

***Rockchip  
RK3228  
Technical Reference Manual***

**Revision 0.1  
Oct. 2015**

Rockchip Confidential

## **Revision History**

<b>Date</b>	<b>Revision</b>	<b>Description</b>
2015-10-16	0.1	Initial Release

Rockchip Confidential

## Table of Content

Table of Content .....	3
Figure Index .....	6
Table Index.....	7
Warranty Disclaimer.....	8
Chapter 1 Introduction .....	9
1.1 Features .....	9
1.2 Block Diagram .....	18
Chapter 2 System Overview .....	19
2.1 Address Mapping.....	19
2.2 System Boot.....	19
2.3 System Interrupt connection.....	21
2.4 System DMA hardware request connection.....	23
Chapter 3 Clock & Reset Unit (CRU) .....	24
3.1 Overview .....	24
3.2 Block Diagram .....	24
3.3 System Reset Solution .....	24
3.4 Function Description .....	25
3.5 PLL Introduction.....	25
3.6 Register Description.....	26
3.7 Timing Diagram .....	83
3.8 Application Notes .....	84
Chapter 4 Embedded Processor (Cortex-A7).....	87
4.1 Overview .....	87
4.2 Block Diagram .....	87
4.3 Function Description .....	87
4.4 Register Description.....	87
Chapter 5 General Register Files (GRF).....	88
5.1 Overview .....	88
5.2 Function Description .....	88
5.3 Register Description.....	88
Chapter 6 Secure General Register Files (SGRF) .....	197
6.1 Overview .....	197
6.2 Function Description .....	197
6.3 Register Description.....	197
Chapter 7 Interconnect.....	226
7.1 Overview .....	226
7.2 Block Diagram .....	226
7.3 Function Description .....	226
7.4 Register Description.....	228
7.5 Application Notes .....	233
Chapter 8 DMC (Dynamic Memory Interface).....	234
8.1 Overview .....	234
8.2 Block Diagram .....	235
8.3 Function Description .....	235
8.4 Register Description.....	237
8.5 Interface Description .....	293
8.6 Application Notes .....	294
Chapter 9 Embedded SRAM.....	302

---

9.1 Overview .....	302
9.2 Block Diagram .....	302
9.3 Function Description .....	302
Chapter 10 Nand Flash Controller (NandC).....	303
10.1 Overview.....	303
10.2 Block Diagram .....	304
10.3 Function Description .....	304
10.4 Register Description.....	305
10.5 Interface Description.....	427
10.6 Application Notes.....	428
Chapter 11 Timer.....	436
11.1 Overview.....	436
11.2 Block Diagram .....	436
11.3 Function Description .....	436
11.4 Register Description.....	437
11.5 Application Notes.....	439
Chapter 12 Generic Interrupt Controller (GIC) .....	440
12.1 Overview.....	440
12.2 Block Diagram .....	440
12.3 Function Description .....	440
12.4 Register Description.....	443
12.5 Interface Description.....	451
12.6 Application Notes.....	451
Chapter 13 DMA Controller(DMAC).....	454
13.1 Overview.....	454
13.2 Block Diagram .....	454
13.3 Function Description .....	455
13.4 Register Description.....	456
13.5 Timing Diagram.....	472
13.6 Interface Description.....	472
13.7 Application Notes.....	473
Chapter 14 System Security .....	478
14.1 Overview.....	478
14.2 Block Diagram .....	478
14.3 Function Description .....	478
14.4 Application Notes.....	483
Chapter 15 EFUSE.....	485
15.1 Overview.....	485
15.2 Block Diagram .....	485
15.3 Function Description .....	485
15.4 Register Description.....	486
15.5 Timing Diagram.....	487
15.6 Application Notes.....	489
Chapter 16 WatchDog .....	490
15.1   Overview .....	490
15.2   Block Diagram.....	490
15.3   Function Description.....	490
15.4   Register Description .....	491
15.5   Application Notes.....	494
Chapter 17 System Debug .....	495
17.1 Overview.....	495

17.2 Block Diagram .....	495
17.3 Function Description .....	495
17.4 Register Description.....	495
17.5 Interface Description.....	496

Rockchip Confidential

## Figure Index

Fig. 1-1 RK3228 Block Diagram .....	18
Fig. 2-1 RK3228 Address Mapping.....	19
Fig. 2-2 RK3228 boot procedure flow.....	20
Fig. 3-1 CRU Architecture .....	24
Fig. 3-2 Reset Architecture Diagram .....	24
Fig. 3-3 PLL Block Diagram .....	26
Fig. 3-4 Chip Power On Reset Timing Diagram .....	84
Fig. 4-1 MP Subsystem architecture .....	87
Fig. 7-1 Interconnect diagram .....	226
Fig. 7-2 Idle request .....	233
Fig. 9-1 Embedded SRAM block diagram .....	302
Fig. 10-1 NandC Block Diagram .....	304
Fig. 10-2 NandC Address Assignment .....	431
Fig. 10-3 NandC Data Format .....	431
Fig. 10-4 NandC LLP Data Format .....	434
Fig. 10-5 NandC LLI Data Format.....	434
Fig. 10-6 FIFO Data Format.....	435
Fig. 11-1 Timer Block Diagram .....	436
Fig. 11-2 Timer Usage Flow.....	437
Fig. 11-3 Timing between timer_en and timer_clk .....	439
Fig. 12-1 GIC Block Diagram .....	440
Fig. 12-2 GIC Interrupt handling state machine .....	442
Fig. 13-1 Block diagram of DMAC.....	454
Fig. 13-2 DMAC operation states.....	455
Fig. 13-3 DMAC request and acknowledge timing .....	472
Fig. 14-1 RK3228 security architecture .....	478
Fig. 14-2 TZMA block diagram .....	479
Fig. 14-3 DMAC_BUS interface .....	479
Fig. 14-4 Software Diagram of Secure and Non-secure.....	484
Fig. 14-5 Embedded SRAM secure memory space setting .....	484
Fig. 15-1 RK3228 eFuse block diagram .....	485
Fig. 15-2 RK3228 efuse32×8 timing diagram in program mode .....	487
Fig. 15-3 RK3228 efuse32×8 timing diagram in read mode.....	487
Fig. 15-4 RK3228 efuse32×32 timing diagram in program mode .....	488
Fig. 15-5 RK3228 efuse32×32 timing diagram in read mode .....	488
Fig. 16-1 WDT block diagram .....	490
Fig. 16-2 WDT Operation Flow .....	491
Fig. 17-1 Debug system structure .....	495
Fig. 17-2 DAP SWJ interface .....	496
Fig. 17-3 SW-DP acknowledgement timing .....	496

## Table Index

Table 2-1 RK3228 Interrupt connection list .....	21
Table 2-2 RK3228 DMAC Hardware request connection list.....	23
Table 7-1 Master NIU .....	226
Table 7-2 DDR configuration.....	228
Table 7-3 Service Module Address .....	228
Table 8-1 DDR IO description .....	293
Table 8-2 DDR PHY TX DLLs Delay Step .....	298
Table 8-3DDR PHY RX DQS Delay Step .....	298
Table 8-4 DM/DQ/DQS/CMD Driver output resistance.....	299
Table 8-5 DM, DQ Signal Drive Strength Register .....	299
Table 8-6 DM/DQ/DQS/CMD Driver output resistance with control bit .....	299
Table 8-7 DM/DQ/DQS RX ODT resistance with control bit .....	300
Table 8-8 Low Power DLL Setting .....	300
Table 8-9 per-bit de-skew tuning resolution .....	300
Table 10-1 NandC Address Mapping.....	305
Table 10-2 NandC Interface Description .....	427
Table 10-3 NandC Interface Connection .....	428
Table 10-4 NandC Page/Spare size for flash.....	432
Table 13-1 DMAC Request Mapping Table .....	454
Table 13-2 DMAC boot interface.....	472
Table 13-3 Source size in CCRn.....	476
Table 13-4 DMAC Instruction sets .....	476
Table 13-5 DMAC instruction encoding .....	477
Table 14-1 bus components security setting .....	480
Table 14-2 RK3228 secure device setting .....	481
Table 14-3 RK3228 device secure input port setting.....	483
Table 15-1 list of allowed modes .....	486
Table 15-2 RK3228 eFuse timing parameters list.....	488
Table 17-1 SW-DP Interface Description .....	496

## **Warranty Disclaimer**

Rockchip Electronics Co., Ltd makes no warranty, representation or guarantee (expressed, implied, statutory, or otherwise) by or with respect to anything in this document, and shall not be liable for any implied warranties of non-infringement, merchantability or fitness for a particular purpose or for any indirect, special or consequential damages.

Information furnished is believed to be accurate and reliable. However, Rockchip Electronics Co., Ltd assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties that may result from its use.

Rockchip Electronics Co., Ltd's products are not designed, intended, or authorized for using as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Rockchip Electronics Co., Ltd's product could create a situation where personal injury or death may occur, should buyer purchase or use Rockchip Electronics Co., Ltd's products for any such unintended or unauthorized application, buyers shall indemnify and hold Rockchip Electronics Co., Ltd and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, expenses, and reasonable attorney fees arising out of, either directly or indirectly, any claim of personal injury or death that may be associated with such unintended or unauthorized use, even if such claim alleges that Rockchip Electronics Co., Ltd was negligent regarding the design or manufacture of the part.

### **Copyright and Patent Right**

Information in this document is provided solely to enable system and software implementers to use Rockchip Electronics Co., Ltd's products. There are no expressed or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

**Rockchip Electronics Co., Ltd does not convey any license under its patent rights nor the rights of others.**

**All copyright and patent rights referenced in this document belong to their respective owners and shall be subject to corresponding copyright and patent licensing requirements.**

### **Trademarks**

Rockchip and Rockchip™ logo and the name of Rockchip Electronics Co., Ltd's products are trademarks of Rockchip Electronics Co., Ltd. and are exclusively owned by Rockchip Electronics Co., Ltd. References to other companies and their products use trademarks owned by the respective companies and are for reference purpose only.

### **Confidentiality**

The information contained herein (including any attachments) is confidential. The recipient hereby acknowledges the confidentiality of this document, and except for the specific purpose, this document shall not be disclosed to any third party.

### **Reverse engineering or disassembly is prohibited.**

**ROCKCHIP ELECTRONICS CO.,LTD. RESERVES THE RIGHT TO MAKE CHANGES IN ITS PRODUCTS OR PRODUCT SPECIFICATIONS WITH THE INTENT TO IMPROVE FUNCTION OR DESIGN AT ANY TIME AND WITHOUT NOTICE AND IS NOT REQUIRED TO UNDATE THIS DOCUMENTATION TO REFLECT SUCH CHANGES.**

### **Copyright © 2015 Rockchip Electronics Co., Ltd.**

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electric or mechanical, by photocopying, recording, or otherwise, without the prior written consent of Rockchip Electronics Co., Ltd.

## Chapter 1 Introduction

RK3228 is a high performance Quad-core application processor for smart TV-Box. Especially it is a high-integration and cost efficient SOC for 4K 10-bit H.265/H.264/VP9 TV-Box.

Quad-core Cortex-A7 is integrates with separately Neon and FPU coprocessor, also shared 256KB L2 Cache. Mali400 MP2 GPU is embedded to support smoothly high-resolution display and mainstream game.

Lots of high-performance interface to get very flexible solution, such as multi-pipe display with HDMI2.0, TV Encoder. Trust Zone and crypto hardware is integrated for support security BOOT. 32bits DDR3/LPDDR3 provides high memory bandwidths for high-performance.

### 1.1 Features

The features listed below which may or may not be present in actual product, may be subject to the third party licensing requirements. Please contact Rockchip for actual product feature configurations and licensing requirements.

#### 1.1.1 Microprocessor

- Quad-core ARM Cortex-A7MP Core processor, a high-performance, low-power and cached application processor
- Full implementation of the ARM architecture v7-A instruction set, ARM Neon Advanced SIMD (single instruction, multiple data) support for accelerated media and signal processing computation
- Separately Integrated Neon and FPU per CPU
- 32KB/32KB L1 I-Cache/D-Cache per CPU.
- Unified 256KB L2 Cache.
- Trustzone technology support

#### 1.1.2 Memory Organization

- Internal on-chip memory
  - BootRom
  - Internal SRAM
- External off-chip memory<sup>®</sup>
  - DDR3/DDR3L/LPDDR2/LPDDR3
  - Async/Toggle/SyncNand Flash(include LBA Nand)

#### 1.1.3 Internal Memory

- Internal BootRom
  - Size : 20KB
  - Support system boot from the following device :
    - ◆ 8bits Async Nand Flash
    - ◆ 8bits toggle Nand Flash
    - ◆ SPI interface
    - ◆ eMMC interface
    - ◆ SDMMC interface
  - Support system code download by the following interface:
    - ◆ USB OTG interface
- Internal SRAM
  - Size : 36KB

#### 1.1.4 External Memory or Storage device

- Dynamic Memory Interface (DDR3/DDR3L/LPDDR2/LPDDR3)
  - Compatible with JEDEC standard
  - DDR3-1600/DDR3L-1600/LPDDR2-800/LPDDR3-1333 SDRAM
  - Supports 32 Bits data width, 2 ranks (chip selects), totally 2GB (max) address space.

- Programmable timing parameters to support DDR3/DDR3L/LPDDR2/LPDDR3 SDRAM from various vendor
- Advanced command reordering and scheduling to maximize bus utilization
- Low power modes, such as power-down and self-refresh for DDR3/LPDDR2/LPDDR3 SDRAM; clock stop and deep power-down for LPDDR2 SDRAM
- Compensation for board delays and variable latencies through programmable pipelines
- Programmable output and ODT impedance with dynamic PVT compensation
- Nand Flash Interface
  - Support 8bits async/toggle/sync nandflash, up to 4 banks
  - Support LBA nandflash
  - 16bits, 24bits, 40bits, 60bits hardware ECC
  - For DDR nandflash, support DLL bypass and 1/4 or 1/8 clock adjust
  - For async/toggle nandflash, support configurable interface timing, maximum data rate is 16bit/cycle
  - Embedded AHB master interface to do data transfer by DMA method
- eMMC Interface
  - Compatible with standard iNAND interface
  - Support MMC4.51 protocol
  - Provide eMMC boot sequence to receive boot data from external eMMC device
  - Support FIFO over-run and under-run prevention by stopping card clock automatically
  - Support CRC generation and error detection
  - Embedded clock frequency division control to provide programmable baud rate
  - Support block size from 1 to 65535Bytes
  - 8bits data bus width
- SD/MMC Interface
  - Compatible with SD3.0, MMC ver4.51
  - Support FIFO over-run and under-run prevention by stopping card clock automatically
  - Support CRC generation and error detection
  - Support block size from 1 to 65535Bytes
  - Data bus width is 4bits

### **1.1.5 System Component**

- CRU (clock & reset unit)
  - Support clock gating control for individual components inside RK3228
  - One oscillator with 24MHz clock input and 4 embedded PLLs
  - Support global soft-reset control for whole SOC, also individual soft-reset for every components
- Timer
  - 6 on-chip 64bits Timers in SoC with interrupt-based operation for non-secure application
  - 2 on-chip 64bits Timers in SoC with interrupt-based operation for secure application
  - Provide two operation modes: free-running and user-defined count
  - Support timer work state checkable
  - Fixed 24MHz clock input
- PWM
  - Four on-chip PWMs with interrupt-based operation
  - Programmable pre-scaled operation to bus clock and then further scaled
  - Embedded 32-bit timer/counter facility
  - Support capture mode
  - Support continuous mode or one-shot mode
  - Provides reference mode and output various duty-cycle waveform

- WatchDog
  - 32 bits watchdog counter width
  - Counter clock is from apb bus clock
  - Counter counts down from a preset value to 0 to indicate the occurrence of a timeout
  - WDT can perform two types of operations when timeout occurs:
    - ◆ Generate a system reset
    - ◆ First generate an interrupt and if this is not cleared by the service routine by the time a second timeout occurs then generate a system reset
  - Programmable reset pulse length
  - Totally 16 defined-ranges of main timeout period
- Bus Architecture
  - 128bit/64-bit/32-bit multi-layer AXI/AHB/APB composite bus architecture
  - 5 embedded AXI interconnect
    - ◆ CPU interconnect with four 64-bits AXI masters, one 64-bits AXI slaves, one 32-bits AHB master and lots of 32-bits AHB/APB slaves
    - ◆ PERI interconnect with two 64-bits AXI masters, one 64-bits AXI slave, five 32-bits AHB masters and lots of 32-bits AHB/APB slaves
    - ◆ Display interconnect with three 128-bits AXI master, four 64-bits AXI masters and one 32-bits AHB slave
    - ◆ GPU interconnect with one 128-bits AXI master with point-to-point AXI-lite architecture and 32-bits APB slave
    - ◆ VCODEC interconnect also with two 64-bits AXI master and two 32-bits AHB slave, they are point-to-point AXI-lite architecture
  - Flexible different QoS solution to improve the utility of bus bandwidth
- Interrupt Controller
  - Support 3 PPI interrupt source and 128 SPI interrupt sources input from different components inside RK3228
  - Support 16 software-triggered interrupts
  - Input interrupt level is fixed , only high-level sensitive
  - Two interrupt outputs (nFIQ and nIRQ)separatelyfor each Cortex-A7, both are low-level sensitive
  - Support different interrupt priority for each interrupt source, and they are always software-programmable
- DMAC
  - Micro-code programming based DMA
  - The specific instruction set provides flexibility for programming DMA transfers
  - Linked list DMA function is supported to complete scatter-gather transfer
  - Support internal instruction cache
  - Embedded DMA manager thread
  - Support data transfer types with memory-to-memory, memory-to-peripheral, peripheral-to-memory
  - Signals the occurrence of various DMA events using the interrupt output signals
  - Mapping relationship between each channel and different interrupt outputs is software-programmable
  - One embedded DMA controller for system
  - DMAC features:
    - ◆ 8 channels totally
    - ◆ 16 hardware request from peripherals
    - ◆ 2 interrupt output
    - ◆ Dual APB slave interface for register configuration, designated as secure and non-secure
    - ◆ Support trustzone technology and programmable secure state for each DMA channel

- Security system
  - Support trustzone technology for the following components inside RK3228
    - ◆ Cortex-A7, support security and non-security mode, switch by software
    - ◆ DMAC, support some dedicated channels work only in security mode
    - ◆ eFuse, only accessed by Cortex-A7 in security mode
    - ◆ Internal memory , part of space is addressed only in security mode, detailed size is software-programmable together with TZMA(trustzone memory adapter) and TZPC(trustzone protection controller)
  - Embedded encryption and decryption engine
    - ◆ Support AES 128/192/256 bits key mode, ECB/CBC/CTR chain mode, Slave/FIFO mode
    - ◆ Support DES/3DES (ECB and CBC chain mode) , 3DES (EDE/ EEE key mode), Slave/FIFO mode
    - ◆ Support SHA1/SHA256/MD5 (with hardware padding) HASH function, FIFO mode only
    - ◆ Support 160 bit Pseudo Random Number Generator (PRNG)
    - ◆ Support PKA 512/1024/2048 bit Exp Modulator

### 1.1.6 Video CODEC

- Embedded memory management unit(MMU)
- Video Decoder
  - Real-time video decoder of MPEG-1, MPEG-2, MPEG-4,H.263, H.264, H.265,VC-1, VP9,VP8, MVC
  - MMU Embedded
  - Supports frame timeout interrupt , frame finish interrupt and bitstream error interrupt
  - Error detection and concealment support for all video formats
  - Output data format YUV420 semi-planar,YUV400(monochrome) ,YUV422 is supported by H.264
  - H.264 10bit up to HP level 5.1 : 2160p@30fps (4096x2304)
  - VP9 :2160p@30fps(4096x2304)
  - HEVC 10bit: 2160p @60fps(4096x2304)
  - MPEG-4 up to ASP level 5 : 1080p@60fps (1920x1088)
  - MPEG-2 up to MP : 1080p@60fps (1920x1088)
  - MPEG-1 up to MP : 1080p@60fps (1920x1088)
  - H.263 : 576p@60fps (720x576)
  - VC-1 up to AP level 3 : 1080p@30fps (1920x1088)
  - VP8 : 1080p@60fps (1920x1088)
  - MVC : 1080p@60fps (1920x1088)
  - For H.264, image cropping not supported
  - For MPEG-4, GMC(global motion compensation) not supported
  - For VC-1, upscaling and range mapping are supported in image post-processor
  - For MPEG-4 SP/H.263, using a modified H.264 in-loop filter to implement deblocking filter in post-processor unit
- Video Encoder
  - Support video encoder for H.264 UP to HP@level4.1, MVC and VP8
  - Only support I and P slices, not B slices
  - Support error resilience based on constrained intra prediction and slices
  - Input data format:
    - ◆ YCbCr 4:2:0 planar
    - ◆ YCbCr 4:2:0 semi-planar
    - ◆ YCbYCr 4:2:2
    - ◆ CbYCrY 4:2:2 interleaved
    - ◆ RGB444 and BGR444
    - ◆ RGB555 and BGR555

- ◆ RGB565 and BGR565
- ◆ RGB888 and BRG888
- ◆ RGB101010 and BRG101010
- Image size is from 96x96 to 1920x1080(Full HD)
- Maximum frame rate is up to 1920x1080 @ 30FPS<sup>®</sup>

### 1.1.7 JPEG CODEC

- JPEG decoder
  - Input JPEG file : YCbCr 4:0:0, 4:2:0, 4:2:2, 4:4:0, 4:1:1 and 4:4:4 sampling formats
  - Output raw image : YCbCr 4:0:0, 4:2:0, 4:2:2, 4:4:0, 4:1:1 and 4:4:4 semi-planar
  - Decoder size is from 48x48 to 8176x8176(66.8Mpixels)
  - Support JPEG ROI(region of image) decode
  - Maximum data rate<sup>®</sup> is up to 76million pixels per second
  - Embedded memory management unit(MMU)

### 1.1.8 Image Enhancement (IEP module)

- Image format support
  - Input data: YUV420/YUV422
  - Output data: YUV420/YUV422
  - YUV swap
  - UV SP/P
  - BT601\_I/BT601\_f/BT709\_I/BT709\_f color space conversion
  - YUV up/down sampling
- De-interlace
  - 3x5 Y motion detection matrix
  - Source width up to 1920
  - Configured high frequency de-interlace
  - I4O2 (Input 4 field, output 2 frame) /I4O1B/I4O1T/I2O1B/I2O1T mode
- Interface
  - 32bit AHB bus slave
  - 64bit AXI bus master
  - Combined interrupt output

### 1.1.9 Graphics Engine

- 3D Graphics Engine :
  - High performance OpenGL ES1.1 and 2.0, OpenVG1.1 etc.
  - Embedded 2 shader cores with shared hierarchical tiler
  - Separate vertex(geometry) and fragment(pixel) processing for maximum parallel throughput
  - Provide MMU and L2 Cache with 64KB size
- 2D Graphics Engine(RGA module) :
  - Source formats :
    - ◆ ABGR8888, XBGR888, ARGB8888, XRGB888
    - ◆ RGB888, RGB565
    - ◆ RGBA5551, RGBA4444
    - ◆ YUV420 planar, YUV420 semi-planar
    - ◆ YUV422 planar, YUV422 semi-planar
    - ◆ YUV 10-bit for YUV420/422 semi-planar
    - ◆ BPP8, BPP4, BPP2, BPP1
  - Destination formats :
    - ◆ ABGR8888, XBGR888, ARGB8888, XRGB888
    - ◆ RGB888, RGB565
    - ◆ RGBA5551, RGBA4444
    - ◆ YUV420 planar, YUV420 semi-planar
    - ◆ YUV422 planar, YUV422 semi-planar

- Pixel Format conversion, BT.601/BT.709
- Max resolution: 8192x8192 source, 4096x4096 destination
- BitBLT
  - ◆ Two source BitBLT:
  - ◆ A+B=B only BitBLT, A support rotate&scale when B fixed
  - ◆ A+B=C second source (B) has same attribute with (C) plus rotation function
- Color fill with gradient fill, and pattern fill
- High-performance stretch and shrink
- Monochrome expansion for text rendering
- New comprehensive per-pixel alpha(color/alpha channel separately)
- Alpha blending modes including Java 2 Porter-Duff compositing blending rules , chroma key, pattern mask, fading
- Dither operation
- 0, 90, 180, 270 degree rotation
- x-mirror, y-mirror & rotation operation

### 1.1.10 Video OUT

- Display Interface
  - Support HDMI 2.0 output up to 4K@60Hz
  - TV Interface: TV encoder 10bit out for DAC
  - HDMI Interface : 24 bit(RGB888 YCbCr444),  
30 bit(RGB101010,YCbCr 420, YCbCr 444)
  - Max output resolution 4K for HDMI, 480i/576i for CVBS
  - 3 display layers :
    - ◆ Display layers of Win0,Win1,HWC
    - ◆ One background layer with programmable 24bits color
    - ◆ One video layer (win0/win1)
      - RGB888, ARGB888, RGB565, YCbCr422, YCbCr420, YCbCr444,YCbCr 420 10bit, YCbCr 422 10bit, YCbCr 444 10bit
      - maximum resolution is 4096x2304,support virtual display
      - 1/8 to 8 scaling up/down engine with arbitrary non-integer ratio
      - 256 level alpha blending(pre-multiplied alpha support)
      - Support transparency color key
      - Support BG, RG, RB swap, xy mirror
      - Support TV Encoder for PAL and NTSC
      - YCbCr2RGB(rec601-mpeg/rec601-jpeg/rec709)
      - YCbCr2RGB(BT2020)
      - RGB2YCbCr(BT601/BT709)
      - RGB2YCbCr(BT2020)
  - BT2020 and BT709/601 conversion
  - Win0 and Win1 layer overlay exchangeable
  - Support replication(16bits to 24bits) and dithering(24bits to 16bits/ 18bits) operation
  - Blank and blank display

### 1.1.11 HDMI

- Support YUV420 4k x 2k @ 60fps
- Support for 4k x 2k and 3D video formats
- Support for up to 10.2bps bandwidth
- HPD input analog comparator
- Compliant HDMI 2.0
- Compliance HDMI compliance Test specification 1.4
- Support HDCP 2.2

### 1.1.12 Audio Interface

- I2S0/I2S1 with 8ch

- I2S0/I2S1 supports up to 8 channels (8xTX, 8xRX)
- Audio resolution from 16bits to 32bits
- Sample rate up to 192KHz
- Provides master and slave work mode, software configurable
- Support 3 I2S formats (normal, left-justified, right-justified)
- Support 4 PCM formats(early, late1, late2, late3)
- I2S and PCM mode cannot be used at the same time
- I2S2/PCM with 2ch
  - Up to 2 channels (2xTX, 2xRX)
  - Audio resolution from 16bits to 32bits
  - Sample rate up to 192KHz
  - Provides master and slave work mode, software configurable
  - Support 3 I2S formats (normal , left-justified , right-justified)
  - Support 4 PCM formats(early , late1 , late2 , late3)
  - I2S and PCM cannot be used at the same time
- SPDIF
  - Support two 16-bit audio data store together in one 32-bit wide location
  - Support biphase format stereo audio data output
  - Support 16 to 31 bit audio data left or right justified in 32-bit wide sample data buffer
  - Support 16, 20, 24 bits audio data transfer in linear PCM mode
  - Support non-linear PCM transfer
- Audio CODEC
  - 24bit DAC with 95dB SNR
  - Support Line-out
  - Support Mono, Stereo, 5.1 HiFi channel performance
  - Integrated digital interpolation and decimation filter.
  - Sampling rate of 8kHz/12kHz/16kHz/24kHz/32kHz/44.1kHz/48kHz/96kHz
  - Optional fractional PLL available that support 6MHz to 20MHz clock input to any clock

### 1.1.13 Connectivity

- SDIO interface
  - Compatible with SDIO 3.0 protocol
  - 4bits data bus widths
- TS interface
  - Supports one TS input channels.
  - Supports 4 TS Input Mode: sync/valid mode in the case of serial TS input; nosync/valid mode, sync/valid, sync/burst mode in the case of parallel TS input.
  - Supports 2 TS sources: demodulators and local memory.
  - Supports 2 Built-in PTIs(Programmable Transport Interface) to process TS simultaneously, and Each PTI supports:
    - ◆ 64 PID filters.
    - ◆ TS descrambling with 16 sets of Control Word under CSA v2.0 standard, up to 104Mbps
    - ◆ 16 PES/ES filters with PTS/DTS extraction and ES start code detection.
    - ◆ 4/8 PCR extraction channels
    - ◆ 64 Section filters with CRC check, and three interrupt mode: stop per unit, full-stop, recycle mode with version number check
    - ◆ PID done and error interrupts for each channel
    - ◆ PCR/DTS/PTS extraction interrupt for each channel
  - Supports 1 PVR(Personal Video Recording) output channel.
  - 1 built-in multi-channel DMA Controller.
- Smart Card

- support card activation and deactivation
- support cold/warm reset
- support Answer to Reset (ATR) response reception
- support T0 for asynchronous half-duplex character transmission
- support T1 for asynchronous half-duplex block transmission
- support automatic operating voltage class selection
- support adjustable clock rate and bit (baud) rate
- support configurable automatic byte repetition
- GMAC 10/100/1000M Ethernet Controller
  - Supports 10/100/1000-Mbps data transfer rates with the RGMII interfaces
  - Supports 10/100-Mbps data transfer rates with the RMII interfaces
  - Supports both full-duplex and half-duplex operation
    - ◆ Supports CSMA/CD Protocol for half-duplex operation
    - ◆ Supports packet bursting and frame extension in 1000 Mbps half-duplex operation
    - ◆ Supports IEEE 802.3x flow control for full-duplex operation
    - ◆ Optional forwarding of received pause control frames to the user application in full-duplex operation
    - ◆ Back-pressure support for half-duplex operation
    - ◆ Automatic transmission of zero-quanta pause frame on deassertion of flow control input in full-duplex operation
  - Preamble and start-of-frame data (SFD) insertion in Transmit, and deletion in Receive paths
  - Automatic CRC and pad generation controllable on a per-frame basis
  - Options for Automatic Pad/CRC Stripping on receive frames
  - Programmable Inter-Frame-Gap (40-96 bit times in steps of 8)
  - Supports a variety of flexible address filtering modes
  - Separate 32-bit status returned for transmission and reception packets
  - Supports IEEE 802.1Q VLAN tag detection for reception frames
  - Support detection of LAN wake-up frames and AMD Magic Packet frames
  - Support checksum off-load for received IPv4 and TCP packets encapsulated by the Ethernet frame
  - Support checking IPv4 header checksum and TCP, UDP, or ICMP checksum encapsulated in IPv4 or IPv6 datagram
  - Comprehensive status reporting for normal operation and transfers with errors
  - Automatic generation of PAUSE frame control or backpressure signal to the GMAC core based on Receive FIFO-fill (threshold configurable) level
  - Handles automatic retransmission of Collision frames for transmission
  - Discards frames on late collision, excessive collisions, excessive deferral and under-run conditions
- Ethernet PHY
  - Integrated IEEE 802.3/802.3u compliant 10/100Mbps Ethernet PHY
  - Supporting both full and half duplex for either 10 or 100 Mb/s data rate
  - Auto MDIX capable
  - Supports wake-on-LAN, EEE
  - 100Base-FX support
  - MII/RMII interface
  - Supports auto-negotiation
  - On-board diagnosis
- SPI Controller
  - Support serial-master and serial-slave mode, software-configurable
  - DMA-based or interrupt-based operation
  - Embedded two 32x16bits FIFO for TX and RX operation respectively
  - Support 2 chip-selects output in serial-master mode

- UART Controller
  - 3 on-chip UART controller inside RK3228
  - DMA-based or interrupt-based operation
  - UART0/1/2 Embedded two 64Bytes FIFO for TX and RX operation respectively
  - Support 5bit,6bit,7bit,8bit serial data transmit or receive
  - Standard asynchronous communication bits such as start, stop and parity
  - Support different input clock for UART operation to get up to 4Mbps or other special baud rate
  - Support non-integer clock divides for baud clock generation
  - Support auto flow control mode
- I2C controller
  - 4 on-chip I2C controller in RK3228
  - Multi-master I2C operation
  - Support 7bits and 10bits address mode
  - Software programmable clock frequency and transfer rate up to 400Kbit/s in the fast mode
  - Serial 8bits oriented and bidirectional data transfers can be made at up to 100Kbit/s in the standard mode
- GPIO
  - 4 groups of GPIO (GPIO0~GPIO3) , 32 GPIOs per group in GPIO0~GPIO3, totally have 128 GPIOs
  - All of GPIOs can be used to generate interrupt to Cortex-A7
  - All of pull-up GPIOs are software-programmable for pull-up resistor or not
  - All of pull-down GPIOs are software-programmable for pull-down resistor or not
  - All of GPIOs are always in input direction in default after power-on-reset
- USB Host2.0
  - Embedded 3 USB Host 2.0 interfaces
  - Compatible with USB Host2.0 specification
  - Supports high-speed(480Mbps), full-speed(12Mbps) and low-speed(1.5Mbps) mode
  - Provides 16 host mode channels
  - Support periodic out channel in host mode
- USB OTG2.0
  - Compatible with USB OTG2.0 specification
  - Supports high-speed(480Mbps), full-speed(12Mbps) and low-speed(1.5Mbps) mode
  - Support up to 9 device mode endpoints in addition to control endpoint 0
  - Support up to 6 device mode IN endpoints including control endpoint 0
  - Endpoints 1/3/5/7 can be used only as data IN endpoint
  - Endpoints 2/4/6 can be used only as data OUT endpoint
  - Endpoints 8/9 can be used as data OUT and IN endpoint
  - Provides 9 host mode channels

### **1.1.14 Others**

- Temperature Sensor(TS-ADC)
  - 10-bits SAR ADC up to 50KS/s sampling rate
  - -40~125C temperature range and 5C temperature resolution
- eFuse
  - Two high-density electrical Fuse is integrated: 256bits (32x8) / 1024bits (32x32)
  - Support standby mode
  - Provide inactive mode, VP must be 0V or Floating in this mode.

- Package Type
  - BGA316 (body: 14mm x 14mm ; ball size : 0.3mm ; ball pitch : 0.65mm)

**Notes :**

- ① : DDR3/LPDDR2/LPDDR3 are not used simultaneously as well as async and sync ddrnand flash
- ②: Actual maximum frame rate will depend on the clock frequency and system bus performance
- ③: Actual maximum data rate will depend on the clock frequency and JPEG compression rate

## 1.2 Block Diagram

The following diagram shows the basic block diagram for RK3228.

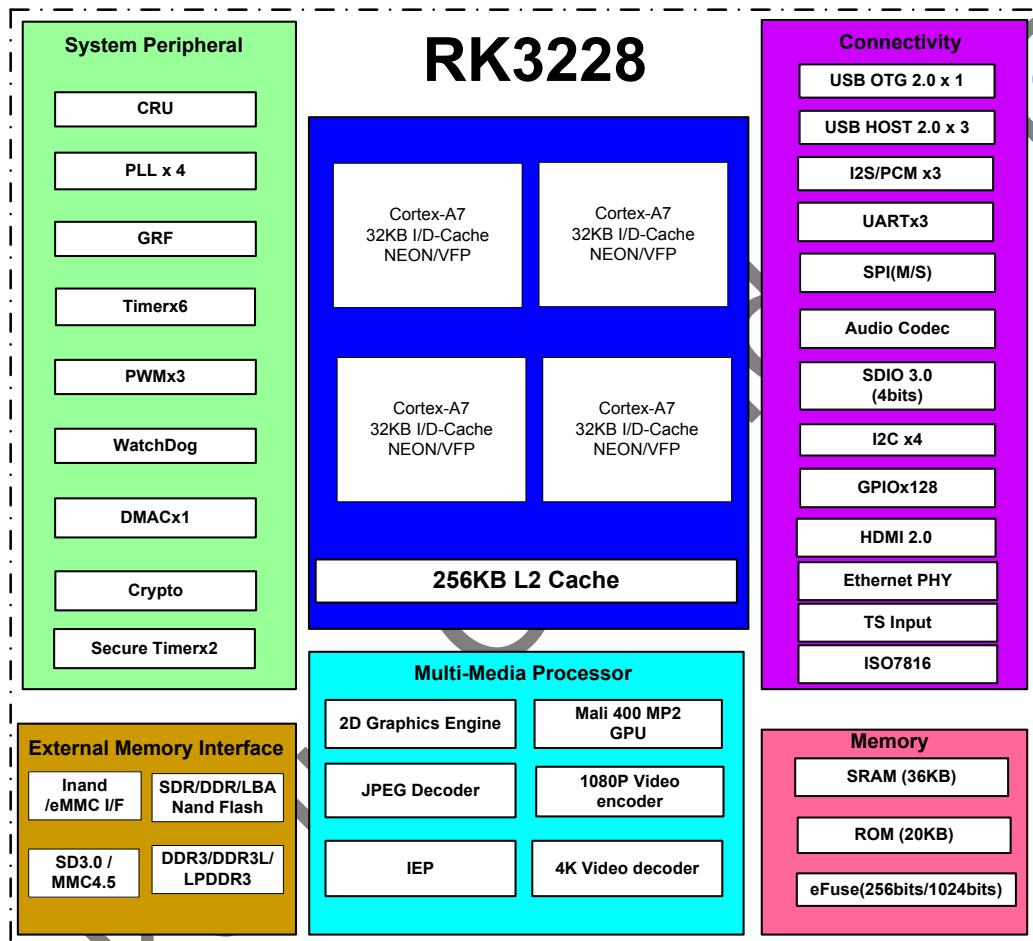


Fig. 1-1 RK3228 Block Diagram

## Chapter 2 System Overview

### 2.1 Address Mapping

RK3228 supports to boot from internal bootrom, which supports remap function by software programming. Remap is controlled by GRF\_SOC\_CON0[10]. When remap is set to 1, the bootrom is mapped to address 0x10100000 and internal memory is mapped to address 0x0.

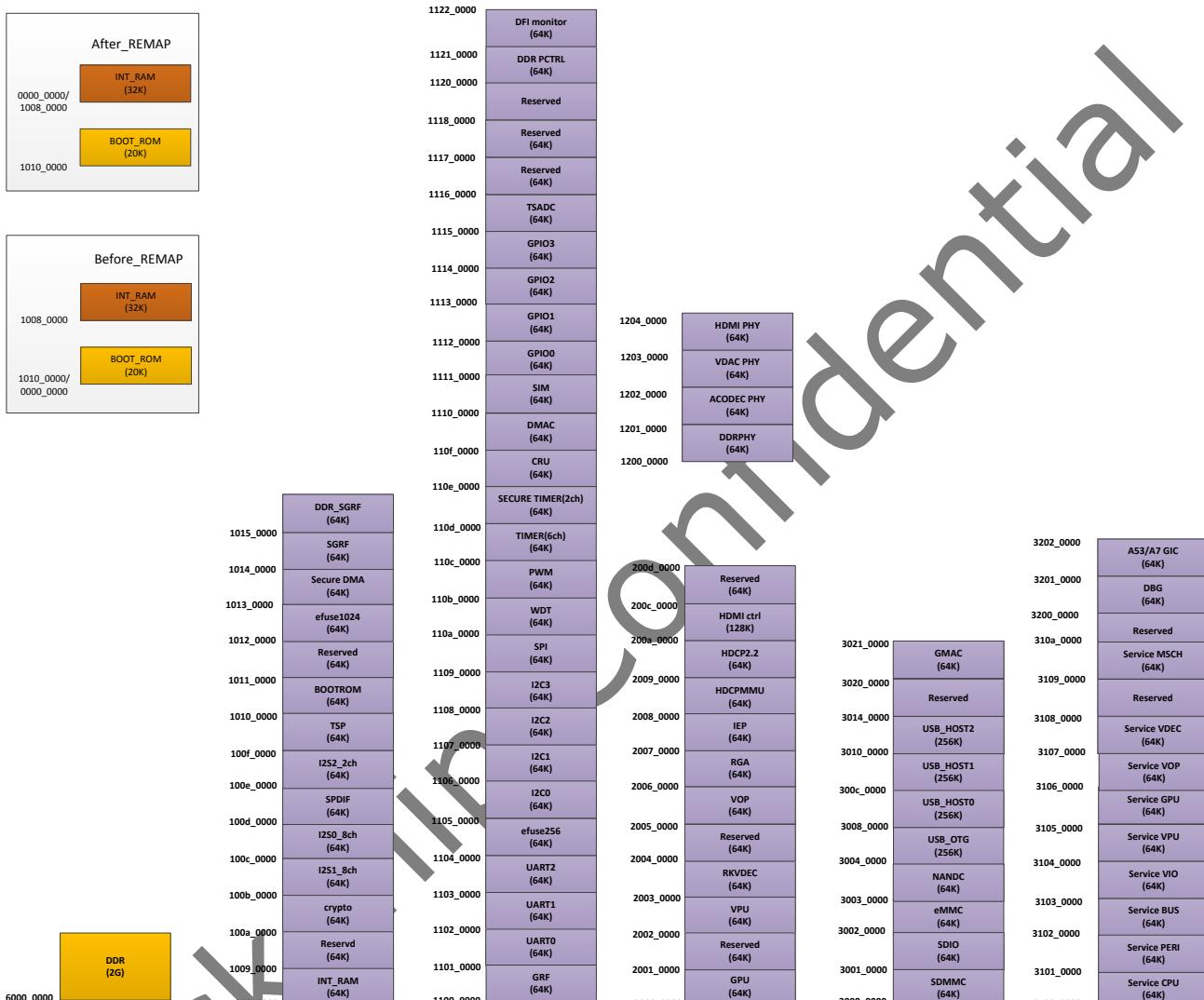


Fig. 2-1 RK3228 Address Mapping

### 2.2 System Boot

RK3228 provides system boot from off-chip devices such as SDMMC card, eMMC memory, 8bits Asynchronous and toggle nand flash, serial nand or nor flash. When boot code is not ready in these devices, also provide system code download into them by USB OTG interface. All of the boot code will be stored in internal bootrom. The following is the whole boot procedure for boot code, which will be stored in bootrom in advance.

The following features are supports.

- Support secure boot mode and non-secure boot mode
- Support system boot from the following device:
  - Asynchronous and toggle Nand Flash, 8bits data width
  - Serial Nand Flash/SPI Nand Flash, 1bit data width
  - Serial Nor Flash, 1bit data width
  - eMMC Interface, 8bits data width

- SDMMC Card, 4bits data width
  - Support system code download by USB OTG
- Following figure shows RK3228 boot procedure flow.

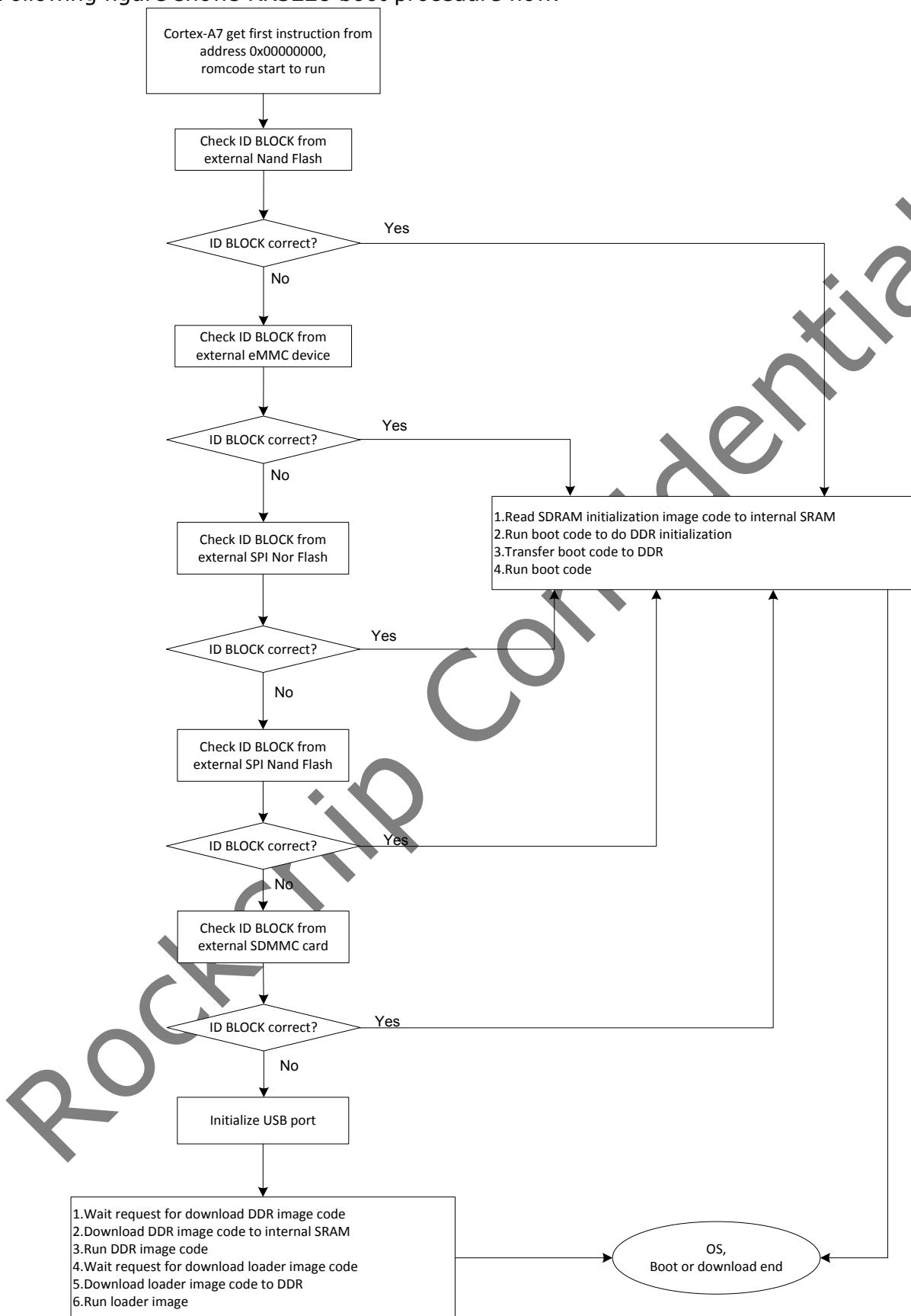


Fig. 2-2 RK3228 boot procedure flow

## 2.3 System Interrupt connection

RK3228 provides an general interrupt controller(GIC) for CPU, which has 128 SPI (shared peripheral interrupts) interrupt sources and 3 PPI(Private peripheral interrupt) interrupt source and separately generates one nIRQ and one nFIQ to CPU. The triggered type for each interrupts is high level sensitive, not programmable. The detailed interrupt sources connection is in the following table. For detailed GIC setting, please refer to Chapter 9.

Table 2-1 RK3228 Interrupt connection list

IRQ Type	IRQ ID	Source(spi)	Polarity
SPI	32	bus_dmac_irq	High level
	33	bus_dmac_irq_abort	High level
	34	ddrctl_intr	High level
	35	ddrmon_int	High level
	36	gpu_irqpp	High level
	37	gpu_irqmmu	High level
	38	gpu_irqgp	High level
	39	rkvdec_dec_irq	High level
	40	rkvdec_mmu_irq	High level
	41	vpu_dec_int	High level
	42	vpu_mmu_int	High level
	43	vpu_enc_int	High level
	44	sdmmc_int	High level
	45	sdio_int	High level
	46	emmc_int	High level
	47	nandc_int	High level
	48	host0_ehci_int	High level
	49	host0_ohci_int	High level
	50	host0_arb_int	High level
	51	host1_ehci_int	High level
	52	host1_ohci_int	High level
	53	host1_arb_int	High level
	54	host2_arb_int	High level
	55	otg_int	High level
	56	gmac_int	High level
	57	gmac_pmt_int	High level
	58	i2s_hdmi_intr	High level
	59	i2s_8ch_intr	High level
	60	i2s_2ch_intr	High level
	61	spdif_8ch_intr	High level
	62	crypto_int	High level
	63	iep_intr	High level
	64	vop_intr	High level
	65	rga_intr	High level
	66	hdcp_intr	High level
	67	hdmi_intr	High level
	68	rki2c0_int	High level

IRQ Type	IRQ ID	Source(spi)	Polarity
	69	rki2c1_int	High level
	70	rki2c2_int	High level
	71	rki2c3_int	High level
	72	wdt_intr	High level
	73	stimer_intr0	High level
	74	stimer_intr1	High level
	75	timer_intr0	High level
	76	timer_intr1	High level
	77	timer_intr2	High level
	78	timer_intr3	High level
	79	timer_intr4	High level
	80	timer_intr5	High level
	81	spi0_intr	High level
	82	rkpwm_int	High level
	83	gpio0_intr	High level
	84	gpio1_intr	High level
	85	gpio2_intr	High level
	86	gpio3_intr	High level
	87	uart0_intr	High level
	88	uart1_intr	High level
	89	uart2_intr	High level
	90	tsadc_int	High level
	91	otg0_bvalid_irq	High level
	92	otg0_id_irq	High level
	93	otg0_linestate_irq	High level
	94	host0_linestate_irq	High level
	95	sd_detectn_irq	High level
	96	Reserved	High level
	97	sdmmc_detect_n	High level
	98	host2_ehci_int	High level
	99	host2_ohci_int	High level
	100	host1_linestate_irq	High level
	101	host2_linestate_irq	High level
	102	macphy_int	High level
	103	hdmi_intr_wakeup	High level
	104	tsp_int	High level
	105	sim_int	High level
	106	Reserved	High level
	107	Reserved	High level
	108	core_npmuirq0	High level
	109	core_npmuirq1	High level
	110	core_npmuirq2	High level
	111	core_npmuirq3	High level
	112	axierrirq	High level

## 2.4 System DMA hardware request connection

RK3228 provides one DMA controller inside the system. The trigger type for each of them is high level, not programmable. For detailed descriptions of DMAC, please refer to Chapter 8.

Table 2-2 RK3228 DMAC Hardware request connection list

Req Number	Source	Polarity
0	I2S2_2ch tx	High level
1	I2S2_2ch rx	High level
2	Uart0 tx	High level
3	Uart0 rx	High level
4	Uart1 tx	High level
5	Uart1 rx	High level
6	Uart2 tx	High level
7	Uart2 rx	High level
8	SPI tx	High level
9	SPI rx	High level
10	SPDIF	High level
11	I2S0_8ch tx	High level
12	I2S0_8ch rx	High level
13	pwm_tx	High level
14	I2S1_8ch_tx	High level
15	I2S1_8ch_rx	High level

## Chapter 3 Clock & Reset Unit (CRU)

### 3.1 Overview

The CRU is an APB slave module that is designed for generating all of the internal and system clocks, resets of chip. CRU generates system clocks from PLL output clock or external clock source, and generates system reset from external power-on-reset, watchdog timer reset or software reset.

CRU supports the following features:

- Compliance to the AMBA APB interface
- Embedded four PLLs
- Flexible selection of clock source
- Supports the respective gating of all clocks
- Supports the respective software reset of all modules

### 3.2 Block Diagram

The CRU comprises with:

- PLL
- Register configuration unit
- Clock generate unit
- Reset generate unit

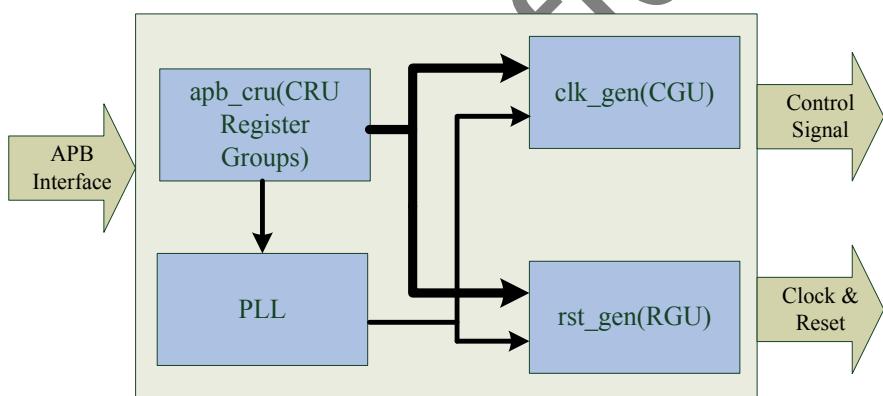


Fig. 3-1 CRU Architecture

### 3.3 System Reset Solution

The following diagram shows reset architecture.

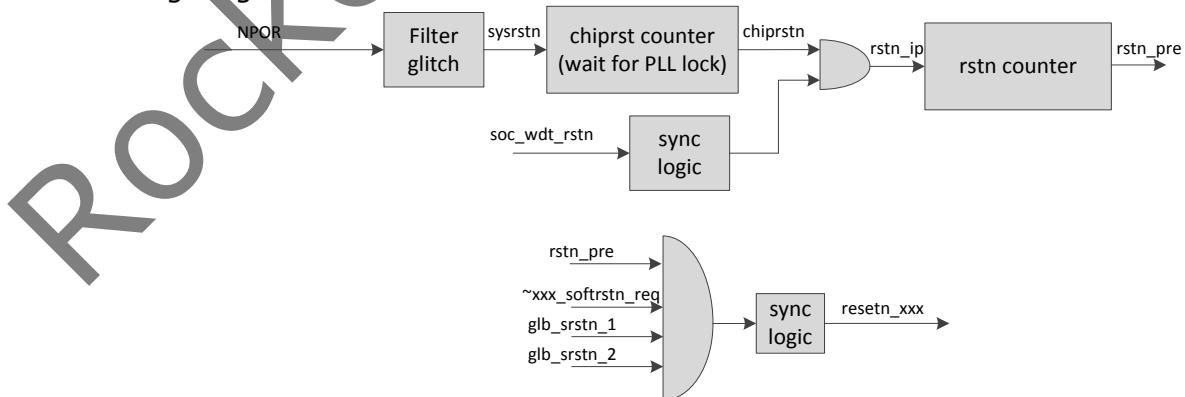


Fig. 3-2 Reset Architecture Diagram

Reset source of each reset signal includes hardware reset(NPOR), SoC watch dog reset(soc\_wdt\_rstn), software reset request(~xxx\_softrstn\_req), global software reset1(glb\_srstn\_1), global software reset2(glb\_srstn\_2).

The 'xxx' of resetn\_xxx and xxx\_softrstn\_req is the module name.

soc\_wdt\_rstn is the reset from watch-dog IP in the SoC.

glb\_srstn\_1 and glb\_srstn\_2 are the global software reset by programming CRU register. When writing register CRU\_GLB\_SRST\_FST\_VALUE as 0xfdb9, glb\_srstn\_1 will be asserted, and when writing register CRU\_GLB\_SRST SND\_VALUE as 0xecaa, glb\_srstn\_2 will be asserted. The two software resets will be self-cleared by hardware. glb\_srstn\_1 will reset the all logic, and glb\_srstn\_2 will reset the all logic except GRF and all GPIOs.

## 3.4 Function Description

There are four PLLs in the chip: ARM PLL, DDR PLL, CODEC PLL and GENERAL PLL, and it supports only one crystal oscillator: 24MHz. Each PLL can only receive 24MHz oscillator. Four PLLs all can be set to slow mode or deep slow mode, directly output selectable 24MHz. When power on or changing PLL setting, we must force PLL into slow mode to ensure output stable clock.

To maximize the flexibility, some of clocks can select divider source from four PLLs. To provide some specific frequency, another solution is integrated: fractional divider. In order to guarantee the performance for divided clock, there is some usage limit, we can only get low frequency and divider factor must be larger than 20.

All clocks can be software gated and all resets can be software generated.

## 3.5 PLL Introduction

### 3.5.1 Overview

The chip uses 3.2GHz PLL for all the PLLs. The 3.2GHz PLL is a general purpose, high-performance PLL-based clock generator. The PLL is a multi-function, general purpose frequency synthesizer. Ultra-wide input and output ranges along with best-in-class jitter performance allow the PLL to be used for almost any clocking application. With excellent supply noise immunity, the PLL is ideal for use in noisy mixed signal SoC environments. By combining ultra-low jitter output clocks into a low power, low area, widely programmable design, we can greatly simplify an SoC by enabling a single macro to be used for all clocking applications in the system.

3.2GHz PLL supports the following features:

- Input frequency range: 1MHz to 800MHz (Integer Mode) and 10MHz to 800MHz (Fractional Mode)
- Output Frequency Range: 16MHz to 3.2GHz
- 24 bit fractional accuracy, and fractional mode jitter performance to nearly match integer mode performance.
- 4:1 VCO frequency range allows PLL to be optimized for minimum jitter or minimum power.
- Isolated analog supply (1.8V) allows for excellent supply rejection in noisy SoC applications.
- Lock Detect Signal indicates when frequency lock has been achieved.

### 3.5.2 Block diagram

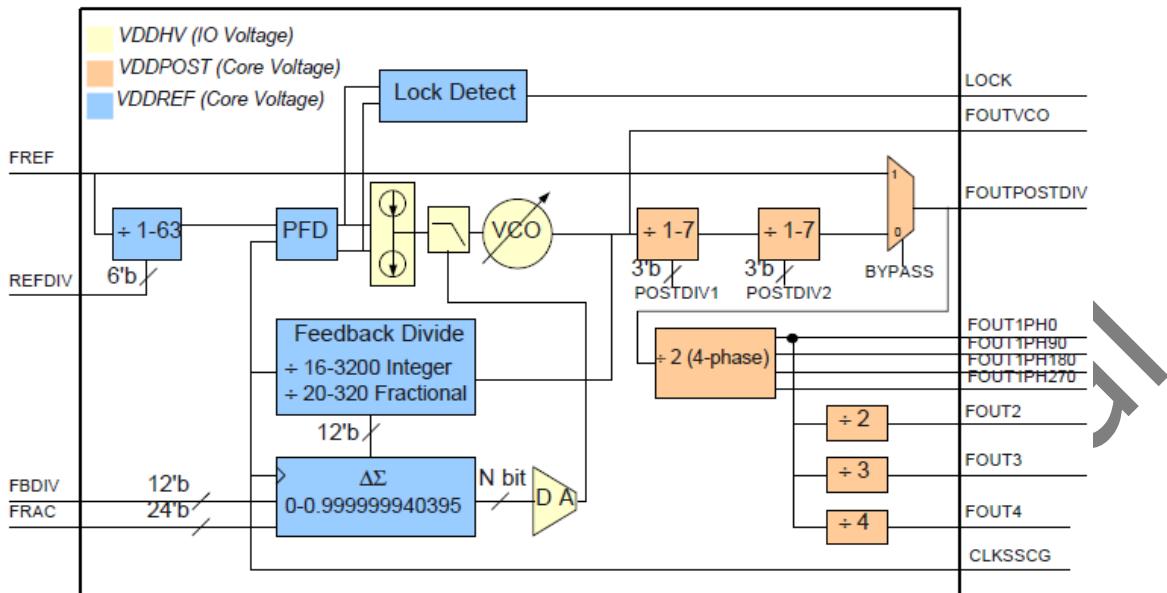


Fig. 3-3 PLL Block Diagram

#### How to calculate the PLL

The Fractional PLL output frequency can be calculated using some simple formulas. These formulas also embedded within the Fractional PLL Verilog model:

If DSMPD = 1 (DSM is disabled, "integer mode")

$$\text{FOUTVCO} = \text{FREF} / \text{REFDIV} * \text{FB DIV}$$

$$\text{FOUTPOSTDIV} = \text{FOUTVCO} / \text{POSTDIV1} / \text{POSTDIV2}$$

If DSMPD = 0 (DSM is enabled, "fractional mode")

$$\text{FOUTVCO} = \text{FREF} / \text{REFDIV} * (\text{FB DIV} + \text{FRAC} / 224)$$

$$\text{FOUTPOSTDIV} = \text{FOUTVCO} / \text{POSTDIV1} / \text{POSTDIV2}$$

Where:

FOUTVCO = Fractional PLL non-divided output frequency

FOUTPOSTDIV = Fractional PLL divided output frequency (output of second post divider)

FREF = Fractional PLL input reference frequency

REFDIV = Fractional PLL input reference clock divider

VCO = Frequency of internal VCO

FB DIV = Integer value programmed into feedback divide

FRAC = Fractional value programmed into DSM

#### Changing the PLL Programming

In most cases the PLL programming can be changed on-the-fly and the PLL will simply slew to the new frequency. However, certain changes have the potential to cause glitches on the PLL output clocks. These changes include:

- Switching into or out of BYPASS mode may cause a glitch on FOUTPOSTDIV
- Changing POSTDIV1 or POSTDIV2 may cause a short pulse with width equal to as little as one VCO period on FOUTPOSTDIV
- Changing POSTDIV could cause a shortened pulse on FOUT1PH\* or FOUT2/3/4
- Asserting PD or FOUTPOSTDIVPD may cause a glitch on FOUTPOSTDIV

## 3.6 Register Description

This section describes the control/status registers of the design.

### 3.6.1 Registers Summary

Name	Offset	Size	Reset Value	Description
CRU_APPL_CON0	0x0000	W	0x0000215e	ARM PLL control register0
CRU_APPL_CON1	0x0004	W	0x00001046	ARM PLL control register1

Name	Offset	Size	Reset Value	Description
CRU_APPL_CON2	0x0008	W	0x00000001	ARM PLL control register2
CRU_DPLL_CON0	0x000c	W	0x000010c8	DDR PLL control register0
CRU_DPLL_CON1	0x0010	W	0x00001043	DDR PLL control register1
CRU_DPLL_CON2	0x0014	W	0x00000001	DDR PLL control register2
CRU_CPLL_CON0	0x0018	W	0x000020fa	Codec PLL control register0
CRU_CPLL_CON1	0x001c	W	0x00001046	Codec PLL control register1
CRU_CPLL_CON2	0x0020	W	0x00000001	CODEC PLL control register2
CRU_GPLL_CON0	0x0024	W	0x00002064	General PLL control register0
CRU_GPLL_CON1	0x0028	W	0x00001042	General PLL control register1
CRU_GPLL_CON2	0x002c	W	0x00000001	GENERAL PLL control register2
CRU_MODE_CON	0x0040	W	0x00000000	System work mode control register
CRU_CLKSEL0_CON	0x0044	W	0x00000100	Internal clock select and divide register0
CRU_CLKSEL1_CON	0x0048	W	0x00003113	Internal clock select and divide register1
CRU_CLKSEL2_CON	0x004c	W	0x00000703	Internal clock select and divide register2
CRU_CLKSEL3_CON	0x0050	W	0x0000001f	Internal clock select and divide register3
CRU_CLKSEL4_CON	0x0054	W	0x00000003	Internal clock select and divide register4
CRU_CLKSEL5_CON	0x0058	W	0x00000303	Internal clock select and divide register5
CRU_CLKSEL6_CON	0x005c	W	0x0000021f	Internal clock select and divide register6
CRU_CLKSEL7_CON	0x0060	W	0x0bb8ea60	Internal clock select and divide register7
CRU_CLKSEL8_CON	0x0064	W	0x0bb8ea60	Internal clock select and divide register8
CRU_CLKSEL9_CON	0x0068	W	0x0000001f	Internal clock select and divide register9
CRU_CLKSEL10_CON	0x006c	W	0x00003100	Internal clock select and divide register10
CRU_CLKSEL11_CON	0x0070	W	0x00000017	Internal clock select and divide register11
CRU_CLKSEL12_CON	0x0074	W	0x00001717	Internal clock select and divide register12
CRU_CLKSEL13_CON	0x0078	W	0x0000021f	Internal clock select and divide register13
CRU_CLKSEL14_CON	0x007c	W	0x0000021f	Internal clock select and divide register14
CRU_CLKSEL15_CON	0x0080	W	0x0000021f	Internal clock select and divide register15

Name	Offset	Size	Reset Value	Description
CRU_CLKSEL16_CON	0x0084	W	0x0000001f	Internal clock select and divide register16
CRU_CLKSEL17_CON	0x0088	W	0x0bb8ea60	Internal clock select and divide register17
CRU_CLKSEL18_CON	0x008c	W	0x0bb8ea60	Internal clock select and divide register18
CRU_CLKSEL19_CON	0x0090	W	0x0bb8ea60	Internal clock select and divide register19
CRU_CLKSEL20_CON	0x0094	W	0x0bb8ea60	Internal clock select and divide register20
CRU_CLKSEL21_CON	0x0098	W	0x0000c2dc	Internal clock select and divide register21
CRU_CLKSEL22_CON	0x009c	W	0x00000f01	Internal clock select and divide register21
CRU_CLKSEL23_CON	0x00a0	W	0x0000412f	Internal clock select and divide register23
CRU_CLKSEL24_CON	0x00a4	W	0x000003c3	Internal clock select and divide register24
CRU_CLKSEL25_CON	0x00a8	W	0x0000011f	Internal clock select and divide register25
CRU_CLKSEL26_CON	0x00ac	W	0x00000000	Internal clock select and divide register26
CRU_CLKSEL27_CON	0x00b0	W	0x00000100	Internal clock select and divide register27
CRU_CLKSEL28_CON	0x00b4	W	0x00004141	Internal clock select and divide register28
CRU_CLKSEL29_CON	0x00b8	W	0x00000000	Internal clock select and divide register29
CRU_CLKSEL30_CON	0x00bc	W	0x0bb8ea60	Internal clock select and divide register30
CRU_CLKSEL31_CON	0x00c0	W	0x00002121	Internal clock select and divide register31
CRU_CLKSEL32_CON	0x00c4	W	0x00000021	Internal clock select and divide register32
CRU_CLKSEL33_CON	0x00c8	W	0x00002121	Internal clock select and divide register33
CRU_CLKSEL34_CON	0x00cc	W	0x00002121	Internal clock select and divide register34
CRU_CLKGATE0_CON	0x00d0	W	0x00000000	Internal clock gating control register0
CRU_CLKGATE1_CON	0x00d4	W	0x00000000	Internal clock gating control register1
CRU_CLKGATE2_CON	0x00d8	W	0x00000000	Internal clock gating control register2

Name	Offset	Size	Reset Value	Description
CRU_CLKGATE3_CON	0x00dc	W	0x00000000	Internal clock gating control register3
CRU_CLKGATE4_CON	0x00e0	W	0x00000000	Internal clock gating control register0
CRU_CLKGATE5_CON	0x00e4	W	0x00000000	Internal clock gating control register5
CRU_CLKGATE6_CON	0x00e8	W	0x00000000	Internal clock gating control register6
CRU_CLKGATE7_CON	0x00ec	W	0x00000000	Internal clock gating control register7
CRU_CLKGATE8_CON	0x00f0	W	0x00000000	Internal clock gating control register8
CRU_CLKGATE9_CON	0x00f4	W	0x00000000	Internal clock gating control register9
CRU_CLKGATE10_CON	0x00f8	W	0x00000000	Internal clock gating control register10
CRU_CLKGATE11_CON	0x00fc	W	0x00000000	Internal clock gating control register11
CRU_CLKGATE12_CON	0x0100	W	0x00000000	Internal clock gating control register12
CRU_CLKGATE13_CON	0x0104	W	0x00000000	Internal clock gating control register13
CRU_CLKGATE14_CON	0x0108	W	0x00000000	Internal clock gating control register14
CRU_CLKGATE15_CON	0x010c	W	0x00000000	Internal clock gating control register15
CRU_SOFRST0_CON	0x0110	W	0x00000000	Internal software reset control register0
CRU_SOFRST1_CON	0x0114	W	0x00000000	Internal software reset control register1
CRU_SOFRST2_CON	0x0118	W	0x00000000	Internal software reset control register2
CRU_SOFRST3_CON	0x011c	W	0x00000000	Internal software reset control register3
CRU_SOFRST4_CON	0x0120	W	0x00000000	Internal software reset control register4
CRU_SOFRST5_CON	0x0124	W	0x00000000	Internal software reset control register5
CRU_SOFRST6_CON	0x0128	W	0x00000000	Internal software reset control register6
CRU_SOFRST7_CON	0x012c	W	0x00000000	Internal software reset control register7
CRU_SOFRST8_CON	0x0130	W	0x00000000	Internal software reset control register8
CRU_MISC_CON	0x0134	W	0x0000a000	SCU control register

Name	Offset	Size	Reset Value	Description
CRU_GLB_CNT_TH	0x0140	W	0x3a980064	global reset wait counter threshold
CRU_GLB_RST_ST	0x0150	W	0x00000000	global reset status
CRU_SDMMC_CON0	0x01c0	W	0x00000004	sdmmc control0
CRU_SDMMC_CON1	0x01c4	W	0x00000000	sdmmc control1
CRU_SDIO_CON0	0x01c8	W	0x00000004	sdio0 control0
CRU_SDIO_CON1	0x01cc	W	0x00000000	sdio0 control1
CRU_EMMC_CON0	0x01d8	W	0x00000004	emmc control0
CRU_EMMC_CON1	0x01dc	W	0x00000000	emmc control1
CRU_GLB_SRST_FST_VAL	0x01f0	W	0x00000000	The first global software reset config value
CRU_GLB_SRST_SND_VAL	0x01f4	W	0x00000000	The second global software reset config value
CRU_PLL_MASK_CON	0x01f8	W	0x00005a5a	PLL config mask control

Notes: Size: **B**- Byte (8 bits) access, **HW**- Half WORD (16 bits) access, **W**-WORD (32 bits) access

### 3.6.2 Detail Register Description

#### CRU\_APPL\_CON0

Address: Operational Base + offset (0x0000)

ARM PLL control register0

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	bit_write_mask control corresponding (bit_write_mask - 16) configuration bit 0: mask 1: unmask
15	RW	0x0	bp pll bypass
14:12	RW	0x2	postdiv1 PLL factor postdiv1
11:0	RW	0x15e	fbdv PLL factor fbdv

#### CRU\_APPL\_CON1

Address: Operational Base + offset (0x0004)

ARM PLL control register1

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	bit_write_mask control corresponding (bit_write_mask - 16) configuration bit 0: mask 1: unmask
15	RW	0x0	rstmode PLL Reset select 0 : internal reset 1 : software reset

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
14	RW	0x0	rst PLL Software Reset 0 : normal 1 : reset
13	RW	0x0	pd PLL software power down, active high
12	RW	0x1	dsmpd when 1, PLL work at integer mode when 0, PLL work at frac mode
11	RO	0x0	reserved
10	RW	0x0	lock PLL lock status
9	RO	0x0	reserved
8:6	RW	0x1	postdiv2 PLL factor postdiv2
5:0	RW	0x06	refdiv PLL factor refdiv

**CRU\_APLL\_CON2**

Address: Operational Base + offset (0x0008)

ARM PLL control register2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:28	RO	0x0	reserved
27	RW	0x0	fout4phasepd 4 phase clock power down, active high
26	RW	0x0	foutvcopd buffered VCO clock power down, active high
25	RW	0x0	foutpostdivpd post divide power down, active high
24	RW	0x0	dacpd PLL cancellation DAC power down, active high
23:0	RW	0x000001	frac apll_frac[23:0]

**CRU\_DPLL\_CON0**

Address: Operational Base + offset (0x000c)

DDR PLL control register0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	bit_write_mask control corresponding (bit_write_mask - 16) configuration bit 0: mask 1: unmask
15	RW	0x0	bp pll bypass

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
14:12	RW	0x1	postdiv1 PLL factor postdiv1
11:0	RW	0x0c8	fbdv PLL factor fbdv

**CRU\_DPLL\_CON1**

Address: Operational Base + offset (0x0010)

DDR PLL control register1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	bit_write_mask control corresponding (bit_write_mask - 16) configuration bit 0: mask 1: unmask
15	RW	0x0	rstmode PLL Reset select 0 : internal reset 1 : software reset
14	RW	0x0	rst PLL Software Reset 0 : normal 1 : reset
13	RW	0x0	pd PLL software power down, active high
12	RW	0x1	dsmpd when 1, PLL work at interger mode when 0, PLL work at frac mode
11	RO	0x0	reserved
10	RW	0x0	lock PLL lock status
9	RO	0x0	reserved
8:6	RW	0x1	postdiv2 PLL factor postdiv2
5:0	RW	0x03	refdiv PLL factor refdiv

**CRU\_DPLL\_CON2**

Address: Operational Base + offset (0x0014)

DDR PLL control register2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:28	RO	0x0	reserved
27	RW	0x0	fout4phasepd 4 phase clock power down, active high
26	RW	0x0	foutvcopd buffered VCO clock power down, active high

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
25	RW	0x0	foutpostdivpd post divide power down, active high
24	RW	0x0	dacpd PLL cancellation DAC power down, active high
23:0	RW	0x000001	frac apll_frac[23:0]

**CRU\_CPLL\_CON0**

Address: Operational Base + offset (0x0018)

Codec PLL control register0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	bit_write_mask control corresponding (bit_write_mask - 16) configuration bit 0: mask 1: unmask
15	RW	0x0	bp pll bypass
14:12	RW	0x2	postdiv1 PLL factor postdiv1
11:0	RW	0x0fa	fbdv PLL factor fbdv

**CRU\_CPLL\_CON1**

Address: Operational Base + offset (0x001c)

Codec PLL control register1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	bit_write_mask control corresponding (bit_write_mask - 16) configuration bit 0: mask 1: unmask
15	RW	0x0	rstmode PLL Reset select 0 : internal reset 1 : software reset
14	RW	0x0	rst PLL Software Reset 0 : normal 1 : reset
13	RW	0x0	pd PLL software power down, active high
12	RW	0x1	dsmpd when 1, PLL work at interger mode when 0, PLL work at frac mode
11	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
10	RW	0x0	lock PLL lock status
9	RO	0x0	reserved
8:6	RW	0x1	postdiv2 PLL factor postdiv2
5:0	RW	0x06	refdiv PLL factor refdiv

**CRU\_CPLL\_CON2**

Address: Operational Base + offset (0x0020)

CODEC PLL control register2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:28	RO	0x0	reserved
27	RW	0x0	fout4phasepd 4 phase clock power down, active high
26	RW	0x0	foutvcopd buffered VCO clock power down, active high
25	RW	0x0	foutpostdivpd post divide power down, active high
24	RW	0x0	dacpd PLL cancellation DAC power down, active high
23:0	RW	0x000001	frac apll_frac[23:0]

**CRU\_GPLL\_CON0**

Address: Operational Base + offset (0x0024)

General PLL control register0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	bit_write_mask control corresponding (bit_write_mask - 16) configuration bit 0: mask 1: unmask
15	RW	0x0	bp pll bypass
14:12	RW	0x2	postdiv1 PLL factor postdiv1
11:0	RW	0x064	fbdiv PLL factor fbdiv

**CRU\_GPLL\_CON1**

Address: Operational Base + offset (0x0028)

General PLL control register1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	bit_write_mask control corresponding (bit_write_mask - 16) configuration bit 0: mask 1: unmask
15	RW	0x0	rstmode PLL Reset select 0 : internal reset 1 : software reset
14	RW	0x0	rst PLL Software Reset 0 : normal 1 : reset
13	RW	0x0	pd PLL software power down, active high
12	RW	0x1	dsmpd when 1, PLL work at integer mode when 0, PLL work at frac mode
11	RO	0x0	reserved
10	RW	0x0	lock PLL lock status
9	RO	0x0	reserved
8:6	RW	0x1	postdiv2 PLL factor postdiv2
5:0	RW	0x02	refdiv PLL factor refdiv

**CRU\_GPLL\_CON2**

Address: Operational Base + offset (0x002c)

GENERAL PLL control register2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:28	RO	0x0	reserved
27	RW	0x0	fout4phasepd 4 phase clock power down, active high
26	RW	0x0	foutvcopd buffered VCO clock power down, active high
25	RW	0x0	foutpostdivpd post divide power down, active high
24	RW	0x0	dacpd PLL cancellation DAC power down, active high
23:0	RW	0x000001	frac apll_frac[23:0]

**CRU\_MODE\_CON**

Address: Operational Base + offset (0x0040)

System work mode control register

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:13	RO	0x0	reserved
12	RW	0x0	gpll_work_mode GENERAL PLL work mode select 1'b0: Slow mode, clock from external 24MHz OSC (default) 1'b1: Normal mode, clock from PLL output
11:9	RO	0x0	reserved
8	RW	0x0	cpll_work_mode CODEC PLL work mode select 1'b0: Slow mode, clock from external 24MHz OSC (default) 1'b1: Normal mode, clock from PLL output
7:5	RO	0x0	reserved
4	RW	0x0	dpll_work_mode DDR PLL work mode select 1'b0: Slow mode, clock from external 24MHz OSC (default) 1'b1: Normal mode, clock from PLL output
3:1	RO	0x0	reserved
0	RW	0x0	apll_work_mode ARM PLL work mode select 1'b0: Slow mode, clock from external 24MHz OSC (default) 1'b1: Normal mode, clock from PLL output

**CRU\_CLKSEL0\_CON**

Address: Operational Base + offset (0x0044)

Internal clock select and divide register0

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RO	0x0	reserved
14:13	RW	0x0	bus_aclk_pll_sel pd_bus aclk pll source selection 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmiphy pll clock
12:8	RW	0x01	bus_aclk_div_con PD bus AXI clock divider frequency clk=clk_src/(div_con+1)
7:6	RW	0x0	core_clk_pll_sel core clock pll source selection 2'b00: select arm pll clock 2'b01: select general pll clock 2'b10: select ddr pll clock

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
5	RO	0x0	reserved
4:0	RW	0x00	a7_core_div_con Control A7 core clock divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL1\_CON**

Address: Operational Base + offset (0x0048)

Internal clock select and divide register1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RO	0x0	reserved
14:12	RW	0x3	bus_pclk_div_con Control pd_bus APB clock divider frequency $clk=clk\_src/(div\_con+1)$
11:10	RO	0x0	reserved
9:8	RW	0x1	bus_hclk_div_con Control pd_bus AHB clock divider frequency $clk=clk\_src/(div\_con+1)$
7	RO	0x0	reserved
6:4	RW	0x1	core_aclk_div_con Control core axi clock divider frequency $clk=clk\_src/(div\_con+1)$
3:0	RW	0x3	clk_core_peri_div_con CORE peri divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL2\_CON**

Address: Operational Base + offset (0x004c)

Internal clock select and divide register2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RO	0x0	reserved
14	RW	0x0	nandc_clk_pll_sel nandc pll source selection. 1'b0: codec_pll_clk 1'b1: general_pll_clk
13	RO	0x0	reserved
12:8	RW	0x07	nandc_div_con Control NANDC clock divider frequency. $clk=clk\_src/(div\_con+1)$
7:5	RO	0x0	reserved

Bit	Attr	Reset Value	Description
4:0	RW	0x03	hclk_vio_div_con Control pd_vio AHB bus clk frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL3\_CON**

Address: Operational Base + offset (0x0050)

Internal clock select and divide register3

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	i2s1_pll_sel Control I2S1 PLL source selection 1'b0: select codec pll clock 1'b1: select general pll clock
14:13	RO	0x0	reserved
12	RW	0x0	i2s1_clkout_sel I2S1 output clock selection 1'b0: select cru generated clock 1'b1: select io input clock
11:10	RO	0x0	reserved
9:8	RW	0x0	i2s1_clk_sel Control I2S1 clock work frequency selection 2'b00: select divider ouput from pll divider 2'b01: select divider ouput from fraction divider 2'b10: select io i2s input clock 2'b11: select 12MHz from osc input
7	RO	0x0	reserved
6:0	RW	0x1f	i2s1_pll_div_con Control I2S1 PLL output divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL4\_CON**

Address: Operational Base + offset (0x0054)

Internal clock select and divide register4

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:13	RO	0x0	reserved
12:8	RW	0x00	clk_24m_div_con Control clk_24m divider frequency $clk=clk\_src/(div\_con+1)$
7:5	RO	0x0	reserved

Bit	Attr	Reset Value	Description
4:0	RW	0x03	clk_monitor_div_con Control clk_monitor divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL5\_CON**

Address: Operational Base + offset (0x0058)

Internal clock select and divide register5

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	gmac_out_pll_sel Control GMAC out clock PLL source selection 1'b0: select codec pll clock 1'b1: select general pll clock
14:13	RO	0x0	reserved
12:8	RW	0x03	gmac_out_div_con Control clk_gmac_out divider frequency $clk=clk\_src/(div\_con+1)$
7	RW	0x0	mac_pll_sel Control MAC clock PLL source selection 1'b0: select codec pll clock 1'b1: select general pll clock
6	RO	0x0	reserved
5	RW	0x0	rmii_extclk_sel Control rmii_extclk frequency selection 1'b0: clk_mac_div_out 1'b1: rmii_clkin
4:0	RW	0x03	clk_mac_div_con Control clk_mac divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL6\_CON**

Address: Operational Base + offset (0x005c)

Internal clock select and divide register6

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	spdif_pll_sel Control SPDIF PLL source selection 1'b0: select codec pll clock 1'b1: select general pll clock
14:10	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
9:8	RW	0x2	spdif_clk_sel Control SPDIF clock work frequency selection 2'b00: select divider output from pll divider 2'b01: select divider output from fraction divider 2'b10: select 12MHz from osc input
7	RO	0x0	reserved
6:0	RW	0x1f	spdif_pll_div_con Control SPDIF PLL output divider frequency clk=clk_src/(div_con+1)

**CRU\_CLKSEL7\_CON**

Address: Operational Base + offset (0x0060)

Internal clock select and divide register7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x0bb8ea60	i2s1_frac_factor Control I2S1 fraction divider frequency High 16-bit for numerator Low 16-bit for denominator

**CRU\_CLKSEL8\_CON**

Address: Operational Base + offset (0x0064)

Internal clock select and divide register8

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x0bb8ea60	i2s0_frac_factor Control I2S0 fraction divider frequency High 16-bit for numerator Low 16-bit for denominator

**CRU\_CLKSEL9\_CON**

Address: Operational Base + offset (0x0068)

Internal clock select and divide register9

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	i2s0_pll_sel Control I2S0_8ch PLL source selection 1'b0: select codec pll clock 1'b1: select general pll clock
14:10	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
9:8	RW	0x0	i2s0_clk_sel Control I2S0_8ch clock work frequency selection 2'b00: select divider output from pll divider 2'b01: select divider output from fraction divider 2'b10: select io i2s input clock 2'b11: select 12MHz from osc input
7	RO	0x0	reserved
6:0	RW	0x1f	i2s0_pll_div_con Control I2S0_8ch PLL output divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL10\_CON**

