	Clear Interrupts. This register is set if any of the interrupts below are active. A read clears the spi_txo_intr, spi_rxu_intr, spi_rxo_intr, and the spi_mst_intr Writing to this register has no effect.	
31:1	Reserved			

DMACR

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
1:0	DMACR	R/W	[1] Transmit DMA Enable. This bit enables/disables the transmit FIFO DMA channel. 0 = Transmit DMA disabled 1 = Transmit DMA enabled [0] Receive DMA Enable. This bit enables/disables the receive FIFO DMA channel. 0 = Receive DMA disabled 1 = Receive DMA enabled	0x0
31:2	Reserved			

DMATDLR

Offset Address: 0x050

Bits	Name	Access	Description	Reset
2:0	DMATDLR	R/W	Transmit Data Level. This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and TDMAE = 1.	0x0
31:3	Reserved			

DMARDLR

Offset Address: 0x054

Bits	Name	Access	Description	Reset
2:0	DMARDLR	R/W	Receive Data Level. This bit field controls the level at which a DMA request is made by the receive logic. the watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or above this field value + 1, and RDMAE=1.	0x0
31:3	Reserved			

DR

Offset Address: 0x060

Bits	Name	Access	Description	Reset
15:0	DR	R/W	Data Register. When writing to this register, you must right-justify the data. Read data are automatically right-justified. Read = Receive FIFO buffer Write = Transmit FIFO buffer Note. The DR register in the SPI occupies thirty-six 32-bit address locations of the memory map to facilitate AHB burst Writing to any of these address locations has the same effect as pushing the data from the pwdata bus into the transmit FIFO. Reading from any of these locations has the same effect as popping data from the receive Reading from any of these locations has the same effect as popping data from the receive FIFO onto the prdata bus.	0x0
31:16	Reserved			

RX_SAMPLE_DLY

Offset Address: 0x0f0

Bits	Name	Access	Description	Reset
7:0	RX_SAMPLE_DLY	R/W	Receive Data (rxd) Sample Delay. This register is used to delay the sample of the rxd input signal. Each value represents a single ssi_clk delay on the sample of the rxd signal. NOTE: If this register is programmed with a value that exceeds the depth of the internal shift registers (DEPTH = 8), a zero (0) delay will be applied to the rxd sample.	0x0
31:8	Reserved			

18.4 eMMC/SD/SDIO Controller

18.4.1 Functional Description

18.4.1.1 functional block diagram

EMMC/SD/SDIO controllers (SDMMC controllers for short) are used to handle operations such as reading and writing data on SD cards and eMMC, as well as external devices (such as Bluetooth, WIFI, etc.) supported by the SDIO protocol.

This chip provides three sets of SDMMC controllers. Among them EMMC supports eMMC4.1 and eMMC4.5 compliant devices.

SDIO0 supports Secure Digital Memory (SD 3.0) compliant devices. SDIO1 supports Secure Digital I/O (SDIO 3.0) compliant devices.

The corresponding function signals and pin positions of the three SDMMC controllers in the chip are shown in the table below.

Table *18-8 SDMMC* Controller Function Signal and Pin Correspondence Table

SDMMC Controller	functional signal	foot position (in dance)
eMMC	EMMC_CLK	EMMC_CLK
	EMMC_CMD	EMMC_CMD
	EMMC_DATA0	EMMC_DAT0
	EMMC_DATA1	EMMC_DAT1
	EMMC_DATA2	EMMC_DAT2
	EMMC_DATA3	EMMC_DAT3
	EMMC_RSTn	EMMC_RSTN
SDIO0	SD_CLK	SD0_CLK
	SD_CMD	SD0_CMD
	SD_DATA0	SD0_D0
	SD_DATA1	SD0_D1
	SD_DATA2	SD0_D2
	SD_DATA3	SD0_D3
	SD_CARD_DETECT	SD0_CD
	SD_POWER_EN	SD0_PWR_EN
SDIO1	SDIO_CLK	SD1_CLK
	SDIO_CMD	SD1_CMD
	SDIO_DATA0	SD1_D0
	SDIO_DATA1	SD1_D1
	SDIO_DATA2	SD1_D2
	SDIO_DATA3	SD1_D3

Figure *18-20* Functional Block Diagram of *SDMMC* Controller

SDMMC functions:

1. Supports SD cards, SDIO devices and eMMC.
2. Transfers data between eMMC/SD/SDIO and system memory data through the internal DMA controller.
3. Supports CRC generation and checking of commands and data.
4. The frequency required for different modes can be generated by the internal de-esser.
5. Provides a mechanism to turn off the internal clock and the interface clock to achieve power saving.
6. Provides 1-bit and 4-bit data transmission interfaces to communicate with the device.
7. Supports read and write operations with block_size equal to 1~2048byte.
8. Supports SDIO protocol, including interrupt interval, suspend, resume and read wait operations.
9. Supports AXI/AHB interface with internal DMA access to system memory.
10. Supports AHB interface for accessing the internal storages through the CPU.

Figure *18-21* Typical Applications

18.4.1.2 Command and Response

The eMMC/SD bus packet consists of three main components: command, response and data. The command and response packets are transmitted via CMD signaling.

- Instruction packet

The command packet is transmitted from the master to the device to indicate the start of an operation. The packet format consists of 48 bits, including a start bit, a transmit bit, a command number, a command parameter, a CRC validation code, and an end bit. This is shown in Figure 18-22.

Figure *18-22 eMMC/SD/SDIO* Command Format

- response packet
  When the device side receives a command, it returns a response according to the different command categories, which is used to display the status or parameters of the device side. The length is 48 bits or 136 bits. This is shown in Figure 18-23.

Figure *18-23 eMMC/SD/SDIO* Response Format

- Data packets: Data packets are used to exchange data between the master and the device. Depending on the requirements, 1-bit (DATA0), 4-bit (DATA0-DATA3) or 7-bit (DATA0-DATA7) can be selected, and each data signal line can choose to transmit 1-bit (single data rate) or 2-bit (dual data rate) in each clock interval. Each data signal line can choose to transmit either 1-bit (single data rate) or 2-bit (dual data rate) in each clock interval. The packet grids are shown in Figure 18-24 to Figure 18-28.

Figure *18-24 eMMC/SD/SDIO 1-bit* Data Packet Format

Figure *18-25 eMMC/SD/SDIO 4-bit* Data Packet Format

Figure *18-26 eMMC/SD/SDIO 8-bit* Data Packet Format

Figure *18-27 4-bit dual data rate* data packet format

Figure *18-28 8-Bit Dual Data Rate* Data Packet Format

Depending on whether or not there is a transfer of information, instructions can be subdivided into the following two types.

- Non-data transmission commands: Commands are transmitted and received through the signal line CMD.

Figure *18-29* Non-Data Transmission Commands*:* Command Transmission and Receive Response via Signal Line *CMD*

- Data transfer commands: In addition to the interaction on the signal line CMD, there is also data transfer on the data lines DAT0~DAT3.

18.4.1.3 data transmission

The data transfer between the master and the device is mainly based on the block unit, and in addition to the data, CRC check bits are also included to verify the correctness of the data. The commonly used methods are single-block data read/write and multi-block data read/write. Compared with single-block data transfer, multi-block data transfer has higher efficiency. The block size of EMMC and SD card is 512byte, while SDIO is more special, it can support 1~2048byte block size, users can define the block size value according to different devices.

(1) Single-block and multi-block read operations are shown in Figure 18-30. A single-block transfer consists of a command, response, data, and CRC. Multiblock transfers are terminated by a reliable STOP CMD.

Figure *18-30* Single-Block and Multi-Block Read Operations

(2) For single and multi-block write operations as shown in Figure 18-31, the transfer process sends a BUSY signal over the DAT0 signal line to notify the master that a write is in progress.

Figure *18-31* Single-Block and Multi-Block Write Operations

18.4.1.4 SD3.0 Supported Speed Modes and Voltage Switching

- Voltage switching program (1.8V → 3.3V)

Step 1: Set PWRSW to 3.3V
=> sd_pwrsw_ctrl (0x030001F4) = 0x00000009
(reg_pwrsw_auto=1, reg_pwrsw_disc=0, reg_pwrsw_vsel=0(3.0v), reg_en_pwrsw=1)

Step 2: Wait 1ms for voltage switching to be completed

- Voltage switching program (3.3V → 1.8V)

Step 1: Set PWRSW to 1.8V
=> sd_pwrsw_ctrl (0x030001F4) = 0x0000000B
(reg_pwrsw_auto=1, reg_pwrsw_disc=0, reg_pwrsw_vsel=1(1.8v), reg_en_pwrsw=1)

Step 2: Wait for 1ms to complete voltage switching.

- Supports speed mode and voltage

The speed modes and voltages supported by SD3.0 are shown in the table below.

Table *18-9 SD3.0* Supported Speeds and Voltages

Support Mode		Support Speed	input voltage
DS (default speed)		25Mhz	1.8V/3.3V
HS (high speed)		50Mhz	1.8V/3.3V
SDR12		25Mhz	1.8V
SDR25		50Mhz	1.8V
DDR50		50Mhz	1.8V
SDR50		100Mhz	1.8V
SDR104		187.5Mhz	1.8V

algorithmic technology

18.4.1.5 eMMC Supported Speed Modes and Voltages

The speed modes and voltages supported by eMMC are listed in the table below.

Table *18-10 eMMC* Supported Speed Modes and Voltages

Support Mode	Support Speed	input voltage
DS (default speed)	26Mhz	1.8V/3.3V
HS (high speed)	52Mhz	1.8V/3.3V
DDR52	52Mhz	1.8V
HS200	187.5Mhz	1.8V

18.4.2 Application Notes

18.4.2.1 Clock Valve Control

Figure *18-32* Clock Off Control Program

As shown in Figure 18-32 for the clock shutdown procedure, the master must ensure that there are no transmissions in progress on the sink before turning off the clock.

(1) Read the temporary memory PRESENT_STS
(2) Checks if bits CMD_INHIBIT and DAT_INHIBIT are both 0.
(3) If any bit is not 0, the transmission is still in progress and needs to be delayed.
(4) If both are 0, you can set CLK_CTL[SD_CLK_EN]=0 to turn off the clock.

18.4.2.2 soft reset (electronics)

When the controller operates abnormally, a soft reset is performed by reliably resetting the configuration register (base address = 0x0300_3000). The addresses of the registers used are as follows.

6. EMMC : SOFT_RSTN_0[reg_soft_reset_x_emmc] (address offset : 0x000, Bit15)
7. SDIO0 : SOFT_RSTN_0[reg_soft_reset_x_sd0] (address offset : 0x000, Bit16)
8. SDIO1 : SOFT_RSTN_0[reg_soft_reset_x_sd1] (address offset : 0x000, Bit17)

18.4.2.3 Interface Clock Configuration

Figure 18-33 shows the interface clock configuration flowchart. the SDMMC controller provides an internal de-esser to allow the user to adjust the required clock frequency according to different protocols and speed modes. The relationship is as follows.

$$FSD_CLK_OUT = FINT_CARD_CLK / (2 \times clk_divisor)$$

In addition to ensuring that no commands or data are still being transmitted when SDMMC changes frequency, it is also important to follow the steps in the Interface Clock Configuration Flowchart to avoid burrs in the clock output to the eMMC/SD unit.

(1) Turns off the interface clock.
(2) Calculate the crossover factor.
(3) Set the crossover factor. Fill in CLK_CTL[FREQ_SEL] with the parameter calculated in (2) and start to turn on the internal clock switch (CLK_CTL[INT_CLK_EN]=1).
(4) Check CLK_CTL[INT_CLK_STABLE] to confirm that the frequency switching is complete.
(5) If not yet completed (CLK_CTL[INT_CLK_STABLE]=0), delay wait.
(6) If switching the clock frequency is complete, the interface clock is turned on.

Figure *18-33* Clock Configuration Flowchart

18.4.2.4 non-data transfer command

- Command Transfer Program

  The command transfer procedure is shown in Figure 18-34.

  (1) You must first check that the register bit PRESENT[CMD_INHIBIT] is 0 to verify

algorithmic technology

that the CMD Line is

No. Still in use.

(2) If the CMD Line is idle, further check whether it is an instruction with busy. If it is not a busy instruction, there is no need to check the status on the DATA Line, and step (5) is executed directly. On the other hand, if it is a busy instruction, step (3) must be executed to confirm whether it is an Abort instruction or not.

(3) If it is Abort instruction, it means that when CMD line finishes transmission, DATA line is also idle at the same time, so you can go to step (5) directly; on the other hand, if it is not Abort instruction, you have to execute step (4) to confirm whether busy on DATA line has been released or not.

(4) Check whether the register bit PRESENT[DAT_INHIBIT] is 0 to confirm whether it is still in use on the DATA Line, and if it is still in use, wait until the transmission is finished, and then execute step (5).

(5) The values of the ARGUMENT and CMD registers are set according to the command requirements.

Figure *18-34* Command Transfer Program

- Command Complete Sequence

The instruction completion procedure is shown in Figure 18-35.

(1) The first interrupt that will wait for Command to complete is NORM_INT_STS[CMD_CMPL].

(2) Set NORM_INT_STS[CMD_CMPL]=1 to clear the CMD_CMPL interrupt state after receiving the interrupt.

(3) Then read RESP1_0, RESP3_2, RESP5_4, and RESP7_6 to get the response value.

(4) If it is an instruction containing information transfer, step (5) will be executed, otherwise skip to step (8).

(5) Wait for data transfer interrupt NORM_INT_STS[XFER_CMPL].

(6) Set NORM_INT_STS[XFER_CMPL]=1 to clear the XFER_CMPL interrupt status after receiving the interrupt.

(7) Check RESP1_0, RESP3_2, RESP5_4, and RESP7_6 registers for error status. If there is no error status, perform step (8) to return that there is no error. If there is an error, step (9) is executed to report the error.

Figure *18-35* Instruction Completion Procedure

18.4.2.5 Stop or abort data transmission

- Abort instruction program
  The abort command is done by CMD12 for eMMC/SD devices and CMD52 for SDIO devices. There are two main conditions for using the timing.
  (1) Stops infinite module data transfer.
  (2) Stop multi-module data transfer.
  The abort command procedure is shown in Figure 18-36, with the following steps.

Figure *18-36* Abort Command Program

There are two types of abort commands: synchronous abort commands and asynchronous abort commands.

- asynchronous abort instruction procedure
  Figure 18-37 shows the program diagram for the asynchronous abort instruction. The detailed steps are as follows.
  (1) Abort commands are executed according to the different transmission modes.
  (2) Set SW_RST_CMD and SW_RST_DAT in SW_RESET register to reset CMD and DAT. Signal wire.
  (3) Check bits SW_RESET[SW_RST_CMD] and SW_RESET[SW_RST_DAT] to confirm that the reset is complete. If both are 0, the program ends. If either of them is 1, go back to step (3) Delay Wait.

Figure *18-37* Programming Diagram for Asynchronous Abort Instruction

- Synchronous abort command procedure
  Figure 18-38 shows the program for the Synchronous Abort instruction. The detailed steps are as follows.
  (1) The write bit BG_CTL[STOP_BG_REQ] stops the transfer action in the Block Gap

area.

(2) Interrupt waiting for transfer completion NORM_INT_STS[XFER_CMPL].

(3) Set NORM_INT_STS[XFER_CMPL]=1 to clear the XFER_CMPL interrupt status after receiving the interrupt.

(4) Abort commands are executed according to the different transmission modes.

(5) Set SW_RST_CMD and SW_RST_DAT in SW_RESET register to reset CMD and DAT. Signal wire.
(6) Check bits SW_RESET[SW_RST_CMD] and SW_RESET[SW_RST_DAT] to confirm that the reset is complete. If both are 0, the program ends. If either of them is 1, go back to step (6) Delay Wait.

Figure *18-38* Programming Diagram for Synchronous Abort Instruction

18.4.2.6 Non-DMA Data Transfer Mode

The procedure for non-DMA data transfer mode is shown in Figure 18-39.The detailed steps are shown below.

(1) Write to the BLK_SIZE register to set the block size.
(2) Write to BLK_CNT register to set the number of blocks.
(3) Write to the ARGUMENT register to set the command parameters.

(4) Write to the XFER_MODE register to set the transmission mode. The Manufacturer can decide the setting according to the usage situation. Include Single or Multiple Block Select, DMA Enable, Block Count Enable, Data Transfer Direction, Auto CMD Enable.

(5) Write to the CMD register to set the type of command and response.
(6) Interrupt waiting for Command completion NORM_INT_STS[CMD_CMPL].
(7) Set NORM_INT_STS[CMD_CMPL]=1 to clear the CMD_CMPL interrupt state after receiving the interrupt.
(8) Then read RESP1_0, RESP3_2, RESP5_4, and RESP7_6 to get the response value.
(9) Step (14) will be executed if it is a read operation and step (10) will be executed if it is a write operation.
(10) Wait for Buffer Write Ready interrupt NORM_INT_STS[BUF_WRDY].
(11) Set NORM_INT_STS[BUFF_WRDY]=1 to clear the BUFF_WRDY interrupt state after receiving the interrupt.
(12) Write the data you want to write to the device to the BUF_DATA register in sequence.
(13) If there are more blocks to write, go back to step (10) until the last block is written, then go to step (18).
(14) Wait for Buffer Read Ready interrupt NORM_INT_STS[BUF_RRDY].
(15) Set NORM_INT_STS[BUFF_RRDY]=1 to clear the BUFF_RRDY interrupt state after receiving the interrupt.
(16) The data received from the device is sequentially read from the BUF_DATA register.
(17) If there are more blocks to read, go back to step (14) until the last block is read, then go to step (18).
(18) Determine if it is a single module transfer, a multi-module transfer or an infinite module transfer. If it is single module or multi-module transmission then skip to step (19). If it is infinite module transmission then skip to step (21) and perform the action of aborting the transmission.
(19) Interrupt waiting for data transfer completion NORM_INT_STS[XFER_CMPL].
(20) Set NORM_INT_STS[CMD_XFER]=1 to clear the XFER_CMPL interrupt state after receiving the interrupt.
(21) Perform the abort transmission procedure.

Figure *18-39* Program for *Non-DMA* Data Transfer Mode

18.4.2.7 SDMA Data Transfer Mode

The SDMA data transfer mode program is shown in Figure 18-40, with the following detailed steps.

(1) Write to the SDMA_SA register to set the starting address of the system memory used for data transfer.
(2) Write to the BLK_SIZE register to set the block size.
(3) Write to BLK_CNT register to set the number of blocks.
(4) Write to the ARGUMENT register to set the command parameters.
(5) Write to the XFER_MODE register to set the transmission mode. The master can decide the setting according to the usage situation. Include Single or Multiple Block Select, DMA Enable, Block Count Enable, Data Transfer Direction, Auto CMD Enable.
(6) Write to the CMD register to set the type of command and response.
(7) Interrupt waiting for Command completion NORM_INT_STS[CMD_CMPL].
(8) Set NORM_INT_STS[CMD_CMPL]=1 to clear the CMD_CMPL interrupt state after receiving the interrupt.
(9) Then read RESP1_0, RESP3_2, RESP5_4, and RESP7_6 to get the response value.
(10) Waiting for data transfer interrupts and DMA interrupts
(11) Read the interrupt status register NORM_INT_STS to determine the interrupt type. If it is a DMA interrupt then skip to step (12), if it is a data transfer interrupt then skip to step (14).
(12) Set NORM_INT_STS[DMA_INT]=1 to clear the DMA_INT status value.
(13) Write to the SDMA_SA register to reset the system memory start address for the next DMA. Then skip to step (10).
(14) Set NORM_INT_STS[DMA_INT]=1 and [NORM_INT_STS[XFER_CMPL]=1 to clear. DMA_INT and XFER_CMPL status values. Then end the program.

Figure *18-40 SDMA* Data Transfer Mode Program

18.4.2.8 ADMA Data Transfer Mode

The ADMA Data Transfer Mode program is shown in Figure 18-41 and the detailed steps are as follows.

(1) Populate the ADMA description sheet into system memory.

(2) Write to the ADMA_SA_L and ADMA_SA_H registers to set the starting address of the system memory used for the description table.
(3) Write to the BLK_SIZE register to set the block size.
(4) Write to BLK_CNT register to set the number of blocks.
(5) Write to the ARGUMENT register to set the command parameters.
(6) Write to the XFER_MODE register to set the transmission mode. The master can decide the setting according to the usage situation. Include Single or Multiple Block Select, DMA Enable, Block Count Enable, Data Transfer Direction, Auto CMD Enable.
(7) Write to the CMD register to set the type of command and response.
(8) Interrupt waiting for Command completion NORM_INT_STS[CMD_CMPL].
(9) Set NORM_INT_STS[CMD_CMPL]=1 to clear the CMD_CMPL interrupt state after receiving the interrupt.
(10) Then read RESP1_0, RESP3_2, RESP5_4, and RESP7_6 to get the response value.
(11) Wait for a data transfer interrupt or an ADMA Error interrupt.
(12) Read the interrupt status registers NORM_INT_STS and ERR_INT_STS to determine the interrupt type. If
Jump to step (13) for ADMA error interrupts or step (15) if it is a data transfer interrupt.
(13) Set ERR_INT_STS[ADMA_ERR]=1 to clear the ADMA_ERR status value.
(14) Enter the ADMA Abort Transaction and execute the Abort Command to abort the data transfer with the device, and check the ADMA Error Status register if necessary to check the cause of the error.
(15) Set NORM_INT_STS[XFER_CMPL]=1 to clear the XFER_CMPL status value. Then end the program.

Figure *18-41 ADMA* Data Transfer Mode Programs

18.4.3 Registers Overview

Table 18-11 shows the overview of SDMMC registers (base address of EMMC is 0x0430_0000, SDIO0 is 0x0431_0000, SDIO1 is 0x0500_0000)

Table *18-11 SDMMC* Register Overview

Name	Address Offset	Description
SDMA_SADDR	0x000	SDMA System Memory Address/ Argument2
BLK_SIZE_AND_CNT	0x004	Block Size and Block Count Register
ARGUMENT	0x008	Argument 1 Register
XFER_MODE_AND_CMD	0x00c	Transfer Mode and Command Register
RESP31_0	0x010	Response Bit 31-0 Regsiter
RESP63_32	0x014	Response Bit 63-32 Regsiter
RESP95_64	0x018	Response Bit 95-64 Regsiter
RESP127_96	0x01c	Response Bit 127-96 Regsiter
BUF_DATA	0x020	Buffer Data Port Register
PRESENT_STS	0x024	Present State Register
host_ctl1_pwr_bg_wup	0x028	Host Control 1 , Power, Block Gap and Wakeup Register
CLK_CTL_SWRST	0x02c	Clock and Reset Control Register
NORM_AND_ERR_INT_STS	0x030	Normal and Error Interrupt Status Register
NORM_AND_ERR_INT_STS_EN	0x034	Normal and Error Interrupt Status Enable Register
NORM_AND_ERR_INT_SIG_EN	0x038	Normal and Error Interrupt Signal Enable Register
AUTO_CMD_ERR_AND_HOST_CTL2	0x03c	Auto CMD Error Status Register and Host Control 2 register
CAPABILITIES1	0x040	Capabilities 1 Register
CAPABILITIES2	0x044	Capabilities 2 Register
FORCE_EVENT_ERR	0x050	Force Event Register for Auto CMD Error Status
ADMA_ERR_STS	0x054	ADMA Error Status Register
ADMA_SADDR_L	0x058	ADMA System Address Register for low 32-bit
ADMA_SADDR_H	0x05c	ADMA System Address Register for high 32-bit
PRESENT_VUL_INIT_DS	0x060	Present Value Register for Initialization and Default Speed
PRESENT_VUL_HS_SDR12	0x064	Present Value Register for High-speed and SDR12
PRESENT_VUL_SDR25_SDR50	0x068	Present Value Register for SDR25 and SDR50
PRESENT_VUL_SDR104_DDR50	0x06c	Present Value Register for SDR104 and DDR50
SLOT_INT_AND_HOST_VER	0x0fc	Slot Interrupt Status and Host Controller Version Register
EMMC_CTRL	0x200	MSHC Control register
EMMC_BOOT_CTL	0x204	eMMC Boot Control Register
CDET_TOUT_CTL	0x208	Card Detect Control Register
MBIU_CTRL	0x20c	MBIU Control register
PHY_TX_RX_DLY	0x240	PHY tx and rx delay line register
PHY_DS_DLY	0x244	PHY DS delay line register
PHY_DLY_STS	0x248	PHY delay line status register
PHY_CONFIG	0x24c	PHY Configuration register

18.4.4 register description

Detailed register descriptions are provided below.

SDMA_SADDR

SDMA System Memory Address/ Argument2
Offset Address: 0x000

Bits	Name	Access	Description	Reset
31:0	SDMA_SA	R/W	Physical system memory address used for DMA transfer and the second argument for Auto CMD23	0x0

BLK_SIZE_AND_CNT

Block Size and Block Count Register
Offset Address: 0x004

Bits	Name	Access	Description	Reset
11:0	XFER_BLK_SIZE	R/W	Block Size of data transfer. - 0x1 : 1 byte - 0x2 : 2 bytes ..... - 0x200 : 512 bytes ..... - 0x800 : 2048 bytes	0x0
14:12	SDMA_BUF_BDARY	R/W	Host SDMA buffer Boundary - 0x0 (4K bytes) - 0x1 (8K bytes) - 0x2 (16K bytes) - 0x3 (32K bytes) - 0x4 (64K bytes) - 0x5 (128K bytes) - 0x6 (256K bytes) - 0x7 (512K bytes)	0x0
15	Reserved			
31:16	BLK_CNT	R/W	Blocks Count for Current Transfer	0x0

ARGUMENT

Argument 1 Register
Offset Address:
0x008

Bits	Name	Access	Description	Reset
31:0	ARGUMENT	R/W	Command Argument 1	0x0

XFER_MODE_AND_CMD

Transfer Mode and Command
Register Offset Address: 0x00c

Bits	Name	Access	Description	Reset
0	DMA_ENABLE	R/W	DMA enable 1 : DMA Data Transfer 0 : No data transfer or Non DMA data transfer	0x0
1	BLK_CNT_ENABLE	R/W	Block Count Enable. This bit is used to enable the block count regiser, which is only relevant for multiple block transfers. 1 : Enable 0 : Disable	0x0

3:2	AUTO_CMD_ENABLE	R/W	Auto CMD enable. This field determines use of auto command funcitons. 0x0 : Auto command Disabled	0x0

Bits	Name	Access	Description	Reset
			0x1 : Auto CMD12 Enable 0x2 : Auto CMD23 Enable 0x3 : Reserved	
4	DAT_XFER_DIR	R/W	Data Transfer Direction Select 1 : Read (card to host) 0 : Write (host to card)	0x0
5	MULTI_BLK_SEL	R/W	Multi/single Block Select 1 : Multiple block transfer 0 : Single block transfer	0x0
6	RESP_TYPE	R/W	Response Type R1/R5 0x0 : R1 (Memory) 0x1 : R5 (SDIO)	0x0
7	RESP_ERR_CHK_ENABLE	R/W	Response Error Check Enable 1 : Enable 0 : Disable	0x0
8	RESP_INT_DISABLE	R/W	Response Interrupt Disable 1 : Disable 0 : Enable	0x0
15:9	Reserved			
17:16	RESP_TYPE_SEL	R/W	Response Type Select 0x0 : No Response 0x1 : Response Length 136 0x2 : Response Length 48 0x3 : Response Length 48 with busy	0x0
18	SUB_CMD_FLAG	R/W	Sub Command Flag 1 : Sub Command 0 : Main Command	0x0
19	CMD_CRC_CHK_ENABLE	R/W	Command CRC check enable 1 : Enable 0 : Disable	0x0
20	CMD_IDX_CHK_ENABLE	R/W	Command Index Check Enable 1 : Enable 0 : Disable	0x0
21	DATA_PRESENT_SEL	R/W	Data Present Select. It is set to 0 for the following. (1) Commands using only CMD line (ex. CMD52) (2) Commands with no data transfer but using busy signal on DAT0 (ex. R1b) (3) Resume command 1 : Data Present 0 : No Data Present	0x0
23:22	CMD_TYPE	R/W	Command Type 0x0 : Normal 0x1 : Suspend (CMD52 for writing "Bus Suspend" in CCCR) 0x2 : CMD52 for writing "Function Select" in CCCR) 0x3 : Abort (CMD12, CMD52 for writing) "I/O Abort" in CCCR)	0x0
29:24	CMD_IDX	R/W	Command Index	0x0
31:30	Reserved			

RESP31_0

Response Bit 31-0 Regsiter

Offset Address: 0x010

Bits	Name	Access	Description	Reset
31:0	RESP31_0	RO	Command Response for RSP [39:8]	

RESP63_32

Response Bit 63-32 Regsiter
Offset Address: 0x014

Bits	Name	Access	Description	Reset
31:0	RESP63_32	RO	Command Response for RSP [71:40]	

RESP95_64

Response Bit 95-64 Regsiter
Offset Address: 0x018

Bits	Name	Access	Description	Reset
31:0	RESP95_64	RO	Command Response for RSP [103:72]	

RESP127_96

Response Bit 127-96 Regsiter
Offset Address: 0x01c

Bits	Name	Access	Description	Reset
31:0	RESP127_96	RO	Command Response for RSP [135:104]	

BUF_DATA

Buffer Data Port
Register Offset Address:
0x020

Bits	Name	Access	Description	Reset
31:0	BUF_DATA	R/W	Buffer Data	0x0

PRESENT_STS

Present State Register
Offset Address: 0x024

Bits	Name	Access	Description	Reset
0	CMD_INHIBIT	RO	Command Inhibit (CMD) 1 : Cannot issue command 0 : Can issue command using only CMD line	
1	CMD_INHIBIT_DAT	RO	Command Inhibit (DAT) 1 : Cannot issue command wihich used the DAT line 0 : Can issue command using only DAT line	
2	DAT_LINE_ACTIVE	RO	DAT Line Active This bit indicates whether one of the DAT line on SD Bus is in use. 1 : DAT Line Active 0 : DAT Line Inactive	
3	RE_TUNE_REQ	RO	Re-Tuning Request 1 : Sampling clock needs re-tuning 0 : Fixed or well tuned sampling clock	
7:4	Reserved			
8	WR_XFER_ACTIVE	RO	Write Transfer Active 1 : Transferring data 0 : No valid data	
9	RD_XFER_ACTIVE	RO	Read Transfer Active 1 : Transferring data	

Bits	Name	Access	Description	Reset
			0 : No valid data	
10	BUF_WR_ENABLE	RO	Buffer Write Enable 1 : Enable 0 : Disable	
11	BUF_RD_ENABLE	RO	Buffer Read Enable 1 : Enable 0 : Disable	
15:12	Reserved			
16	CARD_INSERTED	RO	Card Inserted 1 : Card Inserted 0 : Reset or Debouncing or No card	
17	CARD_STABLE	RO	Card State Stable 1 : No Card or Inserted 0 : Reset or Debouncing	
18	CARD_CD_STS	RO	Card Detect Pin Level 1 : Card Present (SD_CD = 0) 0 : No Card Present (SD_CD = 1)	
19	CARD_WP_STS	RO	Write Protect Switch Pin Level 1 : Write enabled (SD_WP =0) 0 : Write protected (SD_WP = 1)	
23:20	DAT_3_0_STS	RO	DAT[3:0] Line Signal Level	
24	CMD_LINE_STS	RO	CMD Line Signal Level	
31:25	Reserved			

host_ctl1_pwr_bg_wup

Host Control 1 Register
Offset Address: 0x028

Bits	Name	Access	Description	Reset
0	LEC_CTL	R/W	LED Control This bit is used to caution the user not to remove the card while the SD card is being accessed. 1 : LED on 0 : LED off	0x0
1	DAT_XFER_WIDTH	R/W	Data Transfer Width. 1 : 4-bit mode 0 : 1-bit mode	0x0
2	HS_ENABLE	R/W	High Speed Enable 1 : High Speed Enable 0 : Normal Speed Enable	0x0
4:3	DMA_SEL	R/W	DMA Select. 0x0 : SDMA mode 0x1 : Reserved 0x2 : ADMA2 0x3 : ADMA2 or ADMA3	0x0
5	EXT_DAT_WIDTH	R/W	Extended Data Transfer Width 1 : 8-bit mode 0 : Selected by DAT_XFER_WIDTH	0x0
6	CRAD_DET_TEST	R/W	Card Detect Test Level 1 : Card Inserted 0 : No card	0x0
7	CARD_DET_SEL	R/W	Card Detect Signal Selection 1 : CARD_DET_TEST is selected (for test purpose) 0 : SD_CD Is selected	0x0
8	SD_BUS_PWR	R/W	SD Bus Power.	0x0

Bits	Name	Access	Description	Reset
			1 : Power on 0 : Power off	
11:9	SD_BUS_VOL_SEL	R/W	SD Bus Voltage Select 111b : 3.3V 110b : 3.0V 101b : 1.8V 100b - 000b : Reserved	0x0
15:12	Reserved			
16	STOP_BG_REQ	R/W	Stop At Block Gap Request. This bit is used to stop executing read and write transactions at the next block gap for non-DMA, SDMA and ADMA. transfers. 1: Stop 0 : Transfer	0x0
17	CONTINUE_REQ	R/W	Continue Request. This bit is used to restart a transaction, which was stopped using the STOP_BG_REQ. 1 : Restart 0 : Not affect	0x0
18	READ_WAIT	R/W	Read Wait Control 1 : Enable Read Wait Control 0 : Disable Read Wait Control	0x0
19	INT_BG	R/W	Interrupt At Block Gap 1 : Enable 0 : Disabel	0x0
23:20	Reserved			
24	WAKEUP_ON_CARD_INT	R/W	Wakeup Event Enable On Card Interrupt. 1 : Enable 0 : Disable	0x0
25	WAKEUP_ON_CARD_INSERT	R/W	Wakeup Event Enable On Card Insertion. 1 : Enable 0 : Disable	0x0
26	WAKEUP_ON_CARD_REMV	R/W	Wakeup Event Enable On Card Removal. 1 : Enable 0 : Disable	0x0
31:27	Reserved			

CLK_CTL_SWRST

Clock and Timeout Control Register
Offset Address: 0x02c

Bits	Name	Access	Description	Reset
0	INT_CLK_EN	R/W	Internal Clock Enable 1 : Oscillate 0 : Stop	0x0
1	INT_CLK_STABLE	RO	Internal Clock Stable. 1 : Ready 0 : Not Ready	
2	SD_CLK_EN	R/W	SD Clock Enable for Card 1 : Enable 0 : Disable	0x0