Address: Operational Base + offset (0x006c)

Internal clock select and divide register10

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RO	0x0	reserved
14:12	RW	0x3	peri_pclk_div_con Control the divider between aclk_periph and pclk_periph. 3'b000: aclk_periph:pclk_periph = 1:1 3'b001: aclk_periph:pclk_periph = 2:1 3'b011: aclk_periph:pclk_periph = 4:1 3'b111: aclk_periph:pclk_periph = 8:1
11:10	RW	0x0	peri_pll_sel Control peripheral clock PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmi phy pll clock
9:8	RW	0x1	peri_hclk_div_con Control the divider ratio between aclk_periph and hclk_periph. 2'b00: aclk_periph:hclk_periph = 1:1 2'b01: aclk_periph:hclk_periph = 2:1 2'b11: aclk_periph:hclk_periph = 4:1
7:5	RO	0x0	reserved
4:0	RW	0x00	peri_aclk_div_con Control peripheral AXI clock divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL11\_CON**

Address: Operational Base + offset (0x0070)

Internal clock select and divide register11

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
------------	-------------	--------------------	--------------------

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:14	RO	0x0	reserved
13:12	RW	0x0	emmc_pll_sel Control emmc clock PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select 24M 2'b11: select usb phy 480M clock
11:10	RW	0x0	sdio_pll_sel Control sdio clock PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select 24M 2'b11: select usb phy 480M clock
9:8	RW	0x0	mmc0_pll_sel Control mmc clock PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select 24M 2'b11: select usb phy 480M clock
7:0	RW	0x17	mmc0_div_con Control SDMMC0 divider frequency. $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL12\_CON**

Address: Operational Base + offset (0x0074)

Internal clock select and divide register12

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:8	RW	0x17	emmc_div_con Control EMMC divider frequency. $clk=clk\_src/(div\_con+1)$
7:0	RW	0x17	sdio_div_con Control SDIO divider frequency. $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL13\_CON**

Address: Operational Base + offset (0x0078)

Internal clock select and divide register13

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:14	RO	0x0	reserved
13:12	RW	0x0	uart0_pll_sel Control UART0 clock PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select USBPHY 480M clock
11:10	RO	0x0	reserved
9:8	RW	0x2	uart0_clk_sel Control UART0 clock work frequency selection. 2'b00: select divider output from pll divider 2'b01: select divider output from fraction divider 2'b10: select 24MHz from osc input
7	RO	0x0	reserved
6:0	RW	0x1f	uart0_div_con Control UART0 divider frequency. $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL14\_CON**

Address: Operational Base + offset (0x007c)

Internal clock select and divide register14

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:14	RO	0x0	reserved
13:12	RW	0x0	uart1_pll_sel Control UART1 clock PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select USBPHY 480M clock
11:10	RO	0x0	reserved
9:8	RW	0x2	uart1_clk_sel Control UART1 clock work frequency selection. 2'b00: select divider output from pll divider 2'b01: select divider output from fraction divider 2'b10: select 24MHz from osc input
7	RO	0x0	reserved
6:0	RW	0x1f	uart1_div_con Control UART1 divider frequency. $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL15\_CON**

## RK3228 TRM

Address: Operational Base + offset (0x0080)

Internal clock select and divide register15

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:14	RO	0x0	reserved
13:12	RW	0x0	uart2_pll_sel Control UART2 clock PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select USBPHY 480M clock
11:10	RO	0x0	reserved
9:8	RW	0x2	uart2_clk_sel Control UART2 clock work frequency selection. 2'b00: select divider output from pll divider 2'b01: select divider output from fraction divider 2'b10: select 24MHz from osc input
7	RO	0x0	reserved
6:0	RW	0x1f	uart2_div_con Control UART2 divider frequency. $clk=clk\_src/(div\_con+1)$

## CRU\_CLKSEL16\_CON

Address: Operational Base + offset (0x0084)

Internal clock select and divide register16

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	i2s2_pll_sel Control I2S2 PLL source selection. 1'b0: select codec pll clock 1'b1: select general pll clock
14:10	RO	0x0	reserved
9:8	RW	0x0	i2s2_clk_sel Control I2S2 clock work frequency selection. 2'b00: select divider output from pll divider 2'b01: select divider output from fraction divider 2'b11: select 12MHz from osc input
7	RO	0x0	reserved
6:0	RW	0x1f	i2s2_pll_div_con Control I2S2 PLL output divider frequency. $clk=clk\_src/(div\_con+1)$

## CRU\_CLKSEL17\_CON

Address: Operational Base + offset (0x0088)

Internal clock select and divide register17

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0xbb8ea60	uart0_frac_factor Control UART0 fraction divider frequency. High 16-bit for numerator Low 16-bit for denominator

**CRU\_CLKSEL18\_CON**

Address: Operational Base + offset (0x008c)

Internal clock select and divide register18

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0xbb8ea60	uart1_frac_factor Control UART1 fraction divider frequency. High 16-bit for numerator Low 16-bit for denominator

**CRU\_CLKSEL19\_CON**

Address: Operational Base + offset (0x0090)

Internal clock select and divide register19

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0xbb8ea60	uart2_frac_factor Control UART2 fraction divider frequency. High 16-bit for numerator Low 16-bit for denominator

**CRU\_CLKSEL20\_CON**

Address: Operational Base + offset (0x0094)

Internal clock select and divide register20

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0xbb8ea60	spdif_frac_factor Control SPDIF fraction divider frequency. High 16-bit for numerator Low 16-bit for denominator

**CRU\_CLKSEL21\_CON**

Address: Operational Base + offset (0x0098)

Internal clock select and divide register21

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:14	RW	0x3	cec_clk_pll_sel Control CEC PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select io input 24M clock
13:0	RW	0x02dc	cec_clk_div_con Control CEC PLL output divider frequency. $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL22\_CON**

Address: Operational Base + offset (0x009c)

Internal clock select and divide register21

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	tsp_pll_sel Control TSP clock work frequency selection. 1'b0:select codec pll clock 1'b1:select general pll clock
14:13	RO	0x0	reserved
12:8	RW	0x0f	tsp_clk_div_con Control tsp PLL output divider frequency. $clk=clk\_src/(div\_con+1)$
7	RO	0x0	reserved
6:5	RW	0x0	rga_clk_pll_sel Control RGA core clock work frequency selection. 2'b00:select general pll clock 2'b01:select codec pll clock 2'b10:select rga src clock
4:0	RW	0x01	rga_clk_div_con Control RGA PLL output divider frequency. $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL23\_CON**

Address: Operational Base + offset (0x00a0)

Internal clock select and divide register23

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:14	RW	0x1	hdcp_pll_sel Control HDCP clock work frequency selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmi phy pll clock
13:8	RW	0x01	hdcp_div_con Control HDCP divider frequency. $clk=clk\_src/(div\_con+1)$
7	RO	0x0	reserved
6:5	RW	0x1	wifi_pll_sel Control WIFI clock work frequency selection 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select usb phy 480M clock
4:0	RW	0x0f	wifi_div_con Control WIFI divider frequency. $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL24\_CON**

Address: Operational Base + offset (0x00a4)

Internal clock select and divide register24

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:6	RW	0x00f	tsadc_div_con Control SARADC clock divider frequency $clk=clk\_src/(div\_con+1)$
5	RW	0x0	crypto_pll_sel Control CTYPTO clock work frequency selection. 1'b0:select codec pll clock 1'b1:select general pll clock
4:0	RW	0x03	crypto_div_con Control CTYPTO clock divider frequency. $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL25\_CON**

Address: Operational Base + offset (0x00a8)

Internal clock select and divide register25

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:9	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
8	RW	0x1	spi_clk_pll_sel Control SPI clock pll source selection. 1'b0: select codec pll clock 1'b1: select general pll clock
7	RO	0x0	reserved
6:0	RW	0x1f	spi_div_con Control SPI clock divider frequency. $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL26\_CON**

Address: Operational Base + offset (0x00ac)

Internal clock select and divide register26

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:10	RO	0x0	reserved
9:8	RW	0x0	ddr_clk_pll_sel DDR clock pll source selection. 2'b00: select ddr_pll_clk 2'b01: select general_pll_clk 2'b10: select arm_pll
7:2	RO	0x0	reserved
1:0	RW	0x0	ddr_div_sel Control DDR divider frequency. 2'b00: clk_ddr_src:clk_ddrphy = 1:1 2'b01: clk_ddr_src:clk_ddrphy = 2:1 2'b11: clk_ddr_src:clk_ddrphy = 4:1

**CRU\_CLKSEL27\_CON**

Address: Operational Base + offset (0x00b0)

Internal clock select and divide register27

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:8	RW	0x01	vop_dclk_div_con Control VOP clock divider frequency. $clk=clk\_src/(div\_con+1)$
7:2	RO	0x0	reserved
1	RW	0x0	vop_pll_sel Control vop clock PLL source selection. 1'b0: select hdmi phy pll clock 1'b1: select vop dclk

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
0	RW	0x0	vop_dclk_pll_sel Control VOP display clock PLL source selection. 1'b0: select general pll clock 1'b1: select codec pll clock

**CRU\_CLKSEL28\_CON**

Address: Operational Base + offset (0x00b4)

Internal clock select and divide register28

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:14	RW	0x1	cabac_clk_pll_sel Control CABAC clock pll source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmiphy pll clock 2'b11: select usb phy 480m clock
13	RO	0x0	reserved
12:8	RW	0x01	cabac_clk_div_con Control CABAC clock divider frequency $clk=clk\_src/(div\_con+1)$
7:6	RW	0x1	rkvdec_aclk_pll_sel Control RKVDEC clock pll source selection 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmiphy pll clock 2'b11: select usb phy 480m clock
5	RO	0x0	reserved
4:0	RW	0x01	rkvdec_aclk_div_con Control RKVDEC clock divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL29\_CON**

Address: Operational Base + offset (0x00b8)

Internal clock select and divide register29

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:13	RO	0x0	reserved
12	RW	0x0	gmac_clk_src_sel Control gmac clock source selection. 1'b0: select cru generated clock 1'b1: select IO input clock

Bit	Attr	Reset Value	Description
11	RW	0x0	use_inter_macphy_txrx Control gmac_ctrl tx/rx clock clock selection. 1'b0: select cru generated clock 1'b1: select internal mac phy clock
10	RW	0x0	use_inter_macphy_50m Control gmac_ctrl 50M clock clock selection. 1'b0: select cru generated clock 1'b1: select internal mac phy clock
9:8	RW	0x0	macphy_div_con Control MAC phy divider frequency $clk=clk\_src/(div\_con+1)$
7:3	RO	0x0	reserved
2:0	RW	0x0	hdmi phy_div_con Control HDMI phy divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL30\_CON**

Address: Operational Base + offset (0x00bc)

Internal clock select and divide register30

Bit	Attr	Reset Value	Description
31:0	WO	0x0bb8ea60	i2s2_frac_div_con Control i2s2 frac clock divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL31\_CON**

Address: Operational Base + offset (0x00c0)

Internal clock select and divide register31

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RO	0x0	reserved
14:13	RW	0x1	hdcp_aclk_pll_sel Control HDCP AXI clock PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmi phy pll clock 2'b11: select usb phy 480m clock
12:8	RW	0x01	hdcp_aclk_div_con Control HDCP AXI clock divider frequency $clk=clk\_src/(div\_con+1)$
7	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
6:5	RW	0x1	iep_aclk_pll_sel Control IEP AXI clock PLL source selection 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmiphy pll clock 2'b11: select usb phy 480m clock
4:0	RW	0x01	iep_aclk_div_con Control IEP AXI clock divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL32\_CON**

Address: Operational Base + offset (0x00c4)

Internal clock select and divide register32

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:7	RO	0x0	reserved
6:5	RW	0x1	vpu_aclk_pll_sel Control VPU AXI clock PLL source selection 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmiphy pll clock 2'b11: select usb phy 480m clock
4:0	RW	0x01	vpu_aclk_div_con Control VPU AXI clock divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL33\_CON**

Address: Operational Base + offset (0x00c8)

Internal clock select and divide register33

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RO	0x0	reserved
14:13	RW	0x1	rga_src_clk_pll_sel Control RGA source clock PLL source selection 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmiphy pll clock 2'b11: select usb phy 480m clock
12:8	RW	0x01	rga_aclk_div_con Control RGA AXI clock divider frequency $clk=clk\_src/(div\_con+1)$

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
7	RO	0x0	reserved
6:5	RW	0x1	vop_aclk_pll_sel Control VOP AXI clock PLL source selection 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmiphy pll clock 2'b11: select usb phy 480m clock
4:0	RW	0x01	vop_aclk_div_con Control VOP AXI clock divider frequency $clk=clk\_src/(div\_con+1)$

**CRU\_CLKSEL34\_CON**

Address: Operational Base + offset (0x00cc)

Internal clock select and divide register34

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RO	0x0	reserved
14:13	RW	0x1	rkvdec_core_clk_pll_sel RKVDEC CORE clock PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmiphy pll clock 2'b11: select usbphy 480M clock
12:8	RW	0x01	rkvdec_core_clk_div_con RKVDEC CORE clock divider frequency. $clk=clk\_src/(div\_con+1)$
7	RO	0x0	reserved
6:5	RW	0x1	gpu_aclk_pll_sel Control GPU AXI clock PLL source selection. 2'b00: select codec pll clock 2'b01: select general pll clock 2'b10: select hdmiphy pll clock 2'b11: select usb phy 480m clock
4:0	RW	0x01	gpu_aclk_div_con Control GPU AXI clock divider frequency. $clk=clk\_src/(div\_con+1)$

**CRU\_CLKGATE0\_CON**

Address: Operational Base + offset (0x00d0)

Internal clock gating control register0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
------------	-------------	--------------------	--------------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	testclk_gate_en test output clock disable When HIGH, disable clock
14	RW	0x0	clk_i2s1_gate_en I2S1 clock disable. When HIGH, disable clock
13	RW	0x0	clk_i2s1_out_gate_en I2S1 output clock disable. When HIGH, disable clock
12	RO	0x0	reserved
11	RW	0x0	clk_i2s1_frac_src_en I2S1 fraction divider source clock disable. When HIGH, disable clock
10	RW	0x0	clk_i2s1_src_gate_en I2S1 source clock disable. When HIGH, disable clock
9	RW	0x0	clk_i2s2_gate_en I2S2 clock disable. When HIGH, disable clock
8	RW	0x0	clk_i2s2_frac_src_gate_en I2S2 fraction divider source clock disable. When HIGH, disable clock
7	RW	0x0	clk_i2s2_src_gate_en I2S2 source clock disable. When HIGH, disable clock
6	RW	0x0	core_gpll_clk_gate_en CORE gpll clock disable. When HIGH, disable clock
5	RW	0x0	clk_i2s0_gate_en I2S0 clock disable. When HIGH, disable clock
4	RW	0x0	clk_i2s0_frac_src_gate_en I2S0 fraction divider source clock disable. When HIGH, disable clock
3	RW	0x0	clk_i2s0_src_gate_en I2S0 source clock disable. When HIGH, disable clock
2	RW	0x0	clk_ddrphy_src_gate_en clk_ddrphy source clock disable. When HIGH, disable clock

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
1	RW	0x0	bus_src_clk_en BUS clock source clock disable. When HIGH, disable clock
0	RO	0x0	reserved

**CRU\_CLKGATE1\_CON**

Address: Operational Base + offset (0x00d4)

Internal clock gating control register1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:14	RO	0x0	reserved
13	RW	0x0	clk_uart2_frac_src_gate_en UART2 fraction divider source clock disable. When HIGH, disable clock
12	RW	0x0	clk_uart2_src_gate_en UART2 source clock disable. When HIGH, disable clock
11	RW	0x0	clk_uart1_frac_src_gate_en UART1 fraction divider source clock disable. When HIGH, disable clock
10	RW	0x0	clk_uart1_src_gate_en UART1 source clock disable. When HIGH, disable clock
9	RW	0x0	clk_uart0_frac_src_gate_en UART0 fraction divider source clock disable. When HIGH, disable clock
8	RW	0x0	clk_uart0_src_gate_en UART0 source clock disable. When HIGH, disable clock
7	RW	0x0	clk_mac_src_gate_en MAC source clock disable. When HIGH, disable clock
6	RW	0x0	clk_otgphy1_gate_en OTGPHY1 clock(clk_otgphy1) disable. When HIGH, disable clock
5	RW	0x0	clk_otgphy0_gate_en OTGPHY0 clock(clk_otgphy0) disable. When HIGH, disable clock
4	RW	0x0	aclk_hdcp_src_gate_en hdcp source aclk disable. When HIGH, disable clock

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
3	RW	0x0	clk_jtag_gate_en JTAG clock disable. When HIGH, disable clock
2	RW	0x0	aclk_rga_src_en rga source aclk disable. When HIGH, disable clock
1	RW	0x0	aclk_vop_src_en vop source aclk disable. When HIGH, disable clock
0	RW	0x0	clk_nandc_src_gate_en nandc source clock disable. When HIGH, disable clock

**CRU\_CLKGATE2\_CON**

Address: Operational Base + offset (0x00d8)

Internal clock gating control register2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	clk_wifi_src_gate_en WIFI 24M clock disable. When HIGH, disable clock
14	RW	0x0	clk_emmc_src_gate_en EMMC source clock disable. When HIGH, disable clock
13	RW	0x0	clk_sdio_src_gate_en SDIO source clock disable. When HIGH, disable clock
12	RW	0x0	clk_spdif_frac_src_gate_en SPDIF fraction divider source clock disable. When HIGH, disable clock
11	RW	0x0	clk_mmc0_src_gate_en SDMMC0 source clock disable. When HIGH, disable clock
10	RW	0x0	clk_spdif_src_gate_en SPDIF source clock disable. When HIGH, disable clock
9	RW	0x0	clk_spi_src_gate_en SPI source clock disable. When HIGH, disable clock
8	RW	0x0	clk_tsadc_src_gate_en tsadc source clock disable. When HIGH, disable clock

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
7	RW	0x0	clk_crypto_src_en crypto source clock disable. When HIGH, disable clock
6	RW	0x0	clk_tsp_src_en tsp source clock disable. When HIGH, disable clock
5:3	RO	0x0	reserved
2	RW	0x0	clk_gmac_out_en GMAC clock disable. When HIGH, disable clock
1	RW	0x0	clk_ddrmon_en ddr monitor clk disable. When HIGH, disable clock
0	RW	0x0	clk_periph_src_gate_en PERIPH system source clock disable. When HIGH, disable clock

**CRU\_CLKGATE3\_CON**

Address: Operational Base + offset (0x00dc)

Internal clock gating control register3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:14	RO	0x0	reserved
13	RW	0x0	ackl_gpu_src_gate_en GPU AXI source clock disable. When HIGH, disable clock
12	RO	0x0	reserved
11	RW	0x0	ackl_vpu_src_gate_en VPU AXI source clock disable. When HIGH, disable clock
10:9	RO	0x0	reserved
8	RW	0x0	clk_hdmi_cec_src_gate_en HDMI CEC source clock disable. When HIGH, disable clock
7	RW	0x0	clk_hdmi_hdcp_gate_en HDMI HDCP clk disable. When HIGH, disable clock
6	RW	0x0	clk_rga_src_gate_en rga souce clock disable. When HIGH, disable clock
5	RW	0x0	clk_hdcp_gate_en HDCP clock disable. When HIGH, disable clock

Bit	Attr	Reset Value	Description
4	RW	0x0	clk_rkvdec_core_src_gate_en RKVDEC CORE souce clock disable. When HIGH, disable clock
3	RW	0x0	clk_rkvdec_cabac_src_gate_en RKVDEC cabac source clock disable. When HIGH, disable clock
2	RW	0x0	ack_rkvdec_src_gate_en RKVDEC souce clock disable. When HIGH, disable clock
1	RW	0x0	dclk_vop_src_gate_en VOP DCLK souce clock disable. When HIGH, disable clock
0	RW	0x0	ack_iep_src_gate_en iep AXI source clock disable. When HIGH, disable clock

**CRU\_CLKGATE4\_CON**

Address: Operational Base + offset (0x00e0)

Internal clock gating control register0

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:6	RO	0x0	reserved
5	RW	0x0	hclk_rkvdec_gate_en RKVDEC AHB clk disable. When HIGH, disable clock
4	RW	0x0	hclk_video_gate_en VIDEO AHB clk disable. When HIGH, disable clock
3	RO	0x0	reserved
2	RW	0x0	ack_a53_gic400_en gic400 AXI clk disable When HIGH, disable clock
1	RW	0x0	clk_core_periph_gate_en CORE_PERIPH clk disable When HIGH, disable clock
0	RW	0x0	ack_core_en CORE AXI clk disable. When HIGH, disable clock

**CRU\_CLKGATE5\_CON**

Address: Operational Base + offset (0x00e4)

Internal clock gating control register5

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:8	RO	0x0	reserved
7	RW	0x0	clk_macphy_gate_en MACPHY clk disable When HIGH, disable clock
6	RW	0x0	clk_mac_tx_gate_en MAC tx clk disable When HIGH, disable clock
5	RW	0x0	clk_mac_rx_gate_en MAC rx clk disable. When HIGH, disable clock
4	RW	0x0	clk_mac_refout_gate_en MAC refout disable. When HIGH, disable clock
3	RW	0x0	clk_mac_ref_gate_en MAC ref clk disable. When HIGH, disable clock
2	RW	0x0	pclk_periph_gate_en PERIPH APB disable When HIGH, disable clock
1	RW	0x0	hclk_periph_gate_en PERIPH AHB gate_en When HIGH, disable clock
0	RW	0x0	ack_periph_gate_en PERIPH AXI clk disable When HIGH, disable clock

**CRU\_CLKGATE6\_CON**

Address: Operational Base + offset (0x00e8)

Internal clock gating control register6

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:14	RO	0x0	reserved
13	RW	0x0	pclk_ddr_gate_en DDR APB clock disable. When HIGH, disable clock
12:11	RO	0x0	reserved
10	RW	0x0	clk_timer5_gate_en Timer5 clock disable. When HIGH, disable clock

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
9	RW	0x0	clk_timer4_gate_en Timer4 clock disable. When HIGH, disable clock
8	RW	0x0	clk_timer3_gate_en Timer3 clock disable. When HIGH, disable clock
7	RW	0x0	clk_timer2_gate_en Timer2 clock disable. When HIGH, disable clock
6	RW	0x0	clk_timer1_gate_en Timer1 clock disable. When HIGH, disable clock
5	RW	0x0	clk_timer0_gate_en Timer0 clock disable. When HIGH, disable clock
4	RW	0x0	pclk_phy_src_gate_en PHY APB clk disable. When HIGH, disable clock
3	RW	0x0	pclk_bus_gate_en PD_BUS APB clk disable. When HIGH, disable clock
2	RW	0x0	pclk_bus_src_gate_en PD_BUS APB source clk disable. When HIGH, disable clock
1	RW	0x0	hclk_bus_gate_en PD_BUS AHB clock disable. When HIGH, disable clock
0	RW	0x0	ackl_bus_gate_en PD_BUS AXI clock disable. When HIGH, disable clock

**CRU\_CLKGATE7\_CON**

Address: Operational Base + offset (0x00ec)

Internal clock gating control register7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	ackl_gpu_noc_gate_en GPU NOC AXI clk disable. When HIGH, disable clock
14	RW	0x0	ackl_gpu_gate_en GPU AXI clk disable. When HIGH, disable clock
13:2	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
1	RW	0x0	clk4x_ddrphy_gate_en DDR phy clk4x disable. When HIGH, disable clock
0	RW	0x0	clk_ddrphy_gate_en DDR phy clk disable. When HIGH, disable clock

**CRU\_CLKGATE8\_CON**

Address: Operational Base + offset (0x00f0)

Internal clock gating control register8

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	pclk_i2c0_gate_en I2C0 APB clock disable. When HIGH, disable clock
14	RW	0x0	pclk_efuse_256_gate_en EFUSE_256 APB clock disable. When HIGH, disable clock
13	RW	0x0	pclk_efuse_1024_gate_en EFUSE_1024 APB clock disable. When HIGH, disable clock
12	RW	0x0	hclk_crypto_slv_gate_en CRYPTO slave hclk disable. When HIGH, disable clock
11	RW	0x0	hclk_crypto_mst_gate_en CRYPTO master hclk disable. When HIGH, disable clock
10	RW	0x0	hclk_spdif_8ch_gate_en SPDIF_8ch AHB clock disable. When HIGH, disable clock
9	RW	0x0	hclk_i2s2_2ch_gate_en i2s2_2ch AHB clock disable. When HIGH, disable clock
8	RW	0x0	hclk_i2s1_8ch_gate_en i2s1_8ch AHB clock disable. When HIGH, disable clock
7	RW	0x0	hclk_i2s0_8ch_gate_en i2s0_8ch AHB clock disable. When HIGH, disable clock
6	RW	0x0	pclk_ddrmon_gate_en DDR monitor APB clock disable. When HIGH, disable clock

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
5	RW	0x0	clk_ddrupctl_gate_en DDR upctrl clock disable. When HIGH, disable clock
4	RW	0x0	pclk_ddrupctl_gate_en DDR upctrl APB clock disable. When HIGH, disable clock
3	RW	0x0	hclk_rom_gate_en ROM AHB clk disable. When HIGH, disable clock
2	RW	0x0	ack_dmac_bus_gate_en DMAC AXI bus clock disable. When HIGH, disable clock
1	RW	0x0	clk_intmem_mbist_gate_en Intmem mbist clock disable. When HIGH, disable clock
0	RW	0x0	ack_intmem_gate_en Intmem AXI clock disable. When HIGH, disable clock

**CRU\_CLKGATE9\_CON**

Address: Operational Base + offset (0x00f4)

Internal clock gating control register9

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	pclk_tsadc_gate_en TSADC APB clock disable. When HIGH, disable clock
14	RW	0x0	pclk_uart2_gate_en UART2 APB clock disable. When HIGH, disable clock
13	RW	0x0	pclk_uart1_gate_en UART1 APB clock disable. When HIGH, disable clock
12	RW	0x0	pclk_uart0_gate_en UART0 APB clock disable. When HIGH, disable clock
11	RW	0x0	pclk_gpio3_gate_en GPIO3 APB clock disable. When HIGH, disable clock
10	RW	0x0	pclk_gpio2_gate_en GPIO2 APB clock disable. When HIGH, disable clock

Bit	Attr	Reset Value	Description
9	RW	0x0	pclk_gpio1_gate_en GPIO1 APB clock disable. When HIGH, disable clock
8	RW	0x0	pclk_gpio0_gate_en GPIO0 APB clock disable. When HIGH, disable clock
7	RW	0x0	pclk_rk_pwm_gate_en RK PWM APB clock disable. When HIGH, disable clock
6	RW	0x0	pclk_spi_gate_en SPI APB clock disable. When HIGH, disable clock
5	RW	0x0	pclk_stimer_gate_en Stimer APB clock disable. When HIGH, disable clock
4	RW	0x0	pclk_timer_6ch_gate_en Timer_6ch APB clock disable. When HIGH, disable clock
3	RO	0x0	reserved
2	RW	0x0	pclk_i2c3_gate_en I2C3 APB clock disable. When HIGH, disable clock
1	RW	0x0	pclk_i2c2_gate_en I2C1 APB clock disable. When HIGH, disable clock
0	RW	0x0	pclk_i2c1_gate_en I2C1 APB clock disable. When HIGH, disable clock

**CRU\_CLKGATE10\_CON**

Address: Operational Base + offset (0x00f8)

Internal clock gating control register10

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:13	RO	0x0	reserved
12	RW	0x0	clk_tspin_gate_en TSP input clock disable. When HIGH, disable clock
11	RW	0x0	hclk_tsp_gate_en TSP AHB clock disable. When HIGH, disable clock

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
10	RW	0x0	pclk_sim_gate_en smart card controller APB clock disable. When HIGH, disable clock
9	RW	0x0	pclk_phy_noc_gate_en NOC phy APB clock disable. When HIGH, disable clock
8	RW	0x0	pclk_vadcphy_gate_en VDAC phy APB clock disable. When HIGH, disable clock
7	RW	0x0	pclk_hdmiphy_gate_en hdmi phy APB clock disable. When HIGH, disable clock
6	RW	0x0	pclk_sgrf_gate_en SGRF APB clock disable. When HIGH, disable clock
5	RW	0x0	pclk_acodecphy_gater_en Audio codec phy APB clock disable. When HIGH, disable clock
4	RW	0x0	pclk_cru_gate_en CRU APB clock disable. When HIGH, disable clock
3	RW	0x0	pclk_ddrphy_en DDR phy APB clock disable. When HIGH, disable clock
2	RW	0x0	pclk_msch_noc_gate_en MSCH NOC APB clock disable. When HIGH, disable clock
1	RW	0x0	ack_bus_noc_gate_en PD_BUS NOC AXI clock disable. When HIGH, disable clock
0	RW	0x0	pclk_grf_gate_en GRF APB clock disable. When HIGH, disable clock

**CRU\_CLKGATE11\_CON**

Address: Operational Base + offset (0x00fc)

Internal clock gating control register11

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RO	0x0	reserved
14	RW	0x0	hclk_host2_arb_gate_en HOST2 ARB AHB clock disable. Field0000 Description

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
13	RW	0x0	hclk_otg_pmu_gate_en OTG PMU AHB clock disable. When HIGH, disable clock
12	RW	0x0	hclk_otg_gate_en OTG AHB clock disable. When HIGH, disable clock
11	RO	0x0	reserved
10	RW	0x0	hclk_host2_gate_en HOST2 AHB clock disable. When HIGH, disable clock
9	RW	0x0	hclk_host1_arb_gate_en HOST1 ARB AHB clock disable. When HIGH, disable clock
8	RW	0x0	hclk_host1_gate_en HOST1 AHB clock disable. When HIGH, disable clock
7	RW	0x0	hclk_host0_arb_gate_en HOST0 ARB AHB clock disable. When HIGH, disable clock
6	RW	0x0	hclk_host0_gate_en HOST0 AHB clock disable. When HIGH, disable clock
5	RW	0x0	pclk_gmac_gate_en GMAC APB clock disable. When HIGH, disable clock
4	RW	0x0	aclk_gmac_gate_en GMAC AXI clock disable. When HIGH, disable clock
3	RW	0x0	hclk_nandc_gate_en NANDC AHB clock disable. When HIGH, disable clock
2	RW	0x0	hclk_emmc_gate_en EMMC AHB clock disable. When HIGH, disable clock
1	RW	0x0	hclk_sdio_gate_en SDIO AHB clock disable. When HIGH, disable clock
0	RW	0x0	hclk_sdmmc_gate_en SDMMC AHB clock disable. When HIGH, disable clock

**CRU\_CLKGATE12\_CON**

Address: Operational Base + offset (0x0100)

Internal clock gating control register12

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
------------	-------------	--------------------	--------------------

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:3	RO	0x0	reserved
2	RW	0x0	pclk_peri_noc_gate_en Peri NOC APB clock disable. When HIGH, disable clock
1	RW	0x0	hclk_peri_noc_gate_en Peri NOC AHB clock disable. When HIGH, disable clock
0	RW	0x0	ackl_peri_noc_gate_en Peri NOC AXI clock disable. When HIGH, disable clock

**CRU\_CLKGATE13\_CON**

Address: Operational Base + offset (0x0104)

Internal clock gating control register13

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:14	RO	0x0	reserved
13	RW	0x0	hclk_vop_noc_gate_en VOP noc AHB clock disable. When HIGH, disable clock
12	RW	0x0	ackl_vop_noc_gate_en VOP noc AXI clock disable. When HIGH, disable clock
11	RW	0x0	ackl_rga_noc_gate_en RGA noc AXI clock disable. When HIGH, disable clock
10	RW	0x0	ackl_hdcp_noc_gate_en HDCP noc AXI clock disable. When HIGH, disable clock
9	RW	0x0	ackl_iep_noc_gate_en IEP noc AXI clock disable. When HIGH, disable clock
8	RW	0x0	hclk_vio_noc_gate_en VIO NOC AHB clock disable. When HIGH, disable clock
7	RW	0x0	hclk_vio_ahb_arbi_gate_en Vio_ahb_arbi AHB clock disable When HIGH, disable clock

Bit	Attr	Reset Value	Description
6	RW	0x0	hclk_vop_gate_en VOP AHB clock disable. When HIGH, disable clock
5	RW	0x0	ackl_vop_gate_en VOP AXI clock disable. When HIGH, disable clock
4	RO	0x0	reserved
3	RW	0x0	hclk_iep_gate_en IEP AHB clock disable. When HIGH, disable clock
2	RW	0x0	ackl_iep_gate_en IEP AXI clock disable. When HIGH, disable clock
1	RW	0x0	hclk_rga_gate_en RGA AHB clock disable. When HIGH, disable clock
0	RW	0x0	ackl_rga_gate_en RGA AXI clock disable. When HIGH, disable clock

**CRU\_CLKGATE14\_CON**

Address: Operational Base + offset (0x0108)

Internal clock gating control register14

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:13	RO	0x0	reserved
12	RW	0x0	hclk_hdcpmmu_gate_en HDPC MMU AHB clock disable. When HIGH, disable clock
11	RW	0x0	pclk_hdcp_gate_en HDPC APB clock disable. When HIGH, disable clock
10	RW	0x0	ackl_hdcp_gate_en HDPC AXI clock disable. When HIGH, disable clock
9:8	RO	0x0	reserved
7	RW	0x0	hclk_vio_h2p_gate_en VIO AHB to APB clock disable. When HIGH, disable clock
6	RW	0x0	pclk_hdmi_ctrl_gate_en HDMI control APB clock disable. When HIGH, disable clock
5:0	RO	0x0	reserved

**CRU\_CLKGATE15\_CON**

Address: Operational Base + offset (0x010c)

Internal clock gating control register15

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:8	RO	0x0	reserved
7	RW	0x0	hclk_rkvdec_gate_en RKVDEC AHB clock disable. When HIGH, disable clock
6	RW	0x0	ack_rkvdec_gate_en RKVDEC AXI clock disable. When HIGH, disable clock
5	RW	0x0	hclk_vpu_noc_gate_en VPU NOC AHB clock disable. When HIGH, disable clock
4	RW	0x0	ack_vpu_noc_gate_en VPU NOC AXI clock disable. When HIGH, disable clock
3	RW	0x0	hclk_rkvdec_noc_gate_en RKVDEC NOC AHB clock disable. When HIGH, disable clock
2	RW	0x0	ack_rkvdec_noc_gate_en RKVDEC NOC AXI clock disable. When HIGH, disable clock
1	RW	0x0	hclk_vpu_gate_en VPU AHB clock disable. When HIGH, disable clock
0	RW	0x0	ack_vpu_gate_en VPU AXI clock disable. When HIGH, disable clock

**CRU\_SOFTRST0\_CON**

Address: Operational Base + offset (0x0110)

Internal software reset control register0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	I2c_srstn_req L2C software reset request. When HIGH, reset relative logic

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
14	RW	0x0	noc_asrstn_req NOC AXI software reset request. When HIGH, reset relative logic
13	R/W SC	0x0	ackl_core_srstn_req core AXI clock software reset request. When HIGH, reset relative logic
12	RW	0x0	core_top_dbg_srstn_req CPU top debug software reset request. When HIGH, reset relative logic
11	RW	0x0	core3_dbg_srstn_req Core3 CPU debug software reset request. When HIGH, reset relative logic
10	RW	0x0	core2_dbg_srstn_req Core2 CPU debug software reset request. When HIGH, reset relative logic
9	RW	0x0	core1_dbg_srstn_req Core1 CPU debug software reset request. When HIGH, reset relative logic
8	RW	0x0	core0_dbg_srstn_req Core0 CPU debug software reset request. When HIGH, reset relative logic
7	R/W SC	0x0	core3_srstn_req Core3 CPU software reset request. When HIGH, reset relative logic
6	R/W SC	0x0	core2_srstn_req Core2 CPU software reset request. When HIGH, reset relative logic
5	R/W SC	0x0	core1_srstn_req Core1 CPU software reset request. When HIGH, reset relative logic
4	R/W SC	0x0	core0_srstn_req core0 CPU software reset request. When HIGH, reset relative logic
3	R/W SC	0x0	core3_posrstn_req Core3 CPU PO software reset request. When HIGH, reset relative logic
2	R/W SC	0x0	core2_posrstn_req Core2 CPU PO software reset request. When HIGH, reset relative logic
1	R/W SC	0x0	core1_posrstn_req Core1 CPU PO software reset request. When HIGH, reset relative logic
0	R/W SC	0x0	core0_posrstn_req core0 CPU PO software reset request. When HIGH, reset relative logic

**CRU\_SOFTRST1\_CON**

Address: Operational Base + offset (0x0114)

Internal software reset control register1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	efuse256_srstn_req EFUSE256 APB software reset request. When HIGH, reset relative logic
14	RW	0x0	efuse1024_srstn_req EFUSE1024 APB software reset request. When HIGH, reset relative logic
13	RW	0x0	msch_srstn_req Msch software reset request. When HIGH, reset relative logic
12	RW	0x0	dfimon_srstn_req DFI monitor software reset request. Field0000 Description
11	RW	0x0	acodec_psrstn_req Audio codec software reset request. When HIGH, reset relative logic
10	RW	0x0	i2s2_srstn_req I2S2 software reset request. When HIGH, reset relative logic
9	RW	0x0	i2s1_srstn_req I2S1 software reset request. When HIGH, reset relative logic
8	RW	0x0	i2s0_srstn_req I2S0 software reset request. When HIGH, reset relative logic
7	RW	0x0	otg_adp_srstn_req OTG ADP software reset request. When HIGH, reset relative logic
6	RW	0x0	rom_srstn_req ROM software reset request. When HIGH, reset relative logic
5	RW	0x0	intmem_srstn_req Internal memory software reset request. When HIGH, reset relative logic
4	RW	0x0	spdif_srstn_req SPDIF software reset request. When HIGH, reset relative logic

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
3	RW	0x0	bussys_hsrstn_req BUSSYS software reset request. When HIGH, reset relative logic
2	RW	0x0	cpusys_hsrstn_req CPU AHB software reset request. When HIGH, reset relative logic
1:0	RO	0x0	reserved

**CRU\_SOFTRST2\_CON**

Address: Operational Base + offset (0x0118)

Internal software reset control register2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RO	0x0	reserved
14	RW	0x0	i2c3_srstn_req I2C3 software reset request. When HIGH, reset relative logic
13	RW	0x0	i2c2_srstn_req I2C2 software reset request. When HIGH, reset relative logic
12	RW	0x0	i2c1_srstn_req I2C1 software reset request. When HIGH, reset relative logic
11	RW	0x0	i2c0_srstn_req I2C0 software reset request. When HIGH, reset relative logic
10	RW	0x0	phynoc_srstn_req PHY APB NOC software reset request. When HIGH, reset relative logic
9	RW	0x0	uart2_srstn_req UART2 software reset request. When HIGH, reset relative logic
8	RW	0x0	uart1_srstn_req UART1 software reset request. When HIGH, reset relative logic
7	RW	0x0	uart0_srstn_req UART0 software reset request. When HIGH, reset relative logic
6	RW	0x0	periph_noc_psrstn_req Periph_noc apb bus software reset request. When HIGH, reset relative logic

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
5	RW	0x0	periph_noc_hsrstn_req Periph_noc ahb bus software reset request. When HIGH, reset relative logic
4	RW	0x0	periph_noc_asrstn_req Periph_noc axi bus software reset request. When HIGH, reset relative logic
3	RW	0x0	gpio3_srstn_req GPIO3 software reset request. When HIGH, reset relative logic
2	RW	0x0	gpio2_srstn_req GPIO2 software reset request. When HIGH, reset relative logic
1	RW	0x0	gpio1_srstn_req GPIO1 software reset request. When HIGH, reset relative logic
0	RW	0x0	gpio0_srstn_req GPIO0 software reset request. When HIGH, reset relative logic

**CRU\_SOFTRST3\_CON**

Address: Operational Base + offset (0x011c)

Internal software reset control register3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	macphy_srstn_req MAC PHY software reset request. When HIGH, reset relative logic
14:11	RO	0x0	reserved
10	RW	0x0	periph_noc_hsrstn_req PERIPH NOC AHB software reset request. When HIGH, reset relative logic
9	RO	0x0	reserved
8	RW	0x0	gmac_srstn_req GMAC software reset request. When HIGH, reset relative logic
7	RW	0x0	grf_srstn_req GRF software reset request. When HIGH, reset relative logic
6	RW	0x0	sgrf_srstn_req SGRF software reset request. When HIGH, reset relative logic

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
5	RW	0x0	crypto_srstn_req CRYPTO software reset request. When HIGH, reset relative logic
4	RW	0x0	dap_noc_srstn_req DAP NOC software reset request. When HIGH, reset relative logic
3	RW	0x0	dap_srstn_req DAP software reset request. When HIGH, reset relative logic
2	RO	0x0	reserved
1	RW	0x0	a53_gic_srstn_req A53 software reset request. (valid in A53 version) When HIGH, reset relative logic
0	RW	0x0	pwm_srstn_req PWM software reset request. When HIGH, reset relative logic

**CRU\_SOFTRST4\_CON**

Address: Operational Base + offset (0x0120)

Internal software reset control register4

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	ddrmsch_srstn_req DDR memory scheduler software reset request. When HIGH, reset relative logic
14	RW	0x0	usbpor1_srst_req USB combo PHY1 por software reset request. When HIGH, reset relative logic
13	RW	0x0	usbpor0_srst_req USB combo PHY0 por software reset request. When HIGH, reset relative logic
12	RW	0x0	host_ctrl2_srstn_req USB HOST controller2 software reset request. When HIGH, reset relative logic
11	RW	0x0	usbhost2_srstn_req USBHOST2 software reset request. When HIGH, reset relative logic
10	RW	0x0	host_ctrl1_srstn_req USB HOST controller1 software reset request. When HIGH, reset relative logic
9	RW	0x0	usbhost1_srstn_req USBHOST1 software reset request. When HIGH, reset relative logic

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
8	RW	0x0	host_ctrl0_srstn_req USB HOST controller0 software reset request. When HIGH, reset relative logic Host Controller utmi_clk domain reset
7	RW	0x0	usbhost0_srstn_req USBHOST0 software reset request. When HIGH, reset relative logic Host Controller hclk domain reset.
6	RW	0x0	otgc_srstn_req OTG controller software reset request. When HIGH, reset relative logic. OTG Controller utmi_clk domain reset.
5	RW	0x0	usbotg_srstn_req USBOTG software reset request. When HIGH, reset relative logic OTG Controller hclk domain reset.
4	RW	0x0	nandc_srstn_req NANDC software reset request. When HIGH, reset relative logic
3:1	RO	0x0	reserved
0	RW	0x0	dma_srstn_req DMA software reset request. When HIGH, reset relative logic

**CRU\_SOFTRST5\_CON**

Address: Operational Base + offset (0x0124)

Internal software reset control register5

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	vop_noc_asrstn_req VOP NOC AXI software reset request. When HIGH, reset relative logic
14	RW	0x0	host2_echi_srstn_req HOST2 echi software reset request. When HIGH, reset relative logic
13	RW	0x0	host1_echi_srstn_req HOST1 echi software reset request. When HIGH, reset relative logic
12	RW	0x0	host0_echi_srstn_req HOST0 echi software reset request. When HIGH, reset relative logic

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
11	RW	0x0	ddrctrl_psrstn_req DDR controller APB software reset request. When HIGH, reset relative logic
10	RW	0x0	ddrctrl_srstn_req DDR controller software reset request. When HIGH, reset relative logic
9	RW	0x0	ddrphy_psrstn_req DDR PHY APB software reset request. When HIGH, reset relative logic
8	RW	0x0	ddrphy_srstn_req DDR PHY software reset request. When HIGH, reset relative logic
7	RW	0x0	tsadc_srstn_req TSADC software reset request. When HIGH, reset relative logic
6	RW	0x0	tsp_srstn_req TSP work clk software reset request. When HIGH, reset relative logic
5	RW	0x0	tsp_hsrstn_req TSP AHB software reset request. When HIGH, reset relative logic
4	RW	0x0	spi_srstn_req SPI software reset request. When HIGH, reset relative logic
3	RW	0x0	emmc_srstn_req EMMC software reset request. When HIGH, reset relative logic
2	RW	0x0	sdio_srstn_req SDIO software reset request. When HIGH, reset relative logic
1	RW	0x0	sdmmc_srstn_req SDMMC0 software reset request. When HIGH, reset relative logic
0	RW	0x0	smart_card_srstn_req smart_card software reset request. When HIGH, reset relative logic

**CRU\_SOFTRST6\_CON**

Address: Operational Base + offset (0x0128)

Internal software reset control register6

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15	RW	0x0	hdcp_asrstn_req HDCP AXI software reset request. When HIGH, reset relative logic
14	RW	0x0	rga_hsrstn_req RGA AHB clock software reset request. When HIGH, reset relative logic
13	RW	0x0	rga_asrstn_req RGA AXI software reset request. When HIGH, reset relative logic
12	RW	0x0	rga_noc_asrstn_req RGA NOC AXI software reset request. When HIGH, reset relative logic
11	RW	0x0	rga_srstn_req RGA software reset request. When HIGH, reset relative logic
10	RW	0x0	utmi3_srst_req UTMI3 software reset request. When HIGH, reset relative logic
9	RW	0x0	utmi2_srst_req UTMI2 POR software reset request. When HIGH, reset relative logic. USB phy analog domain reset, including both OTG and HOST phy .
8	RW	0x0	utmi1_srst_req UTMI1 software reset request. When HIGH, reset relative logic HOST phy digital domain reset. It should last at least 10 utmi_clk_1 cycles.
7	RW	0x0	utmi0_srstn_req UTMI0 software reset request. When HIGH, reset relative logic OTG phy digital domain reset. It should last at least 10 utmi_clk_0 cycles.
6	RW	0x0	vop_dsrstn_req VOP DCLK software reset request. When HIGH, reset relative logic
5	RW	0x0	vop_hsrstn_req VOP AHB clock software reset request. When HIGH, reset relative logic
4	RW	0x0	vop_asrstn_req VOP AXI clock software reset request. When HIGH, reset relative logic
3	RW	0x0	vio_noc_hsrstn_req VIO NOC AHB software reset request. When HIGH, reset relative logic

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
2	RW	0x0	iep_noc_asrstn_req IEP NOC AXI software reset request. When HIGH, reset relative logic
1	RW	0x0	vio_arbi_hsrstn_req VIO arbitor AHB software reset request. When HIGH, reset relative logic
0	RW	0x0	hdmi_psrstn_req HDMI PCLK software reset request. When HIGH, reset relative logic

**CRU\_SOFTRST7\_CON**

Address: Operational Base + offset (0x012c)

Internal software reset control register7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x0	gpu_noc_asrstn_req GPU NOC AXI software reset request. When HIGH, reset relative logic
14	RW	0x0	gpu_asrstn_req GPU AXI software reset request. When HIGH, reset relative logic
13	RW	0x0	iep_hsrstn_req IEP AHB software reset request. When HIGH, reset relative logic
12	RW	0x0	iep_asrstn_req IEP AXI software reset request. When HIGH, reset relative logic
11	RW	0x0	rkvdec_cabac_srstn_req rkvdec CABAC software reset request. When HIGH, reset relative logic
10	RW	0x0	rkvdec_core_srstn_req rkvdec CORE software reset request. When HIGH, reset relative logic
9	RW	0x0	rkvdec_noc_hsrstn_req rkvdec NOC AHB software reset request. When HIGH, reset relative logic
8	RW	0x0	rkvdec_hsrstn_req rkvdec AHB software reset request. When HIGH, reset relative logic
7	RW	0x0	rkvdec_noc_asrstn_req rkvdec NOC AXI software reset request. When HIGH, reset relative logic

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
6	RW	0x0	rkvdec_asrstn_req rkvdec AXI software reset request. When HIGH, reset relative logic
5	RW	0x0	vpu_noc_hsrstn_req vpu NOC AHB software reset request. When HIGH, reset relative logic
4	RW	0x0	vpu_noc_asrstn_req vpu NOC AXI software reset request. When HIGH, reset relative logic
3:2	RO	0x0	reserved
1	RW	0x0	vpu_hsrstn_req vpu AHB software reset request. When HIGH, reset relative logic
0	RW	0x0	vpu_asrstn_req vpu AXI software reset request. When HIGH, reset relative logic

**CRU\_SOFTRST8\_CON**

Address: Operational Base + offset (0x0130)

Internal software reset control register8

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:14	RO	0x0	reserved
13	RW	0x0	timer_6ch_psrstn_req Timer_6ch_apb bus software reset request. When HIGH, reset relative logic
12	RW	0x0	vdac_srstn_req vdac software reset request. When HIGH, reset relative logic
11	RW	0x0	hdmiphy_srstn_req hdmiphy software reset request. When HIGH, reset relative logic
10:9	RO	0x0	reserved
8	RW	0x0	vio_h2p_srstn_req pd_vio h2p bridge software reset request. When HIGH, reset relative logic
7	RW	0x0	timer5_srstn_req Timer5 software reset request. When HIGH, reset relative logic
6	RW	0x0	timer4_srstn_req Timer4 software reset request. When HIGH, reset relative logic

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
5	RW	0x0	timer3_srstn_req Timer3 software reset request. When HIGH, reset relative logic
4	RW	0x0	timer2_srstn_req Timer2 software reset request. When HIGH, reset relative logic
3	RW	0x0	timer1_srstn_req Timer1 software reset request. When HIGH, reset relative logic
2	RW	0x0	timer0_srstn_req Timer0 software reset request. When HIGH, reset relative logic
1	RW	0x0	dbg_psrstn_req DEBUG APB software reset request. When HIGH, reset relative logic
0	RW	0x0	core_dbg_srstn_req CORE DEBUG software reset request. When HIGH, reset relative logic

**CRU\_MISC\_CON**

Address: Operational Base + offset (0x0134)

SCU control register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15	RW	0x1	usb480_src_sel USB480 and 24M selection 1'b0: select 480M 1'b1: select 24M
14	RW	0x0	usb480m_phy_sel USBphy0 480m and USBphy1 480m selection 1'b0: USBphy0 480m 1'b1: USBphy1 480m
13	RW	0x1	hdmiphy_src_sel HDMIPHY and 24M selection 1'b0: select HDMIPHY output clock 1'b1: select 24M
12	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
11:8	RW	0x0	<p>testclk_sel Output clock selection for test</p> <p>3'b000: outclock_test_t = buf_clk_wifi 3'b001: outclock_test_t = buf_clk_hdmi_cec 3'b010: outclock_test_t = buf_clk_core_2wrap_pre 3'b011: outclock_test_t = buf_clk_ddrphy 3'b100: outclock_test_t = buf_aclk_iep_2wrap_pre 3'b101: outclock_test_t = buf_aclk_gpu_2wrap_pre 3'b110: outclock_test_t = buf_aclk_peri_2wrap_pre 3'b111: outclock_test_t = buf_aclk_cpu_2wrap_pre default: outclock_test_t = buf_clk_wifi</p>
7:3	RO	0x0	reserved
2	RW	0x0	<p>core_wrst_wfien 1'b0:disable 1'b1:enable</p>
1	RW	0x0	<p>core_srst_wfien 1'b0:disable 1'b1:enable</p>
0	RW	0x0	<p>warmrstn_en 1'b0:disable 1'b1:enable</p>

**CRU\_GLB\_CNT\_TH**

Address: Operational Base + offset (0x0140)

global reset wait counter threshold

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x3a98	pll_lock_period PLL lock period
15	RW	0x0	<p>wdt_glb_srst_ctrl watch dog trigger global soft reset select 1'b0: watch_dog trigger second global reset 1'b1: watch_dog trigger first global reset</p>
14	RW	0x0	<p>tsadc_glb_srst_ctrl tsadc trigger global soft reset select 1'b0: tsadc trigger second global reset 1'b1: tsadc trigger first global reset</p>
13:10	RO	0x0	reserved
9:0	RW	0x064	glb_RST_CNT_TH Global soft reset counter threshold

**CRU\_GLB\_RST\_ST**

Address: Operational Base + offset (0x0150)

global reset status

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
5	W1 C	0x0	snd_glb_tsadc_rst_st second global tsadc rst flag 1'b0: last hot reset is not second global tsadc triggered reset 1'b1: last hot reset is second global tsadc triggered reset
4	W1 C	0x0	fst_glb_tsadc_rst_st first global tsadc rst flag 1'b0: last hot reset is not first global tsadc triggered reset 1'b1: last hot reset is first global tsadc triggered reset
3	W1 C	0x0	snd_glb_wdt_RST_ST second global watch_dog rst flag 1'b0: last hot reset is not second global watch_dog triggered reset 1'b1: last hot reset is second global watch_dog triggered reset
2	W1 C	0x0	fst_glb_wdt_RST_ST first global watch_dog rst flag 1'b0: last hot reset is not first global watch_dog triggered reset 1'b1: last hot reset is first global watch_dog triggered reset
1	W1 C	0x0	snd_glb_RST_ST second global rst flag 1'b0: last hot reset is not second global rst 1'b1: last hot reset is second global rst
0	W1 C	0x0	fst_glb_RST_ST first global rst flag 1'b0: last hot reset is not first global rst 1'b1: last hot reset is first global rst

**CRU\_SDMMC\_CON0**

Address: Operational Base + offset (0x01c0)

sdmmc control0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:12	RO	0x0	reserved
11	WO	0x0	sdmmc_drv_sel sdmmc drive select. sdmmc drive select
10:3	WO	0x00	sdmmc_drv_delaynum sdmmc drive delay number. sdmmc drive delay number
2:1	WO	0x2	sdmmc_drv_degree sdmmc drive degree. sdmmc drive degree
0	WO	0x0	sdmmc_init_state sdmmc initial state. sdmmc initial state

**CRU\_SDMMC\_CON1**

Address: Operational Base + offset (0x01c4)

sdmmc control1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:12	RO	0x0	reserved
11	WO	0x0	sdmmc_sample_sel sdmmc sample select. sdmmc sample select
10:3	WO	0x00	sdmmc_sample_delaynum sdmmc sample delay number. sdmmc sample delay number
2:1	WO	0x0	sdmmc_sample_degree sdmmc sample degree. sdmmc sample degree
0	RO	0x0	reserved

**CRU\_SDIO\_CON0**

Address: Operational Base + offset (0x01c8)

sdio0 control0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:12	RO	0x0	reserved
11	WO	0x0	sdio0_drv_sel sdio0 drive select. sdio0 drive select
10:3	WO	0x00	sdio0_drv_delaynum sdio0 drive delay number. sdio0 drive delay number
2:1	WO	0x2	sdio0_drv_degree sdio0 drive degree. sdio0 drive degree
0	WO	0x0	sdio0_init_state sdio0 initial state. sdio0 initial state

**CRU\_SDIO\_CON1**

Address: Operational Base + offset (0x01cc)

sdio0 control1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:12	RO	0x0	reserved
11	WO	0x0	sdio0_sample_sel sdio0 sample select. sdio0 sample select
10:3	WO	0x00	sdio0_sample_delaynum sdio0 sample delay number. sdio0 sample delay number
2:1	WO	0x0	sdio0_sample_degree sdio0 sample degree. sdio0 sample degree
0	RO	0x0	reserved

**CRU\_EMMC\_CON0**

Address: Operational Base + offset (0x01d8)

emmc control0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:12	RO	0x0	reserved
11	WO	0x0	emmc_drv_sel emmc drive select. emmc drive select
10:3	WO	0x00	emmc_drv_delaynum emmc drive delay number. emmc drive delay number
2:1	WO	0x2	emmc_drv_degree emmc drive degree. emmc drive degree
0	WO	0x0	emmc_init_state emmc initial state. emmc initial state

**CRU\_EMMC\_CON1**

Address: Operational Base + offset (0x01dc)

emmc control1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:12	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
11	WO	0x0	emmc_sample_sel emmc sample select. emmc sample select
10:3	WO	0x00	emmc_sample_delaynum emmc sample delay number. emmc sample delay number
2:1	WO	0x0	emmc_sample_degree emmc sample degree. emmc sample degree
0	RO	0x0	reserved

**CRU\_GLB\_SRST\_FST\_VALUE**

Address: Operational Base + offset (0x01f0)

The first global software reset config value

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	glb_srst_fst_value The first global software reset config value If config 0xfd9, it will generate first global software reset.

**CRU\_GLB\_SRST SND\_VALUE**

Address: Operational Base + offset (0x01f4)

The second global software reset config value

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	glb_srst_snd_value The second global software reset config value If config 0xe8, it will generate second global software reset.

**CRU\_PLL\_MASK\_CON**

Address: Operational Base + offset (0x01f8)

Register0000 Abstract

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_mask When every bit HIGH, enable the writing corresponding bit When every bit LOW, don't care the writing corresponding bit
15:0	RW	0x5a5a	pll_mask_con Only when pll_mask_con is 0x5a5a, PLL control register can be configured.

**3.7 Timing Diagram**

Power on reset timing is shown as follow:

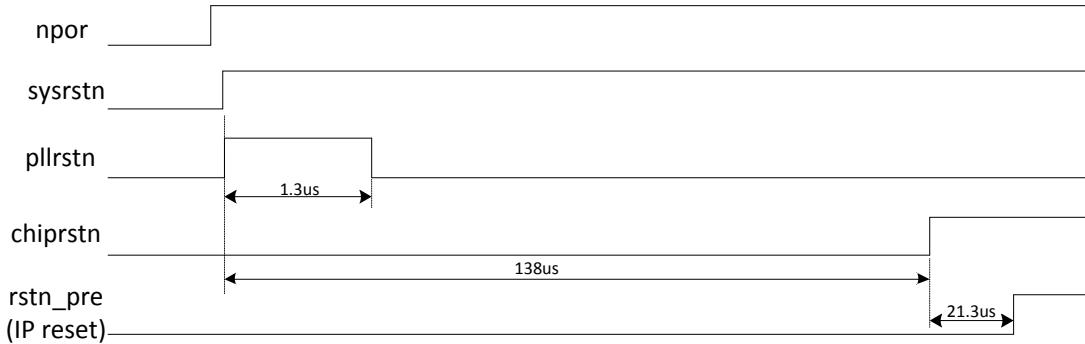


Fig. 3-4 Chip Power On Reset Timing Diagram

Npor is hardware reset signal from out-chip, which is filtered glitch to obtain signal sysrstn. To make PLLs work normally, the PLL reset signal (pllrstn) must maintain high for more than 1us, and PLLs start to lock when pllrstn de-assert, and the PLL max lock time is 1500 PLL REFCLK cycles. And then the system will wait about 138us, and then de-assert reset signal chiprstn. The signal chiprstn is used to generate output clocks in CRU. After CRU start output clocks, the system waits again for 512cycles (21.3us) to de-assert signal rstn\_pre, which is used to generate power on reset of all IPs.

## 3.8 Application Notes

### 3.8.1 PLL usage

#### A. PLL output frequency configuration

FBDIV, POSTDIV1, BYPASS can be configured by programming CRU\_APLL\_CON0, CRU\_DPLL\_CON0 and CRU\_GPLL\_CON0.

DSMPD, REFDIV, POSTDIV2 can be configured by programming CRU\_APLL\_CON1, CRU\_DPLL\_CON1 and CRU\_GPLL\_CON1.

FRAC can be configured by programming CRU\_APLL\_CON2, CRU\_DPLL\_CON2 and CRU\_GPLL\_CON2.

If DSMPD = 1 (DSM is disabled, "integer mode")

$$\text{FOUTVCO} = \text{FREF} / \text{REFDIV} * \text{FBDIV}$$

$$\text{FOUTPOSTDIV} = \text{FOUTVCO} / \text{POSTDIV1} / \text{POSTDIV2}$$

When FREF is 24MHz, and if 700MHz FOUTPOSTDIV is needed. The configuration can be:

$$\begin{aligned} \text{DSMPD} &= 1 \\ \text{REFDIV} &= 6 \\ \text{FBDIV} &= 175 \\ \text{POSTDIV1} &= 1 \\ \text{POSTDIV2} &= 1 \end{aligned}$$

And then

$$\text{FOUTVCO} = \text{FREF} / \text{REFDIV} * \text{FBDIV} = 24/6*175=700$$

$$\text{FOUTPOSTDIV} = \text{FOUTVCO} / \text{POSTDIV1} / \text{POSTDIV2}=700/1/1=700$$

If DSMPD = 0 (DSM is enabled, "fractional mode")

$$\text{FOUTVCO} = \text{FREF} / \text{REFDIV} * (\text{FBDIV} + \text{FRAC} / 224)$$

$$\text{FOUTPOSTDIV} = \text{FOUTVCO} / \text{POSTDIV1} / \text{POSTDIV2}$$

When FREF is 24MHz, and if 491.52MHz FOUTPOSTDIV is needed. The configuration can be:

$$\begin{aligned} \text{DSMPD} &= 0 \\ \text{REFDIV} &= 1 \\ \text{FBDIV} &= 40 \\ \text{FRAC} &= 24'hf5c28f \\ \text{POSTDIV1} &= 2 \\ \text{POSTDIV2} &= 1 \end{aligned}$$

And then

$$\text{FOUTVCO} = \text{FREF} / \text{REFDIV} * (\text{FBDIV} + \text{FRAC} / 224) = 24/1*(40+24'hf5c28f /224)= 983.04$$

$$\text{FOUTPOSTDIV} = \text{FOUTVCO} / \text{POSTDIV1} / \text{POSTDIV2}=983.04/2/1=491.52$$

#### B. PLL setting consideration

- If the POSTDIV value is changed during operation a short pulse (glitch) may occur on FOUTPOSTDIV. The minimum width of the short pulse will be equal to twice the period of

the VCO. Therefore, if the circuitry clocked by the PLL is sensitive to short pulses, the new divide value should be re-timed so that it is synchronous with the rising edge of the output clock (FOUTPOSTDIV). Glitches cannot occur on any of the other outputs.

- For lowest power operation, the minimum VCO and FREF frequencies should be used. For minimum jitter operation, the highest VCO and FREF frequencies should be used. The normal operating range for the VCO is described above in .
- The supply rejection will be worse at the low end of the VCO range so care should be taken to keep the supply clean for low power applications.
- The feedback divider is not capable of dividing by all possible settings due to the use of a power-saving architecture. The following settings are valid for FBDIV:
  - DSMPD=1 (Integer Mode)
  - DSMPD=0 (Fractional Mode)
- The PD input places the PLL into the lowest power mode. In this case, all analog circuits are turned off and FREF will be "ignored". The FOUTPOSTDIV and FOUTVCO pins are forced to logic low (0V).
- The BYPASS pin controls a mux which selects FREF to be passed to the FOUTPOSTDIV when active high. However, the PLL continues to run as it normally would if bypass were low. This is a useful feature for PLL testing since the clock path can be verified without the PLL being required to work. Also, the effect that the PLL induced supply noise has on the output buffering can be evaluated. It is not recommended to switch between BYPASS mode and normal mode for regular chip operation since this may result in a glitch. Also, FOUTPOSTDIVPD should be set low if the PLL is to be used in BYPASS mode.

### 3.8.2 PLL frequency change and lock check

The PLL programming supports changed on-the-fly and the PLL will simply slew to the new frequency.

PLL lock state can be checked in CRU\_APLL\_CON1[10], CRU\_DPLL\_CON1[10], CRU\_CPLL\_CON1[10], CRU\_GPLL\_CON1[10] register. The lock state is high when both original hardware PLL lock and PLL counter lock are high. The PLL counter lock initial value is CRU\_GLB\_CNT\_TH[31:16].

The max delay time is 500 REF\_CLK.

PLL locking consists of three phases.

- Phase 1 is control voltage slewing. During this phase one of the clocks (reference or divide) is much faster than the other, and the PLL frequency adjusts almost continuously. When locking from power down, the divide clock is initially very slow and steadily increases frequency. It will take slightly longer for faster VCO settings when locking from power down, since the PLL must slew further.
- Phase 2 is small signal phase acquisition. During this phase, the internal up/down signals alternate semi-chaotically as the phase slowly adjusts until the two signals are aligned. The duration of this phase depends on the loop bandwidth and is faster with higher bandwidth. Bandwidth can be estimated as FREF / REFDIV / 20 for integer mode and FREF /REFDIV / 40 for fractional mode. The duration of small signal locking is about 1/Bandwidth.
- Phase 3 is the digital cycle count. After the last cycle slip is detected, an internal counter waits 256 FREF / REFDIV cycles before the lock signal goes high. This is frequently the dominant factor in lock time – especially for slower reference clock signals or large reference divide settings. This time can be calculated as 256\*REFDIV/FREF.

### 3.8.3 Fractional divider usage

To get specific frequency, clocks of I2S, SPDIF, UART can be generated by fractional divider. Generally you must set that denominator is 20 times larger than numerator to generate precise clock frequency. So the fractional divider applies only to generate low frequency clock like I2S, UART.

### 3.8.4 Global software reset

Two global software resets are designed in the chip, you can program CRU\_GLB\_SRST\_FST\_VALUE[15:0] as 0xfdb9 to assert the first global software reset glb\_srstn\_1 and program CRU\_GLB\_SRST\_SND\_VALUE[15:0] as 0xecfa8 to assert the second

global software reset glb\_srstn\_2. These two software resets are self-deasserted by hardware.

Glb\_srstn\_1 resets almost all logic.

Glb\_srstn\_2 resets almost all logic except GRF and GPIOs.

Rockchip Confidential

## Chapter 4 Embedded Processor (Cortex-A7)

### 4.1 Overview

The Cortex-A7 MP subsystem of the device is based on the symmetric multiprocessor (SMP) architecture, thus the Dual Cortex-A7 MPU subsystem delivers higher performance and optimal power management, debug and emulation capabilities.

The Cortex-A7 MP subsystem incorporates two Cortex-A7 central processing units (CPUs), level 2(L2) cache shared between the two CPUs, and uses PL310 as L2 cache controller. Each CPU has 32KB of level 1 (L1) instruction cache, 32KB of L1 data cache, separate dedicated power domain, and includes one Neon and Vector Floating Point Unit coprocessors. The Cortex-A7 MP subsystem also includes standard CoreSight components to support SMP debug and emulation, snoop control unit(SCU), interrupt controller (GIC), and clock and reset manager.

The key features of the Cortex-A7 MP subsystem include:

- ARM Coretex-A7 based dual MPU subsystem with SMP architecture
  - Full implements the ARMv7-A architecture profile that includes SIMD and VFP
  - 32KB L1 I-cache and 32KB L1 D-cache per CPU
  - In-order pipeline with direct and indirect branch prediction
  - Harvard Level 1 (L1) memory system with a Memory Management Unit (MMU)
  - SCU ensures memory coherency between the two CPUs
  - Interrupt controller with 128 hardware interrupt inputs
- 128KB L2 cache shared between the two CPUs
  - Fixed line length of 64 bytes
  - Physically indexed and tagged cache
  - 8-way set-associative cache structure
  - Pseudo-random cache replacement policy

### 4.2 Block Diagram

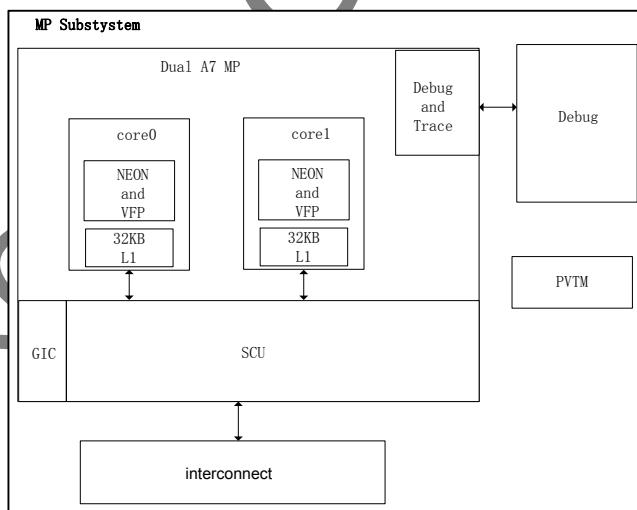


Fig. 4-1 MP Subsystem architecture

### 4.3 Function Description

Please refer to the document Cortex-A7\_MPCore\_Technical\_Reference\_Manual.pdf for the CPU detail description.

### 4.4 Register Description

Please refer to the document Cortex-A7\_MPCore\_Technical\_Reference\_Manual.pdf for the CPU detail description.

## Chapter 5 General Register Files (GRF)

### 5.1 Overview

The general register file will be used to do static set by software, which is composed of many registers for system control.

### 5.2 Function Description

The function of general register file is:

- IOMUX control
- GPIO PAD pulldown and pullup control
- Common system control
- Record the system state

### 5.3 Register Description

#### 5.3.1 Registers Summary

Name	Offset	Size	Reset Value	Description
BENZ_GRF_GPIO0A_IOMUX	0x0000	W	0x00000000	GPIO0A iomux control
BENZ_GRF_GPIO0B_IOMUX	0x0004	W	0x00000000	GPIO0B iomux control
BENZ_GRF_GPIO0C_IOMUX	0x0008	W	0x00000000	GPIO0C iomux control
BENZ_GRF_GPIO0D_IOMUX	0x000c	W	0x00000000	GPIO0D iomux control
BENZ_GRF_GPIO1A_IOMUX	0x0010	W	0x00000000	GPIO1A iomux control
BENZ_GRF_GPIO1B_IOMUX	0x0014	W	0x00000000	GPIO1B iomux control
BENZ_GRF_GPIO1C_IOMUX	0x0018	W	0x00000004	GPIO1C iomux control
BENZ_GRF_GPIO1D_IOMUX	0x001c	W	0x00000000	GPIO1D iomux control
BENZ_GRF_GPIO2A_IOMUX	0x0020	W	0x00000000	GPIO2A iomux control
BENZ_GRF_GPIO2B_IOMUX	0x0024	W	0x00000000	GPIO2B iomux control
BENZ_GRF_GPIO2C_IOMUX	0x0028	W	0x00000000	GPIO2C iomux control
BENZ_GRF_GPIO2D_IOMUX	0x002c	W	0x00000000	GPIO2D iomux control
BENZ_GRF_GPIO3A_IOMUX	0x0030	W	0x00000000	GPIO3A iomux control
BENZ_GRF_GPIO3B_IOMUX	0x0034	W	0x00000000	GPIO3B iomux control
BENZ_GRF_GPIO3C_IOMUX	0x0038	W	0x00000000	GPIO3C iomux control

Name	Offset	Size	Reset Value	Description
BENZ_GRF_GPIO3D_IOMUX	0x003c	W	0x00000000	GPIO3D iomux control
BENZ_GRF_CON_IOMUX	0x0050	W	0x00000000	
BENZ_GRF_GPIO0A_P	0x0100	W	0x00005555	GPIO0A PU/PD control
BENZ_GRF_GPIO0B_P	0x0104	W	0x00009555	GPIO0B PU/PD control
BENZ_GRF_GPIO0C_P	0x0108	W	0x00005556	GPIO0C PU/PD control
BENZ_GRF_GPIO0D_P	0x010c	W	0x0000a9a5	GPIO0D PU/PD control
BENZ_GRF_GPIO1A_P	0x0110	W	0x0000a596	GPIO1A PU/PD control
BENZ_GRF_GPIO1B_P	0x0114	W	0x00006555	GPIO1B PU/PD control
BENZ_GRF_GPIO1C_P	0x0118	W	0x00005556	GPIO1C PU/PD control
BENZ_GRF_GPIO1D_P	0x011c	W	0x00005555	GPIO1D drive strength control
BENZ_GRF_GPIO2A_P	0x0120	W	0x0000595a	GPIO2A drive strength control
BENZ_GRF_GPIO2B_P	0x0124	W	0x0000aaaa	GPIO2B drive strength control
BENZ_GRF_GPIO2C_P	0x0128	W	0x0000aaaa	GPIO2C drive strength control
BENZ_GRF_GPIO2D_P	0x012c	W	0x0000aaaa	GPIO2D drive strength control
BENZ_GRF_GPIO3A_P	0x0130	W	0x0000a556	GPIO3A drive strength control
BENZ_GRF_GPIO3B_P	0x0134	W	0x0000aa81	GPIO3B drive strength control
BENZ_GRF_GPIO3C_P	0x0138	W	0x00006aaa	GPIO3C drive strength control
BENZ_GRF_GPIO3D_P	0x013c	W	0x0000aa54	GPIO3D drive strength control
BENZ_GRF_GPIO0A_E	0x0200	W	0x00005055	GPIO0A drive strength control
BENZ_GRF_GPIO0B_E	0x0204	W	0x00005545	GPIO0B drive strength control
BENZ_GRF_GPIO0C_E	0x0208	W	0x00009155	GPIO0C drive strength control
BENZ_GRF_GPIO0D_E	0x020c	W	0x00005655	GPIO0D drive strength control of N channel
BENZ_GRF_GPIO1A_E	0x0210	W	0x00004555	GPIO1A drive strength control of P channel
BENZ_GRF_GPIO1B_E	0x0214	W	0x00009155	GPIO1B drive strength control of N channel
BENZ_GRF_GPIO1C_E	0x0218	W	0x0000aaa6	GPIO1C drive strength control of P channel
BENZ_GRF_GPIO1D_E	0x021c	W	0x0000aaaa	GPIO1D drive strength control of N channel
BENZ_GRF_GPIO2A_E	0x0220	W	0x0000aaaa	GPIO2A drive strength control
BENZ_GRF_GPIO2B_E	0x0224	W	0x0000aaaa	GPIO2B drive strength control
BENZ_GRF_GPIO2C_E	0x0228	W	0x0000aaaa	GPIO2C drive strength control
BENZ_GRF_GPIO2D_E	0x022c	W	0x0000055a	GPIO2D drive strength control
BENZ_GRF_GPIO3A_E	0x0230	W	0x00005aaa	GPIO3A drive strength control
BENZ_GRF_GPIO3B_E	0x0234	W	0x00009541	GPIO3B drive strength control
BENZ_GRF_GPIO3C_E	0x0238	W	0x00005559	GPIO3C drive strength control
BENZ_GRF_GPIO3D_E	0x023c	W	0x00005554	GPIO3D drive strength control
BENZ_GRF_GPIO0L_SR	0x0300	W	0x00000000	GPIO0A/B SR control
BENZ_GRF_GPIO0H_SR	0x0304	W	0x00000000	GPIO0C/D SR control
BENZ_GRF_GPIO1L_SR	0x0308	W	0x00000000	GPIO1A/B SR control
BENZ_GRF_GPIO1H_SR	0x030c	W	0x00000000	GPIO1C/D SR control

Name	Offset	Size	Reset Value	Description
BENZ_GRF_GPIO2L_SR	0x0310	W	0x00000000	GPIO2A/B SR control
BENZ_GRF_GPIO2H_SR	0x0314	W	0x00000000	GPIO2C/D SR control
BENZ_GRF_GPIO3L_SR	0x0318	W	0x00000000	GPIO3A/B SR control
BENZ_GRF_GPIO3H_SR	0x031c	W	0x00000000	GPIO3C/D SR control
BENZ_GRF_GPIO0L_SMT	0x0380	W	0x00000000	GPIO0A/B smitter control register
BENZ_GRF_GPIO0H_SMT	0x0384	W	0x00000000	GPIO0C/D smitter control register
BENZ_GRF_GPIO1L_SMT	0x0388	W	0x00000000	GPIO1A/B smitter control register
BENZ_GRF_GPIO1H_SMT	0x038c	W	0x00000000	GPIO1C/D smitter control register
BENZ_GRF_GPIO2L_SMT	0x0390	W	0x00000000	GPIO2A/B smitter control register
BENZ_GRF_GPIO2H_SMT	0x0394	W	0x00000000	GPIO2C/D smitter control register
BENZ_GRF_GPIO3L_SMT	0x0398	W	0x00000000	GPIO3A/B smitter control register
BENZ_GRF_GPIO3H_SMT	0x039c	W	0x00000000	GPIO3C/D smitter control register
BENZ_GRF_SOC_CON0	0x0400	W	0x00000100	SoC control register 0
BENZ_GRF_SOC_CON1	0x0404	W	0x00000000	SoC control register 1
BENZ_GRF_SOC_CON2	0x0408	W	0x00000000	SoC control register 2
BENZ_GRF_SOC_CON3	0x040c	W	0x00000000	SoC control register 3
BENZ_GRF_SOC_CON4	0x0410	W	0x0000ffff	SoC control register 4
BENZ_GRF_SOC_CON5	0x0414	W	0x00000003	SoC control register 5
BENZ_GRF_SOC_CON6	0x0418	W	0x00000100	SoC control register 6
BENZ_GRF_SOC_STATUS0	0x0480	W	0x00040000	SoC status register 0
BENZ_GRF_SOC_STATUS1	0x0484	W	0x00000000	SoC status register 1
BENZ_GRF_SOC_STATUS2	0x0488	W	0x063f063f	SoC status register 2
BENZ_GRF_CHIP_ID	0x048c	W	0x00003228	chip_id register
BENZ_GRF_CPU_CON0	0x0500	W	0x00000002	CPU little cluster control register 0
BENZ_GRF_CPU_CON1	0x0504	W	0x00000000	CPU little cluster control register 1
BENZ_GRF_CPU_CON2	0x0508	W	0x00000030	CPU little cluster control register 2
BENZ_GRF_CPU_CON3	0x050c	W	0x00000041	CPU little cluster control register 3
BENZ_GRF_CPU_STATUS0	0x0520	W	0x00000006	CPU status register 0
BENZ_GRF_CPU_STATUS1	0x0524	W	0x000001e0	CPU status register 1
BENZ_GRF_OS_REG0	0x05c8	W	0x00000000	os register 0
BENZ_GRF_OS_REG1	0x05cc	W	0x00000000	os register 1
BENZ_GRF_OS_REG2	0x05d0	W	0x00000000	os register 2
BENZ_GRF_OS_REG3	0x05d4	W	0x00000000	os register 3
BENZ_GRF_OS_REG4	0x05d8	W	0x00000000	os register 4
BENZ_GRF_OS_REG5	0x05dc	W	0x00000000	os register 5
BENZ_GRF_OS_REG6	0x05e0	W	0x00000000	os register 6
BENZ_GRF_OS_REG7	0x05e4	W	0x00000000	os register 7

Name	Offset	Size	Reset Value	Description
BENZ_GRF_DDRC_STAT	0x0604	W	0x00000000	DDRC status register
BENZ_GRF_SIG_DETECT_CON	0x0680	W	0x00000000	External signal detect configue register
BENZ_GRF_SIG_DETECT_CON1	0x0684	W	0x00000000	External signal detect configue register1
BENZ_GRF_SIG_DETECT_STATUS	0x0690	W	0x00000000	External signal detect status register
BENZ_GRF_SIG_DETECT_STATUS1	0x0694	W	0x00000000	External signal detect status register1
BENZ_GRF_SIG_DETECT_CLR	0x06a0	W	0x00000000	External signal detect configue register
BENZ_GRF_SIG_DETECT_CLR1	0x06a4	W	0x00000000	External signal detect configue register1
BENZ_GRF_EMMC_DET	0x06b0	W	0x00000cff	emmc detect register
BENZ_GRF_HOST0_CON0	0x0700	W	0x00000820	host0 control register 0
BENZ_GRF_HOST0_CON1	0x0704	W	0x000004bc	HOST0 control register 1
BENZ_GRF_HOST0_CON2	0x0708	W	0x00000019	HOST0 control register 2
BENZ_GRF_HOST1_CON0	0x0710	W	0x00000820	HOST1 control register 0
BENZ_GRF_HOST1_CON1	0x0714	W	0x000004bc	HOST1 control register 1
BENZ_GRF_HOST1_CON2	0x0718	W	0x00000019	HOST1 control register 2
BENZ_GRF_HOST2_CON0	0x0720	W	0x00000820	HOST2 control register 0
BENZ_GRF_HOST2_CON1	0x0724	W	0x000004bc	HOST2 control register 1
BENZ_GRF_HOST2_CON2	0x0728	W	0x00000019	HOST2 control register 2
BENZ_GRF_USBPHY0_CO_N0	0x0760	W	0x00000000	USB PHY0 control register 0
BENZ_GRF_USBPHY0_CO_N1	0x0764	W	0x00000000	USB PHY0 control register 1
BENZ_GRF_USBPHY0_CO_N2	0x0768	W	0x00000000	USB PHY0 control register 2
BENZ_GRF_USBPHY0_CO_N3	0x076c	W	0x00008518	USB PHY0 control register 3
BENZ_GRF_USBPHY0_CO_N4	0x0770	W	0x0000e007	USB PHY0 control register 4
BENZ_GRF_USBPHY0_CO_N5	0x0774	W	0x000002e7	USB PHY0 control register 5
BENZ_GRF_USBPHY0_CO_N6	0x0778	W	0x00000200	USB PHY0 control register 6
BENZ_GRF_USBPHY0_CO_N7	0x077c	W	0x00005556	USB PHY0 control register 7
BENZ_GRF_USBPHY0_CO_N8	0x0780	W	0x00004555	USB PHY0 control register 8
BENZ_GRF_USBPHY0_CO_N9	0x0784	W	0x00000005	USB PHY0 control register 9

Name	Offset	Size	Reset Value	Description
BENZ_GRF_USBPHY0_CO_N10	0x0788	W	0x000068c0	USB PHY0 control register 10
BENZ_GRF_USBPHY0_CO_N11	0x078c	W	0x00000000	USB PHY0 control register 11
BENZ_GRF_USBPHY0_CO_N12	0x0790	W	0x00000000	USB PHY0 control register 12
BENZ_GRF_USBPHY0_CO_N13	0x0794	W	0x00000000	USB PHY0 control register 13
BENZ_GRF_USBPHY0_CO_N14	0x0798	W	0x00000000	USB PHY0 control register 14
BENZ_GRF_USBPHY0_CO_N15	0x079c	W	0x00008518	USB PHY0 control register 15
BENZ_GRF_USBPHY0_CO_N16	0x07a0	W	0x0000e007	USB PHY0 control register 16
BENZ_GRF_USBPHY0_CO_N17	0x07a4	W	0x000002e7	USB PHY0 control register 17
BENZ_GRF_USBPHY0_CO_N18	0x07a8	W	0x00000200	USB PHY0 control register 18
BENZ_GRF_USBPHY0_CO_N19	0x07ac	W	0x00005556	USB PHY0 control register 19
BENZ_GRF_USBPHY0_CO_N20	0x07b0	W	0x00004555	USB PHY0 control register 20
BENZ_GRF_USBPHY0_CO_N21	0x07b4	W	0x00000005	USB PHY0 control register 21
BENZ_GRF_USBPHY0_CO_N22	0x07b8	W	0x000068c0	USB PHY0 control register 22
BENZ_GRF_USBPHY0_CO_N23	0x07bc	W	0x00000000	USB PHY0 control register 23
BENZ_GRF_USBPHY0_CO_N24	0x07c0	W	0x00000000	USB PHY0 control register 24
BENZ_GRF_USBPHY0_CO_N25	0x07c4	W	0x00000000	USB PHY0 control register 25
BENZ_GRF_USBPHY0_CO_N26	0x07c8	W	0x00000000	USB PHY0 control register 26
BENZ_GRF_USBPHY1_CO_N0	0x0800	W	0x00000000	USB PHY1 control register 0
BENZ_GRF_USBPHY1_CO_N1	0x0804	W	0x00000000	USB PHY1 control register 1
BENZ_GRF_USBPHY1_CO_N2	0x0808	W	0x00000000	USB PHY1 control register 2
BENZ_GRF_USBPHY1_CO_N3	0x080c	W	0x00008518	USB PHY1 control register 3
BENZ_GRF_USBPHY1_CO_N4	0x0810	W	0x0000e007	USB PHY1 control register 4

Name	Offset	Size	Reset Value	Description
BENZ_GRF_USBPHY1_CO_N5	0x0814	W	0x000002e7	USB PHY1 control register 5
BENZ_GRF_USBPHY1_CO_N6	0x0818	W	0x00000200	USB PHY1 control register 6
BENZ_GRF_USBPHY1_CO_N7	0x081c	W	0x00005556	USB PHY1 control register 7
BENZ_GRF_USBPHY1_CO_N8	0x0820	W	0x00004555	USB PHY1 control register 8
BENZ_GRF_USBPHY1_CO_N9	0x0824	W	0x00000005	USB PHY1 control register 9
BENZ_GRF_USBPHY1_CO_N10	0x0828	W	0x000068c0	USB PHY1 control register 10
BENZ_GRF_USBPHY1_CO_N11	0x082c	W	0x00000000	USB PHY1 control register 11
BENZ_GRF_USBPHY1_CO_N12	0x0830	W	0x00000000	USB PHY1 control register 12
BENZ_GRF_USBPHY1_CO_N13	0x0834	W	0x00000000	USB PHY1 control register 13
BENZ_GRF_USBPHY1_CO_N14	0x0838	W	0x00000000	USB PHY1 control register 14
BENZ_GRF_USBPHY1_CO_N15	0x083c	W	0x00008518	USB PHY1 control register 15
BENZ_GRF_USBPHY1_CO_N16	0x0840	W	0x0000e007	USB PHY1 control register 16
BENZ_GRF_USBPHY1_CO_N17	0x0844	W	0x000002e7	USB PHY1 control register 17
BENZ_GRF_USBPHY1_CO_N18	0x0848	W	0x00000200	USB PHY1 control register 18
BENZ_GRF_USBPHY1_CO_N19	0x084c	W	0x00005556	USB PHY1 control register 19
BENZ_GRF_USBPHY1_CO_N20	0x0850	W	0x00004555	USB PHY1 control register 20
BENZ_GRF_USBPHY1_CO_N21	0x0854	W	0x00000005	USB PHY1 control register 21
BENZ_GRF_USBPHY1_CO_N22	0x0858	W	0x000068c0	USB PHY1 control register 22
BENZ_GRF_USBPHY1_CO_N23	0x085c	W	0x00000000	USB PHY1 control register 23
BENZ_GRF_USBPHY1_CO_N24	0x0860	W	0x00000000	USB PHY1 control register 24
BENZ_GRF_USBPHY1_CO_N25	0x0864	W	0x00000000	USB PHY1 control register 25
BENZ_GRF_USBPHY1_CO_N26	0x0868	W	0x00000000	USB PHY1 control register 26
BENZ_GRF_OTG_CON0	0x0880	W	0x00000000	OTG control register 0

Name	Offset	Size	Reset Value	Description
BENZ_GRF_UOC_STATUS_0	0x0884	W	0x00000000	USBPHY status register 0
BENZ_GRF_MAC_CON0	0x0900	W	0x00000000	MAC control register 0
BENZ_GRF_MAC_CON1	0x0904	W	0x00000000	MAC control register 1
BENZ_GRF_MACPHY_CON0	0x0b00	W	0x00002039	MACPHY control register 0
BENZ_GRF_MACPHY_CON1	0x0b04	W	0x00000000	MACPHY control register 1
BENZ_GRF_MACPHY_CON2	0x0b08	W	0x00000000	MACPHY control register 2
BENZ_GRF_MACPHY_CON3	0x0b0c	W	0x00000000	MACPHY control register 3
BENZ_GRF_MACPHY_STATUS	0x0b10	W	0x00000000	MACPHY status register

Notes:Size: **B**- Byte (8 bits) access, **HW**- Half WORD (16 bits) access, **W**-WORD (32 bits) access

### 5.3.2 Detail Register Description

#### BENZ\_GRF\_GPIO0A\_IOMUX

Address: Operational Base + offset (0x0000)  
GPIO0A iomux control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; .... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RW	0x0	gpio0a7_sel GPIO0A[7] iomux select 2'b00: gpio 2'b01: i2c3_sda 2'b10: hdmi_ddcsda 2'b11: reserved
13:12	RW	0x0	gpio0a6_sel GPIO0A[6] iomux select 2'b00: gpio 2'b01: i2c3_scl 2'b10: hdmi_ddcscl 2'b11: reserved

Bit	Attr	Reset Value	Description
11:10	RW	0x0	gpio0a5_sel GPIO0A[5] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
9:8	RW	0x0	gpio0a4_sel GPIO0A[4] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
7:6	RW	0x0	gpio0a3_sel GPIO0A[3] iomux select 2'b00: gpio 2'b01: i2c1_sda 2'b10: sdio_cmd 2'b11: reserved
5:4	RW	0x0	gpio0a2_sel GPIO0A[2] iomux select 2'b00: gpio 2'b01: i2c1_scl 2'b10: reserved 2'b11: reserved
3:2	RW	0x0	gpio0a1_sel GPIO0A[1] iomux select 2'b00: gpio 2'b01: i2c0_sda 2'b10: reserved 2'b11: reserved
1:0	RW	0x0	gpio0a0_sel GPIO0A[0] iomux select 2'b00: gpio 2'b01: i2c0_scl 2'b10: reserved 2'b11: reserved

**BENZ\_GRF\_GPIO0B\_IOMUX**

Address: Operational Base + offset (0x0004)

GPIO0B iomux control

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:14	RW	0x0	<p>gpio0b7_sel GPIO0B[7] iomux select 2'b00: gpio 2'b01: hdmi_hpd 1'b10: reserved 1'b11: reserved</p>
13:12	RW	0x0	<p>gpio0b6_sel GPIO0B[6] iomux select 2'b00: gpio 2'b01: i2s_sdi 2'b10: spi_csn0 2'b11: reserved</p>
11:10	RW	0x0	<p>gpio0b5_sel GPIO0B[5] iomux select 2'b00: gpio 2'b01: i2s_sdo 2'b10: spi_rxd 2'b11: reserved</p>
9:8	RW	0x0	<p>gpio0b4_sel GPIO0B[4] iomux select 2'b00: gpio 2'b01: i2s_lrcktx 2'b10: reserved 2'b11: reserved</p>
7:6	RW	0x0	<p>gpio0b3_sel GPIO0B[3] iomux select 2'b00: gpio 2'b01: i2s_lrckrx 2'b10: spi_txd 2'b11: reserved</p>
5:4	RW	0x0	<p>gpio0b2_sel GPIO0B[2] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved</p>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
3:2	RW	0x0	gpio0b1_sel GPIO0B[1] iomux select 2'b00: gpio 2'b01: i2s_sclk 2'b10: spi_clk 2'b11: reserved
1:0	RW	0x0	gpio0b0_sel GPIO0B[0] iomux select 2'b00: gpio 2'b01: i2s_mclk 2'b10: reserved 2'b11: reserved

**BENZ\_GRF\_GPIO0C\_IOMUX**

Address: Operational Base + offset (0x0008)

GPIO0C iomux control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RW	0x0	gpio0c7_sel GPIO0C[7] iomux select 2'b00: gpio 2'b01: nand_cs1 2'b10: spi1_clk 2'b11: reserved
13:12	RW	0x0	gpio0c6_sel GPIO0C[6] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
11:10	RW	0x0	gpio0c5_sel GPIO0C[5] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved

Bit	Attr	Reset Value	Description
9:8	RW	0x0	gpio0c4_sel GPIO0C[4] iomux select 2'b00: gpio 2'b01: hdmi_cecsda 2'b10: reserved 2'b11: reserved
7:6	RW	0x0	gpio0c3_sel GPIO0C[3] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
5:4	RW	0x0	gpio0c2_sel GPIO0C[2] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
3:2	RW	0x0	gpio0c1_sel GPIO0C[1] iomux select 2'b00: gpio 2'b01: uart0_rstn 2'b10: clk_out1 2'b11: reserved
1:0	RW	0x0	gpio0c0_sel GPIO0C[0] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved

**BENZ\_GRF\_GPIOOD\_IOMUX**

Address: Operational Base + offset (0x000c)  
 GPIOOD iomux control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:14	RW	0x0	gpio0d7_sel GPIO0D[7] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
13:12	RW	0x0	gpio0d6_sel GPIO0D[6] iomux select 2'b00: gpio 2'b01: sdio_pwren 2'b10: pwm1_1 2'b11: reserved
11:10	RW	0x0	gpio0d5_sel GPIO0D[5] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
9:8	RW	0x0	gpio0d4_sel GPIO0D[4] iomux select 2'b00: gpio 2'b01: pwm_2 2'b10: reserved 2'b11: reserved
7:6	RW	0x0	gpio0d3_sel GPIO0D[3] iomux select 2'b00: gpio 2'b01: pwm_1 2'b10: pcm_tx 2'b11: reserved
5:4	RW	0x0	gpio0d2_sel GPIO0D[2] iomux select 2'b00: gpio 2'b01: pwn_0 2'b10: pcm_rx 2'b11: reserved
3:2	RW	0x0	gpio0d1_sel GPIO0D[1] iomux select 2'b00: gpio 2'b01: uart2_cstn 2'b10: reserved 2'b11: reserved

Bit	Attr	Reset Value	Description
1:0	RW	0x0	gpio0d0_sel GPIO0D[0] iomux select 2'b00: gpio 2'b01: uart2_rstn 2'b10: tsadc_shut 2'b11: reserved

**BENZ\_GRF\_GPIO1A\_IOMUX**

Address: Operational Base + offset (0x0010)

GPIO1A iomux control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RW	0x0	gpio1a7_sel GPIO1A[7] iomux select 2'b00: gpio 2'b01: sdmmc_wrprt 2'b10: reserved 2'b11: reserved
13:12	RW	0x0	gpio1a6_sel GPIO1A[6] iomux select 2'b00: reserved 2'b01: reserved 2'b10: reserved 2'b11: reserved
11:10	RW	0x0	gpio1a5_sel GPIO1A[5] iomux select 2'b00: gpio 2'b01: sdio_d3 2'b10: i2s_sdio3 2'b11: reserved
9:8	RW	0x0	gpio1a4_sel GPIO1A[4] iomux select 2'b00: gpio 2'b01: sdio_d2 2'b10: i2s_sdio2 2'b11: reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
7:6	RW	0x0	gpio1a3_sel GPIO1A[3] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
5:4	RW	0x0	gpio1a2_sel GPIO1A[2] iomux select 2'b00: gpio 2'b01: sdio_d1 2'b10: i2s_sdio1 2'b11: reserved
3:2	RW	0x0	gpio1a1_sel GPIO1A[1] iomux select 2'b00: gpio 2'b01: sdio_d0 2'b10: reserved 2'b11: reserved
1:0	RW	0x0	gpio1a0_sel GPIO1A[0] iomux select 2'b00: gpio 2'b01: sdio_clkout 2'b10: reserved 2'b11: reserved

**BENZ\_GRF\_GPIO1B\_IOMUX**

Address: Operational Base + offset (0x0014)

GPIO1B iomux control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RW	0x0	gpio1b7_sel GPIO1B[7] iomux select 2'b00: gpio 2'b01: sdmmc_cmd 2'b10: reserved 2'b11: reserved

Bit	Attr	Reset Value	Description
13:12	RW	0x0	gpio1b6_sel GPIO1B[6] iomux select 2'b00: gpio 2'b01: sdmmc_pwren 2'b10: reserved 2'b11: reserved
11:10	RW	0x0	gpio1b5_sel GPIO1B[5] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
9:8	RW	0x0	gpio1b4_sel GPIO1B[4] iomux select 2'b00: gpio 2'b01: spi_csn1 2'b10: pwm1_2 2'b11: reserved
7:6	RW	0x0	gpio1b3_sel GPIO1B[3] iomux select 2'b00: gpio 2'b01: uart1_rstn 2'b10: pwm1_ir 2'b11: reserved
5:4	RW	0x0	gpio1b2_sel GPIO1B[2] iomux select 2'b00: gpio 2'b01: uart1_sin 2'b10: uart21_sin 2'b11: reserved
3:2	RW	0x0	gpio1b1_sel GPIO1B[1] iomux select 2'b00: gpio 2'b01: uart1_sout 2'b10: uart21_sout 2'b11: spi2_txd
1:0	RW	0x0	gpio1b0_sel GPIO1B[0] iomux select 2'b00: gpio 2'b01: uart1_ctsn 2'b10: clkout_32k 2'b11: reserved

**BENZ\_GRF\_GPIO1C\_IOMUX**

Address: Operational Base + offset (0x0018)

## GPIO1C iomux control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:14	RW	0x0	<p>gpio1c7_sel GPIO1C[7] iomux select 2'b00: gpio 2'b01: nand_cs3 2'b10: emmc_rstnout 2'b11: reserved</p>
13:12	RW	0x0	<p>gpio1c6_sel GPIO1C[6] iomux select 2'b00: gpio 2'b01: nand_cs2 2'b10: emmc_cmd 2'b11: reserved</p>
11:10	RW	0x0	<p>gpio1c5_sel GPIO1C[5] iomux select 2'b00: gpio 2'b01: sdmmc_d3 2'b10: jtag_tms 2'b11: reserved</p>
9:8	RW	0x0	<p>gpio1c4_sel GPIO1C[4] iomux select 2'b00: gpio 2'b01: sdmmc_d2 2'b10: jtag_tck 2'b11: reserved</p>
7:6	RW	0x0	<p>gpio1c3_sel GPIO1C[3] iomux select 2'b00: gpio 2'b01: sdmmc_d1 2'b10: uart2_sin 2'b11: reserved</p>
5:4	RW	0x0	<p>gpio1c2_sel GPIO1C[2] iomux select 2'b00: gpio 2'b01: sdmmc_d0 2'b10: uart2_sout 2'b11: reserved</p>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
3:2	RW	0x1	gpio1c1_sel GPIO1C[1] iomux select 2'b00: gpio 2'b01: sdmmc_detn 2'b10: reserved 2'b11: reserved
1:0	RW	0x0	gpio1c0_sel GPIO1C[0] iomux select 2'b00: gpio 2'b01: sdmmc_clkout 2'b10: reserved 2'b11: reserved

**BENZ\_GRF\_GPIO1D\_IOMUX**

Address: Operational Base + offset (0x001c)

GPIO1D iomux control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RW	0x0	gpio1d7_sel GPIO1D[7] iomux select 2'b00: gpio 2'b01: nand_d7 2'b10: emmc_d7 2'b11: reserved
13:12	RW	0x0	gpio1d6_sel GPIO1D[6] iomux select 2'b00: gpio 2'b01: nand_d6 2'b10: emmc_d6 2'b11: reserved
11:10	RW	0x0	gpio1d5_sel GPIO1D[5] iomux select 2'b00: gpio 2'b01: nand_d5 2'b10: emmc_d5 2'b11: reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
9:8	RW	0x0	gpio1d4_sel GPIO1D[4] iomux select 2'b00: gpio 2'b01: nand_d4 2'b10: emmc_d4 2'b11: reserved
7:6	RW	0x0	gpio1d3_sel GPIO1D[3] iomux select 2'b00: gpio 2'b01: nand_d3 2'b10: emmc_d3 2'b11: reserved
5:4	RW	0x0	gpio1d2_sel GPIO1D[2] iomux select 2'b00: gpio 2'b01: nand_d2 2'b10: emmc_d2 2'b11: reserved
3:2	RW	0x0	gpio1d1_sel GPIO1D[1] iomux select 2'b00: gpio 2'b01: nand_d1 2'b10: emmc_d1 2'b11: reserved
1:0	RW	0x0	gpio1d0_sel GPIO1D[0] iomux select 2'b00: gpio 2'b01: nand_d0 2'b10: emmc_d0 2'b11: reserved

**BENZ\_GRF\_GPIO2A\_IOMUX**

Address: Operational Base + offset (0x0020)

GPIO2A iomux control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:14	RW	0x0	gpio2a7_sel GPIO2A[7] iomux select 2'b00: gpio 2'b01: nand_dqs 2'b10: emmc_clkout 2'b11: reserved
13:12	RW	0x0	gpio2a6_sel GPIO2A[6] iomux select 2'b00: reserved 2'b01: nand_cs0 2'b10: reserved 2'b11: reserved
11:10	RW	0x0	gpio2a5_sel GPIO2A[5] iomux select 2'b00: gpio 2'b01: nand_wp 2'b10: emmc_pwren 2'b11: reserved
9:8	RW	0x0	gpio2a4_sel GPIO2A[4] iomux select 2'b00: gpio 2'b01: nand_rdy 2'b10: emmc1_cmd 2'b11: reserved
7:6	RW	0x0	gpio2a3_sel GPIO2A[3] iomux select 2'b00: gpio 2'b01: nand_rdn 2'b10: spi1_csn1 2'b11: reserved
5:4	RW	0x0	gpio2a2_sel GPIO2A[2] iomux select 2'b00: gpio 2'b01: nand_wrn 2'b10: spi1_csn0 2'b11: reserved
3:2	RW	0x0	gpio2a1_sel GPIO2A[1] iomux select 2'b00: gpio 2'b01: nand_cle 2'b10: spi1_txd 2'b11: reserved

Bit	Attr	Reset Value	Description
1:0	RW	0x0	gpio2a0_sel GPIO2A[0] iomux select 2'b00: gpio 2'b01: nand_ale 2'b10: spi1_rxd 2'b11: reserved

**BENZ\_GRF\_GPIO2B\_IOMUX**

Address: Operational Base + offset (0x0024)

GPIO2B iomux control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RW	0x0	gpio2b7_sel GPIO2B[7] iomux select 2'b00: gpio 2'b01: gmac_rxer 1'b10: ts_d6 1'b11: reserved
13:12	RW	0x0	gpio2b6_sel GPIO2B[6] iomux select 2'b00: gpio 2'b01: gmac_clk 2'b10: mac_link 2'b11: reserved
11:10	RW	0x0	gpio2b5_sel GPIO2B[5] iomux select 2'b00: gpio 2'b01: gmac_txen 2'b10: ts_d7 2'b11: reserved
9:8	RW	0x0	gpio2b4_sel GPIO2B[4] iomux select 2'b00: gpio 2'b01: gmac_mdio 2'b10: ts_sync 2'b11: reserved

Bit	Attr	Reset Value	Description
7:6	RW	0x0	gpio2b3_sel GPIO2B[3] iomux select 2'b00: gpio 2'b01: gmac_rxclk 2'b10: ts_clk 2'b11: reserved
5:4	RW	0x0	gpio2b2_sel GPIO2B[2] iomux select 2'b00: gpio 2'b01: gmac_crs 2'b10: ts_fail 2'b11: reserved
3:2	RW	0x0	gpio2b1_sel GPIO2B[1] iomux select 2'b00: gpio 2'b01: gmac_txclk 2'b10: ts_valid 2'b11: reserved
1:0	RW	0x0	gpio2b0_sel GPIO2B[0] iomux select 2'b00: gpio 2'b01: gmac_rxdv 2'b10: mac_speed_iout 2'b11: reserved

**BENZ\_GRF\_GPIO2C\_IOMUX**

Address: Operational Base + offset (0x0028)

GPIO2C iomux control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RW	0x0	gpio2c7_sel GPIO2C[7] iomux select 2'b00: gpio 2'b01: gmac_txd3 2'b10: card_io 2'b11: reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
13:12	RW	0x0	gpio2c6_sel GPIO2C[6] iomux select 2'b00: gpio 2'b01: gmac_txd2 2'b10: card_det 2'b11: reserved
11:10	RW	0x0	gpio2c5_sel GPIO2C[5] iomux select 2'b00: gpio 2'b01: i2c2_scl 2'b10: gmac_rxd2 2'b11: card_rst
9:8	RW	0x0	gpio2c4_sel GPIO2C[4] iomux select 2'b00: gpio 2'b01: i2c2_sda 2'b10: gmac_rxd3 2'b11: reserved
7:6	RW	0x0	gpio2c3_sel GPIO2C[3] iomux select 2'b00: gpio 2'b01: gmac_txd0 2'b10: ts_d0 2'b11: reserved
5:4	RW	0x0	gpio2c2_sel GPIO2C[2] iomux select 2'b00: gpio 2'b01: gmac_txd1 2'b10: ts_d1 2'b11: reserved
3:2	RW	0x0	gpio2c1_sel GPIO2C[1] iomux select 2'b00: gpio 2'b01: gmac_rxd0 2'b10: ts_d2 2'b11: reserved
1:0	RW	0x0	gpio2c0_sel GPIO2C[0] iomux select 2'b00: gpio 2'b01: gmac_rxd1 2'b10: ts_d3 2'b11: reserved

**BENZ\_GRF\_GPIO2D\_IOMUX**

Address: Operational Base + offset (0x002c)

## GPIO2D iomux control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:14	RW	0x0	<p>gpio2d7_sel</p> <p>GPIO2d[7] iomux select</p> <p>2'b00: gpio</p> <p>2'b01: reserved</p> <p>2'b10: reserved</p> <p>2'b11: reserved</p>
13:12	RW	0x0	<p>gpio2d6_sel</p> <p>GPIO2d[6] iomux select</p> <p>2'b00: gpio</p> <p>2'b01: reserved</p> <p>2'b10: reserved</p> <p>2'b11: reserved</p>
11:10	RW	0x0	<p>gpio2d5_sel</p> <p>GPIO2d[5] iomux select</p> <p>2'b00: gpio</p> <p>2'b01: uart0_ctsn</p> <p>2'b10: reserved</p> <p>2'b11: reserved</p>
9:8	RW	0x0	<p>gpio2d4_sel</p> <p>GPIO2d[4] iomux select</p> <p>2'b00: gpio</p> <p>2'b01: reserved</p> <p>2'b10: reserved</p> <p>2'b11: reserved</p>
7:6	RW	0x0	<p>gpio2d3_sel</p> <p>GPIO2d[3] iomux select</p> <p>2'b00: gpio</p> <p>2'b01: uart0_sin</p> <p>2'b10: reserved</p> <p>2'b11: reserved</p>
5:4	RW	0x0	<p>gpio2d2_sel</p> <p>GPIO2d[2] iomux select</p> <p>2'b00: gpio</p> <p>2'b01: uart0_sout</p> <p>2'b10: reserved</p> <p>2'b11: reserved</p>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
3:2	RW	0x0	gpio2d1_sel GPIO2d[1] iomux select 2'b00: gpio 2'b01: gmac_mdc 2'b10: reserved 2'b11: reserved
1:0	RW	0x0	gpio2d0_sel GPIO2d[0] iomux select 2'b00: gpio 2'b01: gmac_col 2'b10: ts_d5 2'b11: reserved

**BENZ\_GRF\_GPIO3A\_IOMUX**

Address: Operational Base + offset (0x0030)

GPIO3A iomux control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RW	0x0	gpio3a7_sel GPIO3A[7] iomux select 2'b00: gpio 2'b01: uart11_ctsn 2'b10: reserved 2'b11: reserved
13:12	RW	0x0	gpio3a6_sel GPIO3A[6] iomux select 2'b00: gpio 2'b01: uart11_rtsn 2'b10: reserved 2'b11: reserved
11:10	RW	0x0	gpio3a5_sel GPIO3A[5] iomux select 2'b00: gpio 2'b01: sdio1_d3 2'b10: reserved 2'b11: reserved

Bit	Attr	Reset Value	Description
9:8	RW	0x0	gpio3a4_sel GPIO3A[4] iomux select 2'b00: gpio 2'b01: sdio1_d2 2'b10: reserved 2'b11: reserved
7:6	RW	0x0	gpio3a3_sel GPIO3A[3] iomux select 2'b00: gpio 2'b01: sdio1_d1 2'b10: reserved 2'b11: reserved
5:4	RW	0x0	gpio3a2_sel GPIO2A[2] iomux select 2'b00: gpio 2'b01: sdio1_d0 2'b10: reserved 2'b11: reserved
3:2	RW	0x0	gpio3a1_sel GPIO3A[1] iomux select 2'b00: gpio 2'b01: sdio1_cmd 2'b10: reserved 2'b11: reserved
1:0	RW	0x0	gpio3a0_sel GPIO3A[0] iomux select 2'b00: gpio 2'b01: sdio1_clk 2'b10: reserved 2'b11: reserved

**BENZ\_GRF\_GPIO3B\_IOMUX**

Address: Operational Base + offset (0x0034)

GPIO3B iomux control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:14	RW	0x0	gpio3b7_sel GPIO3B[7] iomux select 2'b00: gpio 2'b01: clk_out0 2'b10: reserved 2'b11: reserved
13:12	RW	0x0	gpio3b6_sel GPIO3B[6] iomux select 2'b00: gpio 2'b01: uart11_sout 2'b10: reserved 2'b11: reserved
11:10	RW	0x0	gpio3b5_sel GPIO3B[5] iomux select 2'b00: gpio 2'b01: uart11_sin 2'b10: reserved 2'b11: reserved
9:8	RW	0x0	gpio3b4_sel GPIO3B[4] iomux select 2'b00: gpio 2'b01: pcm_sync 2'b10: reserved 2'b11: reserved
7:6	RW	0x0	gpio3b3_sel GPIO3B[3] iomux select 2'b00: gpio 2'b01: pcm_clk 2'b10: reserved 2'b11: reserved
5:4	RW	0x0	gpio3b2_sel GPIO3B[2] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
3:2	RW	0x0	gpio3b1_sel GPIO3B[1] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
1:0	RW	0x0	gpio3b0_sel GPIO3B[0] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved

**BENZ\_GRF\_GPIO3C\_IOMUX**

Address: Operational Base + offset (0x0038)

GPIO3C iomux control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RW	0x0	gpio3c7_sel GPIO3C[7] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
13:12	RW	0x0	gpio3c6_sel GPIO3C[6] iomux select 2'b00: gpio 2'b01: drive_vbus1 2'b10: reserved 2'b11: reserved
11:10	RW	0x0	gpio3c5_sel GPIO3C[5] iomux select 2'b00: gpio 2'b01: pwm1_0 2'b10: reserved 2'b11: reserved
9:8	RW	0x0	gpio3c4_sel GPIO3C[4] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
7:6	RW	0x0	gpio3c3_sel GPIO3C[3] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
5:4	RW	0x0	gpio3c2_sel GPIO3C[2] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
3:2	RW	0x0	gpio3c1_sel GPIO3C[1] iomux select 2'b00: gpio 2'b01: drive_vbus 2'b10: reserved 2'b11: reserved
1:0	RW	0x0	gpio3c0_sel GPIO3C[0] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved

**BENZ\_GRF\_GPIO3D\_IOMUX**

Address: Operational Base + offset (0x003c)

GPIO3D iomux control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RW	0x0	gpio3d7_sel GPIO3d[7] iomux select 2'b00: gpio 2'b01: testclk_out1 2'b10: spdif1_tx 2'b11: reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
13:12	RW	0x0	gpio3d6_sel GPIO3d[6] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
11:10	RW	0x0	gpio3d5_sel GPIO3d[5] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
9:8	RW	0x0	gpio3d4_sel GPIO3d[4] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
7:6	RW	0x0	gpio3d3_sel GPIO3d[3] iomux select 2'b00: gpio 2'b01: spdif_tx 2'b10: reserved 2'b11: reserved
5:4	RW	0x0	gpio3d2_sel GPIO3d[2] iomux select 2'b00: gpio 2'b01: pwm_ir 2'b10: reserved 2'b11: reserved
3:2	RW	0x0	gpio3d1_sel GPIO3d[1] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved
1:0	RW	0x0	gpio3d0_sel GPIO3d[0] iomux select 2'b00: gpio 2'b01: reserved 2'b10: reserved 2'b11: reserved

**BENZ\_GRF\_CON\_IOMUX**

Address: Operational Base + offset (0x0050)

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15	RW	0x0	grf_con_iomux_gmac
14	RW	0x0	i2s_sdio3_oe
13	RW	0x0	i2s_sdio2_oe
12	RW	0x0	i2s_sdio1_oe
11	RW	0x0	grf_con_iomux_uart1sel
10	RW	0x0	grf_con_iomux_uartdbgena
9	RW	0x0	grf_con_iomux_uartdbgsel
8	RW	0x0	grf_con_iomux_uart2sel
7	RW	0x0	grf_con_iomux_emmcsel
6	RW	0x0	grf_con_iomux_i2sacodecsel
5	RW	0x0	grf_con_iomux_spisel
4	RW	0x0	grf_con_iomux_sdiosel
3	RW	0x0	grf_con_iomux_pwm3sel
2	RW	0x0	grf_con_iomux_pwm2sel
1	RW	0x0	grf_con_iomux_pwm1sel
0	RW	0x0	grf_con_iomux_pwm0sel

**BENZ\_GRF\_GPIO0A\_P**

Address: Operational Base + offset (0x0100)

GPIO0A PU/PD control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:0	RW	0x5555	gpio0a_p GPIO0A PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO0B\_P**

Address: Operational Base + offset (0x0104)

GPIO0B PU/PD control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x9555	gpio0b_p GPIO0B PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO0C\_P**

Address: Operational Base + offset (0x0108)

GPIO0C PU/PD control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:0	RW	0x5556	gpio0c_p GPIO0C PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO0D\_P**

Address: Operational Base + offset (0x010c)

GPIO0D PU/PD control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0xa9a5	gpio0d_p GPIO0D PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO1A\_P**

Address: Operational Base + offset (0x0110)

GPIO1A PU/PD control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:0	RW	0xa596	gpio1a_p GPIO1A PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO1B\_P**

Address: Operational Base + offset (0x0114)

GPIO1B PU/PD control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x6555	gpio1b_p GPIO1B PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO1C\_P**

Address: Operational Base + offset (0x0118)

GPIO1C PU/PD control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:0	RW	0x5556	gpio1c_p GPIO1C PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO1D\_P**

Address: Operational Base + offset (0x011c)

GPIO1D drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x5555	gpio1d_p GPIO1D PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO2A\_P**

Address: Operational Base + offset (0x0120)

GPIO2A drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:0	RW	0x595a	gpio2a_p GPIO2A PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO2B\_P**

Address: Operational Base + offset (0x0124)

GPIO2B drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0xaaaa	gpio2b_p GPIO2B PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO2C\_P**

Address: Operational Base + offset (0x0128)

GPIO2C drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:0	RW	0xaaaa	gpio2c_p GPIO2C PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO2D\_P**

Address: Operational Base + offset (0x012c)

GPIO2D drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0xaaaa	gpio2d_p GPIO2D PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO3A\_P**

Address: Operational Base + offset (0x0130)

GPIO3A drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:0	RW	0xa556	<p>gpio3a_p</p> <p>GPIO3A PU/PD programmation section, every GPIO bit corresponding to 2bits</p> <p>2'b00: Z(Noraml operaton);</p> <p>2'b01: weak 1(pull-up);</p> <p>2'b10: weak 0(pull-down);</p> <p>2'b11: Repeater(Bus keeper)</p>

**BENZ\_GRF\_GPIO3B\_P**

Address: Operational Base + offset (0x0134)

GPIO3B drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0xaa81	<p>gpio3b_p</p> <p>GPIO3B PU/PD programmation section, every GPIO bit corresponding to 2bits</p> <p>2'b00: Z(Noraml operaton);</p> <p>2'b01: weak 1(pull-up);</p> <p>2'b10: weak 0(pull-down);</p> <p>2'b11: Repeater(Bus keeper)</p>

**BENZ\_GRF\_GPIO3C\_P**

Address: Operational Base + offset (0x0138)

GPIO3C drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:0	RW	0x6aaa	gpio3c_p GPIO3C PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO3D\_P**

Address: Operational Base + offset (0x013c)

GPIO3D drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0xaa54	gpio3d_p GPIO3D PU/PD programmation section, every GPIO bit corresponding to 2bits 2'b00: Z(Noraml operaton); 2'b01: weak 1(pull-up); 2'b10: weak 0(pull-down); 2'b11: Repeater(Bus keeper)

**BENZ\_GRF\_GPIO0A\_E**

Address: Operational Base + offset (0x0200)

GPIO0A drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:0	RW	0x5055	<p>gpio0a_e</p> <p>GPIO0A drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO0B\_E**

Address: Operational Base + offset (0x0204)

GPIO0B drive strength control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x5545	<p>gpio0b_e</p> <p>GPIO0B drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO0C\_E**

Address: Operational Base + offset (0x0208)

GPIO0C drive strength control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:0	RW	0x9155	<p>gpio0c_e</p> <p>GPIO0C drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO0D\_E**

Address: Operational Base + offset (0x020c)

GPIO0D drive strength control of N channel

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x5655	<p>gpio0d_e</p> <p>GPIO0D drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO1A\_E**

Address: Operational Base + offset (0x0210)

GPIO1A drive strength control of P channel

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:0	RW	0x4555	<p>gpio1a_e</p> <p>GPIO1A drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO1B\_E**

Address: Operational Base + offset (0x0214)

GPIO1B drive strength control of N channel

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x9155	<p>gpio1b_e</p> <p>GPIO1B drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO1C\_E**

Address: Operational Base + offset (0x0218)

GPIO1C drive strength control of P channel

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:0	RW	0xaaa6	<p>gpio1c_e</p> <p>GPIO1C drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO1D\_E**

Address: Operational Base + offset (0x021c)

GPIO1D drive strength control of N channel

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0xaaaa	<p>gpio1d_e</p> <p>GPIO1D drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO2A\_E**

Address: Operational Base + offset (0x0220)

GPIO2A drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:0	RW	0xaaaa	<p>gpio2a_e</p> <p>GPIO2A drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO2B\_E**

Address: Operational Base + offset (0x0224)

GPIO2B drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0xaaaa	<p>gpio2a_e</p> <p>GPIO2B drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO2C\_E**

Address: Operational Base + offset (0x0228)

GPIO2C drive strength control

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:0	RW	0xaaaa	<p>gpio2c_e</p> <p>GPIO2C drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO2D\_E**

Address: Operational Base + offset (0x022c)

GPIO2D drive strength control

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x055a	<p>gpio2d_e</p> <p>GPIO2D drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO3A\_E**

Address: Operational Base + offset (0x0230)

GPIO3A drive strength control

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:0	RW	0x5aaa	<p>gpio3a_e</p> <p>GPIO3A drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO3B\_E**

Address: Operational Base + offset (0x0234)

GPIO3B drive strength control

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x9541	<p>gpio3b_e</p> <p>GPIO3B drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO3C\_E**

Address: Operational Base + offset (0x0238)

GPIO3C drive strength control

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:0	RW	0x5559	<p>gpio3c_e</p> <p>GPIO3C drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO3D\_E**

Address: Operational Base + offset (0x023c)

GPIO3D drive strength control

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x5554	<p>gpio3d_e</p> <p>GPIO3D drive strength control, every GPIO bit corresponding to 2bits</p> <p>2'b00: 2mA 2'b01: 4mA 2'b10: 8mA 2'b11: 12mA</p>

**BENZ\_GRF\_GPIO00L\_SR**

Address: Operational Base + offset (0x0300)

GPIO0A/B SR control

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:8	RW	0x00	gpio0b_sr GPIO0B slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast
7:0	RW	0x00	gpio0a_sr GPIO0A slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast

**BENZ\_GRF\_GPIO0H\_SR**

Address: Operational Base + offset (0x0304)

GPIO0C/D SR control

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:8	RW	0x00	gpio0d_sr GPIO0D slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast
7:0	RW	0x00	gpio0c_sr GPIO0C slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast

**BENZ\_GRF\_GPIO1L\_SR**

Address: Operational Base + offset (0x0308)

GPIO1A/B SR control

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:8	RW	0x00	gpio1b_sr GPIO1B slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast
7:0	RW	0x00	gpio1a_sr GPIO1A slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast

**BENZ\_GRF\_GPIO1H\_SR**

Address: Operational Base + offset (0x030c)

GPIO1C/D SR control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:8	RW	0x00	gpio1d_sr GPIO1D slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast
7:0	RW	0x00	gpio1c_sr GPIO1C slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast

**BENZ\_GRF\_GPIO2L\_SR**

Address: Operational Base + offset (0x0310)

GPIO2A/B SR control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:8	RW	0x00	gpio2b_sr GPIO2B slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast
7:0	RW	0x00	gpio2a_sr GPIO2A slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast

**BENZ\_GRF\_GPIO2H\_SR**

Address: Operational Base + offset (0x0314)

GPIO2C/D SR control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:8	RW	0x00	gpio2d_sr GPIO2D slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast
7:0	RW	0x00	gpio2c_sr GPIO2C slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast

**BENZ\_GRF\_GPIO3L\_SR**

Address: Operational Base + offset (0x0318)

GPIO3A/B SR control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:8	RW	0x00	gpio3b_sr GPIO3B slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast
7:0	RW	0x00	gpio3a_sr GPIO3A slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast

**BENZ\_GRF\_GPIO3H\_SR**

Address: Operational Base + offset (0x031c)

GPIO3C/D SR control

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:8	RW	0x00	gpio3d_sr GPIO3D slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast
7:0	RW	0x00	gpio3c_sr GPIO3C slew rate control for each bit 1'b0: slow (half frequency) 1'b1: fast

**BENZ\_GRF\_GPIO0L\_SMT**

Address: Operational Base + offset (0x0380)

GPIO0A/B smitter control register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:8	RW	0x00	gpio0b_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.
7:0	RW	0x00	gpio0a_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.

**BENZ\_GRF\_GPIO0H\_SMT**

Address: Operational Base + offset (0x0384)

GPIO0C/D smitter control register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:8	RW	0x00	gpio0d_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.
7:0	RW	0x00	gpio0c_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.

**BENZ\_GRF\_GPIO1L\_SMT**

Address: Operational Base + offset (0x0388)

GPIO1A/B smitter control register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:8	RW	0x00	gpio1b_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.
7:0	RW	0x00	gpio1a_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.

**BENZ\_GRF\_GPIO1H\_SMT**

Address: Operational Base + offset (0x038c)

GPIO1C/D smitter control register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:8	RW	0x00	gpio1d_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.
7:0	RW	0x00	gpio1c_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.

**BENZ\_GRF\_GPIO2L\_SMT**

Address: Operational Base + offset (0x0390)

GPIO2A/B smitter control register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:8	RW	0x00	gpio2b_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.
7:0	RW	0x00	gpio2a_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.

**BENZ\_GRF\_GPIO2H\_SMT**

Address: Operational Base + offset (0x0394)

GPIO2C/D smitter control register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:8	RW	0x00	gpio2d_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.
7:0	RW	0x00	gpio2c_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.

**BENZ\_GRF\_GPIO3L\_SMT**

Address: Operational Base + offset (0x0398)

GPIO3A/B smitter control register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:8	RW	0x00	gpio3b_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.
7:0	RW	0x00	gpio3a_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.

**BENZ\_GRF\_GPIO3H\_SMT**

Address: Operational Base + offset (0x039c)

GPIO3C/D smitter control register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:8	RW	0x00	gpio3d_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.
7:0	RW	0x00	gpio3c_smt Schmitt trigger control. 0: No hysteresis 1: Schmitt trigger enabled.

**BENZ\_GRF\_SOC\_CON0**

Address: Operational Base + offset (0x0400)

SoC control register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
13	RW	0x0	grf_con_host2_arb_pause host2 ehci/ohci arbiter pause control
12	RO	0x0	reserved
11	RW	0x0	grf_con_host1_arb_pause host1 ehci/ohci arbiter pause control
10	RW	0x0	grf_con_host0_arb_pause host0 ehci/ohci arbiter pause control
9	RW	0x0	grf_con_hdmi_cecin_msk hdmi_cecin mask control 1 : active
8	RW	0x1	grf_con_ddrphy_bufferen_core ddr phy bufferen controlled by core 0: enable 1: disable
7	RW	0x0	grf_con_msch_mainpartialpop msch0_mainpartialpop bit control
6	RW	0x0	grf_con_msch_mainaddr3 msch_mainaddr3 bit control
5	RW	0x0	upctl_c_sysreq upctl_c_sysreq bit control
4	RW	0x0	upctl_c_active_in upctl_c_active_in bit control
3	RW	0x0	grf_con_upctl_anfifo upctl_anfifo bit control
2	RW	0x0	grf_con_upctl_aburstint upctl_aburstint bit control
1	RW	0x0	grf_con_upctl_lp_reset_mode upctl_lp_reset_mode bit control
0	RW	0x0	grf_con_ddr_16bit_en ddr0_16bit_en bit control 1 : 16bit enable 0 : 16bit disable

**BENZ\_GRF\_SOC\_CON1**

Address: Operational Base + offset (0x0404)

SoC control register 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:3	RO	0x0	reserved
2	RW	0x0	<p>grf_con_tsadc_tsen_pd_0 The power down signal of temperature sensor 0 : Enable sensor 1 : Power down</p>
1	RW	0x0	<p>grf_con_tsadc_dig_bypass The enable signal of the digital bypass function 0 : Don't bypass digital 1 : Bypass digital</p>
0	RW	0x0	<p>grf_con_tsadc_clk_sel The enable signal of the clock inverter for the analog to digital interface 0 : invert 1 : don't invert</p>

**BENZ\_GRF\_SOC\_CON2**

Address: Operational Base + offset (0x0408)

SoC control register 2

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15	RO	0x0	reserved
14	RW	0x0	grf_hdmi_i2c_sda_msk hdmi_i2c_sda_in mask
13	RW	0x0	grf_hdmi_i2c_scl_msk hdmi_i2c_scl mask
12	RW	0x0	grf_hdmiphy_pll_pd hdmiphy pll power down, active high

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
11	RW	0x0	grf_hdmp_pdata_en hdmpiphy input parallel data enable 1 : enable 0 : disable
10:8	RO	0x0	reserved
7:5	RW	0x0	grf_uart_rts_sel UART polarity selection for rts_n Every bit for one UART, bit2 is for UART2, bit1 is for UART1, bit0 is for UART0 1 : rts_n is high active 0 : rts_n is low active
4:3	RO	0x0	reserved
2:0	RW	0x0	grf_uart_cts_sel UART polarity selection for cts_n Every bit for one UART, bit2 is for UART2, bit1 is for UART1, bit0 is for UART0 1 : cts_n is high active 0 : cts_n is low active

**BENZ\_GRF\_SOC\_CON3**

Address: Operational Base + offset (0x040c)

SoC control register 3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:11	RO	0x0	reserved
10	RW	0x0	gmac_pwr_idlereq send idle request to peri niu 0: disable 1: enable
9	RW	0x0	peri_pwr_idlereq send idle request to peri niu 0: disable 1: enable
8	RW	0x0	gpu_pwr_idlereq send idle request to gpu niu 0: disable 1: enable

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
7	RW	0x0	vdec_pwr_idlereq send idle request to rkvdec niu 0: disable 1: enable
6	RW	0x0	vpu_pwr_idlereq send idle request to vpu niu 0: disable 1: enable
5	RW	0x0	vop_pwr_idlereq send idle request to vop niu 0: disable 1: enable
4	RW	0x0	vio_pwr_idlereq send idle request to vio niu 0: disable 1: enable
3	RW	0x0	sys_pwr_idle_req send idle request to bus niu 0: disable 1: enable
2	RW	0x0	msch_apb_pwr_idlereq send idle request to mschapb niu 0: disable 1: enable
1	RW	0x0	msch_pwr_idlereq send idle request to msch niu 0: disable 1: enable
0	RW	0x0	core_pwr_idlereq send idle request to core niu 0: disable 1: enable

**BENZ\_GRF\_SOC\_CON4**

Address: Operational Base + offset (0x0410)

SoC control register 4

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
------------	-------------	--------------------	--------------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15	RW	0x1	<p>rga_iep_hdcp_mst_Stall noc_vio1_rsp_err_stall bit control 0: stall response 1: error response</p>
14	RW	0x1	<p>vop_mst_Stall noc_vop_req_rsp_err_stall bit control 0: stall response 1: error response</p>
13	RW	0x1	<p>gpu_mst_Stall noc_gpu_req_rsp_err_stall bit control 0: stall response 1: error response</p>
12	RW	0x1	<p>vpu_serv_Stall noc_vpu_serv_rsp_err_stall bit control 0: stall response 1: error response</p>
11	RW	0x1	<p>vop_hslv_Stall noc_vop_ahb_rsp_err_stall bit control 0: stall response 1: error response</p>
10	RW	0x1	<p>vio_hslv_Stall noc_vio_hslv_rsp_err_stall bit control 0: stall response 1: error response</p>
9	RW	0x1	<p>vdec_slv_Stall noc_rkvdec_ahb_rsp_err_stall bit control 0: stall response 1: error response</p>
8	RW	0x1	<p>peri_hslv_Stall noc_peri_rsp_err_stall bit control 0: stall response 1: error response</p>
7	RW	0x1	<p>peri_pslv_Stall noc_peri_apb_rsp_err_stall bit control 0: stall response 1: error response</p>

Bit	Attr	Reset Value	Description
6	RW	0x1	msch_apb_Stall noc_mschapb_rsp_err_stall bit control 0: stall response 1: error response
5	RW	0x1	gpu_serv_Stall noc_gpu_rsp_err_stall bit control 0: stall response 1: error response
4	RW	0x1	crpt_dma_mst_Stall noc_crpt_dma_rsp_err_stall bit control 0: stall response 1: error response
3	RW	0x0	peri_h_Stall Field0000 Abstract noc_peri_p_rsp_err_stall bit control 0: stall response 1: error response
2	RO	0x0	reserved
1	RW	0x1	vdec_mst_Stall noc_rkvdec_rsp_err_stall bit control 0: stall response 1: error response
0	RW	0x1	vcodec_mst_Stall noc_vcodec_rsp_err_stall bit control 0: stall response 1: error response

**BENZ\_GRF\_SOC\_CON5**

Address: Operational Base + offset (0x0414)

SoC control register 5

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:2	RO	0x0	reserved
1	RW	0x1	core_msch_req_link_pwrDiscTargPwrStall noc_msch_req_rsp_err_stall bit control 0: stall response 1: error response

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
0	RW	0x1	core_bus_req_link_pwrDiscTargPwrStall noc_bus_req_rsp_err_stall bit control 0: stall response 1: error response

**BENZ\_GRF\_SOC\_CON6**

Address: Operational Base + offset (0x0418)

SoC control register 6

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:9	RO	0x0	reserved
8	RW	0x1	grf_force_jtag Force select jtag function from sdmmc0 IO 1: IO used for JTAG. 0: IO used for SDMMC
7	RO	0x0	reserved
6:4	RW	0x0	grf_hdmi_vsel hdmi port 3.3v/5v io select bit0 is for hdmi_ddcscl & i2c3_scl bit1 is for hdmi_ddcsda & i2c3_sda bit2 is for hdmi_hpd 1 : IO is 3.3v 0 : IO is 5v
3:0	RW	0x0	grf_io_vsel IO voltage select control 1'b0: 3.3V/2.5V 1'b1: 1.8V

**BENZ\_GRF\_SOC\_STATUS0**

Address: Operational Base + offset (0x0480)

SoC status register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RO	0x0	reserved
23	RO	0x0	grf_stat_vdac_dispdet vdac cable detection output status
22	RO	0x0	vop_dma_finish vop_dma_finish status

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
21:19	RO	0x0	reserved
18	RO	0x1	nandc_masteridle nandc master idle status
17	RO	0x0	timer_en_status5 timer5_en_status
16	RO	0x0	timer_en_status4 timer4_en_status
15	RO	0x0	timer_en_status3 timer3_en_status
14	RO	0x0	timer_en_status2 timer2_en_status
13	RO	0x0	timer_en_status1 timer1_en_status
12	RO	0x0	timer_en_status0 timer0_en_status
11	RO	0x0	gmac_portselect signal indicating the default PHY interface of MAC 1 : MII 0 : GMII
10	RO	0x0	reserved
9:6	RO	0x0	pll_lock PLL lock status, every PLL corresponding to 1bits [0]: ddr pll [1]: arm pll [2]: codec pll [3]: general pll
5	RO	0x0	otg0_utmiotg_vbusvalid otg0_utmiotg_vbusvalid bit status
4	RO	0x0	otg0_bvalid otg0_bvalid bit status
3:2	RO	0x0	otg0_utmi_linestate otg0_utmi_linestate bit status
1	RO	0x0	otg0_utmiotg_iddig otg0_utmiotg_iddig bit status
0	RO	0x0	reserved

**BENZ\_GRF\_SOC\_STATUS1**

Address: Operational Base + offset (0x0484)

SoC status register 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:11	RO	0x0	reserved
10:0	RO	0x000	gpu_idle gpu_idle status

**BENZ\_GRF\_SOC\_STATUS2**

Address: Operational Base + offset (0x0488)

SoC status register 2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:27	RO	0x0	reserved
26	RO	0x1	gmac_pwr_Idle idle status of gmac niu 0: idle status is 0 1: idle status is 1
25	RO	0x1	peri_pwr_Idle idle status of peri niu 0: idle status is 0 1: idle status is 1
24	RO	0x0	gpu_pwr_Idle idle status of gpu niu 0: idle status is 0 1: idle status is 1
23	RO	0x0	vdec_pwr_Idle idle status of rkvdec niu 0: idle status is 0 1: idle status is 1
22	RO	0x0	vpu_pwr_Idle idle status of vpu niu 0: idle status is 0 1: idle status is 1
21	RO	0x1	vop_pwr_Idle idle status of vop niu 0: idle status is 0 1: idle status is 1
20	RO	0x1	vio_pwr_Idle idle status of vio niu 0: idle status is 0 1: idle status is 1
19	RO	0x1	sys_pwr_Idle idle status of bus niu 0: idle status is 0 1: idle status is 1
18	RO	0x1	msch_apb_pwr_Idle idle status of mschapb niu 0: idle status is 0 1: idle status is 1
17	RO	0x1	msch_pwr_Idle idle status of msch niu 0: idle status is 0 1: idle status is 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
16	RO	0x1	core_pwr_Idle idle status of core niu 0: idle status is 0 1: idle status is 1
15:11	RO	0x0	reserved
10	RO	0x1	gmac_pwr_IdleAck idle acknowledge status from gmac niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1
9	RO	0x1	peri_pwr_IdleAck idle acknowledge status from peri niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1
8	RO	0x0	gpu_pwr_IdleAck idle acknowledge status from gpu niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1
7	RO	0x0	vdec_pwr_IdleAck idle acknowledge status from rkvdec niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1
6	RO	0x0	vpu_pwr_IdleAck idle acknowledge status from vpu niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1
5	RO	0x1	vop_pwr_IdleAck idle acknowledge status from vop niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1
4	RO	0x1	vio_pwr_IdleAck idle acknowledge status from vio niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1
3	RO	0x1	sys_pwr_IdleAck idle acknowledge status from bus niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1
2	RO	0x1	msch_apb_pwr_IdleAck idle acknowledge status from mschapb niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1
1	RO	0x1	msch_pwr_IdleAck idle acknowledge status from msch niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1

Bit	Attr	Reset Value	Description
0	RO	0x1	core_pwr_IdleAck idle acknowledge status from core niu 0: idle acknowledge status is 0 1: idle acknowledge status is 1

**BENZ\_GRF\_CHIP\_ID**

Address: Operational Base + offset (0x048c)

chip\_id register

Bit	Attr	Reset Value	Description
31:0	RO	0x00003228	chip_id chip_id

**BENZ\_GRF\_CPU\_CON0**

Address: Operational Base + offset (0x0500)

CPU little cluster control register 0

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:13	RO	0x0	reserved
12	RW	0x0	I2rstdisable cpu dbgI1rstdisable
11:8	RW	0x0	I1rstdisable cpu dbgI1rstdisable
7:3	RO	0x0	reserved
2:0	RW	0x2	ema_I2_data_grf ema_I2_data

**BENZ\_GRF\_CPU\_CON1**

Address: Operational Base + offset (0x0504)

CPU little cluster control register 1

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:12	RO	0x0	reserved
11:8	RW	0x0	cfgte_grf cpu cfgte bit control
7:4	RO	0x0	reserved
3:0	RW	0x0	cfgend_grf cpu cfgend bit control

**BENZ\_GRF\_CPU\_CON2**

Address: Operational Base + offset (0x0508)

CPU little cluster control register 2

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:11	RO	0x0	reserved
10	RW	0x0	grf_con_l2flushreq cpu l2 flush request bit control
9	RW	0x0	grf_con_clrexmonreq cpu clrexmonreq bit control
8	RW	0x0	cfgsdisable_grf cpu cfgsdisable bit control
7	RW	0x0	evento_clear_grf pd_core evento_ack control bit
6	RW	0x0	eventi_grf pd_core eventi control bit
5	RW	0x1	dbgselfaddrv_grf cpu dbgselfaddrv bit control
4	RW	0x1	dbgromaddrv_grf cpu dbgromaddrv bit control
3:0	RO	0x0	reserved

**BENZ\_GRF\_CPU\_CON3**

Address: Operational Base + offset (0x050c)

CPU little cluster control register 3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:11	RO	0x0	reserved
10:9	RW	0x0	grf_hd_emaw hd_emaw bit control
8:6	RW	0x1	grf_hd_ema hd_ema bit control
5	RW	0x0	grf_hs_emas hs_emas bit control
4:3	RW	0x0	grf_hs_emaw hs_emaw bit control
2:0	RW	0x1	grf_hs_ema hs_ema bit control

**BENZ\_GRF\_CPU\_STATUS0**

Address: Operational Base + offset (0x0520)

CPU status register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:21	RO	0x0	reserved
20:17	RO	0x0	ncoreporeset ncoreporeset status
16:13	RO	0x0	ncorerereset ncorerereset status
12:9	RO	0x0	ndbgreset ndbgreset status
8	RO	0x0	grf_con_I2flushdone I2flushdone status
7	RO	0x0	grf_st_clrexmonack clrexmonack status
6:3	RO	0x0	smpnamp_grf smpnamp status
2	RO	0x1	jtagnw_st_grf jtagnw_st status

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
1	RO	0x1	jtagtop_st_grf jtagtop_st status
0	RO	0x0	evento_rising_edge evento_rising_edge status

**BENZ\_GRF\_CPU\_STATUS1**

Address: Operational Base + offset (0x0524)

CPU status register 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:9	RO	0x0	reserved
8:5	RO	0xf	standbywfi_grf standby wifi status
4:1	RO	0x0	standbywfe_grf standby wfe status
0	RO	0x0	standbywfil2_grf standby wifi I2 status

**BENZ\_GRF\_OS\_REG0**

Address: Operational Base + offset (0x05c8)

os register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	grf_os_reg0 grf_os_reg0

**BENZ\_GRF\_OS\_REG1**

Address: Operational Base + offset (0x05cc)

os register 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	grf_os_reg1 grf_os_reg1

**BENZ\_GRF\_OS\_REG2**

Address: Operational Base + offset (0x05d0)

os register 2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	grf_os_reg2 grf_os_reg2

**BENZ\_GRF\_OS\_REG3**

Address: Operational Base + offset (0x05d4)

os register 3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	grf_os_reg3 grf_os_reg3

**BENZ\_GRF\_OS\_REG4**

Address: Operational Base + offset (0x05d8)

os register 4

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	grf_os_reg4 grf_os_reg4

**BENZ\_GRF\_OS\_REG5**

Address: Operational Base + offset (0x05dc)

os register 5

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	grf_os_reg5 grf_os_reg5

**BENZ\_GRF\_OS\_REG6**

Address: Operational Base + offset (0x05e0)

os register 6

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	grf_os_reg6 grf_os_reg6

**BENZ\_GRF\_OS\_REG7**

Address: Operational Base + offset (0x05e4)

os register 7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	grf_os_reg7 grf_os_reg7

**BENZ\_GRF\_DDRC\_STAT**

Address: Operational Base + offset (0x0604)

DDRC status register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:21	RW	0x000	reserved
20	RW	0x0	upctl_c_sysack upctl_c_sysack status
19	RW	0x0	upctl_c_active upctl_c_active status
18:16	RW	0x0	ddrupctl_stat ddrupctl_stat status
15:0	RW	0x0000	ddrupctl_bbflags ddrupctl_bbflags status

**BENZ\_GRF\_SIG\_DETECT\_CON**

Address: Operational Base + offset (0x0680)

## External signal detect configue register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:14	RW	0x0	<p>sd_detect_filter_time_sel sd_detect_filter time sel 00: 5ms 01: 15ms 10: 35ms 11: 50ms</p>
13:12	RW	0x0	<p>host0_ls_filter_time_sel host0_ls_filter time select 00: 100us 01: 500us 10: 1ms 11:10ms</p>
11:10	RW	0x0	<p>otg_ls_filter_time_sel otg_ls_filter time select 00: 100us 01: 500us 10: 1ms 11:10ms</p>
9:8	RW	0x0	<p>otg_id_filter_time_sel otg_id_filter time select 00: 5ms 01:15ms 10: 35ms</p>
7	RO	0x0	reserved
6	RW	0x0	<p>otg_id_fall_edge_detect_en otg_id_fall_edge_detect enable 0: disable 1: enable</p>
5	RW	0x0	<p>otg_id_rise_edge_detect_en otg_id_detect enable 0: disable 1: enable</p>
4	RW	0x0	<p>host0_line_state_detect_en host0_line_state_detect enable 0: disable 1: enable</p>

Bit	Attr	Reset Value	Description
3	RW	0x0	otg_bvalid_detect_en otg_bvalid detect enable 0: disable 1: enable
2	RW	0x0	otg_linestate_detect_en otg_linestate_detect enable 0: disable 1: enable
1	RW	0x0	sd_detect_fall_edge_detect_en sd_detect_falling_edge enable 0: disable 1: enable
0	RW	0x0	sd_detect_rising_edge_dectect_en sd_detect_rising_edge enable 0: disable 1:enable

**BENZ\_GRF\_SIG\_DETECT\_CON1**

Address: Operational Base + offset (0x0684)

External signal detect configue register1

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:12	RO	0x0	reserved
11:10	RW	0x0	host2_ls_filter_time_sel host2_ls_filter time select 00: 100us 01: 500us 10: 1ms 11:10ms
9:8	RW	0x0	host1_ls_filter_time_sel host1_ls_filter time select 00: 100us 01: 500us 10: 1ms 11:10ms
7:2	RO	0x0	reserved

Bit	Attr	Reset Value	Description
1	RW	0x0	host2_line_state_detect_en host2_line_state_detect enable 0: disable 1: enable
0	RW	0x0	host1_line_state_detect_en host1_line_state_detect enable 0: disable 1: enable

**BENZ\_GRF\_SIG\_DETECT\_STATUS**

Address: Operational Base + offset (0x0690)

External signal detect status register

Bit	Attr	Reset Value	Description
31:7	RO	0x0	reserved
6	RW	0x0	otg_id_fall_edge_detect_status otg_id_fall_edge_detect status 0: disable 1: enable
5	RW	0x0	otg_id_rise_edge_detect_status otg_id_detect status 0: disable 1: enable
4	RW	0x0	host0_line_state_detect_status host0_line_state_detect status 0: disable 1: enable
3	RW	0x0	otg_bvalid_detect_status otg_bvalid detect status 0: disable 1: enable
2	RW	0x0	otg_linenstate_detect_status otg_linenstate_detect status 0: disable 1: enable
1	RW	0x0	sd_detect_fall_edge_detect_status sd_detect_falling_edge status 0: disable 1: enable
0	RW	0x0	sd_detect_rising_edge_dectect_status sd_detect_rising_edge status 0: disable 1:enable

**BENZ\_GRF\_SIG\_DETECT\_STATUS1**

Address: Operational Base + offset (0x0694)

## External signal detect status register1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:2	RO	0x0	reserved
1	RW	0x0	host2_line_state_detect_status host2_line_state_detect status 0: disable 1: enable
0	RW	0x0	host1_line_state_detect_status host1_line_state_detect status 0: disable 1: enable

**BENZ\_GRF\_SIG\_DETECT\_CLR**

Address: Operational Base + offset (0x06a0)

External signal detect configue register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:7	RO	0x0	reserved
6	RW	0x0	otg_id_fall_edge_detect_clr otg_id_fall_edge_detect enable 0: disable 1: enable
5	RW	0x0	otg_id_rise_edge_detect_clr otg_id_detect enable 0: disable 1: enable
4	RW	0x0	host0_line_state_detect_clr host0_line_state_detect enable 0: disable 1: enable
3	RW	0x0	otg_bvalid_detect_clr otg_bvalid detect enable 0: disable 1: enable
2	RW	0x0	otg_linestate_detect_clr otg_linestate_detect enable 0: disable 1: enable

Bit	Attr	Reset Value	Description
1	RW	0x0	sd_detect_fall_edge_detect_clr sd_detect_falling_edge enable 0: disable 1: enable
0	RW	0x0	sd_detect_rising_edge_dectect_clr sd_detect_rising_edge enable 0: disable 1:enable

**BENZ\_GRF\_SIG\_DETECT\_CLR1**

Address: Operational Base + offset (0x06a4)

External signal detect configue register1

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:2	RO	0x0	reserved
1	RW	0x0	host2_line_state_detect_clr host2_line_state_detect enable 0: disable 1: enable
0	RW	0x0	host1_line_state_detect_clr host1_line_state_detect enable 0: disable 1: enable

**BENZ\_GRF\_EMMC\_DET**

Address: Operational Base + offset (0x06b0)

emmc detect register

Bit	Attr	Reset Value	Description
31:20	RO	0x0	reserved
19:0	RW	0x00cff	grf_sdmmc_detttime sdmmc card detect time

**BENZ\_GRF\_HOST0\_CON0**

Address: Operational Base + offset (0x0700)

host0 control register 0

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:12	RO	0x0	reserved
11:6	RW	0x20	grf_con_host0_fladj_val_common USB HOST0 fladj_val_common bit control
5:0	RW	0x20	grf_con_host0_fladj_val USB HOST0 fladj bit control

**BENZ\_GRF\_HOST0\_CON1**

Address: Operational Base + offset (0x0704)

HOST0 control register 1

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit 0 ~ bit 15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:13	RO	0x0	reserved
12	RW	0x0	grf_con_host0_ohci_susp_lgcy USB HOST0 ohci_susp_lgcy bit control
11	RW	0x0	grf_con_host0_ohci_cntsel USB HOST0 ohci_cntsel bit control
10	RW	0x1	grf_con_host0_ohci_clkcktrst USB HOST0 ohci_clkcktrst bit control
9	RW	0x0	grf_con_host0_app_prt_ovrcur USB HOST0 app_prt_ovrcur bit control
8	RW	0x0	grf_con_host0_autoppd_on_overcur_en USB HOST0 autoppd_on_overcur_en bit control
7	RW	0x1	grf_con_host0_word_if USB HOST0 word_if bit control
6	RW	0x0	grf_con_host0_sim_mode USB HOST0 sim_mode bit control
5	RW	0x1	grf_con_host0_incrx_en USB HOST0 incr_x_en bit control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
4	RW	0x1	grf_con_host0_incr8_en USB HOST0 incr8_en bit control
3	RW	0x1	grf_con_host0_incr4_en USB HOST0 incr4_en bit control
2	RW	0x1	grf_con_host0_incr16_en USB HOST0 incr16_en bit control
1	RW	0x0	grf_con_host0_hubsetup_min USB HOST0 hubsetup_min bit control
0	RW	0x0	grf_con_host0_app_start_clk USB HOST0 app_start_clk bit control

**BENZ\_GRF\_HOST0\_CON2**

Address: Operational Base + offset (0x0708)

HOST0 control register 2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit 0 ~ bit 15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:6	RO	0x0	reserved
5	RW	0x0	host0_utmi_idpullup USB HOST0 utmi_idpullup bit control
4	RW	0x1	host0_utmi_dmpulldown USB HOST0 dmpulldown bit control
3	RW	0x1	host0_utmi_dppulldown USB HOST0 dppulldown bit control
2	RW	0x0	host0_utmi_dischrgvbus USB HOST0 dischrgvbus bit control
1	RW	0x0	host0_utmi_chrgvbus USB HOST0 utmi_chrgvbus bit control
0	RW	0x1	host0_utmi_drvvbus USB HOST0 utmi_drvvbus bit control

**BENZ\_GRF\_HOST1\_CON0**

Address: Operational Base + offset (0x0710)

HOST1 control register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:12	RO	0x0	reserved
11:6	RW	0x20	grf_con_host1_fladj_val_common USB HOST1 fladj_val_common bit control
5:0	RW	0x20	grf_con_host1_fladj_val USB HOST1 fladj bit control

**BENZ\_GRF\_HOST1\_CON1**

Address: Operational Base + offset (0x0714)

HOST1 control register 1

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:13	RO	0x0	reserved
12	RW	0x0	grf_con_host1_ohci_susp_lgcy USB HOST1 ohci_susp_lgcy bit control
11	RW	0x0	grf_con_host1_ohci_cntsel USB HOST1 ohci_cntsel bit control
10	RW	0x1	grf_con_host1_ohci_clkcktrst USB HOST1 ohci_clkcktrst bit control
9	RW	0x0	grf_con_host1_app_prt_ovrcur USB HOST1 app_prt_ovrcur bit control
8	RW	0x0	grf_con_host1_autoppd_on_overcur_en USB HOST1 autoppd_on_overcur_en bit control
7	RW	0x1	grf_con_host1_word_if USB HOST1 word_if bit control
6	RW	0x0	grf_con_host1_sim_mode USB HOST1 sim_mode bit control
5	RW	0x1	grf_con_host1_incrx_en USB HOST1 incr_x_en bit control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
4	RW	0x1	grf_con_host1_incr8_en USB HOST1 incr8_en bit control
3	RW	0x1	grf_con_host1_incr4_en USB HOST1 incr4_en bit control
2	RW	0x1	grf_con_host1_incr16_en USB HOST1 incr16_en bit control
1	RW	0x0	grf_con_host1_hubsetup_min USB HOST1 hubsetup_min bit control
0	RW	0x0	grf_con_host1_app_start_clk USB HOST1 app_start_clk bit control

**BENZ\_GRF\_HOST1\_CON2**

Address: Operational Base + offset (0x0718)

HOST1 control register 2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit 0 ~ bit 15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:6	RO	0x0	reserved
5	RW	0x0	host1_utmi_idpullup USB HOST1 utmi_idpullup bit control
4	RW	0x1	host1_utmi_dmpulldown USB HOST1 utmi_dmpulldown bit control
3	RW	0x1	host1_utmi_dppulldown USB HOST1 utmi_dppulldown bit control
2	RW	0x0	host1_utmi_dischrgvbus USB HOST1 utmi_dischrgvbus bit control
1	RW	0x0	host1_utmi_chrgvbus USB HOST1 bit control
0	RW	0x1	host1_utmi_drvvbus USB HOST1 utmi_drvvbus bit control

**BENZ\_GRF\_HOST2\_CON0**

Address: Operational Base + offset (0x0720)

HOST2 control register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:12	RO	0x0	reserved
11:6	RW	0x20	grf_con_host2_fladj_val_common USB HOST2 fladj_val_common bit control
5:0	RW	0x20	grf_con_host2_fladj_val USB HOST2 fladj_val bit control

**BENZ\_GRF\_HOST2\_CON1**

Address: Operational Base + offset (0x0724)

HOST2 control register 1

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:13	RO	0x0	reserved
12	RW	0x0	grf_con_host2_ohci_susp_lgcy USB HOST2 ohci_susp_lgcy bit control
11	RW	0x0	grf_con_host2_ohci_cntsel USB HOST2 ohci_cntsel bit control
10	RW	0x1	grf_con_host2_ohci_clkcktrst USB HOST2 ohci_clkcktrst bit control
9	RW	0x0	grf_con_host2_app_prt_ovrcur USB HOST2 app_prt_ovrcur bit control
8	RW	0x0	grf_con_host2_autoppd_on_overcur_en USB HOST2 autoppd_on_overcur_en bit control
7	RW	0x1	grf_con_host2_word_if USB HOST2 word_if bit control
6	RW	0x0	grf_con_host2_sim_mode USB HOST2 sim_mode bit control
5	RW	0x1	grf_con_host2_incrx_en USB HOST2 incr_x_en bit control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
4	RW	0x1	grf_con_host2_incr8_en USB HOST2 incr8_en bit control
3	RW	0x1	grf_con_host2_incr4_en USB HOST2 incr4_en bit control
2	RW	0x1	grf_con_host2_incr16_en USB HOST2 incr16_en bit control
1	RW	0x0	grf_con_host2_hubsetup_min USB HOST2 hubsetup_min bit control
0	RW	0x0	grf_con_host2_app_start_clk USB HOST2 app_start_clk bit control

**BENZ\_GRF\_HOST2\_CON2**

Address: Operational Base + offset (0x0728)

HOST2 control register 2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit 0 ~ bit 15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:6	RO	0x0	reserved
5	RW	0x0	host2_utmi_idpullup USB HOST2 utmi_idpullup bit control
4	RW	0x1	host2_utmi_dmpulldown USB HOST2 utmi_dmpulldown bit control
3	RW	0x1	host2_utmi_dppulldown USB HOST2 utmi_dppulldown bit control
2	RW	0x0	host2_utmi_dischrgvbus USB HOST2 utmi_dischrgvbus bit control
1	RW	0x0	host2_utmi_chrgvbus USB HOST2 utmi_chrgvbus bit control
0	RW	0x1	host2_utmi_drvvbus USB HOST2 utmi_drvvbus bit control

**BENZ\_GRF\_USBPHY0\_CON0**

Address: Operational Base + offset (0x0760)

USB PHY0 control register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:11	RO	0x0	reserved
10	RW	0x0	usb0tg_utmi_iddig USB Plug Indicator Output
9	RW	0x0	usb0tg_utmi_iddig_en usb0tg_utmi_iddig select between grf and phy 1 : from grf 0 : from phy
8	RW	0x0	usb0tg_utmi_dmpulldown Enable DMINUS Pull Down resistor
7	RW	0x0	usb0tg_utmi_dppulldown Enable DPLUS Pull Down resistor
6	RW	0x0	usb0tg_utmi_termselect Termination select between FS/LS and HS Terminations
5:4	RW	0x0	usb0tg_utmi_xcvrselect Transceiver Select between FS/LS and HS Transceivers
3:2	RW	0x0	usb0tg_utmi_opmode Operational mode selector between various modes
1	RW	0x0	usb0tg_utmi_suspend Suspend Mode enable 1'b0 : suspend 1'b1 : normal
0	RW	0x0	usb0tg_en usb0tg select control signal between grf and controller 1'b1 : from grf 1'b0 : from controller

**BENZ\_GRF\_USBPHY0\_CON1**

Address: Operational Base + offset (0x0764)

USB PHY0 control register 1

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:9	RO	0x0	reserved
8	RW	0x0	usbhost0_utmi_dmpulldown Enable DMINUS Pull Down resistor
7	RW	0x0	usbhost0_utmi_dppulldown Enable DPLUS Pull Down resistor
6	RW	0x0	usbhost0_utmi_termselect Termination select between FS/LS and HS Terminations
5:4	RW	0x0	usbhost0_utmi_xcvrselect Transceiver Select between FS/LS and HS Transceivers
3:2	RW	0x0	usbhost0_utmi_opmode Operational mode selector between various modes
1	RW	0x0	usbhost0_utmi_suspend_n Suspend Mode enable 1'b0 : suspend 1'b1 : normal
0	RW	0x0	usbhost0_en usbhost0 select control signal between grf and controller 1'b1 : from grf 1'b0 : from controller

**BENZ\_GRF\_USBPHY0\_CON2**

Address: Operational Base + offset (0x0768)

USB PHY0 control register 2

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:13	RO	0x0	reserved
12	RW	0x0	vdm_src_en_usbotg open dm voltage source

Bit	Attr	Reset Value	Description
11	RW	0x0	vdp_src_en_usbotg open dp voltage source
10	RW	0x0	rdm_pdwn_en_usbotg open dm pull down resistor
9	RW	0x0	idp_src_en_usbotg open dm source current
8	RW	0x0	idm_sink_en_usbotg open dm sink current
7	RW	0x0	idp_sink_en_usbotg open dp sink current
6:5	RO	0x0	reserved
4	RW	0x0	usbphy_commononnn configure PLL clock output in suspend mode
3	RW	0x0	bypasssel_usbotg bypass select
2	RW	0x0	bypassdmen_usbotg bypass dm enable
1	RW	0x0	usbotg_disable_1 bypass OTG function
0	RW	0x0	usbotg_disable_0 bypass OTG function

**BENZ\_GRF\_USBPHY0\_CON3**

Address: Operational Base + offset (0x076c)

USB PHY0 control register 3

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x8518	usbphy_con usbphy control register bit 15~0

**BENZ\_GRF\_USBPHY0\_CON4**

Address: Operational Base + offset (0x0770)

USB PHY0 control register 4

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0xe007	usbphy_con usbphy control register bit 31~16

**BENZ\_GRF\_USBPHY0\_CON5**

Address: Operational Base + offset (0x0774)

USB PHY0 control register 5

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x02e7	usbphy_con usbphy control register bit 47~32

**BENZ\_GRF\_USBPHY0\_CON6**

Address: Operational Base + offset (0x0778)

USB PHY0 control register 6

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0200	usbphy_con usbphy control register bit 63~48

**BENZ\_GRF\_USBPHY0\_CON7**

Address: Operational Base + offset (0x077c)

USB PHY0 control register 7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x5556	usbphy_con usbphy control register bit 79~64

**BENZ\_GRF\_USBPHY0\_CON8**

Address: Operational Base + offset (0x0780)

USB PHY0 control register 8

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x4555	usbphy_con usbphy control register bit 95~80

**BENZ\_GRF\_USBPHY0\_CON9**

Address: Operational Base + offset (0x0784)

USB PHY0 control register 9

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0005	usbphy_con usbphy control register bit 111~96

**BENZ\_GRF\_USBPHY0\_CON10**

Address: Operational Base + offset (0x0788)

USB PHY0 control register 10

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x68c0	<p>usbphy_con</p> <p>usbphy control register bit 127~112</p>

**BENZ\_GRF\_USBPHY0\_CON11**

Address: Operational Base + offset (0x078c)

USB PHY0 control register 11

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x0000	<p>usbphy_con</p> <p>usbphy control register bit 143~128</p>

**BENZ\_GRF\_USBPHY0\_CON12**

Address: Operational Base + offset (0x0790)

USB PHY0 control register 12

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:0	RW	0x0000	usbphy_con usbphy control register bit 159~144

**BENZ\_GRF\_USBPHY0\_CON13**

Address: Operational Base + offset (0x0794)

USB PHY0 control register 13

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	usbphy_con usbphy control register bit 175~160

**BENZ\_GRF\_USBPHY0\_CON14**

Address: Operational Base + offset (0x0798)

USB PHY0 control register 14

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	usbphy_con usbphy control register bit 191~176

**BENZ\_GRF\_USBPHY0\_CON15**

Address: Operational Base + offset (0x079c)

USB PHY0 control register 15

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x8518	usbphy_con usbphy control register bit 207~192

**BENZ\_GRF\_USBPHY0\_CON16**

Address: Operational Base + offset (0x07a0)

USB PHY0 control register 16

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0xe007	usbphy_con usbphy control register bit 223~208

**BENZ\_GRF\_USBPHY0\_CON17**

Address: Operational Base + offset (0x07a4)

USB PHY0 control register 17

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x02e7	usbphy_con usbphy control register bit 239~224

**BENZ\_GRF\_USBPHY0\_CON18**

Address: Operational Base + offset (0x07a8)

USB PHY0 control register 18

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0200	usbphy_con usbphy control register bit 255~240

**BENZ\_GRF\_USBPHY0\_CON19**

Address: Operational Base + offset (0x07ac)

USB PHY0 control register 19

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x5556	usbphy_con usbphy control register bit 271~256

**BENZ\_GRF\_USBPHY0\_CON20**

Address: Operational Base + offset (0x07b0)

USB PHY0 control register 20

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x4555	usbphy_con usbphy control register bit 287~272

**BENZ\_GRF\_USBPHY0\_CON21**

Address: Operational Base + offset (0x07b4)

USB PHY0 control register 21

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0005	usbphy_con usbphy control register bit 303~288

**BENZ\_GRF\_USBPHY0\_CON22**

Address: Operational Base + offset (0x07b8)

USB PHY0 control register 22

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x68c0	usbphy_con usbphy control register bit 319~304

**BENZ\_GRF\_USBPHY0\_CON23**

Address: Operational Base + offset (0x07bc)

USB PHY0 control register 23

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:0	RW	0x0000	usbphy_con usbphy control register bit 335~320

**BENZ\_GRF\_USBPHY0\_CON24**

Address: Operational Base + offset (0x07c0)

USB PHY0 control register 24

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	usbphy_con usbphy control register bit 351~336

**BENZ\_GRF\_USBPHY0\_CON25**

Address: Operational Base + offset (0x07c4)

USB PHY0 control register 25

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	usbphy_con usbphy control register bit 367~352

**BENZ\_GRF\_USBPHY0\_CON26**

Address: Operational Base + offset (0x07c8)

USB PHY0 control register 26

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x0000	usbphy_con usbphy control register bit 383~368

**BENZ\_GRF\_USBPHY1\_CON0**

Address: Operational Base + offset (0x0800)

USB PHY1 control register 0

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:11	RO	0x0	reserved
10	RW	0x0	usbhost1_utmi_iddig USB Plug Indicator Output
9	RW	0x0	usbhost1_utmi_iddig_en host1_utmi_iddig select between grf and phy 1 : from grf 0 : from phy
8	RW	0x0	usbhost1_utmi_dmpulldown Enable DMINUS Pull Down resistor
7	RW	0x0	usbhost1_utmi_dppulldown Enable DPLUS Pull Down resistor
6	RW	0x0	usbhost1_utmi_termselect Termination select between FS/LS and HS Terminations
5:4	RW	0x0	usbhost1_utmi_xcvrselect Transceiver Select between FS/LS and HS Transceivers
3:2	RW	0x0	usbhost1_utmi_opmode Operational mode selector between various modes

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
1	RW	0x0	usbhost1_utmi_suspend Suspend Mode enable 1'b0 : suspend 1'b1 : normal
0	RW	0x0	usbhost1_en host1 select control signal between grf and controller 1'b1 : from grf 1'b0 : from controller

**BENZ\_GRF\_USBPHY1\_CON1**

Address: Operational Base + offset (0x0804)

USB PHY1 control register 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:9	RO	0x0	reserved
8	RW	0x0	usbhost2_utmi_dmpulldown Enable DMINUS Pull Down resistor
7	RW	0x0	usbhost2_utmi_dppulldown Enable DPLUS Pull Down resistor
6	RW	0x0	usbhost2_utmi_termselect Termination select between FS/LS and HS Terminations
5:4	RW	0x0	usbhost2_utmi_xcvrselect Transceiver Select between FS/LS and HS Transceivers
3:2	RW	0x0	usbhost2_utmi_opmode Operational mode selector between various modes
1	RW	0x0	usbhos2_utmi_suspend_n Suspend Mode enable 1'b0 : suspend 1'b1 : normal
0	RW	0x0	usbhost2_en host2 select control signal between grf and controller 1'b1 : from grf 1'b0 : from controller

**BENZ\_GRF\_USBPHY1\_CON2**

Address: Operational Base + offset (0x0808)

USB PHY1 control register 2

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:13	RO	0x0	reserved
12	RW	0x0	vdm_src_en_usbotg open dm voltage source
11	RW	0x0	vdp_src_en_usbotg open dp voltage source
10	RW	0x0	rdm_pdwn_en_usbotg open dm pull down resistor
9	RW	0x0	idp_src_en_usbotg open dm source current
8	RW	0x0	idm_sink_en_usbotg open dm sink current
7	RW	0x0	idp_sink_en_usbotg open dp sink current
6:5	RO	0x0	reserved
4	RW	0x0	usbphy_commononnn configure PLL clock output in suspend mode
3	RW	0x0	bypasssel_usbotg bypass select
2	RW	0x0	bypassdmen_usbotg bypass dm enable
1	RW	0x0	usbotg_disable_1 bypass OTG function
0	RW	0x0	usbotg_disable_0 bypass OTG function

**BENZ\_GRF\_USBPHY1\_CON3**

Address: Operational Base + offset (0x080c)

USB PHY1 control register 3

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x8518	usbphy_con usbphy control register bit 15~0

**BENZ\_GRF\_USBPHY1\_CON4**

Address: Operational Base + offset (0x0810)

USB PHY1 control register 4

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0xe007	usbphy_con usbphy control register bit 31~16

**BENZ\_GRF\_USBPHY1\_CON5**

Address: Operational Base + offset (0x0814)

USB PHY1 control register 5

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x02e7	usbphy_con usbphy control register bit 47~32

**BENZ\_GRF\_USBPHY1\_CON6**

Address: Operational Base + offset (0x0818)

USB PHY1 control register 6

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0200	usbphy_con usbphy control register bit 63~48

**BENZ\_GRF\_USBPHY1\_CON7**

Address: Operational Base + offset (0x081c)

USB PHY1 control register 7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x5556	usbphy_con usbphy control register bit 79~64

**BENZ\_GRF\_USBPHY1\_CON8**

Address: Operational Base + offset (0x0820)

USB PHY1 control register 8

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x4555	usbphy_con usbphy control register bit 95~80

**BENZ\_GRF\_USBPHY1\_CON9**

Address: Operational Base + offset (0x0824)

USB PHY1 control register 9

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x0005	usbphy_con usbphy control register bit 111~96

**BENZ\_GRF\_USBPHY1\_CON10**

Address: Operational Base + offset (0x0828)

USB PHY1 control register 10

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x68c0	usbphy_con usbphy control register bit 127~112

**BENZ\_GRF\_USBPHY1\_CON11**

Address: Operational Base + offset (0x082c)

USB PHY1 control register 11

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>

Bit	Attr	Reset Value	Description
15:0	RW	0x0000	usbphy_con usbphy control register bit 143~128

**BENZ\_GRF\_USBPHY1\_CON12**

Address: Operational Base + offset (0x0830)

USB PHY1 control register 12

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	usbphy_con usbphy control register bit 159~144

**BENZ\_GRF\_USBPHY1\_CON13**

Address: Operational Base + offset (0x0834)

USB PHY1 control register 13

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	usbphy_con usbphy control register bit 175~160

**BENZ\_GRF\_USBPHY1\_CON14**

Address: Operational Base + offset (0x0838)

USB PHY1 control register 14

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	usbphy_con usbphy control register bit 191~176

**BENZ\_GRF\_USBPHY1\_CON15**

Address: Operational Base + offset (0x083c)

USB PHY1 control register 15

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x8518	usbphy_con usbphy control register bit 207~192

**BENZ\_GRF\_USBPHY1\_CON16**

Address: Operational Base + offset (0x0840)

USB PHY1 control register 16

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0xe007	usbphy_con usbphy control register bit 223~208

**BENZ\_GRF\_USBPHY1\_CON17**

Address: Operational Base + offset (0x0844)

USB PHY1 control register 17

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x02e7	<p>usbphy_con</p> <p>usbphy control register bit 239~224</p>

**BENZ\_GRF\_USBPHY1\_CON18**

Address: Operational Base + offset (0x0848)

USB PHY1 control register 18

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x0200	<p>usbphy_con</p> <p>usbphy control register bit 255~240</p>

**BENZ\_GRF\_USBPHY1\_CON19**

Address: Operational Base + offset (0x084c)

USB PHY1 control register 19

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x5556	<p>usbphy_con</p> <p>usbphy control register bit 271~256</p>

**BENZ\_GRF\_USBPHY1\_CON20**

Address: Operational Base + offset (0x0850)

USB PHY1 control register 20

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x4555	usbphy_con usbphy control register bit 287~272

**BENZ\_GRF\_USBPHY1\_CON21**

Address: Operational Base + offset (0x0854)

USB PHY1 control register 21

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0005	usbphy_con usbphy control register bit 303~288

**BENZ\_GRF\_USBPHY1\_CON22**

Address: Operational Base + offset (0x0858)

USB PHY1 control register 22

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;

Bit	Attr	Reset Value	Description
15:0	RW	0x68c0	usbphy_con usbphy control register bit 319~304

**BENZ\_GRF\_USBPHY1\_CON23**

Address: Operational Base + offset (0x085c)

USB PHY1 control register 23

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	usbphy_con usbphy control register bit 335~320

**BENZ\_GRF\_USBPHY1\_CON24**

Address: Operational Base + offset (0x0860)

USB PHY1 control register 24

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	usbphy_con usbphy control register bit 351~336

**BENZ\_GRF\_USBPHY1\_CON25**

Address: Operational Base + offset (0x0864)

USB PHY1 control register 25

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x0000	usbphy_con usbphy control register bit 367~352

**BENZ\_GRF\_USBPHY1\_CON26**

Address: Operational Base + offset (0x0868)

USB PHY1 control register 26

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:0	RW	0x0000	usbphy_con usbphy control register bit 383~368

**BENZ\_GRF\_OTG\_CON0**

Address: Operational Base + offset (0x0880)

OTG control register 0

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:12	RO	0x0	reserved
11	RW	0x0	otg_utmi_fs_se0 OTG utmi_fs_xver_own bit control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
10	RW	0x0	otg_utmi_fs_data OTG utmi_fs_xver_own bit control
9	RW	0x0	otg_utmi_fs_oe OTG utmi_fs_xver_own bit control
8	RW	0x0	otg_utmi_fs_xver_own OTG utmi_fs_xver_own bit control
7:3	RO	0x0	reserved
2	RW	0x0	otg_dbnce_fltr_bypass OTG dbnce_fltr_bypass bit control
1:0	RW	0x0	otg_scaledown_mode OTG scaledown_mode bit control

**BENZ\_GRF\_UOC\_STATUS0**

Address: Operational Base + offset (0x0884)

USBPHY status register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:9	RO	0x0000000	reserved
8	RO	0x0	otg_utmi_vpi otg_utmi_vpi status
7	RO	0x0	otg_utmi_vmi otg_utmi_vmi status
6	RO	0x0	otg_utmi_ls_fs_rcv utmi_ls_fs_rcv status
5	RO	0x0	grf_stat_usbphy0_dp_attached usbphy0_dp_attached status
4	RO	0x0	grf_stat_usbphy0_cp_detected usbphy0_cp_detected status
3	RO	0x0	grf_stat_usbphy0_dcp_detected usbphy0_dcp_detected status
2	RO	0x0	grf_stat_usbphy1_dp_attached usbphy1_dp_attached status
1	RO	0x0	grf_stat_usbphy1_cp_detected usbphy1_cp_detected status
0	RO	0x0	grf_stat_usbphy1_dcp_detected usbphy1_dcp_detected status

**BENZ\_GRF\_MAC\_CON0**

Address: Operational Base + offset (0x0900)

MAC control register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RO	0x0	reserved
13:7	RW	0x00	gmac_clk_rx_dl_cfg RGMII RX clock delayline value
6:0	RW	0x00	gmac_clk_tx_dl_cfg RGMII TX clock delayline value

**BENZ\_GRF\_MAC\_CON1**

Address: Operational Base + offset (0x0904)

MAC control register 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:12	RO	0x0	reserved
11	RW	0x0	grf_con_rmii_mode
10	RW	0x0	rmii_mode RMII mode selection 1'b1: RMII mode 1'b0: MII mode
9:8	RW	0x0	gmac_clk_sel RGMII clock selection 2'b00: 125MHz 2'b11: 25MHz 2'b10: 2.5MHz
7	RW	0x0	rmii_clk_sel RMII clock selection 1'b1: 25MHz 1'b0: 2.5MHz

Bit	Attr	Reset Value	Description
6:4	RW	0x0	gmac_phy_intf_sel PHY interface select 3'b001: RGMII 3'b100: RMII All others: Reserved
3	RW	0x0	gmac_flowctrl GMAC transmit flow control When set high, instructs the GMAC to transmit PAUSE Control frames in Full-duplex mode. In Half-duplex mode, the GMAC enables the Back-pressure function until this signal is made low again
2	RW	0x0	gmac_mac_speed MAC speed 1'b1: 100-Mbps 1'b0: 10-Mbps
1	RW	0x0	gmac_rxclk_dly_ena RGMII RX clock delayline enable 1'b1: enable 1'b0: disable
0	RW	0x0	gmac_txclk_dly_ena RGMII TX clock delayline enable 1'b1: enable 1'b0: disable

**BENZ\_GRF\_MACPHY\_CON0**

Address: Operational Base + offset (0x0b00)

MACPHY control register 0

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15	RW	0x0	macphy_cfg_ref_clk_sel
14	RW	0x0	macphy_cfg_clk_freq for 25 MHz clock input; for 50 MHz clock input.
13	RW	0x1	macphy_cfg_automdix_en Enables auto-detection of MDI/MDIX mode. Refer to "cfg_mode"

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
12	RW	0x0	macphy_cfg_en_high Defines polarity of output enable signals. "0" for active low output enable signal. "mdio_dir, rxdz,miz,rzerz" signal polarity control.
11	RW	0x0	macphy_cfg_fx_mode Enables FX mode
10	RW	0x0	macphy_cfg_adc_bp Field0000 Abstract Puts the ADC by default in bypass mode
9	RW	0x0	macphy_cfg_pll_bp Puts the PLL by default in bypass mode
8	RW	0x0	macphy_cfg_smii_source_sync smii source sync register field. Only relevant for SMII mode
7:6	RW	0x0	macphy_cfg_mii_mode MII mode register field. "00" for MII mode, "01" for RMII mode, "10" for SMII, "11" reserved
5:3	RW	0x7	macphy_cfg_mode MODE register file. "000" - 10BaseT, Half Duplex, Auto negotiation disabled "001" - 10Base-T, Full Duplex, Auto negotiation disabled "010" - 100Base-TX, Half Duplex, Auto-negotiation disabled "011" - 100Base-TX, Full Duplex, Auto-negotiation disabled "100" - 100Base-Tx, Half Duplex, Auto-negotiation Enabled "101" - Repeater mode, 100Base-Tx, Half Duplex, Auto-negotiation Enabled "110" - Power down mode, In this mode phy wake up in power down mode "111" - All capable, Full Duplex, 10 & 100 BT, Auto negotiation enabled, AutoMDIX enable
2	RW	0x0	macphy_cfg_powerup_reset Power Up Reset bit. Default value of powerup_reset bit 0 - Power up reset disabled by default 1 - Power up reset enabled by default
1	RW	0x0	macphy_cfg_power_down Power Down bit. Default value of True power down bit 1 - True power down is active by default 0 - True power down is not active by default
0	RW	0x1	macphy_cfg_enable PHY enable signal (active high). 1 = Enable MACHY IP 0 = Disable MACHY IP

**BENZ\_GRF\_MACPHY\_CON1**

Address: Operational Base + offset (0x0b04)

MACPHY control register 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:8	RO	0x0	reserved
7:3	RW	0x00	macphy_cfg_phy_addr PHY ADD register field. Must be unique in multi-PHY environment (like repeater).
2:0	RW	0x0	macphy_cfg_np_msg_code Next Page Message Code. Automatic generation of Next page with fault code

**BENZ\_GRF\_MACPHY\_CON2**

Address: Operational Base + offset (0x0b08)

MACPHY control register 2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	macphy_cfg_phy_id PHY ID Number,macphy_cfg_phy_id[15:0]

**BENZ\_GRF\_MACPHY\_CON3**

Address: Operational Base + offset (0x0b0c)

MACPHY control register 3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:12	RW	0x0	macphy_cfg_rev_nr Manufacturer's Revision Number
11:6	RW	0x00	macphy_cfg_model_nr Manufacturer's Model Number
5:0	RW	0x00	macphy_cfg_phy_id PHY ID Number,macphy_cfg_phy_id[21:16]

**BENZ\_GRF\_MACPHY\_STATUS**

Address: Operational Base + offset (0x0b10)

MACPHY status register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x00000000	reserved
6	RO	0x0	macphy_stat_speed100 Speed100 indication. Output driven low
5	RO	0x0	macphy_stat_speed10 Speed10 indication. Output is driven low
4	RO	0x0	macphy_stat_duplex Duplex indication (low = full-duplex mode).Output is driven low
3	RO	0x0	macphy_stat_rx RX activity indication.Output is driven low
2	RO	0x0	macphy_stat_link Link ON indication. Output is driven low
1	RO	0x0	macphy_stat_tx TX activity indication.Output is driven low
0	RO	0x0	macphy_stat_powerup_reset Power up reset state signal. To signal to the system that PHY is out of power down mode

## Chapter 6 Secure General Register Files (SGRF)

### 6.1 Overview

The general register file will be used to do static set by software, which is composed of many registers for system control.