3	PLL_EN	R/W	PLL Enable 1 : Enable 0 : Disable	0x0

Bits	Name	Access	Description	Reset
5:4	Reserved			
7:6	UP_FREQ_SEL	R/W	Upper Bits of SDCLK Frequency Select	0x0
15:8	FREQ_SEL	R/W	SDCLK Frequency Select	0x0
19:16	TOUT_CNT	R/W	Data Timeout Counter Value 0x0 : TMCLK x 2^13 0x1 : TMCLK x 2^14 ..... 0xe : TMCLK x 2^ 27 0xf : Reserved	0x0
23:20	Reserved			
24	SW_RST_ALL	R/W	Software Reset For All	0x0
25	SW_RST_CMD	R/W	Software Reset For CMD Line	0x0
26	SW_RST_DAT	R/W	Software Reset For DATA Line	0x0
31:27	Reserved			

NORM_AND_ERR_INT_STS

Normal and Error Interrupt Status Register
Offset Address: 0x030

Bits	Name	Access	Description	Reset
0	CMD_CMPL	RWC	Command Complete	
1	XFER_CMPL	RWC	Transfer Complete	
2	BG_EVENT	RWC	Block Gap Event	
3	DMA_INT	RWC	DMA Interrupt	
4	BUF_WRDY	RWC	Buffer Write Ready	
5	BUF_RRDY	RWC	Buffer Read Ready	
6	CARD_INSERT_INT	RWC	Card Insertion	
7	CARD_REMOV_INT	RWC	Card Removal	
8	CARD_INT	RO	Card Interrupt	
9	INT_A	RO	INT_A. This status is set if INT_A is enabled and INT_A pin is in low level	
10	INT_B	RO	INT_B. This status is set if INT_B is enabled and INT_B pin is in low level	
11	INT_C	RO	INT_C. This status is set if INT_C is enabled and INT_C pin is in low level	
12	RE_TUNE_EVENT	RO	Re-Tuning Event	
13	Reserved			
14	CQE_EVENT	RO	Command Queuing Event	
15	ERR_INT	RO	Error Interrupt	
16	CMD_TOUT_ERR	RWC	Command Timeout Error	
17	CMD_CRC_ERR	RWC	Command CRC Error	
18	CMD_ENDBIT_ERR	RWC	Command End Bit Error	
19	CMD_IDX_ERR	RWC	Command Index Error	
20	DAT_TOUT_ERR	RWC	Data Timeout Error	
21	DAT_CRC_ERR	RWC	Data CRC Error	
22	DAT_ENDBIT_ERR	RWC	Data End Bit Error	

Bits	Name	Access	Description	Reset
23	CURR_LIMIT_ERR	RWC	Current Limit Error	
24	AUTO_CMD_ERR	RWC	Auto Command Error	
25	ADMA_ERR	RWC	ADMA Error	
26	TUNE_ERR	RWC	Tuning Error	
27	Reserved			
28	BOOT_ACK_ERR	RWC		
31:29	Reserved			

NORM_AND_ERR_INT_STS_EN

Normal and Error Interrupt Status Enable Register
Offset Address: 0x034

Bits	Name	Access	Description	Reset
0	CMD_CMPL_EN	R/W	Command Complete Status Enable	0x0
1	XFER_CMPL_EN	R/W	Transfer Complete Status Enable	0x0
2	BG_EVENT_EN	R/W	Block Gap Event Status Enable	0x0
3	DMA_INT_EN	R/W	DMA Interrupt Status Enable	0x0
4	BUF_WRDY_EN	R/W	Buffer Write Ready Status Enable	0x0
5	BUF_RRDY_EN	R/W	Buffer Read Ready Status Enabel	0x0
6	CARD_INSERT_INT_EN	R/W	Card Insertion Status Enable	0x0
7	CARD_REMOV_INT_EN	R/W	Card Removal Status Enable	0x0
8	CARD_INT_EN	R/W	Card Interrupt Status Enable	0x0
9	INT_A_EN	R/W	INT_A Status Enable.	0x0
10	INT_B_EN	R/W	INT_B Status Enable.	0x0
11	INT_C_EN	R/W	INT_C Status Enable.	0x0
12	RE_TUNE_EVENT_EN	R/W	Re-Tuning Event Status Enable	0x0
13	Reserved			
14	CQE_EVENT_EN	R/W	Command Queuing Event Status Enable	0x0
15	ERR_INT_EN	R/W	Error Interrupt Status Enable	0x0
16	CMD_TOUT_ERR_EN	R/W	Command Timeout Error Status Enable	0x0
17	CMD_CRC_ERR_EN	R/W	Command CRC Error Status Enable	0x0
18	CMD_ENDBIT_ERR_EN	R/W	Command End Bit Error Status Enable	0x0
19	CMD_IDX_ERR_EN	R/W	Command Index Error Status Enable	0x0
20	DAT_TOUT_ERR_EN	R/W	Data Timeout Error Status Enable	0x0
21	DAT_CRC_ERR_EN	R/W	Data CRC Error Status Enable	0x0
22	DAT_ENDBIT_ERR_EN	R/W	Data End Bit Error Statue Enable	0x0
23	CURR_LIMIT_ERR_EN	R/W	Current Limit Error Status Enable	0x0
24	AUTO_CMD_ERR_EN	R/W	Auto Command Error Status Enable	0x0
25	ADMA_ERR_EN	R/W	ADMA Error Status Enable	0x0
26	TUNE_ERR_EN	R/W	Tuning Error Status Enable	0x0
27	Reserved			
28	BOOT_ACK_ERR_EN	R/W	Boot Ack Error Status Enable	0x0
31:29	Reserved			

NORM_AND_ERR_INT_SIG_EN

Normal and Error Interrupt Signal Enable Register
Offset Address: 0x038

Bits	Name	Access	Description	Reset
0	CMD_CMPL_SIG_EN	R/W	Command Complete Signal Enable	0x0
1	XFER_CMPL_SIG_EN	R/W	Transfer Complete Signal Enable	0x0
2	BG_EVENT_SIG_EN	R/W	Block Gap Event Signal Enable	0x0
3	DMA_INT_SIG_EN	R/W	DMA Interrupt Signal Enable	0x0
4	BUF_WRDY_SIG_EN	R/W	Buffer Write Ready Signal Enable	0x0

Bits	Name	Access	Description	Reset
5	BUF_RRDY_SIG_EN	R/W	Buffer Read Ready Signal Enabel	0x0
6	CARD_INSERT_INT_SIG_EN	R/W	Card Insertion Signal Enable	0x0
7	CARD_REMOV_INT_SIG_EN	R/W	Card Removal Signal Enable	0x0
8	CARD_INT_SIG_EN	R/W	Card Interrupt Signal Enable	0x0
9	INT_A_SIG_EN	R/W	INT_A Signal Enable.	0x0
10	INT_B_SIG_EN	R/W	INT_B Signal Enable.	0x0
11	INT_C_SIG_EN	R/W	INT_C Signal Enable.	0x0
12	RE_TUNE_EVENT_SIG_EN	R/W	Re-Tuning EventSignal Enable	0x0
13	Reserved			
14	CQE_EVENT_SIG_EN	R/W	CQE EventSignal Enable	0x0
15	Reserved			
16	CMD_TOUT_ERR_SIG_EN	R/W	Command Timeout Error Signal Enable	0x0
17	CMD_CRC_ERR_SIG_EN	R/W	Command CRC Error Signal Enable	0x0
18	CMD_ENDBIT_ERR_SIG_EN	R/W	Command End Bit Error Signal Enable	0x0
19	CMD_IDX_ERR_SIG_EN	R/W	Command Index Error Signal Enable	0x0
20	DAT_TOUT_ERR_SIG_EN	R/W	Data Timeout Error Signal Enable	0x0
21	DAT_CRC_ERR_SIG_EN	R/W	Data CRC Error Signal Enable	0x0
22	DAT_ENDBIT_ERR_SIG_EN	R/W	Data End Bit Error Signal Enable	0x0
23	CURR_LIMIT_ERR_SIG_EN	R/W	Current Limit Error Signal Enable	0x0
24	AUTO_CMD_ERR_SIG_EN	R/W	Auto Command Error Signal Enable	0x0
25	ADMA_ERR_SIG_EN	R/W	ADMA Error Signal Enable	0x0
26	TUNE_ERR_SIG_EN	R/W	Tuning Error Signal Enable	0x0
27	Reserved			
28	BOOT_ACK_ERR_SIG_EN	R/W	Boot Ack Error Signal Enable	0x0
31:29	Reserved			

AUTO_CMD_ERR_AND_HOST_CTL2

Auto CMD Error Status Register and Host Control 2 register
Offset Address: 0x03c

Bits	Name	Access	Description	Reset
0	auto_cmd12_no_exe	RO	Auto CMD12 Not Executed	
1	AUTO_CMD_TOUT_ERR	RO	Auto CMD Timeout Error	
2	AUTO_CMD_CRC_ERR	RO	Auto CMD CRC Error	
3	AUTO_CMD_ENDBIT_ERR	RO	Auto CMD End Bit Error	
4	AUTO_CMD_IDX_ERR	RO	Auto CMD Index Error	
6:5	Reserved			
7	CMD_NOT_ISSUE_BY_CMD12	RO	Command Not Issued By Auto CMD12 Error	
15:8	Reserved			
18:16	UHS_MODE_SEL	R/W	USH Speed Mode Select (for SD) 0x0 : SDR12 0x1 : SDR25 0x2 : SDR50 0x3 : SDR104 0x4 : DDR50 0x5 : Reserved 0x6 : Reserved 0x7 : Reserved eMMC Speed Mode Select (for eMMC) 0x0 : Default speed 0x1 : High speed 0x2 : Reserved	0x0

			0x3 : HS200	

Bits	Name	Access	Description	Reset
			0x4 : DDR52 0x5 : Reserved 0x6 : Reserved 0x7 : Reserved	
19	EN_18_SIG	R/W	1.8V Signaling Enable	0x0
21:20	DRV_SEL	R/W	Driver Strength Select 0x0 : Driver Type B 0x1 : Driver Type A 0x2 : Driver Type C 0x3 : Driver Type D	0x0
22	EXECUTE_TUNE	R/W	Execute Tuning 1 : Execute Tuning 0 : Not Tuned or Tuning Completed	0x0
23	SAMPLE_CLK_SEL	R/W	Sampling Clock Select 1 : Tuned clock is used to sample data 0 : Fixed clock is used to sample data	0x0
29:24	Reserved			
30	ASYNC_INT_EN	R/W	Asynchronous Interrupt Enable. 1 : Enable 0 : Disable	0x0
31	PRESET_VAL_ENABLE	R/W	Preset Value Enable 1 : Automatic Selection by Preset Value are Enabled 0 : SDLCK and Driver Strength are controlled by Host Driver	0x0

CAPABILITIES1

Capabilities 1
Register Offset
Address: 0x040

Bits	Name	Access	Description	Reset
5:0	TOUT_CLK_FREQ	RO	Timeout Clock Frequency Not 0 : 1KHz~ 63KHz or 1Mhz~ 63Mhz	
6	Reserved			
7	TOUT_CLK_UNIT	RO	Timeout Clock Unit 1 : 1MHz 0 : 1KHz	
15:8	BASE_CLK_FREQ	RO	Base Clock Frequency for SD clock 0x0 : Get information through another method 0x1 : 1MHz 0x2 : 2MHz ..... 0xFF : 255Mhz	
17:16	MAX_BLK_LEN	RO	Max Block Length 0x0 : 512 (byte) 0x1 : 1024 0x2 : 2048 0x3 : Reserverd	
18	EMBEDDED_8BIT	RO	8-bit Support for Embedded Device	
19	ADMA2_SUPPORT	RO	ADMA2 Support	
20	Reserved			
21	HS_SUPPORT	RO	High Speed Support	

22	SDMA_SUPPORT	RO	SDMA Support	
23	SUSP_RES_SUPPORT	RO	Suspend/Resume Support	

Bits	Name	Access	Description	Reset
24	V33_SUPPORT	RO	3.3V Support	
25	V30_SUPPORT	RO	3.0V Support	
26	V18_SUPPORT	RO	1.8V Support	
27	Reserved			
28	BUS64_SUPPORT	RO	64-bit System Bus Support	
29	ASYNC_INT_SUPPORT	RO	Asynchronous Interrupt Support	
31:30	SLOT_TYPE	RO	Slot Type 0x0 : Removable Card 0x1 : Embedded Slot 0x2 : Shared Bus Slot	

CAPABILITIES2

Capabilities 2
Register Offset
Address: 0x044

Bits	Name	Access	Description	Reset
0	SDR50_SUPPORT	RO	SDR50 Support	
1	SDR104_SUPPORT	RO	SDR104 Support	
2	DDR50_SUPPORT	RO	DDR50 Support	
3	Reserved			
4	DRV_A_SUPPORT	RO	Driver Type A Support	
5	DRV_C_SUPPORT	RO	Driver Type C Support	
6	DRV_D_SUPPORT	RO	Driver Type D Support	
7	Reserved			
11:8	RETUNE_TIMER	RO	Timer Count for Re-Tuning 0x0 : Disable n : 2^(n-1) seconds 0xB : 1024 seconds 0xC ~ 0xE : Reserved 0xF : Get Information from other source	
12	Reserved			
13	TUNE_SDR50	RO	Use Tuning for SDR50	
15:14	RETUNE_MODE	RO	Re-Tuning Modes	
23:16	CLK_MULTIPLIER	RO	Clock Multiplier	
31:24	Reserved			

FORCE_EVENT_ERR

Force Event Register for Auto CMD Error Status
Offset Address: 0x050

Bits	Name	Access	Description	Reset
0	FORCE_AUTO_CMD12_NOT_EXE	R/W	Force Event for Auto CMD12 Not Executed	0x0
1	FORCE_AUTO_CMD_TOUT_ERR	R/W	Force Event for Auto CMD Timeout error	0x0
2	FORCE_AUTO_CMD_CRC_ERR	R/W	Force Event for Auto CMD CRC Error	0x0
3	FORCE_AUTO_CMD_EBIT_ERR	R/W	Force Event for Auto CMD End Bit Error	0x0
4	FORCE_AUTO_CMD_IDX_ERR	R/W	Force Event for Auto CMD Index Error	0x0
6:5	Reserved			

7	FORCE_AUTO_CMD_NOT_ISSUE	R/W	Force Event for Command Not Issued by Auto CMD12 Error	0x0

Bits	Name	Access	Description	Reset
15:8	Reserved			
16	FORCE_CMD_TOUT_ERR	R/W	Force Event for Auto CMD12 Not Executed	0x0
17	FORCE_CMD_CRC_ERR	R/W	Force Event for CMD Timeout Error	0x0
18	FORCE_CMD_EBIT_ERR	R/W	Force Event for CMD End Bit Error	0x0
19	FORCE_CMD_IDX_ERR	R/W	Force Event for CMD Index Error	0x0
20	FORCE_DAT_TOUT_ERR	R/W	Force Event for DATA Timeout Error	0x0
21	FORCE_DAT_CRC_ERR	R/W	Force Event for DATA End Bit Error	0x0
22	FORCE_DAT_EBIT_ERR	R/W	Force Event for DATA Index Error	0x0
23	FORCE_CURR_LIMIT_ERR	R/W	Force Event for current limit error	0x0
24	FORCE_AUTO_CMD_ERR	R/W	Force Event for Auto CMD Error	0x0
25	FORCE_ADMA_ERR	R/W	Force Event for ADMA Error	0x0
26	FORCE_TUNING_ERR	R/W	Force Event for Tuning Error	0x0
27	Reserved			
28	FORCE_BOOT_ACK_ERR	R/W	Force Event for Response Error	0x0
31:29	Reserved			

ADMA_ERR_STS

ADMA Error Status Register
Offset Address: 0x054

Bits	Name	Access	Description	Reset
1:0	ADMA_ERR_STS	RO	ADMA Error Status 0x0 : ST_STOP (Stop DMA) 0x1 : ST_FDS (Fetch Descriptor) 0x2 : Never set this state 0x3 : ST_TFR (transfer data)	
2	ADMA_LEN_MISMATCH	RO	ADMA Length Mismatch Error	
31:3	Reserved			

ADMA_SADDR_L

ADMA System Address Register for low 32-bit
Offset Address: 0x058

Bits	Name	Access	Description	Reset
31:0	ADMA_SA_L	R/W	ADMA System Address for low 32-bit	0x0

ADMA_SADDR_H

ADMA System Address Register for high 32-bit
Offset Address: 0x05c

Bits	Name	Access	Description	Reset
31:0	ADMA_SA_H	R/W	ADMA System Address for high 32-bit	0x0

PRESENT_VUL_INIT_DS

Present Value Register for Initialization and Default Speed
Offset Address: 0x060

Bits	Name	Access	Description	Reset
31:0	PRESENT_VUL_INIT_DS	RO	Present Value Register for Initialization and Default Speed	

PRESENT_VUL_HS_SDR12

Present Value Register for High-speed and SDR12
Offset Address: 0x064

Bits	Name	Access	Description	Reset
31:0	PRESENT_VUL_HS_SDR12	RO	Present Value Register for High-speed	

Bits	Name	Access	Description	Reset
			and SDR12	

PRESENT_VUL_SDR25_SDR50

Present Value Register for SDR25 and SDR50
Offset Address: 0x068

Bits	Name	Access	Description	Reset
31:0	PRESENT_VUL_SDR25_SDR50	RO	Present Value Register for SDR25 and SDR50	

PRESENT_VUL_SDR104_DDR50

Present Value Register for SDR104 and DDR50
Offset Address: 0x06c

Bits	Name	Access	Description	Reset
31:0	PRESENT_VUL_SDR104_DDR50	RO	Present Value Register for SDR104 and DDR50	

SLOT_INT_AND_HOST_VER

Slot Interrupt Status and Host Controller Version Register
Offset Address: 0x0fc

Bits	Name	Access	Description	Reset
7:0	INT_SLOT	RO	Interrupt Signal for Each Slot	
15:8	Reserved			
23:16	SPEC_VER	RO	Specification Version Number 00h : SD Host 1.00 01h : SD Host 2.00 02h : SD Host 3.00 03h : SD Host 4.00 04h : SD Host 4.10 05h : SD Host 4.20	
31:24	VENDOR_VER	RO	Verdor Version Number	

EMMC_CTRL

MSHC Control
register Offset
Address: 0x200

Bits	Name	Access	Description	Reset
0	EMMC_FUNC_EN	R/W	eMMC Card present	0x0
1	LATANCY_1T	R/W	Latancy 1t for cmd in	0x1
2	CLK_FREE_EN	R/W	Internal clock gating disable control	0x0
3	disable_data_crc_chk	R/W	Disable Data CRC Check	0x0
7:4	Reserved			
8	EMMC_RSTN	R/W	EMMC Device Reset Signal control	0x1
9	EMMC_RSTN_OEN	R/W	Output Enable control for EMMC Device Reset Signal PAD	0x1
11:10	Reserved			
12	CQE_ALGO_SEL	R/W	Scheduler algorithm selected for execution 1 : First come First serve (FCFS_ONLY) 0 : Priority based reordering with FCFS (PRI_REORDER_PLUS_FCFS)	0x0
13	CQE_PREFETCH_DISABLE	R/W	Enable or Disable CQE's PREFETCH Feature 1 : Disable 0 : Enable	0x0

Bits	Name	Access	Description	Reset
15:14	Reserved			
16	timer_clk_sel	R/W	Timer clock source selection 1 : 32K 0 : 100K	0x0
31:17	Reserved			

EMMC_BOOT_CTL

eMMC Boot Control Register
Offset Address: 0x204

Bits	Name	Access	Description	Reset
0	BOOT_MODE_ENABLE	R/W	Mandatory Boot Enable	0x0
1	BOOT_ACK_ENABLE	R/W	Boot Ack Enable	0x0
3:2	Reserved			
7:4	BOOT_TOUT_CNT	R/W	Boot Ack Timeout Counter Value	0x0
8	VALIDATE_BOOT	W	Validate Mandatory Boot Enable Bit	
31:9	Reserved			

CDET_TOUT_CTL

Card Detect Control
Register Offset Address:
0x208

Bits	Name	Access	Description	Reset
15:0	CDET_DEBUUNCE_CNT	R/W	card detect debounce counter	0x000F
31:16	Reserved			

MBIU_CTRL

MBIU Control
register Offset
Address: 0x20c

Bits	Name	Access	Description	Reset
0	UNDEFL_INCR_EN	R/W	Undefined INCR Burst	0x1
1	BURST_INCR4_EN	R/W	INCR4 Burst	0x1
2	BURST_INCR8_EN	R/W	INCR8 Burst	0x1
3	BURST_INCR16_EN	R/W	INCR16 Burst	0x1
31:4	Reserved			

PHY_TX_RX_DLY

PHY tx and rx delay line register
Offset Address: 0x240

Bits	Name	Access	Description	Reset
6:0	PHY_TX_DLY	R/W	PHY tx delay line phase selection	0x0
7	Reserved			
9:8	PHY_TX_SRC	R/W	PHY tx delay line clock source selection 2'b00 : clk_tx 2'b01 : inverse of clk_tx 2'b1x : reserved	0x0
10	PHY_TX_EVEN_ODD	R/W	PHY tx delay line clock source selection	0x0
15:11	Reserved			
22:16	PHY_RX_DLY	R/W	PHY rx delay line phase selection 2'b00 : clk_tx 2'b01 : inverse of clk_tx 2'b1x : reserved	0x0

23	Reserved			
25:24	PHY_RX_SRC	R/W	PHY rx delay line clock source selection	0x0

Bits	Name	Access	Description	Reset
26	PHY_RX_EVEN_ODD	R/W	PHY rx delay line clock source selection	0x0
31:27	Reserved			

PHY_DS_DLY

PHY DS delay line register
Offset Address: 0x244

Bits	Name	Access	Description	Reset
6:0	PHY_DS_DLY	R/W	PHY DS delay line phase selection	0x0
7	Reserved			
9:8	PHY_DS_SRC	R/W	PHY DS delay line clock source selection	0x0
10	PHY_DS_EVEN_ODD	R/W	PHY DS delay line clock source selection	0x0
31:11	Reserved			

PHY_DLY_STS

PHY delay line status register
Offset Address: 0x248

Bits	Name	Access	Description	Reset
0	PHY_TX_LEAD_LAG	RO	PHY tx delay line lead or lag flag	
1	PHY_RX_LEAD_LAG	RO	PHY rx delay line lead or lag flag	
2	PHY_DS_LEAD_LAG	RO	PHY ds delay line lead or lag flag	
31:3	Reserved			

PHY_CONFIG

PHY Configuration
register Offset Address:
0x24c

Bits	Name	Access	Description	Reset
0	PHY_TX_BPS	R/W	PHY tx data path bypass enable 0 : Pipe enable 1 : Bypass	0x1
1	ADJ_TIMING_EN	R/W	Adjust bus timing enable	0x0
7:2	Reserved			
9:8	ADJ_NCR	R/W	Adjust NCR counter	0x0
11:10	ADJ_NCRC	R/W	Adjust NCRC counter	0x0
31:12	Reserved			

18.5 GPIO

18.5.1 summarize

The system is configured with 4 groups of GPIOs (General Purpose Input/Output), GPIO0 to GPIO3.
programmable input and output pins.
The orientation of each pin can be arbitrarily set to either input or output for generating application-specific output signals or collecting application-specific input signals. When set as input pins, the GPIOs can be used as

interrupt sources; when set as output pins, each GPIO can output 0 or 1 independently.

GPIOs can generate maskable interrupts based on the level or trip value of the input signal.GPIOx_INTR_FLAG

(The (x=0~3) signal gives the interrupt controller an indication that an interrupt has occurred.

18.5.2 specificities

The orientation of each pin can be set arbitrarily as input or output.

- When set as an input pin, the GPIO can be used as an interrupt source.
- When set as an output pin, each GPIO can output 0 or 1 independently.

18.5.3 working method

18.5.3.1 interface reset

All four GPIO modules are reset at the same time when the chip is powered up or the system is reset, and the GPIO pins are in the input state by default after the reset.

18.5.3.2 General purpose inputs and outputs

Each pin can be arbitrarily set as an input or output as follows:

Step 1 Configure register GPIO_SWPORTA_DDR to set whether the GPIO is used as an input or an output.

When step 2 is configured as an input pin, read the GPIO_EXT_PORTA register to check the value of the input signal; when configured as an output pin, write the output value to the GPIO_SWPORTA_DR register to control the output power of the GPIO.

Ping.

18.5.3.3 interrupt operation

Each GPIO can be used as an interrupt source, which is controlled by 9 registers such as GPIO_INTEN. These registers allow the user to select the interrupt source, interrupt level polarity, and edge trigger characteristics.

When multiple GPIO interrupts occur at the same time, the interrupts will be unified and reported as a single interrupt (each of the 4 groups of GPIOs will be reported with a single aggregated interrupt flag).

The interrupt source characteristics and interrupt trigger category are determined by the five registers GPIO_INTTYPE_LEVEL, GPIO_INT_POLARITY, GPIO_INTMASK, GPIO_DEBOUNCE, and GPIO_LS_SYNC.

The original and masked state of the interrupt is read via GPIO_RAW_INTSTATUS and GPIO_INTSTATUS. The clearing of interrupt status can be controlled by setting GPIO_PORTA_EOI.

Each GPIO can support interrupts, and the setup procedure is as follows:

Step 1 Configure register GPIO_INTTYPE_LEVEL to select level trigger or edge trigger.

Step 2 Configure register GPIO_INT_POLARITY to select low/high level trigger and falling/rising edge trigger.

Step 3 Write 0xFFFFFFFF to register GPIO_PORTA_EOI to clear the interrupt.

Step 4 Configure the GPIO_INTEN register; enable the GPIO pin interrupt function.

18.5.4 GPIO Register Overview

The base addresses of the four GPIO modules on the chip are shown in Table 18-12.

Table *18-12* Base Addresses of the *4 GPIO* Modules on the Chip

GPIO Module	base address
GPIO0	0x03020000
GPIO1	0x03021000
GPIO2	0x03022000
GPIO3	0x03023000
RTCSYS_GPIO	0x05021000

Table 18-13 shows the offset addresses and definitions of the Group 1 GPIO module (GPIO0) registers; GPIO0 through GPIO3 have the same register definitions.

Table *18-13 GPIO* Register Overview

Name	Address Offset	Description
GPIO_SWPORTA_DR	0x000	Port A data register
GPIO_SWPORTA_DDR	0x004	Port A data direction register
GPIO_INTEN	0x030	Interrupt enable register
GPIO_INTMASK	0x034	Interrupt mask register
GPIO_INTTYPE_LEVEL	0x038	Interrupt level register
GPIO_INT_POLARITY	0x03c	Interrupt polarity register
GPIO_INTSTATUS	0x040	Interrupt status of Port A
GPIO_RAW_INTSTATUS	0x044	Raw interrupt status of Port A (pre-masking)
GPIO_DEBOUNCE	0x048	Debounce enable register
GPIO_PORTA_EOI	0x04c	Port A clear interrupt register
GPIO_EXT_PORTA	0x050	Port A external port register
GPIO_LS_SYNC	0x060	Level-sensitive synchronization enable register

18.5.5 GPIO Register Description

GPIO_SWPORTA_DR

Offset Address: 0x000

Bits	Name	Access	Description	Reset
31:0	GPIO_SWPORTA_DR	R/W	Values written to this register are output on the I/O signals for Port A if the corresponding data direction bits for Port A are set to Output mode and the corresponding control bit for Port A is set to Software mode. corresponding control bit for Port A is set to Software mode. The value read back is equal to the last value written to this register.	0x0

GPIO_SWPORTA_DDR

Offset Address: 0x004

Bits	Name	Access	Description	Reset
31:0	GPIO_SWPORTA_DDR	R/W	Values written to this register independently control the direction of the corresponding data bit in Port A. The default direction can be configured as input or output after system reset through the GPIO_DFLT_DIR_A parameter. 0 - Input (default) 1 - Output	0x0

GPIO_INTEN

Offset Address: 0x030

Bits	Name	Access	Description	Reset
31:0	GPIO_INTEN	R/W	Allows each bit of Port A to be By default the generation of interrupts is disabled. Whenever a 1 is written to a bit of this register, it configures the corresponding bit on Port A to become an interrupt; otherwise, Port A operates as a normal GPIO signal. Interrupts are disabled on the corresponding bits of Port A if the corresponding data direction register is set to Output or if Port A mode is set to Hardware. set to Hardware. 0 - Configure Port A bit as normal GPIO signal (default) 1 - Configure Port A bit as interrupt	0x0

GPIO_INTMASK

Offset Address: 0x034

Bits	Name	Access	Description	Reset
31:0	GPIO_INTMASK	R/W	Controls whether an interrupt on Port A can create an interrupt for the interrupt controller by not masking it. By default, all interrupts bits are By default, all interrupts bits are unmasked. Whenever a 1 is written to a bit in this register, it masks the interrupt generation capability for this signal; otherwise interrupts are allowed through. The unmasked status can be read as well as the resultant status after masking. 0 - Interrupt bits are unmasked (default) 1 - Mask interrupt	0x0

GPIO_INTTYPE_LEVEL

Offset Address: 0x038

Bits	Name	Access	Description	Reset

31:0	GPIO_INTTYPE_LEVEL	R/W	Controls the type of interrupt that can occur on Port A. Whenever a 0 is written to a bit of this register, it configures the interrupt type to be level-sensitive; otherwise, it is edge-sensitive.	0x0

Bits	Name	Access	Description	Reset
			0 - Level-sensitive (default) 1 - Edge-sensitive	

GPIO_INT_POLARITY

Offset Address: 0x03c

Bits	Name	Access	Description	Reset
31:0	GPIO_INT_POLARITY	R/W	Controls the polarity of edge or level sensitivity that can occur on input of Port A. Whenever a 0 is written to a bit of this register, it configures the interrupt type to falling-edge or active-low sensitive; otherwise, it is rising-edge or active-high sensitive. 0 - Active-low (default) 1 - Active-high	0x0

GPIO_INTSTATUS

Offset Address: 0x040

Bits	Name	Access	Description	Reset
31:0	GPIO_INTSTATUS	RO	Interrupt status of Port A	

GPIO_RAW_INTSTATUS

Offset Address: 0x044

Bits	Name	Access	Description	Reset
31:0	GPIO_RAW_INTSTATUS	RO	Raw interrupt of status of Port A (premasking bits)	

GPIO_DEBOUNCE

Offset Address: 0x048

Bits	Name	Access	Description	Reset
31:0	GPIO_DEBOUNCE	R/W	Controls whether an external signal that is the source of an interrupt needs to be debounced to remove any spurious glitches. Writing a 1 to a bit in this register enables the debouncing A signal must be valid for two periods of an external clock before it is internally processed. 0 - No debounce (default) 1 - Enable debounce	0x0

GPIO_PORTA_EOI

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
31:0	GPIO_PORTA_EOI	R/W	Controls the clearing of edge type interrupts from Port A. When a 1 is written into a corresponding bit of this register, the interrupt is cleared. All interrupts are cleared when Port A is not configured for interrupts. 0 - No interrupt clear (default) 1 - Clear interrupt	0x0

GPIO_EXT_PORTA

Offset Address: 0x050

Bits	Name	Access	Description	Reset
31:0	GPIO_EXT_PORTA	RO	When the data direction of Port A is set as Output, reading this location reads the values on the signal. When the data direction of Port A is set as Output, reading this location reads the data register for Port A.	

GPIO_LS_SYNC

Offset Address: 0x060

Bits	Name	Access	Description	Reset
0	GPIO_LS_SYNC	R/W	[0] Synchronization level Writing a 1 to this register results in all level-sensitive interrupts being synchronized to pclk_intr. 0 - No synchronization to pclk_intr (default) 1 - Synchronize to pclk_intr	0x0
31:1	Reserved			

18.6 USB DRD

18.6.1 summarize

The function of USB DRD is that it can act as Host or Device respectively, which can be c h a n g e d by software settings. The transfer protocol is in accordance with USB 2.0 specification, and the maximum transfer rate can be over 40MB/s; the main operation mode of Host/Device is scatter gather DMA, and the details will be described in the sections of Host and Device respectively; the functions of USB DRD are briefly listed below. The main operation modes of Host/Device are scatter gather DMA, the details will be described in the sections of Host and Device respectively; the functions of USB DRD are briefly listed as follows:

The Control Transfer

The Bulk Transfer

The Isochronous Transfer

The Host can be connected to a USB Hub and supports Interrupt Transfer.

The Passed USB Electrical Characterization Test (USBET) for good signal quality and compatibility.

18.6.2 Functional Description

18.6.2.1 system block diagram

The image below shows a block diagram of the system inside the USB DRD:

Figure *18-42 USB DRD* Internal System Block Diagram

18.6.2.2 functional characteristic

The functions of the USB DRD are summarized below:

algorithmic technology

The Compliant with USB2.0 transfer protocol specification
The Backward compatible with USB1.1 transport protocol specification
The Supports three speed modes HS/FS/LS
The Support for Host or Device Functions
The Supports four transfer modes specified by the USB protocol: control transfer, batch transfer, real-time transfer, and interrupt transfer.
The USB Hub can be connected to expand a single port to multiple USB ports
The Up to 127 Device Units via USB Hub Expansion
The Supports USB2.0 suspend/resume power saving mode.
The Supports HID devices such as keyboards and mice
Device mode is mainly used for downloading and updating internal software, but can also be used for other functions such as data transfer.
The Maximum transfer rate can be more than 40MB/s

18.6.3 USBC Function and Register Description

18.6.3.1 USBC Function Description

USB DRD can switch between Host or Device function, you can choose either one to use, but can not work at the same time, its function selection and management is controlled by the USBC block; in addition, Host and device have some Serial Bus events and interrupts triggered, will also place the buffer in this block.

18.6.3.2 USBC Register Summary

Name	Address Offset	Description
GOTGCTL	0x000	Control and Status Register
GOTGINT	0x004	Interrupt Register
GAHBCFG	0x008	AHB Configuration Register
GUSBCFG	0x00c	USB Configuration Register
GRSTCTL	0x010	Reset Register
GINTSTS	0x014	Interrupt Status Register
GINTMSK	0x018	Interrupt Mask Register
GUID	0x03c	User ID Register
GLPMCFG	0x054	Core LPM Configuration Register
GPWRDN	0x058	Power Down Register

18.6.3.3 USBC Register Detailed List

The memory addressing location of USBC is 0x0434_0000, which is represented in the text as USBC_BASE_ADDR. If you want to read or write to it,

the real addressing location in the memory space will be USBC_BASE_ADDR + Offset

The registers are represented by their corresponding relative addresses (Offset), which are described in detail below.