### 6.2 Function Description

The function of secure general register file is:

- Master secure control
- Slave secure control
- DDR secure region control
- Bootrom configuration control
- Bus DMAC configuration control

### 6.3 Register Description

#### 6.3.1 Registers Summary

Name	Offset	Size	Reset Value	Description
SGRF_SOC_CON0	0x0000	W	0x00000000	Secure Control Register 0
SGRF_SOC_CON1	0x0004	W	0x00003fff	Secure Control Register 1
SGRF_SOC_CON2	0x0008	W	0x000003ff	Secure Control Register 2
SGRF_SOC_CON3	0x000c	W	0x00000401	Secure Control Register 3
SGRF_SOC_CON4	0x0010	W	0x00000000	Secure Control Register 4
SGRF_SOC_CON5	0x0014	W	0x00000000	Secure Control Register 5
SGRF_SOC_CON6	0x0018	W	0x00000000	Secure Control Register 6
SGRF_SOC_CON7	0x001c	W	0x0000ffff	Secure Control Register 7
SGRF_SOC_CON8	0x0020	W	0x0000ffff	Secure Control Register 8
SGRF_SOC_CON9	0x0024	W	0x0000ffff	Secure Control Register 9
SGRF_SOC_CON10	0x0028	W	0x0000001f	Secure Control Register 10
SGRF_BUSDMAC_CON0	0x0100	W	0x000000f5	BUS DMAC control register 0
SGRF_BUSDMAC_CON1	0x0104	W	0x0000ffff	BUS DMAC control register 1
SGRF_BUSDMAC_CON2	0x0108	W	0x00000000	BUS DMAC control register 2
SGRF_BUSDMAC_CON3	0x010c	W	0x0000ffff	BUS DMAC control register 3
SGRF_FAST_BOOT_ADDR	0x0180	W	0x00000000	Fast Boot Addr
SGRF_EFUSE_PRG_MASK	0x0200	W	0x00000000	EFUSE1024 control register
SGRF_HDCP_KEY_REG0	0x0280	W	0x00000000	hdcp key register 0
SGRF_HDCP_KEY_REG1	0x0284	W	0x00000000	hdcp key register 1
SGRF_HDCP_KEY_REG2	0x0288	W	0x00000000	hdcp key register 2
SGRF_HDCP_KEY_REG3	0x028c	W	0x00000000	hdcp key register 3
SGRF_HDCP_KEY_REG4	0x0290	W	0x00000000	hdcp key register 4
SGRF_HDCP_KEY_REG5	0x0294	W	0x00000000	hdcp key register 5
SGRF_HDCP_KEY_REG6	0x0298	W	0x00000000	hdcp key register 6
SGRF_HDCP_KEY_REG7	0x029c	W	0x00000000	hdcp key register 7
SGRF_HDCP_KEY_ACCE S_MASK	0x02a0	W	0x00000000	hdcp key access mask register

Name	Offset	Size	Reset Value	Description

Name	Offset	Size	Reset Value	Description
DDR_SGRF_DDR_CON0	0x0000	W	0x00004000	DDR Secure Control Register 0
DDR_SGRF_DDR_CON1	0x0004	W	0x00001fff	DDR Secure Control Register 1
DDR_SGRF_DDR_CON2	0x0008	W	0x00000000	DDR Secure Control Register 2
DDR_SGRF_DDR_CON3	0x000c	W	0x00000000	DDR Secure Control Register 3
DDR_SGRF_DDR_CON4	0x0010	W	0x00000000	DDR Secure Control Register 4
DDR_SGRF_DDR_CON5	0x0014	W	0x00000000	DDR Secure Control Register 5
DDR_SGRF_DDR_CON6	0x0018	W	0x00000000	DDR Secure Control Register 6
DDR_SGRF_DDR_CON7	0x001c	W	0x00000000	DDR Secure Control Register 7
DDR_SGRF_DDR_CON8	0x0020	W	0x00000000	DDR Secure Control Register 8
DDR_SGRF_DDR_CON9	0x0024	W	0x00000000	DDR Secure Control Register 9
DDR_SGRF_DDR_CON10	0x0028	W	0x00000000	DDR Secure Control Register 10
DDR_SGRF_DDR_CON11	0x002c	W	0x00000000	DDR Secure Control Register 11
DDR_SGRF_DDR_CON12	0x0030	W	0x00000000	DDR Secure Control Register 12
DDR_SGRF_DDR_CON13	0x0034	W	0x00000000	DDR Secure Control Register 13
DDR_SGRF_DDR_CON14	0x0038	W	0x00000000	DDR Secure Control Register 14
DDR_SGRF_DDR_CON15	0x003c	W	0x00000000	DDR Secure Control Register 15
DDR_SGRF_DDR_CON16	0x0040	W	0x0000ffff	DDR Secure Control Register 16
DDR_SGRF_DDR_CON17	0x0044	W	0x0000ffff	DDR Secure Control Register 17
DDR_SGRF_DDR_CON18	0x0048	W	0x0000ffff	DDR Secure Control Register 18
DDR_SGRF_DDR_CON19	0x004c	W	0x0000ffff	DDR Secure Control Register 19
DDR_SGRF_DDR_CON20	0x0050	W	0x0000ffff	DDR Secure Control Register 20
DDR_SGRF_DDR_CON21	0x0054	W	0x0000ffff	DDR Secure Control Register 21
DDR_SGRF_DDR_CON22	0x0058	W	0x000000ff	DDR Secure Control Register 22

Notes:Size: **B**- Byte (8 bits) access, **HW**- Half WORD (16 bits) access, **W**-WORD (32 bits) access

### 6.3.2 Detail Register Description

#### SGRF\_SOC\_CON0

Address: Operational Base + offset (0x0000)

Secure Control Register 0

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:12	RO	0x0	reserved
11	RW	0x0	sgrf_con_hdcp_uart_sel bit select control for uart3 and hdcp uart 0: uart3 1: hdcp_uart

Bit	Attr	Reset Value	Description
10	RW	0x0	remap 0: boot from boot-rom 1: boot from int-mem
9:0	RW	0x000	sgrf_con_tzma_r0size tzma_r0size bit control 10'h0: 0KB secure address 10'h1: 4KB secure address 10'h2: 8KB secure address ... 10'hf: 64KB secure address

**SGRF\_SOC\_CON1**

Address: Operational Base + offset (0x0004)

Secure Control Register 1

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:14	RO	0x0	reserved
13	RW	0x1	tsp_rkst_secure_ctrl TSP master secure setting 1'b1: non secure 1'b0: secure
12	RW	0x1	gpu_rd_secure_ctrl GPU master read channel secure setting and gpu_jtag access control 1'b1: non secure, jtag access disable 1'b0: secure, jtag access enable
11	RW	0x1	gpu_wr_secure_ctrl GPU master write channel secure setting 1'b1: non secure 1'b0: secure
10	RW	0x1	video_rd_secure_ctrl video master read channel secure setting 1'b1: non secure 1'b0: secure

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
9	RW	0x1	video_wr_secure_ctrl video master write channel secure setting 1'b1: non secure 1'b0: secure
8	RW	0x1	rkvdec_rd_secure_ctrl rkvdec master read channel secure setting 1'b1: non secure 1'b0: secure
7	RW	0x1	rkvdec_wr_secure_ctrl rkvdec master write channel secure setting 1'b1: non secure 1'b0: secure
6	RW	0x1	hdcp_mst_secure_ctrl HDCP master secure setting 1'b1: non secure 1'b0: secure
5	RW	0x1	rga_mst_wr_secure_ctrl RGA master write channel secure setting 1'b1: non secure 1'b0: secure
4	RW	0x1	rga_mst_rd_secure_ctrl RGA master read channel secure setting 1'b1: non secure 1'b0: secure
3	RW	0x1	iep_mst_wr_secure_ctrl IEP master write channel secure setting 1'b1: non secure 1'b0: secure
2	RW	0x1	iep_mst_rd_secure_ctrl IEP master read channel secure setting 1'b1: non secure 1'b0: secure
1	RW	0x1	vop0_mst_rd_secure_ctrl VOP master read channel secure setting 1'b1: non secure 1'b0: secure
0	RW	0x1	crypto_rkst_secure_ctrl CRYPTO master secure setting 1'b1: non secure 1'b0: secure

**SGRF\_SOC\_CON2**

Address: Operational Base + offset (0x0008)  
 Secure Control Register 2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:10	RO	0x0	reserved
9	RW	0x1	<p>gmac_wr_rkst_secure_ctrl GMAC write channel master secure setting 1'b1: non secure 1'b0: secure</p>
8	RW	0x1	<p>gmac_rd_rkst_secure_ctrl GMAC read channel master secure setting 1'b1: non secure 1'b0: secure</p>
7	RW	0x1	<p>otg_rkst_secure_ctrl OTG master secure setting 1'b1: non secure 1'b0: secure</p>
6	RW	0x1	<p>host2_rkst_secure_ctrl HOST2 master secure setting 1'b1: non secure 1'b0: secure</p>
5	RW	0x1	<p>host1_rkst_secure_ctrl HOST1 master secure setting 1'b1: non secure 1'b0: secure</p>
4	RW	0x1	<p>host0_rkst_secure_ctrl HOST0 master secure setting 1'b1: non secure 1'b0: secure</p>
3	RW	0x1	<p>sdmmc_rkst_secure_ctrl SDMMC master secure setting 1'b1: non secure 1'b0: secure</p>
2	RW	0x1	<p>sdio_rkst_secure_ctrl SDIO master secure setting 1'b1: non secure 1'b0: secure</p>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
1	RW	0x1	emmc_rkst_secure_ctrl EMMC master secure setting 1'b1: non secure 1'b0: secure
0	RW	0x1	nandc_rkst_secure_ctrl NANDC master secure setting 1'b1: non secure 1'b0: secure

**SGRF\_SOC\_CON3**

Address: Operational Base + offset (0x000c)

Secure Control Register 3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:12	RO	0x0	reserved
11	RW	0x0	cp15sdisable ca53 cpu cp15sdisable bit control Disable write access to some secure CP15 registers
10	RW	0x1	dapdeviceen DAP dapdeviceen bit control Enable access to Debug APB from the DAP
9:6	RW	0x0	vinithi ca53 cpu vinithi bit control Location of the exception vectors at reset. It sets the initial value of the V bit in the CP15 SCTRL register 0: Exception vectors start at address 0x00000000. 1: Exception vectors start at address 0xFFFF0000. This signal is sampled only during reset of the processor.
5	RW	0x0	spniden ca53 cpu spniden bit control Secure privileged non-invasive debug enable: 0: Not Enabled. 1: Enabled

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
4	RW	0x0	niden ca53 cpu niden bit control Non-invasive debug enable: 0: Not Enabled. 1: Enabled
3	RW	0x0	spiden ca53 cpu spiden bit control Secure privileged invasive debug enable: 0: Not Enabled. 1: Enabled
2	RW	0x0	dbgen ca53 cpu dbgen bit control Invasive debug enable: 0: Not Enabled. 1: Enabled
1	RW	0x0	cryptodisable ca53 cpu cryptodisable bit control
0	RW	0x1	aa64naa32 ca53 cpu aa64naa32 bit control

**SGRF\_SOC\_CON4**

Address: Operational Base + offset (0x0010)

Secure Control Register 4

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	sgrf_con_rvbaraddr ca53 cpu rvbaraddr of pd_core_l bit control

**SGRF\_SOC\_CON5**

Address: Operational Base + offset (0x0014)

Secure Control Register 5

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:7	RO	0x0	reserved
6	RW	0x0	<p>clk_stimer1_en stimer1 clk enable 1 : enable 0 : disable</p>
5	RW	0x0	<p>clk_stimer0_en stimer0 clk enable 1 : enable 0 : disable</p>
4	RW	0x0	<p>stimer1_srstn_req stimer1 sys-resetn request</p>
3	RW	0x0	<p>stimer0_srstn_req stimer0 sysresetn request</p>
2	RW	0x0	<p>stimer_2ch_psrstn_req stimer presetn request</p>
1	RW	0x0	<p>wdt_psrstn_req watch dog presetn request</p>
0	RW	0x0	<p>pclk_wdt_en watch dog pclk enable signal 1 : enable 0 : disable</p>

**SGRF\_SOC\_CON6**

Address: Operational Base + offset (0x0018)

Secure Control Register 6

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:2	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
1	RW	0x0	sgrf_efuse_prg_en efuse program enable signal control 1 : controlled by efuse_controller 0 : disable
0	RW	0x0	sgrf_efuse_gpio_en efuse_gpio io control bit 1 : controlled by iomux 0 : IO as INPUT, PULLUP

**SGRF\_SOC\_CON7**

Address: Operational Base + offset (0x001c)

Secure Control Register 7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15	RW	0x1	spi_slv_secure_ctrl SPI slave secure setting 1'b1: secure 1'b0: non-secure
14	RW	0x1	i2c3_slv_secure_ctrl I2C3 slave secure setting 1'b1: secure 1'b0: non-secure
13	RW	0x1	i2c2_slv_secure_ctrl I2C2 slave secure setting 1'b1: secure 1'b0: non-secure
12	RW	0x1	i2c1_slv_secure_ctrl I2C1 slave secure setting 1'b1: secure 1'b0: non-secure
11	RW	0x1	i2c0_slv_secure_ctrl I2C0 slave secure setting 1'b1: secure 1'b0: non-secure

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
10	RW	0x1	efuse256_slv_secure_ctrl EFUSE256 slave secure setting 1'b1: secure 1'b0: non-secure
9	RW	0x1	uart2_slv_secure_ctrl UART2 slave secure setting 1'b1: secure 1'b0: non-secure
8	RW	0x1	uart1_slv_secure_ctrl UART1 slave secure setting 1'b1: secure 1'b0: non-secure
7	RW	0x1	uart0_slv_secure_ctrl UART0 slave secure setting 1'b1: secure 1'b0: non-secure
6	RW	0x1	grf_slv_secure_ctrl GRF slave secure setting 1'b1: secure 1'b0: non-secure
5	RW	0x1	i2s2_slv_secure_ctrl I2S2 slave secure setting 1'b1: secure 1'b0: non-secure
4	RW	0x1	spdif_slv_secure_ctrl SPDIF slave secure setting 1'b1: secure 1'b0: non-secure
3	RW	0x1	i2s0_slv_secure_ctrl I2S0 slave secure setting 1'b1: secure 1'b0: non-secure
2	RW	0x1	i2s1_slv_secure_ctrl I2S1 slave secure setting 1'b1: secure 1'b0: non-secure
1	RW	0x1	crypto_slv_secure_ctrl CRYPTO slave secure setting 1'b1: secure 1'b0: non-secure
0	RW	0x1	intmem_slv_secure_ctrl INTMEM slave secure setting 1'b1: secure 1'b0: non-secure

**SGRF\_SOC\_CON8**

Address: Operational Base + offset (0x0020)

Secure Control Register 8

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software. When bit 31=0, bit 15 cannot be written by software;
15	RW	0x1	gpu_slv_secure_ctrl GPU slave secure setting 1'b1: secure 1'b0: non-secure
14	RW	0x1	dfimon_slv_secure_ctrl DFIMON slave secure setting 1'b1: secure 1'b0: non-secure
13	RW	0x1	ddrpctl_slv_secure_ctrl DDRPCTL slave secure setting 1'b1: secure 1'b0: non-secure
12	RW	0x1	acodec_slv_secure_ctrl ACODEC slave secure setting 1'b1: secure 1'b0: non-secure
11	RW	0x1	ddrphy_slv_secure_ctrl DDRPHY slave secure setting 1'b1: secure 1'b0: non-secure
10	RW	0x1	tsadc_slv_secure_ctrl TSADC slave secure setting 1'b1: secure 1'b0: non-secure
9	RW	0x1	gpio3_slv_secure_ctrl GPIO3 slave secure setting 1'b1: secure 1'b0: non-secure
8	RW	0x1	gpio2_slv_secure_ctrl GPIO2 slave secure setting 1'b1: secure 1'b0: non-secure

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
7	RW	0x1	gpio1_slv_secure_ctrl GPIO1 slave secure setting 1'b1: secure 1'b0: non-secure
6	RW	0x1	gpio0_slv_secure_ctrl GPIO0 slave secure setting 1'b1: secure 1'b0: non-secure
5	RW	0x1	dmac_slv_secure_ctrl DMAC slave secure setting 1'b1: secure 1'b0: non-secure
4	RW	0x1	cru_slv_secure_ctrl CRU slave secure setting 1'b1: secure 1'b0: non-secure
3	RW	0x1	stimer_slv_secure_ctrl STIMER slave secure setting 1'b1: secure 1'b0: non-secure
2	RW	0x1	timer_slv_secure_ctrl TIMER slave secure setting 1'b1: secure 1'b0: non-secure
1	RW	0x1	pwm_slv_secure_ctrl PWM slave secure setting 1'b1: secure 1'b0: non-secure
0	RW	0x1	wdt_slv_secure_ctrl WDT slave secure setting 1'b1: secure 1'b0: non-secure

**SGRF\_SOC\_CON9**

Address: Operational Base + offset (0x0024)

Secure Control Register 9

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15	RW	0x1	<p>usbhost0_slv_secure_ctrl USBHOST0 slave secure setting 1'b1: secure 1'b0: non-secure</p>
14	RW	0x1	<p>usbottg_slv_secure_ctrl USBOTG slave secure setting 1'b1: secure 1'b0: non-secure</p>
13	RW	0x1	<p>nandc_slv_secure_ctrl NANDC slave secure setting 1'b1: secure 1'b0: non-secure</p>
12	RW	0x1	<p>emmc_slv_secure_ctrl EMMC slave secure setting 1'b1: secure 1'b0: non-secure</p>
11	RW	0x1	<p>sdio_slv_secure_ctrl SDIO slave secure setting 1'b1: secure 1'b0: non-secure</p>
10	RW	0x1	<p>sdmmc_slv_secure_ctrl SDMMC slave secure setting 1'b1: secure 1'b0: non-secure</p>
9	RW	0x1	<p>hdmi_ctrl_slv_secure_ctrl HDMI_CTRL slave secure setting 1'b1: secure 1'b0: non-secure</p>
8	RW	0x1	<p>vdac_slv_secure_ctrl VDAC slave secure setting 1'b1: secure 1'b0: non-secure</p>
7	RW	0x1	<p>hdmiphy_slv_secure_ctrl HDMIPHY slave secure setting 1'b1: secure 1'b0: non-secure</p>

Bit	Attr	Reset Value	Description
6	RW	0x1	hdcp_slv_secure_ctrl HDCP slave secure setting 1'b1: secure 1'b0: non-secure
5	RW	0x1	hdcpmmu_slv_secure_ctrl HDCPMMU slave secure setting 1'b1: secure 1'b0: non-secure
4	RW	0x1	iep_slv_secure_ctrl IEP slave secure setting 1'b1: secure 1'b0: non-secure
3	RW	0x1	rga_slv_secure_ctrl RGA slave secure setting 1'b1: secure 1'b0: non-secure
2	RW	0x1	vop_slv_secure_ctrl VOP slave secure setting 1'b1: secure 1'b0: non-secure
1	RW	0x1	rkvdec_slv_secure_ctrl RKVDEC slave secure setting 1'b1: secure 1'b0: non-secure
0	RW	0x1	vpu_slv_secure_ctrl VPU slave secure setting 1'b1: secure 1'b0: non-secure

**SGRF\_SOC\_CON10**

Address: Operational Base + offset (0x0028)

Secure Control Register 10

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:8	RO	0x0	reserved
7	RW	0x0	slv_bp_s slave bypass select

Bit	Attr	Reset Value	Description
6:5	RO	0x0	reserved
4	RW	0x1	scr_slv_secure_ctrl SCR slave secure setting 1'b1: secure 1'b0: non-secure
3	RW	0x1	tsp_slv_secure_ctrl TSP slave secure setting 1'b1: secure 1'b0: non-secure
2	RW	0x1	gmac_slv_secure_ctrl GMAC slave secure setting 1'b1: secure 1'b0: non-secure
1	RW	0x1	usbhost2_slv_secure_ctrl USBHOST2 slave secure setting 1'b1: secure 1'b0: non-secure
0	RW	0x1	usbhost1_slv_secure_ctrl Field0000 Abstract USBHOST1 slave secure setting 1'b1: secure 1'b0: non-secure

**SGRF\_BUSDMAC\_CON0**

Address: Operational Base + offset (0x0100)

BUS DMAC control register 0

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:12	RW	0x0	busdmac_boot_addr busdmac_boot_addr[15:12] Configures the address location that contains the first instruction the DMAC executes, when it exits from reset.
11:8	RO	0x0	reserved

Bit	Attr	Reset Value	Description
7:4	RW	0xf	busdma_boot_peri_ns busdma_boot_peri_ns[19:16] Controls the security state of a peripheral request interface, when the BUS DMAC exits from reset.
3	RW	0x0	busdma_boot_from_pc DMAC boot_from_pc input control Controls the location in which the DMAC0 executes its initial instruction, after it exits from reset : 1'b0: DMAC waits for an instruction from APB interface 1'b1: DMAC manager thread executes the instruction that is located at the address that boot_addr[31:0] provided
2	RW	0x1	busdma_boot_manager_ns
1:0	RW	0x1	grf_drtype_busdma DMAC type of acknowledgement or request for peripheral signals: 2'b00: single level request 2'b01: burst level request 2'b10: acknowledging a flush request 2'b11: reserved

**SGRF\_BUSDMACon1**

Address: Operational Base + offset (0x0104)

BUS DMAC control register 1

Bit	Attr	Reset Value	Description
31:16	WO	0x0000	wirte_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0xffff	busdma_boot_peri_ns busdma_boot_peri_ns[15:0] Controls the security state of a peripheral request interface, when the BUS DMAC exits from reset.

**SGRF\_BUSDMACon2**

Address: Operational Base + offset (0x0108)

BUS DMAC control register 2

Bit	Attr	Reset Value	Description

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	wirte_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0x0000	busdma_boot_addr busdma_boot_addr[31:16] BUS DMAC boot_addr[31:16] input control Configures the address location that contains the first instruction the DMAC executes, when it exits from reset.

**SGRF\_BUSDMAC\_CON3**

Address: Operational Base + offset (0x010c)  
BUS DMAC control register 3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	WO	0x0000	wirte_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:0	RW	0xffff	busdma_boot_irq_ns Set bit for BUS DMAC boot_irq_ns input port control

**SGRF\_FAST\_BOOT\_ADDR**

Address: Operational Base + offset (0x0180)  
Fast Boot Addr

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**SGRF\_EFUSE\_PRG\_MASK**

Address: Operational Base + offset (0x0200)  
EFUSE1024 control register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**SGRF\_HDCP\_KEY\_REG0**

Address: Operational Base + offset (0x0280)

hdcp key register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**SGRF\_HDCP\_KEY\_REG1**

Address: Operational Base + offset (0x0284)

hdcp key register 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**SGRF\_HDCP\_KEY\_REG2**

Address: Operational Base + offset (0x0288)

hdcp key register 2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**SGRF\_HDCP\_KEY\_REG3**

Address: Operational Base + offset (0x028c)

hdcp key register 3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**SGRF\_HDCP\_KEY\_REG4**

Address: Operational Base + offset (0x0290)

hdcp key register 4

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**SGRF\_HDCP\_KEY\_REG5**

Address: Operational Base + offset (0x0294)

hdcp key register 5

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**SGRF\_HDCP\_KEY\_REG6**

Address: Operational Base + offset (0x0298)

hdcp key register 6

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**SGRF\_HDCP\_KEY\_REG7**

Address: Operational Base + offset (0x029c)

hdcp key register 7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**SGRF\_HDCP\_KEY\_ACCESS\_MASK**

Address: Operational Base + offset (0x02a0)

hdcp key access mask register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x0	reserved

**DDR\_SGRF\_DDR\_CON0**

Address: Operational Base + offset (0x0000)

DDR Secure Control Register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15	RO	0x0	reserved
14	RW	0x1	rgn0_en DDR secure region0 enable 1'b1: enable 1'b0: disable
13	RO	0x0	reserved
12:0	RW	0x0000	rgn0_base DDR secure region0 start address[32:20]

**DDR\_SGRF\_DDR\_CON1**

Address: Operational Base + offset (0x0004)

DDR Secure Control Register 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:13	RO	0x0	reserved
12:0	RW	0x1fff	rgn0_top DDR secure region0 end address[32:20]

**DDR\_SGRF\_DDR\_CON2**

Address: Operational Base + offset (0x0008)

DDR Secure Control Register 2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15	RO	0x0	reserved
14	RW	0x0	rgn1_en DDR secure region1 enable 1'b1: enable 1'b0: disable
13	RO	0x0	reserved
12:0	RW	0x0000	rgn1_base DDR secure region1 start address[32:20]

**DDR\_SGRF\_DDR\_CON3**

Address: Operational Base + offset (0x000c)

DDR Secure Control Register 3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:13	RO	0x0	reserved
12:0	RW	0x0000	rgn1_top DDR secure region1 end address[32:20]

**DDR\_SGRF\_DDR\_CON4**

Address: Operational Base + offset (0x0010)

DDR Secure Control Register 4

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15	RO	0x0	reserved
14	RW	0x0	<p>rgn2_en</p> <p>DDR secure region2 enable</p> <p>1'b1: enable</p> <p>1'b0: disable</p>
13	RO	0x0	reserved
12:0	RW	0x0000	<p>rgn2_base</p> <p>DDR secure region2 start address[32:20]</p>

**DDR\_SGRF\_DDR\_CON5**

Address: Operational Base + offset (0x0014)

DD Secure Control Register 5

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable</p> <p>bit0~15 write enable</p> <p>When bit 16=1, bit 0 can be written by software .</p> <p>When bit 16=0, bit 0 cannot be written by software;</p> <p>When bit 17=1, bit 1 can be written by software .</p> <p>When bit 17=0, bit 1 cannot be written by software;</p> <p>.....</p> <p>When bit 31=1, bit 15 can be written by software .</p> <p>When bit 31=0, bit 15 cannot be written by software;</p>
15:13	RO	0x0	reserved
12:0	RW	0x0000	<p>rgn2_top</p> <p>DDR secure region2 end address[32:20]</p>

**DDR\_SGRF\_DDR\_CON6**

Address: Operational Base + offset (0x0018)

DD Secure Control Register 6

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15	RO	0x0	reserved
14	RW	0x0	<p>rgn3_en DDR secure region3 enable 1'b1: enable 1'b0: disable</p>
13	RO	0x0	reserved
12:0	RW	0x0000	rgn3_base DDR secure region3 start address[32:20]

**DDR\_SGRF\_DDR\_CON7**

Address: Operational Base + offset (0x001c)

DDR Secure Control Register 7

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:13	RO	0x0	reserved
12:0	RW	0x0000	rgn3_top DDR secure region3 end address[32:20]

**DDR\_SGRF\_DDR\_CON8**

Address: Operational Base + offset (0x0020)

DDR Secure Control Register 8

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15	RO	0x0	reserved
14	RW	0x0	<p>rgn4_en DDR secure region4 enable 1'b1: enable 1'b0: disable</p>
13	RO	0x0	reserved
12:0	RW	0x0000	rgn4_base DDR secure region4 start address[32:20]

**DDR\_SGRF\_DDR\_CON9**

Address: Operational Base + offset (0x0024)

DDR Secure Control Register 9

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:13	RO	0x0	reserved
12:0	RW	0x0000	rgn4_top DDR secure region4 end address[32:20]

**DDR\_SGRF\_DDR\_CON10**

Address: Operational Base + offset (0x0028)

DDR Secure Control Register 10

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15	RO	0x0	reserved
14	RW	0x0	rgn5_en DDR secure region5 enable 1'b1: enable 1'b0: disable
13	RO	0x0	reserved
12:0	RW	0x0000	rgn5_base DDR secure region5 start address[32:20]

**DDR\_SGRF\_DDR\_CON11**

Address: Operational Base + offset (0x002c)

DDR Secure Control Register 11

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;
15:13	RO	0x0	reserved
12:0	RW	0x0000	rgn5_top DDR secure region5 end address[32:20]

**DDR\_SGRF\_DDR\_CON12**

Address: Operational Base + offset (0x0030)

DDR Secure Control Register 12

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15	RO	0x0	reserved
14	RW	0x0	<p>rgn6_en DDR secure region6 enable 1'b1: enable 1'b0: disable</p>
13	RO	0x0	reserved
12:0	RW	0x0000	rgn6_base DDR secure region6 start address[32:20]

**DDR\_SGRF\_DDR\_CON13**

Address: Operational Base + offset (0x0034)

DDR Secure Control Register 13

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15:13	RO	0x0	reserved
12:0	RW	0x0000	rgn6_top DDR secure region6 end address[32:20]

**DDR\_SGRF\_DDR\_CON14**

Address: Operational Base + offset (0x0038)

DDR Secure Control Register 14

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15	RO	0x0	reserved
14	RW	0x0	<p>rgn7_en DDR secure region7 enable 1'b1: enable 1'b0: disable</p>
13	RO	0x0	reserved
12:0	RW	0x0000	<p>rgn7_base DDR secure region7 start address[32:20]</p>

**DDR\_SGRF\_DDR\_CON15**

Address: Operational Base + offset (0x003c)

DDR Secure Control Register 15

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RW	0x0000	<p>write_enable bit0~15 write enable When bit 16=1, bit 0 can be written by software . When bit 16=0, bit 0 cannot be written by software; When bit 17=1, bit 1 can be written by software . When bit 17=0, bit 1 cannot be written by software; ..... When bit 31=1, bit 15 can be written by software . When bit 31=0, bit 15 cannot be written by software;</p>
15	RW	0x0	<p>glb_bp_s global secure control logic bypass</p>
14	RW	0x0	<p>rgnx_s rgnx_s other ddr space secure control bit</p>
13	RO	0x0	reserved
12:0	RW	0x0000	<p>rgn7_top DDR secure region7 end address[32:20]</p>

**DDR\_SGRF\_DDR\_CON16**

Address: Operational Base + offset (0x0040)

DDR Secure Control Register 16

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:8	RW	0xff	gpu_m GPU access DDR secure control 0: ddr is un-secure 1: ddr is secure
7:0	RW	0xff	cpu_m CPU access DDR secure control 0: ddr is un-secure 1: ddr is secure

**DDR\_SGRF\_DDR\_CON17**

Address: Operational Base + offset (0x0044)

DDR Secure Control Register 17

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:8	RW	0xff	crypto_tsp_m CRYPTO/TSP access DDR secure control 0: ddr is un-secure 1: ddr is secure
7:0	RW	0xff	dma_m DMA access DDR secure control 0: ddr is un-secure 1: ddr is secure

**DDR\_SGRF\_DDR\_CON18**

Address: Operational Base + offset (0x0048)

DDR Secure Control Register 18

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:8	RW	0xff	gmac_m GMAC access DDR secure control 0: ddr is un-secure 1: ddr is secure
7:0	RW	0xff	emmc_m EMMC/SDIO/SDMMC/HOST0/1/2/3/OTG access DDR secure control 0: ddr is un-secure 1: ddr is secure

**DDR\_SGRF\_DDR\_CON19**

Address: Operational Base + offset (0x004c)

DDR Secure Control Register 19

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:8	RW	0xff	vop_m VOP access DDR secure control 0: ddr is un-secure 1: ddr is secure
7:0	RW	0xff	nand_m NAND access DDR secure control 0: ddr is un-secure 1: ddr is secure

**DDR\_SGRF\_DDR\_CON20**

Address: Operational Base + offset (0x0050)

DDR Secure Control Register 20

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:8	RW	0xff	hdcp_m HDCP access DDR secure control 0: ddr is un-secure 1: ddr is secure
7:0	RW	0xff	iep_m IEP access DDR secure control 0: ddr is un-secure 1: ddr is secure

**DDR\_SGRF\_DDR\_CON21**

Address: Operational Base + offset (0x0054)

DDR Secure Control Register 21

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:8	RW	0xff	vpu_m VPU access DDR secure control 0: ddr is un-secure 1: ddr is secure
7:0	RW	0xff	rga_m RGA access DDR secure control 0: ddr is un-secure 1: ddr is secure

**DDR\_SGRF\_DDR\_CON22**

Address: Operational Base + offset (0x0058)

DDR Secure Control Register 22

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
7:0	RW	0xff	rkvdec_m RKVDEC access DDR secure control 0: ddr is un-secure 1: ddr is secure

Rockchip Confidential

## Chapter 7 Interconnect

### 7.1 Overview

The chip-level interconnect consists of one interconnect. It enables communication among the modules and subsystems in the device.

The interconnect handles many types of data transfers. It transfers data with a maximum width of 128 bits from the initiator to the target. It is a little-endian platform.

### 7.2 Block Diagram

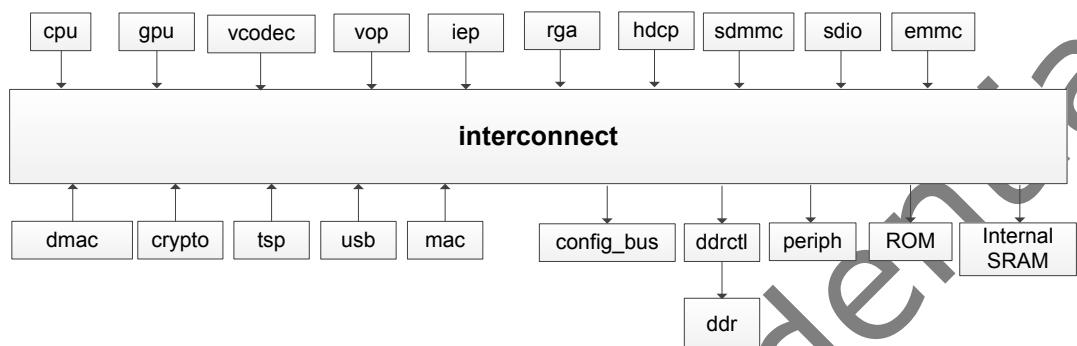


Fig. 7-1 Interconnect diagram

### 7.3 Function Description

#### 7.3.1 Master & Slave

The main interconnect is connected with all the related IPs of the system, the interface between the IP and the interconnect is called as NIU(native interface unit). All the connected NIU are list as bellowing:

Table 7-1 Master NIU

<b>Master NIU</b>	<b>Description</b>
cpu	CPU port, access to any peripheral device
dmac	DMAC master
crypto	Crypto master, used in trustzone technolgy
tsp	TSP master
gpu	GPU master
vcodec	Video codec master
vop	VOP master
iep	IEP master
rga	RGA master
hdcp	HDCP master
sdmmc	SDMMC master
sdio	SDIO master
emmc	eMMC master
mac	GMAC master

Master NIU	Description
usb	USB master

### 7.3.2 QoS management

The interconnect offers 4 modes of qos management:

- None, QoSGenerator is disabled, and priority information are stuck at 0.
- Fixed, QoSGenerator drives applies a fixed urgency to read transactions, and a (possibly different) urgency to write transactions.
- Limiter, QoSGenerator behaves as in fixed mode, but limits the traffic bandwidth coming from that socket, possibly stalling requests if the initiator attempts to exceed its budget.
- Regulator, QoSGenerator promotes are demotes hurry, depending the bandwidth obtained by the initiator is below or beyond a bandwidth budget. As transactions exceeding the bandwidth limit are sent (even though demoted), the regulator mode may be considered as a softer version of the limiter mode.

#### Limiter Behavior

When configured in bandwidth limiter, the unit uses a 23 bit counter to measure the average bandwidth. This counter has a 1/256 byte resolution and works as follows:

- Adds the number of byte rounded up to 16 (1 -> 16) and then multiplied by 256 to the current value, each time a request is sent.
- Subtracts the Bandwidth register value every cycle. If the Counter becomes negative, force it to 0.
- If the Counter value is greater than the Saturation register value multiplied by 16\*256, any incoming request is stalled until this condition disappears. Note that the Counter cannot wrap-around because the maximum value it can reach is: SaturationMax\*16\*256 + BurstMax\*256 = 1023\*4K+4K\*256 = 5116K or 223 = 8192K.

The following example will show the Counter behavior: 32 byte bursts, F=400MHz, BW=200MB/s, T=0.32us. The Bandwidth register will be set to 256\*200/400 = 128, and the Saturation register to 128\*0.32\*400/4096 = 4 (which corresponds to 64 bytes).

#### Regulator Behavior

When configured in bandwidth regulator, the unit uses a 23 bit counter to measure the average bandwidth. This counter has a 1/256 byte resolution and works as follows:

- Adds the number of byte rounded up to 16 and then multiplied by 256 to the current value, each time a response is received. If the result is greater than the Saturation register value multiplied by 16\*256, saturation to this value is applied.
- Subtracts the Bandwidth register value every cycle. If the Counter becomes negative, force it to 0.
- If the Counter value is less than or equal to the Saturation register value multiplied by 16\*256/2, the SocketMst Hurry signal will be set to the HurryHigh register, and HurryLow otherwise. Note that Urgency and Press will be also set to the same value.

The following example will show the Counter behavior: 1Kbyte bursts, F=500MHz, BW=2GB/s, T=2.048us. The Bandwidth register will be set to 256\*2000/500 = 1024, and the Saturation register to 1024\*2.048\*500/4096 = 256 (which corresponds to 4 Kbytes).

#### QoS Generator Programming

Bandwidth: This  $\log_2(\text{socket.wData}/8)+8$  bits register defines the bandwidth in 1/256th byte per cycle unit. This allows a 2 MByte/s resolution at 500MHz. When the bandwidth is given in MByte/s, the value of this register will be equal to  $256 \times \text{BWMB/s} / \text{FMHz}$ .

Saturation: This 10 bits register defines the number of byte used for bandwidth measurement. It is expressed in 16bytes unit (up to 16 Kbyte). Usually the integration window is given in us or in cycle: the value of this register will be equal to  $\text{Bandwidth} \times \text{Tus} \times \text{FMHz} / (256 \times 16)$  or  $\text{Bandwidth} \times \text{Ncycle} / (256 \times 16)$ .

### 7.3.3 Memory Scheduler

Memory scheduler is a special NIU of the interconnect, it mainly deal with the transaction inside the interconnect and convert it to the transaction which the ddr protocol controller can recognize.

Following table shows the software configurable setting for the memory scheduler when the

system connected to different size of ddr device.

The DDRCONF[3:0] is a configurable register inside the interconnect.

R: indicates Row bits

B: indicates Bank bits

C: indicates Column bits

D: indicates Chip selects bits

Table 7-2 DDR configuration

<b>DDR CONF[3:0]</b>	
0	C RRRD RRRR RRRR RRRR RBBC CCCC CCCC ----
1	C RRDR RRRR RRRR RRRR RBBC CCCC CCCC ----
2	C RDRA RRRR RRRR RRRR RBBC CCCC CCCC ----
3	C DRRA RRRR RRRR RRRR RBBC CCCC CCCC ----
4	R RDRA RRRR RRRR RRRR BBBC CCCC CCCC ----
5	R DRRR RRRR RRRR RRRR BBBC CCCC CCCC ----
6	D RRRR RRRR RRRR RRRR BBBC CCCC CCCC ----
7	C CRRR DRRA RRRR RRRR RBBC BCCC CCCC ----
8	C CRRD RRRR RRRR RRRR RBBC BCCC CCCC ----
9	C CRDR RRRR RRRR RRRR RBBC BCCC CCCC ----
10	C CDRA RRRR RRRR RRRR RBBC BCCC CCCC ----
11	C RDRA BRRA RRRR RRRR RRRR CCCC CCCC ----
12	R RDRA BRRA RRRR RRRR RRRR CCCC CCCC ----
13	C RRDB BBRA RRRR RRRR RRRR CCCC CCCC ----
14	C RRRR RRRR RRRR RRRR DBBB CCCC CCCC ----
15	C CRRR RRRR RRRR RRRR RDRA BCCC CCCC ----

## 7.4 Register Description

### 7.4.1 Address Mapping

Table 7-3 Service Module Address

<b>service Module</b>	<b>Base address</b>
service_msch	0x3109_0000
service_rkvdec	0x3107_0000
service_vop	0x3106_0000
service_gpu	0x3105_0000
service_vpu	0x3104_0000
service_vio	0x3103_0000
service_bus	0x3102_0000
service_peri	0x3101_0000
service_core	0x3100_0000

### 7.4.2 Registers Summary

#### service\_msch:

<b>Name</b>	<b>Offset</b>	<b>Size</b>	<b>Reset Value</b>	<b>Description</b>
coreid	0x0000	W	0xe5a68c02	Core ID register

Name	Offset	Size	Reset Value	Description
revisionid	0x0004	W	0x00018100	Revision ID register
ddrconf	0x0008	W	0x00000000	ddr configuration definition.
ddrtiming	0x000c	W	0x1c514256	ddr timing definition.
ddrmode	0x0010	W	0x00000000	ddr mode definition.
readlatency	0x0014	W	0x00000028	read latency register
activate	0x0038	W	0x00000400	activate register
devtodev	0x003c	W	0x00000000	devtodev register

**service\_core:**

Name	Offset	Size	Reset Value	Description
Cpu_qos_Priority	0x0008	W	0x00000005	Priority register
Cpu_qos_Mode	0x000c	W	0x00000003	Mode register
Cpu_qos_Bandwidth	0x0010	W	0x00000040	Bandwidth register
Cpu_qos_Saturation	0x0014	W	0x00000040	Saturation register
Cpu_qos_ExtControl	0x0018	W	0x00000000	External inputs control

Notes: Size: **B**- Byte (8 bits) access, **HW**- Half WORD (16 bits) access, **W**-WORD (32 bits) access  
other master's qos register is same with service\_core:cpup\_qos

### 7.4.3 Detail Register Description

**service\_msch: memory scheduler coreid**

Address: Operational Base + offset (0x0000)

Bit	Attr	Reset Value	Description
31:0	RO	0xe5a68c02	Core type id

**service\_msch: memory scheduler revisionid**

Address: Operational Base + offset (0x0004)

Bit	Attr	Reset Value	Description
31:0	RO	0x00018100	msch' revision id

**service\_msch: memory scheduler ddrconf**

Address: Operational Base + offset (0x0008)

Memory scheduler configuration register

Bit	Attr	Reset Value	Description
31:4	RO	0x0	reserved
3:0	RW	0x0	DdrConf select the ddr rank,row,bank,col sequence

**service\_msch: memory scheduler ddrtiming**

Address: Operational Base + offset (0x000c)

Memory scheduler timing register

Bit	Attr	Reset Value	Description
31	RW	0x0	BwRatio When set to zero, one DRAM clock cycle (two DDR transfers) is used to transfer each word of data. When set to one, two DRAM clock cycles (four DDR transfers) are used to transfer each word of data. This is applicable when half of a DRAM data bus width is used.

Bit	Attr	Reset Value	Description
30:26	RW	0x07	WrToRd The minimum number of scheduler clock cycles between the last DRAM Write command and a Read command (WL x tCkD + tWTR). tCkD is the DRAM clock period.
25:21	RW	0x02	RdToWr The minimum number of scheduler clock cycles between the last DRAM Read command and a Write command (DDR2: 2 x tCkD, DDR3: (RL - WL + 2) x tCkD). tCkD is the DRAM clock period.
20:18	RW	0x4	BurstLen The DRAM burst duration on the DRAM data bus in scheduler clock cycles. Also equal to scheduler clock cycles between two DRAM commands (BL / 2 x tCkD). tCkD is the DRAM clock period.
17:12	RW	0x14	WrToMiss The minimum number of scheduler clock cycles between the last DRAM Write command and a new Read or Write command in another page of the same bank (WL x tCkD + tWR + tRP + tRCD). tCkD is the DRAM clock period.
11:6	RW	0x09	RdToMiss The minimum number of scheduler clock cycles between the last DRAM Read command and a new Read or Write command in another page of the same bank (tRTP + tRP + tRCD - BL x tCkD / 2). tCkD is the DRAM clock period.
5:0	RW	0x16	ActToAct The minimum number of scheduler clock cycles between two consecutive DRAM Activate commands on the same bank (tRC/tCkG). tCkG is the clock period of the SoC DRAM scheduler.

**service\_msch: memory scheduler ddrmode**

Address: Operational Base + offset (0x0010)

Memory scheduler mode register

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved
1	RW	0x0	BwRatioExtended When set to 1, support for 4x Bwratio.
0	RW	0x0	AutoPrecharge When set to one, pages are automatically closed after each access, when set to zero, pages are left opened until an access in a different page occurs

**service\_msch: memory scheduler readlatency**

Address: Operational Base + offset (0x0014)

Memory scheduler read latency register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved

Bit	Attr	Reset Value	Description
7:0	RW	0x28	ReadLatency The DRAM type-specific number of cycles from a scheduler request to a protocol controller response. This is a fixed value depending on the type of DRAM memory. <See SoC-specific memory controller documentation>.

**service\_msch: memory scheduler activate**

Address: Operational Base + offset (0x0038)

Memory scheduler activate register

Bit	Attr	Reset Value	Description
31:11	RO	0x0	reserved
10	RW	0x1	FawBank The number of Banks of a given device involved in the FAW period. Set to zero for 2-bank memories (WideIO). Set to one for memories with 4 banks or more (DDR).
9:4	RW	0x00	Faw The number of cycles for the four bank activate (FAW) period (tFAW).
3:0	RW	0x0	Rrd 'The number of cycles between two consecutive Activate commands on different Banks of the same device (tRRD).

**service\_msch: memory scheduler devtodev**

Address: Operational Base + offset (0x003c)

Memory scheduler devtodev register

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5:4	RW	0x0	BusWrToRd Field0001 Abstract The number of cycles between the last write data to a device and the first read data of another device of a memory array with multiple ranks (2 x tCkD). tCkD is the DRAM clock period.
3:2	RW	0x0	BusRdToWr Field0000 Abstract The number of cycles between the last read data of a device and the first write data to another device of a memory array with multiple ranks (2 x tCkD). tCkD is the DRAM clock period.
1:0	RW	0x0	BusRdToRd The number of cycles between the last read data of a device and the first read data of another device of a memory array with multiple ranks (tCkD). tCkD is the DRAM clock period.

Following is CPU port's QoS register detail description. Other ports have the same register. The only different is the base address's offset.

**service\_core: Cpu\_qos\_Priority**

Address: Operational Base + offset (0x0008)

Priority register

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	RO	0x0	Mark Backward compatibility marker when 0.
30:4	RO	0x0	reserved
3:2	RW	0x1	P1 In Programmable or Bandwidth Limiter mode, the priority level for read transactions. In Bandwidth regulator mode, the priority level when the used throughput is below the threshold. In Bandwidth Regulator mode, P1 should have a value equal or greater than P0.
1:0	RW	0x1	P0 In Programmable or Bandwidth Limiter mode, the priority level for write transactions. In Bandwidth Regulator mode, the priority level when the used throughput is above the threshold. In Bandwidth Regulator mode, P0 should have a value equal or lower than P1.

**service\_core: Cpu\_qos\_Mode**

Address: Operational Base + offset (0x000c)

Mode register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:2	RO	0x0	reserved
1:0	RW	0x3	Mode 0 = Programmable mode: a programmed priority is assigned to each read or write, 1 = Bandwidth Limiter Mode: a hard limit restricts throughput, 2 = Bypass mode: (<See SoC-specific QoS generator documentation>), 3 = Bandwidth Regulator mode: priority decreases when throughput exceeds a threshold.

**service\_core: Cpu\_qos\_Bandwidth**

Address: Operational Base + offset (0x0010)

Bandwidth register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0040	Bandwidth In Bandwidth Limiter or Bandwidth Regulator mode, the bandwidth threshold in units of 1/256th bytes per cycle. For example, 80 MBps on a 250 MHz interface is value 0x0052.

**service\_core: Cpu\_qos\_Saturation**

Address: Operational Base + offset (0x0014)

Saturation register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:10	RO	0x0	reserved

Bit	Attr	Reset Value	Description
9:0	RW	0x040	Saturation In Bandwidth Limiter or Bandwidth Regulator mode, the maximum data count value, in units of 16 bytes. This determines the window of time over which bandwidth is measured. For example, to measure bandwidth within a 1000 cycle window on a 64-bit interface is value 0x1F4.

**service\_core: cpu\_qos Saturation**

Address: Operational Base + offset (0x0018)

External inputs control

Bit	Attr	Reset Value	Description
31:3	RO	0x0	reserved
2	RW	0x0	Replace the External reference by the local clock.
1	RW	0x0	ExtThr input controls Low/High priority instead of bandwidth threshold.
0	RW	0x0	Combines the Socket QoS with Regulator QoS

## 7.5 Application Notes

### 7.5.1 QoS setting

The CPU read channel, VOP read channel, GPU write channel have the external QoS control. After reset each master port both have priority setting as 1. It's recommended that field 0 of QoS. ExtControl set to 1 to enable the external qos control. And priority setting of each master kept at 1.

### 7.5.2 Idle request

The main interconnect supports flushing the ongoing transaction when the software needed to do so.

If the GPU power domain need to disconnect from the main interconnect, Idle request has to be sent to GPU NIU, the NIU will respond a ack, and when it's ready to be disconnect, one Idle signal will be send out . Then, if GPU still have transaction to be sent to the memory scheduler, it will be stalled by the NIU.

If the GPU system power domain is disconnected as the above flow, then CPU want to access to the GPU system, it will response error to CPU.

The sequence is like following figure shows:

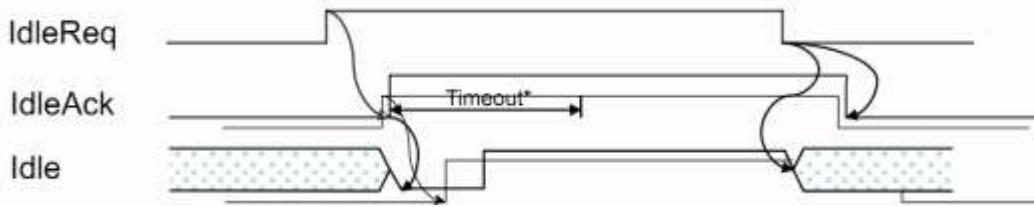


Fig. 7-2 Idle request

The idle request is set by GRF register.

## Chapter 8 DMC (Dynamic Memory Interface)

### 8.1 Overview

The DMC includes two section: dynamic ram protocol controller(PCTL) and PHY. The PCTL SoC application bus interface supports a lowest-latency native application interface (NIF). To maximize data transfer efficiency, NIF commands transfer data without flow control. To simplify command processing, the NIF accepts addresses in rank, bank, row, column format.

The DDR PHY provides control features to ease the customer implementation of digitally controlled features of the PHY such as initialization, DQS gate training, write leveling training and programmable configuration controls. The DDR PHY has built-in self test features to provide support for production testing of the compatible PHY. It also provides a DFI 2.1.1 interface to the PCTL.

The DMC supports the following features:

- Complete, integrated DDR3, LPDDR3 solution
- DFI 2.1.1 interface compatibility
- Up to 1600 Mbps in 1:2 frequency ratio for DDR3 and LPDDR3, using a 400MHz controller clock and 800MHz memory clock.
- Support for x16, x32 DDR3 memories, for a total memory data path width of 32 bits
- Support for x16, x32 LPDDR3 memories, for a total memory data path width of 32 bits
- Up to 2 memory ranks; devices within a rank tie to a common chip select
- Up to 8 open memory banks, maximum of eight per rank
- Per-NIF transaction controllable bank management policies: open-page, close-page
- Low area, low power architecture with minimal buffering on the data, avoiding duplication of storage resources within the system
- PCTL NIF slave interface facilitates easy integration with an external scheduler or standard on-chip buses
- Efficient DDR protocol implementation with in-order column (Read and Write) commands and out-of-order Activate and Precharge commands
- Three clock cycles best case command latency (best case is when a command is to an open page and the shift array in the PCTL is empty)
- 1T or 2T memory command timing
- Automatic power-down and self-refresh entry and exit
- Software and hardware driven self-refresh entry and exit
- Programmable memory initialization
- Partial population of memories, where not all DDR byte lanes are populated with memory chips
- Programmable per rank memory ODT (On-Die Termination) support for reads and writes
- APB interface for PCTL and PHY software-accessible registers
- Automatic DQS gate training
- Automatic Write Leveling training
- At-speed built-in-self-test (BIST) loopback testing on both the address and data channels for DDR PHYs

## 8.2 Block Diagram

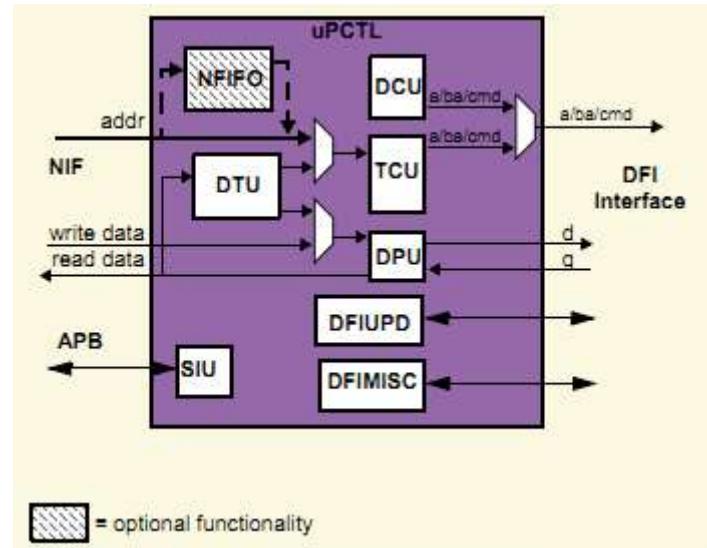


Fig. 8-1 Protocol controller architecture

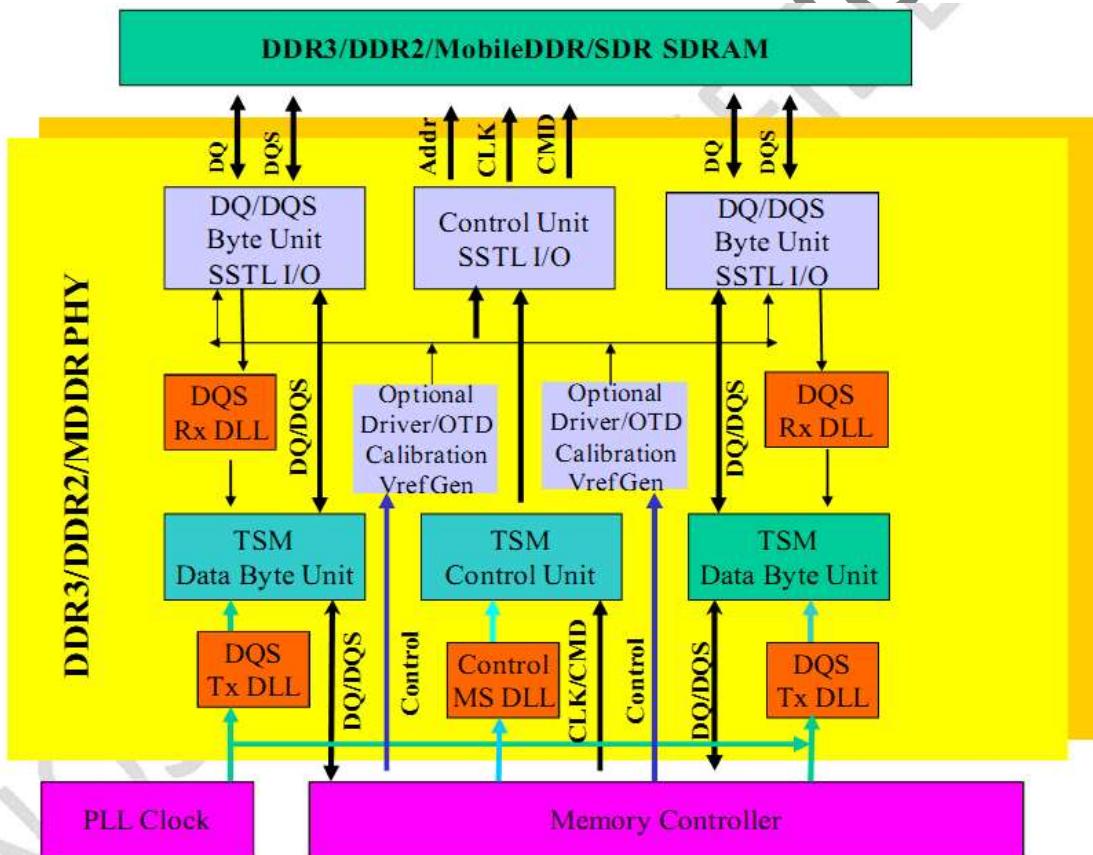


Fig. 8-2 PHY controller architecture

## 8.3 Function Description

### 8.3.1 Protocol controller(PCTL)

PCTL operations are defined in terms of the current state of the Operational State Machine. Software can move PCTL in any of the operational states by issuing commands via the SCTL register. Transitions from one operational state to the other occur pass through a “transitional” state. Transitional states are exited automatically by the PCTL after all the necessary actions required to change operational state have been completed. The current operational state of PCTL is reported by the STAT register and is also available from the p\_ctl\_stat output.

PCTL supports the following operational states:

- Init\_mem - This state is the default state entered after reset. All writable registers can be programmed. While in this state software can program PCTL and initialize the PHY and the memories. The memories are not refreshed and data that has previously been written to the memories may be lost as a result. The Init\_mem state is also used when it is desirable to stop any automatic PCTL function that directly affects the memories, like Power Down and Refresh, or when a software reset of the memory subsystem has to be executed.
- ConFig- This state is used to suspend temporarily the normal NIF traffic and allow software to reprogram PCTL and memories if necessary, while still keeping active the periodic generation of Refresh cycles to the memories. Power Down entry and exit sequences are possible while in ConFigstate.
- Access - This is the operational state where NIF transactions are accepted by the PCTL and converted into memory read and writes. None of the registers can be programmed except SCFG, SCTL, ECCCLR and DTU\* registers.
- Low\_power - Memories are in self-refresh mode. The PCTL does not generate refresh cycles while in this state.

Access and Low\_power states can also be entered and exited by the hardware low power signals (c\_\*)<sup>1</sup>. In case of conflicting software and hardware low-power commands, the resulting operational state taken by the controller can be either one of the two conflicting requests.

Figure 1-3 illustrates the operational and transitional states.

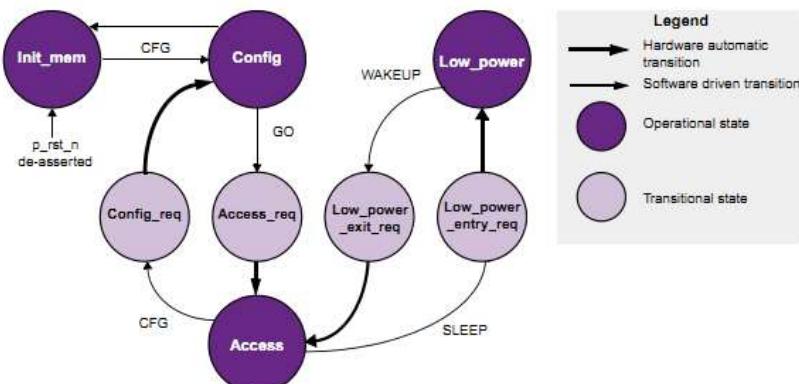


Fig. 8-3 Protocol controller architecture

The controller clock is the same clock driving the memory controller and will be the same frequency as the SDRAM clock (ck). The configuration clock can run at a frequency equal to or less than the controller clock. The configuration clock drives all non-DDR timing logic, such as configuration registers, PHY initialization, output impedance, and so on.

### 8.3.2 DDR PHY

DDR PHY provides turn key physical interface solutions for chip requiring access to DDR3 /LPDDR3 SDRAM device. It is optimized for low power and high speed (up to 1600Mbps for DDR3 and LPDDR3) applications with robust timing and small silicon area. It supports DDR3 and LPDDR3 SDRAM components in the market. The PHY components contain DDR specialized functional and utility SSTL I/Os up to 800MHz, critical timing synchronization module (TSM) and a low power/jitter DLLs with programmable fine-grain control for any SDRAM interface. DDR PHY uses a DFI digital interface to connect the memory controller. All interface timing is in (1/2)XCK(SDRAM) clock domain. The controller to PHY interface is running at (1/2)XCK(SDRAM) therefore read/write bus is four times width. DDR muxing is done in the PHY block together with all related per-byte lane timing adjustment. The interface is fairly generic and support high performance input and output data flow gearing toward 200Mbps to 1600Mbps DDR3, and LPDDR3 SDRAM speed in wide range.

With configurable timing and driving strength and ODT parameters to interface to the wide variety of SDRAMs, the PHY is very flexible with advanced command capability to increase SDRAM operation efficiency.

## 8.4 Register Description

Slave address can be divided into different length for different usage, which is shown as follows.