GOTGCTL

Control and Status
Register Offset Address:
0x000

Bits	Name	Access	Description	Reset
0	SesReqScs	RO	Mode: Device only Session Request Success (SesReqScs) The core sets this bit when a session request initiation is successful. ■ 1'b0: Session request failure ■ 1'b1: Session request success Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
1	SesReq	R/W	Mode: SRP-capable device Session Request (SesReq) The application sets this bit to initiate a session request on the USB. The application can clear this bit by writing a 0 when the Host Negotiation Success Status Change bit in the OTG Interrupt register (GOTGINT.HstNegSucStsChng) is SET. The core clears this bit when the HstNegSucStsChng bit is The core clears this bit when the HstNegSucStsChng bit is cleared. If you use the USB 1.1 Full-Speed Serial Transceiver interface to initiate the session request, the application must wait until the VBUS discharges to 0.2 V, after the B-Session Valid bit in this register (GOTGCTL.BSesVld) is cleared. This discharge time varies between different PHYs and can be obtained from the PHY vendor. ■ 1'b0: No session request ■ 1'b1: Session request Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	0x0
2	VbvalidOvEn	R/W	Mode: Host only VBUS Valid Override Enable (VbvalidOvEn) This bit is used to enable/disable the software to override the Bvalid signal using the GOTGCTL.VbvalidOvVal. ■ 1'b1: Internally Bvalid received from the PHY is overridden with GOTGCTL.VbvalidOvVal. ■ 1'b0: Override is disabled and bvalid signal from the respective PHY selected is used internally by the core. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0

3	VbvalidOvVal	R/W	Mode: Host only VBUS Valid Override Value (VbvalidOvVal) This bit is used to set Override value for vbusvalid signal when GOTGCTL.VbvalidOvEn is set. ■ 1'b0: vbusvalid value is 1'b0 when GOTGCTL.VbvalidOvEn =1	0x0

Bits	Name	Access	Description	Reset
			■ 1'b1: vbusvalid value is 1'b1 when GOTGCTL.VbvalidOvEn =1	
4	AvalidOvEn	R/W	Mode: Host only A-Peripheral Session Valid Override Enable (AvalidOvEn) This bit is used to enable/disable the software to override the Avalid signal using the GOTGCTL.AvalidOvVal. ■ 1'b1: Internally Avalid received from the PHY is overridden with GOTGCTL.AvalidOvVal. ■ 1'b0: Override is disabled and avalid signal from the respective PHY selected is used internally by the core.	0x0
5	AvalidOvVal	R/W	Mode: Host only A-Peripheral Session Valid Override Value (AvalidOvVal) This bit is used to set Override value for Avalid signal when GOTGCTL.AvalidOvEn is set. ■ 1'b0: Avalid value is 1'b0 when GOTGCTL.AvalidOvEn =1 ■ 1'b1: Avalid value is 1'b1 when GOTGCTL.AvalidOvEn =1	0x0
6	BvalidOvEn	R/W	Mode: Device only B-Peripheral Session Valid Override Enable (BvalidOvEn) This bit is used to enable/disable the software to override the Bvalid signal using the GOTGCTL.BvalidOvVal. ■ 1'b1: Internally Bvalid received from the PHY is overridden with GOTGCTL.BvalidOvVal. ■ 1'b0: Override is disabled and bvalid signal from the respective PHY selected is used internally by the force	0x0
7	BvalidOvVal	R/W	Mode: Device only B-Peripheral Session Valid Override Value (BvalidOvVal) This bit is used to set Override value for Bvalid signal when GOTGCTL.BvalidOvEn is set. ■ 1'b0: Bvalid value is 1'b0 when GOTGCTL.BvalidOvEn =1 ■ 1'b1: Bvalid value is 1'b1 when GOTGCTL.BvalidOvEn =1	0x0
8	HstNegScs	RO	Mode: HNP-capable device Host Negotiation Success (HstNegScs) The core sets this bit when host negotiation is successful. The core clears this bit when the HNP Request (HNPReq) bit in this register is set. ■ 1'b0: Host negotiation failure ■ 1'b1: Host negotiation success	
9	HNPReq	R/W	Mode: HNP Capable OTG Device HNP Request (HNPReq) The application sets this bit to initiate an HNP request to the connected USB host. The application can clear this bit by writing a 0 when the Host Negotiation Success Status Change bit in the OTG Interrupt register (GOTGINT.HstNegSucStsChng) is set. The core clears this bit when the HstNegSucStsChng bit is cleared.	0x0

Bits	Name	Access	Description	Reset
			■ 1'b0: No HNP request ■ 1'b1: HNP request	
10	HstSetHNPEn	R/W	Mode: HNP Capable OTG Host Host Set HNP Enable (HstSetHNPEn) The application sets this bit when it has successfully enabled HNP (using the SetFeature.SetHNPEnable command) on the connected device. ■ 1'b0: Host Set HNP is not enabled ■ 1'b1: Host Set HNP is enabled	0x0
11	DevHNPEn	R/W	Mode: HNP Capable OTG Device Device HNP Enabled (DevHNPEn) The application sets this bit when it successfully receives aSetFeature.SetHNPEnable command from the connected USB host. ■ 1'b0: HNP is not enabled in the application ■ 1'b1: HNP is enabled in the application	0x0
12	EHEn	R/W	Embedded Host Enable It is used to select between OTG A Device state Machine andEmbedded Host state machine. ■ 1'b1: Embedded Host State Machine is selected ■ 1'b0: OTG A Device state machine is selected Note: This field is valid only in SRP-Capable OTG Mode (OTG_MODE=0,1)	0x0
14:13	Reserved_00_14_13	RO	Reserved for future use.	
15	DbnceFltrBypass	R/W	Mode: Host and Device Debounce Filter Bypass Bypass Debounce filters for avalid, bvalid, vbusvalid, sessend, and iddig signals when enabled. ■ 1'b0: Disabled ■ 1'b1: Enabled	0x0
16	ConIDSts	RO	Mode: Host and Device Connector ID Status (ConIDSts) Indicates the connector ID status on a connect event. ■ 1'b0: The DWC_otg core is in A-Device mode ■ 1'b1: The DWC_otg core is in B-Device mode	
17	DbncTime	RO	Mode: Host only Long/Short Debounce Time (DbncTime) Indicates the debounce time of a detected connection. ■ 1'b0: Long debounce time, used for physical connections (100 ms +) 2.5 µs) ■ 1'b1: Short debounce time, used for soft connections (2.5 µs)	
18	ASesVld	RO	Mode: Host only A-Session Valid (ASesVld) Indicates the Host mode transceiver status. ■ 1'b0: A-session is not valid ■ 1'b1: A-session is valid	
19	BSesVld	RO	Mode: Device only B-Session Valid (BSesVld) Indicates the Device mode transceiver status.	

Bits	Name	Access	Description	Reset
			■ 1'b0: B-session is not valid. ■ 1'b1: B-session is valid. In OTG mode, you can use this bit to determine if the device is connected or disconnected.	
20	OTGVer	R/W	OTG Version (OTGVer) Indicates the OTG revision. ■ 1'b0: OTG Version 1.3. In this version the core supports Data line pulsing and VBus pulsing for SRP. ■ 1'b1: OTG Version 2.0. In this version the core supports only Data line pulsing for SRP.	0x0
21	CurMod_operation	RO	Current Mode of Operation (CurMod) Mode: Host and Device Indicates the current mode. ■ 1'b0: Device mode ■ 1'b1: Host mode	
26:22	MultValIdBC_operation	RO	Multi Valued ID pin (MultValIdBC) Mode: Host and Device Battery Charger ACA inputs in the following order. ■ Bit 26 - rid_float ■ Bit 25 - rid_gnd ■ Bit 24 - rid_a ■ Bit 23 - rid_b ■ Bit 22 - rid_c	
27	ChirpEn	RO	Chirp On Enable (ChirpEn) Mode: Device Only This bit when programmed to 1'b1 results in the core asserting chirp_on This bit is present before sending an actual Chirp "K" signal on USB. only if OTG_BC_SUPPORT = 1. If OTG_BC_SUPPORT!=1, this bit is a reserved bit.	
31:28	Reserved_00_31_28	RO	Reserved for future use.	

GOTGINT

Interrupt Register
Offset Address:
0x004

Bits	Name	Access	Description	Reset
1:0	Reserved_04_1_0	RO	Reserved for future use. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
2	SesEndDet	RWC	Write Behavior: One to clear Mode: Host and Device Session End Detected (SesEndDet) The core sets this bit when the utmiotg_bvalid signal is This bit can be set only by the core and the application should write 1 to clear it.	

7:3	Reserved_04_7_3	RO	Reserved for future use.

Bits	Name	Access	Description	Reset
8	SesReqSucStsChng	RWC	Write Behavior: One to clear Mode: Host and Device Session Request Success Status Change (SesReqSucStsChng) The core sets this bit on the success or failure of a session request. The application must read the Session Request Success bit in the OTG Control and Status register (GOTGCTL.SesReqScs) to check for success or failure.This bit can be set only by the core and the The application should write 1 to clear it.	
9	HstNegSucStsChng	RWC	Write Behavior: One to clear Mode: Host and Device Host Negotiation Success Status Change (HstNegSucStsChng) The core sets this bit on the success or failure of a USB host The application must read the Host Negotiation Request. Success bit of the OTG Control and Status register (GOTGCTL.HstNegScs) to check for success or This bit can be set only by the core and the application should write 1 to clear it.	
16:10	Reserved_04_16_10	RO	Reserved for future use.	
17	HstNegDet	RWC	Write Behavior: One to clear Mode: Host and Device Host Negotiation Detected (HstNegDet) The core sets this bit when it detects a host negotiation request on This bit can be set only by the core and the application should write 1 to clear it.	
18	ADevTOUTChg	RWC	Write Behavior: One to clear Mode: Host and Device A-Device Timeout Change (ADevTOUTChg) The core sets this bit to indicate that the A-device has timed out while waiting for the B-device to connect.This bit can be set only by the core and the application should write 1 to clear It.	

19	DbnceDone	RWC	Write Behavior: One to clear Mode: Host only Debounce Done (DbnceDone) The core sets this bit when the debounce is completed after the The application can start driving USB reset after This bit is only valid when the HNP Capable or SRP Capable bit is set in the Core USB Configuration register (GUSBCFG.HNPCap or GUSBCFG.SRPCap. This bit can be set only by the core and the application	

Bits	Name	Access	Description	Reset
			should write 1 to clear it.	
20	MultValIpChng	RWC	Write Behavior: One to clear This bit when set indicates that there is a change in the value of at The ACA pin value is at least one ACA pin value. This bit is present only if OTG_BC_SUPPORT=1, otherwise it is reserved.	
31:21	Reserved_04_31_21	RO	Reserved for future use. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	

GAHBCFG

AHB Configuration
Register Offset Address:
0x008

Bits	Name	Access	Description	Reset
0	GlblIntrMsk	R/W	Mode: Host and device Global Interrupt Mask (GlblIntrMsk) The application uses this bit to mask or unmask the interrupt line assertion Irrespective of this bit's setting, the interrupt status registers are updated by the core. ■ 1'b0: Mask the interrupt assertion to the application. ■ 1'b1: Unmask the interrupt assertion to the application. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	0x0

4:1	HBstLen	R/W	Mode: Host and device Burst Length/Type (HBstLen) This field is used in both External and Internal DMA modes. In External DMA mode, these bits appear on dma_burst[3:0] ports, which can be used by an external wrapper to interface the External DMA Controller interface to Synopsys DW_ahb_dmac or ARM PrimeCell. External DMA Mode defines the DMA burst length in terms of 32-bit words. ■ 4'b0000: 1 word ■ 4'b0001: 4 words ■ 4'b0010: 8 words ■ 4'b0011: 16 words ■ 4'b0100: 32 words ■ 4'b0101: 64 words ■ 4'b0110: 128 words ■ 4'b0111: 256 words ■ Others: Reserved Internal DMA Mode-AHB Master burst type. ■ 4'b0000 Single ■ 4'b0001 INCR 4'b0011 ■ INCR4 4'b0101 ■ INCR8 4'b0111 ■ INCR16	0x0

Bits	Name	Access	Description	Reset
			■ Others: Reserved	
5	DMAEn	R/W	Mode: Host and device DMA Enable (DMAEn) ■ 1'b0: Core operates in Slave mode ■ 1'b1: Core operates in a DMA mode This bit is always 0 when Slave-Only mode has been selected.	0x0
6	Reserved_08_6	RO	Reserved for future use. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
7	NPTxFEmpLvl	R/W	Mode: Host and device Non-Periodic TxFIFO Empty Level (NPTxFEmpLvl) This bit is used only in Slave mode. In host mode and with Shared FIFO with device mode, this bit indicates when the Non-Periodic TxFIFO Empty Interrupt bit in the Core Interrupt register (GINTSTS.NPTxFEmp) is triggered. With dedicated FIFO in device mode, this bit indicates when IN endpoint Transmit FIFO empty interrupt (DIEPINTn.TxFEmp) is triggered. Host mode and with Shared FIFO with device mode. ■ 1'b0: GINTSTS.NPTxFEmp interrupt indicates that the Non- Periodic TxFIFO is half empty ■ 1'b1: GINTSTS.NPTxFEmp interrupt indicates that the Non- Periodic TxFIFO is completely empty Dedicated FIFO in device mode. ■ 1'b0: DIEPINTn.TxFEmp interrupt indicates that the IN Endpoint TxFIFO is half empty ■ 1'b1: DIEPINTn.TxFEmp interrupt indicates that the IN Endpoint TxFIFO is completely empty	0x0
8	PTxFEmpLvl	R/W	Mode: Host only Periodic TxFIFO Empty Level (PTxFEmpLvl) Indicates when the Periodic TxFIFO Empty Interrupt bit in the Core is not available. Interrupt register (GINTSTS.PTxFEmp) is triggered. this bit is used only in Slave mode. ■ 1'b0: GINTSTS.PTxFEmp interrupt indicates that the Periodic TxFIFO is half empty ■ 1'b1: GINTSTS.PTxFEmp interrupt indicates that the Periodic TxFIFO is completely empty	0x0
20:9	Reserved_08_20_9	RO	Reserved for future use. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
21	RemMemSupp	R/W	Mode: Host and Device Remote Memory Support (RemMemSupp)	0x0

Bits	Name	Access	Description	Reset
			This bit is programmed to enable the functionality to wait for the system DMA Done Signal for the DMA Write Transfers. ■ GAHBCFG.RemMemSupp=1 The int_dma_req output signal is asserted when HSOTG DMA starts write transfer to the external memory. When the core is done with the Transfers it asserts int_dma_done signal to flag the completion of DMA The core then waits for sys_dma_done signal from the system to proceed further and complete the Data Transfer corresponding to a particular Channel/Endpoint. ■ GAHBCFG.RemMemSupp=0 The int_dma_req and int_dma_done signals are not asserted and the core proceeds with the assertion of the XferComp interrupt as soon as the DMA write transfer is done at the HSOTG Core Boundary and it doesn't wait for the sys_dma_done signal to complete the DATA transfers.	
22	NotiAllDmaWrit	R/W	Mode: Host and Device Notify all DMA Write Transactions (NotiAllDmaWrit) This bit is programmed to enable the System DMA Done functionality for all the DMA write Transactions corresponding to the Channel/Endpoint. this RemMemSupp is set to 1. bit is valid only when GAHBCFG. ■ GAHBCFG.NotiAllDmaWrit = 1 DWC_otg core asserts int_dma_req for all the DMA write transactions on the AHB interface along with int_dma_done, chep_last_transact and chep_number signal informations. the core waits for sys_dma_done signal for all the DMA write transactions in order to complete the transfer of a particular Channel/Endpoint. ■ GAHBCFG.NotiAllDmaWrit = 0 DWC_otg core asserts int_dma_req signal only for the last transaction of DMA write transfer corresponding to a particular Channel/Endpoint. Similarly, the core waits for sys_dma_done signal only for that transaction of DMA write to complete the transfer of a particular Channel/Endpoint.	0x0
23	AHBSingle	R/W	Mode: Host and Device AHBSingleSupport (AHBSingle) This bit when programmed supports Single transfers for the remaining data in a transfer when the DWC_otg core is operating in DMA mode.	0x0

Bits	Name	Access	Description	Reset
			■ 1'b0: This is the default mode. when this bit is set to 1'b0, the remaining data in the transfer is sent using INCR burst size. ■ 1'b1: When set to 1'b1, the remaining data in a transfer is sent using Single burst size. Note: If this feature is enabled, the AHB RETRY and SPLIT transfers still have INCR burst type. Enable this feature when the AHB Slave connected to the DWC_otg core does not support INCR burst (and when Split, and Retry transactions are not being used in the bus.)	
24	InvDescEndiannes s	R/W	Mode: Host and Device Inverse Descriptor Endianness ■ 1'b0: Descriptor endianness is similar to the AHB Master endianness ■ 1'b1. - If the AHB Master endianness is Big Endian, the Descriptor Endianness is Little Endian. - If the AHB Master endianness is Little Endian, the Descriptor Endianness is Big Endian.	0x0
31:25	Reserved_08_31_ 25	RO	Reserved for future use. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	

GUSBCFG

USB Configuration
Register Offset Address:
0x00c

Bits	Name	Access	Description	Reset

2:0	TOutCal	R/W	Mode: Host and Device HS/FS Timeout Calibration (TOutCal) The number of PHY clocks that the application programs in this field is added to the high-speed/full-speed interpacket timeout duration in the core to account for any additional delays introduced by the PHY. This can be required, because the delay introduced by the PHY in the generating the linestate condition can vary from one PHY to another. The USB standard timeout value for high-speed operation is 736 to 816 (inclusive) bit times. The USB standard timeout value for fullspeed operation is 16 to 18 (inclusive) bit times. The application must program this field based on the speed of enumeration. The number of bit times added per PHY clock are: High-speed operation. ■ One 30-MHz PHY clock = 16 bit times ■ One 60-MHz PHY clock = 8 bit times Full-speed operation. ■ One 30-MHz PHY clock = 0.4 bit times	0x0

Bits	Name	Access	Description	Reset
			■ One 60-MHz PHY clock = 0.2 bit times ■ One 48-MHz PHY clock = 0.25 bit times Using the HS as an example, if you set ToutCal to '001' you add one 30MHz PHY clock or 16 bit times. If you set ToutCal to '010' you add two 30MHz PHY clocks or 32 bit times, and so on. The 3 bits allow you to add up to 7 PHY clocks, and the number of bit times depend on the speed, and the PHY clock you are using. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
3	PHYIf	RO	Mode: Host and Device PHY Interface (PHYIf) The application uses this bit to configure the core to support a UTMI+ PHY with an 8- or 16-bit interface. When a ULPI PHY is chosen, this must be set to 8-bit mode. When a ULPI PHY is chosen, this must be set to 8-bit mode. ■ 1'b0: 8 bits ■ 1'b1: 16 bits This bit is writable only If UTMI+ and ULPI were selected. Otherwise, this bit returns the value for the power-on interface Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
4	ULPI_UTMI_Sel	R/W	Mode: Host and Device ULPI or UTMI+ Select (ULPI_UTMI_Sel) The application uses this bit to select either a UTMI+ interface or ULPI Interface. ■ 1'b0: UTMI+ Interface ■ 1'b1: ULPI Interface This bit is writable only If UTMI+ and ULPI was specified for High- Speed PHY Interface(s). Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
5	FSIntf	R/W	Mode: Host and Device Full-Speed Serial Interface Select (FSIntf) The application uses this bit to select either a unidirectional or bidirectional USB 1.1 full-speed serial transceiver interface. ■ 1'b0: 6-pin unidirectional full-speed serial interface ■ 1'b1: 3-pin bidirectional full-speed serial interface If a USB 1.1 Full-Speed Serial Transceiver interface was not selected, this bit is always 0, with Read Only access. If a USB 1.1 FS interface was selected, then the application can set this bit to select between the 3- and 6-pin interfaces, and access is Read and Write.	0x0
6	PHYSel	R/W	Mode: Host and Device	0x0

Bits	Name	Access	Description	Reset
			USB 2.0 High-Speed PHY or USB 1.1 Full-Speed Serial Transceiver Select (PHYSel) The application uses this bit to select either a high-speed UTMI+ or ULPI PHY, or a full-speed transceiver. ■ 1'b0: USB 2.0 high-speed UTMI+ or ULPI PHY ■ 1'b1: USB 1.1 full-speed serial transceiver If a USB 1.1 Full-Speed Serial Transceiver interface was not selected, this bit is always 0, with Read Only access. If a high-speed PHY interface was not selected, this bit is always 1. with Read Only access. If both interface types were selected (parameters have non-zero values), the application uses this bit to select which interface is active, and access is Read and Write.	
7	DDRSel	R/W	Mode: Host and Device ULPI DDR Select (DDRSel) The application uses this bit to select a Single Data Rate (SDR) or Double Data Rate (DDR) or ULPI interface. ■ 1'b0: Single Data Rate ULPI Interface, with 8-bit-wide data bus ■ 1'b1: Double Data Rate ULPI Interface, with 4-bit-wide data bus This bit is valid only when OTG_HSPHY_INTERFACE = 2 or 3.	0x0
8	SRPCap	R/W	Mode: Host and Device SRP-Capable (SRPCap) The application uses this bit to control the DWC_otg core SRP capabilities. If the core operates as a non-SRP-capable B- or B-subsystem, the application can use this bit to control the DWC_otg core SRP capabilities. device, it cannot request the connected A-device (host) to activate VBUS and start a session. ■ 1'b0: SRP capability is not enabled. ■ 1'b1: SRP capability is enabled. This bit is writable only if an SRP mode was specified for Mode of Operation in coreConsultant (parameter OTG_MODE). Otherwise, the reads return 0. If SRP functionality is disabled by the software, the OTG signals on the The PHY domain must be tied to the appropriate values.	0x0

9	HNPCap	R/W	Mode: Host and Device HNP-Capable (HNPCap) The application uses this bit to control the DWC_otg core's HNP capabilities. ■ 1'b0: HNP capability is not enabled. ■ 1'b1: HNP capability is enabled. This bit is writable only if an HNP mode was specified for Mode of Operation in coreConsultant (parameter OTG_MODE). Otherwise, the reads return 0.	0x0

Bits	Name	Access	Description	Reset
			If HNP functionality is disabled by the software, the OTG signals on The PHY domain must be tied to the appropriate values.	
13:10	USBTrdTim	R/W	Mode: Device only USB Turnaround Time (USBTrdTim) Sets the turnaround time in PHY clocks. specifies the response time for a MAC request to the Packet FIFO Controller (PFC) to fetch data from the DFIFO (SPRAM). This must be programmed to ■ 4'h5: When the MAC interface is 16-bit UTMI+. ■ 4'h9: When the MAC interface is 8-bit UTMI+. Note: The values above are calculated for the minimum AHB frequency of 30 MHz. USB turnaround time is critical for certification where long cables and 5-Hubs are used, so If you need the AHB to run at less than 30 MHz, and If USB turnaround time is not critical, then the USB turnaround time is not critical. These bits can be programmed to a larger value.	0x5
14	Reserved_0C_14	RO	Reserved for future use.	
15	PhyLPwrClkSel	R/W	Mode: Host and Device PHY Low-Power Clock Select (PhyLPwrClkSel) Selects either 480-MHz or 48-MHz (low-power) PHY mode. In FS and LS modes, the PHY can usually operate on a 48-MHz clock to Save power. ■ 1'b0: 480-MHz Internal PLL clock ■ 1'b1: 48-MHz External Clock In 480 MHz mode, the UTMI interface operates at either 60 or 30-Hz. MHz, depending upon whether 8- or 16-bit data width is selected. 48-MHz mode, the UTMI interface operates at 48 MHz in FS mode This bit drives the utmi_fsls_low_power core output signal and at either 48 or 6 MHz in LS mode (depending on the PHY vendor). This bit drives the utmi_fsls_low_power core output signal, which is the output signal of the utmi_fsls_low_power core. and is valid only For UTMI+ PHYs.	0x0
16	OtgI2CSel	R/W	Mode: Host and Device UTMIFS or I2C Interface Select (OtgI2CSel) The application uses this bit to select the I2C interface. ■ 1'b0: UTMI USB 1.1 Full-Speed interface for OTG signals ■ 1'b1: I2C interface for OTG signals This bit is writable only if I2C and UTMIFS were specified for Enable. I2C Interface? in coreConsultant (parameter OTG_I2C_INTERFACE) = 2). Otherwise, reads return 0.	0x0

17	ULPIFsLs	R/W	Mode: Host and Device ULPI FS/LS Select (ULPIFsLs) The application uses this bit to select the FS/LS serial interface for This bit is valid only when the FS serial transceiver is	0x0

Bits	Name	Access	Description	Reset
			selected on the ULPI PHY. ■ 1'b0: ULPI interface ■ 1'b1: ULPI FS/LS serial interface (Valid only when RTL parameters OTG_HSPHY_INTERFACE = 2 or 3 and OTG_FSPHY_INTERFACE = 1, 2, or 3) Before setting this bit, the application needs to ensure that GUSBCFG.ULPI_UTMI_SEL = 1'b1.	
18	ULPIAutoRes	R/W	Mode: Host and Device ULPI Auto Resume (ULPIAutoRes) This bit sets the AutoResume bit in the Interface Control register on the ULPI PHY. ■ 1'b0: PHY does not use AutoResume feature. ■ 1'b1: PHY uses AutoResume feature. (Valid only when RTL parameter OTG_HSPHY_INTERFACE = 2 or 3)	0x0
19	ULPIClkSusM	R/W	Mode: Host and Device ULPI Clock SuspendM (ULPIClkSusM) This bit sets the ClockSuspendM bit in the Interface Control register This bit applies only in serial or carkit modes. ■ 1'b0: PHY powers down internal clock during suspend. ■ 1'b1: PHY does not power down internal clock. (Valid only when RTL parameter OTG_HSPHY_INTERFACE = 2 or 3)	0x0
20	ULPIExtVbusDrv	R/W	Mode: Host only ULPI External VBUS Drive (ULPIExtVbusDrv) This bit selects between internal or external supply to drive 5V on VBUS, in ULPI PHY. ■ 1'b0: PHY drives VBUS using internal charge pump (Default). ■ 1'b1: PHY drives VBUS using external supply. (Valid only when RTL parameter OTG_HSPHY_INTERFACE = 2 or 3)	0x0
21	ULPIExtVbusIndicator	R/W	Mode: Host only ULPI External VBUS Indicator (ULPIExtVbusIndicator) This bit indicates to the ULPI PHY to use an external VBUS overcurrent indicator. ■ 1'b0: PHY uses internal VBUS valid comparator. ■ 1'b1: PHY uses external VBUS valid comparator. (Valid only when RTL parameter OTG_HSPHY_INTERFACE = 2 or 3)	0x0
22	TermSelDLPulse	R/W	Mode: Device only TermSel DLine Pulsing Selection (TermSelDLPulse) This bit selects utmi_termselect to drive data line pulse during SRP. ■ 1'b0: Data line pulsing using utmi_txvalid (Default). ■ 1'b1: Data line pulsing using utmi_termsel.	0x0
23	Complement	R/W	Mode: Host only Indicator Complement Controls the PHY to invert the ExternalVbusIndicator input signal, generating the Complement Output. For more information, refer to the ULPI Specification.	0x0

Bits	Name	Access	Description	Reset
			■ 1'b0: PHY does not invert ExternalVbusIndicator signal ■ 1'b1: PHY does invert ExternalVbusIndicator signal This bit is reserved and read-only when OTG_HSPHY_INTERFACE is set to 0 or 1.	
24	Indicator	R/W	Mode: Host only Indicator Pass Through Controls whether the Complement Output is qualified with the Internal Vbus Valid comparator before being used in the Vbus State For more information, refer to the ULPI Specification. ■ 1'b0: Complement Output signal is qualified with the Internal VbusValid comparator. ■ 1'b1: Complement Output signal is not qualified with the Internal VbusValid comparator. This bit is reserved and read-only when OTG_HSPHY_INTERFACE is set to 0 or 1.	0x0
25	ULPI	R/W	Mode: Host only ULPI Interface Protect Disable Controls circuitry built into the PHY For protecting the ULPI interface When the link tri-states STP and data. any pull-ups or pull-downs employed by this feature can be disabled. For more information, please contact your local government. refer to the ULPI Specification. ■ 1'b0: Enables the interface protect circuit ■ 1'b1: Disables the interface protect circuit This bit is reserved and read-only when OTG_HSPHY_INTERFACE is set to 0 or 1.	0x0
26	IC_USBCap	RO	Mode: Host and Device IC_USB-Capable (IC_USBCap) The application uses this bit to control the DWC_otg core's IC_USB capabilities. ■ 1'b0: IC_USB PHY Interface is not selected. ■ 1'b1: IC_USB PHY Interface is selected. This bit is writable only if OTG_ENABLE_IC_USB=1 and OTG_FSPHY_INTERFACE!=0. The reset value depends on the configuration parameter OTG_SELECT_IC_USB when OTG_ENABLE_IC_USB = 1. In all other cases, this bit is set to 1'b0 and the bit is read only.	
27	IC_USBTrafCtl	R/W	Mode: Device only IC_USB TrafficPullRemove Control (IC_USBTrafCtl) When this bit is set, pullup/pulldown resistors are detached from the USB during traffic signaling, per section 6.3.4 of the IC_USB specification. This bit is valid only when configuration parameter OTG_ENABLE_IC_USB = 1 and register field USBCFG.IC_USBCap is set to 1.	0x0
28	TxEndDelay	R/W	Mode: Device only	0x0

Bits	Name	Access	Description	Reset
			Tx End Delay (TxEndDelay) Writing 1'b1 to this bit enables the core to follow the TxEndDelay timings as per UTMI+ specification 1.05 section 4.1.5 for opmode signal during remote wakeup. ■ 1'b0: Normal Mode. ■ 1'b1: Tx End delay.	
29	ForceHstMode	R/W	Mode: Host and device Force Host Mode (ForceHstMode) Writing a 1 to this bit forces the core to host mode irrespective of utmiotg_iddig input pin. ■ 1'b0: Normal Mode. ■ 1'b1: Force Host Mode. After setting the force bit, the application must wait at least 25 ms Before the change to take effect. When the simulation is in scale down mode, waiting for 500 µs is sufficient. this bit is valid only When OTG_MODE = 0, 1 or 2. In all other cases, this bit reads 0.	0x0
30	ForceDevMode	R/W	Mode: Host and device Force Device Mode (ForceDevMode) Writing a 1 to this bit forces the core to device mode irrespective of utmiotg_iddig input pin. ■ 1'b0: Normal Mode. ■ 1'b1: Force Device Mode. After setting the force bit, the application must wait at least 25 ms Before the change to take effect. When the simulation is in scale down mode, waiting for 500 µs is sufficient. this bit is valid only When OTG_MODE = 0, 1 or 2. In all other cases, this bit reads 0.	0x0
31	CorruptTxPkt	R/W	Mode: Host and device Corrupt Tx packet (CorruptTxPkt) This bit is for debug purposes only. Never set this bit to 1. The application should always write 1'b0 to this bit.	0x0

GRSTCTL

Reset Register
Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	CSftRst	RO	Write Behavior: One to set Mode: Host and Device Core Soft Reset (CSftRst) Resets the hclk and phy_clock domains as follows. ■ Clears the interrupts and all the CSR registers except the following register bits. - PCGCCTL.RstPdwnModule - PCGCCTL.GateHclk - PCGCCTL.PwrClmp	

Bits	Name	Access	Description	Reset
			- PCGCCTL.StopPPhyLPwrClkSelclk - GUSBCFG.PhyLPwrClkSel - GUSBCFG.DDRSel - GUSBCFG.PHYSel - GUSBCFG.FSIntf - GUSBCFG.ULPI_UTMI_Sel - GUSBCFG.PHYIf - GUSBCFG.TxEndDelay - GUSBCFG.TermSelDLPulse - GUSBCFG.ULPIClkSusM - GUSBCFG.ULPIAutoRes - GUSBCFG.ULPIFsLs - GGPIO - GPWRDN - GADPCTL - HCFG.FSLSPclkSel - DCFG.DevSpd - DCTL.SftDiscon ■ All module state machines (except the AHB Slave Unit) are reset to the IDLE state, and all the transmit FIFOs and the receive FIFO are flushed. ■ Any transactions on the AHB Master are terminated as soon as possible, and the AHB Master will not be able to provide the necessary information to the AHB Master. after gracefully completing the last data phase of an AHB transfer. any transactions on the USB are terminated immediately. ■ When Hibernation or ADP feature is enabled, the PMU module is not reset by the Core Soft Reset. The application can write to this bit any time it wants to reset the core. self-clearing bit and the core clears this bit after all the necessary logic is reset in the core, which can take several clocks, depending on the current state of the After this bit is cleared, the application must wait at least 3 PHY for the core. clocks before doing any access to the PHY domain (synchronization delay). The application must also must check that bit 31 of this register is 1 (AHB). Master is IDLE) before starting any operation. Typically, software reset is used during software development and also when you dynamically change the PHY selection bits in the USB configuration When you change the PHY, the corresponding clock for The PHY is selected and used in the PHY domain. after a new clock is selected, the PHY domain has to be reset for proper operation. shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
1	PIUFSSftRst	RO	Write Behavior: One to set Mode: Host and Device	

Bits	Name	Access	Description	Reset
			PIU FS Dedicated Controller Soft Reset (PIUFSSftRst) Resets the PIU FS Dedicated Controller All module state machines in FS Dedicated Controller of PIU are reset to the IDLE state. Used to reset the FS Dedicated controller in PIU in case of any PHY Errors like Loss of activity or Babble Error resulting in the PHY remaining in RX state for more than one frame boundary. This is a self clearing bit and core clears this bit after all the necessary logic is Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
2	FrmCntrRst	RO	Write Behavior: One to set Mode: Host only Host Frame Counter Reset (FrmCntrRst) The application writes this bit to reset the (micro)frame number counter inside When the (micro)frame counter is reset, the subsequent SOF sent out by the core has a (micro)frame number of 0. If the application writes 1 to the bit, it may not be able to read back the value as It gets cleared by the core in a few clock cycles. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
3	INTknQFlsh	RWS	Mode: Device only IN Token Sequence Learning Queue Flush (INTknQFlsh) This bit is valid only if OTG_EN_DED_TX_FIFO = 0. The application writes this bit to flush the IN Token Sequence Learning Queue.	
4	RxFFlsh	RO	Write Behavior: One to set Mode: Host and Device RxFIFO Flush (RxFFlsh) The application can flush the entire RxFIFO using this bit, but must first ensure that that the core is not in the middle of a transaction. The application must only write to this bit after checking that the core is neither reading from the RxFIFO nor reading from the RxFIFO. nor writing to the RxFIFO. The application must wait until the bit is cleared before performing any other operations. This bit requires 8 clocks (slowest of PHY or AHB). clock) to clear.	
5	TxFFlsh	RWS	Write Behavior: One to set Mode: Host and Device TxFIFO Flush (TxFFlsh) This bit selectively flushes a single or all transmit FIFOs, but cannot do so if The core is in the midst of a transaction. The application must write this bit only after checking that the core is neither	

Bits	Name	Access	Description	Reset
			writing to the TxFIFO nor reading from the TxFIFO. Verify using these registers. ■ Read - NAK Effective Interrupt ensures the core is not reading from the FIFO ■ Write - GRSTCTL.AHBIdle ensures the core is not writing anything to the FIFO. Flushing is normally recommended when FIFOs are reconfigured or when switching between Shared FIFO and Dedicated Transmit FIFO operation. FIFO flushing is also recommended during device endpoint disable. The The application must wait until the core clears this bit before performing any This bit takes eight clocks to clear, using the slower clock of phy_clk or hclk.	
10:6	TxFNum	R/W	Mode: Host and Device TxFIFO Number (TxFNum) This is the FIFO number that must be flushed using the TxFIFO Flush bit. this field must not be changed until the core clears the TxFIFO Flush bit. ■ 5'h0. - Non-periodic TxFIFO flush in Host mode - Non-periodic TxFIFO flush in device mode when in shared FIFO operation - Tx FIFO 0 flush in device mode when in dedicated FIFO mode ■ 5'h1. - Periodic TxFIFO flush in Host mode - Periodic TxFIFO 1 flush in Device mode when in shared FIFO operation - TXFIFO 1 flush in device mode when in dedicated FIFO mode ■ 5'h2. - Periodic TxFIFO 2 flush in Device mode when in shared FIFO operation - TXFIFO 2 flush in device mode when in dedicated FIFO mode ... ■ 5'hF. - Periodic TxFIFO 15 flush in Device mode when in shared FIFO operation - TXFIFO 15 flush in device mode when in dedicated FIFO mode ■ 5'h10. - Flush all the transmit FIFOs in device or host mode.	0x0
29:11	Reserved_10_29_11	RO	Reserved for future use.	
30	DMAReq	RO	Mode: Host and Device DMA Request Signal (DMAReq) Indicates that the DMA request is in progress. Used for debug.	
31	AHBIdle	RO	Mode: Host and Device AHB Master Idle (AHBIdle) Indicates that the AHB Master State Machine is in the IDLE condition.	