### 8.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
DDR_PCTL_SCFG	0x0000	W	0x00000300	State Configuration Register
DDR_PCTL_SCTL	0x0004	W	0x00000000	Operational State Control Register
DDR_PCTL_STAT	0x0008	W	0x00000000	Operational State Status Register
DDR_PCTL_INTRSTAT	0x000c	W	0x00000000	Interrupt Status Register
DDR_PCTL_MCMD	0x0040	W	0x00100000	Memory Command Register
DDR_PCTL_POWCTL	0x0044	W	0x00000000	Power Up Control Register
DDR_PCTL_POWSTAT	0x0048	W	0x00000000	Power Up Status Register
DDR_PCTL_CMDTSTAT	0x004c	W	0x00000000	Command Timers Status Register
DDR_PCTL_CMDTSTATEN	0x0050	W	0x00000000	Command Timers Status Enable Register
DDR_PCTL_MRRCFG0	0x0060	W	0x00000000	Mode Register Read Configuration 0
DDR_PCTL_MRRSTAT0	0x0064	W	0x00000000	Mode Register Read Status 0 Register
DDR_PCTL_MRRSTAT1	0x0068	W	0x00000000	Mode Register Read Status 1 Register
DDR_PCTL_MCFG	0x0080	W	0x00040020	Memory Configuration Register
DDR_PCTL_PPCFG	0x0084	W	0x00000000	Partially Populated Memories Configuration Register
DDR_PCTL_MSTAT	0x0088	W	0x00000000	Memory Status Register
DDR_PCTL_MCFG1	0x0090	W	0x00000000	Memory Configuration 1 Register
DDR_PCTL_DTUPDES	0x0094	W	0x00000000	DTU Status Register
DDR_PCTL_DTUNA	0x0098	W	0x00000000	DTU Number of Addresses Created Register
DDR_PCTL_DTUNE	0x009c	W	0x00000000	DTU Number of Errors Register
DDR_PCTL_DTUPRD0	0x00a0	W	0x00000000	DTU Parallel Read 0 Register
DDR_PCTL_DTUPRD1	0x00a4	W	0x00000000	DTU Parallel Read 1 Register
DDR_PCTL_DTUPRD2	0x00a8	W	0x00000000	DTU Parallel Read 2 Register
DDR_PCTL_DTUPRD3	0x00ac	W	0x00000000	DTU Parallel Read 3 Register
DDR_PCTL_DTUAWDT	0x00b0	W	0x00000290	DTU Address Width Register
DDR_PCTL_TOGCNT1U	0x00c0	W	0x00000064	Toggle Counter 1us Register
DDR_PCTL_TINIT	0x00c4	W	0x000000c8	t_init Timing Register
DDR_PCTL_TRSTH	0x00c8	W	0x00000000	t_rsth Timing Register
DDR_PCTL_TOGCNT100N	0x00cc	W	0x00000001	Toggle Counter 100ns
DDR_PCTL_TREFI	0x00d0	W	0x00000001	t_refi Timing Register
DDR_PCTL_TMRD	0x00d4	W	0x00000001	t_mrd Timing Register
DDR_PCTL_TRFC	0x00d8	W	0x00000001	
DDR_PCTL_TRP	0x00dc	W	0x00010006	t_trp Timing Register
DDR_PCTL_TRTW	0x00e0	W	0x00000002	t_rtw Timing Register

Name	Offset	Size	Reset Value	Description
DDR_PCTL_TAL	0x00e4	W	0x00000000	AL Register
DDR_PCTL_TCL	0x00e8	W	0x00000004	CL Timing Register
DDR_PCTL_TCWL	0x00ec	W	0x00000003	CWL Timing Register
DDR_PCTL_TRAS	0x00f0	W	0x00000010	t_ras Timing Register
DDR_PCTL_TRC	0x00f4	W	0x00000016	t_rc Timing Register
DDR_PCTL_TRCD	0x00f8	W	0x00000006	t_rcd Timing Register
DDR_PCTL_TRRD	0x00fc	W	0x00000004	t_rrd Timing Register
DDR_PCTL_TRTP	0x0100	W	0x00000003	t_rtp Timing Register
DDR_PCTL_TWR	0x0104	W	0x00000006	t_wr Register
DDR_PCTL_TWTR	0x0108	W	0x00000004	t_wtr Timing Register
DDR_PCTL_TEXSR	0x010c	W	0x00000001	t_exsr Timing Register
DDR_PCTL_TXP	0x0110	W	0x00000001	t_xp Timing Register
DDR_PCTL_TXPDLL	0x0114	W	0x00000000	t_xpdll Timing Register
DDR_PCTL_TZQCS	0x0118	W	0x00000000	t_zqcs Timing Register
DDR_PCTL_TZQCSI	0x011c	W	0x00000000	t_zqcsi Timing Register
DDR_PCTL_TDQS	0x0120	W	0x00000001	t_dqs Timing Register
DDR_PCTL_TCKSRE	0x0124	W	0x00000000	t_cksrc Timing Register
DDR_PCTL_TCKSRX	0x0128	W	0x00000000	t_cksrcx Timing Register
DDR_PCTL_TCKE	0x012c	W	0x00000003	t_cke Timing Register
DDR_PCTL_TMOD	0x0130	W	0x00000000	t_mod Timing Register
DDR_PCTL_TRSTL	0x0134	W	0x00000000	Reset Low Timing Register
DDR_PCTL_TZQCL	0x0138	W	0x00000000	t_zqcl Timing Register
DDR_PCTL_TMRR	0x013c	W	0x00000002	t_mrr Timing Register
DDR_PCTL_TCKESR	0x0140	W	0x00000004	t_ckesr Timing Register
DDR_PCTL_TDPD	0x0144	W	0x00000000	t_dpd Timing Register
DDR_PCTL_DTUWACTL	0x0200	W	0x00000000	DTU Write Address Control
DDR_PCTL_DTURACTL	0x0204	W	0x00000000	DTU Read Address Control Register
DDR_PCTL_DTUCFG	0x0208	W	0x00000000	DTU Configuration Control Register
DDR_PCTL_DTUECTL	0x020c	W	0x00000000	DTU Execute Control Register
DDR_PCTL_DTUWD0	0x0210	W	0x00000000	DTU Write Data #0 Register
DDR_PCTL_DTUWD1	0x0214	W	0x00000000	DTU Write Data #1 Register
DDR_PCTL_DTUWD2	0x0218	W	0x00000000	DTU Write Data #2 Register
DDR_PCTL_DTUWD3	0x021c	W	0x00000000	DTU Write Data #3 Register
DDR_PCTL_DTUWDM	0x0220	W	0x00000000	DTU Write Data Mask Register
DDR_PCTL_DTURD0	0x0224	W	0x00000000	DTU Read Data #0 Register
DDR_PCTL_DTURD1	0x0228	W	0x00000000	DTU Read Data #1 Register
DDR_PCTL_DTURD2	0x022c	W	0x00000000	DTU Read Data #2 Register
DDR_PCTL_DTURD3	0x0230	W	0x00000000	DTU Read Data #3 Register
DDR_PCTL_DTULFSRWD	0x0234	W	0x00000000	DTU LFSR Seed for Write Data Generation Register
DDR_PCTL_DTULFSRRD	0x0238	W	0x00000000	DTU LFSR Seed for Read Data Generation Register
DDR_PCTL_DTUEAF	0x023c	W	0x00000000	DTU Error Address FIFO Register

Name	Offset	Size	Reset Value	Description
DDR_PCTL_DFITCTRLDELAY	0x0240	W	0x00000002	DFI tctrl_delay Register
DDR_PCTL_DFIODTCFG	0x0244	W	0x00000000	DFI ODT Configuration
DDR_PCTL_DFIODTCFG1	0x0248	W	0x06060000	DFI ODT Timing Configuration 1 (for Latency and Length)
DDR_PCTL_DFIODTRANKMAP	0x024c	W	0x00008421	DFI ODT Rank Mapping
DDR_PCTL_DFITPHYWRDATA	0x0250	W	0x00000001	DFI tphy_wrdata Register
DDR_PCTL_DFITPHYWRLAT	0x0254	W	0x00000001	DFI tphy_wrlat Register
DDR_PCTL_DFITRDDATAEN	0x0260	W	0x00000001	DFI trddata_en Register
DDR_PCTL_DFITPHYRDLAT	0x0264	W	0x0000000f	DFI tphy_rdlat Register
DDR_PCTL_DFITPHYUPDTYPE0	0x0270	W	0x00000010	DFI tphyupd_type0 Register
DDR_PCTL_DFITPHYUPDTYPE1	0x0274	W	0x00000010	DFI tphyupd_type1 Register
DDR_PCTL_DFITPHYUPDTYPE2	0x0278	W	0x00000010	DFI tphyupd_type2 Register
DDR_PCTL_DFITPHYUPDTYPE3	0x027c	W	0x00000010	DFI tphyupd_type3 Register
DDR_PCTL_DFITCTRLUPDMIN	0x0280	W	0x00000010	DFI tctrlupd_min Register
DDR_PCTL_DFITCTRLUPDMAX	0x0284	W	0x00000040	DFI tctrlupd_max Register
DDR_PCTL_DFITCTRLUPDDLY	0x0288	W	0x00000008	DFI tctrlupddly Register
DDR_PCTL_DFIUPDCFG	0x0290	W	0x00000003	DFI Update Configuration Register
DDR_PCTL_DFITREFMSKI	0x0294	W	0x00000000	DFI Masked Refresh Interval
DDR_PCTL_DFITCTRLUPDI	0x0298	W	0x00000000	DFI tctrlupd_interval Register
DDR_PCTL_DFITRCFG0	0x02ac	W	0x00000000	DFI Training Configuration 0 Register
DDR_PCTL_DFITRSTAT0	0x02b0	W	0x00000000	DFI Training Status 0 Register
DDR_PCTL_DFITRWRLVLEN	0x02b4	W	0x00000000	DFI Training dfi_wrlvl_en Register
DDR_PCTL_DFITRRDLVLEN	0x02b8	W	0x00000000	DFI Training dfi_rdlvl_en Register
DDR_PCTL_DFITRRDLVLGATEEN	0x02bc	W	0x00000000	DFI Training dfi_rdlvl_gate_en Register
DDR_PCTL_DFISTSTAT0	0x02c0	W	0x00000000	DFI Status Status 0 Register
DDR_PCTL_DFISTCFG0	0x02c4	W	0x00000000	DFI Status Configuration 0 Register

Name	Offset	Size	Reset Value	Description
DDR_PCTL_DFISTCFG1	0x02c8	W	0x00000000	DFI Status Configuration 1 Register
DDR_PCTL_DFITDRAMCLKEN	0x02d0	W	0x00000002	DFI tdram_clk_enable Register
DDR_PCTL_DFITDRAMCLKDIS	0x02d4	W	0x00000002	DFI tdram_clk_disable Register
DDR_PCTL_DFISTCFG2	0x02d8	W	0x00000000	DFI Status Configuration 2 Register
DDR_PCTL_DFISTPARCLR	0x02dc	W	0x00000000	DFI Status Parity Clear Register
DDR_PCTL_DFISTPARLOG	0x02e0	W	0x00000000	DFI Status Parity Log Register
DDR_PCTL_DFLPCFG0	0x02f0	W	0x00070000	DFI Low Power Configuration 0 Register
DDR_PCTL_DFITRWRLVLERSP0	0x0300	W	0x00000000	DFI Training dfi_wrlvl_resp Status 0 Register
DDR_PCTL_DFITRWRLVLERSP1	0x0304	W	0x00000000	DFI Training dfi_wrlvl_resp Status 1 Register
DDR_PCTL_DFITRWRLVLERSP2	0x0308	W	0x00000000	DFI Training dfi_wrlvl_resp Status 2 Register
DDR_PCTL_DFITRRDLVLERSP0	0x030c	W	0x00000000	DFI Training dfi_rdlvl_resp Status 0 Register
DDR_PCTL_DFITRRDLVLERSP1	0x0310	W	0x00000000	DFI Training dfi_rdlvl_resp Status 1 Register
DDR_PCTL_DFITRRDLVLERSP2	0x0314	W	0x00000000	DFI Training dfi_rdlvl_resp Status 2 Register
DDR_PCTL_DFITRWRLVLDelay0	0x0318	W	0x00000000	DFI Training dfi_wrlvl_delay Configuration 0 Register
DDR_PCTL_DFITRWRLVLDelay1	0x031c	W	0x00000000	DFI Training dfi_wrlvl_delay Configuration 1 Register
DDR_PCTL_DFITRWRLVLDelay2	0x0320	W	0x00000000	DFI Training dfi_wrlvl_delay Configuration 2 Register
DDR_PCTL_DFITRRDLVLDelay0	0x0324	W	0x00000000	DFI Training dfi_rdlvl_delay Configuration 0 Register
DDR_PCTL_DFITRRDLVLDelay1	0x0328	W	0x00000000	DFI Training dfi_rdlvl_delay Configuration 1 Register
DDR_PCTL_DFITRRDLVLDelay2	0x032c	W	0x00000000	DFI Training dfi_rdlvl_delay Configuration 2 Register
DDR_PCTL_DFITRRDLVLGATEDELAY0	0x0330	W	0x00000000	DFI Training dfi_rdlvl_gate_delay Configuration 0
DDR_PCTL_DFITRRDLVLGATEDELAY1	0x0334	W	0x00000000	DFI Training dfi_rdlvl_gate_delay Configuration 1
DDR_PCTL_DFITRRDLVLGATEDELAY2	0x0338	W	0x00000000	DFI Training dfi_rdlvl_gate_delay Configuration 2
DDR_PCTL_DFITRCMD	0x033c	W	0x00000000	DFI Training Command Register
DDR_PCTL_IPVR	0x03f8	W	0x00000000	IP Version Register
DDR_PCTL_IPTR	0x03fc	W	0x44574300	IP Type Register

Notes: **S**-Size: **B**- Byte (8 bits) access, **HW**- Half WORD (16 bits) access, **W**-WORD (32 bits) access

Name	Offset	Size	Reset Value	Description
DDRPHY_REG0	0x0000	W	0x0000000c	DDR PHY register 00
DDRPHY_REG1	0x0004	W	0x00000000	DDR PHY register 01
DDRPHY_REG2	0x0008	W	0x00000000	DDR PHY register 02
DDRPHY_REG3	0x000c	W	0x00000022	DDR PHY register 03
DDRPHY_REG4	0x0010	W	0x00000022	DDR PHY register 04
DDRPHY_REG5	0x0014	W	0x00000002	DDR PHY register 05
DDRPHY_REG6	0x0018	W	0x00000002	DDR PHY register 06
DDRPHY_REGB	0x002c	W	0x00000060	DDR PHY register 0B
DDRPHY_REGC	0x0030	W	0x00000000	DDR PHY register 0C
DDRPHY_REG11	0x0044	W	0x000000aa	DDR PHY register 11
DDRPHY_REG12	0x0048	W	0x00000002	DDR PHY register 12
DDRPHY_REG13	0x004c	W	0x0000000c	DDR PHY register 13
DDRPHY_REG14	0x0050	W	0x00000008	DDR PHY register 13
DDRPHY_REG16	0x0058	W	0x000000aa	DDR PHY register 16
DDRPHY_REG20	0x0080	W	0x000000aa	DDR PHY register 20
DDRPHY_REG21	0x0084	W	0x000000aa	DDR PHY register 21
DDRPHY_REG26	0x0098	W	0x0000000c	DDR PHY register 26
DDRPHY_REG27	0x009c	W	0x00000000	DDR PHY register 27
DDRPHY_REG28	0x00a0	W	0x00000001	DDR PHY register 28
DDRPHY_REG70	0x01c0	W	0x00000077	DDR PHY register 70
DDRPHY_REGB0	0x02c0	W	0x00000077	DDR PHY register B0
DDRPHY_REGB1	0x02c4	W	0x00000077	DDR PHY register B1
DDRPHY_REGB2	0x02c8	W	0x00000077	DDR PHY register B2
DDRPHY_REGB3	0x02cc	W	0x00000077	DDR PHY register B3
DDRPHY_REGB4	0x02d0	W	0x00000077	DDR PHY register B4
DDRPHY_REGB5	0x02d4	W	0x00000077	DDR PHY register B5
DDRPHY_REGB6	0x02d8	W	0x00000077	DDR PHY register B6
DDRPHY_REGB7	0x02dc	W	0x00000077	DDR PHY register B7
DDRPHY_REGB8	0x02e0	W	0x00000077	DDR PHY register B8
DDRPHY_REGB9	0x02e4	W	0x00000077	DDR PHY register B9
DDRPHY_REGBA	0x02e8	W	0x00000077	DDR PHY register BA
DDRPHY_REGBB	0x02ec	W	0x00000077	DDR PHY register BB
DDRPHY_REGBC	0x02f0	W	0x00000077	DDR PHY register BC
DDRPHY_REGBD	0x02f4	W	0x00000077	DDR PHY register BD
DDRPHY_REGBE	0x02f8	W	0x00000077	DDR PHY register BE
DDRPHY_REGEC	0x03b0	W	0x00000064	DDR PHY register EC
DDRPHY_REGED	0x03b4	W	0x00000012	DDR PHY register ED
DDRPHY_REGEE	0x03b8	W	0x00000002	DDR PHY register EE
DDRPHY_REGEF	0x03bc	W	0x00000000	DDR PHY register EF
DDRPHY_REGF0	0x03c0	W	0x00000000	DDR PHY register F0
DDRPHY_REGF1	0x03c4	W	0x00000000	DDR PHY register F1
DDRPHY_REGF2	0x03c8	W	0x00000000	DDR PHY register F2

Name	Offset	Size	Reset Value	Description
DDRPHY_REGFA	0x03e8	W	0x00000000	DDR PHY register FA
DDRPHY_REGFB	0x03ec	W	0x00000000	DDR PHY register FB
DDRPHY_REGFC	0x03f0	W	0x00000000	DDR PHY register FC
DDRPHY_REGFD	0x03f4	W	0x00000000	DDR PHY register FD
DDRPHY_REGFE	0x03f8	W	0x00000000	DDR PHY register FE
DDRPHY_REGFF	0x03fc	W	0x00000000	DDR PHY register FF

Notes: Size: **B**- Byte (8 bits) access, **H**W- Half WORD (16 bits) access, **W**-WORD (32 bits) access

## 8.4.2 Detail Register Description

### DDR\_PCTL\_SCFG

Address: Operational Base + offset (0x0000)

State Configuration Register

Bit	Attr	Reset Value	Description
31:12	RO	0x0	reserved
11:8	RW	0x3	<p>bbflags_timing</p> <p>The n_bbflags is a NIF output vector which provides combined information about the status of each memory bank. The de-assertion is based on when precharge, activates, reads/writes are scheduled by the TCU block. It may be possible to de-assert n_bbflags earlier than calculated by the TCU block. Programming bbflags_timing is used to achieve this. The maximum recommended value is: TRP.t_rp. The programmed value is the maximum number of "early" cycles that n_bbflags maybe de-asserted. The actual achieved de-assertion depends on the traffic profile.</p>
7	RO	0x0	reserved
6	RW	0x0	<p>nfifo_nif1_dis</p> <p>For internal use only for NFIFO testing.</p> <p>1'b0 = Only supported setting. 1'b1 = For internal use only.</p>
5:1	RO	0x0	reserved
0	RW	0x0	<p>hw_low_power_en</p> <p>Enables the hardware low-power interface. Allows the system to request via hardware (c_sysreq input) to enter the memories into Self-Refresh. The handshaking between the request and acknowledge hardware low power signals (c_sysreq and c_sysack, respectively) is always performed, but the ddr controller response depends on the value set on this register field and by the value driven on the c_active_in input pin.</p> <p>1'b0 = Disabled. Requests are always denied and ddr controller is unaffected by c_sysreq</p> <p>1'b1 = Enabled. Requests are accepted or denied, depending on the current operational state of ddr controller and on the value of c_active_in.</p>

**DDR\_PCTL\_SCTL**

Address: Operational Base + offset (0x0004)

Operational State Control Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	RO	0x0	reserved
2:0	RW	0x0	<p>state_cmd Issues an operational state transition request to the controller. 3'b000 = INIT (move to Init_mem from Config) 3'b001 = CFG (move to ConFigfrom Init_mem or Access) 3'b010 = GO (move to Access from Config) 3'b011 = SLEEP (move to Low_power from Access) 3'b100 = WAKEUP (move to Access from Low_power) Others = Reserved</p>

**DDR\_PCTL\_STAT**

Address: Operational Base + offset (0x0008)

Operational State Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6:4	RO	0x0	<p>lp_trig Reports the status of what triggered an entry to Low_power state. Is only set if in Low_power state. The individual bits report the following:            - lp_trig[2]: Software driven due to SCTL.state_cmd==SLEEP.            - lp_trig[1]: Hardware driven due to Hardware Low Power Interface.            - lp_trig[0]: Hardware driven due to Auto Self-Refresh (MCFG1.sr_idle&gt;0).            Note, if more than one trigger happens at the exact same time, more than one bit of lp_trig may be asserted high.</p>
3	RO	0x0	reserved
2:0	RO	0x0	<p>ctl_stat Returns the current operational state of the controller. 3'b000 = Init_mem 3'b001 = ConFig 3'b010 = Config_req 3'b011 = Access 3'b100 = Access_req 3'b101 = Low_power 3'b110 = Low_power_entry_req 3'b111 = Low_power_exit_req Others = Reserved</p>

**DDR\_PCTL\_INTRSTAT**

Address: Operational Base + offset (0x000c)

Interrupt Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:2	RO	0x0	reserved
1	RO	0x0	parity_intr Indicates that a DFI parity error has been detected 1'b0 = No error 1'b1 = Parity error
0	RO	0x0	ecc_intr Indicates that an ECC error has been detected 1'b0 = No error 1'b1 = Parity error

**DDR\_PCTL\_MCMD**

Address: Operational Base + offset (0x0040)

Memory Command Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	R/W SC	0x0	start_cmd Start command. When this bit is set to 1, the command operation defined in the cmd_opcode field is started. This bit is automatically cleared by the controller after the command is finished. The application can poll this bit to determine when controller is ready to accept another command. This bit cannot be cleared to 1'b0 by software.
30:28	RO	0x0	reserved
27:24	RW	0x0	cmd_add_del Set the additional delay associated with each command to $2^n$ internal timers clock cycles, where n is the bit field value. If n=0, the delay is 0. Max value is n=10.
23:20	RW	0x1	rank_sel Rank select for the command to be executed. 4'b0001 = Rank 0 4'b0010 = Rank 1 Others are reserved. Multiple 1'b1s in rank_sel mean multiple ranks are selected, which is useful broadcasting commands in parallel to multiple ranks during initialization and configuration of the memories. If MCMD.cmd_opcode=RSTL, all ranks should be selected as it cannot be performed to individual ranks

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
19:17	RW	0x0	<p>bank_addr Mode Register address driven on the memory bank address bits, BA1, BA0, during a Mode Register Set operation, defined by cmd_opcode=MRS. For other values of cmd_opcode, this field is ignored.</p> <p>3'b000 = MR0 3'b001 = MR1 3'b010 = MR2 3'b011 = MR3 Others = Reserved</p>
16:4	RW	0x0000	<p>cmd_addr Mode Register value driven on the memory address bits, A12 to A0, during a Mode Register Set operation defined by cmd_opcode=MRS. For other values of cmd_opcode this field is ignored. Refer to the memory specification for the correct settings of the various bits of this field during a MRS operation.</p>
3:0	RW	0x0	<p>cmd_opcode Command to be issued to the memory.</p> <p>4'b000 = Deselect. This is only used for timing purposes, no actual direct Deselect command is passed to the memories. 4'b0001 = Precharge All (PREA) 4'b0010 = Refresh (REF) 4'b0011 = Mode Register Set (MRS), MRS otherwise 4'b0100 = ZQ Calibration Short (ZQCS, only applies to DDR3) 4'b0101 = ZQ Calibration Long (ZQCL, only applies to DDR3) 4'b0110 = Software Driven Reset (RSTL, only applies to DDR3) 4'b0111 = Reserved 4'b1000 - Mode Register Read (MRR), is SRR in mDRR and is MPR in DDR3 Others - Reserved</p>

**DDR\_PCTL\_POWCTL**

Address: Operational Base + offset (0x0044)

Power Up Control Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:1	RO	0x0	reserved
0	R/W SC	0x0	<p>power_up_start Start the memory power up sequence. When this bit is set to 1'b1, controller starts the CKE and RESET# power up sequence to the memories. This bit is automatically cleared by controller after the sequence is completed. This bit cannot be cleared to 1'b0 by software.</p>

**DDR\_PCTL\_POWSTAT**

Address: Operational Base + offset (0x0048)

## Power Up Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:1	RO	0x0	reserved
0	RO	0x0	power_up_done Returns the status of the memory power-up sequence. 1'b0 = Power-up sequence has not been performed. 1'b1 = Power-up sequence has been performed.

**DDR\_PCTL\_CMDTSTAT**

Address: Operational Base + offset (0x004c)

## Command Timers Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:1	RO	0x0	reserved
0	RO	0x0	cmd_tstat Returns the status of the timers for memory commands. This ANDs all the command timers together. 1'b0 = One or more command timers has not expired. 1'b1 = All command timers have expired.

**DDR\_PCTL\_CMDTSTATEN**

Address: Operational Base + offset (0x0050)

## Command Timers Status Enable Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:1	RO	0x0	reserved
0	RW	0x0	cmd_tstat_en Enables the generation of the status of the timers for memory commands. Is enabled before CMDTSTAT register is read. 1'b0 - Disabled 1'b1 - Enabled

**DDR\_PCTL\_MRRCFG0**

Address: Operational Base + offset (0x0060)

## Mode Register Read Configuration 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x0	mrr_byte_sel Selects which byte's data to store when performing an MRR command via MCMD. LegalValues: 0 .. 8

**DDR\_PCTL\_MRRSTAT0**

Address: Operational Base + offset (0x0064)

## Mode Register Read Status 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RO	0x00	mrrstat_beat3 MRR/MPR read data beat 3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
23:16	RO	0x00	mrrstat_beat2 MRR/MPR read data beat 2
15:8	RO	0x00	mrrstat_beat1 MRR/MPR read data beat 1
7:0	RO	0x00	mrrstat_beat0 MRR/MPR read data beat 0

**DDR\_PCTL\_MRRSTAT1**

Address: Operational Base + offset (0x0068)

Mode Register Read Status 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RO	0x00	mrrstat_beat7 MRR/MPR read data beat 7
23:16	RO	0x00	mrrstat_beat6 MRR/MPR read data beat 6
15:8	RO	0x00	mrrstat_beat5 MRR/MPR read data beat 5
7:0	RO	0x00	mrrstat_beat4 MRR/MPR read data beat 4

**DDR\_PCTL\_MCFG1**

Address: Operational Base + offset (0x007c)

Memory Configuration 1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	RW	0x0	hw_exit_idle_en When this bit is programmed to 1'b1 the c_active_in pin can be used to exit from the automatic clock stop, power down or self-refresh modes.
30:24	RO	0x0	reserved
23:16	RW	0x00	hw_idle Hardware idle period. The c_active output is driven high if the NIF is idle in Access state for hw_idle * 32 * n_clk cycles. The hardware idle function is disabled when hw_idle=0.
15:8	RO	0x0	reserved
7:0	RW	0x00	sr_idle Self-Refresh idle period. Memories are placed into Self-Refresh mode if the NIF is idle in Access state for sr_idle * 32 * n_clk cycles. The automatic self-refresh function is disabled when sr_idle=0.

**DDR\_PCTL\_MCFG**

Address: Operational Base + offset (0x0080)

Memory Configuration Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:20	RW	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
19:18	RW	0x1	<p>tfaw_cfg Field0000 Abstract Sets tFAW to be 4, 5 or 6 times tRRD. 2'b00 = set tFAW=4*tRRD 2'b01 = set tFAW=5*tRRD 2'b10 = set tFAW=6*tRRD</p>
17	RW	0x0	<p>pd_exit_mode Selects the mode for Power Down Exit. For DDR3, the power down exit mode setting in controller must be consistent with the value programmed into the power down exit mode bit of MR0. 1'b0 = slow exit 1'b1 = fast exit</p>
16	RW	0x0	<p>pd_type Sets the Power down type. 1'b0 = Precharge Power Down 1'b1 = Active Power Down</p>
15:8	RW	0x00	<p>pd_idle Power-down idle period in n_clk cycles. Memories are placed into power-down mode if the NIF is idle for pd_idlen_clk cycles. The automatic power down function is disabled when pd_idle=0.</p>
7:5	RO	0x0	reserved
4	RW	0x0	<p>stagger_cs For multi-rank commands from the DCU, stagger the assertion of CS_N to odd and even ranks by one n_clk cycle. This is useful when using RDIMMs, when multi-rank commands may be interpreted as writes to control words in the register chip. 1'b0 = Do not stagger CS_N 1'b1 = Stagger CS_N</p>
3	RW	0x0	<p>two_t_en Enables 2T timing for memory commands. 1'b0= Disabled 1'b1 = Enabled</p>
2:1	RO	0x0	reserved
0	RW	0x0	<p>mem_bl DDR Burst Length. The BL setting in DDR3 must be consistent with the value programmed into the BL field of MR0. 1'b0 = BL4, Burst length of 4 1'b1 = BL8, Burst length of 8 (MR0.BL=2'b00 for DDR3)</p>

**DDR\_PCTL\_PPCFG**

Address: Operational Base + offset (0x0084)

Partially Populated Memories Configuration Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:9	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
8:1	RW	0x00	<p>rpmem_dis Reduced Population Disable bits. Setting these bits disables the corresponding NIF/DDR data lanes from writing or reading data. Lane 0 is always present, hence only 8 bits are required for the remaining lanes including the ECC lane.</p> <p>In 1:2 mode bit 0 of rpmem_dis covers n_wdata/n_rdata/m_ctl_d/m_phy_q[63:32], bit 1 [95:64] etc.</p> <p>In 1:1 mode bit 0 of rpmem_dis covers n_wdata/n_rdata/m_ctl_d/m_phy_q[31:16], bit 2 [47:32] etc.</p> <p>There are no restrictions on which byte lanes can be disabled, other than byte lane 0 is required. Gaps between enabled byte lanes are allowed. For each bit:</p> <p>1'b0 = lane exists 1'b1 = lane is disabled</p>
0	RW	0x0	<p>ppmem_en Partially Population Enable bit. Setting this bit enables the partial population of external memories where the entire application bus is routed to a reduced size memory system. The lower half of the SDRAM data bus, bit 0 up to bit 15, is the active portion when Partially Populated memories are enabled.</p> <p>1'b0 = Disabled 1'b1 = Enabled</p>

**DDR\_PCTL\_MSTAT**

Address: Operational Base + offset (0x0088)

Memory Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	RO	0x0	reserved
2	RO	0x0	<p>self_refresh Indicates if controller, through auto self-refresh, has placed the memories in Self-Refresh.</p> <p>1'b0 = Memory is not in Self-Refresh 1'b1 = Memory is in Self-Refresh</p>
1	RO	0x0	<p>clock_stop Indicates if controller has placed the memories in Clock Stop.</p> <p>1'b0 = Memory is not in Clock Stop 1'b1 = Memory is in Clock Stop</p>
0	RO	0x0	<p>power_down Indicates if controller has placed the memories in Power Down.</p> <p>1'b0 = Memory is not in Power Down 1'b1 = Memory is in Power-Down</p>

**DDR\_PCTL\_DTUPDES**

Address: Operational Base + offset (0x0094)

DTU Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:14	RO	0x0	reserved
13	RO	0x0	dtu_rd_missing Indicates if one or more read beats of data did not return from memory.
12:9	RO	0x0	dtu_eaffl Indicates the number of entries in the FIFO that is holding the log of error addresses for data comparison
8	RO	0x0	dtu_random_error Indicates that the random data generated had some failures when written and read to the memories
7	RO	0x0	dtu_err_b7 Detected at least 1 bit error for bit 7 in the programmable data buffers
6	RO	0x0	dtu_err_b6 Detected at least 1 bit error for bit 6 in the programmable data buffers
5	RO	0x0	dtu_err_b5 Detected at least 1 bit error for bit 5 in the programmable data buffers
4	RO	0x0	dtu_err_b4 Detected at least 1 bit error for bit 4 in the programmable data buffers
3	RO	0x0	dtu_err_b3 Detected at least 1 bit error for bit 3 in the programmable data buffers
2	RO	0x0	dtu_err_b2 Detected at least 1 bit error for bit 2 in the programmable data buffers
1	RO	0x0	dtu_err_b1 Detected at least 1 bit error for bit 1 in the programmable data buffers
0	RO	0x0	dtu_err_b0 Detected at least 1 bit error for bit 0 in the programmable data buffers

**DDR\_PCTL\_DTUNA**

Address: Operational Base + offset (0x0098)

DTU Number of Addresses Created Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	dtu_num_address Indicates the number of addresses that were created on the NIF interface during random data generation.

**DDR\_PCTL\_DTUNE**

Address: Operational Base + offset (0x009c)

## DTU Number of Errors Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	dtu_num_errors Indicates the number of errors that were detected on the readback of the NIF data during random data generation.

**DDR\_PCTL\_DTUPRD0**

Address: Operational Base + offset (0x00a0)

## DTU Parallel Read 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0000	dtu_allbits_1 Allows all the bit ones from each of the 16 received read bytes to be read in parallel. Used as part of read data eye training where a transition is required to be monitored to train the eye.
15:0	RO	0x0000	dtu_allbits_0 Allows all the bit zeros from each of the 16 received read bytes to be read in parallel. Used as part of read data eye training where a transition is required to be monitored to train the eye.

**DDR\_PCTL\_DTUPRD1**

Address: Operational Base + offset (0x00a4)

## DTU Parallel Read 1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0000	dtu_allbits_3 Allows all the bit threes from each of the 16 received read bytes to be read in parallel. Used as part of read data eye training where a transition is required to be monitored to train the eye.
15:0	RO	0x0000	dtu_allbits_2 Allows all the bit twos from each of the 16 received read bytes to be read in parallel. Used as part of read data eye training where a transition is required to be monitored to train the eye.

**DDR\_PCTL\_DTUPRD2**

Address: Operational Base + offset (0x00a8)

## DTU Parallel Read 2 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0000	dtu_allbits_5 Allows all the bit fives from each of the 16 received read bytes to be read in parallel. Used as part of read data eye training where a transition is required to be monitored to train the eye.
15:0	RO	0x0000	dtu_allbits_4 Allows all the bit fours from each of the 16 received read bytes to be read in parallel. Used as part of read data eye training where a transition is required to be monitored to train the eye.

**DDR\_PCTL\_DTUPRD3**

Address: Operational Base + offset (0x00ac)

DTU Parallel Read 3 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0000	dtu_allbits_7 Allows all the bit sevens from each of the 16 received read bytes to be read in parallel. Used as part of read data eye training where a transition is required to be monitored to train the eye.
15:0	RO	0x0000	dtu_allbits_6 Allows all the bit sixes from each of the 16 received read bytes to be read in parallel. Used as part of read data eye training where a transition is required to be monitored to train the eye.

**DDR\_PCTL\_DTUAWDT**

Address: Operational Base + offset (0x00b0)

DTU Address Width Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:11	RO	0x0	reserved
10:9	RW	0x1	number_ranks Number of supported memory ranks. 2'b00 = 1 rank 2'b01 = 2 ranks
8	RO	0x0	reserved
7:6	RW	0x2	row_addr_width Width of the memory row address bits. 2'b00 = 13 bits wide 2'b01 = 14 bits wide 2'b10 = 15 bits wide 2'b11 = 16 bits wide
5	RO	0x0	reserved
4:3	RW	0x2	bank_addr_width Field0000 Abstract Width of the memory bank address bits. 2'b00 = 2 bits wide (4 banks) 2'b01 = 3 bits wide (8 banks) Others = Reserved
2	RO	0x0	reserved
1:0	RW	0x0	column_addr_width Field0000 Abstract Width of the memory column address bits. 2'b00 = 7 bits wide 2'b01 = 8 bits wide 2'b10 = 9 bits wide 2'b11 = 10 bits wide

**DDR\_PCTL\_TOGCNT1U**

Address: Operational Base + offset (0x00c0)

## Toggle Counter 1us Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:10	RO	0x0	reserved
9:0	RW	0x064	toggle_counter_1u The number of internal timers clock cycles

**DDR\_PCTL\_TINIT**

Address: Operational Base + offset (0x00c4)

t\_init Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:9	RO	0x0	reserved
8:0	RW	0x0c8	t_init Defines the time period (in us) to hold dfi_cke and dfi_reset_n stable during the memory power up sequence. The value programmed must correspond to at least 200us. The actual time period defined is TINIT * TOGCNT1U * internal timers clock .period

**DDR\_PCTL\_TRSTH**

Address: Operational Base + offset (0x00c8)

t\_rsth Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:10	RO	0x0	reserved
9:0	RW	0x000	t_rsth Defines the time period (in us) to hold the dfi_reset_n signal high after it is de-asserted during the DDR3 Power Up/Reset sequence. The value programmed for DDR3 must correspond to minimum 500us of delay. The actual time period defined is TRSTH * TOGCNT1U * internal timers clock period.

**DDR\_PCTL\_TOGCNT100N**

Address: Operational Base + offset (0x00cc)

Toggle Counter 100ns

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6:0	RW	0x01	toggle_counter_100n The number of internal timers clock cycles.

**DDR\_PCTL\_TREFI**

Address: Operational Base + offset (0x00d0)

t\_refi Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RW	0x01	t_refi Defines the time period (in 100ns units) of the Refresh interval. The actual time period defined is TREFI * TOGCNT100N * internal timers clock period.

**DDR\_PCTL\_TMRD**

Address: Operational Base + offset (0x00d4)

t\_mrd Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	RO	0x0	reserved
2:0	RW	0x1	t_mrd Mode Register Set command cycle time in memory clock cycles. DDR3: Time from MRS to MRS command. DDR3 Legal Values: 2..4

**DDR\_PCTL\_TRFC**

Address: Operational Base + offset (0x00d8)

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:9	RO	0x0	reserved
8:0	RW	0x001	t_rfc Refresh to Active/Refresh command time in memory clock cycles. DDR3 Legal Values: 36.. 374

**DDR\_PCTL\_TRP**

Address: Operational Base + offset (0x00dc)

t\_trp Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:18	RO	0x0	reserved
17:16	RW	0x1	prea_extra Additional cycles required for a Precharge All (PREA) command - in addition to t_rp. In terms of memory clock cycles DDR3 Value: 0
15:4	RO	0x0	reserved
3:0	RW	0x6	t_rp Precharge period in memory clock cycles. DDR3 Legal Values: 5..14

**DDR\_PCTL\_TRTW**

Address: Operational Base + offset (0x00e0)

t\_rtw Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x2	t_rtw Read to Write turnaround time in memory clock cycles. DDR3 Legal Values: 2..10

**DDR\_PCTL\_TAL**

Address: Operational Base + offset (0x00e4)

AL Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x0	t_al Additive Latency in memory clock cycles. For DDR3 this must be 0, CL-1, CL-2 depending weather the AL value in MR1 is 0,1, or 2 respectively. CL is the CAS latency programmed into MR0. DDR3 Legal Values: 0, CL-1, CL-2 (depending on AL=0,1,2 in MR1)

**DDR\_PCTL\_TCL**

Address: Operational Base + offset (0x00e8)

CL Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x4	t_cl CAS Latency in memory clock cycles. The controllersetting must match the value programmed into the CL field of MR0. DDR3 Legal Value: CL

**DDR\_PCTL\_TCWL**

Address: Operational Base + offset (0x00ec)

CWL Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x3	t_cwl CAS Write Latency in memory clock cycles. For DDR3, the setting must match the value programmed in the memory CWL field of MR2. DDR3 Legal Value: CWL

**DDR\_PCTL\_TRAS**

Address: Operational Base + offset (0x00f0)

t\_ras Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RO	0x0	reserved
5:0	RW	0x10	t_ras Activate to Precharge command time in memory clock cycles. DDR3 Legal Values: 15..38

**DDR\_PCTL\_TRC**

Address: Operational Base + offset (0x00f4)

t\_rc Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
5:0	RW	0x16	t_rc Row Cycle time in memory clock cycles. Specifies the minimum Activate to Activate distance for accesses to same bank. DDR3 Legal Values: 20..52

**DDR\_PCTL\_TRCD**

Address: Operational Base + offset (0x00f8)

t\_rcd Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x6	t_rcd Row to Column delay in memory clock cycles. Specifies the minimum Activate to Column distance. DDR3 Legal Values: 5..14

**DDR\_PCTL\_TRRD**

Address: Operational Base + offset (0x00fc)

t\_rrd Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x4	t_rrd Row-to-Row delay in memory clock cycles. Specifies the minimum Activate-to-Activate distance for consecutive accesses to different banks in the same rank. DDR3 Legal Values: 4..8

**DDR\_PCTL\_TRTP**

Address: Operational Base + offset (0x0100)

t\_rtp Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x3	t_rtp Read to Precharge time in memory clock cycles. Specifies the minimum distance Read to Precharge for consecutive accesses to same bank. DDR3 Legal Values: 3..8

**DDR\_PCTL\_TWR**

Address: Operational Base + offset (0x0104)

t\_wr Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:5	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
4:0	RW	0x06	t_wr Write recovery time in memory clock cycles. When using close page the controller setting must be consistent with the WR field setting of MRO. DDR3 Legal Values: 6..16

**DDR\_PCTL\_TWTR**

Address: Operational Base + offset (0x0108)

t\_wtr Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x4	t_wtr Write to Read turnaround time, in memory clock cycles. DDR3 Legal Values: 3..8

**DDR\_PCTL\_TEXSR**

Address: Operational Base + offset (0x010c)

t\_exsr Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:10	RO	0x0	reserved
9:0	RW	0x001	t_exsr Exit Self-Refresh to first valid command delay, in memory clock cycles. For DDR3, this should be programmed to match tXSDLL (SRE to a command requiring DLL locked) as defined by the memory device specification. DDR3 Typical Value: 512

**DDR\_PCTL\_TXP**

Address: Operational Base + offset (0x0110)

t\_xp Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	RO	0x0	reserved
2:0	RW	0x1	t_xp Exit Power Down to first valid command delay when DLL is on (fast exit), measured in memory clock cycles. Legal Values: 1..7

**DDR\_PCTL\_TXPDLL**

Address: Operational Base + offset (0x0114)

t\_xpdll Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
5:0	RW	0x00	t_xpdll Exit Power Down to first valid command delay when DLL is off (slow exit), measured in memory clock cycles. DDR3 Legal Values: 3..63

**DDR\_PCTL\_TZQCS**

Address: Operational Base + offset (0x0118)

t\_zqcs Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6:0	RW	0x00	t_zqcs SDRAM ZQ Calibration Short period, in memory clock cycles. Should be programmed to match the tZQCS timing value as defined in the memory specification. DDR3 Typical Value: 64

**DDR\_PCTL\_TZQCSI**

Address: Operational Base + offset (0x011c)

t\_zqcsi Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	t_zqcsi SDRAM ZQCS interval, measured in Refresh interval units. The total time period defined is TZQCSI*TREFI * TOGCNT100N * internal timers clock period. Programming a value of 0 in t_zqcsi disables the auto-ZQCS functionality in controller. DDR3 Legal Values: 0..4294967295

**DDR\_PCTL\_TDQS**

Address: Operational Base + offset (0x0120)

t\_dqs Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	RO	0x0	reserved
2:0	RW	0x1	t_dqs Additional data turnaround time in memory clock cycles for accesses to different ranks. Used to increase the distance between column commands to different ranks, allowing more tolerance as the driver source changes on the bidirectional DQS and/or DQ signals. DDR3 Legal Values: 1..7

**DDR\_PCTL\_TCKSRE**

Address: Operational Base + offset (0x0124)

t\_cksre Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:5	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
4:0	RW	0x00	t_cksr In DDR3, this is the time after Self-Refresh Entry that CKE is held high before going low. In memory clock cycles. Specifies the clock disable delay after SRE. This should be programmed to match the greatest value between 10ns and 5 memory clock periods. DDR3 Legal Values: 5..15

**DDR\_PCTL\_TCKSRX**

Address: Operational Base + offset (0x0128)

t\_cksr Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:5	RO	0x0	reserved
4:0	RW	0x00	t_cksr In DDR3, this is the time (before Self-Refresh Exit) that CKE is maintained high before issuing SRX. In memory clock cycles. Specifies the clock stable time before SRX. This should be programmed to match the greatest value between 10ns and 5 memory clock periods. DDR3 Legal Values: 5..15

**DDR\_PCTL\_TCKE**

Address: Operational Base + offset (0x012c)

t\_cke Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	RO	0x0	reserved
2:0	RW	0x3	t_cke CKE minimum pulse width in memory clock cycles. DDR3 Legal Values: 3..6

**DDR\_PCTL\_TMOD**

Address: Operational Base + offset (0x0130)

t\_mod Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:5	RO	0x0	reserved
4:0	RW	0x00	t_mod In DDR3 mode, this is the time from MRS to any valid non-MRS command (except DESELECT or NOP) in memory clock cycles. DDR3 Legal Values: 0..31

**DDR\_PCTL\_TRSTL**

Address: Operational Base + offset (0x0134)

Reset Low Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
6:0	RW	0x00	t_rstl Memory Reset Low time, in memory clock cycles. Defines the time period to hold dfi_reset_n signal low during a software driven DDR3 Reset Operation. The value programmed must correspond to at least 100ns of delay. DDR3 Legal Values: 1..127

**DDR\_PCTL\_TZQCL**

Address: Operational Base + offset (0x0138)

t\_zqcl Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:10	RO	0x0	reserved
9:0	RW	0x000	t_zqcl SDRAM ZQ Calibration Long period in memory clock cycles. If DDR3, should be programmed to match the memory tZQinit timing value for the first ZQCL command during memory initialization; should be programmed to match tZQoper timing value after reset and initialization. DDR3 Legal Values: 0..1023

**DDR\_PCTL\_TMRR**

Address: Operational Base + offset (0x013c)

t\_mrr Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RW	0x02	t_mrr Time for a Mode Register Read (MRR command from MCMD).

**DDR\_PCTL\_TCKESR**

Address: Operational Base + offset (0x0140)

t\_ckesr Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x4	t_ckesr Minimum CKE low width for Self-Refresh entry to exit timing in memory clock cycles. Recommended settings: For DDR3 : t_ckesr = t_cke + 1 DDR3 Legal Values: 4..7

**DDR\_PCTL\_TDPPD**

Address: Operational Base + offset (0x0144)

t\_dpd Timing Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:10	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
9:0	RW	0x000	t_dpd Minimum Deep Power Down time. Is in terms of us. When a MCMD.DPDE command occurs, TDPD time is waited before MCMD.start_cmd can be cleared. MCMD_cmd_add_del (if any) does not start until TDPD has completed. This ensures TDPD requirement for the memory is not violated. The actual time period defined is TDPD* TOGCNT1U * internal timers clock period. DDR3 Legal Values: 0

**DDR\_PCTL\_DTUWACTL**

Address: Operational Base + offset (0x0200)

DTU Write Address Control

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:30	RW	0x0	dtu_wr_rank Write rank to where data is to be targeted
29	RO	0x0	reserved
28:13	RW	0x0000	dtu_wr_row Write row to where data is to be targeted
12:10	RW	0x0	dtu_wr_bank Write bank to where data is to be targeted
9:0	RW	0x000	dtu_wr_col FWrite column to where data is to be targeted

**DDR\_PCTL\_DTURACTL**

Address: Operational Base + offset (0x0204)

DTU Read Address Control Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:30	RW	0x0	dtu_rd_rank Read rank from where data comes
29	RO	0x0	reserved
28:13	RW	0x0000	dtu_rd_row Read row from where data comes
12:10	RW	0x0	dtu_rd_bank Read bank from where data comes
9:0	RW	0x000	dtu_rd_col Read column from where data comes

**DDR\_PCTL\_DTUCFG**

Address: Operational Base + offset (0x0208)

DTU Configuration Control Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:16	RW	0x00	dtu_row_increments Number of times to increment the row address when generating random data, up to a maximum of 127 times.

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15	RW	0x0	dtu_wr_multi_rd When set puts the DTU into write once multiple reads mode.
14	RW	0x0	dtu_data_mask_en Controls whether random generated data masks are transmitted. Unless enabled all data bytes are written to memory and expected to be read from memory.
13:10	RW	0x0	dtu_target_lane Selects one of the byte lanes for data comparison into the programmable read data buffer.
9	RW	0x0	dtu_generate_random Generate transfers using random data, otherwise generate transfers from the programmable write data buffers.
8	RW	0x0	dtu_incr_banks When the column address rolls over increment the bank address until we reach and conclude bank 7.
7	RW	0x0	dtu_incr_cols Increment the column address until we saturate. Return to zero if DTUCFG.dtu_incr_banks is set to 1 and we are not at bank 7.
6:1	RW	0x00	dtu_nalen Length of the NIF transfer sequence that is passed through the controller for each created address.
0	RW	0x0	dtu_enable When set, allows the DTU module to take ownership of the NIF interface: 1: DTU enabled 0: DTU disabled

**DDR\_PCTL\_DTUECTL**

Address: Operational Base + offset (0x020c)

DTU Execute Control Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	RO	0x0	reserved
2	R/W SC	0x0	wr_multi_rd_RST When set, resets the DTU in write once multiple reads mode, to allow a new write to be performed. This bit automatically clears.
1	R/W SC	0x0	run_error_reports When set, initiates the calculation of the error status bits. This bit automatically clears when the re-calculation is done. This is only used in debug mode to verify the comparison logic.
0	R/W SC	0x0	run_dtu When set, initiates the running of the DTU read and write transfer. This bit automatically clears when the transfers are completed

**DDR\_PCTL\_DTUWDO**

Address: Operational Base + offset (0x0210)

## DTU Write Data #0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	dtu_wr_byte3 Write data byte
23:16	RW	0x00	dtu_wr_byte2 Write data byte
15:8	RW	0x00	dtu_wr_byte1 Write data byte
7:0	RW	0x00	dtu_wr_byte0 Write data byte

**DDR\_PCTL\_DTUWD1**

Address: Operational Base + offset (0x0214)

## DTU Write Data #1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	dtu_wr_byte7 Write data byte
23:16	RW	0x00	dtu_wr_byte6 Write data byte
15:8	RW	0x00	dtu_wr_byte5 Write data byte
7:0	RW	0x00	dtu_wr_byte4 Write data byte

**DDR\_PCTL\_DTUWD2**

Address: Operational Base + offset (0x0218)

## DTU Write Data #2 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	dtu_wr_byte11 Write data byte
23:16	RW	0x00	dtu_wr_byte10 Write data byte
15:8	RW	0x00	dtu_wr_byte9 Write data byte
7:0	RW	0x00	dtu_wr_byte8 Write data byte

**DDR\_PCTL\_DTUWD3**

Address: Operational Base + offset (0x021c)

## DTU Write Data #3 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	dtu_wr_byte15 Write data byte
23:16	RW	0x00	dtu_wr_byte14 Write data byte

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:8	RW	0x00	dtu_wr_byte13 Write data byte
7:0	RW	0x00	dtu_wr_byte12 Write data byte

**DDR\_PCTL\_DTUWDM**

Address: Operational Base + offset (0x0220)

DTU Write Data Mask Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	dm_wr_byte0 Write data mask bit, one bit for each byte. Each bit should be 0 for a byte lane that contains valid write data.

**DDR\_PCTL\_DTURDO**

Address: Operational Base + offset (0x0224)

DTU Read Data #0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RO	0x00	dtu_rd_byte3 Read byte
23:16	RO	0x00	dtu_rd_byte2 Read byte
15:8	RO	0x00	dtu_rd_byte1 Read byte
7:0	RO	0x00	dtu_rd_byte0 Read byte

**DDR\_PCTL\_DTURD1**

Address: Operational Base + offset (0x0228)

DTU Read Data #1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RO	0x00	dtu_rd_byte7 Read byte
23:16	RO	0x00	dtu_rd_byte6 Read byte
15:8	RO	0x00	dtu_rd_byte5 Read byte
7:0	RO	0x00	dtu_rd_byte4 Read byte

**DDR\_PCTL\_DTURD2**

Address: Operational Base + offset (0x022c)

DTU Read Data #2 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RO	0x00	dtu_rd_byte11 Read byte
23:16	RO	0x00	dtu_rd_byte10 Read byte
15:8	RO	0x00	dtu_rd_byte9 Read byte
7:0	RO	0x00	dtu_rd_byte8 Read byte

**DDR\_PCTL\_DTURD3**

Address: Operational Base + offset (0x0230)

DTU Read Data #3 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RO	0x00	dtu_rd_byte15 Read byte
23:16	RO	0x00	dtu_rd_byte14 Read byte
15:8	RO	0x00	dtu_rd_byte13 Read byte
7:0	RO	0x00	dtu_rd_byte12 Read byte

**DDR\_PCTL\_DTULFSRWD**

Address: Operational Base + offset (0x0234)

DTU LFSR Seed for Write Data Generation Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	dtu_lfsr_wseed This is the initial seed for the random write data generation LFSR (linear feedback shift register), shared with the write mask generation.

**DDR\_PCTL\_DTULFSRRD**

Address: Operational Base + offset (0x0238)

DTU LFSR Seed for Read Data Generation Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	dtu_lfsr_rseed This is the initial seed for the random read data generation LFSR (linear feedback shift register), this is shared with the read mask generation. The read data mask is reconstructed the same as the write data mask was created, allowing the "on the fly comparison" ignore bytes which were not written.

**DDR\_PCTL\_DTUEAF**

Address: Operational Base + offset (0x023c)

DTU Error Address FIFO Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:30	RO	0x0	ea_rank Indicates the rank that the error occurred in during random data generation. There could be a number of entries in this FIFO. If FIFO is empty one reads zeroes.
29	RO	0x0	reserved
28:13	RO	0x0000	ea_row Indicates the row that the error occurred in during random data generation. There could be a number of entries in this FIFO. If FIFO is empty one reads zeroes.
12:10	RO	0x0	ea_bank Indicates the bank that the error occurred in during random data generation. There could be a number of entries in this FIFO. If FIFO is empty one reads zeroes
9:0	RO	0x000	ea_column Indicates the column address that the error occurred in during random data generation. There could be a number of entries in this FIFO. If FIFO is empty one reads zeroes.

**DDR\_PCTL\_DFITCTRLDELAY**

Address: Operational Base + offset (0x0240)

DFI tctrl\_delay Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x2	tctrl_delay Specifies the number of DFI clock cycles after an assertion or deassertion of the DFI control signals that the control signals at the PHY-DRAM interface reflect the assertion or de-assertion. If the DFI clock and the memory clock are not phase-aligned, this timing parameter should be rounded up to the next integer value.

**DDR\_PCTL\_DFIODTCFG**

Address: Operational Base + offset (0x0244)

DFI ODT Configuration

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12	RW	0x0	rank1_odt_default Default ODT value of rank 1 when there is no read/write activity
11	RW	0x0	rank1_odt_write_sel Enable/disable ODT for rank 1 when a write access is occurring on this rank
10	RW	0x0	rank1_odt_write_nse Enable/disable ODT for rank 1 when a write access is occurring on a different rank

Bit	Attr	Reset Value	Description
9	RW	0x0	rank1_odt_read_sel Enable/disable ODT for rank 1 when a read access is occurring on this rank
8	RW	0x0	rank1_odt_read_nsel Enable/disable ODT for rank 1 when a read access is occurring on a different rank
7:5	RO	0x0	reserved
4	RW	0x0	rank0_odt_default Default ODT value of rank 0 when there is no read/write activity
3	RW	0x0	rank0_odt_write_sel Enable/disable ODT for rank 0 when a write access is occurring on this rank
2	RW	0x0	rank0_odt_write_nsel Enable/disable ODT for rank 0 when a write access is occurring on a different rank
1	RW	0x0	rank0_odt_read_sel Enable/disable ODT for rank 0 when a read access is occurring on this rank
0	RW	0x0	rank0_odt_read_nsel Enable/disable ODT for rank 0 when a read access is occurring on a different rank

**DDR\_PCTL\_DFIODTCFG1**

Address: Operational Base + offset (0x0248)

DFI ODT Timing Configuration 1 (for Latency and Length)

Bit	Attr	Reset Value	Description
31:27	RO	0x0	reserved
26:24	RW	0x6	odt_len_bl8_r ODT length for BL8 read transfers Length of dfi_odt signal for BL8 reads. This is in terms of SDR cycles. For BL4 reads, the length of dfi_odt is always 2 cycles shorter than the value in this register field.
23:19	RO	0x0	reserved
18:16	RW	0x6	odt_len_bl8_w ODT length for BL8 write transfers Length of dfi_odt signal for BL8 writes. This is in terms of SDR cycles. For BL4 writes, the length of dfi_odt is always 2 cycles shorter than the value in this register field.
15:13	RO	0x0	reserved
12:8	RW	0x00	odt_lat_r Field0000 Abstract ODT latency for reads Latency after a read command that dfi_odt is set. This is in terms of SDR cycles.
7:5	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
4:0	RW	0x00	odt_lat_w ODT latency for writes Latency after a write command that dfi_odt is set. This is in terms of SDR cycles

**DDR\_PCTL\_DFIODTRANKMAP**

Address: Operational Base + offset (0x024c)

DFI ODT Rank Mapping

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RO	0x0	reserved
5:4	RW	0x2	odt_rank_map1 Rank mapping for dfi_odt[1] Determines which rank access(es) will cause dfi_odt[1] to be asserted Bit 5= 1: dfi_odt[1] will be asserted to terminate rank 1 Bit 4= 1: dfi_odt[1] will be asserted to terminate rank 0
3:2	RO	0x4	reserved
1:0	RW	0x1	odt_rank_map0 Rank mapping for dfi_odt[0] Determines which rank access(es) will cause dfi_odt[0] to be asserted Bit 1= 1: dfi_odt[0] will be asserted to terminate rank 1 Bit 0= 1: dfi_odt[0] will be asserted to terminate rank 0

**DDR\_PCTL\_DFITPHYWRDATA**

Address: Operational Base + offset (0x0250)

DFI tphy\_wrdata Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RO	0x0	reserved
5:0	RW	0x01	tphy_wrdata Specifies the number of DFI clock cycles between when the dfi_wrdata_en signal is asserted to when the associated write data is driven on the dfi_wrdata signal. This has no impact on performance, only adjusts the relative time between enable and data transfer.

**DDR\_PCTL\_DFITPHYWRLAT**

Address: Operational Base + offset (0x0254)

DFI tphy\_wrlat Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RO	0x0	reserved
5:0	RW	0x01	tphy_wrlat Specifies the number of DFI clock cycles between when a write command is sent on the DFI control interface and when the dfi_wrdata_en signal is asserted.

**DDR\_PCTL\_DFITRDDATAEN**

Address: Operational Base + offset (0x0260)

## DFI trddata\_en Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RO	0x0	reserved
5:0	RW	0x01	trddata_en Specifies the number of DFI clock cycles from the assertion of a read command on the DFI to the assertion of the dfi_rddata_en signal.

**DDR\_PCTL\_DFITPHYRDLAT**

Address: Operational Base + offset (0x0264)

## DFI tphy\_rdlat Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RO	0x0	reserved
5:0	RW	0x0f	tphy_rdlat Specifies the maximum number of DFI clock cycles allowed from the assertion of the dfi_rddata_en signal to the assertion of the dfi_rddata_valid signal.

**DDR\_PCTL\_DFITPHYUPDTYPE0**

Address: Operational Base + offset (0x0270)

## DFI tphyupd\_type0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:12	RO	0x0	reserved
11:0	RW	0x010	tphyupd_type0 Specifies the maximum number of DFI clock cycles that the dfi_phyupd_req signal may remain asserted after the assertion of the dfi_phyupd_ack signal for dfi_phyupd_type = 0x0. The dfi_phyupd_req signal may de-assert at any cycle after the assertion of the dfi_phyupd_ack signal.

**DDR\_PCTL\_DFITPHYUPDTYPE1**

Address: Operational Base + offset (0x0274)

## DFI tphyupd\_type1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:12	RO	0x0	reserved
11:0	RW	0x010	tphyupd_type1 Specifies the maximum number of DFI clock cycles that the dfi_phyupd_req signal may remain asserted after the assertion of the dfi_phyupd_ack signal for dfi_phyupd_type = 0x1. The dfi_phyupd_req signal may de-assert at any cycle after the assertion of the dfi_phyupd_ack signal.

**DDR\_PCTL\_DFITPHYUPDTYPE2**

Address: Operational Base + offset (0x0278)

## DFI tphyupd\_type2 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:12	RO	0x0	reserved
11:0	RW	0x010	tphyupd_type2 Specifies the maximum number of DFI clock cycles that the dfi_phyupd_req signal may remain asserted after the assertion of the dfi_phyupd_ack signal for dfi_phyupd_type = 0x2. The dfi_phyupd_req signal may de-assert at any cycle after the assertion of the dfi_phyupd_ack signal.

**DDR\_PCTL\_DFITPHYUPDTYPE3**

Address: Operational Base + offset (0x027c)

DFI tphyupd\_type3 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:12	RO	0x0	reserved
11:0	RW	0x010	tphyupd_type3 Specifies the maximum number of DFI clock cycles that the dfi_phyupd_req signal may remain asserted after the assertion of the dfi_phyupd_ack signal for dfi_phyupd_type = 0x3. The dfi_phyupd_req signal may de-assert at any cycle after the assertion of the dfi_phyupd_ack signal.

**DDR\_PCTL\_DFITCTRLUPDMIN**

Address: Operational Base + offset (0x0280)

DFI tctrlupd\_min Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0010	tctrlupd_min Specifies the minimum number of DFI clock cycles that the dfi_ctrlupd_req signal must be asserted.

**DDR\_PCTL\_DFITCTRLUPDMAX**

Address: Operational Base + offset (0x0284)

DFI tctrlupd\_max Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0040	tctrlupd_max Specifies the maximum number of DFI clock cycles that the dfi_ctrlupd_req signal can assert.

**DDR\_PCTL\_DFITCTRLUPDDLY**

Address: Operational Base + offset (0x0288)

DFI tctrlupddly Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
3:0	RW	0x8	tctrlupd_dly Delay in DFI clock cycles between time a controller-initiated update could be started and time controller-initiated update actually starts (dfi_ctrlupd_req going high).

**DDR\_PCTL\_DFIUPDCFG**

Address: Operational Base + offset (0x0290)

DFI Update Configuration Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:2	RO	0x0	reserved
1	RW	0x1	dfi_phyupd_en Enables the support for acknowledging PHY-initiated updates: 1'b0 = Disabled 1'b1 = Enabled
0	RW	0x1	dfi_ctrlupd_en Enables the generation of controller-initiated updates: 1'b0 = Disabled 1'b1 = Enabled

**DDR\_PCTL\_DFITREFMSKI**

Address: Operational Base + offset (0x0294)

DFI Masked Refresh Interval

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RW	0x00	trefmski Time period of the masked Refresh interval This value is only used if TREFI==0. Defines the time period (in 100ns units) of the masked Refresh (REFMSK) interval. The actual time period defined is DFITREFMSKI* TOGCNT100N * internal timers clock period.

**DDR\_PCTL\_DFITCTRLUPDI**

Address: Operational Base + offset (0x0298)

DFI tctrlupd\_interval Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	tctrlupd_interval DFI controller-initiated updates interval, measured in terms of Refresh interval units. If TREFI!=0, the time period is defined as DFITCTRLUPDI*TREFI * TOGCNT100N * internal timers clock period. If TREFI==0 and DFITREFMSKI!=0, the period changes to DFITCTRLUPDI * DFITREFMSKI * TOGCNT100N * internal timers clock period. Programming a value of 0 is the same as programming a value of 1; for instance, a controller-initiated update occurs every Refresh interval.

**DDR\_PCTL\_DFITRCFG0**

Address: Operational Base + offset (0x02ac)

DFI Training Configuration 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:20	RO	0x0	reserved
19:16	RW	0x0	dfi_wrlvl_rank_sel Determines the value to drive on the output signal dfi_wrlvl_cs_n. The value on dfi_wrlvl_cs_n is the inverse of the setting in this field.
15:13	RO	0x0	reserved
12:4	RW	0x000	dfi_rdlvl_edge Determines the value to drive on the output signal dfi_rdlvl_edge. The value on dfi_rdlvl_edge is the same as the setting in this field.
3:0	RW	0x0	dfi_rdlvl_rank_sel Determines the value to drive on the output signal dfi_rdlvl_cs_n. The value on dfi_rdlvl_cs_n is the inverse of the setting in this field.

**DDR\_PCTL\_DFITRSTAT0**

Address: Operational Base + offset (0x02b0)

DFI Training Status 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:18	RO	0x0	reserved
17:16	RO	0x0	dfi_wrlvl_mode Reports the value of the input signal dfi_wrlvl_mode.
15:10	RO	0x0	reserved
9:8	RO	0x0	dfi_rdlvl_gate_mode Reports the value of the input signal dfi_rdlvl_gate_mode.
7:2	RO	0x0	reserved
1:0	RO	0x0	dfi_rdlvl_mode Reports the value of the input signal dfi_rdlvl_mode.

**DDR\_PCTL\_DFITRWRLVLEN**

Address: Operational Base + offset (0x02b4)

DFI Training dfi\_wrlvl\_en Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:9	RO	0x0	reserved
8:0	RW	0x000	dfi_wrlvl_en Determines the value to drive on the output signal dfi_wrlvl_en.

**DDR\_PCTL\_DFITRRDLVLEN**

Address: Operational Base + offset (0x02b8)

DFI Training dfi\_rdlvl\_en Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:9	RO	0x0	reserved
8:0	RW	0x000	dfi_rdlvl_en Determines the value to drive on the output signal dfi_rdlvl_en.

**DDR\_PCTL\_DFITRRDLVLGATEEN**

Address: Operational Base + offset (0x02bc)

DFI Training dfi\_rdlvl\_gate\_en Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:9	RO	0x0	reserved
8:0	RW	0x000	dfi_rdlvl_gate_en Determines the value to drive on the output signal dfi_rdlvl_gate_en.

**DDR\_PCTL\_DFISTSTAT0**

Address: Operational Base + offset (0x02c0)

DFI Status Status 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:25	RO	0x0	reserved
24:16	RO	0x000	dfi_data_byte_disable Reports the value of the output signal dfi_data_byte_disable.
15:6	RO	0x0	reserved
5:4	RO	0x0	dfi_freq_ratio Reports the value of the output signal dfi_freq_ratio.
3:2	RO	0x0	reserved
1	RO	0x0	dfi_init_start Reports the value of the output signal dfi_init_start.
0	RO	0x0	dfi_init_complete Reports the value of the input signal dfi_init_complete.

**DDR\_PCTL\_DFISTCFG0**

Address: Operational Base + offset (0x02c4)

DFI Status Configuration 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	RO	0x0	reserved
2	RW	0x0	dfi_data_byte_disable_en Enables the driving of the dfi_data_byte_disable signal. The value driven on dfi_data_byte_disable is dependent on the setting of PPCFG register. 1'b0 - Drive dfi_data_byte_disable to default value of all zeroes. 1'b1 - Drive dfi_data_byte_disable according to value as defined by PPCFG register setting. Note: should be set to 1'b1 only after PPCFG is correctly set.
1	RO	0x0	reserved
0	RW	0x0	dfi_init_start Sets the value of the dfi_init_start signal. 1'b0 - dfi_init_start is driven low 1'b1 - dfi_init_start is driven high

**DDR\_PCTL\_DFISTCFG1**

Address: Operational Base + offset (0x02c8)

DFI Status Configuration 1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:1	RO	0x0	reserved
0	RW	0x0	<p>dfi_dram_clk_disable_en Enables support of the dfi_dram_clk_disable signal with Self-Refresh (SR).</p> <p>1'b0 - Disable dfi_dram_clk_disable support in relation to SR 1'b1 - Enable dfi_dram_clk_disable support in relation to SR</p>

**DDR\_PCTL\_DFITDRAMCLKEN**

Address: Operational Base + offset (0x02d0)

DFI tdram\_clk\_enable Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x2	<p>tdram_clk_enable Specifies the number of DFI clock cycles from the de-assertion of the dfi_dram_clk_disable signal on the DFI until the first valid rising edge of the clock to the DRAM memory devices, at the PHY-DRAM boundary. If the DFI clock and the memory clock are not phase-aligned, this timing parameter should be rounded up to the next integer value.</p>

**DDR\_PCTL\_DFITDRAMCLKDIS**

Address: Operational Base + offset (0x02d4)

DFI tdram\_clk\_disable Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x2	<p>tdram_clk_disable Specifies the number of DFI clock cycles from the assertion of the dfi_dram_clk_disable signal on the DFI until the clock to the DRAM memory devices, at the PHY-DRAM boundary, maintains a low value. If the DFI clock and the memory clock are not phase-aligned, this timing parameter should be rounded up to the next integer value.</p>

**DDR\_PCTL\_DFISTCFG2**

Address: Operational Base + offset (0x02d8)

DFI Status Configuration 2 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:2	RO	0x0	reserved

Bit	Attr	Reset Value	Description
1	RW	0x0	parity_en Enables the DFI parity generation feature (driven on output signal dfi_parity_in) 1'b0 - Disable DFI parity generation 1'b1 - Enable DFI parity generation
0	RW	0x0	parity_intr_en Enable interrupt generation for DFI parity error (from input signal dfi_parity_error). 1'b0 - Disable interrupt 1'b1 - Enable interrupt

**DDR\_PCTL\_DFISTPARCLR**

Address: Operational Base + offset (0x02dc)

DFI Status Parity Clear Register

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved
1	R/W SC	0x0	parity_log_clr Set this bit to 1'b1 to clear the DFI Status Parity Log register (DFISTPARLOG). 1'b0 = Do not clear DFI status Parity Log register 1'b1 = Clear DFI status Parity Log register
0	R/W SC	0x0	parity_intr_clr Set this bit to 1'b1 to clear the interrupt generated by a DFI parity error (as enabled by DFISTCFG2.parity_intr_en). It also clears the INTRSTAT.parity_intr register field. It is automatically cleared by hardware when the interrupt has been cleared.

**DDR\_PCTL\_DFISTPARLOG**

Address: Operational Base + offset (0x02e0)

DFI Status Parity Log Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	parity_err_cnt Increments any time the DFI parity logic detects a parity error(s) (on dfi_parity_error).

**DDR\_PCTL\_DFLPCFG0**

Address: Operational Base + offset (0x02f0)

DFI Low Power Configuration 0 Register

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:28	RW	0x0	<p>dfi_lp_wakeup_dpd</p> <p>Value to drive on dfi_lp_wakeup signal when Deep Power Down mode is entered. Determines the DFI's tlp_wakeup time:</p> <ul style="list-style-type: none"> <li>4'b0000 - 16 cycles</li> <li>4'b0001 - 32 cycles</li> <li>4'b0010 - 64 cycles</li> <li>4'b0011 - 128 cycles</li> <li>4'b0100 - 256 cycles</li> <li>4'b0101 - 512 cycles</li> <li>4'b0110 - 1024 cycles</li> <li>4'b0111 - 2048 cycles</li> <li>4'b1000 - 4096 cycles</li> <li>4'b1001 - 8192 cycles</li> <li>4'b1010 - 16384 cycles</li> <li>4'b1011 - 32768 cycles</li> <li>4'b1100 - 65536 cycles</li> <li>4'b1101 - 131072 cycles</li> <li>4'b1110 - 262144 cycles</li> <li>4'b1111 - Unlimited</li> </ul>
27:25	RO	0x0	reserved
24	RW	0x0	<p>dfi_lp_en_dpd</p> <p>Enables DFI Low Power interface handshaking during Deep Power Down Entry/Exit.</p> <ul style="list-style-type: none"> <li>1'b0 - Disabled</li> <li>1'b1 - Enabled</li> </ul>
23:20	RO	0x0	reserved
19:16	RW	0x7	<p>dfi_tlp_resp</p> <p>Setting for tlp_resp time. Same value is used for both Power Down and Self-refresh and Deep Power Down modes. DFI 2.1 specification, recommends using value of 7 always.</p>

Bit	Attr	Reset Value	Description
15:12	RW	0x0	<p>dfi_lp_wakeup_sr</p> <p>Value to drive on dfi_lp_wakeup signal when Self-Refresh mode is entered. Determines the DFI's tlp_wakeup time:</p> <ul style="list-style-type: none"> <li>4'b0000 - 16 cycles</li> <li>4'b0001 - 32 cycles</li> <li>4'b0010 - 64 cycles</li> <li>4'b0011 - 128 cycles</li> <li>4'b0100 - 256 cycles</li> <li>4'b0101 - 512 cycles</li> <li>4'b0110 - 1024 cycles</li> <li>4'b0111 - 2048 cycles</li> <li>4'b1000 - 4096 cycles</li> <li>4'b1001 - 8192 cycles</li> <li>4'b1010 - 16384 cycles</li> <li>4'b1011 - 32768 cycles</li> <li>4'b1100 - 65536 cycles</li> <li>4'b1101 - 131072 cycles</li> <li>4'b1110 - 262144 cycles</li> <li>4'b1111 - Unlimited</li> </ul>
11:9	RO	0x0	reserved
8	RW	0x0	<p>dfi_lp_en_sr</p> <p>Enables DFI Low Power interface handshaking during Self-Refresh Entry/Exit.</p> <ul style="list-style-type: none"> <li>1'b0 - Disabled</li> <li>1'b1 - Enabled</li> </ul>
7:4	RW	0x0	<p>dfi_lp_wakeup_pd</p> <p>Value to drive on dfi_lp_wakeup signal when Power Down mode is entered. Determines the DFI's tlp_wakeup time:</p> <ul style="list-style-type: none"> <li>4'b0000 - 16 cycles</li> <li>4'b0001 - 32 cycles</li> <li>4'b0010 - 64 cycles</li> <li>4'b0011 - 128 cycles</li> <li>4'b0100 - 256 cycles</li> <li>4'b0101 - 512 cycles</li> <li>4'b0110 - 1024 cycles</li> <li>4'b0111 - 2048 cycles</li> <li>4'b1000 - 4096 cycles</li> <li>4'b1001 - 8192 cycles</li> <li>4'b1010 - 16384 cycles</li> <li>4'b1011 - 32768 cycles</li> <li>4'b1100 - 65536 cycles</li> <li>4'b1101 - 131072 cycles</li> <li>4'b1110 - 262144 cycles</li> <li>4'b1111 - Unlimited</li> </ul>
3:1	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
0	RW	0x0	dfi_lp_en_pd Enables DFI Low Power interface handshaking during Power Down Entry/Exit. 1'b0 - Disabled 1'b1 - Enabled

**DDR\_PCTL\_DFITRWRLVLRESP0**

Address: Operational Base + offset (0x0300)  
DFI Training dfi\_wrlvl\_resp Status 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	dfi_wrlvl_resp0 Reports the status of the dif_wrlvl_resp[31:0] signal.

**DDR\_PCTL\_DFITRWRLVLRESP1**

Address: Operational Base + offset (0x0304)  
DFI Training dfi\_wrlvl\_resp Status 1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	dfi_wrlvl_resp1 Reports the status of the dif_wrlvl_resp[63:32] signal.

**DDR\_PCTL\_DFITRWRLVLRESP2**

Address: Operational Base + offset (0x0308)  
DFI Training dfi\_wrlvl\_resp Status 2 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RO	0x00	dfi_wrlvl_resp2 Reports the status of the dif_wrlvl_resp[71:64] signal.

**DDR\_PCTL\_DFITRRDLVLRESP0**

Address: Operational Base + offset (0x030c)  
DFI Training dfi\_rdlvl\_resp Status 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	dfi_rdlvl_resp0 Reports the status of the dif_rdlvl_resp[31:0] signal.

**DDR\_PCTL\_DFITRRDLVLRESP1**

Address: Operational Base + offset (0x0310)  
DFI Training dfi\_rdlvl\_resp Status 1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	dfi_rdlvl_resp1 Reports the status of the dif_rdlvl_resp[63:32] signal.

**DDR\_PCTL\_DFITRRDLVLRESP2**

Address: Operational Base + offset (0x0314)  
DFI Training dfi\_rdlvl\_resp Status 2 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RO	0x00	dfi_rdlvl_resp2 Reports the status of the dif_rdlvl_resp[71:64] signal.

**DDR\_PCTL\_DFITRWRLVLDELAY0**

Address: Operational Base + offset (0x0318)

DFI Training dfi\_wrlvl\_delay Configuration 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	dfi_wrlvl_delay0 Sets the value to be driven on the signal dfi_wrlvl_delay_x[31:0].

**DDR\_PCTL\_DFITRWRLVLDELAY1**

Address: Operational Base + offset (0x031c)

DFI Training dfi\_wrlvl\_delay Configuration 1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	dfi_wrlvl_delay1 Sets the value to be driven on the signal dfi_wrlvl_delay_x[63:32].

**DDR\_PCTL\_DFITRWRLVLDELAY2**

Address: Operational Base + offset (0x0320)

DFI Training dfi\_wrlvl\_delay Configuration 2 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RW	0x00	dfi_wrlvl_delay2 Sets the value to be driven on the signal dfi_wrlvl_delay_x[71:64].

**DDR\_PCTL\_DFITRRDLVLDELAY0**

Address: Operational Base + offset (0x0324)

DFI Training dfi\_rdlvl\_delay Configuration 0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	dfi_rdlvl_delay0 Sets the value to be driven on the signal dfi_rdlvl_delay_x[31:0].

**DDR\_PCTL\_DFITRRDLVLDELAY1**

Address: Operational Base + offset (0x0328)

DFI Training dfi\_rdlvl\_delay Configuration 1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	dfi_rdlvl_delay1 Sets the value to be driven on the signal dfi_rdlvl_delay_x[63:32].

**DDR\_PCTL\_DFITRRDLVLDELAY2**

Address: Operational Base + offset (0x032c)

DFI Training dfi\_rdlvl\_delay Configuration 2 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RW	0x00	dfi_rdlvl_delay2 Sets the value to be driven on the signal dfi_rdlvl_delay_x[71:64].

**DDR\_PCTL\_DFITRRDLVLGATEDELAY0**

Address: Operational Base + offset (0x0330)

DFI Training dfi\_rdlvl\_gate\_delay Configuration 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	dfi_rdlvl_gate_delay0 Sets the value to be driven on the signal dfi_rdlvl_gate_delay_x[31:0].

**DDR\_PCTL\_DFITRRDLVLGATEDELAY1**

Address: Operational Base + offset (0x0334)

DFI Training dfi\_rdlvl\_gate\_delay Configuration 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	dfi_rdlvl_gate_delay1 Sets the value to be driven on the signal dfi_rdlvl_gate_delay_x[63:32].

**DDR\_PCTL\_DFITRRDLVLGATEDELAY2**

Address: Operational Base + offset (0x0338)

DFI Training dfi\_rdlvl\_gate\_delay Configuration 2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RW	0x00	dfi_rdlvl_gate_delay2 Sets the value to be driven on the signal dfi_rdlvl_gate_delay_x[71:64].

**DDR\_PCTL\_DFITRCMD**

Address: Operational Base + offset (0x033c)

DFI Training Command Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	R/W SC	0x0	dfitrcmd_start DFI Training Command Start. When this bit is set to 1, the command operation defined in the dfitrcmd_opcode field is started. This bit is automatically cleared by the controller after the command is finished. The application can poll this bit to determine when controller is ready to accept another command. This bit cannot be cleared to 1b0 by software.
30:13	RO	0x0	reserved
12:4	RW	0x000	dfitrcmd_en DFI Training Command Enable. Selects which bits of chosen DFI Training command to drive to 1'b1.
3:2	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
1:0	RW	0x0	dfitrcmd_opcode DFI Training Command Opcode. Select which DFI Training command to generate for one n_clk cycle: 2'b00 - dfi_wrlvl_load 2'b01 - dfi_wrlvl_strobe 2'b10 - dfi_rdlvl_load 2'b11 - Reserved.

**DDR\_PCTL\_IPVR**

Address: Operational Base + offset (0x03f8)

IP Version Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	ip_version ASCII value for each number in the version.

**DDR\_PCTL\_IPTR**

Address: Operational Base + offset (0x03fc)

IP Type Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x44574300	ip_type Contains the IP's identification code, which is an ASCII value to identify the component and it is currently set to the string "DWC". This value never changes.

**DDRPHY\_REG0**

Address: Operational Base + offset (0x0000)

DDR PHY register 00

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3	RW	0x1	soft reset 1, active low
2	RW	0x1	soft reset 0, active low
1:0	RO	0x0	reserved

**DDRPHY\_REG1**

Address: Operational Base + offset (0x0004)

DDR PHY register 01

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:2	RO	0x0	reserved
1:0	RW	0x0	PHY working condition select 0x0: ddr3 PHY mode 0x1: ddr2 PHY mode 0x2: lpddr2/3 PHY mode 0x3: reserved

**DDRPHY\_REG2**

Address: Operational Base + offset (0x0008)

DDR PHY register 02

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:6	RW	0x0	Write leveling CS select signal 2'b00: select CS0 and CS1 2'b01: select CS1 2'b10: select CS0
5:4	RW	0x0	DQS gating calibration CS select signal 2'b00: select CS0 and CS1 2'b01: select CS1 2'b10: select CS0
3	RW	0x0	Write leveling calibration bypass mode, active high
2	RW	0x0	Write leveling calibration control, active high
1	RW	0x0	DQS gating calibration bypass mode, active high
0	RW	0x0	DQS gating calibration control, active high

**DDRPHY\_REG3**

Address: Operational Base + offset (0x000c)

DDR PHY register 03

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6:4	RW	0x2	right channel A read ODT delay by read odt configure bypass mode
3	RO	0x0	reserved
2:0	RW	0x2	left channel A read ODT delay by read odt configure bypass mode

**DDRPHY\_REG4**

Address: Operational Base + offset (0x0010)

DDR PHY register 04

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6:4	RW	0x2	right channel B read ODT delay by read odt configure bypass mode
3	RO	0x0	reserved
2:0	RW	0x2	left channel B read ODT delay by read odt configure bypass mode

**DDRPHY\_REG5**

Address: Operational Base + offset (0x0014)

DDR PHY register 05

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RW	0x02	Write leveling load mode[7:0]

**DDRPHY\_REG6**

Address: Operational Base + offset (0x0018)

DDR PHY register 06

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:6	RW	0x0	Write leveling load mode select[1:0]
5:0	RW	0x02	Write leveling load mode[13:8]

**DDRPHY\_REGB**

Address: Operational Base + offset (0x002c)

DDR PHY register 0B

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x6	CL value DDR2 /DDR3 CAS Latency LPDDR2/3 RL value
3:0	RW	0x0	AL value DDR2/DDR3 additive latency

**DDRPHY\_REGC**

Address: Operational Base + offset (0x0030)

DDR PHY register 0C

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3:0	RW	0x0	CWL value DDR3 WRITE CAS Latency LPDDR2/3 WL value

**DDRPHY\_REG11**

Address: Operational Base + offset (0x0044)

DDR PHY register 11

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0xa	CMD PRCOMP, except for CK/CKB. The larger the value, the stronger the drive strength.
3:0	RW	0xa	CMD NRCOMP, except for CK/CKB. The larger the value, the stronger the drive strength.

**DDRPHY\_REG12**

Address: Operational Base + offset (0x0048)

DDR PHY register 12

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:2	RO	0x0	reserved
1	RW	0x1	CMD weak pull up enable, active low
0	RW	0x0	CMD weak pull down enable, active high

**DDRPHY\_REG13**

Address: Operational Base + offset (0x004c)

DDR PHY register 13

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:5	RO	0x0	reserved
4	RW	0x0	CMD DLL clock phase select in bypass mode 0: no delay      1: 90°delay
3	RW	0x1	CMD DLL enable 0: disable      1: enable
2:0	RW	0x4	CMD AND ADDRESS DLL delay 0: no delay      1: 22.5°delay 2: 45°delay      3: 67.5°delay 4: 90°delay      5: 112.5°delay 6: 135°delay      7: 157.5°delay

**DDRPHY\_REG14**

Address: Operational Base + offset (0x0050)

DDR PHY register 13

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3	RW	0x1	CK DLL clock phase select in bypass mode 0: no delay      1: 90°delay
2:0	RW	0x0	CK DLL delay 0: no delay      1: 22.5°delay 2: 45°delay      3: 67.5°delay 4: 90°delay      5: 112.5°delay 6: 135°delay      7: 157.5°delay

**DDRPHY\_REG16**

Address: Operational Base + offset (0x0058)

DDR PHY register 16

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0xa	CK/CKB PRCOMP. The larger the value, the stronger the drive strength.
3:0	RW	0xa	CK/CKB NRCOMP. The larger the value, the stronger the drive strength.