GINTSTS

Interrupt Status Register
Offset Address: 0x014

Bits	Name	Access	Description	Reset
0	CurMod	RO	Mode: Host and Device Current Mode of Operation (CurMod) Indicates the current mode. ■ 1'b0: Device mode ■ 1'b1: Host mode Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
1	ModeMis	RWC	Write Behavior: One to clear Mode: Host and Device Mode Mismatch Interrupt (ModeMis) The core sets this bit when the application is trying to access. ■ A Host mode register, when the core is operating in Device mode ■ A Device mode register, when the core is operating in Host mode The register access is completed on the AHB with an OKAY response. but is ignored by the core internally and does not affect the operation of the core. This bit can be set only by the core and the application should write 1 Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
2	OTGInt	RO	Mode: Host and Device OTG Interrupt (OTGInt) The core sets this bit to indicate an OTG protocol event. The application must read the OTG Interrupt Status (GOTGINT) register to determine the exact event that caused this interrupt. the application must clear the appropriate status bit in the GOTGINT register to clear This bit. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
3	Sof	RWC	Write Behavior: One to clear Mode: Host and Device Start of (micro)Frame (Sof) In Host mode, the core sets this bit to indicate that an SOF (FS), micro-SOF (HS), or Keep-Alive (LS) is transmitted on the USB. The The application must write a 1 to this bit to clear the interrupt. In Device mode, in the core sets this bit to indicate that an SOF token has been received on the USB. The application can read the Device	

Bits	Name	Access	Description	Reset
			Status register to get the current (micro)Frame number. this interrupt is seen only when the core is operating at either HS or FS.This bit can be set only by the core and the application must write 1 to clear it. Note: The register may return 1'b1 if read immediately after power on If the register bit reads 1'b1 immediately after power on reset, it does not indicate that an SOF has been sent (in host mode), or SOF has been received (in device mode). The read value of this interrupt is valid only after a valid connection between host and device is established. If the bit is set after power on reset, the application can clear the bit.	
4	RxFLvl	RO	Mode: Host and Device RxFIFO Non-Empty (RxFLvl) Indicates that there is at least one packet pending to be read from the RxFIFO.	
5	NPTxFEmp	RO	Mode: Host and Device Non-periodic TxFIFO Empty (NPTxFEmp) This interrupt is asserted when the Non-periodic TxFIFO is either half or completely empty, and there is space for at least one entry to be written to the Non-periodic Transmit Request Queue. The half or completely empty status is determined by the Non-periodic TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl). In host mode, the application can use GINTSTS.NPTxFEmp with the OTG_EN_DED_TX_FIFO parameter set to either 1 or 0. In device mode, the application uses GINTSTS.NPTxFEmp when OTG_EN_DED_TX_FIFO=0. When OTG_EN_DED_TX_FIFO=1, the application uses GINTSTS. the application uses DIEPINTn.TxFEmp.	
6	GINNakEff	RO	Mode: Device only Global IN Non-periodic NAK Effective (GINNakEff) Indicates that the Set Global Non-periodic IN NAK bit in the Device Control register (DCTL.SGNPInNak) set by the application has taken That is, the core has sampled the Global IN NAK bit This bit can be cleared by clearing the Clear Global Non-periodic IN NAK bit in the Device Control register (DCTL.CGNPInNak). This interrupt does not necessarily mean that a NAK handshake is sent out on the USB. the STALL bit takes precedence over the NAK bit.	

7	GOUTNakEff	RO	Mode: Device only

Bits	Name	Access	Description	Reset
			Global OUT NAK Effective (GOUTNakEff) Indicates that the Set Global OUT NAK bit in the Device Control register (DCTL.SGOUTNak) set by the application has taken effect in This bit can be cleared by writing the Clear Global OUT NAK bit in the Device Control register (DCTL.CGOUTNak).	
8	ULPICKINT_I2CCKINT	RWC	Write Behavior: One to clear Mode: Host and Device ULPI Carkit Interrupt (ULPICKINT) The core sets this interrupt when a ULPI Carkit interrupt is received. The core's PHY sets ULPI Carkit interrupt in UART or Audio mode. This field is used only if the Carkit interface was enabled in coreConsultant (parameter OTG_ULPI_CARKIT = 1). Otherwise, reads return 0. I2C Carkit Interrupt (I2CCKINT) The core sets this interrupt when a Carkit interrupt is received. The core's PHY sets the I2C Carkit interrupt in Audio mode. This field is used only if the I2C interface was enabled in coreConsultant (parameter OTG_I2C_ INTERFACE = 1). INTERFACE = 1). Otherwise, the reads return 0.	
9	I2CINT	RWC	Write Behavior: One to clear Mode: Host and Device I2C Interrupt (I2CINT) The core sets this interrupt when I2C access is completed on the I2C interface. This field is used only if the I2C interface was enabled in coreConsultant (parameter OTG_I2C_INTERFACE = 1). Otherwise, the reads return 0.	
10	ErlySusp	RWC	Write Behavior: One to clear Mode: Device only Early Suspend (ErlySusp) The core sets this bit to indicate that an Idle state has been detected. on the USB For 3 ms.	
11	USBSusp	RWC	Write Behavior: One to clear Mode: Device only USB Suspend (USBSusp) The core sets this bit to indicate that a suspend was detected on the USB. The core enters the Suspend state when there is no activity on the linestate signal for an extended period of time.	
12	USBRst	RWC	Write Behavior: One to clear Mode: Device only USB Reset (USBRst) The core sets this bit to indicate that a reset is detected on the USB.	
13	EnumDone	RWC	Write Behavior: One to clear Mode: Device only Enumeration Done (EnumDone)	

Bits	Name	Access	Description	Reset
			The core sets this bit to indicate that speed enumeration is complete. The application must read the Device Status (DSTS) register to obtain the enumerated speed.	
14	ISOOutDrop	RWC	Write Behavior: One to clear Mode: Device only Isochronous OUT Packet Dropped Interrupt (ISOOutDrop) The core sets this bit when it fails to write an isochronous OUT packet. into the RxFIFO because the RxFIFO does not have enough space to accommodate a maximum packet size packet for the isochronous OUT endpoint.	
15	EOPF	RWC	Write Behavior: One to clear Mode: Device only End of Periodic Frame Interrupt (EOPF) Indicates that the period specified in the Periodic Frame Interval field of the Device Configuration register (DCFG.PerFrInt) has been reached in the current microframe.	
16	RstrDoneInt	RWC	Mode: Host and Device Restore Done Interrupt (RstrDoneInt) The core sets this bit to indicate that the Restore command after Hibernation was completed by the core. The core continues from Suspend state into the mode dictated by the PCGCCTL.RestoreMode field. This bit is valid only when the Hibernation feature is enabled. (OTG_EN_PWRPOPT=2).	
17	EPMis	RO	Write Behavior: One to clear Mode: Device only Endpoint Mismatch Interrupt (EPMis) Note: This interrupt is valid only in shared FIFO operation. Indicates that an IN token has been received for a non-periodic endpoint, but the data for another endpoint is present in the top of the Non-periodic Transmit FIFO and the IN endpoint mismatch count programmed by the application has expired.	
18	IEPInt	RO	Mode: Device only IN Endpoints Interrupt (IEPInt) The core sets this bit to indicate that an interrupt is pending on one of the IN endpoints of the core (in Device mode). The application must read the Device All Endpoints Interrupt (DAINT) register to determine the exact number of the IN endpoint on Device IN Endpoint-n Interrupt (DIEPINTn) register to determine the exact cause of the interrupt. the The application must clear the appropriate status bit in the corresponding DIEPINTn register to clear this bit.	
19	OEPInt	RO	Mode: Device only	

Bits	Name	Access	Description	Reset
			OUT Endpoints Interrupt (OEPInt) The core sets this bit to indicate that an interrupt is pending on one of the OUT endpoints of the core (in Device mode). The application must read the Device All Endpoints Interrupt (DAINT) register to determine the exact number of the OUT endpoint on which the interrupt occurred, and Then read the corresponding Device OUT Endpoint-n Interrupt (DOEPINTn) register to determine the exact cause of the The application must clear the appropriate status bit in the corresponding DOEPINTn register to clear this bit.	
20	incompISOIN	RWC	Write Behavior: One to clear Mode: Device only Incomplete Isochronous IN Transfer (incompISOIN) The core sets this interrupt to indicate that there is at least one isochronous IN endpoint on which the transfer is not completed in the This interrupt is asserted along with the End of Periodic Frame Interrupt (EOPF) bit in this register. Note: This interrupt is not asserted in Scatter/Gather DMA. mode.	
21	incomplP_incompISOOUT	RWC	Write Behavior: One to clear Incomplete Periodic Transfer (incomplP) Mode: Host only In Host mode, the core sets this interrupt bit when there are incomplete periodic transactions still pending which are scheduled for the current microframe. Incomplete Isochronous OUT Transfer (incompISOUT) Mode: Device only In Device mode, the core sets this interrupt to indicate that there is at least one isochronous OUT endpoint on which the transfer is not This interrupt is asserted along with the End of Periodic Frame Interrupt (EOPF) bit in this register.	
22	FetSusp	RWC	Write Behavior: One to clear Mode: Device only Data Fetch Suspended (FetSusp) This interrupt is valid only in DMA mode. The core has stopped fetching data For IN endpoints due to the unavailability of TxFIFO space or Request Queue space. This interrupt is used by the application For an endpoint mismatch algorithm. application For an endpoint mismatch algorithm. For example, after detecting an endpoint mismatch, the application. ■ Sets a Global non-periodic IN NAK handshake ■ Disables In endpoints	

Bits	Name	Access	Description	Reset
			■ Flushes the FIFO ■ Determines the token sequence from the IN Token Sequence Learning Queue ■ Re-enables the endpoints ■ Clears the Global non-periodic IN NAK handshake If the Global non-periodic IN NAK is cleared, the core has not yet fetched data for the IN endpoint, and the IN token is received. The core generates an 'IN token received when FIFO empty' interrupt. DWC_otg then sends the host a NAK response. To avoid this scenario, the host must be able to send the host a NAK response. The application can check the GINTSTS.FetSusp interrupt, which ensures that the FIFO is full before clearing a Global NAK handshake. Alternatively, the application can mask the "IN token received when FIFO empty" interrupt when clearing a Global IN NAK Handshake.	
23	ResetDet	RWC	Write Behavior: One to clear Mode: Device only Reset detected Interrupt (ResetDet) In Device mode, this interrupt is asserted when a reset is detected on the USB in partial power-down mode when the device is in Suspend. In Host mode, this interrupt is not asserted.	
24	PrtInt	RO	Mode: Host only Host Port Interrupt (PrtInt) The core sets this bit to indicate a change in port status of one of the DWC_otg core ports in Host mode. The application must read the Host Port Control and Status (HPRT) register to determine the exact The application must clear the event that caused this interrupt. appropriate status bit in the Host Port Control and Status register to clear this bit.	

25	HChInt	RO	Mode: Host only Host Channels Interrupt (HChInt) The core sets this bit to indicate that an interrupt is pending on one of The application must read the channels of the core (in Host mode). The application must read the Host All Channels Interrupt (HAINT) register to determine the exact number of the channel on which the interrupt occurred and then read the corresponding Host Channel-n Interrupt (HCINTn) register to The application must clear the exact cause of the interrupt. the appropriate status bit in the HCINTn register to clear this	

Bits	Name	Access	Description	Reset
			bit.	
26	PTxFEmp	RO	Mode: Host only Periodic TxFIFO Empty (PTxFEmp) This interrupt is asserted when the Periodic Transmit FIFO is either half or completely empty and there is space for at least one entry to be written in the Periodic Request Queue. The half or completely empty status is determined by the Periodic TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.PTxFEmpLvl).	
27	LPM_Int	RWC	Write Behavior: One to clear Mode: Host and Device LPM Transaction Received Interrupt (LPM_Int) ■ Device Mode - This interrupt is asserted when the device receives an LPM transaction and responds with a non-ERRORed response. ■ Host Mode - This interrupt is asserted when the device responds to an LPM transaction with a non-ERRORed response or when the host core has completed LPM transactions for the programmed number of times (GLPMCFG.RetryCnt).	
28	ConIDStsChng	RWC	Write Behavior: One to clear Mode: Host and Device Connector ID Status Change (ConIDStsChng) The core sets this bit when there is a change in connector ID status.	
29	DisconnInt	RWC	Write Behavior: One to clear Mode: Host only Disconnect Detected Interrupt (DisconnInt) Asserted when a device disconnect is detected.	
30	SessReqInt	RWC	Write Behavior: One to clear Mode: Host and Device Session Request/New Session Detected Interrupt (SessReqInt) In Host mode, this interrupt is asserted when a session request is detected from the device. In Device mode, this interrupt is asserted when the utmisrp_bvalid The signal goes high.	
31	WkUpInt	RWC	Write Behavior: One to clear Mode: Host and Device Resume/Remote Wakeup Detected Interrupt (WkUpInt) Wakeup Interrupt during Suspend(L2) or LPM(L1) state. ■ During Suspend (L2). - Device Mode - This interrupt is asserted only when Host Initiated Resume is detected on USB. - Host Mode - This interrupt is asserted only when Device Initiated Remote Wakeup is detected on USB. ■ During LPM (L1). - Device Mode - This interrupt is asserted for either Host Initiated Resume or Device Initiated Remote Wakeup on USB.	

Bits	Name	Access	Description	Reset
			- Host Mode - This interrupt is asserted for either Host Initiated Resume or Device Initiated Remote Wakeup on USB.	

GINTMSK

Interrupt Mask
Register Offset
Address: 0x018

Bits	Name	Access	Description	Reset
0	Reserved_18_0	RO	Reserved for future use. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
1	ModeMisMsk	R/W	Mode: Host and Device Mode Mismatch Interrupt Mask (ModeMisMsk) Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	0x0
2	OTGIntMsk	R/W	Mode: Host and Device OTG Interrupt Mask (OTGIntMsk) Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
3	SofMsk	R/W	Mode: Host and Device Start of (micro)Frame Mask (SofMsk)	0x0
4	RxFLvlMsk	R/W	Mode: Host and Device Receive FIFO Non-Empty Mask (RxFLvlMsk)	0x0
5	NPTxFEmpMsk	R/W	Mode: Host and Device Non-periodic TxFIFO Empty Mask (NPTxFEmpMsk)	0x0
6	GINNakEffMsk	R/W	Mode: Device only Global Non-periodic IN NAK Effective Mask (GINNakEffMsk)	0x0
7	GOUTNakEffMsk	R/W	Mode: Device only Global OUT NAK Effective Mask (GOUTNakEffMsk)	0x0
8	ULPICKINTMsk_I2CCKINTMsk	R/W	ULPI Carkit Interrupt Mask (ULPICKINTMsk) Mode: Host and Device I2C Carkit Interrupt Mask (I2CCKINTMsk) Mode: Host and Device	0x0
9	I2CIntMsk	R/W	Mode: Host and Device I2C Interrupt Mask (I2CIntMsk)	0x0
10	ErlySuspMsk	R/W	Mode: Device only Early Suspend Mask (ErlySuspMsk)	0x0
11	USBSuspMsk	R/W	Mode: Device only USB Suspend Mask (USBSuspMsk)	0x0
12	USBRstMsk	R/W	Mode: Device only USB Reset Mask (USBRstMsk)	0x0
13	EnumDoneMsk	R/W	Mode: Device only Enumeration Done Mask (EnumDoneMsk)	0x0
14	ISOOutDropMsk	R/W	Mode: Device only Isochronous OUT Packet Dropped Interrupt Mask (ISOOutDropMsk)	0x0
15	EOPFMsk	R/W	Mode: Device only End of Periodic Frame Interrupt Mask (EOPFMsk)	0x0

16	RstrDoneIntMsk	R/W	Mode: Host and Device Restore Done Interrupt Mask (RstrDoneIntMsk)	0x0

Bits	Name	Access	Description	Reset
			This field is valid only when the Hibernation feature is enabled. (OTG_EN_PWROPT=2).	
17	EPMisMsk	R/W	Mode: Device only Endpoint Mismatch Interrupt Mask (EPMisMsk)	0x0
18	IEPIntMsk	R/W	Mode: Device only IN Endpoints Interrupt Mask (IEPIntMsk)	0x0
19	OEPIntMsk	R/W	Mode: Device only OUT Endpoints Interrupt Mask (OEPIntMsk)	0x0
20	incompISOINMsk	R/W	Mode: Device only Incomplete Isochronous IN Transfer Mask (incompISOINMsk) This bit is enabled only when device periodic endpoints are enabled in Dedicated TxFIFO mode.	0x0
21	incomplPMsk_incompISOOUTMsk	R/W	Incomplete Periodic Transfer Mask (incomplPMsk) Mode: Host only Incomplete Isochronous OUT Transfer Mask (incompISOOUTMsk) Mode: Device only	0x0
22	FetSuspMsk	R/W	Mode: Device only Data Fetch Suspended Mask (FetSuspMsk)	0x0
23	ResetDetMsk	R/W	Mode: Device only Reset detected Interrupt Mask (ResetDetMsk)	0x0
24	PrtIntMsk	R/W	Mode: Host only Host Port Interrupt Mask (PrtIntMsk)	0x0
25	HChIntMsk	R/W	Mode: Host only Host Channels Interrupt Mask (HChIntMsk)	0x0
26	PTxFEmpMsk	R/W	Mode: Host only Periodic TxFIFO Empty Mask (PTxFEmpMsk)	0x0
27	LPM_IntMsk	R/W	Mode: Host and Device LPM Transaction received interrupt Mask (LPM_IntMsk)	0x0
28	ConIDStsChngMsk	R/W	Mode: Host and Device Connector ID Status Change Mask (ConIDStsChngMsk)	0x0
29	DisconnIntMsk	R/W	Mode: Host and Device Disconnect Detected Interrupt Mask (DisconnIntMsk)	0x0
30	SessReqIntMsk	R/W	Mode: Host and Device Session Request/New Session Detected Interrupt Mask (SessReqIntMsk)	0x0
31	WkUpIntMsk	R/W	Mode: Host and Device Resume/Remote Wakeup Detected Interrupt Mask (WkUpIntMsk) The WakeUp bit is used for LPM state wake up in a way similar to that of wake up in suspend state.	0x0

GUID

User ID Register
Offset Address:
0x03c

Bits	Name	Access	Description	Reset
31:0	UserID	R/W	User ID (UserID) Application-programmable ID field. Reset: Configurable	0x0

GLPMCFG

Core LPM Configuration
Register Offset Address: 0x054

Bits	Name	Access	Description	Reset
0	LPMCap	R/W	Mode: Host and Device LPM-Capable (LPMCap) The application uses this bit to control the DWC_otg core LPM capabilities. If the core operates as a non-LPM-capable host, it cannot request the connected device or hub to activate LPM mode. If the core operates as a non-LPM-capable device, it cannot respond to any LPM transactions. ■ 1b0: LPM capability is not enabled ■ 1b1: LPM capability is enabled This bit is writable only if an LPM mode was specified for Mode of Operation in coreConsultant (parameter OTG_ENABLE_LPM). Otherwise, reads return 0. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
1	AppL1Res	R/W	Mode: Device only LPM response programmed by application (AppL1Res) Handshake response to LPM token pre-programmed by device application The response depends on GLPMCFG.LPMCap. if GLPMCFG.LPMCap is 1'b0, then the core always responds with NYET response. If GLPMCFG.LPMCap is 1'b1, the core response is as follows. ■ 1: ACK Even though ACK is pre-programmed, the core Device responds with ACK The LPM transaction is successful if. - No PID/CRC5 Errors in either EXT token or LPM token (else ERROR) - Valid bLinkState = 0001B (L1) received in LPM transaction (else STALL) - No data pending in transmit queue (else NYET). ■ 0: NYET The pre-programmed software bit is over-ridden for response to LPM token when. - The received bLinkState is not L1 (STALL response), or - An error is detected in either of the LPM token packets because of corruption (ERROR response). Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	0x0
5:2	HIRD	R/W	Mode: Host and Device ■ EnBESL = 1'b0 Host-Initiated Resume Duration (HIRD) Host Mode: The value of HIRD to be sent in an LPM transaction. This value is also used to initiate resume for a duration TL1HubDrvResume1 for host initiated	0x0

algorithmic technology

Bits	Name	Access	Description	Reset
			resume. Device Mode (Read-Only): This field is updated with the Received LPM Token HIRD bmAttribute when an ACK, NYET, or STALL response is sent to an LPM transaction. Sl. No HIRD[3:0] THIRD (µs) 1 4'b0000 50 2 4'b0001 125 3 4'b0010 200 4 4'b0011 275 5 4'b0100 350 6 4'b0101 425 7 4'b0110 500 8 4'b0111 575 9 4'b1000 650 10 4'b1001 725 11 4'b1010 800 12 4'b1011 875 13 4'b1100 950 14 4'b1101 1025 15 4'b1110 1100 16 4'b1111 1175 Reset: 4'b0000 ■ EnBESL = 1'b1 Best Effort Service Latency (BESL) Host Mode: The value of BESL to be sent in an LPM transaction. This value is also used to initiate resume for a duration TL1HubDrvResume1 for host initiated resume. Device Mode (Read-Only): This field is updated with the Received LPM Token BESL bmAttribute when an ACK, NYET, or STALL response is sent to an LPM transaction. Sl. No BESL[3:0] TBESL (µs) 1 4'b0000 125 2 4'b0001 150 3 4'b0010 200 4 4'b0011 300 5 4'b0100 400 6 4'b0101 500 7 4'b0110 1000 8 4'b0111 2000 9 4'b1000 3000 10 4'b1001 4000 11 4'b1010 5000 12 4'b1011 6000 13 4'b1100 7000 14 4'b1101 8000 15 4'b1110 9000 16 4'b1111 10000 Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
6	bRemoteWake	R/W	Mode: Host and Device	0x0

Bits	Name	Access	Description	Reset
			RemoteWakeEnable (bRemoteWake) Host Mode: The value of remote wake up to be sent in the wIndex field of LPM transaction. Device Mode (Read-Only): This field is updated with the Received LPM Token bRemoteWake bmAttribute when an ACK, NYET, or STALL response is sent to an LPM transaction.	
7	EnblSlpM	R/W	Mode: Host and Device Enable utmi_sleep_n (EnblSlpM) ULPI Interface: The application uses this bit to control the utmi_sleep_n assertion to the PHY when in L1 state. For the host, this bit is valid only in "local". device" mode. ■ 1b0: utmi_sleep_n assertion from the core is not transferred to the external PHY. ■ 1b1: utmi_sleep_n assertion from the core is transferred to the external PHY. Note: When a ULPI interface is configured, enabling this bit results in a write to Bit 7 of the ULPI Function Control register. The Synopsys ULPI PHY supports writing to this bit, and in the L1 state asserts SleepM when utmi_l1_suspend_n cannot be asserted. Other Interfaces: The application uses this bit to control utmi_sleep_n assertion For the host, this bit is valid only in Local Device mode. ■ 1'b0: utmi_sleep_n assertion from the core is not transferred to the external PHY. ■ 1'b1: utmi_sleep_n assertion from the core is transferred to the external PHY when utmi_l1_suspend_n cannot be asserted.	0x0
12:8	HIRD_Thres	R/W	Mode: Host and Device BESL or HIRD Threshold (HIRD_Thres) Device Mode. ■ EnBESL = 1'b0: The core puts the PHY into deep low power mode in L1 (by core asserting L1SuspendM) when HIRD value is greater than or equal to the value defined in this field HIRD_Thres[3:0] and HIRD_Thres[4] is set to 1'b1. ■ EnBESL = 1'b1: The core puts the PHY into deep low power mode in L1 (by core asserting L1SuspendM) when BESL value is greater than or equal to the value defined in this field BESL_Thres[3:0] and BESL_Thres[4] is set to 1'b1. ■ DCTL.DeepSleepBESLReject = 1'b1: In device initiated resume, the core expects the Host to resume service to the device within the BESL value	0x0

Bits	Name	Access	Description	Reset
			corresponding to L1 exit time specified in HIRD_Thresh[3:0]. The Device sends a NYET response when the received HIRD in LPM token is greater than HIRD threshold. ■ Note: To differentiate between Deep Sleep and Shallow sleep HIRD greater The HIRD threshold comparison is done. For differentiating between NYET or ACK response for LPM token HIRD greater than HIRD Threshold comparison is used. Host Mode: The core puts the PHY into deep low power mode in L1 (by core). asserting L1SuspendM) when HIRD_Thres[4] is set to 1b1. HIRD_Thres[3:0] specifies the time for which resume signaling is to be reflected by host (TL1HubDrvResume2) on the USB bus when it detects device initiated resume. HIRD_Thres must not be programmed with a value greater than 4'b1100 in Host mode, because this exceeds maximum TL1HubDrvResume2. Sl. No ----Thres[3:0] Host Mode Resume Signaling Time (µs) --- - --- - --- - --- - --------- - --- - EnBESL = 1'b0 --- EnBESL = 1'b1 1 4'b0000 60 75 2 4'b0001 135 100 3 4'b0010 -210 150 4 4'b0011 285 250 5 4'b0100 360 350 6 4'b0101 435 450 7 4'b0110 510 950 8 4'b0111 585 Invalid 9 4'b1000 660 Invalid 10 4'b1001 735 Invalid 11 4'b1010 810 Invalid 12 4'b1011 885 Invalid 13 4'b1100 960 Invalid 14 4'b1101 Invalid Invalid 15 4'b1110 Invalid Invalid 16 4'b1111 Invalid Invalid The following truth table explains the difference in behavior between the UTMI and the UTMI. and ULPI interface in different modes of operation. Bit 7 --Bit 6 ---sleep_n ---l1_suspend_n --suspend_n -- Mode of Operation 0 -- ---- 1--- --- 1 - -- -- 1 - -- -------- -- 1 - -- ------- Normal Operation 0 -- ---- 0--- --- 1 - -- -- 1 - -- -------- -- 0 1 1 - -------- L2 Suspend 1 -- ---- 0--- --- 1 - -- -- 0 1 -- -------- -- 1 - -- -------- L1 Deep Sleep 1 -- ---- 1--- --- 0 - -- -- 1 - -- -------- -- 1 - -- -------- L1 Shallow Sleep	

14:13	CoreL1Res	RO	Mode: Host and Device LPM Response (CoreL1Res) Device Mode: The response of the core to LPM transaction	

Bits	Name	Access	Description	Reset
			received is reflected in these two bits. Host Mode: Handshake response received from local device for LPM transaction ■ 11 - ACK ■ 10 - NYET ■ 01 - STALL ■ 00 - ERROR (No handshake response)	
15	SlpSts	RO	Mode: Device only Port Sleep Status (SlpSts) This bit is set as long as a Sleep condition is present on the USB bus. The core is set as long as a Sleep condition is present on the USB bus. enters the Sleep state when an ACK response is sent to an LPM transaction and The TL1TokenRetry timer has expired. To stop the PHY clock, the application must set the Port Clock Stop bit, which asserts the PHY Suspend input signal. The application must rely on SlpSts and not ACK in CoreL1Res to confirm that the application has not been implemented. transition into sleep. The core comes out of sleep. ■ When there is any activity on the USB linestate ■ When the application writes to the Remote Wakeup Signaling bit in the Device Control register (DCTL.RmtWkUpSig) or when the application resets or softdisconnects the device. Host Mode: The host transitions to Sleep (L1) state as a side-effect of a successful LPM transaction by the core to the local port with ACK response from The read value of this bit reflects the current Sleep status of the port. The core clears this bit after. ■ The core detects a remote L1 Wakeup signal. ■ The application sets the Port Reset bit or the Port L1Resume bit in the HPRT register, or ■ The application sets the L1Resume/ Remote Wakeup Detected Interrupt bit or Disconnect Detected Interrupt bit in the Core Interrupt register (GINTSTS.L1WkUpInt or GINTSTS.DisconnInt, respectively). Values. ■ 1b0: Core not in L1 ■ 1b1: Core in L1	

16	L1ResumeOK	RO	Mode: Host and device Sleep State Resume OK (L1ResumeOK) Indicates that the application or host can start resume from Sleep state. This bit is valid in LPM sleep (L1) state. It is set in sleep mode after a delay of 50 µs (TL1Residency). This bit is reset when SlpSts is 0. ■ 1b1: The application or core can start Resume from Sleep state	

Bits	Name	Access	Description	Reset
			■ 1b0: The application or core cannot start Resume from Sleep state	
20:17	LPM_Chnl_Indx	R/W	Mode: Host only LPM Channel Index (LPM_Chnl_Indx) The channel number on which the LPM transaction has to be applied while sending an LPM transaction to the local device. Based on the LPM channel index. The core automatically inserts the device address and end point number. programmed in the corresponding channel into the LPM transaction.	0x0
23:21	LPM_Retry_Cnt	R/W	Mode: Host only LPM Retry Count (LPM_Retry_Cnt) When the device gives an ERROR response, this is the number of additional LPM retries that the host performs until a valid device response (STALL, NYET, or ACK) is received.	0x0
24	SndLPM	RWS	Write Behavior: One to set Mode: Host only Send LPM Transaction (SndLPM) When the application software sets this bit, an LPM transaction containing two tokens, EXT and LPM is sent. The hardware clears this bit once a valid response (STALL, NYET, or ACK) is received from the Device or the core has finished transmitting the programmed number of LPM retries. Note: This bit must be set only when the host is connected to a local port.	
27:25	LPM_RetryCnt_Sts	RO	Mode: Host only LPM Retry Count Status (LPM_RetryCnt_Sts) Number of LPM Host Retries still remaining to be transmitted for the current LPM sequence.	
28	EnBESL	R/W	Mode: Host and device Enable Best Effort Service Latency (BESL) This bit enables the BESL feature as defined in the LPM errata. ■ 1'b0: The core works as described in the following document. USB 2.0 Link Power Management Addendum Engineering Change Notice to the USB 2.0 specification, July 16, 2007 ■ 1'b1: The core works as per the LPM Errata	0x0

29	RstrSlpSts	R/W	Mode: Device only Restore SlpSts (RstrSlpSts) When the application power-gates the core (partial power-down or hibernation), the application will not be able to use it. The application needs to program this bit to restore the LPM status in the core. Based on the BESL value received from the Host, the application needs to The application should program this bit during restore process. depending on whether it decided to put the core in Shallow Sleep (Clock Gating)	0x0

Bits	Name	Access	Description	Reset
			Only) or Deep Sleep (Power Gating) mode. ■ 1'b0: The application puts the core in Shallow Sleep mode based on the BESL value from the Host. ■ 1'b1: The application puts the core in Deep Sleep mode based on the BESL value from the Host.	
30	HSICCon	R/W	Mode: Host and device HSIC-Connect (HSICCon) The application must use this bit to initiate the HSIC Attach sequence. Host Mode: Once this bit is set, the Host Core configures to drive HSIC Idle state (STROBE=1&DATA=0) on the bus. It then waits for device to initiate the HSIC Connect sequence. Device Mode: Once this bit is set, the Device Core waits for HSIC Idle linestate After receiving the Idle linestate it then initiates the HSIC Connect. This bit is valid only if OTG_ENABLE_HSIC = 1, if_sel_hsic = 1, and InvSelHSIC = Otherwise, it is read-only.	0x0
31	InvSelHsic	R/W	Mode: Host and device HSIC-Invert Select HSIC (InvSelHsic) The application uses this bit to control the DWC_otg core HSIC enable/disable. This bit overrides and functionally inverts the if_sel_hsic input port signal. If the core is non-HSIC-capable, it can connect to only PHYs that are not HSIC capable. If the core is HSIC-capable, it can connect only to PHYs that are HSIC capable. ■ If if_sel_hsic input signal is 1. - InvSelHsic = 1b1: HSIC capability is not enabled - InvSelHsic = 1b0: HSIC capability is enabled ■ If if_sel_hsic input signal is 0. - InvSelHsic = 1b1: HSIC capability is enabled - InvSelHsic = 1b0: HSIC capability is not enabled This bit is writable only if HSIC mode is specified for Mode of Operation in coreConsultant (parameter OTG_ENABLE_HSIC). This bit is valid only if OTG_ENABLE_HSIC is enabled. Otherwise, reads return 0.	0x0