**DDRPHY\_REG20**

Address: Operational Base + offset (0x0080)

## DDR PHY register 20

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0xa	<p>Left channel A PRCOMP. The larger the value, the stronger the drive strength in the scope from A_DQ0 to A_DQ7.</p> <p>Note:</p> <p>Right channel A is the same, and the register offset is 0xc0;          Left channel B is the same, and the register offset is 0x100;          Right channel B is the same, and the register offset is 0x140;</p>
3:0	RW	0xa	<p>Left channel A NRCOMP. The larger the value, the stronger the drive strength in the scope from A_DQ0 to A_DQ7.</p> <p>Note:</p> <p>Right channel A is the same, and the register offset is 0xc0;          Left channel B is the same, and the register offset is 0x100;          Right channel B is the same, and the register offset is 0x140;</p>

**DDRPHY\_REG21**

Address: Operational Base + offset (0x0084)

DDR PHY register 21

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0xa	<p>Left channel A PRCOMP. The larger the value, the stronger the drive strength in the scope from A_DQ0 to A_DQ7.</p> <p>Note:</p> <p>Right channel A is the same, and the register offset is 0xc4;          Left channel B is the same, and the register offset is 0x104;          Right channel B is the same, and the register offset is 0x144;</p>
3:0	RW	0xa	<p>Left channel A read pull-up ODT. The larger the value, the smaller the pull-up resistance in the scope from A_DQ0 to A_DQ7.</p> <p>Note:</p> <p>Right channel A is the same, and the register offset is 0xc4;          Left channel B is the same, and the register offset is 0x104;          Right channel B is the same, and the register offset is 0x144;</p>

**DDRPHY\_REG26**

Address: Operational Base + offset (0x0098)

DDR PHY register 26

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:5	RO	0x0	reserved

Bit	Attr	Reset Value	Description								
4	RW	0x0	<p>Left channel A write DQ DLL phase select in bypass mode.</p> <p>0: no delay      1: 90°delay</p> <p>Note:</p> <p>Right channel A is the same, and the register offset is 0xd8;            Left channel B is the same, and the register offset is 0x118;            Right channel B is the same, and the register offset is 0x158;</p>								
3	RW	0x1	<p>Left channel A write DQ DLL enable, active HIGH.</p> <p>Note:</p> <p>Right channel A is the same, and the register offset is 0xd8;            Left channel B is the same, and the register offset is 0x118;            Right channel B is the same, and the register offset is 0x158;</p>								
2:0	RW	0x4	<p>Left channel A write DQ DLL delay</p> <table> <tr> <td>0: no delay</td> <td>1: 22.5°delay</td> </tr> <tr> <td>2: 45°delay</td> <td>3: 67.5°delay</td> </tr> <tr> <td>4: 90°delay</td> <td>5: 112.5°delay</td> </tr> <tr> <td>6: 135°delay</td> <td>7: 157.5°delay</td> </tr> </table> <p>Note:</p> <p>Right channel A is the same, and the register offset is 0xd8;            Left channel B is the same, and the register offset is 0x118;            Right channel B is the same, and the register offset is 0x158;</p>	0: no delay	1: 22.5°delay	2: 45°delay	3: 67.5°delay	4: 90°delay	5: 112.5°delay	6: 135°delay	7: 157.5°delay
0: no delay	1: 22.5°delay										
2: 45°delay	3: 67.5°delay										
4: 90°delay	5: 112.5°delay										
6: 135°delay	7: 157.5°delay										

**DDRPHY\_REG27**

Address: Operational Base + offset (0x009c)

DDR PHY register 27

Bit	Attr	Reset Value	Description								
31:4	RO	0x0	reserved								
3	RW	0x0	<p>Left channel A write DQS DLL phase select in bypass mode.</p> <p>0: no delay      1: 90°delay</p> <p>Note:</p> <p>Right channel A is the same, and the register offset is 0xdc;            Left channel B is the same, and the register offset is 0x11c;            Right channel B is the same, and the register offset is 0x15c;</p>								
2:0	RW	0x0	<p>Left channel A write DQS DLL delay</p> <table> <tr> <td>0: no delay</td> <td>1: 22.5°delay</td> </tr> <tr> <td>2: 45°delay</td> <td>3: 67.5°delay</td> </tr> <tr> <td>4: 90°delay</td> <td>5: 112.5°delay</td> </tr> <tr> <td>6: 135°delay</td> <td>7: 157.5°delay</td> </tr> </table> <p>Note:</p> <p>Right channel A is the same, and the register offset is 0xdc;            Left channel B is the same, and the register offset is 0x11c;            Right channel B is the same, and the register offset is 0x15c;</p>	0: no delay	1: 22.5°delay	2: 45°delay	3: 67.5°delay	4: 90°delay	5: 112.5°delay	6: 135°delay	7: 157.5°delay
0: no delay	1: 22.5°delay										
2: 45°delay	3: 67.5°delay										
4: 90°delay	5: 112.5°delay										
6: 135°delay	7: 157.5°delay										

**DDRPHY\_REG28**

Address: Operational Base + offset (0x00a0)

DDR PHY register 28

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:2	RO	0x0	reserved
1:0	RW	0x1	<p>Left channel A read DQS DLL delay            0: no delay                    1: 22.5°delay            2: 45°delay                    3: 67.5°delay</p> <p>Note:            Right channel A is the same, and the register offset is 0xe0;            Left channel B is the same, and the register offset is 0x120;            Right channel B is the same, and the register offset is 0x160;</p>

**DDRPHY\_REG70**

Address: Operational Base + offset (0x01c0)

DDR PHY register 70

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	<p>CS0 A_DM0 RX de-skew            Note:            The 0x01c4 offset register is for CS0 A_DQ0 RX de-skew            ...            The 0x01e4 offset register is for CS0 A_DQS0 RX de-skew            The 0x01e8 offset register is for CS0 A_DQSB0 RX de-skew            The 0x01ec offset register is for CS0 A_DM1 RX de-skew            The 0x01f0 offset register is for CS0 A_DQ8 RX de-skew            ...            The 0x0210 offset register is for CS0 A_DQS1 RX de-skew            The 0x0214 offset register is for CS0 A_DQSB1 RX de-skew            ...            The following offset register is for CS0 B channel            The start offset register for CS1 is 0x0300</p>
3:0	RW	0x7	<p>CS0 A_DM0 TX de-skew            Note:            The 0x01c4 offset register is for CS0 A_DQ0 TX de-skew            ...            The 0x01e4 offset register is for CS0 A_DQS0 TX de-skew            The 0x01e8 offset register is for CS0 A_DQSB0 TX de-skew            The 0x01ec offset register is for CS0 A_DM1 TX de-skew            The 0x01f0 offset register is for CS0 A_DQ8 TX de-skew            ...            The 0x0210 offset register is for CS0 A_DQS1 TX de-skew            The 0x0214 offset register is for CS0 A_DQSB1 TX de-skew            ...            The following offset register is for CS0 B channel            The start offset register for CS1 is 0x0300</p>

**DDRPHY\_REGB0**

Address: Operational Base + offset (0x02c0)

DDR PHY register B0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	A1 de-skew
3:0	RW	0x7	A0 de-skew

**DDRPHY\_REGB1**

Address: Operational Base + offset (0x02c4)

DDR PHY register B1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	A3 de-skew
3:0	RW	0x7	A2 de-skew

**DDRPHY\_REGB2**

Address: Operational Base + offset (0x02c8)

DDR PHY register B2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	A5 de-skew
3:0	RW	0x7	A4 de-skew

**DDRPHY\_REGB3**

Address: Operational Base + offset (0x02cc)

DDR PHY register B3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	A7 de-skew
3:0	RW	0x7	A6 de-skew

**DDRPHY\_REGB4**

Address: Operational Base + offset (0x02d0)

DDR PHY register B4

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	A9 de-skew
3:0	RW	0x7	A8 de-skew

**DDRPHY\_REGB5**

Address: Operational Base + offset (0x02d4)

DDR PHY register B5

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
7:4	RW	0x7	A11 de-skew
3:0	RW	0x7	A10 de-skew

**DDRPHY\_REGB6**

Address: Operational Base + offset (0x02d8)

DDR PHY register B6

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	A13 de-skew
3:0	RW	0x7	A12 de-skew

**DDRPHY\_REGB7**

Address: Operational Base + offset (0x02dc)

DDR PHY register B7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	A15 de-skew
3:0	RW	0x7	A14 de-skew

**DDRPHY\_REGB8**

Address: Operational Base + offset (0x02e0)

DDR PHY register B8

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	B1 de-skew
3:0	RW	0x7	B0 de-skew

**DDRPHY\_REGB9**

Address: Operational Base + offset (0x02e4)

DDR PHY register B9

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	RAS# de-skew
3:0	RW	0x7	B2 de-skew

**DDRPHY\_REGBA**

Address: Operational Base + offset (0x02e8)

DDR PHY register BA

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	WE# de-skew
3:0	RW	0x7	CAS# de-skew

**DDRPHY\_REGBB**

Address: Operational Base + offset (0x02ec)  
 DDR PHY register BB

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	CKB de-skew
3:0	RW	0x7	CK de-skew

**DDRPHY\_REGBC**

Address: Operational Base + offset (0x02f0)  
 DDR PHY register BC

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	CKE de-skew
3:0	RW	0x7	ODT0 de-skew

**DDRPHY\_REGBD**

Address: Operational Base + offset (0x02f4)  
 DDR PHY register BD

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	CSB0 de-skew
3:0	RW	0x7	RESETN de-skew

**DDRPHY\_REGBE**

Address: Operational Base + offset (0x02f8)  
 DDR PHY register BE

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RW	0x7	CSB1 de-skew
3:0	RW	0x7	ODT1 de-skew

**DDRPHY\_REGEC**

Address: Operational Base + offset (0x03b0)  
 DDR PHY register EC

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RW	0x64	PLL Feedback divide [7:0]

**DDRPHY\_REGED**

Address: Operational Base + offset (0x03b4)  
 DDR PHY register ED

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:5	RO	0x0	reserved
4	RW	0x1	PLL clock out enable 1: out enable 0: close clock out

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
3:2	RO	0x0	reserved
1	RW	0x1	PLL power down enable 1: power down 0: power up
0	RW	0x0	PLL Feedback divide [8]

**DDRPHY\_REGEE**

Address: Operational Base + offset (0x03b8)

DDR PHY register EE

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:5	RO	0x0	reserved
4:0	RW	0x02	PLL Pre-divide [4:0]

**DDRPHY\_REGEF**

Address: Operational Base + offset (0x03bc)

DDR PHY register EF

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:1	RO	0x0	reserved
0	RW	0x0	Select clock source 1: select DDR PHY internal PLL 0: select System PLL

**DDRPHY\_REGFO**

Address: Operational Base + offset (0x03c0)

DDR PHY register F0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3	RO	0x0	Channel B High 8bit write leveling done
2	RO	0x0	Channel B Low 8bit write leveling done
1	RO	0x0	Channel A High 8bit write leveling done
0	RO	0x0	Channel A Low 8bit write leveling done

**DDRPHY\_REGF1**

Address: Operational Base + offset (0x03c4)

DDR PHY register F1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:4	RO	0x0	Channel A High 8bit write leveling dqs value
3:0	RO	0x0	Channel A Low 8bit write leveling dqs value

**DDRPHY\_REGF2**

Address: Operational Base + offset (0x03c8)

DDR PHY register F2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
7:4	RO	0x0	Channel B High 8bit write leveling dqs value
3:0	RO	0x0	Channel B Low 8bit write leveling dqs value

**DDRPHY\_REGFA**

Address: Operational Base + offset (0x03e8)

DDR PHY register FA

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3	RO	0x0	Channel B High 8bit dqs gate sample dqs value(idqs) (3)
2	RO	0x0	Channel B Low 8bit dqs gate sample dqs value(idqs) (3)
1	RO	0x0	Channel A High 8bit dqs gate sample dqs value(idqs) (3)
0	RO	0x0	Channel A Low 8bit dqs gate sample dqs value(idqs) (3)

**DDRPHY\_REGFB**

Address: Operational Base + offset (0x03ec)

DDR PHY register FB

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6:4	RO	0x0	Calibration get the dll configure channel A low 8bit(3)
3	RO	0x0	Calibration get the ophsel configure channel A low 8bit(3)
2:0	RO	0x0	Calibration get the cyclesel configure channel A low 8bit(3)

**DDRPHY\_REGFC**

Address: Operational Base + offset (0x03f0)

DDR PHY register FC

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6:4	RO	0x0	Calibration get the dll configure channel A high 8bit(3)
3	RO	0x0	Calibration get the ophsel configure channel A high 8bit(3)
2:0	RO	0x0	Calibration get the cyclesel configure channel A high 8bit(3)

**DDRPHY\_REGFD**

Address: Operational Base + offset (0x03f4)

DDR PHY register FD

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6:4	RO	0x0	Calibration get the dll configure channel B low 8bit(3)
3	RO	0x0	Calibration get the ophsel configure channel B low 8bit(3)
2:0	RO	0x0	Calibration get the cyclesel configure channel B low 8bit(3)

**DDRPHY\_REGFE**

Address: Operational Base + offset (0x03f8)

DDR PHY register FE

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6:4	RO	0x0	Calibration get the dll configure channel B high 8bit(3)
3	RO	0x0	Calibration get the ophsel configure channel B high 8bit(3)
2:0	RO	0x0	Calibration get the cyclesel configure channel B high 8bit(3)

**DDRPHY\_REGFF**

Address: Operational Base + offset (0x03fc)

DDR PHY register FF

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:4	RO	0x0	reserved
3	RO	0x0	Channel B High 8bit Calibration done
2	RO	0x0	Channel B Low 8bit Calibration done
1	RO	0x0	Channel A High 8bit Calibration done
0	RO	0x0	Channel A Low 8bit Calibration done

**Note:**

left channel A signals: A\_DQS0, A\_DQSB0, A\_DQ7~A\_DQ0, A\_DM0

right channel A signals: A\_DQS1, A\_DQSB1, A\_DQ15~A\_DQ8, A\_DM1

left channel B signals: B\_DQS0, B\_DQSB0, B\_DQ7~B\_DQ0, B\_DM0

right channel B signals: B\_DQS1, B\_DQSB1, B\_DQ15~B\_DQ8, B\_DM1

The delayed phase is in SDRAM CK clock domain, whose frequency is equivalent to SDRAM clock.

## 8.5 Interface Description

DDR IOs are listed as following Table.

Table 8-1 DDR IO description

<b>Pin Name</b>	<b>Description</b>
CK	Active-high clock signal to the memory device.
CK_B	Active-low clock signal to the memory device.
CKE	Active-high clock enable signal to the memory device for two chip select.
CS_Bi (i=0,1)	Active-low chip select signal to the memory device. There are two

	chip select.
RAS_B	Active-low row address strobe to the memory device.
CAS_B	Active-low column address strobe to the memory device.
WE_B	Active-low write enable strobe to the memory device.
BA[2:0]	Bank address signal to the memory device.
A[15:0]	Address signal to the memory device.
DQ[31:0]	Bidirectional data line to the memory device.
DQS[3:0]	Active-high bidirectional data strobes to the memory device.
DQS_B[3:0]	Active-low bidirectional data strobes to the memory device.
DM[3:0]	Active-low data mask signal to the memory device.
ODTi (i=0,1)	On-Die Termination output signal for two chip select.
RESET	DDR3 reset signal.

## 8.6 Application Notes

### 8.6.1 State transition of PCTL

To operate PCTL, the programmer must be familiar with the available operational states and how to transition to each state from the current state.

Every software programmable register is accessible only during certain operational states. For information about what registers are accessible in each state, refer to "Software Registers," which provides this information in each register description. The general rule is that the PCTL must be in the Init\_mem or ConFigstates to successfully write most of the registers. The following tables provide the programming sequences for moving to the various states of the state machine.

#### Moving to the Init\_mem State

Step	Application	PCTL
1	Read STAT register	Returns the current PCTL state.
2	If STATctl_stat = Init_mem, go to END.	
3	If STATctl_stat =Config, go to Step9.	
4	If STATctl_stat =Access, go to Step8.	
5	If STATctl_stat = Low_power, go to Step7.	
6	Goto Step1.	PCTL is in a Transitional state and not in any of the previous operational states.
7	Write WAKEUP to SCTL.state_cmdand poll STATctl_stat= Access.	Issues SRX, moves to the Access state, updates STATctl_stat =Access when complete.
8	Write CFG to SCTL.state_cmdand poll STATctl_stat=ConFig	PCTL stalls the NIF; completes any pending transaction; issues PREA if required; moves into the ConFigstate; updates STATctl_stat =ConFigwhencomplete.
9	Write INIT to SCTL.state_cmdand poll STATctl_stat=Init_mem	Moves into the Init_memstate and updates STATctl_stat =Init_mem.
END		PCTL is in Init_mem state.

#### Moving to ConFigState

Step	Application	PCTL
1	Read STAT register.	Returns the current PCTL state.
2	If STATctl_stat= Config, goto END.	
3	If STATctl_stat= Low_power,gotoStep6.	

Step	Application	PCTL
4	If STATctl_stat= Init_memor Access,gotoStep7.	
5	GotoStep1.	PCTL is in a transitional state and is not in any of the previous operational states.
6	WriteWAKEUPtoCTL.state_cmd and poll STATctl_stat= Access.	Issues SRX, moves to the Access state, and updates STATctl_stat= Access when complete.
7	WriteCFGtoSCTL.state_cmd and poll STATctl_stat= ConFig	PCTL stalls the NIF; completes any pending transaction; issues PREA if required; moves into the ConFig state; and updates STATctl_stat = ConFig when complete.
END		PCTL is in ConFig state.

### Moving to Access State

Step	Application	PCTL
1	Read STATregister	Returns the current PCTL state.
2	If STATctl_stat= Access,goto END.	
3	If STATctl_stat= Config, gotoStep9	
4	If STATctl_stat= Init_mem, gotoStep8	
5	If STATctl_stat= Low_power, go to Step7.	
6	GotoStep1.	PCTL is in a transitional state and is not in any of the previous operational states.
7	WriteWAKEUPtoSCTL.state_cmd and poll STATctl_stat= Access. GotoEND	Issues SRX, moves to the Access state, updates STATctl_stat= Access.
8	WriteCFGtoSCTL.state_cmd and poll STATctl_stat= ConFig	Moves to the ConFig state, updates STATctl_stat= ConFig when complete.
9	WriteGOtoSCTL.state_cmd and poll STATctl_stat= Access.	Moves to the Access state, updates STATctl_stat= Access when complete.
END		PCTL is in Access state.

### Moving to Low Power State

Step	Application	PCTL
1	Read STAT register.	Returns current PCTL state.
2	If STATctl_stat =Low_power, goto END.	
3	If STATctl_stat = Access, gotoStep9	
4	If STATctl_stat = Config, gotoStep8	
5	If STATctl_stat = Init_mem, go toStep7.	
6	Goto Step1.	PCTL is in a transitional state and is not in any of the previous operational states.
7	WriteCFG to SCTL.state_cmd and poll STATctl_stat=ConFig	Moves to the ConFig state, updates STATctl_stat = ConFig when complete.
8	WriteGOtoSCTL.state_cmd and poll STATctl_stat= Access.	Moves to the Access state, updates STATctl_stat = Access when complete.

Step	Application	PCTL
9	WriteSLEEPtoSCTL.state_cmdandpoll STAT.ctl_stat=Low_power.	IssuesPDXif necessary;completes anypending transactions;issuesPRECommand; finally, issuesSRE andupdates STAT.ctl_stat = Low_power.
END		PCTLis inLowPowerstate

## 8.6.2 Initialization

### DMC Initialization

DDR PHY power-up reset sequence:

1. Configure relative system information of grf\_ddrc0\_con0 register;
2. Configure AL,CL... information of PHY registers.
3. Configure timing registers of PCTL.
4. Start PHY initialization with DDR\_PCTL\_DFISTCFG0[0] register of PCTL and wait PHY initialization finish with DDR\_PCTL\_DFISTSTAT0[0] register of PCTL.
5. Power up DRAM by DDR\_PCTL\_POWCTL[0] register of PCTL and wait power up DRAM finish with DDR\_PCTL\_POWSATA[0] register of PCTL.
6. Initiate DRAM with MRS command sent by PCTL.
7. After ddr sdram initialization done, start PHY dqs calibration with PHYREG02 register and wait calibration finish with PHYREGFF register.
8. (optional) After dqs calibration, start write leveling training with PHYREG02 register and wait write leveling training finish with PHYJREGF0 register.
9. Configure PCTL to move to Access State.
10. Start Write and Read.

### DDR3 Initialization Sequence

The initialization steps for DDR3 SDRAMs are as follows:

1. Write Deselect command to MCMD, specifying an additional delay in terms of internal timers clock cycles of at least tXPR.
2. Poll MCMD.start\_cmd = 0.
3. Write MR2 command to MCMD and poll MCMD.start\_cmd = 0.
4. Write MR3 command to MCMD and poll MCMD.start\_cmd = 0.
5. Write MR1 command to MCMD and poll MCMD.start\_cmd = 0.
6. Write MR0 command to MCMD, for "DLL Reset", and poll MCMD.start\_cmd = 0.
7. Write ZQCL command to MCMD and poll MCMD.start\_cmd = 0. If MCFG1.zq\_resistor\_shared=1 ensure that ZQCL commands are sent to one rank at a time.

### LPDDR3 Initialization Sequence

The initialization steps for LPDDR2/LPDDR3 SDRAMs are as follows:

1. Write MRW(Reset) command to MCMD and poll MCMD.start\_cmd = 0. MRW(Reset) is to Mode Register Address 0x3F and the value should be 0x00. So MCMD.lpddr23\_addr = 0x003F.
2. Write Deselect command to MCMD, specifying an additional delay in terms of internal timers clock cycles of at least tINIT5 = 10 us. May need to send more than 1 Deselect command to satisfy tINIT5 = 10 us.
3. If ZQ Calibration command is supported (LPDDR2-S4 or LPDDR3), Write MRW(ZQ initialization calibration) command to MCMD and poll MCMD.start\_cmd = 0. MRW(ZQ initialization calibration) is to Mode Register Address 0x0A and the value should be 0xFF. So MCMD.lpddr23\_addr = 0xFF0A. Set MCMD.cmd\_add\_del such that tZQINIT=1us is met before next step. If MCFG1.zq\_resistor\_shared=1 ensure that ZQINIT commands are sent to one rank at a time.
4. Write MR2 command to MCMD and poll MCMD.start\_cmd = 0.
5. Write MR1 command to MCMD and poll MCMD.start\_cmd = 0.
6. Write MR3 command to MCMD and poll MCMD.start\_cmd = 0.

## 8.6.3 Low Power Operation

Low\_power state can be entered/exited via following ways:

- Software control of PCTL State machine (highest priority)

- Hardware Low Power Interface (middle priority)
- Auto Self-Refresh feature (lowest priority)

Note the priority of requests from Access to Low\_power is highlighted above. The STAT.ip\_trig register field reports which of the 3 requests caused the entry to Low\_power state.

### Software control of PCTL State

The application can request via software to enter the memories into Self-Refresh state by issuing the SLEEP command by programming SCTL.PCTL responds to the software request by moving into the Low\_power operational state and issuing the SRE command to the memories. Note that the Low\_power state can only be reached from the Access state.

In a similar fashion, the application requests to exit the memories from Self-Refresh by issuing a WAKEUP command by programming SCTL. PCTL responds to the WAKEUP command issuing SRX and restoring normal NIF address channel operation.

### Hardware Low Power Interface

The hardware low power interface can also be used to enter/exit Self-Refresh. The functionality is enabled by setting SCFG.hw\_low\_power\_en=1. Once that bit is set, the input c\_sysreq has the ability to trigger entry into the Low Power configuration state just like the software methodology (SCTL.state\_cmd=SLEEP). A hardware Low Power entry trigger will be ignored/denied if the input c\_active\_in=1 or n\_valid=1. It may be accepted if c\_active\_in=0 and n\_valid=0, depending on the current state of the PCTL. When SCFG.hw\_low\_power\_en=1, the outputs c\_sysack and c\_active provide feedback as required by the AXI low power interface specification (this interface's operation is defined by the AXI specification). c\_sysack acknowledges the request to go into the Low\_power state, and c\_active indicates when the PCTL is actually in the Low\_power state.

The c\_active output could also be used by an external Low Power controller to decide when to request a transition to low power. When MCFG1.hw\_idle > 0, c\_active = 1'b0 indicates that the NIF has been idle for at least MCFG1.hw\_idle \* 32 \* n\_clk cycles while in the Access state. When in low power the c\_active output can be used by an external Low Power controller to trigger a low power exit. c\_active will be driven high when either c\_active\_in or n\_valid are high. The path from c\_active\_in and n\_valid to c\_active is asynchronous so even if the clocks have been removed c\_active will assert. The Low Power controller should re-enable the clocks when c\_active is driven high while in the Low\_power state.

### Auto Power Down/Self-Refresh

The Power Down and/or Self-Refresh sequence is automatically started by PCTL when the NIF address channel is idle for a number of cycles, depending on the programmed value in MCFG.pd\_idle and MCFG1.sr\_idle.

Following table outlines the effect of these settings in conjunction with NIF being idle.

<b>pd_idle</b>	<b>sr_idle</b>	<b>Memory modes</b>	<b>Memory Type</b>
0	0	none	All
>0	0	Power Down	All
0	>0	Self-Refresh	All
>0	>0	Power Down -> Self Refresh <sup>3</sup>	All

Note:

1. Power Down is entered if NIF is idle for pd\_idle. Following on from that, if NIF continues to be idle for a further sr\_idle\*32 cycles, Power Down is exited and Self-Refresh is entered.

2. Following on from that, if NIF continues to be idle for a further sr\_idle\*32 cycles, Power Down is exited and Self-Refresh is entered.

### Removing PCTL's n\_clk

In DDR3 and LPDDR3, the relationship between SRE/SRX and stopping/starting the memory clock (CK) are formalized and are accounted for automatically by PCTL. With DDR3 and LPDDR3, CK should only be stopped after PCTL has reached the Low\_power state. The current operational state can be verified by reading STAT.ctl\_stat. The CK must be started and stable before the Software or Hardware Low Power Interface attempts to take the memory out of Self-Refresh.

PCTL's n\_clk can be safely removed when PCTL is in Low Power state. The sequences outlined in following two tables should be followed for safe operation:

<b>Step</b>	<b>Application</b>	<b>PCTL</b>
-------------	--------------------	-------------

1	Write SLEEP to SCTL.state_cmd and poll STAT.ctl_stat = LOW_POWER.	Tells PCTL to move memories into Self-Refresh and waits until this completes.
2	Write TREFI=0. Also, write DFITCRLUPDI=0 and DFIREFMSKI=0, if they are not already 0.	Stops any MC-driven DFI updates occurring internally with PCTL
3	Wait a minimum interval which is equivalent to the PCTL's Refresh Interval (previous value of TREFI*TOGCNT100N*internal timers clock period;	Ensures any already scheduled PHY/PVT updates have completed successfully.
4	Stop toggling n_clk to PCTL.	n_clk logic inside PCTL is stopped.
end		

Step	Application	PCTL
1	Drive c_active_in low	Confirms that system external to PCTL can accept a Low-power request
2	Drive c_sysreq low	System Low-power request
3	Wait for PCTL to drive c_sysack low	PCTL Low-power request acknowledgement
4	Check value of c_active when Step 3 occurs. - ifc_active=1, request denied. Cannot remove n_clk. Go to END. - ifc_active=0, request accepted.	PCTL low-power request status response
5	Stop toggling n_clk to PCTL	n_clk logic inside PCTL is stopped
end		

#### 8.6.4 TX DLLs

All high speed IO signals' phase can be adjusted by TX DLLs. Table 1-3 illustrates these DLLs.

Table 8-2 DDR PHY TX DLLs Delay Step

Offset	Bit	Control Signal Phase	Default	Description
0x4c	2~0	CMD	0x4	CMD DLL delay step
0x50	2~0	CK	0x0	CK DLL delay step
0x98	2~0	A_DM0, A_DQ7~A_DQ0	0x4	DM and DQ DLL Signal delay step
0xd8	2~0	A_DM1, A_DQ15~A_DQ8	0x4	
0x118	2~0	B_DM0, B_DQ7~B_DQ0	0x4	
0x158	2~0	B_DM1, B_DQ15~B_DQ8	0x4	
0x9c	2~0	A_DQS0, A_DQSB0	0x0	TX DQS DLL Signal delay step
0xdc	2~0	A_DQS1, A_DQSB1	0x0	
0x11c	2~0	B_DQS0, B_DQSB0	0x0	
0x15c	2~0	B_DQS1, B_DQSB1	0x0	

Step 0x0 values means no phase delay, and 0x4 increases delay phase to 90 deg, 0x7 values corresponds to maximum phase delay. All DLLs having 8 delay steps which can get 90 deg phase delay by setting 0x4.

#### 8.6.5 RX DLLs

The RX DLLs are used for sample RX DQS signals with proper phase delay and pulse edges. The DQS squelch (Rx Squelch) signal opens a window for passing RX DQS pulses, both RX DQS and DQS squelch signal phase can be adjusted by corresponding DLLs.

Table 8-3 DDR PHY RX DQS Delay Step

Offset	Bit	Control Signal Phase	Default	Description
--------	-----	----------------------	---------	-------------

0x0a0	1~0	A_DQS0,A_DQSB0	0x1	Read DQS DLL delay phase
0x0e0	1~0	A_DQS1,A_DQSB1	0x1	
0x120	1~0	A_DQS0,A_DQSB0	0x1	
0x160	1~0	A_DQS1,A_DQSB1	0x1	
0x0b0	7~5	left channel A DQS gating	0x1	CS0 DQS gating delay, unit x1 clock cycle
0x0f0	7~5	right channel A DQS gating	0x1	
0x130	7~5	left channel B DQS gating	0x1	
0x170	7~5	reft channel B DQS gating	0x1	
0x0b4	7~5	left channel A DQS gating	0x1	CS1 DQS gating delay, unit x1 clock cycle
0x0f4	7~5	right channel A DQS gating	0x1	
0x134	7~5	left channel B DQS gating	0x1	
0x174	7~5	reft channel B DQS gating	0x1	
0x0b0	4~3	left channel A DQS gating	0x1	CS0 additive and accumulative DQS gating delay, unit 4x clock cycle
0x0f0	4~3	right channel A DQS gating	0x1	
0x130	4~3	left channel B DQS gating	0x1	
0x170	4~3	reft channel B DQS gating	0x1	
0x0b4	4~3	left channel A DQS gating	0x1	CS1 additive and accumulative DQS gating delay, unit 4x clock cycle
0x0f4	4~3	right channel A DQS gating	0x1	
0x134	4~3	left channel B DQS gating	0x1	
0x174	4~3	reft channel B DQS gating	0x1	
0x0b0	2~0	left channel A DQS gating	0x4	CS0 additive and accumulative DQS gating delay, unit per DLL step
0x0f0	2~0	right channel A DQS gating	0x4	
0x130	2~0	left channel B DQS gating	0x4	
0x170	2~0	reft channel B DQS gating	0x4	
0x0b4	2~0	left channel A DQS gating	0x4	CS1 additive and accumulative DQS gating delay, unit per DLL step
0x0f4	2~0	right channel A DQS gating	0x4	
0x134	2~0	left channel B DQS gating	0x4	
0x174	2~0	reft channel B DQS gating	0x4	

### 8.6.6 High Speed IO Drive Strength

The tuning range of driver resistance is 28ohm to 138ohm. By default, 0x5 is 46ohm for DDR3 CMD driver. When the control bit is set to be larger, the drive strength becomes stronger.

Table 8-4 DM/DQ/DQS/CMD Driver output resistance

Offset	Bit	Default	Description
0x44	7~4	0xa	adjustable CMD pull-up resistance
	3~0	0xa	adjustable CMD pull-down resistance
0x58	7~4	0xa	adjustable CK pull-up resistance
	3~0	0xa	adjustable CK pull-down resistance

Table 8-5 DM, DQ Signal Drive Strength Register

Offset	Bit	Default	Description
0x80	7~4	0xa	pull-up driving resistance for A_DQ0~A_DQ7
	3~0	0xa	pull-down driving resistance for A_DQ0~A_DQ7
0xc0	7~4	0xa	pull-up driving resistance for A_DQ8~A_DQ15
	3~0	0xa	pull-down driving resistance for A_DQ8~A_DQ15
0x100	7~4	0xa	pull-up driving resistance for B_DQ0~B_DQ7
	3~0	0xa	pull-down driving resistance for B_DQ0~B_DQ7
0x140	7~4	0xa	pull-up driving resistance for B_DQ8~B_DQ15
	3~0	0xa	pull-down driving resistance for B_DQ8~B_DQ15

The value is larger, the drive strength is stronger.

Table 8-6 DM/DQ/DQS/CMD Driver output resistance with control bit

Control bit	4'b0000	4'b0001	4'b0010	4'b0011	4'b0100	4'b0101	4'b0110	4'b0111
-------------	---------	---------	---------	---------	---------	---------	---------	---------

Pull-up resistance	$+\infty$	272ohm	135ohm	91ohm	68ohm	54ohm	45ohm	38ohm
Pull-down resistance	$+\infty$	272ohm	135ohm	91ohm	68ohm	54ohm	45ohm	38ohm
<b>Control bit</b>	<b>4'b1000</b>	<b>4'b1001</b>	<b>4'b1010</b>	<b>4'b1011</b>	<b>4'b1100</b>	<b>4'b1101</b>	<b>4'b1110</b>	<b>4'b1111</b>
Pull-up resistance	68ohm	54ohm	45ohm	39ohm	34ohm	30ohm	27ohm	25ohm
Pull-down resistance	68ohm	54ohm	45ohm	39ohm	34ohm	30ohm	27ohm	25ohm

Table 8-7 DM/DQ/DQS RX ODT resistance with control bit

<b>Control bit</b>	<b>4'b0000</b>	<b>4'b0001</b>	<b>4'b0010</b>	<b>4'b0011</b>	<b>4'b0100</b>	<b>4'b0101</b>	<b>4'b0110</b>	<b>4'b0111</b>
Pull-up resistance	$+\infty$	1116ohm	558ohm	372ohm	279ohm	223ohm	186ohm	159ohm
Pull-down resistance	$+\infty$	1116ohm	558ohm	372ohm	279ohm	223ohm	186ohm	159ohm
<b>Control bit</b>	<b>4'b1000</b>	<b>4'b1001</b>	<b>4'b1010</b>	<b>4'b1011</b>	<b>4'b1100</b>	<b>4'b1101</b>	<b>4'b1110</b>	<b>4'b1111</b>
Pull-up resistance	139ohm	124ohm	112ohm	101ohm	93ohm	86ohm	80ohm	74ohm
Pull-down resistance	139ohm	124ohm	112ohm	101ohm	93ohm	86ohm	80ohm	74ohm

## 8.6.7 PHY Low Speed Mode (200MHz)

DDR PHY supports low speed to high speed DDR3 by using two operating mode: normal delay line mode up to 800Mbps or more, low power mode where we support any speed up to 533Mbps. If all TX DLLs are bypassed, the PHY will enter low power state.

The DDR PHY enters low power mode when setting DLLs into Bypass mode. Table 10-9 illustrates related register settings.

Table 8-8 Low Power DLL Setting

Offset	Bit	Default	Low power Setting	Description
0x290	4	0x0	0x1	right channel B TX DQ DLL in bypass mode
	3	0x0	0x1	left channel B TX DQ DLL in bypass mode
	2	0x0	0x1	right channel A TX DQ DLL in bypass mode
	1	0x0	0x1	left channel A TX DQ DLL in bypass mode
	0	0x0	0x1	CMD/CK DLL in bypass mode
	0x4c	4	0x0	CMD DLL phase select
	0x50	3	0x0	CK DLL phase select
	0x98	4	0x0	A_DQ0~A_DQ7 TX DLL phase select
	0x9c	3	0x0	A_DQS0/A_DQSB0 TX DLL phase select
	0xd8	4	0x0	A_DQ8~A_DQ15 TX DLL phase select
	0xdc	3	0x0	A_DQS1/A_DQSB1 TX DLL phase select
	0x118	4	0x0	B_DQ0~B_DQ7 TX DLL phase select
	0x11c	3	0x0	B_DQS0/B_DQSB0 TX DLL phase select
	0x158	4	0x0	B_DQ8~B_DQ15 TX DLL phase select
	0x15c	3	0x0	B_DQS1/B_DQSB1 TX DLL phase select

## 8.6.8 Per bit de-skew tuning

Per-bit de-skew is designed for compensating PCB trace mismatch, DDR PHY support skew individually adjustable for all PHY signals. There are eight steps for each bit de-skew adjusting, and the adjust resolution under different corners is shown below:

Table 8-9 per-bit de-skew tuning resolution

	<b>ff</b>	<b>tt</b>	<b>ss</b>
de-skew resolution	15ps	20ps	32ps

Pre-bit de-skew is realized with inverter chain delay, per-bit de-skew control signals select how much inverters are connected to data path, the minimum resolution is determined by the two inverters minimum delay.

TX path deskew and RX path deskew employ same delay line, and they have same deskew tuning resolution. Minimum RX deskew tuning resolution can be about 28ps with SMIC55II tt corner process, and we can re-design tuning resolution according to system and customer requirement.

## 8.6.9 DDR PHY Calibration

DDR PHY auto dqs calibration function has been implemented in the PHY. The entire training processes only need to configure the register to start and wait for finish.

The entire training process is as follows:

1. PHY's register is reset, the setup is complete.
2. Send the initial command to dram and complete dram initialization.
3. Set the PHY's register beginning calibration.

Offset	Bit	Default	Description
0x8	5~4	0x0	DQS gating calibration CS select signal 2'b00: select CS0 and CS1    2'b01: select CS1 2'b10: select CS0
	1	0x0	set calibration bypass mode(1:bypass mode; 0:nomal)
	0	0x0	set calibration start (1: start; 0: stop)

4. Wait for the calibration finish by PHYREGFF.
5. Normal read and writes operation can begin.

## 8.6.10 DDR PHY Write Leveling Training

DDR PHY auto write leveling training function has been implemented in the PHY. The entire training processes only need to configure the register to start and wait for finish.

The entire training process is as follows:

1. PHY's register is reset, the setup is complete.
2. Send the initial command to dram and complete dram initialization.
3. Set the PHYREG05 and PHYREG06 to configure the dram mode register which used to enable dram write leveling training function.
4. Set the PHY's register to begin training.

Offset	Bit	Default	Description
0x8	7~6	0x0	Write leveling CS select signal 2'b00: select CS0 and CS1    2'b01: select CS1 2'b10: select CS0
	3	0x0	Write leveling calibration bypass mode, active high
	2	0x0	Write leveling calibration control, active high

5. Wait for the calibration finish by PHYREGF0.
6. Normal read and writes operation can begin.

## Chapter 9 Embedded SRAM

### 9.1 Overview

The Embedded SRAM is the AXI slave device, which supports read and write access to provide system fast access data storage

#### 9.1.1 Features supported

- Provide 36KB access space
- Support security and non-security access
- Security or non-security space is software programmable
- Security space is nx4KB(up to whole memory space)
- Support 64bit AXI bus

#### 9.1.2 Features not supported

- Don't support AXI lock transaction
- Don't support AXI exclusive transaction
- Don't support AXI cache function
- Don't support AXI protection function

### 9.2 Block Diagram

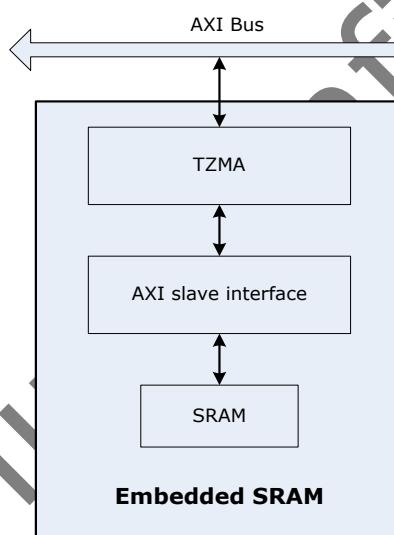


Fig. 9-1 Embedded SRAM block diagram

### 9.3 Function Description

#### 9.3.1 TZMA

Please refer to 7.3.3 for TZMA functional description

#### 9.3.2 AXI slave interface

The AXI slave interface is bridge which translate AXI bus access to SRAM interface.

#### 9.3.3 Embedded SRAM access path

The Embedded SRAM can only be accessed by Cortex-A7, DMAC\_BUS and CRYPTO

#### 9.3.4 Remap

The Embedded SRAM support remap.

Before remap, the Embedded SRAM address range is 0x1008\_0000~0x1008\_ffff,

After set remap, (ref Security GRF register SGRF\_SCON0, bit[10]), the system can still access the Embedded SRAM by the old address. at same time, the system also can access the Embedded SRAM by the new address 0x1010\_0000 ~ 0x1010\_ffff (include the bootaddr)

## Chapter 10 Nand Flash Controller (NandC)

### 10.1 Overview

Nand Flash Controller (NandC) is used to control data transmission from host to flash device or from flash device to host. NandC is connected to AHB BUS through an AHB Master and an AHB Slave. The data transmission between host and external memory can be done through AHB Master interface or AHB Slave interface.

NandC supports the following features:

- Software Interface Type
  - Support directly mode
  - Support LLP(Linked List Pointer) mode
- Flash Interface Type
  - Support Asynchronous Flash Interface with 8bits data width ("Asyn8x" for short)
  - Support Asynchronous Flash Interface with 16bits data width ("Asyn16x" for short)
  - Support ONFI Synchronous Flash Interface ("ONFI Syn" for short)
  - Support Toggle Flash Interface ("Toggle" for short)
  - Support 4 flash devices at most
- Flash Type
  - Support Managed NAND Flash(LBA) and Raw NAND Flash(NO-LBA)
  - Support SLC/MLC/TLC Flash
- Flash Interface Timing
  - Asyn8x: configurable timing, one byte per two host clocks at the fastest speed
  - Asyn16x: configurable timing, two bytes per two host clocks at the fastest speed
  - ONFI Syn: configurable timing, two bytes per two host clocks at the fastest speed
  - Toggle: configurable timing, two byte per two host clocks at the fastest speed
- Randomizer Ability
  - Support two randomizer mode with different polynomial
  - Support two randomizer width, 8bit and 16bit parallel
- BCH/ECC Ability
  - 16bit/1KB BCH/ECC: support 16bitBCH/ECC, which can detect and correct up to 16 error bits in every 1K bytes data
  - 24bit/1KB BCH/ECC: support 24bitBCH/ECC, which can detect and correct up to 24 error bits in every 1K bytes data
  - 40bit/1KB BCH/ECC: support 40bitBCH/ECC, which can detect and correct up to 40 error bits in every 1K bytes data
  - 60bit/1KB BCH/ECC: support 60bitBCH/ECC, which can detect and correct up to 60 error bits in every 1K bytes data
  - 8bit/512B BCH/ECC: support 8bitBCH/ECC, which can detect and correct up to 8 error bits in every 512 bytes data
  - 12bit/512B BCH/ECC: support 12bitBCH/ECC, which can detect and correct up to 12 error bits in every 512 bytes data
  - 20bit/512B BCH/ECC: support 20bitBCH/ECC, which can detect and correct up to 20 error bits in every 512 bytes data
  - 30bit/512B BCH/ECC: support 30bitBCH/ECC, which can detect and correct up to 30 error bits in every 512 bytes data
  - 16bit/512B BCH/ECC: support 16bitBCH/ECC, which can detect and correct up to 16 error bits in every 512 bytes data
  - 24bit/512B BCH/ECC: support 24bitBCH/ECC, which can detect and correct up to 24 error bits in every 512 bytes data
  - 40bit/512B BCH/ECC: support 40bitBCH/ECC, which can detect and correct up to 40 error bits in every 512 bytes data
  - 60bit/512B BCH/ECC: support 60bitBCH/ECC, which can detect and correct up to 60 error bits in every 512 bytes data
- Transmission Ability
  - Support 16K bytes data transmission at a time at most
  - Support two transfer working modes: Bypass or DMA

- Support two transfer codeword size for Managed NAND Flash: 1024 bytes/codeword or 512 bytes/codeword
- Internal Memory
  - 2 built-in SRAMs, and the size is 1k bytes respectively
  - Can be accessed by other masters
  - Can be operated in pingpong mode by other masters

## 10.2 Block Diagram

NandC comprises with:

- MIF: AHB Master Interface
- SIF : AHB Slave Interface
- SRIF : SRAM Interface
- TRANSC : Transfer Controller
- LLPC : LLP Controller
- BCHENC : BCH Encoder
- BCHDEC : BCH Decoder
- RANDMZ : Randomizer
- FIF\_GEN : Flash Interface Generation
- DLC : Delay Line Controller

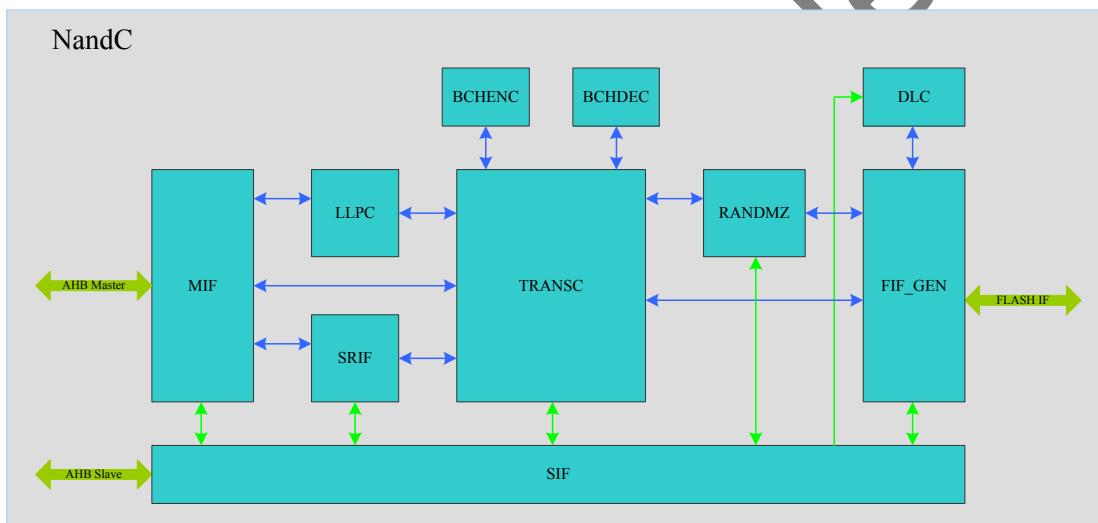


Fig. 10-1 NandC Block Diagram

## 10.3 Function Description

### 10.3.1 AHB Interface

There is an AHB master interface in NandC, which is selectable and configurable. It is responsible for transferring data from external memory to internal memory when flash program, or inverse when flash read; and transferring LLP data from external memory to internal register file when LLP is active.

There is an AHB slave interface in NandC. It is responsible for accessing registers and internal memories. The addresses of these registers and memories are listed in Register Description section.

### 10.3.2 Flash Type/Flash Interface

Flash device with different types of interfaces is supported. These interfaces include: asynchronous 8bits flash interface, asynchronous 16bits flash interface, ONFI synchronous flash interface, toggle flash interface, and so on. You can select one of them by software (configure FMCTL) to suit for these devices. Also you can configure their timing parameters by software (configure FMWAIT\_ASYN/FMWAIT\_SYN) to have your desired rate.

### 10.3.3 Linked List Pointer Mode (LLP)

To save the software resource and improve the performance, a LLP is add, which is selectable.

When LLP is selected, the flash operation instructions stored in external memory with specific format should be loaded for flash working. The detailed format and working flow are referred to 15.7.8.

### 10.3.4 BCH Encoder/BCH Decoder

The BCH Encoder is responsible for encoding data to be written into flash device. The max encoded length is 1133bytes,in which the data length is 1024bytes, system information is 4bytes, BCH code is 105bytes.

The BCH Decoder is responsible for decoding data read from flash device. The max decoded length is 1133bytes, in which the data length is 1024bytes, spare length is 109bytes.

### 10.3.5 Randomizer

To improve device lifetime, a randomizer is added in NandC. It includes two parts: Scrambler and Descrambler, which is responsible for scrambling data to be written into flash after bch encoding, and descrambling data read from flash before bch decoding.

### 10.3.6 Delay Line Controller

For ONFI Synchronous Flash or Toggle Flash, the data read from flash follows with a strobe signal: DQS, where a skew between them exists. To remove the skew and improve the timing between data and DQS, a Delay Line Controller is needed. It is responsible for detecting the phase of the signal similar to DQS, determining the element number to be shifted, and then shifting the DQS with the determined number.

## 10.4 Register Description

### 10.4.1 Internal Address Mapping

Slave address can be divided into different length for different usage, which is shown as follows.

Table 10-1 NandC Address Mapping

Base Address[12:8]	Device	Address Length	Offset Address Range
5'b00_00x(x=0, 1)	FLR	512 BYTE	0x0000 ~ 0x01ff
5'b00_01x(x=0, 1)	SPR	512 BYTE	0x0200 ~ 0x03ff
5'b00_10x(x=0, 1)	FLR1	512 BYTE	0x0400 ~ 0x05ff
5'b01_000	Flash0	256 BYTE	0x0800 ~ 0x08ff
5'b01_001	Flash1	256 BYTE	0x0900 ~ 0x09ff
5'b01_010	Flash2	256 BYTE	0xa00 ~ 0xaaff
5'b01_011	Flash3	256 BYTE	0xb00 ~ 0xbff
5'b01_100	Flash4	256 BYTE	0xc00 ~ 0cff
5'b01_101	Flash5	256 BYTE	0xd00 ~ 0xdff
5'b01_110	Flash6	256 BYTE	0xe00 ~ 0xeaff
5'b01_111	Flash7	256 BYTE	0xf00 ~ 0xffff
5'b10_0xx(x=0, 1)	Sram0	1K BYTE	0x1000 ~ 0x13ff
5'b10_1xx(x=0, 1)	Sram1	1K BYTE	0x1400 ~ 0x17ff
5'b11_000	FIFO	1K BYTE	0x1800 ~ 0x1bff

### 10.4.2 Registers Summary

Name	Offset	Size	Reset Value	Description
NANDC_FMCTL	0x0000	W	0x00000200	Flash Interface Control Register
NANDC_FMWAIT_ASYN	0x0004	W	0x3f3ff7ff	Flash Timing Control Register For Asynchronous Timing
NANDC_FLCTL	0x0008	W	0x00100000	Internal Transfer Control Register
NANDC_BCHCTL	0x000c	W	0x00000008	BCH Control Register

Name	Offset	Size	Reset Value	Description
NANDC_MTRANS_CFG	0x0010	W	0x000001d0	Bus Transfer Configuration Register
NANDC_MTRANS_SADDR0	0x0014	W	0x00000000	Start Address Register For Page Data Transmission
NANDC_MTRANS_SADDR1	0x0018	W	0x00000000	Start Address Register For Spare Data Transmission
NANDC_MTRANS_STAT	0x001c	W	0x00000000	Bus Transfer Status Register
NANDC_BCHST0	0x0020	W	0x04000000	BCH Status Register For Codeword 0~1
NANDC_BCHST1	0x0024	W	0x00000000	BCH Status Register For Codeword 2~3
NANDC_BCHST2	0x0028	W	0x00000000	BCH Status Register For Codeword 4~5
NANDC_BCHST3	0x002c	W	0x00000000	BCH Status Register For Codeword 6~7
NANDC_BCHST4	0x0030	W	0x00000000	BCH Status Register For Codeword 8~9
NANDC_BCHST5	0x0034	W	0x00000000	BCH Status Register For Codeword 10~11
NANDC_BCHST6	0x0038	W	0x00000000	BCH Status Register For Codeword 12~13
NANDC_BCHST7	0x003c	W	0x00000000	BCH Status Register For Codeword 14~15
NANDC_BCHLOC0	0x0040	W	0x00000000	BCH Error Bit Location Number Register For Codeword 0~5
NANDC_BCHLOC1	0x0044	W	0x00000000	BCH Error Bit Location Number Register For Codeword 6~11
NANDC_BCHLOC2	0x0048	W	0x00000000	BCH Error Bit Location Number Register For Codeword 12~17
NANDC_BCHLOC3	0x004c	W	0x00000000	BCH Error Bit Location Number Register For Codeword 24~29
NANDC_BCHLOC4	0x0050	W	0x00000000	BCH Error Bit Location Number Register For Codeword 24~29
NANDC_BCHLOC5	0x0054	W	0x00000000	BCH Error Bit Location Number Register For Codeword 30~31
NANDC_BCHLOC6	0x0058	W	0x00000000	Highest Bit For BCH Error Bit Location Number Register
NANDC_BCHDE0_0	0x0070	W	0x00000000	BCH decode result of 0th error bit for codeword 0
NANDC_BCHDE0_1	0x0074	W	0x00000000	BCH decode result of 1th error bit for codeword 0
NANDC_BCHDE0_2	0x0078	W	0x00000000	BCH decode result of 2th error bit for codeword 0
NANDC_BCHDE0_3	0x007c	W	0x00000000	BCH decode result of 3th error bit for codeword 0

Name	Offset	Size	Reset Value	Description
NANDC_BCHDE0_4	0x0080	W	0x00000000	BCH decode result of 4th error bit for codeword 0
NANDC_BCHDE0_5	0x0084	W	0x00000000	BCH decode result of 5th error bit for codeword 0
NANDC_BCHDE0_6	0x0088	W	0x00000000	BCH decode result of 6th error bit for codeword 0
NANDC_BCHDE0_7	0x008c	W	0x00000000	BCH decode result of 7th error bit for codeword 0
NANDC_BCHDE0_8	0x0090	W	0x00000000	BCH decode result of 8th error bit for codeword 0
NANDC_BCHDE0_9	0x0094	W	0x00000000	BCH decode result of 9th error bit for codeword 0
NANDC_BCHDE0_10	0x0098	W	0x00000000	BCH decode result of 10th error bit for codeword 0
NANDC_BCHDE0_11	0x009c	W	0x00000000	BCH decode result of 11th error bit for codeword 0
NANDC_BCHDE0_12	0x00a0	W	0x00000000	BCH decode result of 12th error bit for codeword 0
NANDC_BCHDE0_13	0x00a4	W	0x00000000	BCH decode result of 13th error bit for codeword 0
NANDC_BCHDE0_14	0x00a8	W	0x00000000	BCH decode result of 14th error bit for codeword 0
NANDC_BCHDE0_15	0x00ac	W	0x00000000	BCH decode result of 15th error bit for codeword 0
NANDC_BCHDE0_16	0x00b0	W	0x00000000	BCH decode result of 16th error bit for codeword 0
NANDC_BCHDE0_17	0x00b4	W	0x00000000	BCH decode result of 17th error bit for codeword 0
NANDC_BCHDE0_18	0x00b8	W	0x00000000	BCH decode result of 18th error bit for codeword 0
NANDC_BCHDE0_19	0x00bc	W	0x00000000	BCH decode result of 19th error bit for codeword 0
NANDC_BCHDE0_20	0x00c0	W	0x00000000	BCH decode result of 20th error bit for codeword 0
NANDC_BCHDE0_21	0x00c4	W	0x00000000	BCH decode result of 21th error bit for codeword 0
NANDC_BCHDE0_22	0x00c8	W	0x00000000	BCH decode result of 22th error bit for codeword 0
NANDC_BCHDE0_23	0x00cc	W	0x00000000	BCH decode result of 23th error bit for codeword 0
NANDC_BCHDE1_0	0x00d0	W	0x00000000	BCH decode result of 0th error bit for codeword 1
NANDC_BCHDE1_1	0x00d4	W	0x00000000	BCH decode result of 1th error bit for codeword 1

Name	Offset	Size	Reset Value	Description
NANDC_BCHDE1_2	0x00d8	W	0x00000000	BCH decode result of 2th error bit for codeword 1
NANDC_BCHDE1_3	0x00dc	W	0x00000000	BCH decode result of 3th error bit for codeword 1
NANDC_BCHDE1_4	0x00e0	W	0x00000000	BCH decode result of 4th error bit for codeword 1
NANDC_BCHDE1_5	0x00e4	W	0x00000000	BCH decode result of 5th error bit for codeword 1
NANDC_BCHDE1_6	0x00e8	W	0x00000000	BCH decode result of 6th error bit for codeword 1
NANDC_BCHDE1_7	0x00ec	W	0x00000000	BCH decode result of 7th error bit for codeword 1
NANDC_BCHDE1_8	0x00f0	W	0x00000000	BCH decode result of 8th error bit for codeword 1
NANDC_BCHDE1_9	0x00f4	W	0x00000000	BCH decode result of 9th error bit for codeword 1
NANDC_BCHDE1_10	0x00f8	W	0x00000000	BCH decode result of 10th error bit for codeword 1
NANDC_BCHDE1_11	0x00fc	W	0x00000000	BCH decode result of 11th error bit for codeword 1
NANDC_BCHDE1_12	0x0100	W	0x00000000	BCH decode result of 12th error bit for codeword 1
NANDC_BCHDE1_13	0x0104	W	0x00000000	BCH decode result of 13th error bit for codeword 1
NANDC_BCHDE1_14	0x0108	W	0x00000000	BCH decode result of 14th error bit for codeword 1
NANDC_BCHDE1_15	0x010c	W	0x00000000	BCH decode result of 15th error bit for codeword 1
NANDC_BCHDE1_16	0x0110	W	0x00000000	BCH decode result of 16th error bit for codeword 1
NANDC_BCHDE1_17	0x0114	W	0x00000000	BCH decode result of 17th error bit for codeword 1
NANDC_BCHDE1_18	0x0118	W	0x00000000	BCH decode result of 18th error bit for codeword 1
NANDC_BCHDE1_19	0x011c	W	0x00000000	BCH decode result of 19th error bit for codeword 1
NANDC_BCHDE1_20	0x0120	W	0x00000000	BCH decode result of 20th error bit for codeword 1
NANDC_BCHDE1_21	0x0124	W	0x00000000	BCH decode result of 21th error bit for codeword 1
NANDC_BCHDE1_22	0x0128	W	0x00000000	BCH decode result of 22th error bit for codeword 1
NANDC_BCHDE1_23	0x012c	W	0x00000000	BCH decode result of 23th error bit for codeword 1
NANDC_DLL_CTL_REG0	0x0130	W	0x007f7f05	DLL Control Register 0

Name	Offset	Size	Reset Value	Description
NANDC_DLL_CTL_REG1	0x0134	W	0x00000022	DLL Control Register 1
NANDC_DLL_OBS_REG0	0x0138	W	0x00000200	DLL Status Register
NANDC_RANDMZ_CFG	0x0150	W	0x00000000	Randomizer Configure Register
NANDC_FMWAIT_SYN	0x0158	W	0x00000000	Flash Timing Control Register For Synchronous Timing
NANDC_MTRANS_STAT2	0x015c	W	0x00000000	Bus Transfer Status Register2
NANDC_NANDC_VER	0x0160	W	0x00000701	NandC Version Register
NANDC_LLP_CTL	0x0164	W	0x00000000	LLP Control Register
NANDC_LLP_STAT	0x0168	W	0x00000001	LLP Status Register
NANDC_INTEN	0x016c	W	0x00000000	NandC Interrupt Enable Register
NANDC_INTCLR	0x0170	W	0x00000000	NandC Interrupt Clear Register
NANDC_INTST	0x0174	W	0x00000000	NandC Interrupt Status Register
NANDC_SPARE0_0	0x0200	W	0xffffffff	System Information for codeword 0
NANDC_SPARE0_1	0x0204	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_2	0x0208	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_3	0x020c	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_4	0x0210	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_5	0x0214	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_6	0x0218	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_7	0x021c	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_8	0x0220	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_9	0x0224	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_10	0x0228	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_11	0x022c	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE1_0	0x0230	W	0xffffffff	System Information for codeword 1
NANDC_SPARE1_1	0x0234	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_2	0x0238	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_3	0x023c	W	0x00000000	Spare Data and BCH Encode Information for codeword 1

Name	Offset	Size	Reset Value	Description
NANDC_SPARE1_4	0x0240	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_5	0x0244	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_6	0x0248	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_7	0x024c	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_8	0x0250	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_9	0x0254	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_10	0x0258	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_11	0x025c	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE0_12	0x0260	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_13	0x0264	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_14	0x0268	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_15	0x026c	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_16	0x0270	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_17	0x0274	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_18	0x0278	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_19	0x027c	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_20	0x0280	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_21	0x0284	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_22	0x0288	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_23	0x028c	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_24	0x0290	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_25	0x0294	W	0x00000000	Spare Data and BCH Encode Information for codeword 0

Name	Offset	Size	Reset Value	Description
NANDC_SPARE0_26	0x0298	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE0_27	0x029c	W	0x00000000	Spare Data and BCH Encode Information for codeword 0
NANDC_SPARE1_12	0x02a0	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_13	0x02a4	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_14	0x02a8	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_15	0x02ac	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_16	0x02b0	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_17	0x02b4	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_18	0x02b8	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_19	0x02bc	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_20	0x02c0	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_21	0x02c4	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_22	0x02c8	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_23	0x02cc	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_24	0x02d0	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_25	0x02d4	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_26	0x02d8	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_SPARE1_27	0x02dc	W	0x00000000	Spare Data and BCH Encode Information for codeword 1
NANDC_BCHDE0_24	0x0400	W	0x00000000	BCH decode result of 24th error bit for codeword 0
NANDC_BCHDE0_25	0x0404	W	0x00000000	BCH decode result of 25th error bit for codeword 0
NANDC_BCHDE0_26	0x0408	W	0x00000000	BCH decode result of 26th error bit for codeword 0
NANDC_BCHDE0_27	0x040c	W	0x00000000	BCH decode result of 27th error bit for codeword 0

Name	Offset	Size	Reset Value	Description
NANDC_BCHDE0_28	0x0410	W	0x00000000	BCH decode result of 28th error bit for codeword 0
NANDC_BCHDE0_29	0x0414	W	0x00000000	BCH decode result of 29th error bit for codeword 0
NANDC_BCHDE0_30	0x0418	W	0x00000000	BCH decode result of 30th error bit for codeword 0
NANDC_BCHDE0_31	0x041c	W	0x00000000	BCH decode result of 31th error bit for codeword 0
NANDC_BCHDE0_32	0x0420	W	0x00000000	BCH decode result of 32th error bit for codeword 0
NANDC_BCHDE0_33	0x0424	W	0x00000000	BCH decode result of 33th error bit for codeword 0
NANDC_BCHDE0_34	0x0428	W	0x00000000	BCH decode result of 34th error bit for codeword 0
NANDC_BCHDE0_35	0x042c	W	0x00000000	BCH decode result of 35th error bit for codeword 0
NANDC_BCHDE0_36	0x0430	W	0x00000000	BCH decode result of 36th error bit for codeword 0
NANDC_BCHDE0_37	0x0434	W	0x00000000	BCH decode result of 37th error bit for codeword 0
NANDC_BCHDE0_38	0x0438	W	0x00000000	BCH decode result of 38th error bit for codeword 0
NANDC_BCHDE0_39	0x043c	W	0x00000000	BCH decode result of 39th error bit for codeword 0
NANDC_BCHDE0_40	0x0440	W	0x00000000	BCH decode result of 40th error bit for codeword 0
NANDC_BCHDE0_41	0x0444	W	0x00000000	BCH decode result of 41th error bit for codeword 0
NANDC_BCHDE0_42	0x0448	W	0x00000000	BCH decode result of 42th error bit for codeword 0
NANDC_BCHDE0_43	0x044c	W	0x00000000	BCH decode result of 43th error bit for codeword 0
NANDC_BCHDE0_44	0x0450	W	0x00000000	BCH decode result of 44th error bit for codeword 0
NANDC_BCHDE0_45	0x0454	W	0x00000000	BCH decode result of 45th error bit for codeword 0
NANDC_BCHDE0_46	0x0458	W	0x00000000	BCH decode result of 46th error bit for codeword 0
NANDC_BCHDE0_47	0x045c	W	0x00000000	BCH decode result of 47th error bit for codeword 0
NANDC_BCHDE0_48	0x0460	W	0x00000000	BCH decode result of 48th error bit for codeword 0
NANDC_BCHDE0_49	0x0464	W	0x00000000	BCH decode result of 49th error bit for codeword 0

Name	Offset	Size	Reset Value	Description
NANDC_BCHDE0_50	0x0468	W	0x00000000	BCH decode result of 50th error bit for codeword 0
NANDC_BCHDE0_51	0x046c	W	0x00000000	BCH decode result of 51th error bit for codeword 0
NANDC_BCHDE0_52	0x0470	W	0x00000000	BCH decode result of 52th error bit for codeword 0
NANDC_BCHDE0_53	0x0474	W	0x00000000	BCH decode result of 53th error bit for codeword 0
NANDC_BCHDE0_54	0x0478	W	0x00000000	BCH decode result of 54th error bit for codeword 0
NANDC_BCHDE0_55	0x047c	W	0x00000000	BCH decode result of 55th error bit for codeword 0
NANDC_BCHDE0_56	0x0480	W	0x00000000	BCH decode result of 56th error bit for codeword 0
NANDC_BCHDE0_57	0x0484	W	0x00000000	BCH decode result of 57th error bit for codeword 0
NANDC_BCHDE0_58	0x0488	W	0x00000000	BCH decode result of 58th error bit for codeword 0
NANDC_BCHDE0_59	0x048c	W	0x00000000	BCH decode result of 59th error bit for codeword 0
NANDC_BCHDE1_24	0x0490	W	0x00000000	BCH decode result of 24th error bit for codeword 1
NANDC_BCHDE1_25	0x0494	W	0x00000000	BCH decode result of 25th error bit for codeword 1
NANDC_BCHDE1_26	0x0498	W	0x00000000	BCH decode result of 26th error bit for codeword 1
NANDC_BCHDE1_27	0x049c	W	0x00000000	BCH decode result of 27th error bit for codeword 1
NANDC_BCHDE1_28	0x04a0	W	0x00000000	BCH decode result of 28th error bit for codeword 1
NANDC_BCHDE1_29	0x04a4	W	0x00000000	BCH decode result of 29th error bit for codeword 1
NANDC_BCHDE1_30	0x04a8	W	0x00000000	BCH decode result of 30th error bit for codeword 1
NANDC_BCHDE1_31	0x04ac	W	0x00000000	BCH decode result of 31th error bit for codeword 1
NANDC_BCHDE1_32	0x04b0	W	0x00000000	BCH decode result of 32th error bit for codeword 1
NANDC_BCHDE1_33	0x04b4	W	0x00000000	BCH decode result of 33th error bit for codeword 1
NANDC_BCHDE1_34	0x04b8	W	0x00000000	BCH decode result of 34th error bit for codeword 1
NANDC_BCHDE1_35	0x04bc	W	0x00000000	BCH decode result of 35th error bit for codeword 1

Name	Offset	Size	Reset Value	Description
NANDC_BCHDE1_36	0x04c0	W	0x00000000	BCH decode result of 3th error bit for codeword 1
NANDC_BCHDE1_37	0x04c4	W	0x00000000	BCH decode result of 37th error bit for codeword 1
NANDC_BCHDE1_38	0x04c8	W	0x00000000	BCH decode result of 38th error bit for codeword 1
NANDC_BCHDE1_39	0x04cc	W	0x00000000	BCH decode result of 39th error bit for codeword 1
NANDC_BCHDE1_40	0x04d0	W	0x00000000	BCH decode result of 40th error bit for codeword 1
NANDC_BCHDE1_41	0x04d4	W	0x00000000	BCH decode result of 41th error bit for codeword 1
NANDC_BCHDE1_42	0x04d8	W	0x00000000	BCH decode result of 42th error bit for codeword 1
NANDC_BCHDE1_43	0x04dc	W	0x00000000	BCH decode result of 43th error bit for codeword 1
NANDC_BCHDE1_44	0x04e0	W	0x00000000	BCH decode result of 44th error bit for codeword 1
NANDC_BCHDE1_45	0x04e4	W	0x00000000	BCH decode result of 45th error bit for codeword 1
NANDC_BCHDE1_46	0x04e8	W	0x00000000	BCH decode result of 46th error bit for codeword 1
NANDC_BCHDE1_47	0x04ec	W	0x00000000	BCH decode result of 47th error bit for codeword 1
NANDC_BCHDE1_48	0x04f0	W	0x00000000	BCH decode result of 48th error bit for codeword 1
NANDC_BCHDE1_49	0x04f4	W	0x00000000	BCH decode result of 49th error bit for codeword 1
NANDC_BCHDE1_50	0x04f8	W	0x00000000	BCH decode result of 50th error bit for codeword 1
NANDC_BCHDE1_51	0x04fc	W	0x00000000	BCH decode result of 51th error bit for codeword 1
NANDC_BCHDE1_52	0x0500	W	0x00000000	BCH decode result of 52th error bit for codeword 1
NANDC_BCHDE1_53	0x0504	W	0x00000000	BCH decode result of 53th error bit for codeword 1
NANDC_BCHDE1_54	0x0508	W	0x00000000	BCH decode result of 54th error bit for codeword 1
NANDC_BCHDE1_55	0x050c	W	0x00000000	BCH decode result of 55th error bit for codeword 1
NANDC_BCHDE1_56	0x0510	W	0x00000000	BCH decode result of 56th error bit for codeword 1
NANDC_BCHDE1_57	0x0514	W	0x00000000	BCH decode result of 57th error bit for codeword 1

Name	Offset	Size	Reset Value	Description
NANDC_BCHDE1_58	0x0518	W	0x00000000	BCH decode result of 58th error bit for codeword 1
NANDC_BCHDE1_59	0x051c	W	0x00000000	BCH decode result of 59th error bit for codeword 1
NANDC_BCHST8	0x0520	W	0x00000000	BCH Status Register For Codeword 16~17
NANDC_BCHST9	0x0524	W	0x00000000	BCH Status Register For Codeword 18~19
NANDC_BCHST10	0x0528	W	0x00000000	BCH Status Register For Codeword 20~21
NANDC_BCHST11	0x052c	W	0x00000000	BCH Status Register For Codeword 22~23
NANDC_BCHST12	0x0530	W	0x00000000	BCH Status Register For Codeword 24~25
NANDC_BCHST13	0x0534	W	0x00000000	BCH Status Register For Codeword 26~27
NANDC_BCHST14	0x0538	W	0x00000000	BCH Status Register For Codeword 28~29
NANDC_BCHST15	0x053c	W	0x00000000	BCH Status Register For Codeword 30~31
NANDC_RANDMZ_SEED13_0	0x0600	W	0x00000000	Seed 0 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_1	0x0604	W	0x00000000	Seed 1 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_2	0x0608	W	0x00000000	Seed 2 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_3	0x060c	W	0x00000000	Seed 3 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_4	0x0610	W	0x00000000	Seed 4 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_5	0x0614	W	0x00000000	Seed 5 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_6	0x0618	W	0x00000000	Seed 6 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_7	0x061c	W	0x00000000	Seed 7 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_8	0x0620	W	0x00000000	Seed 8 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_9	0x0624	W	0x00000000	Seed 9 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_10	0x0628	W	0x00000000	Seed 10 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_11	0x062c	W	0x00000000	Seed 11 for Toshiba 13 Power Polynomial Randomizer

Name	Offset	Size	Reset Value	Description
NANDC_RANDMZ_SEED13_12	0x0630	W	0x00000000	Seed 12 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_13	0x0634	W	0x00000000	Seed 13 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_14	0x0638	W	0x00000000	Seed 14 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED13_15	0x063c	W	0x00000000	Seed 15 for Toshiba 13 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_0	0x0640	W	0x00000000	Seed 0 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_1	0x0644	W	0x00000000	Seed 1 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_2	0x0648	W	0x00000000	Seed 2 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_3	0x064c	W	0x00000000	Seed 3 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_4	0x0650	W	0x00000000	Seed 4 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_5	0x0654	W	0x00000000	Seed 5 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_6	0x0658	W	0x00000000	Seed 6 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_7	0x065c	W	0x00000000	Seed 7 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_8	0x0660	W	0x00000000	Seed 8 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_9	0x0664	W	0x00000000	Seed 9 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_10	0x0668	W	0x00000000	Seed 10 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_11	0x066c	W	0x00000000	Seed 11 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_12	0x0670	W	0x00000000	Seed 12 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_13	0x0674	W	0x00000000	Seed 13 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_14	0x0678	W	0x00000000	Seed 14 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED17_15	0x067c	W	0x00000000	Seed 15 for Toshiba 17 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_0	0x0680	W	0x00000000	Seed 0 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_1	0x0684	W	0x00000000	Seed 1 for Toshiba 19 Power Polynomial Randomizer

Name	Offset	Size	Reset Value	Description
NANDC_RANDMZ_SEED19_2	0x0688	W	0x00000000	Seed 2 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_3	0x068c	W	0x00000000	Seed 3 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_4	0x0690	W	0x00000000	Seed 4 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_5	0x0694	W	0x00000000	Seed 5 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_6	0x0698	W	0x00000000	Seed 6 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_7	0x069c	W	0x00000000	Seed 7 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_8	0x06a0	W	0x00000000	Seed 8 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_9	0x06a4	W	0x00000000	Seed 9 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_10	0x06a8	W	0x00000000	Seed 10 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_11	0x06ac	W	0x00000000	Seed 11 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_12	0x06b0	W	0x00000000	Seed 12 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_13	0x06b4	W	0x00000000	Seed 13 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_14	0x06b8	W	0x00000000	Seed 14 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED19_15	0x06bc	W	0x00000000	Seed 15 for Toshiba 19 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_0	0x06c0	W	0x00000000	Seed 0 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_1	0x06c4	W	0x00000000	Seed 1 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_2	0x06c8	W	0x00000000	Seed 2 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_3	0x06cc	W	0x00000000	Seed 3 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_4	0x06d0	W	0x00000000	Seed 4 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_5	0x06d4	W	0x00000000	Seed 5 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_6	0x06d8	W	0x00000000	Seed 6 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_7	0x06dc	W	0x00000000	Seed 7 for Toshiba 23 Power Polynomial Randomizer

Name	Offset	Size	Reset Value	Description
NANDC_RANDMZ_SEED23_8	0x06e0	W	0x00000000	Seed 8 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_9	0x06e4	W	0x00000000	Seed 9 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_10	0x06e8	W	0x00000000	Seed 10 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_11	0x06ec	W	0x00000000	Seed 11 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_12	0x06f0	W	0x00000000	Seed 12 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_13	0x06f4	W	0x00000000	Seed 13 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_14	0x06f8	W	0x00000000	Seed 14 for Toshiba 23 Power Polynomial Randomizer
NANDC_RANDMZ_SEED23_15	0x06fc	W	0x00000000	Seed 15 for Toshiba 23 Power Polynomial Randomizer
NANDC_FLASH0_DATA	0x0800	W	0x00000000	data to be write into or read from flash0
NANDC_FLASH0_ADDR	0x0804	W	0x00000000	flash0 address
NANDC_FLASH0_CMD	0x0808	W	0x00000000	command send to flash0
NANDC_FLASH0_DATA_SYN	0x080c	W	0x00000000	data to be write into or read from flash0 for Synchronous flash
NANDC_FLASH1_DATA	0x0900	W	0x00000000	data to be write into or read from flash1
NANDC_FLASH1_ADDR	0x0904	W	0x00000000	flash1 address
NANDC_FLASH1_CMD	0x0908	W	0x00000000	command send to flash1
NANDC_FLASH1_DATA_SYN	0x090c	W	0x00000000	data to be write into or read from flash1 for Synchronous flash
NANDC_FLASH2_DATA	0x0a00	W	0x00000000	data to be write into or read from flash2
NANDC_FLASH2_ADDR	0x0a04	W	0x00000000	flash2 address
NANDC_FLASH2_CMD	0x0a08	W	0x00000000	command send to flash2
NANDC_FLASH2_DATA_SYN	0x0a0c	W	0x00000000	data to be write into or read from flash2 for Synchronous flash
NANDC_FLASH3_DATA	0x0b00	W	0x00000000	data to be write into or read from flash3
NANDC_FLASH3_ADDR	0x0b04	W	0x00000000	flash3 address
NANDC_FLASH3_CMD	0x0b08	W	0x00000000	command send to flash3
NANDC_FLASH3_DATA_SYN	0x0b0c	W	0x00000000	data to be write into or read from flash3 for Synchronous flash

Name	Offset	Size	Reset Value	Description
NANDC_FLASH4_DATA	0x0c00	W	0x00000000	data to be write into or read from flash4
NANDC_FLASH4_ADDR	0x0c04	W	0x00000000	flash4 address
NANDC_FLASH4_CMD	0x0c08	W	0x00000000	command send to flash4
NANDC_FLASH4_DATA_SYN	0x0c0c	W	0x00000000	data to be write into or read from flash4 for Synchronous flash
NANDC_FLASH5_DATA	0x0d00	W	0x00000000	data to be write into or read from flash5
NANDC_FLASH5_ADDR	0x0d04	W	0x00000000	flash5 address
NANDC_FLASH5_CMD	0x0d08	W	0x00000000	command send to flash5
NANDC_FLASH5_DATA_SYN	0x0d0c	W	0x00000000	data to be write into or read from flash5 for Synchronous flash
NANDC_FLASH6_DATA	0x0e00	W	0x00000000	data to be write into or read from flash6
NANDC_FLASH6_ADDR	0x0e04	W	0x00000000	flash6 address
NANDC_FLASH6_CMD	0x0e08	W	0x00000000	command send to flash6
NANDC_FLASH6_DATA_SYN	0x0e0c	W	0x00000000	data to be write into or read from flash6 for Synchronous flash
NANDC_FLASH7_DATA	0x0f00	W	0x00000000	data to be write into or read from flash7
NANDC_FLASH7_ADDR	0x0f04	W	0x00000000	flash7 address
NANDC_FLASH7_CMD	0x0f08	W	0x00000000	command send to flash7
NANDC_FLASH7_DATA_SYN	0x0f0c	W	0x00000000	data to be write into or read from flash7 for Synchronous flash

Notes:Size: **B**- Byte (8 bits) access, **HW**- Half WORD (16 bits) access, **W**-WORD (32 bits) access

### 10.4.3 Detail Register Description

#### NANDC\_FMCTL

Address: Operational Base + offset (0x0000)

Flash Interface Control Register

Bit	Attr	Reset Value	Description
31:27	RO	0x0	reserved
26:24	RW	0x0	read_delay The number of delay cycle to capture the flash data after posedge of rdn.
23:18	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
17	RO	0x0	flash_abort_stat Function1: flash_abort_stat, RO. Function2: flash_abort_clear, RW, auto clear. flash_abort_stat is set to 1 when flash abort if flash_abort_en=1, set to 0 when flash_abort_clear=1.
16	RW	0x0	flash_abort_en Flash abort protect enable signal, 1 active. 0: Flash abort protect disable. 1: Flash abort protect enable. Notes: 1. when in dma mode, if the time from last read operation start to the last read valid exceeds 1024 cycles(nclk), flash_abort_stat is set to high. 2. when in bypass mode, if the time from current read operation start to the read valid exceed 1024 cycles(nclk), flash_abort_stat is set to high. 3. when in llp bypass read/read match mode, when the operation is long than 1024 cycles(nclk), flash_abort_stat is set to high.
15	RW	0x0	syn_mode Toggle enable signal, 1 active. 0: ONFI synchronous flash. 1: Toggle synchronous flash.
14	RW	0x0	syn_clken Synchronous flash clock enable signal, 1 active. Only available in Synchronous Mode. 0: flash clock is disabled. 1: flash clock is enabled.
13	RW	0x0	tm Timing mode indication. 0: Asynchronous Mode. 1: Synchronous Mode (Toggle or ONFI Synchronous).
12	RW	0x0	dwidth Flash data bus width indication. 0: 8bits, active in both Asynchronous Mode flash and Synchronous Mode flash. 1: 16bits, active only in Asynchronous Mode flash.
11	RO	0x0	reserved
10	RW	0x0	fifo_empty fifo empty signal. 1'b0: fifo is not empty; 1'b1: fifo is empty;

Bit	Attr	Reset Value	Description
9	RO	0x1	frdy Flash ready/busy indicate signal. 0: flash is busy. 1: flash is ready. This bit is the sample of the pin of R/Bn.
8	RW	0x0	wp Flash write protect. 0: flash program/erase disabled. 1: flash program/erase enabled. This bit is output to the pin of WPn.
7	RW	0x0	fcs7 Flash memory chip 7 select control. 1: hold flash memory chip select activity. 0: flash memory chip select activity free.
6	RW	0x0	fcs6 Flash memory chip 6 select control. 1: hold flash memory chip select activity. 0: flash memory chip select activity free.
5	RW	0x0	fcs5 Flash memory chip 5 select control. 1: hold flash memory chip select activity. 0: flash memory chip select activity free.
4	RW	0x0	fcs4 Flash memory chip 4 select control. 1: hold flash memory chip select activity. 0: flash memory chip select activity free.
3	RW	0x0	fcs3 Flash memory chip 3 select control. 1: hold flash memory chip select activity. 0: flash memory chip select activity free.
2	RW	0x0	fcs2 Flash memory chip 2 select control. 1: hold flash memory chip select activity. 0: flash memory chip select activity free.
1	RW	0x0	fcs1 Flash memory chip 1 select control. 1: hold flash memory chip select activity. 0: flash memory chip select activity free.
0	RW	0x0	fcs0 Flash memory chip 0 select control. 1: hold flash memory chip select activity. 0: flash memory chip select activity free.

**NANDC\_FMWAIT\_ASYN**

Address: Operational Base + offset (0x0004)

## Flash Timing Control Register For Asynchronous Timing

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	RO	0x0	reserved
30	RW	0x0	fmw_dly_en fmw_dly enable signal,1 active.
29:24	RW	0x3f	fmw_dly The number of delay cycle between two codeword transmission.
23	RO	0x0	reserved
22:18	RW	0x0f	wait_frdy_dly The number of delay cycle to accept the flash ready signal.
17:12	RW	0x3f	csrw When in Asynchronous mode or Toggle address/command mode, this field specifies the number of processor clock cycles from the falling edge of CSn to the falling edge of RDn or WRn. The min value of csrw is 0.
11	RW	0x0	hard_rdy Hardware handshaking controller bit. When asserted, an external device asserts signal "RDY" to extend a wait-state access and the rest bits in this register will be ignored.
10:5	RW	0x3f	rwpw When in Asynchronous mode or Toggle address/command mode, this field specifies the width of RDn or WRn in processor clock cycles, $0x0 \leq rwpw \leq 0x3f$ .
4:0	RW	0x1f	rwcs When in Asynchronous mode or Toggle address/command mode, this field specifies the number of processor clock cycles from the rising edge of RDn or WRn to the rising edge of CSn, $0x0 \leq rwcs \leq 0x1f$ .

**NANDC\_FLCTL**

Address: Operational Base + offset (0x0008)

Internal Transfer Control Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	RO	0x0	reserved
30	RW	0x0	bypass_fifo_mode The enable signal for bypass with fifo mode. 1'b0: disable fifo mode 1'b1: enable fifo mode
29	RW	0x0	async_tog_mix Nandc async mode and tog mode compatible control 0: async write data can't be read by tog read 1: async write data can be read by tog read
28	RW	0x0	low_power Nandc low power control 0: normal mode 1: low power mode

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
27:22	RW	0x00	<p>page_num Transmission codeword number in internal DMA mode when bus-mode is master-mode 1~32: 1~32 codeword. default: not support.</p> <p>Notes:</p> <ul style="list-style-type: none"> <li>a. Only active in internal DMA mode</li> <li>b. Only active when bus-mode is master-mode</li> </ul>
21	RW	0x0	<p>page_size Transmission codeword size in internal DMA mode 0: 1024bytes/codeword 1: 512bytes/codeword</p>
20	RO	0x1	<p>tr_rdy Internal DMA transmission ready indication. 0: internal DMA transmission is busy 1: internal DMA transmission is ready</p> <p>When reading flash, tr_rdy should not be set to 1 until all data transmission and correct finished.</p> <p>When programming flash, tr_rdy should not be set to 1 until all data transmission finished.</p> <p>Notes: Only active in internal DMA mode.</p>
19	RO	0x0	reserved
18:12	RW	0x00	<p>spare_size Spare byte number when lba_en=1. 0&lt;= spare_size&lt;=109. When spare_size&gt;=109, it is treated as 0.</p> <p>Notes: The spare_size must be even number when flash is ONFI Synchronous Flash or Asynchronous Flash with 16bits data width.</p>
11	RW	0x0	<p>lba_en LBA mode indication, 1 active. 0: NO-LBA mode, NandC should transfer both page data and spare data in every codeword, and the page size is 1024 bytes or 512 bytes determined by BCHCTL[16](bchpage), spare size is 32/46/74 bytes or 109 bytes determined by BCHCTL[4] and BCHCTL[18]. 1: LBA mode, NandC should transfer both page data and spare data in every codeword, and the page size is 1024 bytes or 512 bytes determined by FLCTL[21](page_size), spare size is determined by FLCTL[17:12](spare_size).</p> <p>Notes:</p> <ul style="list-style-type: none"> <li>a. When lba_en is active, BCH CODEC should be disabled, spare_size and page_size are configurable.</li> <li>b. When lba_en is active, cor_able is inactive.</li> </ul>

Bit	Attr	Reset Value	Description
10	RW	0x0	<p>cor_able Auto correct enable indication, 1 active. 0: auto correct disable 1: auto correct enable Notes: a. Only active in internal DMA mode. b. lba_en is prior to cor_able. When lba_en=1, cor_able is ignored.</p>
9:8	RO	0x0	reserved
7	RW	0x0	<p>flash_st_mod Mode for NandC to start internal data transmission in internal DMA mode. 0: busy mode: hardware should not start internal data transmission until flash is ready even flash_st is asserted. 1: ready mode: hardware should start internal data transmission directly when flash_st is asserted. Notes: Only active in internal DMA mode.</p>
6:5	RW	0x0	<p>tr_count Transmission codeword number in internal DMA mode when bus-mode is slave-mode. 00: 0 codeword need transferred 01: 1 codeword need transferred 10: 2 codeword need transferred 11: not supported Notes: a. Only active in internal DMA mode. b. Only active when bus-mode is slave-mode.</p>
4	RW	0x0	<p>st_addr Start buffer address. 0: start transfer from sram0 1: start transfer from sram1 Notes: Only active in internal DMA mode.</p>
3	RW	0x0	<p>bypass NandC internal DMA bypass indication. 0: bypass the internal DMA, data are transferred to/from flash by direct path. 1: internal DMA active, data are transferred to/from flash by internal DMA.</p>

Bit	Attr	Reset Value	Description
2	R/W SC	0x0	<p>flash_st</p> <p>Start signal for NandC to transfer data between flash and internal buffer in internal DMA mode. When asserted, it will auto cleared.</p> <p>0: not start transmission 1: start transmission</p> <p>Notes: Only active in internal DMA mode</p>
1	RW	0x0	<p>flash_rdn</p> <p>Indicate data flow direction.</p> <p>0: NandC read data from flash. 1: NandC write data to flash</p>
0	R/W SC	0x0	<p>flash_RST</p> <p>NandC software reset indication. When asserted, it will auto cleared.</p> <p>0: not software reset 1: software reset</p> <p>Notes: flash_RST is prior to flash_st</p>

**NANDC\_BCHCTL**

Address: Operational Base + offset (0x000c)

BCH Control Register

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28	RW	0x0	<p>bch_toddr</p> <p>enable signal for storing bch decode status into ddr.</p> <p>1'b0: disable; 1'b1: enable;</p>
27	RO	0x0	reserved
26:19	RW	0x00	<p>bchthres</p> <p>BCH error number threshold</p>
18	RW	0x0	<p>bchmode1</p> <p>High bit of BCH mode selection for 40bitBCH or 60bitBCH.</p> <p>BchMode=bchmode1, bchmode0:</p> <p>00: 16bitBCH 01: 24bitBCH 10: 40bitBCH 11: 60bitBCH</p>
17	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
16	RW	0x0	<p>bchpage The data size indication when BCH is active. 0: 1024 bytes, all the 1024 bytes data in codeword are valid data to be transferred. 1: 512 bytes, higher 512bytes are valid, and lower 512bytes are invalid and stuffed with 0xff.</p> <p>Notes:</p> <ul style="list-style-type: none"> <li>a. Only active when data transferred in internal DMA mode.</li> <li>b. Only active for asynchronous flash.</li> </ul>
15:8	RW	0x00	<p>addr BCH active range selection. BCH should be active when access in range address.</p>
7:5	RW	0x0	<p>region BCH active region selection indication. 000: Flash memory 0 region (flash 0) 001: Flash memory 1 region (flash 1) 010: Flash memory 2 region (flash 2) 011: Flash memory 3 region (flash 3) 100: Flash memory 4 region (flash 4) 101: Flash memory 5 region (flash 5) 110: Flash memory 6 region (flash 6) 111: Flash memory 7 region (flash 7)</p>
4	RW	0x0	<p>bchmode0 BCH mode selection indication. BCH mode is determined by both bchmode0 and bchmode1, detailed information is showed in BCHCTL[18].</p>
3	RW	0x1	<p>bchepd BCH encoder/decoder power down indication. 0: BCH encoder/decoder working. 1: BCH encoder/decoder not working.</p>
2	RW	0x0	<p>mode_addrare BCH address care mode selection indication. 0: address care. 1: address not care.</p> <p>Notes: This bit is just active for data transmission in bypass mode, but not for command and address transmission.</p>
1	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
0	R/W SC	0x0	<p>bchrst BCH software reset indication, When asserted, it will auto cleared. 0: not software reset 1: software reset Notes: a. BCH Decoder should be software reset before decode begin. b. bch software reset should be used with NandC software reset at the same time.</p>

**NANDC\_MTRANS\_CFG**

Address: Operational Base + offset (0x0010)

Bus Transfer Configuration Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15	R/W SC	0x0	<p>ahb_rst ahb master interface software reset, auto cleared</p>
14	RW	0x0	<p>fl_pwd Flash power down indication, 1 active. 0: Flash power on, data transferred through master interface is data that to be written into or read from flash. 1: Flash power down, data transferred through master interface is not data that to be written into or read from flash. NandC is just used as DMA for external memory and internal memory.</p>
13:9	RW	0x00	<p>incr_num AHB Master incr num indication. incr_num=1~16. When burst=001, software should configure incr_num. Notes: Only active for master-mode.</p>
8:6	RW	0x7	<p>burst AHB Master burst type indication: 000 : Single transfer 011 : 4-beat burst 101 : 8-beat Burst 111 : 16-beat burst default : not supported Notes: Only active for master-mode.</p>

Bit	Attr	Reset Value	Description
5:3	RW	0x2	<p>hsize AHB Master data size indication: 000 : 8 bits 001 : 16 bits 010 : 32 bits default : not supported</p> <p>Notes: Only active for master-mode.</p>
2	RW	0x0	<p>bus_mode Bus interface selection. 0: Slave interface, flash data is transferred through slave interface 1: Master interface, flash data is transferred through master interface</p>
1	RW	0x0	<p>ahb_wr Data transfer direction through master interface. 0: read direction(external memory -&gt;internal memory) 1: write direction (internal memory-&gt;external memory)</p> <p>Notes: a. Only active for master-mode. b. When read flash(flash_rdn=0), ahb_wr=1; when program flash(flash_rdn=1), ahb_wr=0.</p>
0	R/W SC	0x0	<p>ahb_wr_st Start indication for loading data from external memory to internal memory or storing data from internal memory to external memory through master. When asserted, it will auto cleared.only active when fl_pwd is 1</p> <p>Notes: a. Only active for master-mode and fl_pwd=1. b. When fl_pwd=0, flash is active, NandC start to transfer data through master interface if flash_st=1 c. When fl_pwd=1, flash is not active, NandC start to transfer data through master interface if ahb_wr_st=1</p>

**NANDC\_MTRANS\_SADDR0**

Address: Operational Base + offset (0x0014)

Start Address Register For Page Data Transmission

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	<p>saddr0 Start address for page data transmission.</p> <p>Notes: a. Only active for master-mode. b. Should be aligned with hsize in MTRANS_CFG[5:3].</p>

**NANDC\_MTRANS\_SADDR1**

Address: Operational Base + offset (0x0018)

## Start Address Register For Spare Data Transmission

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	saddr1 Start address for spare data. Notes: a. Only active for master-mode. b. Should be aligned with hsize in MTRANS_CFG[5:3].