GPWRDN

Power Down Register
Offset Address: 0x058

Bits	Name	Access	Description	Reset
0	PMUIntSel	R/W	Mode: Host and Device PMU Interrupt Select (PMUIntSel) When the hibernation functionality is selected (OTG_EN_PWROPT = 2), a write to this bit with 1'b1 enables the PMU to generate interrupts During this state, all interrupts from the	0x0

Bits	Name	Access	Description	Reset
			DWC_otg_core modules are blocked to the application. Note: This bit must be set to 1'b1 before the core is put into hibernation. ■ 1'b0: Internal DWC_otg_core interrupt is selected ■ 1'b1: External DWC_otg_pmu interrupt is selected Note: This bit must not be written to during normal mode of operation. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
1	PMUActv	R/W	Mode: Host and Device PMU Active (PMUActv) This bit enables or disables the PMU logic. ■ 1'b0: Disable PMU module ■ 1'b1: Enable PMU module Note: This bit must not be written to during normal mode of operation. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	0x0
2	Restore	R/W	Mode: Host and Device Restore The application must program this bit to enable or disable restore mode from the PMU module. ■ 1'b0: DWC_otg in normal mode of operation ■ 1'b1: DWC_otg in restore mode Note: This bit must not be written to during normal mode of operation. This bit is valid only when OTG_EN_PWROPT = 2. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
3	PwrDnClmp	R/W	Mode: Host and Device Power Down Clamp (PwrDnClmp) The application must program this bit to enable or disable the clamps to all the outputs of the DWC_otg core module to prevent the corruption of other active logic. ■ 1'b0: Disable PMU power clamp ■ 1'b1: Enable PMU power clamp	0x0
4	PwrDnRst_n	R/W	Mode: Host and Device Power Down ResetN (PwrDnRst_n) The application must program this bit to reset the DWC_otg core during the Hibernation exit process or during ADP when powering up the core (if the DWC_otg core was powered off during ADP process). ■ 1'b1: DWC_otg is in normal operation ■ 1'b0: Reset DWC_otg Note: This bit must not be written to during normal mode of operation.	0x1
5	PwrDnSwtch	R/W	Mode: Host and Device	0x0

Bits	Name	Access	Description	Reset
			Power Down Switch (PwrDnSwtch) This bit indicates to the DWC_otg core whether the VDD switch is in ON or OFF state. ■ 1'b0: DWC_otg is in ON state ■ 1'b1: DWC_otg is in OFF state Note: This bit must not be written to during normal mode of operation.	
6	DisableVBUS	R/W	Mode: Host and Device DisableVBUS Host Mode. The application must program this bit if HPRT0.PrtPwr was programmed to 0 before switching off the Core. this indicates to the PMU whether session was ended before entering Hibernation. ■ 1'b0: HPRT0.PrtPwr was not programmed to 0. ■ 1'b1: HPRT0.PrtPwr was programmed to 0. Device Mode. The application must program this bit to inform the PMU whether the The bvalid valid signal is high (session valid) or low (session end) whenever the core is switched off. ■ 1'b0: bvalid signal is High (Session Valid) ■ 1'b1: bvalid signal is Low (Session End) This bit is valid only when GPWRDN.PMUActv is 1.	0x0
7	LnStsChng	RWC	Write Behavior: One to clear Mode: Host and Device Line State Change (LnStsChng) This interrupt is asserted when there is a linestate change detected by The application must read GPWRDN.Linestate to determine the current linestate on USB. ■ 1'b0: No LineState change on USB ■ 1'b1: LineState change on USB This bit is valid only when GPWRDN.PMUActv is 1 and OTG_EN_PWROPT = 2.	
8	LineStageChange Msk	R/W	Mode: Host and Device Mask For LineStateChange interrupt (LineStageChangeMsk) This bit is valid only when OTG_EN_PWROPT = 2.	0x0
9	ResetDetected	RWC	Write Behavior: One to clear Mode: Device only ResetDetected This field indicates that Reset has been detected by the PMU module. This field generates an interrupt. ■ 1'b0: Reset Not Detected ■ 1'b1: Reset Detected This bit is valid only when OTG_EN_PWROPT = 2.	
10	Mask_ResetDetM sk	R/W	Mode: Device only Mask For ResetDetected interrupt (ResetDetMsk) This bit is valid only when OTG_EN_PWROPT = 2.	0x0
11	DisconnectDetect	RWC	Write Behavior: One to clear Mode: Host only DisconnectDetect This field indicates that Disconnect has been detected by the	

Bits	Name	Access	Description	Reset
			PMU. This field generates an interrupt. After detecting disconnect during hibernation the application must not restore the core, but instead start the initialization process. ■ 1'b0: Disconnect not detected ■ 1'b1: Disconnect detected This bit is valid only when OTG_EN_PWROPT = 2.	
12	DisconnectDetect Msk	R/W	Mode: Host only Mask For DisconnectDetect Interrupt (DisconnectDetectMsk) This bit is valid only when OTG_EN_PWROPT = 2.	0x0
13	ConnectDet	RO	Mode: Host and Device Write Behavior: One to clear ConnectDet This field indicates that a new connect has been detected. ■ 1'b0: Connect not detected ■ 1'b1: Connect detected This bit is valid only when OTG_EN_PWROPT = 2.	
14	ConnDetMsk	R/W	Mode: Host and Device ConnDetMsk Mask for ConnectDet interrupt This bit is valid only when OTG_EN_PWROPT = 2.	0x0
15	SRPDetect	RWC	Mode: Host only SRPDetect This field indicates that SRP has been detected by the PMU. This field After detecting SRP during hibernation the The application must not restore the core. the initialization process. ■ 1'b0: SRP not detected ■ 1'b1: SRP detected	
16	SRPDetectMsk	R/W	Mode: Host only Mask For SRPDetect Interrupt (SRPDetectMsk)	0x0
17	StsChngInt	RWC	Write Behavior: One to clear Status Change Interrupt (StsChngInt) This field indicates a status change in either the IDDIG or BSessVld signal. ■ 1'b0: No Status change ■ 1'b1: status change detected After receiving this interrupt the application must read the GPWRDN register and interpret the change in IDDIG or BSesVld with respect to the previous value stored by the application. Note: When the Battery Charger is Enabled and the ULPI interface is used, if StsChngInt is received and the application reads GPWRDN register and determines that it is due to a change in the value of IDDIG, then StsChngInt may be generated once again within the next	

Bits	Name	Access	Description	Reset
			few clock cycles. This occurs because of an ambiguity in the implementation of Battery Charger Support over the ULPI interface. After receiving the StsChngInt for the second time the application can once again read The application then processes the GPWRDN register. However, this time the value IDDIG (or BSesVld) will not have changed. second interrupt but no further action will be required as a result.	
18	StsChngIntMsk	R/W	Mode: Host and Device Mask For StsChng Interrupt (StsChngIntMsk)	0x0
20:19	LineState	RO	Mode: Host and Device LineState This field indicates the current linestate on USB as seen by the PMU module. ■ 2'b00: DM = 0, DP = 0 ■ 2'b01: DM = 0, DP = 1 ■ 2'b10: DM = 1, DP = 0 ■ 2'b11: Not-defined This bit is valid only when GPWRDN.PMUActv is 1.	
21	IDDIG	RO	Mode: Host and Device IDDIG This bit indicates the status of the IDDIG signal. the application must read this bit after receiving GPWRDN.StsChngInt and decode based on the previous value stored by the application. Indicates the current mode. ■ 1'b1: Device mode ■ 1'b0: Host mode This bit is valid only when GPWRDN.PMUActv is 1.	
22	BSessVld	RO	Mode: Device only B Session Valid (BSessVld) This field reflects the B session valid status signal from the PHY. ■ 1'b0: B-Valid is 0 ■ 1'b1: B-Valid is 1 This bit is valid only when GPWRDN.PMUActv is 1.	
23	ADPInt	RWC	Write Behavior: One to clear Mode: Host and Device ADP Interrupt (ADPInt) This bit is set whenever there is a ADP event	
28:24	MultValIdBC	RO	Mode: Host and Device MultValIdBC (MultValIdBC) Battery Charger ACA inputs in the following order. ■ Bit 28 - rid_float ■ Bit 27 - rid_gnd ■ Bit 26 - rid_a ■ Bit 25 - rid_b ■ Bit 24 - rid_c These bits are present only if BC_SUPPORT = 1. Otherwise, these bits are present only if BC_SUPPORT = 1. bits are reserved and will read 5'h0.	

Bits	Name	Access	Description	Reset
			Reset: As per ACA input	
31:29	Reserved_58_31_29	RO	Reserved for future use.	

18.6.4 Host Initialization Procedure Description

After completing the [Frequency Startup Procedure] and [Mode Switching and Initialization Procedure], it is necessary to execute the initialization procedure of XHCI as listed below, and then you can start the four standard types of transmission according to the demand, and the details of how to start the standard transmission can be referred to the specification book of XHCI, so I won't repeat it here:

Set the GINTMSK.PrtInt register to unmask state.

Set the HCFG register to FS device or HS device.

Set HPRT.PrtPwr register to 1, this setting will turn on the VBUS on the USB bus.

Wait for the HPRT0.PrtConnDet interrupt to occur, indicating that a device is connected to the USB downstream port.

Set the HPRT.PrtRst register to 1 to start USB port reset

Wait at least 10ms for USB port reset to complete handshake.

Set HPRT.PrtRst to 0, complete the USB port reset program and wait for the HPRT.PrtEnChng interrupt to occur.

Read the HPRT.PrtSpd register to get the enumeration speed value to set the HFIR register to configure the corresponding PHY Clock.

Set the RXFSIZE register to configure the RXFIFO size

Set GNPTXFSIZ register to configure the size of non-periodic transmission TXFIFO Set HPTXFSIZ register to configure the size of periodic transmission TXFIFO

18.6.5 Host Register Description

The base address of Host register is **0x0434_0000** in the whole memory space, in the text, this base address will be represented by HOST_BASE_ADDR, so the real address of each register of Host controller in the memory space will be [HOST_BASE_ADDR+Relative Address].

18.6.5.1 register summary

Name	Address Offset	Description

HCFG	0x400	Host Configuration Register
HFIR	0x404	Host Frame Interval Register
HFNUM	0x408	Host Frame Number/Frame Time Remaining Register
HPTXSTS	0x410	Host Periodic Transmit FIFO/Queue Status Register
HAINT	0x414	Host All Channels Interrupt Register
HAINTMSK	0x418	Host All Channels Interrupt Mask Register
HFLBAddr	0x41c	Host Frame List Base Address Register
HCCHARn	0x500	Host Channel-n Characteristics Register

Name	Address Offset	Description
HCDMAn	0x514	Host Channel-n DMA Address Register
HCDMABn	0x51c	Host Channel-n DMA Buffer Address Register

18.6.5.2 Register Detailed List

HCFG

Host Configuration Register
Offset Address: 0x400

Bits	Name	Access	Description	Reset
1:0	FSLSPclkSel	R/W	FS/LS PHY Clock Select (FSLSPclkSel) When the core is in FS Host mode. ■ 2'b00: PHY clock is running at 30/60 MHz ■ 2'b01: PHY clock is running at 48 MHz ■ Others: Reserved When the core is in LS Host mode. ■ 2'b00: PHY clock is running at 30/60 MHz. when the UTMI+/ULPI PHY Low Power mode is not selected, use 30/60 MHz. ■ 2'b01: PHY clock is running at 48 MHz. when the UTMI+ PHY Low Power mode is selected, use 48MHz If the PHY supplies a 48MHz power supply, then the PHY will be able to use the 48MHz power supply. MHz clock during LS mode. ■ 2'b10: PHY clock is running at 6 MHz. In USB 1.1 FS mode, use 6 MHz. MHz when the UTMI+ PHY Low Power mode is selected and the The PHY supplies a 6 MHz clock during LS mode. If you select a 6 MHz clock, the PHY can be used as the clock for the LS mode. clock during LS mode, you must do a soft reset. ■ 2'b11: Reserved Notes. ■ When Core in FS mode, the internal and external clocks have the same frequency. ■ When Core in LS mode. - If FSLSPclkSel = 2'b00: Internal and external clocks have the same frequency - If FSLSPclkSel = 2'b10: Internal clock is divided by eight version of external 48 MHz clock (utmifs_clk). Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0

2	FSLSSupp	R/W	FS- and LS-Only Support (FSLSSupp) The application uses this bit to control the core's enumeration speed. Using this bit, the application can make the core enumerate as a FS host, even If the connected device supports HS traffic. do not make	0x0

Bits	Name	Access	Description	Reset
			changes to this field after initial programming. ■ 1'b0: HS/FS/LS, based on the maximum speed supported by the connected device ■ 1'b1: FS/LS-only, even if the connected device can support HS Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
6:3	Reserved_400_6_3	RO	Reserved for future use. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
7	Ena32KHzS	R/W	Enable 32 KHz Suspend mode (Ena32KHzS) This bit can be set only if FS PHY interface is selected. Otherwise, this When FS PHY interface is selected and this bit is set, the core expects that the PHY clock is switched from 48 MHz to 32 KHz during Suspend.	0x0
15:8	ResValid	R/W	Resume Validation Period (ResValid) This field is effective only when HCFG.Ena32KHzS is set. It controls the Resume period when the core resumes from Suspend. The core counts the ResValid number of clock cycles to detect a valid resume when this is set.	0x2
22:16	Reserved_400_22_16	RO	Reserved for future use.	
23	DescDMA	R/W	Enable Scatter/gather DMA in Host mode (DescDMA) When the Scatter/Gather DMA option is selected during configuration of the the RTL, the application can set this bit during initialization to enable the Scatter/Gather DMA operation. Note: This bit must be modified only once after a reset. the following combinations are available for programming. ■ GAHBCFG.DMAEn=0,HCFG.DescDMA=0 => Slave mode ■ GAHBCFG.DMAEn=0,HCFG.DescDMA=1 => Invalid ■ GAHBCFG.DMAEn=1,HCFG.DescDMA=0 => Buffered DMA mode ■ GAHBCFG.DMAEn=1,HCFG.DescDMA=1 => Scatter/Gather DMA mode In non-Scatter/Gather DMA mode, this bit is reserved.	0x0
25:24	FrListEn	R/W	Frame List Entries (FrListEn) The value in the register specifies the number of entries in the Frame list. This field is valid only in Scatter/Gather DMA mode. ■ 2'b00: Reserved ■ 2'b01: 8 Entries ■ 2'b10: 16 Entries ■ 2'b11: 32 Entries	0x0

Bits	Name	Access	Description	Reset
			In non-Scatter/Gather DMA mode, these bits are reserved.	
26	PerSchedEna	R/W	Enable Periodic Scheduling (PerSchedEna) Applicable in Host Scatter/Gather DMA mode only. Enables periodic Initially, the bit is res and the core does not As soon as this bit is set, the core gets ready to start scheduling periodic channels and sets HCFG.PerSchedStat. The setting of HCFG.PerSchedStat indicates the core has enabled periodic scheduling. Once HCFG.PerSchedEna is set, the application is not supposed to reset the bit unless HCFG.PerSchedStat is set. as soon as as this bit is reset, the core gets ready to stop scheduling periodic channels and resets HCFG.PerSchedStat. In non-Scatter/Gather DMA mode, this bit is reserved.	0x0
30:27	Reserved_400_30_27	RO	Reserved for future use.	
31	ModeChTimEn	R/W	Mode Change Ready Timer Enable (ModeChTimEn) This bit is used to enable/disable the Host core to wait 200 PHY clock cycles at the end of Resume to change the opmode signal to the PHY to 00 after Suspend or LPM. ■ 1'b0: The Host core waits for either 200 PHY clock cycles or a linestate of SE0 at the end of resume to the change the opmode from 2'b10 to 2'b00 ■ 1'b1: The Host core waits only for a linestate of SE0 at the end of resume to change the opmode from 2'b10 to 2'b00.	0x0

HFIR

Host Frame Interval
Register Offset Address:
0x404

Bits	Name	Access	Description	Reset
15:0	FrInt	R/W	Frame Interval (FrInt) The value that the application programs to this field specifies the interval between two consecutive SOFs (FS) or micro- SOFs (HS) or Keep-Alive This field contains the number of PHY clocks that constitute the PHY clocks (HS). This field contains the number of PHY clocks that constitute the required frame interval. The default value set in this field for an FS operation when the PHY clock frequency is 60 MHz. The application can write a value to This register only after the Port Enable bit of the Host Port Control and Status register (HPRT.PrtEnaPort) has been set. if no value is programmed, the core	0xEA60

			calculates the value based on the PHY clock specified in the	

Bits	Name	Access	Description	Reset
			FS/LS PHY Clock Select field of the Host Configuration register (HCFG.FSLSPclkSel). Do not change the value of this field after the initial configuration. ■ 125 µs * (PHY clock frequency for HS) ■ 1 ms * (PHY clock frequency for FS/LS) Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
16	HFIRRldCtrl	R/W	Reload Control (HFIRRldCtrl) This bit allows dynamic reloading of the HFIR register during run time. ■ 1'b0: The HFIR cannot be reloaded dynamically ■ 1'b1: The HFIR can be dynamically reloaded during runtime. This bit needs to be programmed during initial configuration. and its value must not be changed during runtime.	0x0
31:17	Reserved_404_31_17	RO	Reserved for future use. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	

HFNUM

Host Frame Number/Frame Time Remaining Register
Offset Address: 0x408

Bits	Name	Access	Description	Reset
15:0	FrNum	RO	Frame Number (FrNum) This field increments when a new SOF is transmitted on the USB, and is reset to 0 when it reaches 16'h3FFF. This field is writable only if Remove Optional Features? was not selected in coreConsultant (OTG_RM_OTG_FEATURES = 0). Otherwise, reads return the frame number value. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
31:16	FrRem	RO	Frame Time Remaining (FrRem) Indicates the amount of time remaining in the current microframe (HS) or Frame (FS/LS), in terms of PHY clocks. This field decrements on each PHY When it reaches zero, this field is reloaded with the value in the Frame Interval register and a new SOF is transmitted on the USB. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	

HPTXSTS

Host Periodic Transmit FIFO/Queue Status Register
Offset Address: 0x410

Bits	Name	Access	Description	Reset
15:0	PTxFSpcAvail	RO	Periodic Transmit Data FIFO Space Available (PTxFSpcAvail)	

Bits	Name	Access	Description	Reset
			Indicates the number of free locations available to be written to in the Periodic Values are in terms of 32-bit words. ■ 16'h0: Periodic TxFIFO is full ■ 16'h1: 1 word available ■ 16'h2: 2 words available ■ 16'hn: n words available (n: 0 ~ 32,768) ■ 16'h8000: 32,768 words available ■ Others: Reserved Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
23:16	PTxQSpcAvail	RO	Periodic Transmit Request Queue Space Available (PTxQSpcAvail) Indicates the number of free locations available to be written in the Periodic Transmit Request Queue. This queue holds both IN and OUT requests. ■ 8'h0: Periodic Transmit Request Queue is full ■ 8'h1: 1 location available ■ 8'h2: 2 locations available ■ n: n locations available (n: 0~16) ■ Others: Reserved Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
31:24	PTxQTop	RO	Top of the Periodic Transmit Request Queue (PTxQTop) This indicates the entry in the Periodic Tx Request Queue that is currently This register is used for debugging. ■ Bit [31]: Odd/Even (micro)Frame - 1'b0: send in even (micro)Frame - 1'b1: send in odd (micro)Frame ■ Bits [30:27]: Channel/endpoint number ■ Bits [26:25]: Type - 2'b00: IN/OUT - 2'b01: Zero-length packet - 2'b10: CSPLIT - 2'b11: Disable channel command ■ Bit [24]: Terminate (last entry for the selected channel or endpoint)	

HAINT

Host All Channels Interrupt Register
Offset Address: 0x414

Bits	Name	Access	Description	Reset
15:0	HAINT	RO	Channel Interrupts (HAINT) One bit per channel: Bit 0 for Channel 0, bit 15 for Channel 15 Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	

31:16	Reserved_414_31	RO	Reserved for future use

Bits	Name	Access	Description	Reset
	_16			

HAINTMSK

Host All Channels Interrupt Mask Register
Offset Address: 0x418

Bits	Name	Access	Description	Reset
15:0	HAINTMsk	R/W	Channel Interrupt Mask (HAINTMsk) One bit per channel: Bit 0 for channel 0, bit 15 for channel 15 Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
31:16	Reserved_418_31_16	RO	Reserved for future use	

HFLBAddr

Host Frame List Base Address Register
Offset Address: 0x41c

Bits	Name	Access	Description	Reset
31:0	HFLBAddr	R/W	The starting address of the Frame list. This register is used only for Isochronous and Interrupt Channels. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0

HCCHARn

Host Channel-n Characteristics
Register Offset Address: 0x500

Bits	Name	Access	Description	Reset
10:0	MPS	R/W	Maximum Packet Size (MPS) Indicates the maximum packet size of the associated endpoint. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
14:11	EPNum	R/W	Endpoint Number (EPNum) Indicates the endpoint number on the device serving as the data source or sink. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	0x0
15	EPDir	R/W	Endpoint Direction (EPDir) Indicates whether the transaction is IN or OUT. ■ 1'b0: OUT ■ 1'b1: IN Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0

16	Reserved_500_16	RO	Reserved for future use.	
17	LSpdDev	R/W	Low-Speed Device (LSpdDev) This field is set by the application to indicate that this	0x0

Bits	Name	Access	Description	Reset
			channel is communicating to a low-speed device. The application must program this bit when a low speed device is The DWC_otg Host core is connected to the host through an FS HUB. uses this field to drive the XCVR_SELECT signal to 2'b11 while communicating to the LS Device through the FS hub. Note: In a peer to peer setup, the DWC_otg Host core ignores this bit even if it is set by the application software.	
19:18	EPType	R/W	Endpoint Type (EPType) Indicates the transfer type selected. ■ 2'b00: Control ■ 2'b01: Isochronous ■ 2'b10: Bulk ■ 2'b11: Interrupt	0x0
21:20	EC	R/W	Multi Count (MC) / Error Count (EC) When the Split Enable bit of the Host Channel-n Split Control register is set to "0", it will be set to "0". (HCSPLTn.SpltEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe for this periodic endpoint. For non-periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched for this channel before the internal DMA engine changes arbitration. ■ 2'b00: Reserved. This field yields undefined results. ■ 2'b01: 1 transaction ■ 2'b10: 2 transactions to be issued for this endpoint per microframe ■ 2'b11: 3 transactions to be issued for this endpoint per microframe When HCSPLTn.SpltEna is set (1'b1), this field indicates the number of immediate retries to be performed for a periodic split transaction on This field must be set to at least 2'b01.	0x0
28:22	DevAddr	R/W	Device Address (DevAddr) This field selects the specific device serving as the data source or sink.	0x0
29	OddFrm	R/W	Odd Frame (OddFrm) This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions. ■ 1'b0: Even (micro)frame ■ 1'b1: Odd (micro)frame This field is not applicable for Scatter/Gather DMA mode and need not be programmed by the application and is ignored by the core.	0x0
30	ChDis	RWS	Write Behavior: One to set Channel Disable (ChDis)	

Bits	Name	Access	Description	Reset
			The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer for that channel is complete. The application must wait for the Channel Disabled interrupt before treating the channel as disabled.	
31	ChEna	RWS	Write Behavior: One to set Channel Enable (ChEna) When Scatter/Gather mode is enabled. ■ 1'b0: Indicates that the descriptor structure is not yet ready. ■ 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled. This field is set by the application and cleared by the OTG host. ■ 1'b0: Channel disabled ■ 1'b1: Channel enabled	

HCDMAn

Host Channel-n DMA Address Register
Offset Address: 0x514

Bits	Name	Access	Description	Reset
31:0	DMAAddr	R/W	DMA Address (DMAAddr) This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. this register is incremented on every AHB transaction. shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0

HCDMABn

Host Channel-n DMA Buffer Address Register
Offset Address: 0x51c

Bits	Name	Access	Description	Reset
31:0	DMABufAddr	R/W	DMA Address (DMAAddr) Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only In Scatter/Gather DMA mode. Otherwise this field is reserved. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0

18.6.6 Device Initialization Procedure

Be sure to start and switch to the Device function after completing the [Frequency Startup Program] in 13.8.6.1, and follow the steps below:
Sets the following bits in the DCFG register.
DescDMA is 1 to enable descriptor DMA
mode Device Speed is HS or FS.
Status Bit for Non-Zero Transmission
Interval value for periodic transmissions
Sets the FIFO Threshold size for DMA transfers.
Clear the DCTL.SftDiscon bit to allow the device to start the Connection action on the Host Clear the following GINTMSK bits
USB Port Reset mask
Enumeration done mask
Early suspend mask
USB suspend mask
SOF mask
Wait for the GINTSTS.USBReset interrupt to occur and start the USB Reset initialization process.
Wait for the GINTSTS.EnumerationDone interrupt to occur, indicating that the USB Reset program has completed and is going to read the DSTS register to get the enumeration speed and perform the enumeration initialization procedure

18.6.7 Device Register Description

The base address of the Device register is **0x0434_0000** in the whole memory space. In the text, this base address will be represented by DEV_BASE_ADDR, so the real address of each register in the memory space will be [DEV_BASE_ADDR+Relative Address].

18.6.7.1 register summary

Name	Address Offset	Description
DCFG	0x800	Device Configuration Register
DCTL	0x804	Device Control Register
DSTS	0x808	Device Status Register
DIEPMSK	0x810	Device IN Endpoint Common Interrupt Mask Register
DOEPMSK	0x814	Device OUT Endpoint Common Interrupt Mask Register
DAINT	0x818	Device All Endpoints Interrupt Register
DAINTMSK	0x81c	Device Endpoints Interrupt Mask Register

DIEPEMPMSK	0x834	Device IN Endpoint FIFO Empty Interrupt Mask Register
DEACHINT	0x838	Device Each Endpoint Interrupt Register
DEACHINTMSK	0x83c	Device Each Endpoint Interrupt Register Mask

18.6.7.2 Register Detailed List

DCFG

Device Configuration Register
Offset Address: 0x800

Bits	Name	Access	Description	Reset
1:0	DevSpd	R/W	Device Speed (DevSpd) Indicates the speed at which the application requires the core to enumerate. or the maximum speed the application can support. However, the actual bus speed is determined only after the chirp sequence is completed, and is based on the speed of the USB host to which the core is connected. ■ 2'b00: High speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) ■ 2'b01: Full speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) ■ 2'b10: Low speed (USB 1.1 transceiver clock is 6 MHz). If you select 6MHz LS mode, you must do a soft reset. ■ 2'b11: Full speed (USB 1.1 transceiver clock is 48 MHz) Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
2	NZStsOUTHShk	R/W	Non-Zero-Length Status OUT Handshake (NZStsOUTHShk) The application can use this field to select the handshake the core sends on receiving a non zero-length data packet during the OUT transaction of a Control transfer's Status stage. ■ 1'b1: Send a STALL handshake on a non zero-length status OUT transaction and do not send the received OUT packet to the application. ■ 1'b0: Send the received OUT packet to the application (zero-length or non zero-length) and send a handshake based on the NAK and STALL bits for the endpoint in the Device Endpoint Control register. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	0x0
3	Ena32KHzSusp	R/W	Enable 32 KHz Suspend mode (Ena32KHzSusp) This bit can be set only if FS PHY interface is selected. Otherwise, this bit If the FS PHY interface is chosen and this bit is set, the PHY clock during Suspend must be switched from 48 MHz to 32 KHz. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
10:4	DevAddr	R/W	Device Address (DevAddr) The application must program this field after every SetAddress control	0x0

Bits	Name	Access	Description	Reset
			command.	
12:11	PerFrInt	R/W	Periodic Frame Interval (PerFrInt) Indicates the time within a (micro)frame at which the application must be notified using the End Of Periodic Frame Interrupt. This can be used to determine if all the isochronous traffic for that (micro)frame is complete. ■ 2'b00: 80% of the (micro)frame interval ■ 2'b01: 85% ■ 2'b10: 90% ■ 2'b11: 95%	0x0
13	EnDevOutNak	R/W	Enable Device OUT NAK (EnDevOutNak) This bit enables setting NAK for Bulk OUT endpoints after the transfer is completed for Device mode Descriptor DMA mode. ■ 1'b0: The core does not set NAK after Bulk OUT transfer complete ■ 1'b1: The core sets NAK after Bulk OUT transfer complete This is a one time programmable bit after reset like any other DCFG register bits. This bit is valid only when OTG_EN_DESC_DMA == 1'b1.	0x0
14	XCVRDLY	R/W	Enables or disables delay between xcvr_sel and txvalid during device chirp ■ 1'b1: Enable delay between xcvr_sel and txvalid during Device chirp ■ 1'b0: No delay between xcvr_sel and txvalid during Device chirp	0x0
15	ErraticIntMsk	R/W	Mode: Device Erratic Error Interrupt Mask ■ 1'b1: Mask early suspend interrupt on erratic error ■ 1'b0: Early suspend interrupt is generated on erratic error	0x0
17:16	Reserved_800_17_16	RO	Reserved for future use.	
22:18	EPMisCnt	R/W	IN Endpoint Mismatch Count (EPMisCnt) This field is valid only in shared FIFO operation. The application programs this field with a count that determines when the The core generates an Endpoint Mismatch interrupt (GINTSTS.EPMis). The core loads this value into an internal counter and decrements it. The counter is reloaded whenever there is a match or when the counter expires. the width of this counter depends on the depth of the Token Queue.	0x8
23	DescDMA	R/W	Enable Scatter/Gather DMA in Device mode (DescDMA). When the Scatter/Gather DMA option is selected during configuration of the RTL, the application can set this bit during initialization to enable the Scatter/Gather DMA operation. Note: This bit must be modified only once after a reset. The following combinations are available for programming. ■ GAHBCFG.DMAEn=0,DCFG.DescDMA=0 => Slave mode ■ GAHBCFG.DMAEn=0,DCFG.DescDMA=1 => Invalid	0x0

Bits	Name	Access	Description	Reset
			■ GAHBCFG.DMAEn=1,DCFG.DescDMA=0 => Buffer DMA mode ■ GAHBCFG.DMAEn=1,DCFG.DescDMA=1 => Scatter/Gather DMA mode	
25:24	PerSchIntvl	R/W	Periodic Scheduling Interval (PerSchIntvl) PerSchIntvl must be programmed only for Scatter/Gather DMA mode. This field specifies the amount of time the Internal DMA engine must allocate For fetching periodic IN endpoint data. based on the number of periodic endpoints, this value must be specified as 25,50 or 75% of (micro)frame. When any periodic endpoints are active, the internal DMA engine allocates the specified amount of time in fetching periodic IN endpoint data. When no periodic endpoints are active, the internal DMA engine services non-periodic endpoints, ignoring this field. After the specified time within a (micro)frame, the DMA switches to fetching for non-periodic endpoints. ■ 2'b00: 25% of (micro)frame. ■ 2'b01: 50% of (micro)frame. ■ 2'b10: 75% of (micro)frame. ■ 2'b11: Reserved.	0x0
31:26	ResValid	R/W	Resume Validation Period (ResValid) This field is effective only when DCFG.Ena32KHzSusp is set. it controls the resume period when the core resumes from suspend. The core counts for "ResValid" number of clock cycles to detect a valid resume when this bit is set.	0x2

DCTL

Device Control
Register Offset
Address: 0x804

Bits	Name	Access	Description	Reset

0	RmtWkUpSig	R/W	Remote Wakeup Signaling (RmtWkUpSig) When the application sets this bit, the core initiates remote signaling to wake the USB host. The application must set this bit to instruct the core to exit the Suspend state. The application must set this bit to instruct the core to exit the Suspend state. If LPM is enabled and the core is in the L1 (Sleep) state, when the application sets this bit, the core must clear this bit 1- 15 ms after setting it. If LPM is enabled and the core is in the L1 (Sleep) state, when the application sets this bit, the core initiates L1 remote signaling to wake up the USB host. the application must set this bit to instruct the core to exit the Sleep As specified in the LPM specification, the hardware automatically clears this bit 50 µs (TL1DevDrvResume) after being The application must not set this bit when GLPMCFG bRemoteWake from the previous LPM	0x0

Bits	Name	Access	Description	Reset
			transaction is zero. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
1	SftDiscon	R/W	Soft Disconnect (SftDiscon) The application uses this bit to signal the DWC_otg core to do a soft disconnect. As long as this bit is set, the host does not see that the device is connected, and the device does not receive signals on the USB. As long as this bit is set, the host does not see that the device is connected, and the device does not receive signals on the USB. The core stays in the disconnected state until the application clears this bit. ■ 1'b0: Normal operation. When this bit is cleared after a soft disconnect, the core drives the phy_opmode_o signal on the UTMI+ to 2'b00, which generates a device connect event to the USB host. When the device is reconnected, the USB host restarts device enumeration. ■ 1'b1: The core drives the phy_opmode_o signal on the UTMI+ to 2'b01, which generates a device disconnect event to the USB host. Note: This bit can be also used for ULPI/FS Serial interfaces. Note: This bit is not impacted by a soft reset. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	0x1
2	GNPINNakSts	RO	Global Non-periodic IN NAK Status (GNPINNakSts) ■ 1'b0: A handshake is sent out based on the data availability in the transmit FIFO. ■ 1'b1: A NAK handshake is sent out on all non-periodic IN endpoints, irrespective of the data availability in the transmit FIFO. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
3	GOUTNakSts	RO	Global OUT NAK Status (GOUTNakSts) ■ 1'b0: A handshake is sent based on the FIFO Status and the NAK and STALL bit settings. ■ 1'b1: No data is written to the RxFIFO, irrespective of space Sends a NAK handshake on all packets, except on SETUP transactions. all isochronous OUT packets are dropped.	
6:4	TstCtl	R/W	Test Control (TstCtl) ■ 3'b000: Test mode disabled ■ 3'b001: Test_J mode ■ 3'b010: Test_K mode ■ 3'b011: Test_SE0_NAK mode ■ 3'b100: Test_Packet mode ■ 3'b101: Test_Force_Enable ■ Others: Reserved	0x0

7	SGNPInNak	RWC	Set Global Non-periodic IN NAK (SGNPInNak) A write to this field sets the Global Non-periodic IN NAK.The application uses this bit to send a NAK handshake on all nonperiodic The core can also set this bit when a timeout condition is detected on a non-periodic endpoint in shared FIFO operation. The application must set this bit only after making sure that	

Bits	Name	Access	Description	Reset
			the Global IN NAK Effective bit in the Core Interrupt Register (GINTSTS.GINNakEff) is cleared.	
8	CGNPInNak	RWC	Clear Global Non-periodic IN NAK (CGNPInNak) A write to this field clears the Global Non-periodic IN NAK.	
9	SGOUTNak	RWC	Set Global OUT NAK (SGOUTNak) A write to this field sets the Global OUT NAK. The application uses this bit to send a NAK handshake on all OUT endpoints. The application must set the this bit only after making sure that the Global OUT NAK Effective bit in the Core Interrupt Register (GINTSTS.GOUTNakEff) is cleared.	
10	CGOUTNak	RWC	Clear Global OUT NAK (CGOUTNak) A write to this field clears the Global OUT NAK.	
11	PWROnPrgDone	R/W	Power-On Programming Done (PWROnPrgDone) The application uses this bit to indicate that register programming is complete after a wake-up from Power Down. mode.	0x0
12	Reserved_804_12	RO	Reserved for future use.	
14:13	GMC	R/W	Global Multi Count (GMC) GMC must be programmed only once after initialization. Applicable only for Scatter/Gather DMA mode. This indicates the number of packets to be serviced for that end point before moving to the next end point. This indicates the number of packets to be serviced for that end point before moving to the next end point. ■ 2'b00: Invalid. ■ 2'b01: 1 packet. ■ 2'b10: 2 packets. ■ 2'b11: 3 packets. The value of this field automatically changes to 2'h1 when DCFG.DescDMA is set to 1. When Scatter/Gather DMA mode is set to 2'h1, the value of this field automatically changes to 2'h1 when DCFG. is disabled, this field is reserved and reads 2'b00.	0x0

15	IgnrFrmNum	R/W	Ignore frame number for isochronous endpoints (IgnrFrmNum) in Slave Mode (GAHBCFG.DMAEn=0). This bit is not valid in Slave mode and should not be programmed to 1.Non-Scatter/Gather DMA mode (GAHBCFG.DMAEn=1,DCFG.DescDMA=0). This bit is not used when Threshold mode is enabled and should not be programmed to 1. In non-Scatter/Gather DMA mode, the application receives transfer complete interrupt after transfers for multiple (micro)frames are completed. ■ When Scatter/Gather DMA mode is disabled, this field is used by the application to enable periodic transfer interrupt. The application can program The application can program periodic endpoint transfers for multiple (micro)frames. - 0: Periodic transfer interrupt feature is disabled; the application must program transfers for periodic endpoints every (micro)frame - 1: Packets are not flushed when an ISOC IN token is received for an elapsed frame. the core ignores the frame number, sending packets as soon as the packets are ready, and the corresponding token is received. This field is also used by the application to enable periodic transfer interrupts. Scatter/Gather DMA Mode (GAHBCFG.DMAEn=1,DCFG.DescDMA=1).	0x0

Bits	Name	Access	Description	Reset
			This bit is not applicable to high-speed, high-bandwidth transfers and should not be programmed to 1. In addition, this bit is not used when Threshold mode is enabled and should not be programmed to 1. ■ 0: The core transmits the packets only in the frame number in which they are intended to be transmitted. ■ 1: Packets are not flushed when an ISOC IN token is received for an elapsed frame. The core ignores the frame number, sending packets as soon as the packets are ready, and the corresponding token is received The core ignores the frame number, sending packets as soon as the packets are ready, and the corresponding token is received. set, there must be only one packet per descriptor.	
16	NakOnBble	R/W	NAK on Babble Error (NakOnBble) Set NAK automatically on babble (NakOnBble). The core sets NAK automatically for the endpoint on which babble is Received.	0x0
17	EnContOnBNA	R/W	Enable Continue on BNA (EnContOnBNA) This bit enables the DWC_otg core to continue on BNA for Bulk OUT and INTR OUT endpoints. With this feature enabled, when a Bulk OUT or INTR OUT endpoint receives a BNA interrupt the core starts processing the descriptor that caused the BNA interrupt after the endpoint re-enables the endpoint. ■ 1'b0: After receiving BNA interrupt, the core disables the endpoint. when the endpoint is re-enabled by the application, the core starts processing from the DOEPDMA descriptor. ■ 1'b1: After receiving BNA interrupt, the core disables the endpoint. when the endpoint is re-enabled by the application, the core starts processing from the descriptor that received the BNA interrupt. This bit is valid only when OTG_EN_DESC_DMA == 1'b1. It is a one-time programmable after reset bit like any other DCTL register bits.	0x0
18	DeepSleepBESL Reject	R/W	Deep Sleep BESL Reject Core rejects LPM requests with HIRD value greater than HIRD threshold programmed. NYET response is sent for LPM tokens with HIRD value greater than HIRD threshold. By default, the Deep Sleep BESL Reject feature is disabled.	0x0
31:19	Reserved_804_31_19	RO	Reserved for future use.	

DSTS

Device Status
Register Offset
Address: 0x808

Bits	Name	Access	Description	Reset

0	SuspSts	RO	Suspend Status (SuspSts) In Device mode, this bit is set as long as a Suspend condition is detected on The core enters the Suspend state when there is no activity on the USB. phy_line_state_i signal for an extended period of time. The core comes out of the suspend under the following conditions. ■ If there is any activity on the phy_line_state_i signal ■ If the application writes to the Remote Wakeup Signaling bit in the Device Control register (DCTL.RmtWkUpSig). Shadow: Yes	

Bits	Name	Access	Description	Reset
			Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	
2:1	EnumSpd	RO	Enumerated Speed (EnumSpd) Indicates the speed at which the DWC_otg core has come up after speed detection through a chirp sequence. ■ 2'b00: High speed (PHY clock is running at 30 or 60 MHz) ■ 2'b01: Full speed (PHY clock is running at 30 or 60 MHz) ■ 2'b10: Low speed (PHY clock is running at 6 MHz) ■ 2'b11: Full speed (PHY clock is running at 48 MHz) Low speed is not supported for devices using a UTMI+ PHY.	
3	ErrticErr	RO	Erratic Error (ErrticErr) The core sets this bit to report any erratic errors (phy_rxvalid_i/phy_rxvldh_i). or phy_rxactive_i is asserted For at least 2 ms, due to PHY error) seen on The UTMI+. Due to erratic errors, the DWC_otg core goes into Suspend state and an interrupt is generated to the application with Early Suspend bit of the Core Interrupt register (GINTSTS.ErlySusp). If the early suspend is asserted because of an erratic error, the application can only perform a soft disconnect recover.	
7:4	Reserved_808_7_4	RO	Reserved for future use.	
21:8	SOFFN	RO	Frame or Microframe Number of the Received SOF (SOFFN) When the core is operating at high speed, this field contains a microframe number. When the core is operating at full or low speed, this field contains a Frame number. Note: This register may return a non zero value if read immediately after In case the register bit reads non zero immediately after power on reset it does not indicate that SOF has been received from the host. The read value of this interrupt is valid only after a valid connection between host and device is established.	
23:22	DevLnSts	RO	Device Line Status (DevLnSts) Indicates the current logic level USB data lines ■ Bit [23]: Logic level of D+ ■ Bit [22]: Logic level of D-	
31:24	Reserved_808_31_24	RO	Reserved for future use. Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	

DIEPMSK

Device IN Endpoint Common Interrupt Mask Register
Offset Address: 0x810

Bits	Name	Access	Description	Reset
0	DiXferComplMsk	R/W	Transfer Completed Interrupt Mask (XferComplMsk) Shadow: Yes	0x0

Bits	Name	Access	Description	Reset
			Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
1	DiEPDisbldMsk	R/W	Endpoint Disabled Interrupt Mask (EPDisbldMsk)	0x0
2	DiAHBErrMsk	R/W	AHB Error Mask (AHBErrMsk)	0x0
3	TimeOUTMsk	R/W	Timeout Condition Mask (TimeOUTMsk) (Non-isochronous endpoints)	0x0
4	INTknTXFEmpMsk	R/W	IN Token Received When TxFIFO Empty Mask (INTknTXFEmpMsk)	0x0
5	INTknEPMisMsk	R/W	IN Token received with EP Mismatch Mask (INTknEPMisMsk)	0x0
6	INEPNakEffMsk	R/W	IN Endpoint NAK Effective Mask (INEPNakEffMsk)	0x0
7	Reserved_810_7	RO	Reserved for future use.	
8	TxfifoUndrnMsk	R/W	Fifo Underrun Mask (TxfifoUndrnMsk)	0x0
9	BNAInIntrMsk	R/W	BNA Interrupt Mask (BNAInIntrMsk) This bit is valid only when Device Descriptor DMA is enabled.	0x0
12:10	Reserved_810_12_10	RO	Reserved for future use.	
13	DiNAKMsk	R/W	NAK interrupt Mask (NAKMsk) Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	0x0
31:14	Reserved_810_31_14	RO	Reserved for future use.	

DOEPMSK

Device OUT Endpoint Common Interrupt Mask Register
Offset Address: 0x814

Bits	Name	Access	Description	Reset
0	XferComplMsk	R/W	Transfer Completed Interrupt Mask (XferComplMsk) Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
1	EPDisbldMsk	R/W	Endpoint Disabled Interrupt Mask (EPDisbldMsk)	0x0
2	AHBErrMsk	R/W	AHB Error (AHBErrMsk)	0x0
3	SetUPMsk	R/W	SETUP Phase Done Mask (SetUPMsk) Applies to control endpoints only.	0x0
4	OUTTknEPdisMsk	R/W	OUT Token Received when Endpoint Disabled Mask (OUTTknEPdisMsk) Applies to control OUT endpoints only.	0x0
5	StsPhseRcvdMsk	R/W	Status Phase Received Mask (StsPhseRcvdMsk)	0x0
6	Back2BackSETup Msk	R/W	Back-to-Back SETUP Packets Received Mask (Back2BackSETupMsk) Applies to control OUT endpoints only.	0x0
7	Reserved_814_7	RO	Reserved for future use.	
8	OutPktErrMsk	R/W	OUT Packet Error Mask (OutPktErrMsk)	0x0
9	BnaOutIntrMsk	R/W	BNA interrupt Mask (BnaOutIntrMsk)	0x0
11:10	Reserved_814_11 _10	RO	Reserved for future use.	
12	BbleErrMsk	R/W	Babble Error interrupt Mask (BbleErrMsk)	0x0
13	NAKMsk	R/W	NAK interrupt Mask (NAKMsk)	0x0
14	NYETMsk	R/W	NYET interrupt Mask (NYETMsk) Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel	0x0

Bits	Name	Access	Description	Reset
31:15	Reserved_814_31_15	RO	Reserved for future use.	

DAINT

Device All Endpoints Interrupt Register
Offset Address: 0x818

Bits	Name	Access	Description	Reset
15:0	InEpInt	RO	OUT Endpoint Interrupt Bits (OutEPInt) One bit per OUT endpoint. Bit 16 for OUT endpoint 0, bit 31 for OUT endpoint 15 Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
31:16	OutEPInt	RO	OUT Endpoint Interrupt Bits (OutEPInt) One bit per OUT endpoint. Bit 16 for OUT endpoint 0, bit 31 for OUT endpoint 15	

DAINTMSK

Device Endpoints Interrupt Mask Register
Offset Address: 0x81c

Bits	Name	Access	Description	Reset
15:0	InEpMsk	R/W	IN EP Interrupt Mask Bits (InEpMsk) One bit per IN Endpoint. Bit 0 for IN EP 0, bit 15 for IN EP 15 The value of this field depends on the number of IN endpoints that are configured. Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
31:16	OutEpMsk	R/W	OUT EP Interrupt Mask Bits (OutEpMsk) One per OUT endpoint. Bit 16 for OUT EP 0, bit 31 for OUT EP 15 The value of this field depends on the number of OUT endpoints that are configured.	0x0

DIEPEMPMSK

Device IN Endpoint FIFO Empty Interrupt Mask Register
Offset Address: 0x834

Bits	Name	Access	Description	Reset
15:0	InEpTxfEmpMsk	R/W	IN EP Tx FIFO Empty Interrupt Mask Bits (InEpTxfEmpMsk) These bits acts as mask bits for DIEPINTn. TxFEmp interrupt One bit per IN Endpoint. ■ Bit 0 for IN endpoint 0 ... ■ Bit 15 for endpoint 15 Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
31:16	Reserved_834_31_16	RO	Reserved for future use.	

DEACHINT

Device Each Endpoint Interrupt Register

Offset Address: 0x838

Bits	Name	Access	Description	Reset
15:0	EchInEpInt	RO	IN Endpoint Interrupt Bits (EchInEpInt) One bit per IN Endpoint. ■ Bit 0 for IN endpoint 0 ... ■ Bit 15 for endpoint 15 Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	
31:16	EchOutEPInt	RO	OUT Endpoint Interrupt Bits (EchOutEPInt) One bit per OUT endpoint. ■ Bit 16 for OUT endpoint 0 ... ■ Bit 31 for OUT endpoint 15	

DEACHINTMSK

Device Each Endpoint Interrupt Register Mask
Offset Address: 0x83c

Bits	Name	Access	Description	Reset
15:0	EchInEpMsk	R/W	IN EP Interrupt Mask Bits (EchInEpMsk) One bit per IN Endpoint. ■ Bit 0 for IN endpoint 0 ... ■ Bit 15 for endpoint 15 Shadow: Yes Shadow Ctrl: vs_1t Shadow Read Select: shrd_sel One-Way: Enabled	0x0
31:16	EchOutEpMsk	R/W	OUT EP Interrupt Mask Bits (EchOutEpMsk) One per OUT Endpoint. ■ Bit 16 for IN endpoint 0 ... ■ Bit 31 for endpoint 15	0x0

18.7 SARADC

18.7.1 summarize

The SARADC is an analog to digital converter controller. Up to 3 independent channels are available on this chip

18.7.2 specificities

- Controller operating frequency 12.5MHz.
- The scanning frequency cannot be higher than 320K/s;

- 12bit sampling accuracy, 3 independent channels;

- Three channels can be triggered at once for sequential scanning;
- Automatically report interruptions when scanning is complete.

18.7.3 working method

The CPU configures the scanning channels, three channels can be configured at the same time, and starts the SARADC for channel scanning. After channel scanning is completed for all enabled channels, the system is notified of the completion of scanning through interrupts, and the CPU can obtain the conversion results.

18.7.4 SARADC Register Overview

SARADC Base address 0x030F0000
RTCSYS_SARADC Base address 0x0502C000

Name	Address Offset	Description
saradc_ctrl	0x004	control register
saradc_status	0x008	staus register
saradc_cyc_set	0x00c	saradc waveform setting register
saradc_ch1_result	0x014	channel 1 result register
saradc_ch2_result	0x018	channel 2 result register
saradc_ch3_result	0x01c	channel 3 result register
saradc_intr_en	0x020	interrupt enable register
saradc_intr_clr	0x024	interrupt clear register
saradc_intr_sta	0x028	interrupt status register
saradc_intr_raw	0x02c	interrupt raw status register

18.7.5 SARADC Register Description

saradc_ctrl

Offset Address: 0x004

Bits	Name	Access	Description	Reset
0	reg_saradc_en	RWS	when re_saradc_en is set , saradc start to measure the channels enabled in reg_saradc_sel	
3:1	Reserved			
7:4	reg_saradc_sel	R/W	select channel (1~3)	0x0
31:8	Reserved			

saradc_status

Offset Address: 0x008

Bits	Name	Access	Description	Reset
0	sta_saradc_busy	RO	busy rise when re_saradc_en is set	
3:1	Reserved			
7:4	sta_saradc_ch_busy	RO	per channel busy status	
15:8	Reserved			

19:16	sta_saradc_st	RO	fsm status for debug	
24:20	sta_saradc_cycle	RO	sample cycle for debug	

Bits	Name	Access	Description	Reset
31:25	Reserved			

saradc_cyc_set

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
4:0	reg_saradc_cyc_settling	R/W	saradc startup cycle = 1 + reg_saradc_cyc_settling , default is 16 cycle	0xF
7:5	Reserved			
11:8	reg_saradc_cyc_samp	R/W	saradc sample window = 1 + reg_saradc_cyc_samp , default is 4 cycle	0x3
15:12	reg_saradc_cyc_clkdiv	R/W	saradc clock divider , freq = ip_clk/(1+clk_div) , default is 25M/2 = 12.5M = 80ns	0x1
19:16	reg_saradc_cyc_comp	R/W	saradc compare cycle = 1+ reg_saradc_cyc_comp , default is 12 cycles	0xB
31:20	Reserved			

saradc_ch1_result

Offset Address: 0x014

Bits	Name	Access	Description	Reset
11:0	sta_saradc_ch1_result	RO	ch1 measure result	
14:12	Reserved			
15	sta_saradc_ch1_valid	RO	ch1 measure result is valid. The valid status will be cleared when this channel is re-triggered to remeasure.	
31:16	Reserved			

saradc_ch2_result

Offset Address: 0x018

Bits	Name	Access	Description	Reset
11:0	sta_saradc_ch2_result	RO	ch2 measure result	
14:12	Reserved			
15	sta_saradc_ch2_valid	RO	ch2 measure result is valid. The valid status will be cleared when this channel is re-triggered to remeasure.	
31:16	Reserved			

saradc_ch3_result

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
11:0	sta_saradc_ch3_result	RO	ch3 measure result	
14:12	Reserved			
15	sta_saradc_ch3_valid	RO	ch3 measure result is valid. The valid status will be cleared when this channel is re-triggered to remeasure.	
31:16	Reserved			

saradc_intr_en

Offset Address: 0x020

Bits	Name	Access	Description	Reset
0	sta_saradc_intr_en	R/W	interrupt enable (mask)	0x0
31:1	Reserved			

saradc_intr_clr

Offset Address: 0x024

Bits	Name	Access	Description	Reset
0	sta_saradc_intr_clr	RWC	interrupt clear	
31:1	Reserved			

saradc_intr_sta

Offset Address: 0x028

Bits	Name	Access	Description	Reset
0	sta_saradc_intr_sta	RO	interrup masked status [0]: all channels measurenment in this time is finished	
31:1	Reserved			

saradc_intr_raw

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
0	sta_saradc_intr_raw	RO	interrupt raw status [0]: all channels measurenment in this time is finished	
31:1	Reserved			

18.8 Temperature sensor

18.8.1 summarize

The chip has two built-in temperature sensors, which can be used by the CPU to monitor the chip temperature periodically. The power management module can be triggered to reset the system shutdown when the chip is overheated and unresponsive to avoid the danger of overheating.

18.8.2 working method

- Single measurement time configuration: reg_tempsen_accsel = 1 (1024T) Configuration condition, the single measurement time required is (1/(25M/12)*(1024+2+64) ~ 523.2us).
- Cycle time configuration: reg_tempsen_auto_prediv is preset to 24 so that reg_tempsen_auto_cycle is 1us per unit. The period between two measurements should be configured so that reg_tempsen_auto_cycle is greater than 524. For example, if the

measurement is done once per second, reg_tempsen_auto_cycle = 1000000.

- Measurement Channel Configuration: If reg_tempsen_sel = 3, two temperature sensors will be triggered at a time.

Measurement.

- Configure high and low temperature monitoring: Configure the temperature for triggering high temperature alarm and low temperature response tempsen_chx_temp_th
- Configuration Interrupt
- Enable temperature sensor for measurement: Configure reg_tempsen_en = 1 for measurement and wait for interrupt.

Table *18-14 Tempsensor* Interrupt Signal Description Table

interrupt bit	code		descriptive
[0]	Irq_Temp0_measure		Tempsens0. Measurement complete.
[1]	Irq_Temp1_measure		Tempsens1 measurements are complete.
[2]	reserved		reservations
[3]	reserved		reservations
[4]	Irq_Temp0_over_high_level		Tempsens0 Temperature greater than, equal to high temperature critical bounds
[5]	Irq_Temp1_over_high_level		Tempsens1 Temperature greater than or equal to the high temperature threshold. bounds
[6]	reserved		reservations
[7]	reserved		reservations
[8]	Irq_Temp0_under_low_level		Tempsens0 Temperature less than or equal to the low-temperature threshold. bounds
[9]	Irq_Temp1_under_low_level		Tempsens1 Temperature less than or equal to the low temperature threshold. bounds
[10]	reserved		reservations
[11]	reserved		reservations
[12]	Irq_Temp0_over_high_cont		Tempsens0 Temperature greater than, equal to high temperature threshold has reached reg_tempsen_ovhl_cnt_to_irq times

[13]	Irq_Temp1_over_high_cont		Tempsens1 Temperature greater than, equal to high temperature threshold has reached reg_tempsen_ovhl_cnt_to_irq times
[14]	reserved		reservations
[15]	reserved		reservations
[16]	Irq_Temp0_under_low_cont		Tempsens0 Temperature less than, equal to the low temperature threshold has reached reg_tempsen_udll_cnt_to_irq times
[17]	Irq_Temp1_under_low_cont		Tempsens1 Temperature less than, equal to low temperature threshold has reached reg_tempsen_udll_cnt_to_irq times
[18]	reserved		reservations
[19]	reserved		reservations

interrupt bit	code		descriptive
[20]	Irq_Temp0_overheat		Tempsens0 Temperature greater than, equal to overheat threshold bounds
[21]	Irq_Temp1_overheat		Tempsens1 Temperature greater than, equal to overheating threshold bounds
[22]	reserved		reservations
[23]	reserved		reservations

Figure *18-43* Temperature Measurement Time, Count and Interrupt Generation Relationship

- Check the temperature measurement result: sta_tempsen_chX_result Record the previous temperature measurement result.
  sta_tempsen_chX_max_result T h e maximum value of the measured temperature.
  tempsen_chX_temp_th_cnt Registers the number of consecutive high and low temperatures.
- Configure the overheat protection temperature reg_tempsen_overheat_th, overheat reset request countdown time reg_tempsen_overheat_cycle and enable reg_overheat_reset_en.
  Refer to the RTC Registers section again to configure the RTC register hw_thm_shdn_en,
  RTC_EN_THM_SHDN , RTC_THM_SHDN_AUTO_REBOOT enable overheat trigger power down or reboot. When overheat occurs, the temp sensor controller will send out an interrupt and start counting down, and when sta_tempsen_overheat_countdown equals to 1, then it will trigger the RTC's

request for power down protection. If the software intervenes and performs temperature control before that, reg_overheat_reset_clr can be configured to clear the countdown, but in the next measurement when the result is still overheat, the interrupt of overheat protection will be triggered again and the countdown will be triggered.

18.8.3 Temperature sensor register overview

BassAddress: 0x030A0000

Name	Address Offset	Description
tempsen_version	0x000	ip version number

Name	Address Offset	Description
tempsen_ctrl	0x004	control register
tempsen_status	0x008	staus register
tempsen_set	0x00c	Temperature sensor macro setting
tempsen_intr_en	0x010	interrupt enable
tempsen_intr_clr	0x014	interrupt clear
tempsen_intr_sta	0x018	interupt status
tempsen_intr_raw	0x01c	interrupt raw status
tempsen_ch0_result	0x020	Temperature sensor channel 0 result
tempsen_ch1_result	0x024	temperature sensor channel 1 result
tempsen_ch0_temp_th	0x040	Temperature sensor channel 0 threshold
tempsen_ch1_temp_th	0x044	Temperature sensor channel 1 threshold
Overheat_th	0x060	overheat threshold register
tempsen_auto_period	0x064	auto sample setting register
tempsen_overheat_ctrl	0x068	overheat control register
tempsen_overheat_countdown	0x06c	overheat status register
tempsen_ch0_temp_th_cnt	0x070	counter of channel 0 over/under threshold event
tempsen_ch1_temp_th_cnt	0x074	counter of channel 1 over/under threshold event

18.8.4 Temperature sensor register description

tempsen_version

Offset Address: 0x000

Bits	Name	Access	Description	Reset
31:0	reg_ip_version	RO	verision 1.0	

tempsen_ctrl

Offset Address: 0x004

Bits	Name	Access	Description	Reset
0	reg_tempsen_en	R/W	when re_tempsen_en is set , tempsen start to measure the channel set in reg_tempsen_sel	0x0
3:1	Reserved			
7:4	reg_tempsen_sel	R/W	Temperature sense channel selection	0x0
15:8	Reserved			
23:16	reg_tempsen_ovhl_cnt_to_irq	R/W	counting threshold of high temperature	0x8
31:24	reg_tempsen_udll_cnt_to_irq	R/W	counting threshold of low temperature	0x8

tempsen_status

Offset Address: 0x008

Bits	Name	Access	Description	Reset
0	sta_tempsen_busy	RO	busy status rise when re_tempsen_en is set	
31:1	Reserved			

tempsen_set

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
0	reg_tempsen_bgen	R/W	sensor macro bandgap enable	0x0
1	reg_tempsen_chopen	R/W	sensor macro chopper function enable	0x1

Bits	Name	Access	Description	Reset
2	reg_tempsen_choppol	R/W	sensor macro chopper polarity when CHOPEN=0	0x1
3	reg_tempsen_clkpol	R/W	sensor macro clock polarity when DA_TEMPSEN_EN=0	0x1
5:4	reg_tempsen_chopsel	R/W	sensor macro chop period, 0:128T, 1:256T, 2:512T, 3:1024T	0x2
7:6	reg_tempsen_accsel	R/W	sensor macro accumulate period. 0:512t, 1:1024t, 2:2048t, 3:4096t	0x1
15:8	reg_tempsen_cyc_clkdiv	R/W	clock divider for sensor macro, freq = ip_clk/(1+clk_div) , default is 25M/12 = 2.083M , T = 0.48us	0xB
17:16	reg_tempsen_tsel	R/W	sensor macro test selection, please keep 0	0x0
18	reg_tempsen_en_bjt_test	R/W	sensor macro test selection, please keep 0	0x0
31:19	Reserved			

tempsen_intr_en

Offset Address: 0x010

Bits	Name	Access	Description	Reset
31:0	sta_tempsen_intr_en	R/W	interrupt enable	0x0

tempsen_intr_clr

Offset Address: 0x014

Bits	Name	Access	Description	Reset
31:0	sta_tempsen_intr_clr	RWC	interrupt clear	

tempsen_intr_sta

Offset Address: 0x018

Bits	Name	Access	Description	Reset
31:0	sta_tempsen_intr_sta	RO	interrupt masked status	

tempsen_intr_raw

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
31:0	sta_tempsen_intr_raw	RO	interrupt raw status. [3:0] ch3~ch0 measurement finish [7:4] ch3~ch0 measurement result is higher than high temperature threshold [11:8] ch3~ch0 measurement result is lower than low temperature threshold [15:12] ch3~ch0's high temperature event count is more than threshold [19:16] ch3~ch0's low temperature event count is more than threshold [23:20] ch3~ch0 measurement result is higher than overheat temperature	

tempsen_ch0_result

Offset Address: 0x020

Bits	Name	Access	Description	Reset
12:0	sta_tempsen_ch0_result	RO	channel 0 current temperature measurement result	

15:13	Reserved		

Bits	Name	Access	Description	Reset
28:16	sta_tempsen_ch0_max_result	RO	channel 0 max temperature measurement result	
30:29	Reserved			
31	clr_tempsen_ch0_max_result	RWC	write 1 to clear channel 0 max temperature measurement result	

tempsen_ch1_result

Offset Address: 0x024

Bits	Name	Access	Description	Reset
12:0	sta_tempsen_ch1_result	RO	channel 1 current temperature measurement result	
15:13	Reserved			
28:16	sta_tempsen_ch1_max_result	RO	channel 1 max temperature measurement result	
30:29	Reserved			
31	clr_tempsen_ch1_max_result	RWC	write 1 to clear channel 1 max temperature measurement result	

tempsen_ch0_temp_th

Offset Address: 0x040

Bits	Name	Access	Description	Reset
12:0	reg_tempsen_ch0_hi_th	R/W	Channel 0 high temperature threshold to trigger interrupt	0x0
15:13	Reserved			
28:16	reg_tempsen_ch0_lo_th	R/W	channel 0 low temperature threshold to trigger interrupt	0x0
31:29	Reserved			

tempsen_ch1_temp_th

Offset Address: 0x044

Bits	Name	Access	Description	Reset
12:0	reg_tempsen_ch1_hi_th	R/W	channel 1 high temperature threshold to trigger interrupt	0x0
15:13	Reserved			
28:16	reg_tempsen_ch1_lo_th	R/W	channel 1 low temperature threshold to trigger interrupt	0x0
31:29	Reserved			

Overheat_th

Offset Address: 0x060

Bits	Name	Access	Description	Reset
12:0	reg_tempsen_overheat_th	R/W	overheat temperature threshold	0x0
31:13	Reserved			

tempsen_auto_period

Offset Address: 0x064

Bits	Name	Access	Description	Reset
23:0	reg_tempsen_auto_cycle	R/W	auto measure period. t_measure = reg_tempsen_auto_cycle*T_prediv	0x0
31:24	reg_tempsen_auto_prediv	R/W	a predivider setting for auto measure period. t_prediv = (25M/(reg_tempsen_auto_prediv+1))	0x18

tempsen_overheat_ctrl

Offset Address: 0x068

Bits	Name	Access	Description	Reset
29:0	reg_tempsen_overheat_cycle	R/W	After overheat event happens, the cycle count will be loaded to a counter and trigger counting down. when counting down to 1, a reset signal will be issue to power control unit.	0x100000
30	reg_overheat_reset_clr	RWC	write 1 to stop overheat reset counting.	
31	reg_overheat_reset_en	R/W	enable overheat reset counting down.	0x0

tempsen_overheat_countdown

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
29:0	sta_tempsen_overheat_countdown	RO	overheat reset counter	
30	Reserved			
31	sta_overheat_reset	RO	overheat reset signal status	

tempsen_ch0_temp_th_cnt

Offset Address: 0x070

Bits	Name	Access	Description	Reset
7:0	sta_ch0_over_hi_temp_th_cnt	RO	channel 0 high temperature event count status	
15:8	sta_ch0_under_lo_temp_th_cnt	RO	channel 0 low temperature event count status	
16	reg_ch0_temp_th_cnt_clr	RWC	write 1 to clear channel 0 temperature event count	
31:17	Reserved			

tempsen_ch1_temp_th_cnt

Offset Address: 0x074

Bits	Name	Access	Description	Reset
7:0	sta_ch1_over_hi_temp_th_cnt	RO	channel 1 high temperature event count status	
15:8	sta_ch1_under_lo_temp_th_cnt	RO	channel 1 low temperature event count status	
16	reg_ch1_temp_th_cnt_clr	RWC	write 1 to clear channel 1 temperature event count	
31:17	Reserved			

18.9 PWM

18.9.1 summarize

The chip provides a set of 4 independent PWM signal outputs.

18.9.2 specificities

The clock source of PWM is either 100MHz or 148.5MHz, and the default is 100MHz. 4 PWMs can be operated independently:

- Internal 30-bit counter with configurable output period and high/low level taps
- Supports up to 50MHz (100MHz/2) or 74.25MHz (148.5MHz/2) outputs down to approximately 0.093Hz (100MHz/(2^30-1)) or 0.138Hz (148.5MHz/(2^32-1)).
- Supports both continuous output (PWMMODE = 0) and fixed number of pulses output (PWMMODE = 1).
- Supports 4 PWM synchronous output modes (SHIFTMODE = 1), 4 PWM outputs can be adjusted via configuration registers.

18.9.3 working method

The basic PWM configuration flow is as follows (using PWM0 as an example):

1. Based on the selected clock source, the square wave period and the number of low level beats to be output are calculated.
2. Write the corresponding values to registers HLPERIOD0, PERIOD0
3. To operate in continuous output mode, configure PWMMODE to 0, and after setting PWMSTART[0] to 1, set PWMSTART[0] to 1.
   PWM0 starts outputting until PWMSTART[0] is set to 0 and then ends outputting.
4. To operate in fixed pulse count output mode, configure PWMMODE to 1, and write the number of square waves to be output into register PCOUNT0. Set PWMSTART[0] to 1, PWM0 will start outputting, and will end automatically when the set number of square waves is reached, and the status register PWMDONE will turn from 0 to 1.

Figure *18-44 PWM Continuous Mode*

Figure *18-45 PWM Pulse count mode*

For example, to output a waveform with a frequency of 1MHz, 75% low level and 16 pulses.

1. Using the default 100MHz clock source, the number of cycles (PERIOD0) is configured as 100MHz / 1MHz = 100, and the number of low levels (HLPERIOD0) is configured as 100 x 75% = 75. The number of pulses (PCOUNT0) is configured as 16.
2. PWMSTART[0] starts to output waveform after writing 1.
3. Read register PWMDONE[0] until the value is 1, indicating that the output is complete
4. Register PULSECOUNT0 can be read to confirm the output pulse count status value.

To enable PWM again, you need to write 0 and then 1 to PWMSTART[0] to reset the counter and status register.

When the 4 PWMs are to be operated in synchronous output mode, first configure SHIFTMODE to 1, and the process is as follows:

1. Configure HLPERIOD0/PERIOD0, HLPERIOD1/PERIOD1, HLPERIOD2/PERIOD2, HLPERIOD3/PERIOD3 to be the same value.
2. Depending on the phase difference of the 4 square waveforms that need to be staggered, configure the appropriate value to register SHIFTCOUNT0, and then set the value to SHIFTCOUNT0.
   shiftcount1, shiftcount2, shiftcount3
3. Configure PWMSTART[3:0] to be 4'hF and set SHIFTSTART to 1. The counters of all 4 channels will start counting at the same time and output the PWM waveform of the nth channel when the counter value is equal to SHIFTCOUNTn.

Figure *18-46 PWM Continuous Shift Mode*

For example, to simultaneously output 4 square waveforms with a frequency of 1KHz and a 75% low level, each waveform is staggered by 1/4 cycle.

1. Using the default 100MHz clock source, the number of cycles is configured as 100MHz / 1KHz = 100,000, and the number of low levels is configured as 100,000 x 75% = 75,000.
2. Configuration SHIFTCOUNT0 = 0, SHIFTCOUNT1 = 100,000 x 1/4 = 25,000, SHIFTCOUNT2 =
   100,000 x 2/4 = 50,000, SHIFTCOUNT3 = 100,000 x 3/4 = 75,000
3. PWMSTART[3:0] is set to 4'hF, and SHIFTSTART is set to 1. 4 PWMs output the first pulse in sequence.