**NANDC\_MTRANS\_STAT**

Address: Operational Base + offset (0x001c)

Bus Transfer Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:22	RO	0x0	reserved
21:16	RO	0x00	mtrans_cnt finished counter for codeword transmission through Master interface Notes: Only active for master-mode.
15:0	RO	0x0000	bus_err Bus error indication for codeword0~15. [0] : bus error for codeword 0 ..... [15] : bus error for codeword 15 Notes: Only active for master-mode.

**NANDC\_BCHST0**

Address: Operational Base + offset (0x0020)

BCH Status Register For Codeword 0~1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	RO	0x0	reserved
30	RO	0x0	err_hnum1_h1 Highest bit of err_hnum1
29	RO	0x0	err_tnum1_h1 Highest bit of err_tnum1
28	RO	0x0	err_hnum0_h1 Highest bit of err_hnum0
27	RO	0x0	err_tnum0_h1 Highest bit of err_tnum0
26	RO	0x1	bchrdy Ready indication for bch encoder/decoder, 1 active. 0: bch encoder/decoder is busy 1: bch encoder/decoder is ready
25:21	RO	0x00	err_hnum1_l5 Lower 5 bits of number of error bits found in first 512bytes of 1st backup codeword

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
20:16	RO	0x00	err_tnum1_15 Lower 5 bits of number of error bits found in 1st backup codeword
15	RO	0x0	fail1 Indication for the 1st backup codeword decoded failed or not. 0: decode successfully 1: decode fail
14	RO	0x0	done1 Indication for finishing decoding the 1st backup codeword 0: not finished 1: finished
13	RO	0x0	errf1 Indication for error found in 1st backup codeword. 0: no error 1: error found
12:8	RO	0x00	err_hnum0_15 Lower 5 bits of number of error bits found in first 512bytes of current backup codeword
7:3	RO	0x00	err_tnum0_15 Lower 5 bits of number of error bits found in current backup codeword
2	RO	0x0	fail0 Indication for current backup codeword decode failed or not 0: decode successfully 1: decode fail
1	RO	0x0	done0 Indication for finishing decoding the current backup codeword. 0: not finished 1: finished
0	RO	0x0	errf0 Indication for error found in current backup codeword. 0: no error 1: error found

**NANDC\_BCHST1**

Address: Operational Base + offset (0x0024)

BCH Status Register For Codeword 2~3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	RO	0x0	reserved
30	RO	0x0	err_hnum3_h1 Highest bit of err_hnum3
29	RO	0x0	err_tnum3_h1 Highest bit of err_tnum3
28	RO	0x0	err_hnum2_h1 Highest bit of err_hnum2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
27	RO	0x0	err_tnum2_h1 Highest bit of err_tnum2
26	RO	0x0	reserved
25:21	RO	0x00	err_hnum3_l5 Lower 5 bits of number of error bits found in first 512bytes of 3th backup codeword
20:16	RO	0x00	err_tnum3_l5 Lower 5 bits of number of error bits found in 3th backup codeword
15	RO	0x0	fail3 Indication for the 3th backup codeword decoded failed or not. 0: decode successfully 1: decode fail
14	RO	0x0	done3 Indication for finishing decoding the 3th backup codeword 0: not finished 1: finished
13	RO	0x0	errf3 Indication for error found in 3th backup codeword. 0: no error 1: error found
12:8	RO	0x00	err_hnum2_l5 Lower 5 bits of number of error bits found in first 512bytes of 2th backup codeword
7:3	RO	0x00	err_tnum2_l5 Lower 5 bits of number of error bits found in 2th backup codeword
2	RO	0x0	fail2 Indication for 2th backup codeword decode failed or not 0: decode successfully 1: decode fail
1	RO	0x0	done2 Indication for finishing decoding the 2th backup codeword. 0: not finished 1: finished
0	RO	0x0	errf2 Indication for error found in 2th backup codeword. 0: no error 1: error found

**NANDC\_BCHST2**

Address: Operational Base + offset (0x0028)

BCH Status Register For Codeword 4~5

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
------------	-------------	--------------------	--------------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd4_cwd5 BCHST information for 4th and 5th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST3**

Address: Operational Base + offset (0x002c)

BCH Status Register For Codeword 6~7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd6_cwd7 BCHST information for 6th and 7th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST4**

Address: Operational Base + offset (0x0030)

BCH Status Register For Codeword 8~9

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd8_cwd9 BCHST information for 8th and 9th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST5**

Address: Operational Base + offset (0x0034)

BCH Status Register For Codeword 10~11

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd10_cwd11 BCHST information for 10th and 11th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST6**

Address: Operational Base + offset (0x0038)

BCH Status Register For Codeword 12~13

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd12_cwd13 BCHST information for 12th and 13th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST7**

Address: Operational Base + offset (0x003c)

BCH Status Register For Codeword 14~15

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd14_cwd15 BCHST information for 14th and 15th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHLOC0**

Address: Operational Base + offset (0x0040)

BCH Error Bit Location Number Register For Codeword 0~5

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:30	RO	0x0	reserved
29:25	RO	0x00	err_loc5_I5 Lower 5 bits of number of 8bit error location in 5th backup codeword
24:20	RO	0x00	err_loc4_I5 Lower 5 bits of number of 8bit error location in 4th backup codeword
19:15	RO	0x00	err_loc3_I5 Lower 5 bits of number of 8bit error location in 3rd backup codeword
14:10	RO	0x00	err_loc2_I5 Lower 5 bits of number of 8bit error location in 2nd backup codeword
9:5	RO	0x00	err_loc1_I5 Lower 5 bits of number of 8bit error location in 1st backup codeword
4:0	RO	0x00	err_loc0_I5 Lower 5 bits of number of 8bit error location in current backup codeword

**NANDC\_BCHLOC1**

Address: Operational Base + offset (0x0044)

BCH Error Bit Location Number Register For Codeword 6~11

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:30	RO	0x0	reserved
29:25	RO	0x00	err_loc11_I5 Lower 5 bits of number of 8bit error location in 11th backup codeword
24:20	RO	0x00	err_loc10_I5 Lower 5 bits of number of 8bit error location in 10th backup codeword
19:15	RO	0x00	err_loc9_I5 Lower 5 bits of number of 8bit error location in 9th backup codeword

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
14:10	RO	0x00	err_loc8_I5 Lower 5 bits of number of 8bit error location in 8th backup codeword
9:5	RO	0x00	err_loc7_I5 Lower 5 bits of number of 8bit error location in 7th backup codeword
4:0	RO	0x00	err_loc6_I5 Lower 5 bits of number of 8bit error location in 6th backup codeword

**NANDC\_BCHLOC2**

Address: Operational Base + offset (0x0048)

BCH Error Bit Location Number Register For Codeword 12~17

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:30	RO	0x0	reserved
29:25	RO	0x00	err_loc17_I5 Lower 5 bits of number of 8bit error location in 17th backup codeword
24:20	RO	0x00	err_loc16_I5 Lower 5 bits of number of 8bit error location in 16th backup codeword
19:15	RO	0x00	err_loc15_I5 Lower 5 bits of number of 8bit error location in 15th backup codeword
14:10	RO	0x00	err_loc14_I5 Lower 5 bits of number of 8bit error location in 14th backup codeword
9:5	RO	0x00	err_loc13_I5 Lower 5 bits of number of 8bit error location in 13th backup codeword
4:0	RO	0x00	err_loc12_I5 Lower 5 bits of number of 8bit error location in 12th backup codeword

**NANDC\_BCHLOC3**

Address: Operational Base + offset (0x004c)

BCH Error Bit Location Number Register For Codeword 24~29

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:30	RO	0x0	reserved
29:25	RO	0x00	err_loc23_I5 Lower 5 bits of number of 8bit error location in 23th backup codeword
24:20	RO	0x00	err_loc22_I5 Lower 5 bits of number of 8bit error location in 22th backup codeword

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
19:15	RO	0x00	err_loc21_15 Lower 5 bits of number of 8bit error location in 21th backup codeword
14:10	RO	0x00	err_loc20_15 Lower 5 bits of number of 8bit error location in 20th backup codeword
9:5	RO	0x00	err_loc19_15 Lower 5 bits of number of 8bit error location in 19th backup codeword
4:0	RO	0x00	err_loc18_15 Lower 5 bits of number of 8bit error location in 18th backup codeword

**NANDC\_BCHLOC4**

Address: Operational Base + offset (0x0050)

BCH Error Bit Location Number Register For Codeword 24~29

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:30	RO	0x0	reserved
29:25	RO	0x00	err_loc29_15 Lower 5 bits of number of 8bit error location in 29th backup codeword
24:20	RO	0x00	err_loc28_15 Lower 5 bits of number of 8bit error location in 28th backup codeword
19:15	RO	0x00	err_loc27_15 Lower 5 bits of number of 8bit error location in 27th backup codeword
14:10	RO	0x00	err_loc26_15 Lower 5 bits of number of 8bit error location in 26th backup codeword
9:5	RO	0x00	err_loc25_15 Lower 5 bits of number of 8bit error location in 25th backup codeword
4:0	RO	0x00	err_loc24_15 Lower 5 bits of number of 8bit error location in 24th backup codeword

**NANDC\_BCHLOC5**

Address: Operational Base + offset (0x0054)

BCH Error Bit Location Number Register For Codeword 30~31

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:10	RO	0x0	reserved
9:5	RO	0x00	err_loc31_15 Lower 5 bits of number of 8bit error location in 31th backup codeword

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
4:0	RO	0x00	err_loc30_l5 Lower 5 bits of number of 8bit error location in 30th backup codeword

**NANDC\_BCHLOC6**

Address: Operational Base + offset (0x0058)

Highest Bit For BCH Error Bit Location Number Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	RO	0x0	err_loc31_h1 High bit for numbers of 8bit error location in 31th codeword
30	RO	0x0	err_loc30_h1 High bit for numbers of 8bit error location in 30th codeword
29	RO	0x0	err_loc29_h1 High bit for numbers of 8bit error location in 29th codeword
28	RO	0x0	err_loc28_h1 High bit for numbers of 8bit error location in 28th codeword
27	RO	0x0	err_loc27_h1 High bit for numbers of 8bit error location in 27th codeword
26	RO	0x0	err_loc26_h1 High bit for numbers of 8bit error location in 26th codeword
25	RO	0x0	err_loc25_h1 High bit for numbers of 8bit error location in 25th codeword
24	RO	0x0	err_loc24_h1 High bit for numbers of 8bit error location in 24th codeword
23	RO	0x0	err_loc23_h1 High bit for numbers of 8bit error location in 23th codeword
22	RO	0x0	err_loc22_h1 High bit for numbers of 8bit error location in 22th codeword
21	RO	0x0	err_loc21_h1 High bit for numbers of 8bit error location in 21th codeword
20	RO	0x0	err_loc20_h1 High bit for numbers of 8bit error location in 20th codeword
19	RO	0x0	err_loc19_h1 High bit for numbers of 8bit error location in 19th codeword
18	RO	0x0	err_loc18_h1 High bit for numbers of 8bit error location in 18th codeword
17	RO	0x0	err_loc17_h1 High bit for numbers of 8bit error location in 17th codeword
16	RO	0x0	err_loc16_h1 High bit for numbers of 8bit error location in 16th codeword
15	RO	0x0	err_loc15_h1 High bit for numbers of 8bit error location in 15th codeword
14	RO	0x0	err_loc14_h1 High bit for numbers of 8bit error location in 14th codeword

Bit	Attr	Reset Value	Description
13	RO	0x0	err_loc13_h1 High bit for numbers of 8bit error location in 13th codeword
12	RO	0x0	err_loc12_h1 High bit for numbers of 8bit error location in 12th codeword
11	RO	0x0	err_loc11_h1 High bit for numbers of 8bit error location in 11th codeword
10	RO	0x0	err_loc10_h1 High bit for numbers of 8bit error location in 10th codeword
9	RO	0x0	err_loc9_h1 High bit for numbers of 8bit error location in 9th codeword
8	RO	0x0	err_loc8_h1 High bit for numbers of 8bit error location in 8th codeword
7	RO	0x0	err_loc7_h1 High bit for numbers of 8bit error location in 7th codeword
6	RO	0x0	err_loc6_h1 High bit for numbers of 8bit error location in 6th codeword
5	RO	0x0	err_loc5_h1 High bit for numbers of 8bit error location in 5th codeword
4	RO	0x0	err_loc4_h1 High bit for numbers of 8bit error location in 4th codeword
3	RO	0x0	err_loc3_h1 High bit for numbers of 8bit error location in 3th codeword
2	RO	0x0	err_loc2_h1 High bit for numbers of 8bit error location in 2th codeword
1	RO	0x0	err_loc1_h1 High bit for numbers of 8bit error location in 1th codeword
0	RO	0x0	err_loc0_h1 High bit for numbers of 8bit error location in 0th codeword

**NANDC\_BCHDE0\_0**

Address: Operational Base + offset (0x0070)

BCH decode result of 0th error bit for codeword 0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:8	RO	0x000	offset The offset byte address of the error bit. The value is 11bit, which is the byte offset address in the codeword. The address can be divided into different part for different use, showed as follows. 0 ~1023: page data 1024~1027: system information 1028~1055: bch information for 16bitBCH 1028~1069: bch information for 24bitBCH 1028~1097: bch information for 40bitBCH 1028~1132: bch information for 60bitBCH

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
7:0	RO	0x00	err_val The error value of corresponding error byte

**NANDC\_BCHDE0\_1**

Address: Operational Base + offset (0x0074)  
BCH decode result of 1th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_1 decode result of 1th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_2**

Address: Operational Base + offset (0x0078)  
BCH decode result of 2th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_2 decode result of 2th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_3**

Address: Operational Base + offset (0x007c)  
BCH decode result of 3th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_3 decode result of 3th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_4**

Address: Operational Base + offset (0x0080)  
BCH decode result of 4th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_4 decode result of 4th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_5**

Address: Operational Base + offset (0x0084)  
 BCH decode result of 5th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_5 decode result of 5th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_6**

Address: Operational Base + offset (0x0088)  
 BCH decode result of 6th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_6 decode result of 6th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_7**

Address: Operational Base + offset (0x008c)  
 BCH decode result of 7th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_7 decode result of 7th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_8**

Address: Operational Base + offset (0x0090)  
 BCH decode result of 8th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_8 decode result of 8th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_9**

Address: Operational Base + offset (0x0094)  
 BCH decode result of 9th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
18:0	RO	0x00000	bchde0_9 decode result of 9th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_10**

Address: Operational Base + offset (0x0098)  
BCH decode result of 10th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_10 decode result of 10th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_11**

Address: Operational Base + offset (0x009c)  
BCH decode result of 11th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_11 decode result of 11th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_12**

Address: Operational Base + offset (0x00a0)  
BCH decode result of 12th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_12 decode result of 12th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_13**

Address: Operational Base + offset (0x00a4)  
BCH decode result of 13th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_13 decode result of 13th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_14**

Address: Operational Base + offset (0x00a8)

BCH decode result of 14th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_14 decode result of 14th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_15**

Address: Operational Base + offset (0x00ac)

BCH decode result of 15th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_15 decode result of 15th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_16**

Address: Operational Base + offset (0x00b0)

BCH decode result of 16th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_16 decode result of 16th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_17**

Address: Operational Base + offset (0x00b4)

BCH decode result of 17th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_17 decode result of 17th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_18**

Address: Operational Base + offset (0x00b8)

BCH decode result of 18th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

Bit	Attr	Reset Value	Description
18:0	RO	0x00000	bchde0_18 decode result of 18th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_19**

Address: Operational Base + offset (0x00bc)  
BCH decode result of 19th error bit for codeword 0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_19 decode result of 1th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_20**

Address: Operational Base + offset (0x00c0)  
BCH decode result of 20th error bit for codeword 0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_20 decode result of 20th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_21**

Address: Operational Base + offset (0x00c4)  
BCH decode result of 21th error bit for codeword 0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_21 decode result of 21th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_22**

Address: Operational Base + offset (0x00c8)  
BCH decode result of 22th error bit for codeword 0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_22 decode result of 22th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_23**

Address: Operational Base + offset (0x00cc)

BCH decode result of 23th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_23 decode result of 23th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE1\_0**

Address: Operational Base + offset (0x00d0)

BCH decode result of 0th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:8	RO	0x000	offset The offset byte address of the error bit. The value is 11bit, which is the byte offset address in the codeword. The address can be divided into different part for different use, showed as follows. 0 ~1023: page data 1024~1027: system information 1028~1055: bch information for 16bitBCH 1028~1069: bch information for 24bitBCH 1028~1097: bch information for 40bitBCH 1028~1132: bch information for 60bitBCH
7:0	RO	0x00	err_val The error value of corresponding error byte

**NANDC\_BCHDE1\_1**

Address: Operational Base + offset (0x00d4)

BCH decode result of 1th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_1 decode result of 1th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_2**

Address: Operational Base + offset (0x00d8)

BCH decode result of 2th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
18:0	RO	0x00000	bchde1_2 decode result of 2th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_3**

Address: Operational Base + offset (0x00dc)  
BCH decode result of 3th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_3 decode result of 3th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_4**

Address: Operational Base + offset (0x00e0)  
BCH decode result of 4th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_4 decode result of 4th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_5**

Address: Operational Base + offset (0x00e4)  
BCH decode result of 5th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_5 decode result of 5th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_6**

Address: Operational Base + offset (0x00e8)  
BCH decode result of 6th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_6 decode result of 6th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_7**

Address: Operational Base + offset (0x00ec)

BCH decode result of 7th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_7 decode result of 7th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_8**

Address: Operational Base + offset (0x00f0)

BCH decode result of 8th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_8 decode result of 8th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_9**

Address: Operational Base + offset (0x00f4)

BCH decode result of 9th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_9 decode result of 9th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_10**

Address: Operational Base + offset (0x00f8)

BCH decode result of 10th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_10 decode result of 10th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_11**

Address: Operational Base + offset (0x00fc)

BCH decode result of 11th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
18:0	RO	0x00000	bchde1_11 decode result of 11th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_12**

Address: Operational Base + offset (0x0100)  
BCH decode result of 12th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_12 decode result of 12th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_13**

Address: Operational Base + offset (0x0104)  
BCH decode result of 13th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_13 decode result of 13th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_14**

Address: Operational Base + offset (0x0108)  
BCH decode result of 14th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_14 decode result of 14th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_15**

Address: Operational Base + offset (0x010c)  
BCH decode result of 15th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_15 decode result of 15th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_16**

Address: Operational Base + offset (0x0110)

BCH decode result of 16th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_16 decode result of 16th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_17**

Address: Operational Base + offset (0x0114)

BCH decode result of 17th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_17 decode result of 17th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_18**

Address: Operational Base + offset (0x0118)

BCH decode result of 18th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_18 decode result of 18th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_19**

Address: Operational Base + offset (0x011c)

BCH decode result of 19th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_19 decode result of 19th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_20**

Address: Operational Base + offset (0x0120)

BCH decode result of 20th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

Bit	Attr	Reset Value	Description
18:0	RO	0x00000	bchde1_20 decode result of 20th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_21**

Address: Operational Base + offset (0x0124)  
BCH decode result of 21th error bit for codeword 1

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_21 decode result of 21th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_22**

Address: Operational Base + offset (0x0128)  
BCH decode result of 22th error bit for codeword 1

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_22 decode result of 22th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_23**

Address: Operational Base + offset (0x012c)  
BCH decode result of 23th error bit for codeword 1

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_23 decode result of 23th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_DLL\_CTL\_REG0**

Address: Operational Base + offset (0x0130)  
DLL Control Register 0

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved
23:16	RW	0x7f	dll_dqs_dly_bypass Holds the read DQS delay setting when the DLL is operating in bypass mode.

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:8	RW	0x7f	dll_dqs_dly Holds the read DQS delay setting when the DLL is operating in normal mode. Typically, this value is 1/4 of a clock cycle. Each increment of this field represents 1/512th of a clock cycle.
7:0	RW	0x05	dll_start_point DLL Start Point Control. This value is loaded into the DLL at initialization and is the value at which the DLL will begin searching for a lock. Each increment of this field represents 1/128th of a clock cycle.

**NANDC\_DLL\_CTL\_REG1**

Address: Operational Base + offset (0x0134)

DLL Control Register 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:12	RO	0x0	reserved
11:4	RW	0x02	dll_incr DLL Increment Value. This sets the increment used by the DLL when searching for a lock. It is recommended keeping this field small (around 0x4) to keep the steps gradual
3:2	RW	0x0	dll_qtren Quarter flag of DLL, active in no-bypass mode. 01:1/4 fclk, dqs_dly=128. 10:1/8 fclk, dqs_dly=64. Default: dqs_dly=dll_dqs_dly(DLL_CTL_REG0[15:8]). When dll_qtr='b01 or 'b10, software not need to configure dll_dqs_dly, and hardware should delay the input signal for 1/4 or 1/8 fclk cycle time; When dll_qtr=0, software need to configure dll_dqs_dly.
1	RW	0x1	dll_bypass DLL Bypass Control, 1active 0: dll not bypass, dll_dqs_dleay= dqs_dly 1: dll bypass, dll_dqs_dleay= dll_dqs_dly_bypass
0	RW	0x0	dll_start Start signal for DLL, 1 active. Notes: It will keep high until dll disabled.

**NANDC\_DLL\_OBS\_REG0**

Address: Operational Base + offset (0x0138)

DLL Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:9	RO	0x01	dll_dqs_delay_value Report the delay value for the read DQS signal

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
8:1	RO	0x00	dll_lock_value Reports the DLL encoder value from the master DLL to the slave DLL's. The slaves use this value to set up their delays for the clk_wr and read DQS signals.
0	RO	0x0	dll_lock DLL Lock indication: 0: DLL has not locked 1: DLL is locked.

**NANDC\_RANDMZ\_CFG**

Address: Operational Base + offset (0x0150)

Randomizer Configure Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	RW	0x0	randmz_en Randomizer enable indication, 1 active. 0: Randomizer active 1: Randomizer not active Notes: a. Not active when data transmission in bypass mode. b. Just active for data, but not for address and command. c. Not active when BchPage=1.
30:29	RW	0x0	randmz_mode Randomizer mode: 00- Samsung randomizer Polynomial= $1+x+x^{15}$ 10- Samsung randomizer Polynomial= $1+x^{14}+x^{15}$ 01-TOSHIBA randomizer
28:24	RW	0x00	page_offset basic seed rotation bits for every 16page
23:20	RW	0x0	cwd_offset basic seed start point for every page
19:0	RW	0x00000	randmz_seed when Samsung randomizer: The seed for randomizer(initial value); when Toshiba randomizer: Seed Agitation Register.

**NANDC\_FMWAIT\_SYN**

Address: Operational Base + offset (0x0158)

Flash Timing Control Register For Synchronous Timing

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15	RW	0x0	ssyn_xle_sel ALE/CLE selection signal for ONFI synchronous flash: 0: ALE/CLE aligned to the falling edge of WRN 1: ALE/CLE aligned to the center of WRN low level

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
14:9	RW	0x00	<p>pst Write/Read Postamble time for ONFI synchronous mode or Toggle data mode. This field specifies the number of processor clock cycle for Postamb- le time.</p>
8:3	RW	0x00	<p>pre Write/Read Preamble time for ONFI synchronous mode or Toggle data mode. This field specifies the number of processor clock cycle for preamble time.</p>
2:0	RW	0x0	<p>fclk Half hclk cycle number for flash clock for ONFI synchronous mode or Toggle data mode</p>

**NANDC\_MTRANS\_STAT2**

Address: Operational Base + offset (0x015c)

Bus Transfer Status Register2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RO	0x0000	<p>bus_err2 Bus error indication for codeword16~31. [0] : bus error for codeword 16 ..... [15] : bus error for codeword 31 Notes: Only active for master-mode.</p>

**NANDC\_NANDC\_VER**

Address: Operational Base + offset (0x0160)

Nandc Version Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x000000701	<p>version Version indication for NANDC</p>

**NANDC\_LLP\_CTL**

Address: Operational Base + offset (0x0164)

LLP Control Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RW	0x00000000	<p>llp_loc Starting address for LLI0, 64byte align</p>
5	RW	0x0	<p>llp_frdy Working time for FOP_WAIT_FRDY for all FOP in first LLP group: 0: FOP_WAIT_FRDY begin working when started 1: FOP_WAIT_FRDY not begin working until 16 cycles later after started</p>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
4:3	RO	0x0	reserved
2	R/W SC	0x0	llp_RST Reset signal for LLP. When asserted, it will auto cleared.
1	RW	0x0	llp_mode 0-current LLI only has FOP 1-current LLI has both CFG and FOP
0	RW	0x0	llp_en Enable signal for LLP 0-LLP disable 1-LLP enable

**NANDC\_LL\_STAT**

Address: Operational Base + offset (0x0168)

LLP Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:6	RO	0x00000000	llp_stat latest LLI_LOC finished, 64byte align
5:2	RO	0x0	reserved
1	RO	0x0	llp_err error status for llp load or execute 0-llp is correct 1-llp is error
0	RO	0x1	llp_rdy ready status for all llp load 0-llp load is busy 1-llp load is ready

**NANDC\_INTEN**

Address: Operational Base + offset (0x016c)

NandC Interrupt Enable Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6	RW	0x0	master_idle_int_en Enable for master idle interrupt 0-interrupt disable 1-interrupt enable When master_idle_int_en is active, an interrupt is generated if posedge of master idle happen

Bit	Attr	Reset Value	Description
5	RW	0x0	<p>flash_abort_int_en Enable for flash read abort interrupt. 0: interrupt disable 1: interrupt enable</p> <p>When flash_abort_int_en is active, an interrupt is generated if DQS input is abort.</p> <p>Available when flash interface is ONFI synchronous or toggle. When read data number is out of range of flash page size, dqs input is abort. An interrupt is generated if flash_abort_int_en is enable</p>
4	RW	0x0	<p>llp_int_en Enable for LLP finished interrupt. 0: interrupt disable 1: interrupt enable</p> <p>When llp_en_en is active, an interrupt is generated if LLP operation is finished</p>
3	RW	0x0	<p>bchfail_int_en Enable for bch fail interrupt. 0-interrupt disable 1-interrupt enable</p> <p>When bchfail_int_en is active, an interrupt is generated if bch decode failed</p>
2	RW	0x0	<p>bcherr_int_en Enable for bch error interrupt. 0-interrupt disable 1-interrupt enable</p> <p>When bcherr_int_en is active, an interrupt is generated if bch decode error bit is larger than bchthres(BCHCTL[26:19])</p>
1	RW	0x0	<p>frdy_int_en Enable for flash_rdy interrupt 0-interrupt disable 1-interrupt enable</p> <p>When frdy_int_en is active, an interrupt is generated if flash R/B# changes from 0 to 1</p>
0	RW	0x0	<p>dma_int_en Enable for internal DMA transfer finished interrupt 0-interrupt disable 1-interrupt enable</p> <p>When dma_int_en is active, an interrupt is generated if page_num(FLCTL[27:22]) of flash data transfer in DMA mode is finished</p>

**NANDC\_INTCLR**

Address: Operational Base + offset (0x0170)

NandC Interrupt Clear Register

Bit	Attr	Reset Value	Description
-----	------	-------------	-------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6	R/W SC	0x0	master_idle_int_clr Clear for master idle interrupt. When asserted, this bit will be auto cleared. 0: interrupt not cleared 1: interrupt cleared
5	R/W SC	0x0	flash_abort_int_clr Clear for flash abort interrupt. When asserted, this bit will be auto cleared. 0: interrupt not cleared 1: interrupt cleared Available when flash interface is ONFI synchronous or toggle
4	R/W SC	0x0	llp_int_clr Clear for LLP finished interrupt. When asserted, this bit will be auto cleared. 0: interrupt not cleared 1: interrupt cleared
3	R/W SC	0x0	bchfail_int_clr Clear for bch decode fail interrupt. When asserted, this bit will be auto cleared. 0-interrupt cleared 1-interrupt not cleared
2	R/W SC	0x0	bcherr_int_clr Clear for bch error interrupt. When asserted, this bit will be auto cleared. 0-interrupt cleared 1-interrupt not cleared
1	R/W SC	0x0	frdy_int_clr Clear for flash_rdy interrupt. When asserted, this bit will be auto cleared. 0-interrupt cleared 1-interrupt not cleared
0	R/W SC	0x0	dma_int_clr Clear for internal DMA transfer finished interrupt. When asserted, this bit will be auto cleared. 0-interrupt cleared 1-interrupt not cleared

**NANDC\_INTST**

Address: Operational Base + offset (0x0174)

NandC Interrupt Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:7	RO	0x0	reserved
6	RO	0x0	master_idle_int_stat Status for master idle interrupt, high active

Bit	Attr	Reset Value	Description
5	RO	0x0	flash_abort_int_stat Status for flash abort, high active Available when flash interface is ONFI synchronous or toggle
4	RO	0x0	llp_int_stat Status for LLP finished interrupt, high active
3	RO	0x0	bchfail_int_stat Status for bch decode fail interrupt, high active
2	RO	0x0	bcherr_int_stat Status for bch error interrupt, high active
1	RO	0x0	frdy_int_stat Status for flash_rdy interrupt, high active
0	RO	0x0	dma_int_stat Status for internal DMA transfer finished interrupt, high active

**NANDC\_SPARE0\_0**

Address: Operational Base + offset (0x0200)

System Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0xff	system_3 the 4th system byte of codeword 0
23:16	RW	0xff	system_2 the 3rd system byte of codeword 0
15:8	RW	0xff	system_1 the 2nd system byte of codeword 0
7:0	RW	0xff	system_0 the 1st system byte of codeword 0

**NANDC\_SPARE0\_1**

Address: Operational Base + offset (0x0204)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_2**

Address: Operational Base + offset (0x0208)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_3**

Address: Operational Base + offset (0x020c)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_4**

Address: Operational Base + offset (0x0210)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_5**

Address: Operational Base + offset (0x0214)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_6**

Address: Operational Base + offset (0x0218)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_7**

Address: Operational Base + offset (0x021c)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_8**

Address: Operational Base + offset (0x0220)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_9**

Address: Operational Base + offset (0x0224)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_10**

Address: Operational Base + offset (0x0228)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_11**

Address: Operational Base + offset (0x022c)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_0**

Address: Operational Base + offset (0x0230)

System Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0xff	system_3 the 4th system byte of codeword 1
23:16	RW	0xff	system_2 the 3rd system byte of codeword 1
15:8	RW	0xff	system_1 the 2nd system byte of codeword 1
7:0	RW	0xff	system_0 the 1st system byte of codeword 1

**NANDC\_SPARE1\_1**

Address: Operational Base + offset (0x0234)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_2**

Address: Operational Base + offset (0x0238)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_3**

Address: Operational Base + offset (0x023c)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_4**

Address: Operational Base + offset (0x0240)

Spare Data and BCH Encode Information for codeword 1

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_5**

Address: Operational Base + offset (0x0244)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_6**

Address: Operational Base + offset (0x0248)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_7**

Address: Operational Base + offset (0x024c)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_8**

Address: Operational Base + offset (0x0250)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_9**

Address: Operational Base + offset (0x0254)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_10**

Address: Operational Base + offset (0x0258)

Spare Data and BCH Encode Information for codeword 1

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_11**

Address: Operational Base + offset (0x025c)

Spare Data and BCH Encode Information for codeword 1

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_12**

Address: Operational Base + offset (0x0260)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_13**

Address: Operational Base + offset (0x0264)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_14**

Address: Operational Base + offset (0x0268)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_15**

Address: Operational Base + offset (0x026c)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_16**

Address: Operational Base + offset (0x0270)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_17**

Address: Operational Base + offset (0x0274)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_18**

Address: Operational Base + offset (0x0278)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_19**

Address: Operational Base + offset (0x027c)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_20**

Address: Operational Base + offset (0x0280)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_21**

Address: Operational Base + offset (0x0284)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_22**

Address: Operational Base + offset (0x0288)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_23**

Address: Operational Base + offset (0x028c)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_24**

Address: Operational Base + offset (0x0290)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_25**

Address: Operational Base + offset (0x0294)

Spare Data and BCH Encode Information for codeword 0

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_26**

Address: Operational Base + offset (0x0298)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH0_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH0_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH0_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE0\_27**

Address: Operational Base + offset (0x029c)

Spare Data and BCH Encode Information for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RW	0x00	BCH0_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_12**

Address: Operational Base + offset (0x02a0)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
------------	-------------	--------------------	--------------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_13**

Address: Operational Base + offset (0x02a4)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_14**

Address: Operational Base + offset (0x02a8)

Spare Data and BCH Encode Information for codeword 1

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_15**

Address: Operational Base + offset (0x02ac)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_16**

Address: Operational Base + offset (0x02b0)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_17**

Address: Operational Base + offset (0x02b4)

Spare Data and BCH Encode Information for codeword 1

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_18**

Address: Operational Base + offset (0x02b8)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_19**

Address: Operational Base + offset (0x02bc)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_20**

Address: Operational Base + offset (0x02c0)

Spare Data and BCH Encode Information for codeword 1

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_21**

Address: Operational Base + offset (0x02c4)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_22**

Address: Operational Base + offset (0x02c8)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_23**

Address: Operational Base + offset (0x02cc)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_24**

Address: Operational Base + offset (0x02d0)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_25**

Address: Operational Base + offset (0x02d4)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25

Bit	Attr	Reset Value	Description
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_26**

Address: Operational Base + offset (0x02d8)

Spare Data and BCH Encode Information for codeword 1

Bit	Attr	Reset Value	Description
31:24	RW	0x00	BCH1_4x_3 (4x+3)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
23:16	RW	0x00	BCH1_4x_2 (4x+2)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~9 40bitBCH: x=0~16 60bitBCH: x=0~25
15:8	RW	0x00	BCH1_4x_1 (4x+1)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~25
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_SPARE1\_27**

Address: Operational Base + offset (0x02dc)

Spare Data and BCH Encode Information for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RW	0x00	BCH1_4x_0 (4x+0)th byte of bch spare bits 16bitBCH: x=0~6 24bitBCH: x=0~10 40bitBCH: x=0~17 60bitBCH: x=0~26

**NANDC\_BCHDE0\_24**

Address: Operational Base + offset (0x0400)

BCH decode result of 24th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_24 decode result of 24th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_25**

Address: Operational Base + offset (0x0404)

BCH decode result of 25th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_25 decode result of 25th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_26**

Address: Operational Base + offset (0x0408)

BCH decode result of 26th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_26 decode result of 26th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_27**

Address: Operational Base + offset (0x040c)

BCH decode result of 27th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
18:0	RO	0x00000	bchde0_27 decode result of 27th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_28**

Address: Operational Base + offset (0x0410)

BCH decode result of 28th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_28 decode result of 28th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_29**

Address: Operational Base + offset (0x0414)

BCH decode result of 29th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_29 decode result of 29th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_30**

Address: Operational Base + offset (0x0418)

BCH decode result of 30th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_30 decode result of 30th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_31**

Address: Operational Base + offset (0x041c)

BCH decode result of 31th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_31 decode result of 31th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_32**

Address: Operational Base + offset (0x0420)

BCH decode result of 32th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_3 decode result of 3th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_33**

Address: Operational Base + offset (0x0424)

BCH decode result of 33th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_33 decode result of 33th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_34**

Address: Operational Base + offset (0x0428)

BCH decode result of 3th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_34 decode result of 34th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_35**

Address: Operational Base + offset (0x042c)

BCH decode result of 35th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_35 decode result of 35th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_36**

Address: Operational Base + offset (0x0430)

BCH decode result of 36th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
18:0	RO	0x00000	bchde0_36 decode result of 36th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_37**

Address: Operational Base + offset (0x0434)  
BCH decode result of 37th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_37 decode result of 37th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_38**

Address: Operational Base + offset (0x0438)  
BCH decode result of 38th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_38 decode result of 38th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_39**

Address: Operational Base + offset (0x043c)  
BCH decode result of 39th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_39 decode result of 39th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_40**

Address: Operational Base + offset (0x0440)  
BCH decode result of 40th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_40 decode result of 40th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_41**

Address: Operational Base + offset (0x0444)

BCH decode result of 41th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_41 decode result of 41th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_42**

Address: Operational Base + offset (0x0448)

BCH decode result of 42th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_42 decode result of 42th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_43**

Address: Operational Base + offset (0x044c)

BCH decode result of 43th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_43 decode result of 43th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_44**

Address: Operational Base + offset (0x0450)

BCH decode result of 44th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_44 decode result of 44th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_45**

Address: Operational Base + offset (0x0454)

BCH decode result of 45th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

Bit	Attr	Reset Value	Description
18:0	RO	0x00000	bchde0_45 decode result of 45th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_46**

Address: Operational Base + offset (0x0458)  
BCH decode result of 46th error bit for codeword 0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_46 decode result of 46th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_47**

Address: Operational Base + offset (0x045c)  
BCH decode result of 47th error bit for codeword 0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_47 decode result of 47th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_48**

Address: Operational Base + offset (0x0460)  
BCH decode result of 48th error bit for codeword 0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_48 decode result of 48th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_49**

Address: Operational Base + offset (0x0464)  
BCH decode result of 49th error bit for codeword 0

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_49 decode result of 49th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_50**

Address: Operational Base + offset (0x0468)

BCH decode result of 50th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_50 decode result of 50th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_51**

Address: Operational Base + offset (0x046c)

BCH decode result of 51th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_51 decode result of 51th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_52**

Address: Operational Base + offset (0x0470)

BCH decode result of 52th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_52 decode result of 52th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_53**

Address: Operational Base + offset (0x0474)

BCH decode result of 53th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_53 decode result of 53th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_54**

Address: Operational Base + offset (0x0478)

BCH decode result of 54th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
18:0	RO	0x00000	bchde0_54 decode result of 54th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_55**

Address: Operational Base + offset (0x047c)  
BCH decode result of 55th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_55 decode result of 55th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_56**

Address: Operational Base + offset (0x0480)  
BCH decode result of 56th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_56 decode result of 56th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_57**

Address: Operational Base + offset (0x0484)  
BCH decode result of 57th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_57 decode result of 57th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_58**

Address: Operational Base + offset (0x0488)  
BCH decode result of 58th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_58 decode result of 58th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE0\_59**

Address: Operational Base + offset (0x048c)

BCH decode result of 59th error bit for codeword 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde0_59 decode result of 59th error bit for codeword 0. Bit assignment is similar to BCHDE0_0 register. For more description, please refer to BCHDE0_0 register.

**NANDC\_BCHDE1\_24**

Address: Operational Base + offset (0x0490)

BCH decode result of 24th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_24 decode result of 24th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_25**

Address: Operational Base + offset (0x0494)

BCH decode result of 25th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_25 decode result of 25th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_26**

Address: Operational Base + offset (0x0498)

BCH decode result of 26th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_26 decode result of 26th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_27**

Address: Operational Base + offset (0x049c)

BCH decode result of 27th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
18:0	RO	0x00000	bchde1_27 decode result of 27th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_28**

Address: Operational Base + offset (0x04a0)

BCH decode result of 28th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_28 decode result of 28th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_29**

Address: Operational Base + offset (0x04a4)

BCH decode result of 29th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_29 decode result of 29th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_30**

Address: Operational Base + offset (0x04a8)

BCH decode result of 30th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_30 decode result of 30th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_31**

Address: Operational Base + offset (0x04ac)

BCH decode result of 31th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_31 decode result of 31th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_32**

Address: Operational Base + offset (0x04b0)

BCH decode result of 32th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_32 decode result of 32th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_33**

Address: Operational Base + offset (0x04b4)

BCH decode result of 33th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_33 decode result of 33th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_34**

Address: Operational Base + offset (0x04b8)

BCH decode result of 34th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_34 decode result of 34th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_35**

Address: Operational Base + offset (0x04bc)

BCH decode result of 35th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_35 decode result of 35th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_36**

Address: Operational Base + offset (0x04c0)

BCH decode result of 3th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
18:0	RO	0x00000	bchde1_36 decode result of 36th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_37**

Address: Operational Base + offset (0x04c4)  
BCH decode result of 37th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_37 decode result of 37th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_38**

Address: Operational Base + offset (0x04c8)  
BCH decode result of 38th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_38 decode result of 38th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_39**

Address: Operational Base + offset (0x04cc)  
BCH decode result of 39th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_39 decode result of 39th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_40**

Address: Operational Base + offset (0x04d0)  
BCH decode result of 40th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_40 decode result of 40th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_41**

Address: Operational Base + offset (0x04d4)

BCH decode result of 41th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_41 decode result of 41th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_42**

Address: Operational Base + offset (0x04d8)

BCH decode result of 42th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_42 decode result of 42th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_43**

Address: Operational Base + offset (0x04dc)

BCH decode result of 43th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_43 decode result of 43th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_44**

Address: Operational Base + offset (0x04e0)

BCH decode result of 44th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_44 decode result of 44th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_45**

Address: Operational Base + offset (0x04e4)

BCH decode result of 45th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
18:0	RO	0x00000	bchde1_45 decode result of 45th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_46**

Address: Operational Base + offset (0x04e8)  
BCH decode result of 46th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_46 decode result of 46th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_47**

Address: Operational Base + offset (0x04ec)  
BCH decode result of 47th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_47 decode result of 47th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_48**

Address: Operational Base + offset (0x04f0)  
BCH decode result of 48th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_48 decode result of 48th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_49**

Address: Operational Base + offset (0x04f4)  
BCH decode result of 49th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_49 decode result of 49th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_50**

Address: Operational Base + offset (0x04f8)

BCH decode result of 50th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_50 decode result of 50th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_51**

Address: Operational Base + offset (0x04fc)

BCH decode result of 51th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_51 decode result of 51th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_52**

Address: Operational Base + offset (0x0500)

BCH decode result of 52th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_52 decode result of 52th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_53**

Address: Operational Base + offset (0x0504)

BCH decode result of 53th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_53 decode result of 53th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_54**

Address: Operational Base + offset (0x0508)

BCH decode result of 54th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
18:0	RO	0x00000	bchde1_54 decode result of 54th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_55**

Address: Operational Base + offset (0x050c)  
BCH decode result of 55th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_55 decode result of 55th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_56**

Address: Operational Base + offset (0x0510)  
BCH decode result of 56th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_56 decode result of 56th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_57**

Address: Operational Base + offset (0x0514)  
BCH decode result of 57th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_57 decode result of 57th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_58**

Address: Operational Base + offset (0x0518)  
BCH decode result of 58th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_58 decode result of 58th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHDE1\_59**

Address: Operational Base + offset (0x051c)

BCH decode result of 59th error bit for codeword 1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RO	0x00000	bchde1_59 decode result of 59th error bit for codeword 1. Bit assignment is similar to BCHDE1_0 register. For more description, please refer to BCHDE1_0 register.

**NANDC\_BCHST8**

Address: Operational Base + offset (0x0520)

BCH Status Register For Codeword 16~17

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd16_cwd17 BCHST information for 16th and 17th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST9**

Address: Operational Base + offset (0x0524)

BCH Status Register For Codeword 18~19

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd18_cwd19 BCHST information for 18th and 19th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST10**

Address: Operational Base + offset (0x0528)

BCH Status Register For Codeword 20~21

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd20_cwd21 BCHST information for 20th and 21th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST11**

Address: Operational Base + offset (0x052c)

BCH Status Register For Codeword 22~23

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd22_cwd23 BCHST information for 22th and 23th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST12**

Address: Operational Base + offset (0x0530)

BCH Status Register For Codeword 24~25

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd24_cwd25 BCHST information for 24th and 25th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST13**

Address: Operational Base + offset (0x0534)

BCH Status Register For Codeword 26~27

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd26_cwd27 BCHST information for 26th and 27th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST14**

Address: Operational Base + offset (0x0538)

BCH Status Register For Codeword 28~29

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd28_cwd29 BCHST information for 28th and 29th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_BCHST15**

Address: Operational Base + offset (0x053c)

BCH Status Register For Codeword 30~31

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	bchst_cwd30_cwd31 BCHST information for 30th and 31th backup codeword. Bit assignment is similar to BCHST1 register. For more description, please refer to BCHST1 register.

**NANDC\_RANDMZ\_SEED13\_0**

Address: Operational Base + offset (0x0600)

Seed 0 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_0 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_1**

Address: Operational Base + offset (0x0604)  
 Seed 1 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_1 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_2**

Address: Operational Base + offset (0x0608)  
 Seed 2 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_2 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_3**

Address: Operational Base + offset (0x060c)  
 Seed 3 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_3 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_4**

Address: Operational Base + offset (0x0610)  
 Seed 4 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_4 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_5**

Address: Operational Base + offset (0x0614)  
 Seed 5 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_5 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_6**

Address: Operational Base + offset (0x0618)  
 Seed 6 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_6 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_7**

Address: Operational Base + offset (0x061c)

Seed 7 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_7 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_8**

Address: Operational Base + offset (0x0620)

Seed 8 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_8 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_9**

Address: Operational Base + offset (0x0624)

Seed 9 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_9 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_10**

Address: Operational Base + offset (0x0628)

Seed 10 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_10 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_11**

Address: Operational Base + offset (0x062c)

Seed 11 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_11 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_12**

Address: Operational Base + offset (0x0630)

Seed 12 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
------------	-------------	--------------------	--------------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_12 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_13**

Address: Operational Base + offset (0x0634)

Seed 13 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_13 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_14**

Address: Operational Base + offset (0x0638)

Seed 14 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_14 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED13\_15**

Address: Operational Base + offset (0x063c)

Seed 15 for Toshiba 13 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	RO	0x0	reserved
12:0	RW	0x0000	randmz_seed13_15 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_0**

Address: Operational Base + offset (0x0640)

Seed 0 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_0 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_1**

Address: Operational Base + offset (0x0644)

Seed 1 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_1 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_2**

Address: Operational Base + offset (0x0648)

Seed 2 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_2 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_3**

Address: Operational Base + offset (0x064c)

Seed 3 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_3 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_4**

Address: Operational Base + offset (0x0650)

Seed 4 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_4 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_5**

Address: Operational Base + offset (0x0654)

Seed 5 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_5 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_6**

Address: Operational Base + offset (0x0658)

Seed 6 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_6 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_7**

Address: Operational Base + offset (0x065c)

Seed 7 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
16:0	RW	0x00000	randmz_seed17_7 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_8**

Address: Operational Base + offset (0x0660)

Seed 8 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_8 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_9**

Address: Operational Base + offset (0x0664)

Seed 9 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_9 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_10**

Address: Operational Base + offset (0x0668)

Seed 10 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_10 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_11**

Address: Operational Base + offset (0x066c)

Seed 11 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_11 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_12**

Address: Operational Base + offset (0x0670)

Seed 12 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_12 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_13**

Address: Operational Base + offset (0x0674)  
 Seed 13 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_13 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_14**

Address: Operational Base + offset (0x0678)  
 Seed 14 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_14 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED17\_15**

Address: Operational Base + offset (0x067c)  
 Seed 15 for Toshiba 17 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:17	RO	0x0	reserved
16:0	RW	0x00000	randmz_seed17_15 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_0**

Address: Operational Base + offset (0x0680)  
 Seed 0 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_0 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_1**

Address: Operational Base + offset (0x0684)  
 Seed 1 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_1 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_2**

Address: Operational Base + offset (0x0688)  
 Seed 2 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_2 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_3**

Address: Operational Base + offset (0x068c)

Seed 3 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_3 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_4**

Address: Operational Base + offset (0x0690)

Seed 4 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_4 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_5**

Address: Operational Base + offset (0x0694)

Seed 5 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_5 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_6**

Address: Operational Base + offset (0x0698)

Seed 6 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_6 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_7**

Address: Operational Base + offset (0x069c)

Seed 7 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_7 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_8**

Address: Operational Base + offset (0x06a0)

Seed 8 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
------------	-------------	--------------------	--------------------

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_8 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_9**

Address: Operational Base + offset (0x06a4)

Seed 9 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_9 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_10**

Address: Operational Base + offset (0x06a8)

Seed 10 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_10 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_11**

Address: Operational Base + offset (0x06ac)

Seed 11 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_11 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_12**

Address: Operational Base + offset (0x06b0)

Seed 12 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_12 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_13**

Address: Operational Base + offset (0x06b4)

Seed 13 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x00000	randmz_seed19_13 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_14**

Address: Operational Base + offset (0x06b8)

Seed 14 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x000000	randmz_seed19_14 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED19\_15**

Address: Operational Base + offset (0x06bc)

Seed 15 for Toshiba 19 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:19	RO	0x0	reserved
18:0	RW	0x000000	randmz_seed19_15 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_0**

Address: Operational Base + offset (0x06c0)

Seed 0 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x0000000	randmz_seed23_0 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_1**

Address: Operational Base + offset (0x06c4)

Seed 1 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x0000000	randmz_seed23_1 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_2**

Address: Operational Base + offset (0x06c8)

Seed 2 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x0000000	randmz_seed23_2 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_3**

Address: Operational Base + offset (0x06cc)

Seed 3 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
22:0	RW	0x000000	randmz_seed23_3 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_4**

Address: Operational Base + offset (0x06d0)

Seed 4 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_4 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_5**

Address: Operational Base + offset (0x06d4)

Seed 5 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_5 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_6**

Address: Operational Base + offset (0x06d8)

Seed 6 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_6 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_7**

Address: Operational Base + offset (0x06dc)

Seed 7 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_7 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_8**

Address: Operational Base + offset (0x06e0)

Seed 8 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_8 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_9**

Address: Operational Base + offset (0x06e4)  
 Seed 9 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_9 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_10**

Address: Operational Base + offset (0x06e8)  
 Seed 10 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_10 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_11**

Address: Operational Base + offset (0x06ec)  
 Seed 11 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_11 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_12**

Address: Operational Base + offset (0x06f0)  
 Seed 12 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_12 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_13**

Address: Operational Base + offset (0x06f4)  
 Seed 13 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_13 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_14**

Address: Operational Base + offset (0x06f8)  
 Seed 14 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_14 The seed for randomizer(initial value);

**NANDC\_RANDMZ\_SEED23\_15**

Address: Operational Base + offset (0x06fc)

Seed 15 for Toshiba 23 Power Polynomial Randomizer

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:23	RO	0x0	reserved
22:0	RW	0x000000	randmz_seed23_15 The seed for randomizer(initial value);

**NANDC\_FLASH0\_DATA**

Address: Operational Base + offset (0x0800)

data to be write into or read from flash0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	flash0_data valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH0_DATA[15:0] is used for Asynchronous 16 bits mode or Toggle mode and FLASH0_DATA[7:0] is used for 8 bits Asynchronous mode. If Synchronous mode is selected, please refer to FLASH0_DATA_SYN.

**NANDC\_FLASH0\_ADDR**

Address: Operational Base + offset (0x0804)

flash0 address

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash0_addr page or block address send to flash0 before flash erase,read or write operation

**NANDC\_FLASH0\_CMD**

Address: Operational Base + offset (0x0808)

command send to flash0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash0_cmd contain the command write to flash0

**NANDC\_FLASH0\_DATA\_SYN**

Address: Operational Base + offset (0x080c)

data to be write into or read from flash0 for Synchronous flash

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:0	RW	0x0000	flash0_data_syn valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH0_DATA_SYN[15:0] is used for Synchronous flash

**NANDC\_FLASH1\_DATA**

Address: Operational Base + offset (0x0900)

data to be write into or read from flash1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	flash1_data valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH1_DATA[15:0] is used for Asynchronous 16 bits mode or Toggle mode and FLASH1_DATA[7:0] is used for 8 bits Asynchronous mode. If Synchronous mode is selected, please refer to FLASH1_DATA_SYN.

**NANDC\_FLASH1\_ADDR**

Address: Operational Base + offset (0x0904)

flash1 address

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash1_addr page or block address send to flash1 before flash erase, read or write operation

**NANDC\_FLASH1\_CMD**

Address: Operational Base + offset (0x0908)

command send to flash1

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash1_cmd contain the command write to flash1

**NANDC\_FLASH1\_DATA\_SYN**

Address: Operational Base + offset (0x090c)

data to be write into or read from flash1 for Synchronous flash

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	flash1_data_syn valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH1_DATA_SYN[15:0] is used for Synchronous flash

**NANDC\_FLASH2\_DATA**

Address: Operational Base + offset (0x0a00)

data to be write into or read from flash2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	flash2_data valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH2_DATA[15:0] is used for Asynchronous 16 bits mode or Toggle mode and FLASH2_DATA[7:0] is used for 8 bits Asynchronous mode. If Synchronous mode is selected, please refer to FLASH2_DATA_SYN.

**NANDC\_FLASH2\_ADDR**

Address: Operational Base + offset (0x0a04)

flash2 address

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash2_addr page or block address send to flash2 before flash erase, read or write operation

**NANDC\_FLASH2\_CMD**

Address: Operational Base + offset (0x0a08)

command send to flash2

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash2_cmd contain the command write to flash2

**NANDC\_FLASH2\_DATA\_SYN**

Address: Operational Base + offset (0x0a0c)

data to be write into or read from flash2 for Synchronous flash

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	flash2_data_syn valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH2_DATA_SYN[15:0] is used for Synchronous flash

**NANDC\_FLASH3\_DATA**

Address: Operational Base + offset (0x0b00)

data to be write into or read from flash3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:0	RW	0x0000	flash3_data valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH3_DATA[15:0] is used for Asynchronous 16 bits mode or Toggle mode and FLASH3_DATA[7:0] is used for 8 bits Asynchronous mode. If Synchronous mode is selected, please refer to FLASH3_DATA_SYN.

**NANDC\_FLASH3\_ADDR**

Address: Operational Base + offset (0x0b04)

flash3 address

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash3_addr page or block address send to flash3 before flash erase,read or write operation

**NANDC\_FLASH3\_CMD**

Address: Operational Base + offset (0x0b08)

command send to flash3

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash3_cmd contain the command write to flash3

**NANDC\_FLASH3\_DATA\_SYN**

Address: Operational Base + offset (0x0b0c)

data to be write into or read from flash3 for Synchronous flash

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	flash3_data_syn valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH3_DATA_SYN[15:0] is used for Synchronous flash

**NANDC\_FLASH4\_DATA**

Address: Operational Base + offset (0x0c00)

data to be write into or read from flash4

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:0	RW	0x0000	flash4_data valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH4_DATA[15:0] is used for Asynchronous 16 bits mode or Toggle mode and FLASH4_DATA[7:0] is used for 8 bits Asynchronous mode. If Synchronous mode is selected, please refer to FLASH4_DATA_SYN.

**NANDC\_FLASH4\_ADDR**

Address: Operational Base + offset (0x0c04)

flash4 address

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash4_addr page or block address send to flash4 before flash erase,read or write operation

**NANDC\_FLASH4\_CMD**

Address: Operational Base + offset (0x0c08)

command send to flash4

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash4_cmd contain the command write to flash4

**NANDC\_FLASH4\_DATA\_SYN**

Address: Operational Base + offset (0x0c0c)

data to be write into or read from flash4 for Synchronous flash

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	flash4_data_syn valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH4_DATA_SYN[15:0] is used for Synchronous flash

**NANDC\_FLASH5\_DATA**

Address: Operational Base + offset (0x0d00)

data to be write into or read from flash5

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:0	RW	0x0000	flash5_data valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH5_DATA[15:0] is used for Asynchronous 16 bits mode or Toggle mode and FLASH5_DATA[7:0] is used for 8 bits Asynchronous mode. If Synchronous mode is selected, please refer to FLASH5_DATA_SYN.

**NANDC\_FLASH5\_ADDR**

Address: Operational Base + offset (0x0d04)

flash5 address

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash5_addr page or block address send to flash5 before flash erase,read or write operation

**NANDC\_FLASH5\_CMD**

Address: Operational Base + offset (0x0d08)

command send to flash5

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash5_cmd contain the command write to flash5

**NANDC\_FLASH5\_DATA\_SYN**

Address: Operational Base + offset (0x0d0c)

data to be write into or read from flash5 for Synchronous flash

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	flash5_data_syn valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH5_DATA_SYN[15:0] is used for Synchronous flash

**NANDC\_FLASH6\_DATA**

Address: Operational Base + offset (0x0e00)

data to be write into or read from flash6

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
15:0	RW	0x0000	flash6_data valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH6_DATA[15:0] is used for Asynchronous 16 bits mode or Toggle mode and FLASH6_DATA[7:0] is used for 8 bits Asynchronous mode. If Synchronous mode is selected, please refer to FLASH6_DATA_SYN.

**NANDC\_FLASH6\_ADDR**

Address: Operational Base + offset (0x0e04)

flash6 address

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash6_addr page or block address send to flash6 before flash erase,read or write operation

**NANDC\_FLASH6\_CMD**

Address: Operational Base + offset (0x0e08)

command send to flash6

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	WO	0x00	flash6_cmd contain the command write to flash6

**NANDC\_FLASH6\_DATA\_SYN**

Address: Operational Base + offset (0x0e0c)

data to be write into or read from flash6 for Synchronous flash

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved
15:0	RW	0x0000	flash6_data_syn valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH6_DATA_SYN[15:0] is used for Synchronous flash

**NANDC\_FLASH7\_DATA**

Address: Operational Base + offset (0x0f00)

data to be write into or read from flash7

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:16	RO	0x0	reserved

Bit	Attr	Reset Value	Description
15:0	RW	0x0000	flash7_data valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH7_DATA[15:0] is used for Asynchronous 16 bits mode or Toggle mode and FLASH7_DATA[7:0] is used for 8 bits Asynchronous mode. If Synchronous mode is selected, please refer to FLASH7_DATA_SYN.

**NANDC\_FLASH7\_ADDR**

Address: Operational Base + offset (0x0f04)

flash7 address

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	WO	0x00	flash7_addr page or block address send to flash7 before flash erase,read or write operation

**NANDC\_FLASH7\_CMD**

Address: Operational Base + offset (0x0f08)

command send to flash7

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	WO	0x00	flash7_cmd contain the command write to flash7

**NANDC\_FLASH7\_DATA\_SYN**

Address: Operational Base + offset (0x0f0c)

data to be write into or read from flash7 for Synchronous flash

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0000	flash7_data_syn valid for flash bypass internal dma mode(FLCTL[3]=0) FLASH7_DATA_SYN[15:0] is used for Synchronous flash

**10.5 Interface Description**

Table 10-2 NandC Interface Description

Module Pin	Direction	Pad Name	IOMUX Setting
flash_wp	O	FLASH_WP	GRF_GPIO2A_IOMUX[11:10]=2'b01
flash_ale	O	FLASH_ALE	GRF_GPIO2A_IOMUX[1:0]=2'b01
flash_cle	O	FLASH_CLE	GRF_GPIO2A_IOMUX[3:2]=2'b01
flash_wrn	O	FLASH_WRN	GRF_GPIO2A_IOMUX[5:4]=2'b01
flash_rdn	O	FLASH_RDN	GRF_GPIO2A_IOMUX[7:6]=2'b01
flash_data[0]	I/O	FLASH_DATA[0]	GRF_GPIO1D_IOMUX[1:0]=2'b01
flash_data[1]	I/O	FLASH_DATA[1]	GRF_GPIO1D_IOMUX[3:2]=2'b01
flash_data[2]	I/O	FLASH_DATA[2]	GRF_GPIO1D_IOMUX[5:4]=2'b01
flash_data[3]	I/O	FLASH_DATA[3]	GRF_GPIO1D_IOMUX[7:6]=2'b01

Module Pin	Direction	Pad Name	IOMUX Setting
flash_data[4]	I/O	FLASH_DATA[4]	GRF_GPIO1D_IOMUX[9:8]=2'b01
flash_data[5]	I/O	FLASH_DATA[5]	GRF_GPIO1D_IOMUX[11:10]=2'b01
flash_data[6]	I/O	FLASH_DATA[6]	GRF_GPIO1D_IOMUX[13:12]=2'b01
flash_data[7]	I/O	FLASH_DATA[7]	GRF_GPIO1D_IOMUX[15:14]=2'b01
flash_data[8]	I/O	N/A	N/A
flash_data[9]	I/O	N/A	N/A
flash_data[10]	I/O	N/A	N/A
flash_data[11]	I/O	N/A	N/A
flash_data[12]	I/O	N/A	N/A
flash_data[13]	I/O	N/A	N/A
flash_data[14]	I/O	N/A	N/A
flash_data[15]	I/O	N/A	N/A
flash_dqs	I/O	FLASH_DQS	GRF_GPIO2A_IOMUX[15:14]=2'b01
flash_rdy	I	FLASH_RDY	GRF_GPIO2A_IOMUX[9:8]=2'b01
flash_csn0	O	FLASH0_CSNO	GRF_GPIO2A_IOMUX[13:12]=2'b01
flash_csn1	O	FLASH0_CSNI	GRF_GPIO0C_IOMUX[15:14]=2'b01
flash_csn2	O	FLASH0_CSN2	GRF_GPIO1C_IOMUX[13:12]=2'b01
flash_csn3	O	FLASH0_CSN3	GRF_GPIO1C_IOMUX[15:14]=2'b01

Notes: I=input, O=output, I/O=input/output, bidirectional

Furthermore, different IOs are selected and connected to different flash interface, which is shown as follows.

Table 10-3 NandC Interface Connection

Module Pin	Direction	Flash Interface			
		Asyn8x	Asyn16x	ONFI	Toggle
flash_csn <i>i</i> (i=0~3)	O	✓	-	✓	✓
flash_wp	O	✓	✓	✓	✓
flash_ale	O	✓	✓	✓	✓
flash_cle	O	✓	✓	✓	✓
flash_wrn	O	✓	✓	✓	✓
flash_rdn	O	✓	✓	✓	✓
flash_data[7:0]	I/O	✓	✓	✓	✓
flash_data[15:8]	I/O	-	-	-	-
flash_dqs	I/O	-	-	✓	✓
flash_rdy	I	✓	✓	✓	✓

## 10.6 Application Notes

### 10.6.1 BCHST/BCHLOC/BCHDE/SPARE Application

#### 1. BCHST

There are 8 BCHST-registers in NandC to store 16 codeword's BCH decode status(bchst) information. Every register stores 2 codeword's bchst information except BCHST0, which not only includes bchst information, but also includes one bit for bchrny.

Let bchst\_cwd0~bchst\_cwd15 be the bchst information for 16 codewords. In BCHST-registers, the latest codeword's bchst is stored into bchst\_cwd0, and the former is shifted into bchst\_cwd1. That is, bchst\_cwd0 → bchst\_cwd1 → ..... → bchst\_cwd15. Therefore, for example, if 16 codewords are decoded, then bchst\_cwd0 is the bch decode status for codeword15, and bchst\_cwd15 is the bch decode status for codeword0.

bchst\_cwd0 = {BCHST0[28], BCHST0[12:8], BCHST0[27], BCHST0[7:3], BCHST0[2:0]}

bchst\_cwd1 = {BCHST0[30], BCHST0[25:21], BCHST0[29], BCHST0[20:16],

BCHST0[15:13]}

bchst\_cwd2 = {BCHST1[28], BCHST1[12:8], BCHST1[27], BCHST1[7:3], BCHST1[2:0]}

bchst\_cwd3 = {BCHST1[30], BCHST1[25:21], BCHST1[29], BCHST1[20:16],

BCHST1[15:13]}

bchst\_cwd4 = {BCHST2[28], BCHST2[12:8], BCHST2[27], BCHST2[7:3], BCHST2[2:0]}

bchst\_cwd5 = {BCHST2[30], BCHST2[25:21], BCHST2[29], BCHST2[20:16],

```
BCHST2[15:13]}
bchst_cwd6 = {BCHST3[28], BCHST3[12:8] , BCHST3[27], BCHST3[7:3], BCHST3[2:0]}
bchst_cwd7 = {BCHST3[30], BCHST3[25:21], BCHST3[29], BCHST3[20:16],
BCHST3[15:13]}
bchst_cwd8 = {BCHST4[28], BCHST4[12:8] , BCHST4[27], BCHST4[7:3], BCHST4[2:0]}
bchst_cwd9 = {BCHST4[30], BCHST4[25:21], BCHST4[29], BCHST4[20:16],
BCHST4[15:13]}
bchst_cwd10 = {BCHST5[28], BCHST5[12:8] , BCHST5[27], BCHST5[7:3], BCHST5[2:0]}
bchst_cwd11 = {BCHST5[30], BCHST5[25:21], BCHST5[29], BCHST5[20:16],
BCHST5[15:13]}
bchst_cwd12 = {BCHST6[28], BCHST6[12:8] , BCHST6[27], BCHST6[7:3], BCHST6[2:0]}
bchst_cwd13 = {BCHST6[30], BCHST6[25:21], BCHST6[29], BCHST6[20:16],
BCHST6[15:13]}
bchst_cwd14 = {BCHST7[28], BCHST7[12:8] , BCHST7[27], BCHST7[7:3], BCHST7[2:0]}
bchst_cwd15 = {BCHST7[30], BCHST7[25:21], BCHST7[29], BCHST7[20:16],
BCHST7[15:13]}
```

## 2. BCHLOC

There are 4 BCHLOC-registers in NandC to store 16 codeword's bch decode location(bchloc) information.

Let bchloc\_cwd0~bchloc\_cwd15 be the bchloc information for the 16 codeword. In BCHLOC registers, the latest codeword's bchloc is stored into bchloc\_cwd0, and the former is shifted into bchloc\_cwd1. That is, bchloc\_cwd0→bchloc\_cwd1→...→bchloc\_cwd15. Therefore, for example, if 16 codeword are decoded, then bchloc\_cwd0 is the bch decode status for codeword15, and bchloc\_cwd15 is the bch decode status for codeword0.

```
bchloc_cwd0 = {BCHLOC3[0], BCHLOC0[4:0]}
bchloc_cwd1 = {BCHLOC3[1], BCHLOC0[9:5]}
bchloc_cwd2 = {BCHLOC3[2], BCHLOC0[14:10]}
bchloc_cwd3 = {BCHLOC3[3], BCHLOC0[19:15]}
bchloc_cwd4 = {BCHLOC3[4], BCHLOC0[24:20]}
bchloc_cwd5 = {BCHLOC3[5], BCHLOC0[29:25]}
bchloc_cwd6 = {BCHLOC3[6], BCHLOC1[4:0]}
bchloc_cwd7 = {BCHLOC3[7], BCHLOC1[9:5]}
bchloc_cwd8 = {BCHLOC3[8], BCHLOC1[14:10]}
bchloc_cwd9 = {BCHLOC3[9], BCHLOC1[19:15]}
bchloc_cwd10 = {BCHLOC3[10], BCHLOC1[24:20]}
bchloc_cwd11 = {BCHLOC3[11], BCHLOC1[29:25]}
bchloc_cwd12 = {BCHLOC3[12], BCHLOC2[4:0]}
bchloc_cwd13 = {BCHLOC3[13], BCHLOC2[9:5]}
bchloc_cwd14 = {BCHLOC3[14], BCHLOC2[14:10]}
bchloc_cwd15 = {BCHLOC3[15], BCHLOC2[19:15]}
```

## 3. BCHDE

BCHDE includes two register-groups, BCHDE0 and BCHDE1. Each group has 60 registers: BCHDE0\_0~BCHDE0\_59 and BCHDE1\_0~BCHDE1\_59. BCHDE0\_n(n=0~59) is the decode information of the nth error bit for codeword in sram0, and BCHDE1\_n(n=0~59) is the decode information of the nth error bit for codeword in sram1.

The needed number of BCHDE registers is determined by bchmode. That is:

- When 16bitBCH selected, BCHDEM\_0 ~ BCHDEM\_15 are available
- When 24bitBCH selected, BCHDEM\_0 ~ BCHDEM\_23 are available
- When 40bitBCH selected, BCHDEM\_0 ~ BCHDEM\_39 are available
- When 60bitBCH selected, BCHDEM\_0 ~ BCHDEM\_59 are available

## 4. SPARE

SPARE includes two register-groups, SPARE0 and SPARE1. Each group has 28 registers: SPARE0\_0~SPARE0\_27 and SPARE1\_0~SPARE1\_27.

When in bch encoding, SPARE0\_0 stores system information for codeword in sram0, SPARE0\_n(n=1~27) stores encode information for codeword in sram0; SPARE1\_0 stores system information for codeword in sram1, SPARE1\_n( n=1~27) stores encode information for codeword in sram1.

When in bch decoding, SPARE0\_n(n=0~27) stores the spare data read from flash for codeword in sram0 if BCHCTL[28]=0, the bch decode status(BCHST) information which reflects the current codeword will be written into ddr instead of SPARE0\_1 if BCHCTL[28]=1; SPARE1\_n(n=0~27) stores the spare data read from flash for codeword in sram1 if BCHCTL[28]=0, the bch decode status(BCHST) information which reflects the current codeword will be written into ddr instead of SPARE1\_1 if BCHCTL[28]=1.

The format of bch decode status (BCHCTL) is as follows:

BCHCTL= {BCHST0[28], BCHST0[12:8], BCHST0[27], BCHST0[7:3], BCHST0[2:0]}

The needed number of BCHDE registers is determined by bchmode. That is:

- a. When 16bitBCH selected, spare data=28bytes, SPAREm\_0~SPAREm\_7 are available
- b. When 24bitBCH selected, spare data=42bytes, SPAREm\_0~SPAREm\_10 and SPAREm\_11[15:0] are available
- c. When 40bitBCH selected, spare data=70bytes, SPAREm\_0~SPAREm\_17 and SPAREm\_18[15:0] are available
- d. When 60bitBCH selected, spare data=105bytes, SPAREm\_0~SPAREm\_26 and SPAREm\_27[7:0] are available

## 10.6.2 Bus Mode Application

MTRANS\_CFG[2] determines whether the data load/store between internal memory and external memory is through slave interface or master interface.

### 1. Slave Mode

When MTRANS\_CFG[2]=0, slave is selected. i. e. , flash data load/store between internal memory and external memory is through slave interface by cpu or external DMA.

In this mode, software should store page data into internal memory and spare data into SPARE registers before starting flash program operation; and should load page data from internal memory and spare data from SPARE registers after finishing flash read operation.

In this mode, MTRANS\_CFG, MTRANS\_SADDR0 and MTRANS\_SADDR1 are unused. The transfer codeword number is determined by FLCTL[6:5], and the maximum number is 2.

The judgment condition for finishing data transfer is FLCTL[20]. When FLCTL[20] is high, it means that data transfer is finished.

### 2. Master Mode

When MTRANS\_CFG[2]=1, master is selected. i. e. , flash data load/store between internal memory and external memory is through master interface.

In this mode, software should initialize page data and spare data into external memory, and set their addresses in MTRANS\_SADDR0 and MTRANS\_SADDR1 respectively before starting flash program operation. Similarly, software should configure MTRANS\_SADDR0 and MTRANS\_SADDR1 respectively before starting flash read operation and could read data from addresses in MTRANS\_SADDR0 and MTRANS\_SADDR1 after NandC transfer finish.

In this mode, MTRANS\_CFG, MTRANS\_SADDR0 and MTRANS\_SADDR1 are used. The transfer codeword number is determined by FLCTL[26:22], and the maximum number is 16.

The judgment condition for finishing data transfer is FLCTL[20]. When FLCTL[20] is high, it means that data transmission is finished.

When MTRANS\_CFG[2]=1, page data and spare data are stored in the continuous space of external memory respectively.

For page data, source address is named Saddr0, specified in MTRANS\_SADDR0. The space can be divided into many continuous units, and the unit size(named PUnit) is 1024 bytes or 512 bytes determined by FLCTL[21] and FLCTL[11]:

- a. when FLCTL[11]=0, PUnit is always equal to 1024 bytes
- b. when FLCTL[11]=1 and FLCTL[21]=0, PUnit is equal to 1024 bytes
- c. when FLCTL[11]=1 and FLCTL[21]=1, PUnit is equal to 512 bytes

For spare data, source address is named Saddr1, specified in MTRANS\_SADDR1. The space can be divided into many continuous units, and the unit size(named SUnit) is 64 bytes or 128 bytes determined by BCHCTL[18], FLCTL[11] and FLCTL[21]:

- a. When FLCTL[11]=0 and BCHCTL[18]=0, SUnit is equal to 64 bytes
- b. When FLCTL[11]=0 and BCHCTL[18]=1, SUnit is equal to 128 bytes
- c. When FLCTL[11]=1 and FLCTL[21]=0, SUnit is equal to 128 bytes
- d. When FLCTL[11]=1 and FLCTL[21]=1, SUnit is equal to 64 bytes

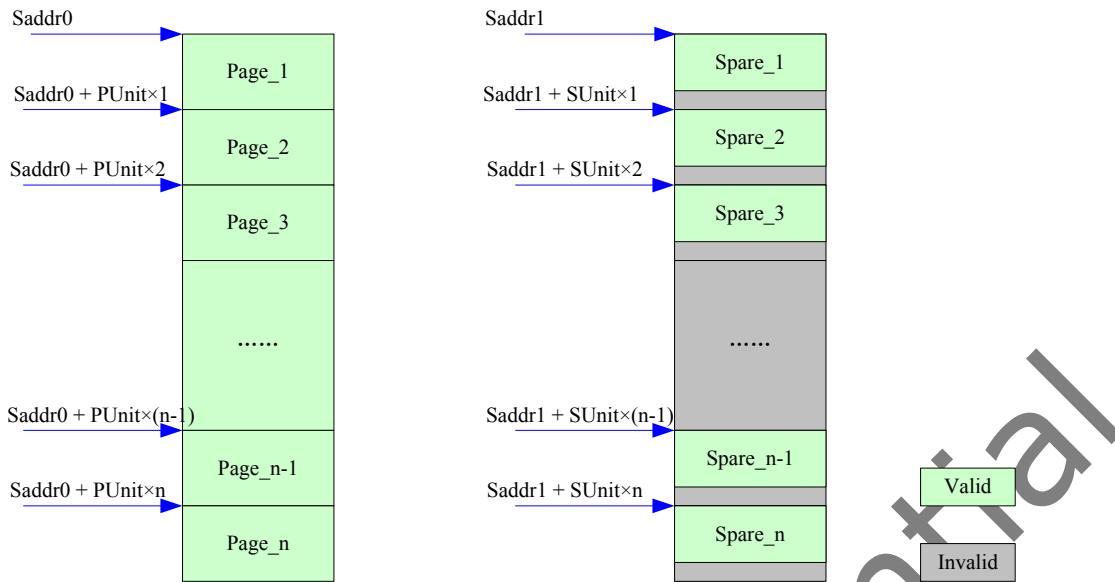


Fig. 10-2 NandC Address Assignment

The detailed format for page data and spare data in every unit is shown in following figures.

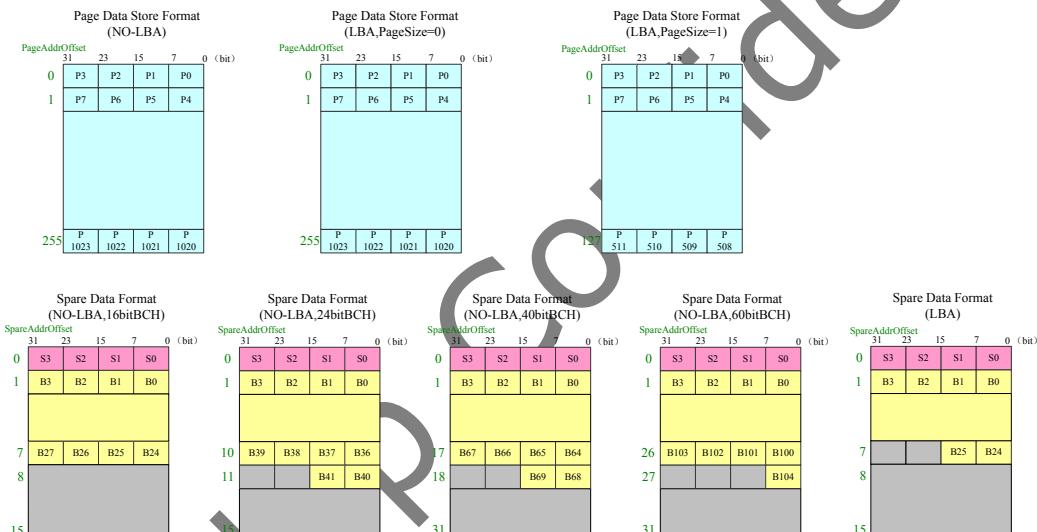


Fig. 10-3 NandC Data Format

### 10.6.3 BchPage Application

BCHCTL[16] determines whether codeword size for page data is 1024 bytes or 512 bytes when FLCTL[11] is 0.

#### 1. 1024bytes

When BCHCTL[16]=0, BchPage=0, hardware needs to write 1024 bytes page data and spare data into flash or read 1024 bytes page data and spare data from flash. All the 1024 bytes page data and spare data are encoded when writing or decoded when reading.

#### 2. 512bytes

When BCHCTL[16]=1, BchPage=1, hardware needs to write 512 bytes page data and spare data into flash or read 512 bytes page data and spare data from flash.

In this mode, the page data unit size for BCH encoder and BCH decoder still is 1024byte. So to support BCH encoder and decoder, software should configure page data as follows: 1th~512th bytes are invalid data which must be stuffed with 0xff, 513th~1024th bytes are valid page data.

However, Randomizer function is not supported under this condition.

### 10.6.4 PageSize/SpareSize Application

FLCTL[21] determines whether the codeword size is 1024 bytes or 512 bytes when FLCTL[11] is 1.

#### 1. Big Page

When FLCTL[11]=0(LbaEn=0), the flash to be operated is Raw NAND Flash. Every codeword size is 1024 bytes and FLCTL[21] should always be set to 0, and the PageStep in external memory is 1024 bytes if bus mode is master mode.

At this mode, the spare size and SpareStep in external memory are determined by BCH Mode as follows:

BCH Mode=16bitBCH: spare size=(28+4)bytes , SpareStep=64bytes

BCH Mode=24bitBCH: spare size=(42+4)bytes , SpareStep=64bytes

BCH Mode=40bitBCH: spare size=(70+4)bytes , SpareStep=128bytes

BCH Mode=60bitBCH: spare size=(105+4)bytes, SpareStep=128bytes

## **2. Small Page**

When FLCTL[11]=1, LbaEn=1, the flash to be operated is Managed NAND Flash. Every codeword size could be 1024 bytes or 512 bytes according to FLCTL[21]. If FLCTL[21]=0, codeword size is 1024 bytes, PageStep in external memory is 1024 bytes, and SpareStep is 128bytes. If FLCTL[21]=1, codeword size is 512 bytes, PageStep in external memory is 512 bytes, and SpareStep is 64 bytes.