Set SHIFTSTART to 0 to end the output, and read PWMDONE[3:0] until the value is 4'hF, indicating that all 4 outputs are completed.

18.9.4 PWM Register Overview

An overview of the PWM registers is shown in Table 18-15.

Table *18-15 PWM* Register Overview

Name	Address Offset	Description
HLPERIOD0	0x000	PWM0 Low level beat count
PERIOD0	0x004	PWM0 Square wave period beat count
HLPERIOD1	0x008	PWM1 Low level beat count
PERIOD1	0x00c	PWM1 Square wave period beat count

Name	Address Offset	Description
HLPERIOD2	0x010	PWM2 Low Level Taps
PERIOD2	0x014	PWM2 Square Wave Period Beats
HLPERIOD3	0x018	PWM3 Low Level Taps
PERIOD3	0x01c	PWM3 Square Wave Period Beats
POLARITY	0x040	PWM mode setting
PWMSTART	0x044	Enable PWM output
PWMDONE	0x048	PWM end state
PWMUPDATE	0x04c	Dynamically load PWM cycle parameters
PCOUNT0	0x050	Setting the number of PWM0 pulses
PCOUNT1	0x054	Setting the number of PWM1 pulses
PCOUNT2	0x058	Setting the number of PWM2 pulses
PCOUNT3	0x05c	Setting the number of PWM3 pulses
PULSECOUNT0	0x060	PWM0 Output pulse counter status
PULSECOUNT1	0x064	PWM1 Pulse counter status has been output
PULSECOUNT2	0x068	PWM2 Output Pulse Counter Status
PULSECOUNT3	0x06c	PWM3 Output Pulse Counter Status
SHIFTCOUNT0	0x080	Synchronization mode PWM0 Initial phase difference
SHIFTCOUNT1	0x084	Synchronization mode PWM1 Initial phase difference
SHIFTCOUNT2	0x088	Synchronization mode PWM2 Initial phase difference
SHIFTCOUNT3	0x08c	Synchronization mode PWM3 Initial phase difference
SHIFTSTART	0x090	PWM Synchronization Mode Enable
PWM_OE	0x0d0	PWM IO output enable

18.9.5 PWM Register Description

HLPERIOD0

Offset Address: 0x000

Bits	Name	Access	Description	Reset
29:0	HLPERIOD0	R/W	PWM0 Low Level Taps (in clk_pwm) When POLARITY[0] is 0, this value is a low level tap, and when POLARITY[0] is 1, this value is a high level tap. Flat beat count.	0x1

PERIOD0

Offset Address: 0x004

Bits	Name	Access	Description	Reset
29:0	PERIOD0	R/W	PWM0 Square wave period beats (unit is clk_pwm) Note that the PERIOD value must be greater than the HLPERIOD Value.	0x2

HLPERIOD1

Offset Address: 0x008

Bits	Name	Access	Description	Reset
29:0	HLPERIOD1	R/W	PWM1 Low Level Taps (in clk_pwm) When POLARITY[1] is 0, this value is tapped low.	0x1

Bits	Name	Access	Description	Reset
			number, when POLARITY[1] is 1, this value is the high power Flat beat count.	

PERIOD1

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
29:0	PERIOD1	R/W	PWM1 Number of square wave period beats (unit is clk_pwm) Note that the PERIOD value must be greater than the HLPERIOD Value.	0x2

HLPERIOD2

Offset Address: 0x010

Bits	Name	Access	Description	Reset
29:0	HLPERIOD2	R/W	PWM2 Low Level Taps (in clk_pwm) When POLARITY[2] is 0, this value is a low level tap, and when POLARITY[2] is 1, this value is a high level tap. Flat beat count.	0x1

PERIOD2

Offset Address: 0x014

Bits	Name	Access	Description	Reset
29:0	PERIOD2	R/W	PWM2 Square wave period beats (unit is clk_pwm) Note that the PERIOD value must be greater than the HLPERIOD Value.	0x2

HLPERIOD3

Offset Address: 0x018

Bits	Name	Access	Description	Reset
29:0	HLPERIOD3	R/W	PWM3 Low Level Taps (in clk_pwm) When POLARITY[3] is 0, this value is a low level tap, and when POLARITY[3] is 1, this value is a high level tap. Flat beat count.	0x1

PERIOD3

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
29:0	PERIOD3	R/W	PWM3 Square Wave Cycle Beats (Unit is clk_pwm) Note that the PERIOD value must be greater than the HLPERIOD Value.	0x2

POLARITY

Offset Address: 0x040

Bits	Name	Access	Description	Reset
3:0	POLARITY	R/W	PWM0~3 Signal Polarity [n] = 0: PWMn defaults to a low output level [n] = 1: PWMn outputs high by default	0x0
7:4	Reserved			
11:8	PWMMODE	R/W	PWM0~3 Operation Mode [n+8] = 0: PWMn operation in continuous out mode [n+8] = 1: PWMn operation on fixed pulse outputs	0x0

Bits	Name	Access	Description	Reset
			paradigm	
15:12	Reserved			
16	SHIFTMODE	R/W	Enable PWM synchronized phase output mode 0 = PWM0~3 all operate in general mode. 1 = PWM0~3 operation on 4 synchronized output modules style	0x0
19:17	Reserved			
20	pclk_force_en	R/W	APB Clock Gating Control 0 = enable APB clock gating to automatically turn off the clock when idle 1 = APB Clock remains constantly on	0x0
31:21	Reserved			

PWMSTART

Offset Address: 0x044

Bits	Name	Access	Description	Reset
3:0	PWMSTART	R/W	Enable PWM0~3 [n] = 0: stop PWMn [n] = 1: output PWMn When PWMMODE is set to 0, write 0 to bit n and then write 1 to start PWMn output until writing 0 to bit n stops output. When PWMMODE is set to 1, write bit n to 1 to start PWMn output, and when the number of pulses output is equal to 1, the output will stop. Output is automatically stopped at the PCOUNTn value. The output is automatically stopped when the SHIFTMODE set 1, PWMSTART[3:0] do Enables the signal output of PWM0~3, PWM Startup is now controlled by SHIFTSTART.	0x0
31:4	Reserved			

PWMDONE

Offset Address: 0x048

Bits	Name	Access	Description	Reset
3:0	PWMDONE	RO	PWM0~3 End output state [n] = 1: PWMn ended output When PWMSTART[n] is set from 0 to 1, it will send a signal to The memory value is cleared to 0.	
31:4	Reserved			

PWMUPDATE

Offset Address: 0x04c

Bits	Name	Access	Description	Reset

3:0	PWMUPDATE	R/W	Dynamic loading of PWM parameters When PWMSTART is written from 0 to 1, register value (HLPERIODn, PERIODn) will be used in the PWM internal The new value will be stored temporarily. If you want to change the waveform dynamically in the PWM output, write the new values to HLPERIODn and PERIODn, and then write 1 to PWMUPDATE[n]. Write 0 again for the new value to take effect.	0x0
31:4	Reserved			

PCOUNT0

Offset Address: 0x050

Bits	Name	Access	Description	Reset
23:0	PCOUNT0	R/W	PWM0 Number of pulses (Set value must be greater than 0) Valid only when PWMMODE[0] is set to 1.	0x1
31:24	Reserved			

PCOUNT1

Offset Address: 0x054

Bits	Name	Access	Description	Reset
23:0	PCOUNT1	R/W	PWM1 Number of pulses (Set value must be greater than 0) Valid only when PWMMODE[1] is set to 1.	0x1
31:24	Reserved			

PCOUNT2

Offset Address: 0x058

Bits	Name	Access	Description	Reset
23:0	PCOUNT2	R/W	PWM2 Number of pulses (Set value must be greater than 0) Valid only when PWMMODE[2] is set to 1.	0x1
31:24	Reserved			

PCOUNT3

Offset Address: 0x05c

Bits	Name	Access	Description	Reset
23:0	PCOUNT3	R/W	PWM3 Number of pulses (Set value must be greater than 0) Valid only when PWMMODE[3] is set to 1.	0x1
31:24	Reserved			

PULSECOUNT0

Offset Address: 0x060

Bits	Name	Access	Description	Reset
23:0	PULSECOUNT0	RO	PWM0 Number of pulses already output Status	
31:24	Reserved			

PULSECOUNT1

Offset Address: 0x064

Bits	Name	Access	Description	Reset
23:0	PULSECOUNT1	RO	PWM1 Number of pulses already output Status	
31:24	Reserved			

PULSECOUNT2

Offset Address: 0x068

Bits	Name	Access	Description	Reset
23:0	PULSECOUNT2	RO	PWM2 Number of pulses output status	

31:24	Reserved			

PULSECOUNT3

Offset Address: 0x06c

Bits	Name	Access	Description	Reset
23:0	PULSECOUNT3	RO	PWM3 Number of output pulses status	
31:24	Reserved			

SHIFTCOUNT0

Offset Address: 0x080

Bits	Name	Access	Description	Reset
23:0	SHIFTCOUNT0	R/W	PWM0 Phase difference of the first pulse output (in clk_pwm) Valid only when SHIFTMODE is set to 1.	0x0
31:24	Reserved			

SHIFTCOUNT1

Offset Address: 0x084

Bits	Name	Access	Description	Reset
23:0	SHIFTCOUNT1	R/W	PWM1 Phase difference of the first pulse output (in clk_pwm) Valid only when SHIFTMODE is set to 1.	0x0
31:24	Reserved			

SHIFTCOUNT2

Offset Address: 0x088

Bits	Name	Access	Description	Reset
23:0	SHIFTCOUNT2	R/W	PWM2 Phase difference of the first pulse output (in clk_pwm) Valid only when SHIFTMODE is set to 1.	0x0
31:24	Reserved			

SHIFTCOUNT3

Offset Address: 0x08c

Bits	Name	Access	Description	Reset
23:0	SHIFTCOUNT3	R/W	PWM3 Phase difference of the first pulse output (in clk_pwm) Valid only when SHIFTMODE is set to 1.	0x0
31:24	Reserved			

SHIFTSTART

Offset Address: 0x090

Bits	Name	Access	Description	Reset
0	SHIFTSTART	R/W	Enable PWM output in synchronized mode When SHIFTMODE is set to 1, this register is written to 1. After that, it starts to output PWM0~3.	0x0
31:1	Reserved			

PWM_OE

Offset Address: 0x0d0

Bits	Name	Access	Description	Reset
3:0	PWM_OE	R/W	PWM0~3 IO output enable 1 = IO is output, 0 = IO is input	0x0
31:4	Reserved			

18.10 Key scan

18.10.1 summarize

Keyscan supports a matrix of up to 8x8 = 64 keys. If you don't need that many keys, you can decide which rows or columns to mask out or keep. You can decide which rows or columns you want to mask out or keep. Depending on the software needs, you can use snapshot mode or FIFO mode to get the key information.

18.10.2 working method

Figure *18-47 Keyscan* Architecture Block Diagram

When the state machine (FSM) is in rest mode (no keys are pressed), all rows output 0, and col is in input mode with weak pullup enabled (weak pullup is set in the ioblk-associated registers, not in the keyscan module). When any of the

When a key is pressed, the col side will see a value other than all 1's after debounce. This indicates that a key was pressed. The FSM will start a scan. The sequence Row[0] -> Row[7] is such that only one bit outputs 0 at a time (the rest are in HiZ high resistance state). The results are updated into an array each time.

The FSM will keep scanning until all rows are scanned with all 1's indicating that there are no keystrokes pressed, then it will go back to rest mode (all rows output 0).

18.10.3 Basic settings

The reg_row_mask, reg_col_mask and reg_enable in KEYSCAN0 can be used to optionally stall some IOs when the 8x8 matrix is not used, without outputting or referencing their inputs. Default is all off. So you need to turn it on.
The reg_db_col in KEYSCAN_CONFIG2 determines how much debounce time the column input takes.
It can only be used.
The reg_slow_div in KEYSCAN_CONFIG1 determines how long the IP's FSM stays in each stage. Remember that this number must be larger than the debounce time. Otherwise, it will be wrong if the IO reads before debounce after transition.

The reg_wait_cntr in KEYSCAN_CONFIG3 can be used to slow down the scanning speed. This is because the keyscan module will keep scanning as long as the key is pressed. The counter can be set to wait a fixed amount of time before starting a new scan. The counter can be controlled to wait a fixed amount of time before each new scan is started, in order to reduce the scanning frequency.

18.10.4 Use FIFO mode.

When using FIFO mode, the IP will store the 64 key values in an array each time it scans in. Each time any key has a different state than the last scan, it will push the index and current value (0/1) of that key into the FIFO. So the numbers in [5:0] specify which key was pressed. The [6] indicates whether the key was pressed (0) or released (1). When the FIFO is not empty, an IRQ is issued. The advantage of this mode is that it eliminates the need for the software to do a bit-by-bit re-lock to check which bit has changed. The disadvantage of this mode is that KEY_SCAN_FIFO is a read register that automatically pops, so it should be handled with care.

Turns on reg_irq_fifo_not_empty_enable for KEYSCAN_IRQ_ENABLE.
After receiving an IRQ, read the reg_irq_fifo_not_empty in KEYSCAN_IRQ_FLAG and check the
KEYSCAN_FIFO_STATUS reg_fifo_not_empty. then read the contents of

KEYSCAN_FIFO. Until it is clear, then clear KEY_SCAN_IRQ_CLEAR. End IRQ retine.

18.10.5 Use snapshot array mode.

When a snapshot array is used, the values of the 64 keys currently scanned in are stored in an array within the IP. If this
If the contents of the array are different from the contents of KEYSCAN_SNAPSHOT_ARRAY, an IRQ will be sent. The software can trigger KEYSCAN_SNAPSHOT_TRIG to capture the current array into a snapshot array, and then slowly compare what has changed from what it knew before.

Turns on reg_irq_snapshot_change_enable for KEYSCAN_IRQ_ENABLE. Upon receipt of an IRQ, reads the trigger KEYSCAN_SNAPSHOT_TRIG, and then interprets the contents of KEYSCAN_SNAPSHOT_ARRAY.
Clear KEY_SCAN_IRQ_CLEAR again. End IRQ retine.

18.10.6 Key scan register overview

Name	Address Offset	Description
KEYSCAN_CONFIG0	0x000	
KEYSCAN_CONFIG1	0x004	
KEYSCAN_CONFIG2	0x008	
KEYSCAN_CONFIG3	0x00c	
KEYSCAN_SNAPSHOT_ARRAY	0x014	
KEYSCAN_SNAPSHOT_TRIG	0x01c	
KEYSCAN_FIFO_STATUS	0x020	
KEYSCAN_FIFO	0x024	
KEYSCAN_IRQ_ENABLE	0x028	
KEYSCAN_IRQ_FLAG	0x02c	
KEYSCAN_IRQ_CLEAR	0x030	

18.10.7 Key scan register description

KEYSCAN_CONFIG0

Offset Address: 0x000

Bits	Name	Access	Description	Reset

7:0	reg_row_mask	R/W	ROW[7:0] Mask 0 = enable 1 = disable	0xff
15:8	reg_col_mask	R/W	COL[7:0] Mask 0 = enable	0xff

Bits	Name	Access	Description	Reset
			1 = disable	
16	reg_enable	R/W	keyscan enable 0 = disable 1 = enable	0x0
31:17	Reserved			

KEYSCAN_CONFIG1

Offset Address: 0x004

Bits	Name	Access	Description	Reset
23:0	reg_slow_div	R/W	slow divider (MUST BE BIGGER THAN reg_db_col) Each step is IP clock frequency divide by reg_slow_div Scan frequency = IP clock freq / ((reg_slow_div+1) * (9+reg_wait_count+1)) IDLE -> ROW0 -> ROW1 -> ROW2->ROW3->ROW4->ROW5->ROW6 ->ROW7->UPDATE->WAIT->IDLE	0xff
31:24	Reserved			

KEYSCAN_CONFIG2

Offset Address: 0x008

Bits	Name	Access	Description	Reset
15:0	reg_db_col	R/W	column input debounce counter (IP clock cycle)	0x64
31:16	Reserved			

KEYSCAN_CONFIG3

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
7:0	reg_wait_cntr	R/W	wait interval between each scan (unit is reg_slow_div count)	0x10
31:8	Reserved			

KEYSCAN_SNAPSHOT_ARRAY

Offset Address: 0x014

Bits	Name	Access	Description	Reset
63:0	reg_cpu_snapshot_array	RO	CPU snapshot array result (0 = press, 1 = not press) [0] = Row 0 , Col 0 [1] = Row 0 , Col 1 … [7] = Row 0, Col 7 [8] = Row 1, Col 0 [63] = Row 7, Col 7 [N] = Row Y, Col X (N = Y*8 + X)	

KEYSCAN_SNAPSHOT_TRIG

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
0	reg_cpu_snapshot_toggle	W1T	Write 1 to Trigger snapshot array to	

Bits	Name	Access	Description	Reset
			update When current result is different from snapshot result, irq happen To solve the IRQ, write 1 to trigger the snapshot array to copy from current array and start checking which bit is different from previous state	
31:1	Reserved			

KEYSCAN_FIFO_STATUS

Offset Address: 0x020

Bits	Name	Access	Description	Reset
3:0	reg_fifo_count	RO	FIFO content count 0 = Empty 1 = one content in FIFO N = N content in FIFO	
4	reg_fifo_not_empty	RO	FIFO not empty flag 0 = Empty 1 = Not empty	
31:5	Reserved			

KEYSCAN_FIFO

Offset Address: 0x024

Bits	Name	Access	Description	Reset
6:0	reg_fifo_rdata	ROC	read data from FIFO (Auto POP) - check FIFO empty-ness before read [6] 0 = press, 1 = not-press [5:0] = index Row = INT(index/8) Col = mod(index,8) 63 = Row 7 , Column 7 13 = Row 1 , Clumne 5	
31:7	Reserved			

KEYSCAN_IRQ_ENABLE

Offset Address: 0x028

Bits	Name	Access	Description	Reset
0	reg_irq_fifo_not_empty_enable	R/W	FIFO mode IRQ Enable 0 = Disable 1 = Enable	0x0
3:1	Reserved			
4	reg_irq_snapshot_change_enable	R/W	Snapshot mode IRQ Enable 0 = Disable 1 = Enable	0x0
31:5	Reserved			

KEYSCAN_IRQ_FLAG

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
0	reg_irq_fifo_not_empty	RO	FIFO not empty IRQ flag 0 = Empty 1 = Not empty	
3:1	Reserved			

Bits	Name	Access	Description	Reset
4	reg_irq_snapshot_change	RO	Snapshot change IRQ flag 0 = No change 1 = Change	
31:5	Reserved			

KEYSCAN_IRQ_CLEAR

Offset Address: 0x030

Bits	Name	Access	Description	Reset
0	reg_irq_fifo_not_empty_clear_w1t	W1T	FIFO not empty IRQ Clear (Write 1 clear)	
3:1	Reserved			
4	reg_irq_snapshot_change_clear_w1t	W1T	Snapshot Change IRQ Clear (Write 1) clear)	
31:5	Reserved			

18.11 Wiegand

18.11.1 summarize

Figure *18-48 wiegand* signaling bus transfer *0/1* method

The Wiegand interface uses two single-ended signals, D0/D1. When the buses are idle both are high, and on D0 there is a

A low pulse means a "0" was transmitted, a low pulse on D1 means a "1" was transmitted.

Wiegand is commonly used in access control systems. There are two common formats, Wiegand 26/34, which represent the number of bits in a packet. These two formats are described below.

18.11.1.1 Wiegand 26

Figure *18-49 Wiegand 26* Format

18.11.1.2 Wiegand 34

Figure *18-50 Wiegand 34 format*

F = Facility Code
U = User code

Some access cards have a string of numbers on the back of the card, convert it to hexDec 0002262506 Hex : 22_85_EA
0x22 is his Facility code (Dec = 34).
0x85EA is his user code (Dec = 34282).

Figure *18-51* Numerical Significance of Common Magnetic Cards *and* Magnetic Buttons

The 34, 34282 seen in the back is the Facility code & user code in decimal after disassembly PS. This IP TX RX does not handle parity insertion or checking. It is handled by the software.

18.11.2 working method

Figure *18-52 wiegand* architecture block diagram

The Wiegand module consists of a TX and a RX, which can be used in one or both directions. When you TX out, RX will block it. So the RX will not receive the signal from the TX. TX supports push pull mode or open drain mode.

18.11.2.1 TX

Before TX, set the TX high time and low time, and the bit number of the packet, which is defined by MSB 1st or LSB 1st . Then put the
After the data is put into TX_BUFFER register, use TX_TRIG to send the data out.
After the TX transmission is complete, you can rely on the TX finish interrupt or polling TX_BUSY status to determine when the next packet is ready.

18.11.2.2 RX

Before RX, set the debounce time and the expected bit number of the received packet.

When there is a low pulse at D0 D1, RX
The data is pushed into the temp buffer. When the number of received bits reaches the expected number of bits for a packet, the

It pushes the temp buffer into RX_BUFFER and sends an interrupt to the software to handle it. The temp buffer will continue to receive the next data.

If an idle timeout occurs on D0 D1, the packet is forced to be treated as a packet even if the bit number is not reached.

The high bit of the RX BUFFER records the total number of bits received by the packet, whether it is due to a timeout, or whether there is an overflow before the software reads it.

Each received packet can be determined by rx_buffer_rececived interrupt or RX_BUFFER_VALID.
If there is valid data in RX_BUFFER, trigger RX_BUFFER_CLEAR to clear it after RX Data is taken.
RX_BUFFER to receive the next packet.

18.11.3 Wiegand Register Overview

Name	Address Offset	Description
TX_CONFIG0	0x000	
TX_CONFIG1	0x004	
TX_CONFIG2	0x008	
TX_BUFFER	0x00c	
TX_TRIG	0x014	
TX_BUSY	0x018	
TX_DEBUG	0x01c	
RX_CONFIG0	0x020	
RX_CONFIG1	0x024	
RX_CONFIG2	0x028	
RX_BUFFER	0x02c	
RX_BUFFER_VALID	0x038	
RX_BUFFER_CLEAR	0x03c	
RX_DEBUG	0x040	
IRQ_ENABLE	0x044	
IRQ_FLAG	0x048	
IRQ_CLEAR	0x04c	

18.11.4 Wiegand Register Description

TX_CONFIG0

Offset Address: 0x000

Bits	Name	Access	Description	Reset
23:0	reg_tx_lowtime	R/W	TX Low width , unit = cycle	0xff
31:24	Reserved			

TX_CONFIG1

Offset Address: 0x004

Bits	Name	Access	Description	Reset
23:0	reg_tx_hightime	R/W	TX High width , unit = cycle	0xff
31:24	Reserved			

TX_CONFIG2

Offset Address: 0x008

Bits	Name	Access	Description	Reset
6:0	reg_tx_bitcount	R/W	TX Frame bit count per transmit , unit = bit	0x18
7	Reserved			
8	reg_tx_msb1st	R/W	TX Transmit from MSB or LSB 0 : LSB 1st , from tx_buffer[0] --> tx_buffer[reg_tx_bitcount] 1 : MSB 1st , from tx_buffer[reg_tx_bitcount] –> tx_buffer[0]	0x0
15:9	Reserved			
16	reg_tx_opendrain	R/W	TX using push-pull mode or opendrain mode 0 : push-pull mode 1 : opendrain mode	0x0
31:17	Reserved			

TX_BUFFER

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
63:0	reg_tx_buffer	R/W	TX buffer content	0x00

TX_TRIG

Offset Address: 0x014

Bits	Name	Access	Description	Reset
0	reg_tx_trig_w1t	W1T	Trigger transmittion Write 1 trigger (please check reg_tx_busy before transmit)	
31:1	Reserved			

TX_BUSY

Offset Address: 0x018

Bits	Name	Access	Description	Reset
0	reg_tx_busy	RO	0 = idle, allow to trigger transmission 1 = busy, do not trigger any more tranmission or it will be ignored.	
31:1	Reserved			

TX_DEBUG

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
2:0	reg_tx_fsm	RO	TX Finite State Machine current state 0 : idle 1 :: Wait bus idle 2 : tx_start 3 : transmit low 4 : transmit high 5 : tx_stop	
7:3	Reserved			
14:8	reg_tx_pointer	RO	TX pointer current position indicate how many bit is still not yet send	
31:15	Reserved			

RX_CONFIG0

Offset Address: 0x020

Bits	Name	Access	Description	Reset
15:0	reg_rx_debounce	R/W	RX input debounce time (unit is cycle)	0xff
31:16	Reserved			

RX_CONFIG1

Offset Address: 0x024

Bits	Name	Access	Description	Reset
31:0	reg_idle_timeout	R/W	Bus timeout cycle count When bus is idle for idle_timeout cycle, bus is expected to be back to idle If some bit has received but not yet accumulate to rx_bitcount, it will also treat as a complete packet.	0xfff

RX_CONFIG2

Offset Address: 0x028

Bits	Name	Access	Description	Reset
6:0	reg_rx_bitcount	R/W	RX Expected Frame bit count , unit = bit	0x18
7	Reserved			
8	reg_rx_msb1st	R/W	RX Received sequence 0 : LSB 1st, 1st data is put in reg_rx_buffer[0]->[1]->[2].... 1 : MSB 1st, 1st data is put in reg_rx_buffer[reg_rx_bitcount]->[0]	0x0
11:9	Reserved			
12	reg_rx_enable	R/W	RX Enable 0 : disable 1 : Enable	0x0
31:13	Reserved			

RX_BUFFER

Offset Address: 0x02c

Bits	Name	Access	Description	Reset
72:0	reg_rx_buffer	RO	RX Buffer [63:0] = fifo = Indicate received content [70:64] = fifo_bit_count = How many The effective bit is in rx_buffer[63:0].	

Bits	Name	Access	Description	Reset
			[71] = idle_reach = This RX is terminated by bus idle timeout [72] = overflow = This RX just overwrite an un-read message	
95:73	Reserved			

RX_BUFFER_VALID

Offset Address: 0x038

Bits	Name	Access	Description	Reset
0	reg_rx_buffer_valid	RO	reg_rx_buffer validity 0 : not valid 1 : valid	
31:1	Reserved			

RX_BUFFER_CLEAR

Offset Address: 0x03c

Bits	Name	Access	Description	Reset
0	reg_rx_buffer_clear_w1t	W1T	reg_rx_buffer clear (write 1 clear)	
31:1	Reserved			

RX_DEBUG

Offset Address: 0x040

Bits	Name	Access	Description	Reset
0	reg_businidle	RO	bus in idle indication 0 : bus is not in idle 1 : bus is in idle more than reg_rx_idle_timeout cycle	
31:1	Reserved			

IRQ_ENABLE

Offset Address: 0x044

Bits	Name	Access	Description	Reset
0	reg_irq_tx_finish_enable	R/W	TX Finish IRQ Enable (to inform all data has been being transmitted, ready for next) 0 : Disable 1 : Enable	0x0
3:1	Reserved			
4	reg_irq_rx_overflow_enable	R/W	RX Overflow IRQ Enable 0 : Disable 1 : Enable	0x0
7:5	Reserved			
8	reg_irq_rx_received_enable	R/W	RX Received IRQ Enable 0 : Disable 1 : Enable	0x0
31:9	Reserved			

IRQ_FLAG

Offset Address: 0x048

Bits	Name	Access	Description	Reset
0	reg_irq_tx_finish	RO	TX Finish IRQ Flag 0 : no IRQ 1 : IRQ (one tranmission has been	

Bits	Name	Access	Description	Reset
			completed)	
3:1	Reserved			
4	reg_irq_rx_overflow	RO	RX overflow IRQ Flag 0 : no IRQ 1 : IRQ (rx buffer is not pop and new) data has overwritten)	
7:5	Reserved			
8	reg_irq_rx_received	RO	RX received IRQ Flag 0 : no IRQ 1 : RX buffer has new data	
31:9	Reserved			

IRQ_CLEAR

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
0	reg_irq_tx_finish_clear_w1t	W1T	TX Finish IRQ Clear , Write 1 to clear reg_irq_tx_finish flag	
3:1	Reserved			
4	reg_irq_rx_overflow_clear_w1t	W1T	RX Overflow IRQ Clear . Write 1 to clear reg_irq_rx_overflow flag	
7:5	Reserved			
8	reg_irq_rx_received_clear_w1t	W1T	RX Received IRQ Clear . Write 1 to clear reg_irq_rx_received flag	
31:9	Reserved			

18.12 IRRX Infrared Interface

18.12.1 summarize

Receives external infrared data via the IRRX unit.

18.12.2 Features

1. Supports NEC coding mode (with repeat code).
2. Supports Philips RC5/RC6 coding modes
3. Supports Sony coding modes
4. Supports Wake on Infrared function

18.12.3 working method

Figure *18-53 IRRX* Operating Modes

The software pre-sets the format of the received infrared data, when the IRRX module receives the infrared signal, it will decode it, and the code which conforms to the pre-set format will be sent to the CPU through the interrupt, and the CPU will carry out the corresponding operation according to the code.

18.12.4 IRRX Register Overview

Name	Address Offset	Description
IR_EN	0x000	
IR_MODE	0x004	
IR_CFG	0x008	
IR_FRAME	0x00c	
int_en	0x010	
int_clr	0x014	
int_msk	0x018	

Name	Address Offset	Description
int	0x01c	
int_raw	0x020	
IR_SYMBOL_CFG0	0x030	
IR_SYMBOL_CFG1	0x034	
IR_SYMBOL_CFG2	0x038	
IR_SYMBOL_CFG3	0x03c	
IR_SYMBOL_CFG4	0x040	
IR_SYMBOL_CFG5	0x044	
IR_SYMBOL_CFG6	0x048	
IR_SYMBOL_CFG7	0x04c	
IR_CLOCK_CTRL	0x050	
IR_DATA0	0x080	
IR_DATA1	0x084	
IR_DATA2	0x088	
IR_DATA3	0x08c	
IR_DATA4	0x090	
IR_NEC_DATA0	0x0a8	
IR_SONY_DATA0	0x0ac	
IR_SONY_DATA1	0x0b0	
IR_PHILIPS_DATA0	0x0b4	
IR_PHILIPS_DATA1	0x0b8	
IR_PRD_REC0	0x0e0	
IR_PRD_REC1	0x0e4	
IR_PRD_REC2	0x0e8	
IR_PRD_REC3	0x0ec	
IR_PRD_REC4	0x0f0	
IR_PRD_REC5	0x0f4	
SPARE_0	0xff0	
SPARE_1	0xff4	
SPARE_RO	0xff8	
DATA_CODE	0xffc	

18.12.5 IRRX Register Description

Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	reg_ir_rx_en	R/W	ir receiver enable	0x0
1	reg_ir_rx_rst	R/W	ir receiver reset	0x0
15:2	Reserved			
16	reg_ir_init_done	RO	ir receiver ready	
31:17	Reserved			

Offset Address: 0x004

Bits	Name	Access	Description	Reset
1:0	reg_ir_mode	R/W	ir receiver mode 0x0: pulse distance coding 0x2: bi-phase coding, RC5 0x3: bi-phase coding, RC6	0x0
7:2	Reserved			
8	reg_periodic_mode	R/W	periodic sample mode 0x1: periodic sample, reg_ic_mode is ignored	0x0
31:9	Reserved			

Offset Address: 0x008

Bits	Name	Access	Description	Reset
7:0	reg_tick_prd	R/W	tick period	0x18
11:8	reg_sample_prd	R/W	sample period	0x9
14:12	reg_debounce	R/W	Input signal debounce control	0x4
15	Reserved			
16	reg_import_inv	R/W	Input signal polarity control	0x1
17	reg_export_inv	R/W	output data polarity control	0x0
18	reg_sony_format	R/W	tx transmit stop burst at frame end 0x1: SONY formate 0x0: NEC formate	0x0

Bits	Name	Access	Description	Reset
19	reg_repeat_support	R/W	support simplfied repeat code	0x1
20	reg_bit_edge_sel	R/W	data bit selection used in pulse distance coding 0x1: SONY formate 0x0: NEC formate	0x0
31:21	Reserved			

Offset Address: 0x00c

Bits	Name	Access	Description	Reset
7:0	reg_length	R/W	ir receiver data length	0x20
8	reg_slength	R/W	ir receiver repeate code length 0x1: TC9012 0x0: others	0x0
15:9	Reserved			
18:16	reg_lead_p_ratio	R/W	ratio of leading symbol period and strobe period, used in periodic sample mode	0x4
31:19	Reserved			

Offset Address: 0x010

Bits	Name	Access	Description	Reset
0	reg_rx_done_int_en	R/W	rx_done interrupt enable	0x1
1	reg_frame_err_int_en	R/W	frame_err interrupt enable	0x1
2	reg_frame_ovf_int_en	R/W	frame_ovf interrupt enable	0x1
3	reg_release_int_en	R/W	release interrupt enable	0x1
4	reg_repeat_int_en	R/W	repeat interrupt enable	0x1
31:5	Reserved			

Offset Address: 0x014

Bits	Name	Access	Description	Reset
0	reg_rx_done_int_clr	W1T	rx_done interrupt clear	
1	reg_frame_err_int_clr	W1T	frame_err interrupt clear	
2	reg_frame_ovf_int_clr	W1T	frame_ovf interrupt clear	
3	reg_release_int_clr	W1T	release interrupt clear	

Bits	Name	Access	Description	Reset
4	reg_repeat_int_clr	W1T	repeat interrupt clear	
31:5	Reserved			

Offset Address: 0x018

Bits	Name	Access	Description	Reset
0	reg_rx_done_int_msk	R/W	rx_done interrupt mask	0x0
1	reg_frame_err_int_msk	R/W	frame_err interrupt mask	0x0
2	reg_frame_ovf_int_msk	R/W	frame_ovf interrupt mask	0x0
3	reg_release_int_msk	R/W	release interrupt mask	0x0
4	reg_repeat_int_msk	R/W	repeat interrupt mask	0x0
31:5	Reserved			

Offset Address: 0x01c

Bits	Name	Access	Description	Reset
0	reg_rx_done_int	RO	rx_done interrpt	
1	reg_frame_err_int	RO	frame_err interrpt	
2	reg_frame_ovf_int	RO	frame_ovf interrpt	
3	reg_release_int	RO	release interrpt	
4	reg_repeat_int	RO	repeat interrpt	
31:5	Reserved			

Offset Address: 0x020

Bits	Name	Access	Description	Reset
0	reg_rx_done_int_raw	RO	rx_done interrupt raw value	
1	reg_frame_err_int_raw	RO	frame_err interrupt raw value	
2	reg_frame_ovf_int_raw	RO	frame_ovf interrupt raw value	
3	reg_release_int_raw	RO	release interrupt raw value	
4	reg_repeat_int_raw	RO	repeat interrupt raw value	
31:5	Reserved			

Offset Address: 0x030

Bits	Name	Access	Description	Reset
11:0	reg_ir_rx_lead_p	R/W	lead symbol postive interval	0x383
15:12	Reserved			

Bits	Name	Access	Description	Reset
23:16	reg_ir_rx_lead_p_tol	R/W	lead symbol postive interval tolerance reg_ir_rx_lead_p + reg_ir_rx_lead_p_tol <= 12'FFF reg_ir_rx_lead_p - reg_ir_rx_lead_p_tol >= 12'000	0x48
31:24	Reserved			

Offset Address: 0x034

Bits	Name	Access	Description	Reset
11:0	reg_ir_rx_lead_n	R/W	lead symbol negtive interval	0x1c1
15:12	Reserved			
23:16	reg_ir_rx_lead_n_tol	R/W	lead symbol negtive interval tolerance reg_ir_rx_lead_n + reg_ir_rx_lead_n_tol <= 12'FFF reg_ir_rx_lead_n - reg_ir_rx_lead_n_tol >= 12'000	0x24
31:24	Reserved			

Offset Address: 0x038

Bits	Name	Access	Description	Reset
7:0	reg_ir_rx_stop	R/W	stop symbol postive interval	0x37
11:8	reg_ir_rx_stop_tol	R/W	stop symbol postive interval tolerance reg_ir_rx_stop + reg_ir_rx_stop_tol <= 8'FF reg_ir_rx_stop - reg_ir_rx_stop_tol >= 8'00	0x3
15:12	Reserved			
23:16	reg_ir_rx_bit_p	R/W	data symbol postive interval	0x37

Bits	Name	Access	Description	Reset
31:24	reg_ir_rx_bit_p_tol	R/W	data symbol postive interval tolerance reg_ir_rx_bit_p + reg_ir_rx_bit_p_tol <= 8'FF reg_ir_rx_bit_p - reg_ir_rx_bit_p_tol >= 8'00	0x3

Offset Address: 0x03c

Bits	Name	Access	Description	Reset
11:0	reg_ir_rx_bit_one	R/W	data one totol interval	0xe0
15:12	Reserved			
23:16	reg_ir_rx_bit_one_tol	R/W	data one total interval tolerance reg_ir_rx_bit_one + reg_ir_rx_bit_one_tol <= 12'FFF reg_ir_rx_bit_one - reg_ir_rx_bit_one_tol >= 12'000	0x18
31:24	Reserved			

Offset Address: 0x040

Bits	Name	Access	Description	Reset
11:0	reg_ir_rx_bit_zero	R/W	data zero totol interval	0x6f
15:12	Reserved			
23:16	reg_ir_rx_bit_zero_tol	R/W	data zero total interval tolerance reg_ir_rx_bit_zero + reg_ir_rx_bit_zero_tol <= 12'FFF reg_ir_rx_bit_zero - reg_ir_rx_bit_zero_tol >= 12 '000	0x09
31:24	Reserved			

Offset Address: 0x044

Bits	Name	Access	Description	Reset
15:0	reg_ir_rx_release_time	R/W	time for wait repeat code, >108ms.	0x27de
31:16	Reserved			

Offset Address: 0x048

Bits	Name	Access	Description	Reset
11:0	reg_ir_rx_slead_p	R/W	repeate code lead symbol postive interval	0x383
15:12	Reserved			
23:16	reg_ir_rx_slead_p_tol	R/W	repeate code lead symbol postive interval tolerance reg_ir_rx_slead_p + reg_ir_rx_slead_p_tol <= 12'FFF reg_ir_rx_slead_p - reg_ir_rx_slead_p _tol >= 12'000	0x48
31:24	Reserved			

Offset Address: 0x04c

Bits	Name	Access	Description	Reset
11:0	reg_ir_rx_slead_n	R/W	repeate code lead symbol negtive interval	0x1c1
15:12	Reserved			
23:16	reg_ir_rx_slead_n_tol	R/W	repeate code lead symbol negtive interval tolerance reg_ir_rx_slead_n + reg_ir_rx_slead_n_tol <= 12'FFF reg_ir_rx_slead_n - reg_ir_rx_slead_n_tol >= 12'000	0x24
31:24	Reserved			

Offset Address: 0x050

Bits	Name	Access	Description	Reset
0	reg_pclock_auto_ctrl	R/W	pclk auto-gating control	0x0
1	reg_lpclock_switch_en	R/W	ip clock could be switched to 32KHz	0x0
7:2	Reserved			
8	reg_skip_lead_p	R/W	skip lead pulse check	0x0
31:9	Reserved			

Offset Address: 0x080

Bits	Name	Access	Description	Reset
31:0	reg_ir_rx_data0	RO	recevier data[31:0]	

Offset Address: 0x084

Bits	Name	Access	Description	Reset
31:0	reg_ir_rx_data1	RO	recevier data[63:32]	

Offset Address: 0x088

Bits	Name	Access	Description	Reset
31:0	reg_ir_rx_data2	RO	recevier data[95:64]	

Offset Address: 0x08c

Bits	Name	Access	Description	Reset
31:0	reg_ir_rx_data3	RO	recevier data[127:96]	

Offset Address: 0x090

Bits	Name	Access	Description	Reset
31:0	reg_ir_rx_data4	RO	recevier data[159:128]	

Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
31:0	reg_ir_rx_nec_32bit	RO	receiver data, nec format	

Offset Address: 0x0ac

Bits	Name	Access	Description	Reset
11:0	reg_ir_rx_sony_12bit	RO	receiver data, sony D7C5 format	
15:12	Reserved			
30:16	reg_ir_rx_sony_15bit	RO	receiver data, sony D7C8 format	
31	Reserved			

Offset Address: 0x0b0

Bits	Name	Access	Description	Reset
19:0	reg_ir_rx_sony_20bit	RO	receiver data, sony D7C13 format	
31:20	Reserved			

Offset Address: 0x0b4

Bits	Name	Access	Description	Reset
11:0	reg_ir_rx_philips_rc5	RO	receiver data, RC5 format	
31:12	Reserved			

Offset Address: 0x0b8

Bits	Name	Access	Description	Reset
19:0	reg_ir_rx_philips_rc6	RO	receiver data, RC6 format	
31:20	Reserved			

Offset Address: 0x0e0

Bits	Name	Access	Description	Reset
11:0	reg_start_p0_min	RO	start phase0 minimul width	
15:12	Reserved			
27:16	reg_start_p0_max	RO	start phase0 maximul width	
31:28	Reserved			

Offset Address: 0x0e4

Bits	Name	Access	Description	Reset
11:0	reg_start_p1_min	RO	start phase1 minimul width	
15:12	Reserved			
27:16	reg_start_p1_max	RO	start phase1 maximul width	
31:28	Reserved			

Offset Address: 0x0e8

Bits	Name	Access	Description	Reset
11:0	reg_bit_p0_min	RO	bit phase0 minimul width	
15:12	Reserved			
27:16	reg_bit_p0_max	RO	bit phase0 maximul width	
31:28	Reserved			

Offset Address: 0x0ec

Bits	Name	Access	Description	Reset
11:0	reg_bit_p1_min	RO	bit phase1 minimul width	
15:12	Reserved			

Bits	Name	Access	Description	Reset
27:16	reg_bit_p1_max	RO	bit phase1 maximul width	
31:28	Reserved			

Offset Address: 0x0f0

Bits	Name	Access	Description	Reset
11:0	reg_end_min	RO	end phase minimuml width	
15:12	Reserved			
27:16	reg_end_max	RO	end phase maximuml width	
31:28	Reserved			

Offset Address: 0x0f4

Bits	Name	Access	Description	Reset
15:0	reg_frame_min	RO	frame minimul width	
31:16	reg_frame_max	RO	frame maximul width	

19 Security subsystem module

The chip provides a separate security subsystem module that is responsible for providing specific security functions. The security subsystem module includes the following security function modules

- Crypto DMA
- True Random Number Generator
- Secure Debug Protection (SDU)

Crypto DMA unit provides hardware acceleration for symmetric key encryption/decryption and hash, Security eFuse unit is responsible for providing system security settings and security keys for security subsystems, and True Messy Number Generation unit provides random numbers with qualified messiness for security systems.

Figure *19-1* Security Subsystem Modules

19.1 CryptoDMA

19.1.1 summarize

CryptoDMA is a hardware gas pedal that implements symmetric key algorithms, hash algorithms, and BASE64 conversions. It supports symmetric algorithms:

algorithmic technology

AES 128/192/256, DES/TDES, SM4, and hash algorithms: SHA-1/SHA256. Direct memory access for key encryption/decryption of data blocks or hash algorithms via linked-list instruction strings.

- The symmetric algorithm is suitable for accelerated hard-core encryption and decryption of data, and supports a variety of packet encryption and block concatenation processing methods, including ECB, CBC, CTR.
- The implementation of the AES (Advanced Encryption Standard) algorithm is compliant with the FIPS 197 standard. the implementation of the DES (Data Encryption Standard)/TDES algorithm is compliant with ISO/IEC 18033-3.

- Patchwork algorithms are suitable for data integrity checks and accelerated digital signature operations. sha1 and sha256, compliant with the FIPS180-2 standard.
- BASE64 operations are suitable for storing binary data such as MIME emails or URLs within a text message.

particulars

19.1.2 Functional characteristics

The CryptoDMA module has the following functional features:

- Supports symmetric encryption/decryption algorithm AES and packet encryption modes ECB/CBC/CTR. 128 key lengths are supported. bit, 256-bit, and the key can be configured by the security operating system or by a chain-list command.
- Supports symmetric encryption/decryption algorithm SM4 and packet encryption modes ECB/CBC/CTR.
- Supports symmetric encryption/decryption algorithms DES/TDES and packet encryption modes ECB/CBC/CTR.
- Support for hash algorithms SHA1, SHA256
- Supports CPU configuration for inputting PIO data and DMA reading of table command input data.
- Supports circular linked table structure, supports splicing multiple linked table data.
- Provides interrupt status query, interrupt masking and interrupt clearing functions.

19.1.3 DMA Function Description

CryptoDMA provides memory direct access DMA function, the application program only needs to provide the chain table instruction to the target data block to start the CryptoDMA DMA function, until it receives the interrupt notification of completion, the block encryption/decryption or hash operation is indicated as the end of the block, and the result will be output to the target address.

19.1.4 Symmetric Key Algorithm Group Encryption Mode Functional Description

The symmetric key algorithm AES/DES/SM4 both support ECB/CBC/CTR group encryption mode.

19.1.4.1 ECB mode

In the ECB (Electronic CodeBook) model, the encryption and decryption algorithms are directly applied to each packet of data by the operations of each packet. This feature allows the encryption of plaintext and decryption of ciphertext to be performed independently by any group of block data.

Figure *19-2 ECB* Model

19.1.4.2 CBC model

CBC (Cipher Block Chaining) the input plaintext grouping is firstly differentiated from the input vector IV (Intialization Vector) or the result of the ciphertext of the previous grouping before encrypting the operation, the encryption operation in CBC mode must start from the first block data grouping, and after that, the encryption operation requires the ciphertext obtained from the previous grouping to be encrypted. The encryption operation in CBC mode must start from the first block data grouping. When decrypting, the current ciphertext can be decrypted and the previous ciphertext can be used to obtain the plaintext.

Figure *19-3 CBC* Model

19.1.4.3 CTR Mode

CTR (Counter) is the use of encryption or decryption of a different set of series to ensure the independence of encrypted data processing and security, generally used to encrypt the cumulative series and the plaintext after the different or operation.

Figure *19-4 CTR* Model

19.1.5 CryptoDMA Register Overview

Name	Address Offset	Description
dma_ctrl	0x000	DMA controll register
int_mask	0x004	interrupt mask
des_base_0	0x008	descriptor base low address
des_base_1	0x00c	descriptor base high address
spacc_int_raw	0x010	interrupt
secure_key_valid	0x014	key valid
des_addr_0	0x018	current descirptor low address
des_addr_1	0x01c	current descirptor high address
PIO_cmd_data_0	0x080	PIO command0
PIO_cmd_data_1	0x084	PIO command1
PIO_cmd_data_2	0x088	PIO command2
PIO_cmd_data_3	0x08c	PIO command3
PIO_cmd_data_4	0x090	PIO command4
PIO_cmd_data_5	0x094	PIO command5

PIO_cmd_data_6	0x098	PIO command6
PIO_cmd_data_7	0x09c	PIO command7

Name	Address Offset	Description
PIO_cmd_data_8	0x0a0	PIO command8
PIO_cmd_data_9	0x0a4	PIO command9
PIO_cmd_data_10	0x0a8	PIO command10
PIO_cmd_data_11	0x0ac	PIO command11
PIO_cmd_data_12	0x0b0	PIO command12
PIO_cmd_data_13	0x0b4	PIO command13
PIO_cmd_data_14	0x0b8	PIO command14
PIO_cmd_data_15	0x0bc	PIO command15
PIO_cmd_data_16	0x0c0	PIO command16
PIO_cmd_data_17	0x0c4	PIO command17
PIO_cmd_data_18	0x0c8	PIO command18
PIO_cmd_data_19	0x0cc	PIO command19
PIO_cmd_data_20	0x0d0	PIO command20
PIO_cmd_data_21	0x0d4	PIO command21
key_data_0	0x100	cipher key data 0
key_data_1	0x104	cipher key data 1
key_data_2	0x108	cipher key data 2
key_data_3	0x10c	cipher key data 3
key_data_4	0x110	cipher key data 4
key_data_5	0x114	cipher key data 5
key_data_6	0x118	cipher key data 6
key_data_7	0x11c	cipher key data 7
key_data_8	0x120	cipher key data 8
key_data_9	0x124	cipher key data 9
key_data_10	0x128	cipher key data 10
key_data_11	0x12c	cipher key data 11
key_data_12	0x130	cipher key data 12
key_data_13	0x134	cipher key data 13
key_data_14	0x138	cipher key data 14
key_data_15	0x13c	cipher key data 15
key_data_16	0x140	cipher key data 16
key_data_17	0x144	cipher key data 17
key_data_18	0x148	cipher key data 18
key_data_19	0x14c	cipher key data 19
key_data_20	0x150	cipher key data 20
key_data_21	0x154	cipher key data 21
key_data_22	0x158	cipher key data 22
key_data_23	0x15c	cipher key data 23
ini_data_0	0x180	initial vector data 0
ini_data_1	0x184	initial vector data 1
ini_data_2	0x188	initial vector data 2
ini_data_3	0x18c	initial vector data 3
ini_data_4	0x190	initial vector data 4
ini_data_5	0x194	initial vector data 5
ini_data_6	0x198	initial vector data 6
ini_data_7	0x19c	initial vector data 7
ini_data_8	0x1a0	initial vector data 8
ini_data_9	0x1a4	initial vector data 9
ini_data_10	0x1a8	initial vector data 10
ini_data_11	0x1ac	initial vector data 11
sha_data_0	0x1c0	SHA paramenter0
sha_data_1	0x1c4	SHA paramenter1
sha_data_2	0x1c8	SHA paramenter2
sha_data_3	0x1cc	SHA paramenter3
sha_data_4	0x1d0	SHA paramenter4

Name	Address Offset	Description
sha_data_5	0x1d4	SHA paramenter5
sha_data_6	0x1d8	SHA paramenter6
sha_data_7	0x1dc	SHA paramenter7

19.1.6 CryptoDMA Register Description

(base address 0x02060000)

dma_ctrl

dma_ctrl
Offset Address: 0x000

Bits	Name	Access	Description	Reset
0	dma_en	R/W	DMA Channel Enable Control 0: Channeling forbidden energy; 1: Channel Enable.	0x0
1	decriptor_mode	R/W	Channel Command Mode 0: PIO mode 1:Describe the key table schema	0x0
15:2	Reserved			
23:16	max_read_burst	R/W	Maximum read burst value	0x0
31:24	max_write_burs	R/W	Maximum Write Burst Value	0x0

int_mask

int_mask
Offset Address: 0x004

Bits	Name	Access	Description	Reset
0	int_enc_mask	R/W	Encryption and Decryption Interrupt Masking	0x0
1	int_hash_mask	R/W	Hash Interrupt Masking	0x0
31:2	Reserved		reservations	

des_base_0

des_base_0
Offset Address: 0x008

Bits	Name	Access	Description	Reset
31:0	des_base_0	R/W	Description Key Table Address_Low Address	0x0

des_base_1

des_base_1
Offset Address: 0x00c

Bits	Name	Access	Description	Reset
31:0	des_base_1	R/W	Description key table address_high address	0x0

spacc_int_raw

spacc_int_raw
Offset Address: 0x010

Bits	Name	Access	Description	Reset

31:0	spacc_int_raw	R/W	DMA interrupt	0x0

secure_key_valid

secure_key_valid
Offset Address:
0x014

Bits	Name	Access	Description	Reset
31:0	secure_key_valid	R/W	key rms value **One-Way:** Enabled	0x0

des_addr_0

des_addr_0
Offset Address: 0x018

Bits	Name	Access	Description	Reset
31:0	des_addr_0	R/W	Description key table address offset_low address	0x0

des_addr_1

des_addr_1
Offset Address: 0x01c

Bits	Name	Access	Description	Reset
31:0	des_addr_1	R/W	Description key table address offset_high address	0x0

PIO_cmd_data_0

PIO mode descriptor
Offset Address: 0x080

Bits	Name	Access	Description	Reset
0	PIO_cmd_data_0	R/W	PIO Mode Description Key Table, 0: Mode Description Key Table not valid 1: Schema description keytab valid	0x0
1	PIO_cmd_data_0_1	R/W	Describes the last descriptor of the key table	0x0
2	PIO_cmd_data_0_2	R/W	PIO Mode Interrupt Enable 0: Interrupts the disablement; 1: Interrupt enable.	0x0
3	PIO_cmd_data_0_3	R/W	Succession Description Key Table Address Selection 0: Use the successor address to the current address. 1:Using the Succession Address Register Value	0x0
7:4	PIO_cmd_data_0_4	R/W	reservations	0x0
8	PIO_cmd_data_0_8	R/W	DMA Use Bypass	0x0
9	PIO_cmd_data_0_9	R/W	Using AES encryption and decryption	0x0
10	pio_cmd_data_0_10	R/W	Using DES Add/Drop	0x0
11	pio_cmd_data_0_11	R/W	Using SM4 encryption and decryption	0x0
12	pio_cmd_data_0_12	R/W	Using SHA Operations	0x0
13	pio_cmd_data_0_13	R/W	Using BASE64 encoding and decoding	0x0
15:14	Reserved			

19:16	pio_cmd_data_0_16	R/W	Operational key selection Using aes/des/sm4 operations: 1: Describe subkey 2:Key 2 4:Key 1 8:Milan 0 Using sha 1:Descriptor parameters 2: Reservations 4: Reservations 8: Using the SHA Parameter Registers Remaining reservations	0x0
23:20	pio_cmd_data_0_20	R/W	First Calculation IV Selection	0x0

Bits	Name	Access	Description	Reset
			Using aes/des/sm4 operations. 1: Descriptor IV 2:IV2 4:IV1 8:IV0	
26:24	pio_cmd_data_0_24	R/W	Sequential Algorithm IV Selection Using aes/des/sm4 operations. 1: Descriptor IV 2:IV2 4:IV1 8:IV0	0x0
27	pio_cmd_data_0_27	R/W	Key Choice	0x0
31:28	pio_cmd_data_0_28	R/W	reservations	0x0

PIO_cmd_data_1

PIO mode descriptor
Offset Address: 0x084

Bits	Name	Access	Description	Reset
0	PIO_cmd_data_1_0	R/W	Encryption/decryption selection 1-encryption/0-decription hash parameter selection 1-need parameter/0-no need	0x0
1	PIO_cmd_data_1_1	R/W	CBC mode 0-ECB/1-CBC	0x0
2	PIO_cmd_data_1_2	R/W	CTR Mode 1-CTR	0x0
5:3	PIO_cmd_data_1_3	R/W	Key Mode 100-128bit/010-192bit/001-256bit for aes 0-DES/1-TDES sha mode 0-SHA1/1-SHA256	0x0
31:6	PIO_cmd_data_1_6	R/W	reservations	0x0

PIO_cmd_data_2

PIO mode descriptor
Offset Address: 0x088

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_2	R/W	reservations	0x0

PIO_cmd_data_3

PIO mode descriptor
Offset Address: 0x08c

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_3	R/W	Succession Descriptor Address_Low Address	0x0

PIO_cmd_data_4

PIO mode descriptor
Offset Address: 0x090

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_4	R/W	Succession Descriptor Address_High Address	0x0

PIO_cmd_data_5

PIO mode descriptor
Offset Address: 0x094

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_5	R/W	DMA Source Address_Low Address	0x0

PIO_cmd_data_6
PIO mode descriptor

Offset Address: 0x098

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_6	R/W	DMA Source Address_High Address	0x0

PIO_cmd_data_7

PIO mode descriptor
Offset Address: 0x09c

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_7	R/W	DMA Destination Address_Low Address	0x0

PIO_cmd_data_8

PIO mode descriptor
Offset Address: 0x0a0

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_8	R/W	DMA Destination Address_High Address	0x0

PIO_cmd_data_9

PIO mode descriptor
Offset Address: 0x0a4

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_9	R/W	SHA message size	0x0

PIO_cmd_data_10

PIO mode descriptor
Offset Address: 0x0a8

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_10	R/W	reservations	0x0

PIO_cmd_data_11

PIO mode descriptor
Offset Address: 0x0ac

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_11	R/W	BASE64 Target Message Size	0x0

PIO_cmd_data_12

PIO mode descriptor
Offset Address: 0x0b0

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_12	R/W	reservations	0x0

PIO_cmd_data_13

PIO mode descriptor
Offset Address: 0x0b4

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_13	R/W	reservations	0x0

PIO_cmd_data_14

PIO mode descriptor
Offset Address: 0x0b8

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_14	R/W	reservations	0x0

PIO_cmd_data_15
PIO mode descriptor

Offset Address: 0x0bc

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_15	R/W	reservations	0x0

PIO_cmd_data_16

PIO mode descriptor
Offset Address: 0x0c0

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_16	R/W	reservations	0x0

PIO_cmd_data_17

PIO mode descriptor
Offset Address: 0x0c4

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_17	R/W	reservations	0x0

PIO_cmd_data_18

PIO mode descriptor
Offset Address: 0x0c8

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_18	R/W	reservations	0x0

PIO_cmd_data_19

PIO mode descriptor
Offset Address: 0x0cc

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_19	R/W	reservations	0x0

PIO_cmd_data_20

PIO mode descriptor
Offset Address: 0x0d0

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_20	R/W	reservations	0x0

PIO_cmd_data_21

PIO mode descriptor
Offset Address:
0x0d4

Bits	Name	Access	Description	Reset
31:0	PIO_cmd_data_21	R/W	reservations	0x0

key_data_0

3key
Offset Address: 0x100

Bits	Name	Access	Description	Reset
31:0	key_data_0	RO	keys	

key_data_1

key
Offset Address: 0x104

Bits	Name	Access	Description	Reset
31:0	key_data_1	RO	keys	

key_data_2

key
Offset Address: 0x108

Bits	Name	Access	Description	Reset
31:0	key_data_2	RO	keys	

key_data_3

key
Offset Address: 0x10c

Bits	Name	Access	Description	Reset
31:0	key_data_3	RO	keys	

key_data_4

key
Offset Address: 0x110

Bits	Name	Access	Description	Reset
31:0	key_data_4	RO	keys	

key_data_5

key
Offset Address: 0x114

Bits	Name	Access	Description	Reset
31:0	key_data_5	RO	keys	

key_data_6

key
Offset Address: 0x118

Bits	Name	Access	Description	Reset
31:0	key_data_6	RO	keys	

key_data_7

key
Offset Address: 0x11c

Bits	Name	Access	Description	Reset
31:0	key_data_7	RO	keys	

key_data_8

key
Offset Address: 0x120

Bits	Name	Access	Description	Reset
31:0	key_data_8	RO	keys	

key_data_9

key
Offset Address: 0x124

Bits	Name	Access	Description	Reset
31:0	key_data_9	RO	keys	

key_data_10

key
Offset Address: 0x128

Bits	Name	Access	Description	Reset
31:0	key_data_10	RO	keys	

key_data_11

key
Offset Address: 0x12c

Bits	Name	Access	Description	Reset
31:0	key_data_11	RO	keys	

key_data_12

key
Offset Address: 0x130

Bits	Name	Access	Description	Reset
31:0	key_data_12	RO	keys	

key_data_13

key
Offset Address: 0x134

Bits	Name	Access	Description	Reset
31:0	key_data_13	RO	keys	

key_data_14

key
Offset Address: 0x138

Bits	Name	Access	Description	Reset
31:0	key_data_14	RO	keys	

key_data_15

key
Offset Address: 0x13c

Bits	Name	Access	Description	Reset
31:0	key_data_15	RO	keys	

key_data_16

key
Offset Address: 0x140

Bits	Name	Access	Description	Reset
31:0	key_data_16	RO	keys	

key_data_17

key
Offset Address: 0x144

Bits	Name	Access	Description	Reset
31:0	key_data_17	RO	keys	

key_data_18

key
Offset Address: 0x148

Bits	Name	Access	Description	Reset
31:0	key_data_18	RO	keys	

key_data_19

key
Offset Address: 0x14c

Bits	Name	Access	Description	Reset
31:0	key_data_19	RO	keys	

key_data_20

key
Offset Address: 0x150

Bits	Name	Access	Description	Reset
31:0	key_data_20	RO	keys	

key_data_21

key
Offset Address: 0x154

Bits	Name	Access	Description	Reset
31:0	key_data_21	RO	keys	

key_data_22

key
Offset Address: 0x158

Bits	Name	Access	Description	Reset
31:0	key_data_22	RO	keys	

key_data_23

key
Offset Address: 0x15c

Bits	Name	Access	Description	Reset
31:0	key_data_23	RO	keys	

ini_data_0

3iv
Offset Address: 0x180

Bits	Name	Access	Description	Reset
31:0	ini_data_0	RO	starting parameter	

ini_data_1

iv
Offset Address: 0x184

Bits	Name	Access	Description	Reset
31:0	ini_data_1	RO	starting parameter	

ini_data_2

iv
Offset Address: 0x188

Bits	Name	Access	Description	Reset
31:0	ini_data_2	RO	starting parameter	

ini_data_3

iv
Offset Address: 0x18c

Bits	Name	Access	Description	Reset
31:0	ini_data_3	RO	starting parameter	

ini_data_4

iv

Offset Address: 0x190

Bits	Name	Access	Description	Reset
31:0	ini_data_4	RO	starting parameter	

ini_data_5

iv

Offset Address: 0x194

Bits	Name	Access	Description	Reset
31:0	ini_data_5	RO	starting parameter	

ini_data_6

iv

Offset Address: 0x198

Bits	Name	Access	Description	Reset
31:0	ini_data_6	RO	starting parameter	

ini_data_7

iv

Offset Address: 0x19c

Bits	Name	Access	Description	Reset
31:0	ini_data_7	RO	starting parameter	

ini_data_8

iv

Offset Address: 0x1a0

Bits	Name	Access	Description	Reset
1:0	ini_data_8	RO	starting parameter	
31:2	Reserved			

ini_data_9

iv

Offset Address: 0x1a4

Bits	Name	Access	Description	Reset
1:0	ini_data_9	RO	starting parameter	
31:2	Reserved			

ini_data_10

iv

Offset Address: 0x1a8

Bits	Name	Access	Description	Reset
31:0	ini_data_10	RO	starting parameter	

ini_data_11

iv

Offset Address: 0x1ac

Bits	Name	Access	Description	Reset
31:0	ini_data_11	RO	starting parameter	

sha_data_0

sha parameter
Offset Address: 0x1c0

Bits	Name	Access	Description	Reset
31:0	sha_data_0	RO	SHA Parameters	

sha_data_1

sha parameter
Offset Address: 0x1c4

Bits	Name	Access	Description	Reset
31:0	sha_data_1	RO	SHA Parameters	

sha_data_2

sha parameter
Offset Address: 0x1c8

Bits	Name	Access	Description	Reset
31:0	sha_data_2	RO	SHA Parameters	

sha_data_3

sha parameter
Offset Address: 0x1cc

Bits	Name	Access	Description	Reset
31:0	sha_data_3	RO	SHA Parameters	

sha_data_4

sha parameter
Offset Address: 0x1d0

Bits	Name	Access	Description	Reset
31:0	sha_data_4	RO	SHA Parameters	

sha_data_5

sha parameter
Offset Address: 0x1d4

Bits	Name	Access	Description	Reset
31:0	sha_data_5	RO	SHA Parameters	

sha_data_6

sha parameter
Offset Address: 0x1d8

Bits	Name	Access	Description	Reset
31:0	sha_data_6	RO	SHA Parameters	

sha_data_7

sha parameter
Offset Address: 0x1dc

Bits	Name	Access	Description	Reset
31:0	sha_data_7	RO	SHA Parameters	

19.2 Secure debugging firewall

In order to provide debug or test access to or control of the internal functions of the chip, the chip provides several debug interfaces, such as JTAG, I2C, and other external interfaces. Without proper protection, these interfaces can be easily utilized to directly or indirectly attack the security mechanisms of the chip or to access internal confidential information. In order to protect and control these interfaces, secure debugging firewalls are used.

19.2.1 summarize

The chip supports three main debugging interfaces.

1. RISCV JTAG: The RISCV processor has a built-in debugging interface that allows users to access ARM internal registers through the JTAG interface.
2. I2C: Chip provides debugging interface
3. Test interface: The chip provides a dedicated debugging interface for production testing.

For the JTAG/I2C interface, the secure debugging firewall provides specific protection controls for access to it.

For Test interface interfaces, the secure debugging interface provides a separate class of protection control (Test access) for access to these debugging classes, and the secure debugging firewall provides three connection control states

- Open: Allows external access through the connector without additional control.
- Protected: No external access is allowed until the password is entered through the I2C interface to unprotect the device.
- Closed: External access through this interface is not allowed, and cannot be re-opened by other means.

19.2.2 Status Inquiry and Password Input Interface (I2C)

The secure debugging firewall provides a standalone I2C interface for the external chip to check the current status of the debugging interface via I2C and enter the corresponding password to reopen the interface in the protected state. An external I2C ID is required to connect to the firewall interface.

The I2C interface register addresses are as follows

Table *19-1* Status Query *I2C Interface Register* Addresses

byte address	bitwidth	signal name	description
0	32	i2c_REE_password [31:0]	128-bit debugging interface cipher
4	32	i2c_REE_password [63:32]	128-bit debugging interface cipher
8	32	i2c_REE_password [95:64]	128-bit debugging interface cipher
C	32	i2c_REE_password [127:96]	128-bit debugging interface cipher
10	32	Reserved	reservations

byte address	bitwidth	signal name	description
14	32	Reserved	reservations
18	32	Reserved	reservations
1C	32	Reserved	reservations
20	32	Reserved	reservations
24	32	Reserved	reservations
28	32	Reserved	reservations
2C	32	Reserved	reservations
30	32	i2c_TST_password [31:0]	128-bit password entry field for test ports
34	32	i2c_TST_password [63:32]	128-bit test debugging interface cipher
38	32	i2c_TST_password [95:64]	128-bit test debugging interface cipher
3C	32	i2c_TST_password [127:96]	128-bit test debugging interface cipher
40	1	REE_PW_update	Update Password Comparison Results
44	1	Reserved	reservations
48	1	Reserved	reservations
4C	1	TST_PW_update	Updated test interface password comparison results
80	32	Chip_UID0 (LSB of ID)	Device ID
84	32	Chip_UID1 (MSB of ID)	Device ID
88	32	MKTSEG	Market Segment Number
8C	32	DBG_MODE	Chip Debug Interface Protection Setup Status [1:0] : Interface protected mode, 0:open;1:protected:2/3:closed [3:2] : Reservations [5:4] : Reservations [7:6] : Test interface protected mode, 0:open;1:protected:2/3:closed [8]: Compare using HASH
90	32	reserved	reservations
94	4	DBG_PROT_STATUS	Current state of debugging protection [0] : Interface protection state, 0:open 1:closed [1] :: Reservations [2] :: Reservations [3] ::Test interface protection state, 0:open 1:closed

19.2.3 Status Inquiry and Password Entry Process

19.2.3.1 Status Inquiry Process

- (Step 1) Control the external I2C to send out the start signal.
- (Step 2) Firewall I2C ID from I2C (preset 0x56)
- (Step 3) Read the address 0x04001A94 from I2C to get the current debug interface protection status.

19.2.3.2 Password Entry Process

- (Step 1) Control the external I2C to send out the start signal.
- (Step 2) I2C ID of debug interface firewall from I2C (preset 0x56)

- (Step 3) Read address 0x04001A80/0x04001A84 from I2C to get the device serial number Device ID, read 0x04001A88 to get the market distinction number

- (Step 4) Prepare the unlock code corresponding to each category by using the serial number of the equipment and the market differentiation number.
- (Step 5) Read the address 0x94 from I2C to get the current protection status of the debug interface and recognize whether it is locked or not.
- (Step 6) Take the non-secure debugging interface as an example, the non-secure password is written to the address by I2C. 0x04001A00/0x04001A04/0x04001A08/0x04001A0C
- (Step 7) I2C Write arbitrary value to address 0x04001A10 to update the password compare value
- (Step 5) Read the address 0x94 from I2C to get the current protection status of the debug interface and recognize whether to unlock or not.

19.3 Efuse Controller

4Kbit eFuse space is integrated inside the chip, and Efuse is programmed and read with Efuse Ctrl.

The main features of Efuse Ctrl include:

- Provides a double bit eFuse protection mechanism, consisting of two physical eFuse bits that form a single bit logic RMS, which is equivalent to providing 2Kbit of transient memory space to improve the robustness of eFuse burning or data retention
- Power-On-Reset automatically loads the efuse contents into the registers, which provides the necessary configuration settings for the chip system and reduces the number of times the efuse needs to be read to improve the service life.
- Provides efuse programming, read, verify read and power-down commands and content security protection mechanisms.

The Efuse data registers are divided into two areas, a non-secure area and a secure area. The non-secure area is accessible to all modules, while the secure area is accessible only to secure modules. The non-secure area stores system configuration and public messages, while the secure area stores security configuration, keys and passwords.

Figure *19-5 eFuse CTRL* Module Architecture