At this mode, the spare size is configured in FLCTL[18:12], and the max available number is 109.

In the summary, the total data size in every codeword for flash or for software including page data and spare data, is determined by BCHCTL[16], FLCTL[11], FLCTL[21], BCHCTL[4], BCHCTL[18]. Their relationship is shown as follows.

Table 10-4 NandC Page/Spare size for flash

page/spare size for software	page size /codeword	spare size /codeword
FLCTL[11]=0	16bitECC	1024 byte (4+28)byte
	24bitECC	1024 byte (4+42)byte
	40bitECC	1024 byte (4+70)byte
	60bitECC	1024 byte (4+105)byte
FLCTL[11]=1	FLCTL[21]=0	1024 byte FLCTL[18:12]
	FLCTL[21]=1	512 byte FLCTL[18:12]

Notes: that "page/spare size for flash" means that hardware should transfer these numbers of bytes in every codeword to or from flash.

## **10.6.5 Randomizer Application**

RANDMZ\_CFG[31] determines whether randomizer is enable or not. When RANDMZ\_CFG[31] equals to 1, randomizer is active. Data should be scrambled before written into flash, and descrambled after read from flash.

RANDMZ\_CFG[30] determines the randomizer polynomial.

When RANDMZ\_CFG[30]=0, Polynomial= $1+x^{18}+x^{23}$

When RANDMZ\_CFG[30]=1, Polynomial= $1+x^{14}+x^{15}$

RANDMZ\_CFG[22:0] is the seed for randomizer. It should be ensured that data in the same page should have the same randomizer polynomial and randomizer seed when in flash program or flash read operation.

The data unit for randomizer is one codeword(data+spare).

However, Randomizer is just available for data transfer by internal DMA mode, but not by for bypass mode. Furthermore, it should not be enable if BCHCTL[16]=0 (BchPage=512bytes).

## **10.6.6 DLL Application**

When Toggle Flash or ONFI Synchronous Flash interface is active, DLL should be used to adjust DQS input with DQ when reading flash.

There are 2 registers for DLL configuration(DLL\_CFG\_REG0 and DLL\_CFG\_REG1), and 1 register for DLL status(DLL\_OBS\_REG0).

The usage guide is as follows:

If bypass mode is used, you should set *dll\_bypass* in DLL\_CFG\_REG1[1] to 1, and set *dll\_dqs\_dly\_bypass* in DLL\_CFG\_REG0[23:16] to determine the dll element number needed. And then set *dll\_start* in DLL\_CFG\_REG1[0] to 1 to start the DLL.

If auto adjusting is used, you should set *dll\_bypass* in DLL\_CFG\_REG1[1] to 0, and set the *dll\_start\_point* in DLL\_CFG\_REG0[7:0] and *dll\_incr* in DLL\_CFG\_REG1[11:4]. You also should

set the adjusting mode *dll\_qtren* in DLL\_CFG\_REG1[3:2] to compute the dll element number needed. If *dll\_qtren*=2'b00, the dll element number is determined by *dll\_dqs\_dly* in DLL\_CFG\_REG0[15:8]; otherwise, it is 1/4 or 1/8 of the total number of dll elements used for *dll\_qtren*=2'b01 or *dll\_qtren*=2'b10 separately. The last step is to set *dll\_start* in DLL\_CFG\_REG1[0] to 1 to start the DLL.

If you want to monitor the dll working status, you could read DLL\_OBS\_REG0. If DLL\_OBS\_REG0[0]=0, it means that DLL is not locked, and still in detecting status. Otherwise, it means that DLL is locked, and *dll\_lock\_value* in DLL\_OBS\_REG0[8:1] is the total number of dll elements used, *dll\_dqs\_delay\_value* in DLL\_OBS\_REG0[16:9] is the total number of DQS delay used.

### 10.6.7 NandC Interrupt Application

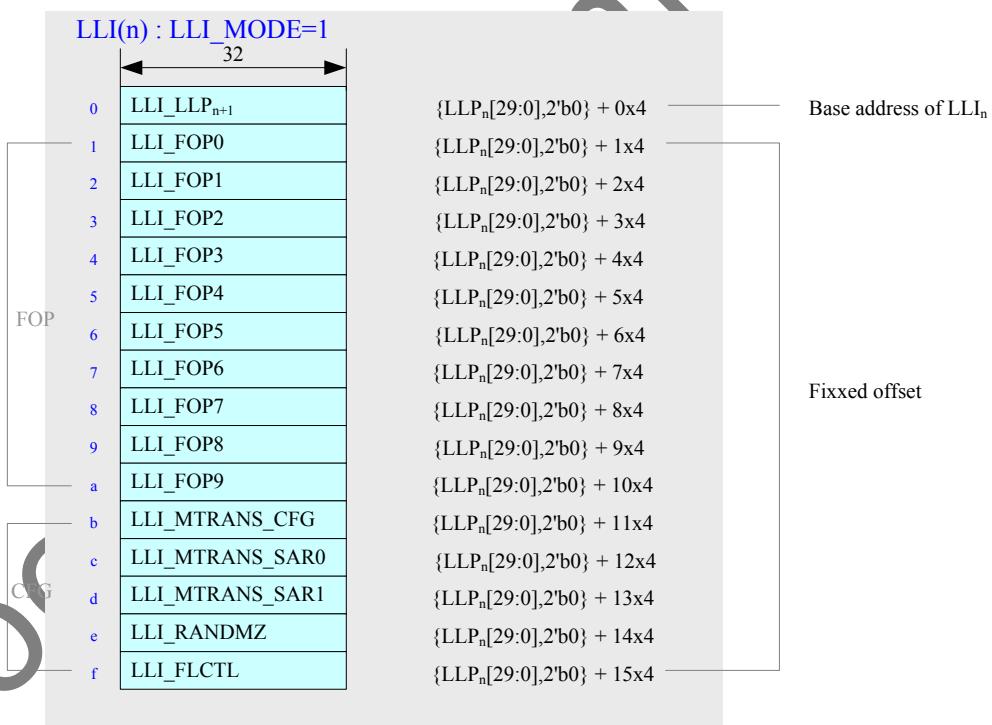
NandC has 1 interrupt output signal and 4 interrupt sources: dma finish interrupt source, flash ready interrupt source, bch error interrupt source, bch fail interrupt source. When one or more of these interrupt source are enabled, NandC interrupt is asserted if one or more interrupt source is high. Software can determine the interrupt source by reading INTST and clear interrupt by writing corresponding bit in INTCLR.

### 10.6.8 LLP Application

LLP is used in NandC to store and execute instruction groups configured in external memory by software. When LLPCTL[0]=1, LLP is active, NandC will load instruction groups stored in {LLPCTL[31:6], 6'h0} and execute them. Next instruction groups should not be loaded until current instruction execution finished.

#### 1. LLP Structure

The structure of LLP is shown as follows:



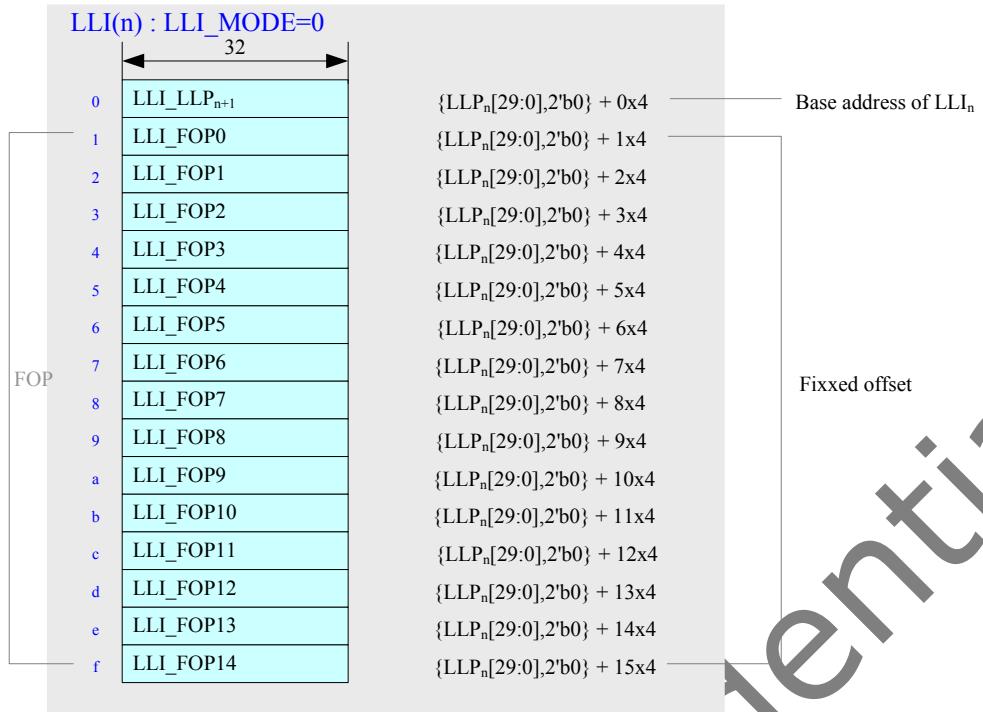
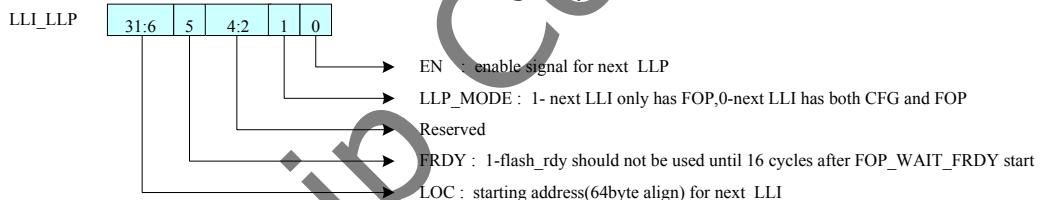


Fig. 10-4 NandC LLP Data Format

LLI\_MODE is determined by LLPCTL[1]. If current operation is flash program or flash read, then LLI\_MODE=1 is need; otherwise, LLI\_MODE=0 is workable.  
In addition, you could do more than one flash operation in one LLP group, but you should not separate one flash operation into two LLI groups.

## 2. LLI Format

- a. LLI\_LL\_Pn+1 stores the address for next LLI group data



- b. LLI\_FOP0~LLI\_FOP14 store the flash operation instruction

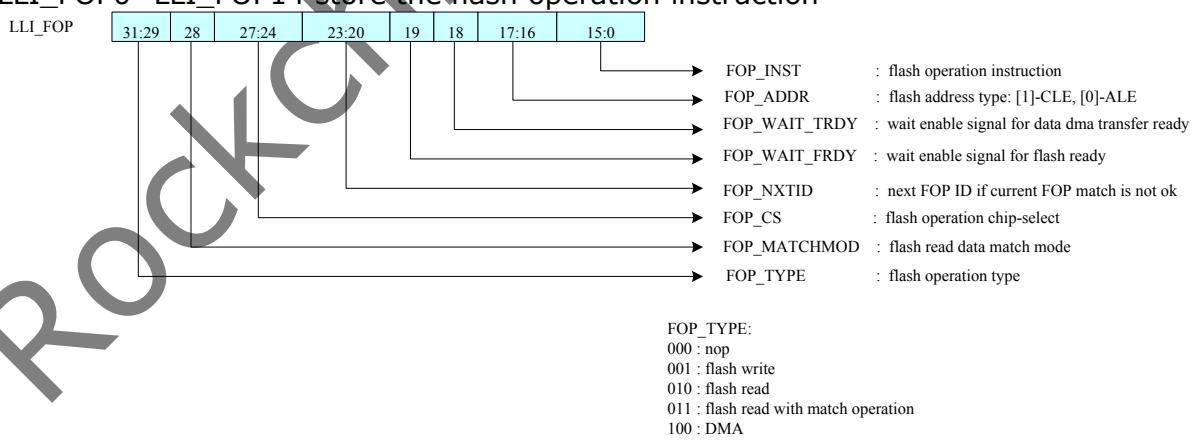


Fig. 10-5 NandC LLI Data Format

When FOP\_TYPE=3'b011, match operation is active, and the PATTERN is LLI\_FOP[15:0]. It is matched when "RDATA|PATTERN=PATTERN" with FOP\_MATCHMOD=0, or when "RDATA&PATTERN=PATTERN" with FOP\_MATCHMOD=1.

- c. LLI\_MTRANS\_CFG/LLI\_MTRANS\_SADDR0/LLI\_MTRANS\_SADDR1/LLI\_RANDMZ/LLI\_FLCTL store the configuration for MTRANS\_CFG/MTRANS\_SADDR0/MTRANS\_SADDR1/RANDMZ/FLCTL.

## 3. LLP Working Mode

There are two working modes for LLP:

- Normal mode: LLPCTL[0] is kept to 1 until all LLP loading and executing finished. Software can monitor the progress by LLPSTAT[31:6], LLPSTAT[0].
- Pause mode: LLPCTL[0] is changed from 1 to 0 during LLP loading or LLP executing. NandC should not stop working until current LLP executing finished. Software can monitor the progress by LLPSTAT[31:6], LLPSTAT[0].

### 10.6.9 FIFO Application

FIFO in NandC is used to store command, address and data temporarily that are intend to write to the external flash. The FIFO is 32-bit wide and 8-location deep, user can access it by configuring the NandC to work on bypass(FLCTL[3]=1) and fifo enable(FLCTL[30]=1) mode. The format of data written into fifo is as follows.

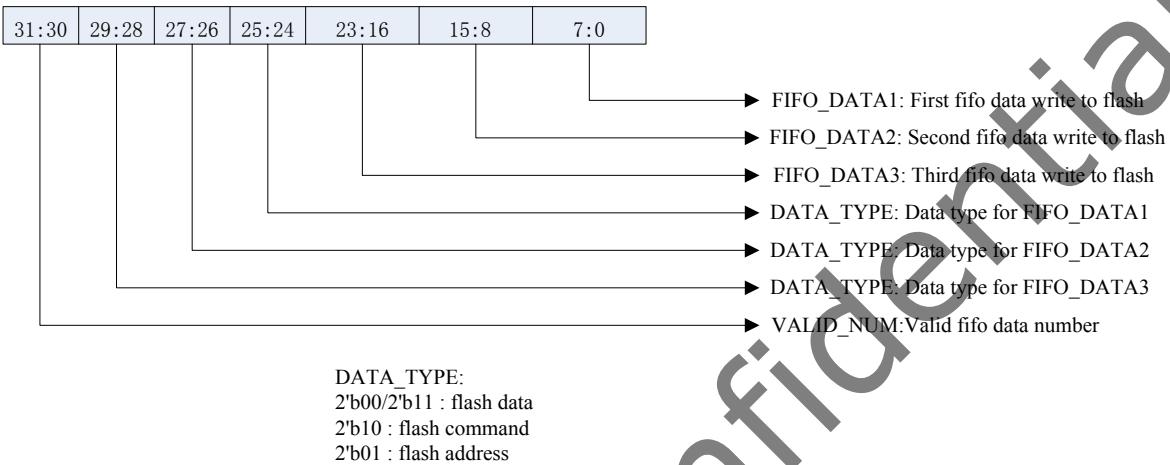


Fig. 10-6 FIFO Data Format

Command, address or data write to external flash are pushed into fifo first and pop up automatically if the fifo is not empty. Then FIFO\_DATA1, FIFO\_DATA2 and FIFO\_DATA3 are written to flash by NandC in order. The method to determine the end of write operation is to read register FMCTL. If FMCTL[10] is set , it means that the fifo is empty and flash write operation is over.

## Chapter 11 Timer

### 11.1 Overview

Timer is a programmable timer peripheral. This component is an APB slave device. In RK3228 there are 6 Timers and 2 Secure Timers(STimer).

Timer5 and STimer0~1 count up from zero to a programmed value and generate an interrupt when the counter reaches the programmed value.

Timer0~4 count down from a programmed value to zero and generate an interrupt when the counter reaches zero.

Timer supports the following features:

- Timer0~5 is used for no-secure, STimer0~1 is used for secure.
- Two operation modes: free-running and user-defined count.
- Maskable for each individual interrupt.

### 11.2 Block Diagram

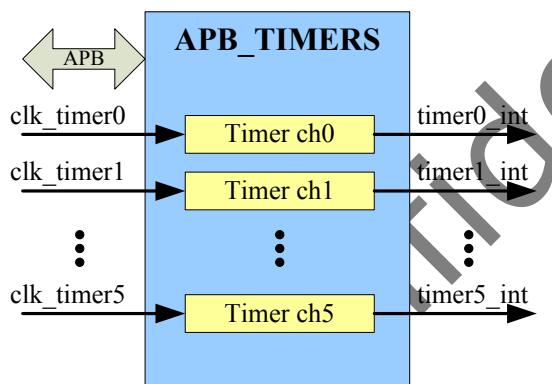


Fig. 11-1 Timer Block Diagram

The above figure shows the architecture of the APB timers (include six programmable timer channels) that in the bus subsystem. The Stimers that in the bus subsystem only include two programmable timer channels.

### 11.3 Function Description

#### 11.3.1 Timer clock

TIMER0~ TIMER5 and STIMER0~1 are in the pd\_bus subsystem. The timer clock is 24MHz OSC.

#### 11.3.2 Programming sequence

1. Initialize the timer by the `TIMERn_CONTROLREG` ( $0 \leq n \leq 5$ ) register:
  - Disable the timer by writing a "0" to the timer enable bit (bit 0). Accordingly, the `timer_en` output signal is de-asserted.
  - Program the timer mode—user-defined or free-running—by writing a "0" or "1" respectively, to the timer mode bit (bit 1).
  - Set the interrupt mask as either masked or not masked by writing a "0" or "1" respectively, to the timer interrupt mask bit (bit 2).
2. Load the timer count value into the `TIMERn_LOAD_COUNT1` ( $0 \leq n \leq 5$ ) and `TIMERn_LOAD_COUNT0` ( $0 \leq n \leq 5$ ) register.
3. Enable the timer by writing a "1" to bit 0 of `TIMERn_CONTROLREG` ( $0 \leq n \leq 5$ ).

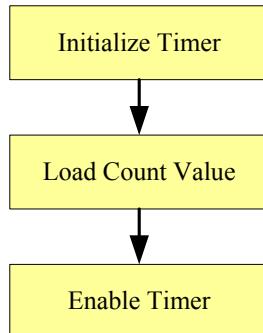


Fig. 11-2 Timer Usage Flow

### 11.3.3 Loading a timer count value

For the descending Timers(Timer0~4).The initial value for each timer—that is, the value from which it counts down—is loaded into the timer using the load count register (TIMERn\_LOAD\_COUNT1 and TIMERn\_LOAD\_COUNT0). Two events can cause a timer to load the initial value from its load count register:

- Timer is enabled after reset or disabled.
- Timer counts down to 0, when timer is configured into free-running mode.

For the incremental Timers(Timer5 and STimer0~1).The initial value for each timer is zero. The count register will count up to the value loaded in the register TIMERn\_LOAD\_COUNT1 and TIMERn\_LOAD\_COUNT0. Two events can cause a timer to load zero:

- Timer is enabled after reset or disabled.
- Timer counts up to the value stored in TIMERn\_LOAD\_COUNT1 and TIMERn\_LOAD\_COUNT0, when timer is configured into free-running mode.

### 11.3.4 Timer mode selection

- User-defined count mode – Timer loads TIMERn\_LOAD\_COUNT1 and TIMERn\_LOAD\_COUNT0 registers (for descending timers) or zero (for incremental timers ) as initial value. When the timer counts down to 0 (for descending timers) or counts up to the value in TIMERn\_LOAD\_COUNT1 and TIMERn\_LOAD\_COUNT0 (for incremental timers ),it will not automatically reload the count register. User need to disable timer firstly and follow the programming sequence to make timer work again.
- Free-running mode – Timer loads the TIMERn\_LOAD\_COUNT1 and TIMERn\_LOAD\_COUNT0(for descending timers) or zero (for incremental timers)register as initial value. Timer will automatically reload the count register, when timer counts down to 0 (for descending timers) or counts up to the value in TIMERn\_LOAD\_COUNT1 and TIMERn\_LOAD\_COUNT0 (for incremental timers).

## 11.4 Register Description

This section describes the control/status registers of the design. Software should read and write these registers using 32-bits accesses.

### 11.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
TIMER_n_LOAD_COUNT0	0x0000	W	0x00000000	Timer n Load Count Register
TIMER_n_LOAD_COUNT1	0x0004	W	0x00000000	Timer n Load Count Register
TIMER_n_CURRENT_VALUE0	0x0008	W	0x00000000	Timer n Current Value Register
TIMER_n_CURRENT_VALUE1	0x000c	W	0x00000000	Timer n Current Value Register
TIMER_n_CONTROLREG	0x0010	W	0x00000000	Timer n Control Register
TIMER_n_INTSTATUS	0x0018	W	0x00000000	Timer Interrupt Stauts Register

Notes:Size:**B**- Byte (8 bits) access, **HW**- Half WORD (16 bits) access, **W**-WORD (32 bits) access

### 11.4.2 Detail Register Description

#### TIMER\_n\_LOAD\_COUNT0

Address: Operational Base + offset (0x00)

Timer n Load Count Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	load_count_low bits Low 32 bits value to be loaded into Timer n. This is the value from which counting commences.

**TIMER\_n\_LOAD\_COUNT1**

Address: Operational Base + offset (0x04)

Timer n Load Count Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x00000000	load_count_high bits High 32 bits value to be loaded into Timer n. This is the value from which counting commences.

**TIMER\_n\_CURRENT\_VALUE0**

Address: Operational Base + offset (0x08)

Timer n Current Value Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	current_cnt_lowbits Low 32 bits of current value of timer n.

**TIMER\_n\_CURRENT\_VALUE1**

Address: Operational Base + offset (0x0c)

Timer n Current Value Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	current_cnt_highbits High 32 bits of current value of timer n.

**TIMER\_n\_CONTROLREG**

Address: Operational Base + offset (0x10)

Timer n Control Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	RO	0x0	reserved
2	RW	0x0	int_en Timer interrupt mask 0: mask 1: not mask
1	RW	0x0	timer_mode Timer mode. 0: free-running mode 1: user-defined count mode
0	RW	0x0	timer_en Timer enable. 0: disable 1: enable

**TIMER\_n\_INTSTATUS**

Address: Operational Base + offset (0x18)

Timer Interrupt Stauts Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:1	RO	0x0	reserved
0	W1C	0x0	int_pd This register contains the interrupt status for timer n. Write 1 to this register will clear the interrupt.

**11.5 Application Notes**

In the chip, the timer\_clk is from 24MHz OSC, asynchronous to the pclk. When user disables the timer enables bit (bit 0 of TIMERn\_CONTROLREG ( $0 \leq n \leq 5$ )), the timer\_en output signal is de-asserted, and timer\_clk will stop. When user enables the timer, the timer\_en signal is asserted and timer\_clk will start running.

The application is only allowed to re-config registers when timer\_en is low.

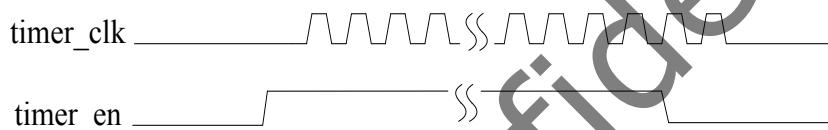


Fig. 11-3 Timing between timer\_en and timer\_clk

Please refer to function description section for the timer usage flow.

## Chapter 12 Generic Interrupt Controller (GIC)

### 12.1 Overview

The generic interrupt controller (GIC400) in this device has two interfaces, the distributor interface connects to the interrupt source, and the CPU interface connects to Cortex-A7.

It supports the following features:

- Supports 128 hardware interrupt inputs
- Masking of any interrupts
- Prioritization of interrupts
- Distribution of the interrupts to the target Cortex-A7 processor(s)
- Generation of interrupts by software
- Supports Security Extensions

### 12.2 Block Diagram

The generic interrupt controller comprises with:

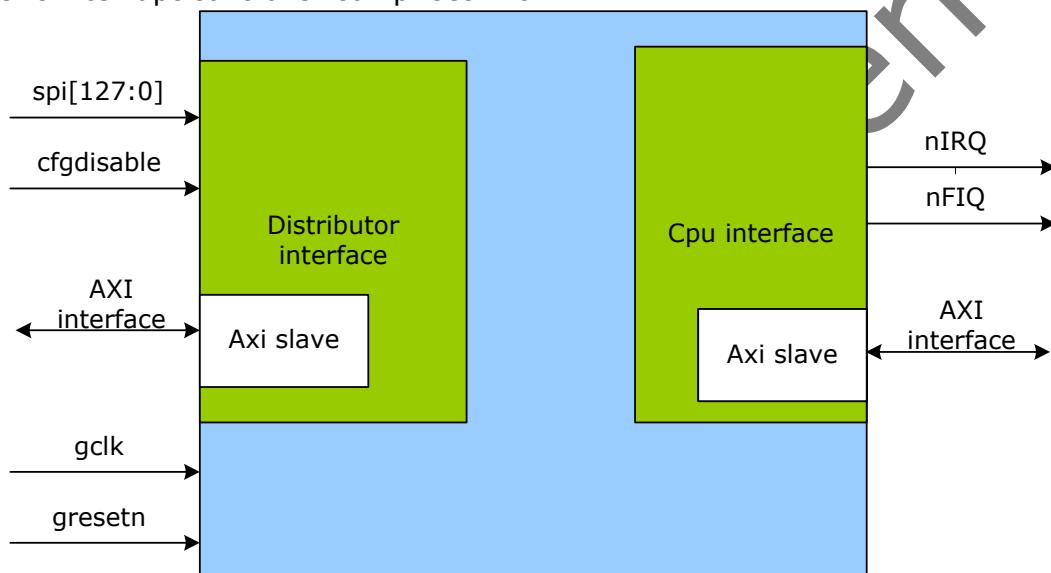


Fig. 12-1 GIC Block Diagram

### 12.3 Function Description

This GIC architecture splits logically into a Distributor block and one CPU interface block, as Figure 12-1 shows.

#### Distributor

This performs interrupt prioritization and distribution to the CPU interface that connect to the processor in the system.

#### CPU interface

CPU interface performs priority masking and preemption handling for a connected processor in the system.

#### 12.3.1 The Distributor

The Distributor centralizes all interrupt sources, determines the priority of each interrupt, and for CPU interface dispatches the interrupt with the highest priority to the interface for priority masking and preemption handling.

The Distributor provides a programming interface for:

- Globally enabling the forwarding of interrupts to the CPU interface
- Enabling or disabling each interrupt
- Setting the priority level of each interrupt
- Setting the target processor list of each interrupt
- Setting each peripheral interrupt to be level-sensitive or edge-triggered
- If the GIC implements the Security Extensions, setting each interrupt as either

- Secure or Non-secure
- Sending a Software-generated interrupt (SGI) to processor.
- Visibility of the state of each interrupt
- A mechanism for software to set or clear the pending state of a peripheral interrupt.

**Interrupt ID**

Interrupts from sources are identified using ID numbers. CPU interface can see up to 160 interrupts.

The GIC assigns interrupt these 128 ID numbers as follows:

- Interrupt numbers ID32-ID127 are used for SPIs(shared peripheral interrupts).
- ID0-ID15 are used for SGI.
- ID16-ID31 are used for Private peripheral interrupt (PPI).

The GIC architecture reserves interrupt ID numbers 1022-1023 for special purposes.

**ID1022**

The GIC returns this value to a processor in response to an interrupt acknowledge only when the following apply:

- The interrupt acknowledge is a Secure read
- The highest priority pending interrupt is Non-secure
- The AckCtl bit in the Secure ICCICR is set to 0
- The priority of the interrupt is sufficient for it to be signalled to the processor.

Interrupt ID 1022 informs secure software that there is a Non-secure interrupt of sufficient priority to be signalled to the processor, that must be handled by Non-secure software. In this situation the secure software might alter its schedule to permit Non-secure software to handle the interrupt, to minimize the interrupt latency.

**ID1023**

This value is returned to a processor, in response to an interrupt acknowledge, if there is no pending interrupt with sufficient priority for it to be signalled to the processor.

On a processor that implements the Security Extensions, Secure software treats values of 1022 and 1023 as spurious interrupts.

### **12.3.2 CPU interface**

CPU interface block provides the interface for a processor that operates with the GIC. CPU interface provides a programming interface for:

- Enabling the signal of interrupt requests by the CPU interface
- Acknowledging an interrupt
- Indicating completion of the processing of an interrupt
- Setting an interrupt priority mask for the processor
- Defining the preemption policy for the processor
- Determining the highest priority pending interrupt for the processor.

When enabled, CPU interface takes the highest priority pending interrupt for its connected processor and determines whether the interrupt has sufficient priority for it to signal the interrupt request to the processor.

To determine whether to signal the interrupt request to the processor the CPU interface considers the interrupt priority mask and the preemption settings for the processor. At any time, the connected processor can read the priority of its highest priority active interrupt from a CPU interface register.

The processor acknowledges the interrupt request by reading the CPU interface Interrupt Acknowledge register. The CPU interface returns one of:

The ID number of the highest priority pending interrupt, if that interrupt is of sufficient priority to generate an interrupt exception on the processor. This is the normal response to an interrupt acknowledge.

Exceptionally, an ID number that indicates a spurious interrupt.

When the processor acknowledges the interrupt at the CPU interface, the Distributor changes the status of the interrupt from pending to either active, or active and pending. At this point the CPU interface can signal another interrupt to the processor, to preempt interrupts that are active on the processor. If there is no pending interrupt with sufficient priority for signaling to the processor, the interface de-asserts the interrupt request signal to the processor.

When the interrupt handler on the processor has completed the processing of an interrupt, it writes to the CPU interface to indicate interrupt completion. When this happens, the

distributor changes the status of the interrupt either:

- From active to inactive
- From active and pending to pending.

### 12.3.3 Interrupt handling state machine

The distributor maintains a state machine for each supported interrupt on CPU interface. Following figure shows an instance of this state machine, and the possible state transitions.

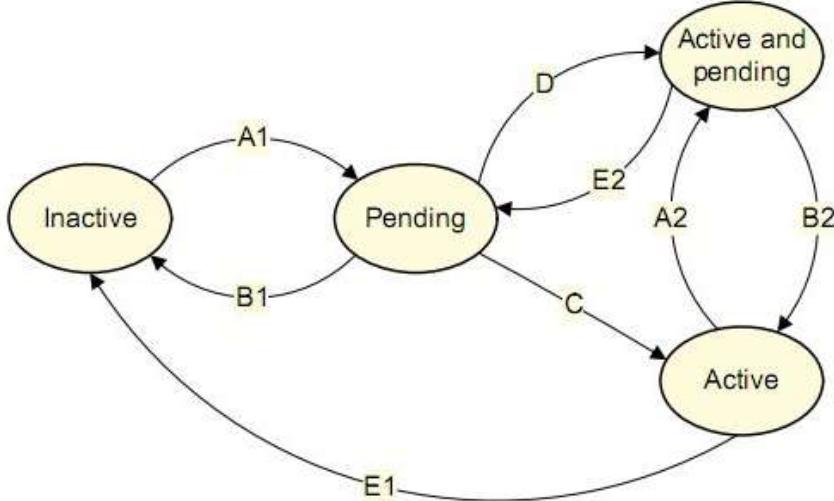


Fig. 12-2 GIC Interrupt handling state machine

#### Transition A1 or A2, add pending status

For an SGI:

- Occurs on a write to an ICDSGIR that specifies the processor as a target.
- If the GIC implements the Security Extensions and the write to the ICDSGIR is Secure, the transition occurs only if the security configuration of the specified SGI, for the CPU interface, corresponds to the ICDSGIR.SATT bit value.

For an SPI, occurs if either:

- a peripheral asserts an interrupt signal
- software writes to an ICDISPR.

#### Transition B1 or B2, remove pending status

Not applicable to SGIs:

- a pending SGI must transition through the active state, or reset, to remove its pending status.
- an active and pending SGI must transition through the pending state, or reset, to remove its pending status.

For an SPI, occurs if either:

- the level-sensitive interrupt is pending only because of the assertion of an input signal, and that signal is deasserted
- the interrupt is pending only because of the assertion of an edge-triggered interrupt signal, or a write to an ICDISPR, and software writes to the corresponding ICDICPR.

#### Transition C

If the interrupt is enabled and of sufficient priority to be signalled to the processor, occurs when software reads from the ICCIAR.

#### Transition D

For an SGI, occurs if the associated SGI is enabled and the Distributor forwards it to the CPU interface at the same time that the processor reads the ICCIAR to acknowledge a previous instance of the SGI. Whether this transition occurs depends on the timing of the read of the ICCIAR relative to the reforwarding of the SGI.

For an SPI:

- Occurs if all the following apply:
  - The interrupt is enabled.
  - Software reads from the ICCIAR. This read adds the active state to the interrupt.

- For a level-sensitive interrupt, the interrupt signal remains asserted. This is usually the case, because the peripheral does not deassert the interrupt until the processor has serviced the interrupt.
- For an edge-triggered interrupt, whether this transition occurs depends on the timing of the read of the ICCIAR relative to the detection of the reassertion of the interrupt. Otherwise the read of the ICCIAR causes transition C, possibly followed by transition A2.

## 12.4 Register Description

### 12.4.1 GIC Distributor interface register summary

Name	Offset	Size	Reset	Description
GICD_ICDDCR	0x000	W	0x0	Distributor Control Register
GICD_ICDICTR	0x004	W	-	Interrupt Controller Type Register
GICD_ICDIIDR	0x008	W	-	Distributor Implementer Identification Register
GICD_ICDISR	0x080	W	-	Interrupt Security Registers
-	-	-	-	reserved
GICD_ICDISER	0x100-0x17C	W	-	Interrupt Set-Enable Registers
GICD_ICDICER	0x180-0x1FC	W	-	Interrupt Clear-Enable Registers
GICD_ICDISPR	0x200-0x27C	W	0x0	Interrupt Set-Pending Registers
GICD_ICDICPR	0x280-0x2FC	W	0x0	Interrupt Clear-Pending Registers
GICD_ICDABR	0x300-0x37C	W	0x0	Active Bit Registers
-	-	-	-	reserved
GICD_ICDIPR	0x400-0x7F8	B	0x0	Interrupt Priority Registers
-	-	-	-	reserved
GICD_ICDIPTR	0x800-0x81C	B	-	Interrupt Processor Targets
-	-	-	-	reserved
GICD_ICDICFR	0xC00-0xCFC	W	-	Interrupt Configuration Registers
-	-	-	-	Reserved
GICD_ICPPISR	-	W	-	PPI Status Register
GICD_ICSPISR	0xD04-0xD1C	W	-	SPI Status Registers
-	-	-	-	Reserved
GICD_ICDSGIR	0xF00	W	-	Software Generated Interrupt Register

Notes:

Size: **B** – Byte (8 bits) access, **HW** – Half WORD (16 bits) access, **W** – WORD (32 bits) access

### 12.4.2 GIC Distributor interface detail register description

#### GICD\_ICDDCR

Address: Operational Base+0x0

Distributor Control Register

Bit	Attr	Reset Value	Description
31:1	-	-	reserved
1	RW	0x0	1'b0: disables all Non-secure interrupts control bits in the distributor from changing state because of any external stimulus change that occurs on the corresponding SPI or PPI signals 1'b1: enables the distributor to update register locations for Non-secure interrupts

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
0	RW	0x0	1'b0: disables all Secure interrupt control bits in the distributor from changing state because of any external stimulus change that occurs on the corresponding SPI or PPI signals. 1'b1: enables the distributor to update register locations for Secure interrupts.

**GICD\_ICDICTR**

Address: Operational Base+0x4  
Interrupt Controller Type Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:11	-	-	reserved
15:11	R	0x0	Returns the number of Lockable Shared Peripheral Interrupts (LSPIs) that the controller contains. The encoding is: 5'b11111: 31 LSPIs, that are the interrupts of IDs 32-62. When CFGSDISABLE is HIGH, the interrupt controller prevents writes to any register location that controls the operating state of an LSPI.
10	R	0x1	Indicates whether the GIC implements the Security Extensions. 1'b0: Security Extensions not implemented. 1'b1: Security Extensions implemented
9:8	-	-	reserved
7:5	R	0x0	Indicates the number of implemented CPU interface. The number of implemented CPU interface is one more than the value of this field, for example if this field is 3'b011, there are four CPU interface. In this product ,only one CPU interface is implemented.
4:0	R	0x2	Indicates the mAXIum number of interrupts that the GIC supportsa. If the value of this field is N, the mAXIum number of interrupts is 32(N+1). The interrupt ID range is from 0 to one less than the number of IDs. For example: 5'b00011: Up to 128 interrupt lines, interrupt IDs 0-127. The mAXIum number of interrupts is 1020 (5'b11111).

**GICD\_ICDIIDR**

Address: Operational Base+0x8  
Distributor Implementer Identification Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	R	0x0	product identifier.
23:20	-	-	reserved
19:16	R	0x0	variant number. Typically, this field is used to distinguish product variants, or major revisions of a product
15:12	R	0x0	revision number. Typically, this field is used to distinguish minor revisions of a product
11:0	R	0x0	Contains the JEP106 code of the company that implemented the GIC Distributor:a Bits [11:8]: The JEP106 continuation code of the implementer. Bits [7]: Always 0. Bits [6:0]: The JEP106 identity code of the implementer.

**GICD\_ICDISR**

Address: Operational Base+0x80

Interrupt Security Registers

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x0	For each bit: 1'b0: The corresponding interrupt is Secure. 1'b1: The corresponding interrupt is Non-secure.

For interrupt ID N, when DIV and MOD are the integer division and modulo operations:

- the corresponding ICDISR number, M, is given by  $M = N \text{ DIV } 32$
- the offset of the required ICDISR is  $(0x080 + (4*M))$
- the bit number of the required Security status bit in this register is  $N \text{ MOD } 32$ .

**GICD\_ICDISER**

Address: Operational Base+0x100

Interrupt Set-Enable Registers

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW		For SPIs, for each bit: Reads 1'b0: The corresponding interrupt is disabled. 1'b1: The corresponding interrupt is enabled. Writes 1'b0: Has no effect. 1'b1: Enables the corresponding interrupt. A subsequent read of this bit returns the value 1.

For interrupt ID N, when DIV and MOD are the integer division and modulo operations:

- the corresponding ICDISER number, M, is given by  $M = N \text{ DIV } 32$
- the offset of the required ICDISER is  $(0x100 + (4*M))$
- the bit number of the required Set-enable bit in this register is  $N \text{ MOD } 32$ .

**GICD\_ICDICER**

Address: Operational Base+0x180

Interrupt Clear-Enable Registers

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x0	For SPI, for each bit: Reads 1'b0: The corresponding interrupt is disabled. 1'b1: The corresponding interrupt is enabled. Writes 1'b0: Has no effect. 1'b1: Disables the corresponding interrupt. A subsequent read of this bit returns the value 0.

For interrupt ID N, when DIV and MOD are the integer division and modulo operations:

- the corresponding ICDICER number, M, is given by  $M = N \text{ DIV } 32$
- the offset of the required ICDICER is  $(0x180 + (4*M))$
- the bit number of the required Clear-enable bit in this register is  $N \text{ MOD } 32$ .

**GICD\_ICDISPR**

Address: Operational Base+0x200

Interrupt Set-Pending Registers

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RW	0x0	For each bit: Reads 1'b0: The corresponding interrupt is not pending on any processor. 1'b1: <ul style="list-style-type: none"> <li>• For SGIs, the corresponding interrupt is pending on this processor.</li> </ul>

Bit	Attr	Reset Value	Description
			<ul style="list-style-type: none"> <li>For SPIs, the corresponding interrupt is pending on at least one processor.</li> </ul> <p>Writes For SPIs:</p> <p>1'b0: Has no effect.</p> <p>1'b1: The effect depends on whether the interrupt is edge-triggered or level-sensitive:</p> <p>Edge-triggered</p> <p>Changes the status of the corresponding interrupt to:</p> <ul style="list-style-type: none"> <li>pending if it was previously inactive</li> <li>active and pending if it was previously active.</li> </ul> <p>Has no effect if the interrupt is already pending.</p> <p>Level sensitive</p> <p>If the corresponding interrupt is not pending, changes the status of the corresponding interrupt to:</p> <ul style="list-style-type: none"> <li>pending if it was previously inactive</li> <li>active and pending if it was previously active.</li> </ul> <p>If the interrupt is already pending:</p> <ul style="list-style-type: none"> <li>because of a write to the ICDISPR, the write has no effect</li> <li>because the corresponding interrupt signal is asserted, the write has no effect on the status of the interrupt, but the interrupt remains pending if the interrupt signal is deasserted.</li> </ul>

For interrupt ID N, when DIV and MOD are the integer division and modulo operations:

- the corresponding ICDISPR number, M, is given by  $M = N \text{ DIV } 32$
- the offset of the required ICDISPR is  $(0x200 + (4*M))$
- the bit number of the required Set-pending bit in this register is  $N \text{ MOD } 32$ .

### GICD\_ICDICPR

Address: Operational Base+0x280

Interrupt Clear-Pending Registers

Bit	Attr	Reset Value	Description
31:0	RW	0x0	<p>For each bit:</p> <p>Reads</p> <p>1'b0: The corresponding interrupt is not pending on any processor</p> <p>1'b1:</p> <ul style="list-style-type: none"> <li>For SGIs, the corresponding interrupt is pending on this processor.</li> <li>For SPIs, the corresponding interrupt is pending on at least one processor.</li> </ul> <p>Writes</p> <p>For SPIs:</p> <p>1'b0: Has no effect.</p> <p>1'b1: The effect depends on whether the interrupt is edge-triggered or level-sensitive:</p> <p>Edge-triggered</p> <p>Changes the status of the corresponding interrupt:</p> <ul style="list-style-type: none"> <li>inactive if it was previously pending</li> <li>active if it was previously active and</li> </ul>

Bit	Attr	Reset Value	Description
			<p>pending. Has no effect if the interrupt is not pending.</p> <p>Level-sensitive</p> <p>If the corresponding interrupt is pending only because of a write to the ICDISPR, the write changes the status of the interrupt to:</p> <ul style="list-style-type: none"> <li>• inactive if it was previously pending</li> <li>• active if it was previously active and pending. Otherwise the interrupt remains pending if the interrupt signal remains asserted.</li> </ul>

For interrupt ID N, when DIV and MOD are the integer division and modulo operations:

- the corresponding ICDICPR number, M, is given by  $M = N \text{ DIV } 32$
- the offset of the required ICDICPR is  $(0x280 + (4*M))$
- the bit number of the required Set-pending bit in this register is  $N \text{ MOD } 32$ .

### GICD\_ICDABR

Address: Operational Base+0x300

Active Bit Registers

Bit	Attr	Reset Value	Description
31:0	R		<p>For each bit:</p> <p>1'b0: Corresponding interrupt is not active.</p> <p>1'b1: Corresponding interrupt is active.</p>

For interrupt ID N, when DIV and MOD are the integer division and modulo operations:

- the corresponding ICDABR number, M, is given by  $M = N \text{ DIV } 32$
- the offset of the required ICDABR is  $(0x300 + (4*M))$
- the bit number of the required Active bit in this register is  $N \text{ MOD } 32$ .

### GICD\_ICDIPR

Address: Operational Base+0x400

Interrupt Priority Registers

Bit	Attr	Reset Value	Description
7:0	RW	0x0	The lower the value, the greater the priority of the corresponding interrupt.

For interrupt ID N:

- the corresponding ICDIPR number, M, is given by  $M = N$
- the offset of the required ICDIPR is  $(0x400 + M)$

### GICD\_ICDIPTR

Address: Operational Base+0x800

Interrupt Processor Targets Registers

Bit	Attr	Reset Value	Description
7:0	RW	0x1	This register is not used. As in our product, there is only one processor.

### GICD\_ICDICFR

Address: Operational Base+0xc00

Interrupt Configuration Registers

Bit	Attr	Reset Value	Description
2F+1	RW	0x0	<p>F=0,1,2,3....15</p> <p>The encoding is:</p> <p>1'b0: Corresponding interrupt is level-sensitive.</p> <p>1'b1: Corresponding interrupt is edge-triggered.</p>

For interrupt ID N, when DIV and MOD are the integer division and modulo operations:

- the corresponding ICDICFR number, M, is given by  $M = N \text{ DIV } 16$
- the offset of the required ICDIPTR is  $(0xC00 + (4*M))$
- the required Priority field in this register, F, is given by  $F = N \text{ MOD } 16$ , where field 0 refers to register bits [1:0], field 1 refers to bits [3:2], and so on, up to field 15 refers to bits [31:30]

**GICD\_ICDSGIR**

Address: Operational Base+0xf00

Software Generated Interrupt Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:26	-	-	reserved
25:24	W	0x0	2'b00: Send the interrupt to the CPU interface specified in the CPUTargetListfielda. 2'b01: Send the interrupt to all CPU interface except the CPU interface that requested the interrupt. 2'b10: Send the interrupt only to the CPU interface that requested the interrupt. 2'b11: Reserved
23:16	W	0x0	When TargetList Filter = 2'bb00, defines the CPU interface the Distributor must send the interrupt to. Each bit of CPUTargetList[7:0] refers to the corresponding CPU interface, for example CPUTargetList[0] corresponds to CPU interface 0. Setting a bit to 1 sends the interrupt to the corresponding interface.
15	W	0x0	If the GIC implements the Security Extensions, this field is writable only using a Secure access. Any Non-secure write to the ICDSGIR issues an SGI only if the specified SGI is programmed as Non-secure, regardless of the value of bit [15] of the write. Specifies the required security value of the SGI: 1'b0: Send the SGI specified in the SGINTID field to a specified CPU interface only if the SGI is configured as Secure on that interface. 1'b1: Send the SGI specified in the SGINTID field to a specified CPU interface only if the SGI is configured as Non-secure on that interface
14:4	-	-	reserved
3:0	W	0x0	The Interrupt ID of the SGI to send to the specified CPU interface. The value of this field is the Interrupt ID, in the range 0-15, for example a value of 4'b0011 specifies Interrupt ID 3

**12.4.3 GIC CPU interface register summary**

<b>Name</b>	<b>Offset</b>	<b>Size</b>	<b>Reset Value</b>	<b>Description</b>
GICC_ICCICR	0x00	W	0x0	CPU Interface Control Register
GICC_ICCPMR	0x04	W	0x0	Interrupt Priority Mask Register
GICC_ICCBPR	0x08	W	0x0	Binary Point Register
GICC_ICCIAR	0x0C	W	0x3ff	Interrupt Acknowledge Register
GICC_ICCEOIR	0x10	W	-	End of Interrupt Register
GICC_ICCRPR	0x14	W	0xff	Running Priority Register
GICC_ICCHPIR	0x18	W	0x3ff	Highest Pending Interrupt Register

Name	Offset	Size	Reset Value	Description
GICC_ICCABPR	0x1C	W	0x0	Aliased Binary Point Register
GICC_ICCIIDR	0xFC	W	0x0	CPU Interface Identification Register

*Notes:*Size: **B** – Byte (8 bits) access, **HW** – Half WORD (16 bits) access, **W** – WORD (32 bits) access

## 12.4.4 GIC CPU interface detail register description

### GICC\_ICCICR

Address: Operational Base+0x0

CPU Interface Control Register

Bit	Attr	Reset Value	Description
31:5	-	-	reserved
4	RW	0x0	<p>Controls whether the CPU interface uses the Secure or Non-secure Binary Point Register for preemption.</p> <p>1'b0: To determine any preemption, use:            • the Secure Binary Point Register for Secure interrupts            • the Non-secure Binary Point Register for Non-secure interrupts.</p> <p>1'b1: To determine any preemption use the Secure Binary Point Register for both Secure and Non-secure interrupts.</p>
3	RW	0x0	<p>Controls whether the GIC signals Secure interrupts to a target processor using the FIQ or the IRQ signal.</p> <p>1'b0: Signal Secure interrupts using the IRQ signal. 1'b1: Signal Secure interrupts using the FIQ signal. The GIC always signals Non-secure interrupts using the IRQ signal.</p>
2	RW	0x0	<p>Controls whether a Secure read of the ICCIAR, when the highest priority pending interrupt is Non-secure, causes the CPU interface to acknowledge the interrupt.</p> <p>1'b0: If the highest priority pending interrupt is Non-secure, a Secure read of the ICCIAR returns an Interrupt ID of 1022. The read does not acknowledge the interrupt, and the pending status of the interrupt is unchanged.</p> <p>1'b1: If the highest priority pending interrupt is Non-secure, a Secure read of the ICCIAR returns the Interrupt ID of the Non-secure interrupt. The read acknowledges the interrupt, and the status of the interrupt becomes active, or active and pending.</p>
1	RW	0x0	An alias of the Enable bit in the Non-secure ICCICR. This alias bit means Secure software can enable the signalling of Non-secure interrupts. 1'b0: Disable signalling of Non-secure interrupts. 1'b1: Enable signalling of Non-secure interrupts
0	RW	0x0	Global enable for the signalling of Secure interrupts by

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
			the CPU interface to the connected processors. 1'b0: Disable signalling of Secure interrupts. 1'b1: Enable signalling of Secure interrupts

**GICC\_ICCPMR**Address: Operational Base+0x4  
Interrupt Priority Mask Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	-	-	reserved
7:0	RW	0x0	The priority mask level for the CPU interface. If the priority of an interrupt is higher than the value indicated by this field, the interface signals the interrupt to the processor. If the GIC supports fewer than 256 priority levels then some bits are read as zero(RAZ)/write ignore(WI), as follows: 128 supported levels Bit [0] = 1'b0. 64 supported levels Bit [1:0] = 2'b00. 32 supported levels Bit [2:0] = 3'b000. 16 supported levels Bit [3:0] = 4'b0000

**GICC\_ICCBPR**Address: Operational Base+0x8  
Binary Point Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	-	-	reserved
2:0	RW	0x0	The value of this field controls how the 8-bit interrupt priority field is split into a group priority field, used to determine interrupt preemption, and a subpriority field.

**GICC\_ICCIAR**Address: Operational Base+0xc  
Interrupt Acknowledge Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	-	-	reserved
12:10	RO	0x0	For SGIs in a multiprocessor implementation, this field identifies the processor that requested the interrupt. It returns the number of the CPU interface that made the request, for example a value of 3 (3'b011) means the request was generated by a write to the IDCSFGIR on CPU interface 3. For all other interrupts this field is RAZ.
9:0	RO	0x0	The interrupt ID.

**GICC\_ICCEOIR**Address: Operational Base+0x10  
End of Interrupt Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:13	-	-	reserved
12:10	WO	0x0	On a multiprocessor implementation, on completion of the processing of an SGI, this field contains the CPUID value from the corresponding ICCIAR access.
9:0	WO	0x0	The ACKINTID value from the corresponding ICCIAR access.

**GICC\_ICCRPR**

Address: Operational Base+0x14

Running Priority Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	-	-	reserved
7:0	RO	0x0	The priority value of the highest priority interrupt that is active on the CPU interface.

**GICC\_ICCABPR**

Address: Operational Base+0x18

Aliased Binary Point Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:3	-	-	reserved
2:0	RW	0x0	Provides an alias of the Non-secure ICCBPR.

**GICC\_ICCHPIR**

Address: Operational Base+0x1c

Highest Pending Interrupt Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:10	-	-	reserved
9:0	R	0x0	The interrupt ID of the highest priority pending interrupt.

**GICC\_ICCIIDR**

Address: Operational Base+0xfc

CPU Interface Identification Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:20	R	0x390	An IMPLEMENTATION DEFINED product identifier.
19:16	R	0x1	For an implementation that complies with this specification, the value is 0x1
15:12	R	0x2	An IMPLEMENTATION DEFINED revision number for the CPU interface.
11:0	R	0x43B	Contains the JEP106 code of the company that implemented the GIC CPU interface: Bits [11:8]: The JEP106 continuation code of the implementer. Bit [7]: Always 0. Bits [6:0]: The JEP106 identity code of the implementer.

## 12.5 Interface Description

Both distributor interface and CPU interface are secure accessed only after reset.

When the signal cfgsdisable is HIGH, it enhances the security of the GIC by preventing write accesses to security-critical configuration registers. This signal is low after reset, it can be configured through TZPC registers.

## 12.6 Application Notes

### 12.6.1 General handling of interrupts

The GIC operates on interrupts as follows:

1. The GIC determines whether each interrupt is enabled. An interrupt that is not enabled has no further effect on the GIC.(Enables an interrupt by writing to the appropriate ICDISER bit, disables an interrupt by writing to the appropriate ICDICER bit)
2. For each enabled interrupt that is pending, the Distributor determines the targeted processor.

3. For processor, the Distributor determines the highest priority pending interrupt, based on the priority information it holds for each interrupt, and forwards the interrupt to the CPU interface.

4. The CPU interface compares the interrupt priority with the current interrupt priority for the processor, determined by a combination of the Priority Mask Register, the current preemption settings, and the highest priority active interrupt for the processor. If the interrupt has sufficient priority, the GIC signals an interrupt exception request to the processor.

5. When the processor takes the interrupt exception, it reads the ICCIAR in its CPU interface to acknowledge the interrupt. This read returns an Interrupt ID that the processor uses to select the correct interrupt handler. When it recognizes this read, the GIC changes the state of the interrupt:

- If the pending state of the interrupt persists when the interrupt becomes active, or if the interrupt is generated again, from pending to active and pending.
- Otherwise, from pending to active

6. When the processor has completed handling the interrupt, it signals this completion by writing to the ICCEOIR in the GIC

### **Generating an SGI**

A processor generates an SGI by writing to an ICDSGIR.

### **12.6.2 Interrupt prioritization**

Software configures interrupt prioritization in the GIC by assigning a priority value to each interrupt source. Priority values are 8-bit unsigned binary.

In this product, GIC implements 64 priority levels. So only the highest 6 bits are valid, the lower 2 bits read as zero.

In the GIC prioritization scheme, lower numbers have higher priority, that is, the lower the assigned priority value the higher the priority of the interrupt. The highest interrupt priority always has priority field value 0.

The ICDIPRs hold the priority value for each supported interrupt. To determine the number of priority bits implemented write 0xFF to an ICDIPR priority field and read back the value stored.

### **Preemption**

A CPU interface supports forwarding of higher priority pending interrupts to a target processor before an active interrupt completes. A pending interrupt is only forwarded if it has a higher priority than all of:

- the priority of the highest priority active interrupt on the target processor, the running priority for the processor, see Running Priority Register (ICCRPR) .
- The priority mask, see Priority masking.
- The priority group, see Priority grouping.

Preemption occurs at the time when the processor acknowledges the new interrupt, and starts to service it in preference to the previously active interrupt or the currently running process. When this occurs, the initial active interrupt is said to have been preempted. Starting to service an interrupt while another interrupt is still active is sometimes described as interrupt nesting.

### **Priority masking**

The ICCPMR for a CPU interface defines a priority threshold for the target processor, see Interrupt Priority Mask Register. The GIC only signals pending interrupts with a higher priority than this threshold value to the target processor. A value of zero, the register reset value, masks all interrupts to the associated processor.

The GIC always masks an interrupt that has the largest supported priority field value. This provides an additional means of preventing an interrupt being signalled to any processor.

### **Priority grouping**

Priority grouping splits each priority value into two fields, the group priority and the subpriority fields. The GIC uses the group priority field to determine whether a pending interrupt has sufficient priority to preempt a currently active interrupt.

The binary point field in the ICCBPR controls the split of the priority bits into the two parts. This 3-bit field specifies how many of the least significant bits of the 8-bit interrupt priority field are excluded from the group priority field, as following table shows.

<b>Binary point value</b>	<b>Group priority field</b>	<b>Subpriority field</b>	<b>Field with binary point</b>
0	[7:1]	[0]	ggggggg.s
1	[7:2]	[1:0]	gggggg.ss
2	[7:3]	[2:0]	gggggsss
3	[7:4]	[3:0]	gggg.ssss
4	[7:5]	[4:0]	ggg.sssss
5	[7:6]	[5:0]	gg.sssssss
6	[7]	[6:0]	g.sssssss
7	No preemption	[7:0]	.ssssssss

Where multiple pending interrupts share the same group priority, the GIC uses the subpriority field to resolve the priority within a group.

### 12.6.3 The effect of the Security Extensions on interrupt handling

If a GIC CPU interface implements the Security Extensions, it provides two interrupt output signals, IRQ and FIQ:

- The CPU interface always uses the IRQ exception request for Non-secure interrupts
- Software can configure the CPU interface to use either IRQ or FIQ exception requests for Secure interrupts.

#### Security Extensions support

Software can detect support for the Security Extensions by reading the ICDICTR.SecurityExtn bit, see Interrupt Controller Type Register (ICDICTR).

Secure software makes Secure writes to the ICDISRs to configure each interrupt as Secure or Non-secure, see Interrupt Security Registers (ICDISRn).

In addition:

- The banking of registers provides independent control of Secure and Non-secure interrupts.
- The Secure copy of the ICCICR has additional fields to control the processing of Secure and Non-secure interrupts, see CPU Interface Control Register (ICCICR) These fields are:
  - the SBPR bit, that affects the preemption of Non-secure interrupts.
  - the FIQEn bit, that controls whether the interface signals Secure interrupt to the processor using the IRQ or FIQ interrupt exception requests.
  - the AckCtl bit, that affects the acknowledgment of Non-secure interrupts.
  - the EnableNS bit, that controls whether Non-secure interrupts are signaled to the processor, and is an alias of the Enable bit in the Non-secure ICCICR.
- The Non-secure copy of the ICCBPR is aliased as the ICCABPR, see Aliased
- Binary Point Register (ICCBP). This is a Secure register, meaning it is only accessible by Secure accesses.

#### Effect of the Security Extensions on interrupt acknowledgement

When a processor takes an interrupt, it acknowledges the interrupt by reading the ICCIAR. A read of the ICCIAR always acknowledges the highest priority pending interrupt for the processor performing the read.

If the highest priority pending interrupt is a Secure interrupt, the processor must make a Secure read of the ICCIAR to acknowledge it.

By default, the processor must make a Non-secure read of the ICCIAR to acknowledge a Non-secure interrupt. If the AckCtl bit in the Secure ICCICR is set to 1 the processor can make a Secure read of the ICCIAR to acknowledge a Non-secure interrupt.

If the read of the ICCIAR does not match the security of the interrupt, taking account of the AckCtl bit value for a Non-secure interrupt, the ICCIAR read does not acknowledge any interrupt and returns the value:

- 1022 for a Secure read when the highest priority interrupt is Non-secure
- 1023 for a Non-secure read when the highest priority interrupt is Secure.

## Chapter 13 DMA Controller(DMAC)

### 13.1 Overview

DMAC is mainly used for data transfer of the following slaves: I2S0, I2S1, I2S2, SPDIF, UART0, UART1, UART2, SPI, PWM.

Following table shows the DMAC request mapping scheme.

Table 13-1 DMAC Request Mapping Table

Req number	Source	Polarity
0	I2S2_2ch tx	High level
1	I2S2_2ch rx	High level
2	Uart0 tx	High level
3	Uart0 rx	High level
4	Uart1 tx	High level
5	Uart1 rx	High level
6	Uart2 tx	High level
7	Uart2 rx	High level
8	SPI tx	High level
9	SPI rx	High level
10	SPDIF	High level
11	I2S0_8ch tx	High level
12	I2S0_8ch rx	High level
13	pwm_tx	High level
14	I2S1_8ch_tx	High level
15	I2S1_8ch_rx	High level

DMAC supports the following features:

- Supports 16 peripheral request.
- Up to 64bits data size
- 8 channel at the same time
- Up to burst 16
- 16 interrupts output and 1 abort output
- Supports 64 MFIFO depth.

### 13.2 Block Diagram

Following figure shows the block diagram of DMAC.

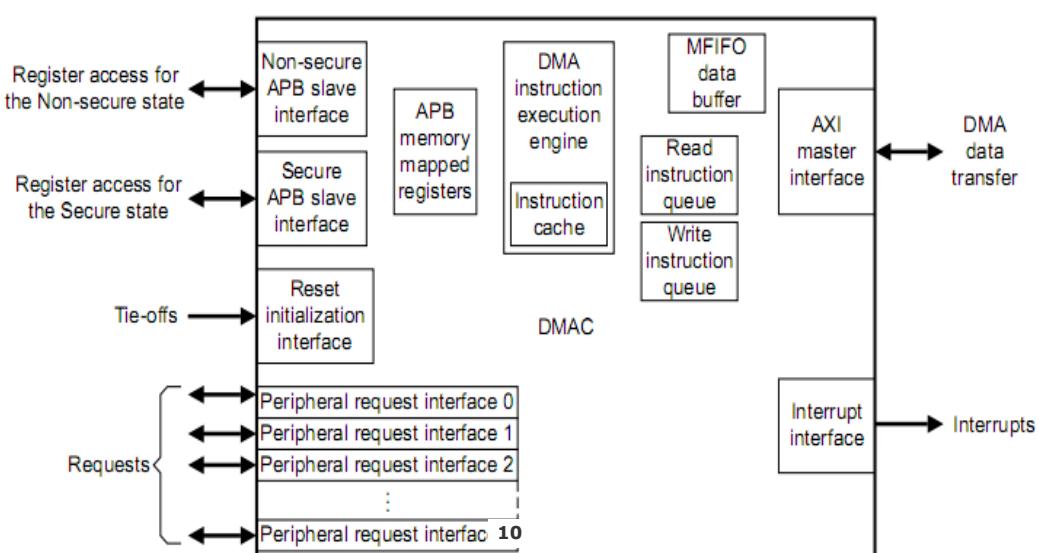


Fig. 13-1 Block diagram of DMAC

## 13.3 Function Description

### 13.3.1 Introduction

The DMAC contains an instruction processing block that enables it to process program code that controls a DMA transfer. The program code is stored in a region of system memory that the DMAC accesses using its AXI interface. The DMAC stores instructions temporarily in a cache.

DMAC supports 8 channels, each channel capable of supporting a single concurrent thread of DMA operation. In addition, a single DMA manager thread exists, and you can use it to initialize the DMA channel threads. The DMAC executes up to one instruction for each AXI clock cycle. To ensure that it regularly executes each active thread, it alternates by processing the DMA manager thread and then a DMA channel thread. It uses a round-robin process when selecting the next active DMA channel thread to execute.

The DMAC uses variable-length instructions that consist of one to six bytes. It provides a separate Program Counter (PC) register for each DMA channel. When a thread requests an instruction from an address, the cache performs a look-up. If a cache hit occurs, then the cache immediately provides the data. Otherwise, the thread is stalled while the DMAC uses the AXI interface to perform a cache line fill. If an instruction is greater than 4 bytes, or spans the end of a cache line, the DMAC performs multiple cache accesses to fetch the instruction. When a cache line fill is in progress, the DMAC enables other threads to access the cache, but if another cache miss occurs, this stalls the pipeline until the first line fill is complete.

When a DMA channel thread executes a load or store instruction, the DMAC adds the instruction to the relevant read or write queue. The DMAC uses these queues as an instruction storage buffer prior to it issuing the instructions on the AXI bus. The DMAC also contains a Multi First-In-First-Out (MFIFO) data buffer that it uses to store data that it reads, or writes, during a DMA transfer.

### 13.3.2 Operating states

Following figure shows the operating states for the DMA manager thread and DMA channel threads.

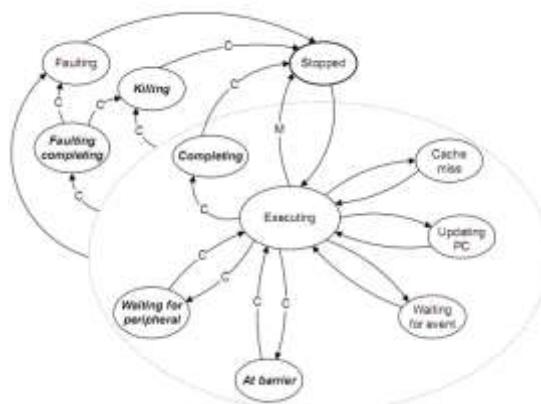


Fig. 13-2 DMAC operation states

Notes: arcs with no letter designator indicate state transitions for the DMA manager and DMA channel threads, otherwise use is restricted as follows:

C DMA channel threads only.

M DMA manager thread only.

After the DMAC exits from reset, it sets all DMA channel threads to the stopped state, and the status of boot\_from\_pc(tie-off interface of dmac) controls the DMA manager thread state: boot\_from\_pc is LOW :DMA manager thread moves to the Stopped state. boot\_from\_pc is HIGH :DMA manager thread moves to the Executing state.

## 13.4 Register Description

### 13.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
DMAC_DSR	0x0000	W	0x00000000	DMA Manager Status Register
DMAC_DPC	0x0004	W	0x00000000	DMA Program Counter Register
DMAC_INTEN	0x0020	W	0x00000000	Interrupt Enable Register
DMAC_EVENT_RIS	0x0024	W	0x00000000	Event-Interrupt Raw Status Register
DMAC_INTMIS	0x0028	W	0x00000000	Interrupt Status Register
DMAC_INTCLR	0x002c	W	0x00000000	Interrupt Clear Register
DMAC_FSRD	0x0030	W	0x00000000	Fault Status DMA Manager Register
DMAC_FSRC	0x0034	W	0x00000000	Fault Status DMA Channel Register
DMAC_FTRD	0x0038	W	0x00000000	Fault Type DMA Manager Register
DMAC_FTR0	0x0040	W	0x00000000	Fault Type DMA Channel Register
DMAC_FTR1	0x0044	W	0x00000000	Fault Type DMA Channel Register
DMAC_FTR2	0x0048	W	0x00000000	Fault Type DMA Channel Register
DMAC_FTR3	0x004c	W	0x00000000	Fault Type DMA Channel Register
DMAC_FTR4	0x0050	W	0x00000000	Fault Type DMA Channel Register
DMAC_FTR5	0x0054	W	0x00000000	Fault Type DMA Channel Register
DMAC_FTR6	0x0058	W	0x00000000	Fault Type DMA Channel Register
DMAC_FTR7	0x005c	W	0x00000000	Fault Type DMA Channel Register
DMAC_CSR0	0x0100	W	0x00000000	Channel Status Registers
DMAC_CPC0	0x0104	W	0x00000000	Channel Program Counter Registers
DMAC_CSR1	0x0108	W	0x00000000	Channel Status Registers
DMAC_CPC1	0x010c	W	0x00000000	Channel Program Counter Registers
DMAC_CSR2	0x0110	W	0x00000000	Channel Status Registers
DMAC_CPC2	0x0114	W	0x00000000	Channel Program Counter Registers
DMAC_CSR3	0x0118	W	0x00000000	Channel Status Registers
DMAC_CPC3	0x011c	W	0x00000000	Channel Program Counter Registers
DMAC_CSR4	0x0120	W	0x00000000	Channel Status Registers
DMAC_CPC4	0x0124	W	0x00000000	Channel Program Counter Registers
DMAC_CSR5	0x0128	W	0x00000000	Channel Status Registers
DMAC_CPC5	0x012c	W	0x00000000	Channel Program Counter Registers
DMAC_CSR6	0x0130	W	0x00000000	Channel Status Registers
DMAC_CPC6	0x0134	W	0x00000000	Channel Program Counter Registers
DMAC_CSR7	0x0138	W	0x00000000	Channel Status Registers

Name	Offset	Size	Reset Value	Description
DMAC_CPC7	0x013c	W	0x00000000	Channel Program Counter Registers
DMAC_SAR0	0x0400	W	0x00000000	Source Address Registers
DMAC_DAR0	0x0404	W	0x00000000	Destination Address Registers
DMAC_CCR0	0x0408	W	0x00000000	Channel Control Registers
DMAC_LC0_0	0x040c	W	0x00000000	Loop Counter 0 Registers
DMAC_LC1_0	0x0410	W	0x00000000	Loop Counter 1 Registers
DMAC_SAR1	0x0420	W	0x00000000	Source Address Registers
DMAC_DAR1	0x0424	W	0x00000000	Destination Address Registers
DMAC_CCR1	0x0428	W	0x00000000	Channel Control Registers
DMAC_LC0_1	0x042c	W	0x00000000	Loop Counter 0 Registers
DMAC_LC1_1	0x0430	W	0x00000000	Loop Counter 1 Registers
DMAC_SAR2	0x0440	W	0x00000000	Source Address Registers
DMAC_DAR2	0x0444	W	0x00000000	Destination Address Registers
DMAC_CCR2	0x0448	W	0x00000000	Channel Control Registers
DMAC_LC0_2	0x044c	W	0x00000000	Loop Counter 0 Registers
DMAC_LC1_2	0x0450	W	0x00000000	Loop Counter 1 Registers
DMAC_SAR3	0x0460	W	0x00000000	Source Address Registers
DMAC_DAR3	0x0464	W	0x00000000	Destination Address Registers
DMAC_CCR3	0x0468	W	0x00000000	Channel Control Registers
DMAC_LC0_3	0x046c	W	0x00000000	Loop Counter 0 Registers
DMAC_LC1_3	0x0470	W	0x00000000	Loop Counter 1 Registers
DMAC_SAR4	0x0480	W	0x00000000	Source Address Registers
DMAC_DAR4	0x0484	W	0x00000000	Destination Address Registers
DMAC_CCR4	0x0488	W	0x00000000	Channel Control Registers
DMAC_LC0_4	0x048c	W	0x00000000	Loop Counter 0 Registers
DMAC_LC1_4	0x0490	W	0x00000000	Loop Counter 1 Registers
DMAC_SAR5	0x04a0	W	0x00000000	Source Address Registers
DMAC_DAR5	0x04a4	W	0x00000000	Destination Address Registers
DMAC_CCR5	0x04a8	W	0x00000000	Channel Control Registers
DMAC_LC0_5	0x04ac	W	0x00000000	Loop Counter 0 Registers
DMAC_LC1_5	0x04b0	W	0x00000000	Loop Counter 1 Registers
DMAC_SAR6	0x04c0	W	0x00000000	Source Address Registers
DMAC_DAR6	0x04c4	W	0x00000000	Destination Address Registers
DMAC_CCR6	0x04c8	W	0x00000000	Channel Control Registers
DMAC_LC0_6	0x04cc	W	0x00000000	Loop Counter 0 Registers
DMAC_LC1_6	0x04d0	W	0x00000000	Loop Counter 1 Registers
DMAC_SAR7	0x04e0	W	0x00000000	Source Address Registers
DMAC_DAR7	0x04e4	W	0x00000000	Destination Address Registers
DMAC_CCR7	0x04e8	W	0x00000000	Channel Control Registers
DMAC_LC0_7	0x04ec	W	0x00000000	Loop Counter 0 Registers
DMAC_LC1_7	0x04f0	W	0x00000000	Loop Counter 1 Registers
DMAC_DBGSTATUS	0x0d00	W	0x00000000	Debug Status Register
DMAC_DBGCMD	0x0d04	W	0x00000000	Debug Command Register

Name	Offset	Size	Reset Value	Description
DMAC_DBGINST0	0x0d08	W	0x00000000	Debug Instruction-0 Register
DMAC_DBGINST1	0x0d0c	W	0x00000000	Debug Instruction-1 Register
DMAC_CR0	0xe00	W	0x00047051	Configuration Register 0
DMAC_CR1	0xe04	W	0x00000057	Configuration Register 1
DMAC_CR2	0xe08	W	0x00000000	Configuration Register 2
DMAC_CR3	0xe0c	W	0x00000000	Configuration Register 3
DMAC_CR4	0xe10	W	0x00000006	Configuration Register 4
DMAC_CRDn	0xe14	W	0x02094733	DMA Configuration Register
DMAC_WD	0xe80	W	0x00000000	DMA Watchdog Register

Notes: **S**-Size: **B**- Byte (8 bits) access, **HW**- Half WORD (16 bits) access, **W**-WORD (32 bits) access

### 13.4.2 Detail Register Description

#### DMAC\_DSR

Address: Operational Base + offset (0x0000)

DMA Manager Status Register

Bit	Attr	Reset Value	Description
31:10	RO	0x0	reserved
9	RO	0x0	Provides the security status of the DMA manager thread: 0 = DMA manager operates in the Secure state 1 = DMA manager operates in the Non-secure state.
8:4	RO	0x00	When the DMA manager thread executes a DMAWFE instruction, it waits for the following event to occur: b00000 = event[0] b00001 = event[1] b00010 = event[2] ... b11111 = event[31].
3:0	RO	0x0	The operating state of the DMA manager: b0000 = Stopped b0001 = Executing b0010 = Cache miss b0011 = Updating PC b0100 = Waiting for event b0101-b1110 = reserved b1111 = Faulting.

#### DMAC\_DPC

Address: Operational Base + offset (0x0004)

DMA Program Counter Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	Program counter for the DMA manager thread

#### DMAC\_INTEN

Address: Operational Base + offset (0x0020)

Interrupt Enable Register

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	<p>Program the appropriate bit to control how the DMAC responds when it executes DMASEV:</p> <p>Bit [N] = 0 If the DMAC executes DMASEV for the event-interrupt resource N then the DMAC signals event N to all of the threads. Set bit [N] to 0 if your system design does not use irq[N] to signal an interrupt request.</p> <p>Bit [N] = 1 If the DMAC executes DMASEV for the event-interrupt resource N then the DMAC sets irq[N] HIGH. Set bit [N] to 1 if your system designer requires irq[N] to signal an interrupt request.</p>

**DMAC\_EVENT\_RIS**

Address: Operational Base + offset (0x0024)

Event-Interrupt Raw Status Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	<p>Returns the status of the event-interrupt resources:</p> <p>Bit [N] = 0 Event N is inactive or irq[N] is LOW.</p> <p>Bit [N] = 1 Event N is active or irq[N] is HIGH.</p>

**DMAC\_INTMIS**

Address: Operational Base + offset (0x0028)

Interrupt Status Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	<p>Provides the status of the interrupts that are active in the DMAC:</p> <p>Bit [N] = 0 Interrupt N is inactive and therefore irq[N] is LOW.</p> <p>Bit [N] = 1 Interrupt N is active and therefore irq[N] is HIGH</p>

**DMAC\_INTCLR**

Address: Operational Base + offset (0x002c)

Interrupt Clear Register

Bit	Attr	Reset Value	Description
31:0	WO	0x00000000	<p>Controls the clearing of the irq outputs:</p> <p>Bit [N] = 0 The status of irq[N] does not change.</p> <p>Bit [N] = 1 The DMAC sets irq[N] LOW if the INTEN Register programs the DMAC to signal an interrupt.</p> <p>Otherwise, the status of irq[N] does not change.</p>

**DMAC\_FSRD**

Address: Operational Base + offset (0x0030)

Fault Status DMA Manager Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	<p>Provides the fault status of the DMA manager. Read as:</p> <p>0 = the DMA manager thread is not in the Faulting state</p> <p>1 = the DMA manager thread is in the Faulting state.</p>

**DMAC\_FSRC**

Address: Operational Base + offset (0x0034)

Fault Status DMA Channel Register

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	Each bit provides the fault status of the corresponding channel. Read as: Bit [N] = 0 No fault is present on DMA channel N. Bit [N] = 1 DMA channel N is in the Faulting or Faulting completing state.

**DMAC\_FTRD**

Address: Operational Base + offset (0x0038)

Fault Type DMA Manager Register

Bit	Attr	Reset Value	Description
31	RO	0x0	reserved
30	RO	0x0	If the DMA manager aborts, this bit indicates if the erroneous instruction was read from the system memory or from the debug interface: 0 = instruction that generated an abort was read from system memory 1 = instruction that generated an abort was read from the debug interface.
29:17	RO	0x0	reserved
16	RO	0x0	Indicates the AXI response that the DMAC receives on the RRESP bus, after the DMA manager performs an instruction fetch: 0 = OKAY response 1 = EXOKAY, SLVERR, or DECERR response
15:6	RO	0x0	reserved
5	RO	0x0	Indicates if the DMA manager was attempting to execute DMAWFE or DMASEV with inappropriate security permissions: 0 = DMA manager has appropriate security to execute DMAWFE or DMASEV 1 = a DMA manager thread in the Non-secure state attempted to execute either: DMAWFE to wait for a secure event DMASEV to create a secure event or secure interrupt
4	RO	0x0	Indicates if the DMA manager was attempting to execute DMAGO with inappropriate security permissions: 0 = DMA manager has appropriate security to execute DMAGO 1 = DMA manager thread in the Non-secure state attempted to execute DMAGO to create a DMA channel operating in the Secure state.
3:2	RO	0x0	reserved
1	RO	0x0	Indicates if the DMA manager was attempting to execute an instruction operand that was not valid for the configuration of the DMAC: 0 = valid operand 1 = invalid operand.

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
0	RW	0x0	Indicates if the DMA manager was attempting to execute an undefined instruction: 0 = defined instruction 1 = undefined instruction.

**DMAC\_FTR0~DMAC\_FTR7**

Address: Operational Base + offset (0x0040)

Operational Base+0x44  
 Operational Base+0x48  
 Operational Base+0x4C  
 Operational Base+0x50  
 Operational Base+0x54  
 Operational Base+0x58  
 Operational Base+0x5C

Fault Type DMA Channel Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31	RO	0x0	Indicates if the DMA channel has locked-up because of resource starvation: 0 = DMA channel has adequate resources 1 = DMA channel has locked-up because of insufficient resources. This fault is an imprecise abort
30	RO	0x0	If the DMA channel aborts, this bit indicates if the erroneous instruction was read from the system memory or from the debug interface: 0 = instruction that generated an abort was read from system memory 1 = instruction that generated an abort was read from the debug interface. This fault is an imprecise abort but the bit is only valid when a precise abort occurs.
29:19	RO	0x0	reserved
18	RO	0x0	Indicates the AXI response that the DMAC receives on the RRESP bus, after the DMA channel thread performs a data read: 0 = OKAY response 1 = EXOKAY, SLVERR, or DECERR response. This fault is an imprecise abort
17	RO	0x0	Indicates the AXI response that the DMAC receives on the BRESP bus, after the DMA channel thread performs a data write: 0 = OKAY response 1 = EXOKAY, SLVERR, or DECERR response. This fault is an imprecise abort.

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
16	RO	0x0	Indicates the AXI response that the DMAC receives on the RRESP bus, after the DMA channel thread performs an instruction fetch: 0 = OKAY response 1 = EXOKAY, SLVERR, or DECERR response. This fault is a precise abort.
15:14	RO	0x0	reserved
13	RO	0x0	Indicates if the MFIFO did not contain the data to enable the DMAC to perform the DMAST: 0 = MFIFO contains all the data to enable the DMAST to complete 1 = previous DMALDs have not put enough data in the MFIFO to enable the DMAST to complete. This fault is a precise abort.
12	RO	0x0	Indicates if the MFIFO prevented the DMA channel thread from executing DMALD or DMAST. Depending on the instruction: DMALD 0 = MFIFO contains sufficient space 1 = MFIFO is too small to hold the data that DMALD requires. DMAST 0 = MFIFO contains sufficient data 1 = MFIFO is too small to store the data to enable DMAST to complete. This fault is an imprecise abort
11:8	RO	0x0	reserved
7	RO	0x0	Indicates if a DMA channel thread, in the Non-secure state, attempts to program the CCRn Register to perform a secure read or secure write: 0 = a DMA channel thread in the Non-secure state is not violating the security permissions 1 = a DMA channel thread in the Non-secure state attempted to perform a secure read or secure write. This fault is a precise abort
6	RO	0x0	Indicates if a DMA channel thread, in the Non-secure state, attempts to execute DMAWFP, DMALDP, DMASTP, or DMAFLUSHP with inappropriate security permissions: 0 = a DMA channel thread in the Non-secure state is not violating the security permissions 1 = a DMA channel thread in the Non-secure state attempted to execute either: <ul style="list-style-type: none"><li>o DMAWFP to wait for a secure peripheral</li><li>o DMALDP or DMASTP to notify a secure peripheral</li><li>o DMAFLUSHP to flush a secure peripheral.</li></ul> This fault is a precise abort.

Bit	Attr	Reset Value	Description
5	RO	0x0	Indicates if the DMA channel thread attempts to execute DMAWFE or DMASEV with inappropriate security permissions: 0 = a DMA channel thread in the Non-secure state is not violating the security permissions 1 = a DMA channel thread in the Non-secure state attempted to execute either: DMAWFE to wait for a secure event DMASEV to create a secure event or secure interrupt. This fault is a precise abort.
4:2	RO	0x0	reserved
1	RO	0x0	Indicates if the DMA channel thread was attempting to execute an instruction operand that was not valid for the configuration of the DMAC: 0 = valid operand 1 = invalid operand. This fault is a precise abort.
0	RO	0x0	Indicates if the DMA channel thread was attempting to execute an undefined instruction: 0 = defined instruction 1 = undefined instruction. This fault is a precise abort

**DMAC\_CSR0~DMAC\_CSR7**

Address: Operational Base+0x100

Operational Base+0x108  
 Operational Base+0x110  
 Operational Base+0x118  
 Operational Base+0x120  
 Operational Base+0x128  
 Operational Base+0x130  
 Operational Base+0x138

Channel Status Registers

Bit	Attr	Reset Value	Description
31:22	RO	0x0	reserved
21	RO	0x0	The channel non-secure bit provides the security of the DMA channel: 0 = DMA channel operates in the Secure state 1 = DMA channel operates in the Non-secure state
20:16	RO	0x0	reserved
15	RO	0x0	When the DMA channel thread executes DMAWFP this bit indicates if the periph operand was set: 0 = DMAWFP executed with the periph operand not set 1 = DMAWFP executed with the periph operand set
14	RO	0x0	When the DMA channel thread executes DMAWFP this bit indicates if the burst or single operand were set: 0 = DMAWFP executed with the single operand set 1 = DMAWFP executed with the burst operand set.

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
13:9	RO	0x0	reserved
8:4	RO	0x00	If the DMA channel is in the Waiting for event state or the Waiting for peripheral state then these bits indicate the event or peripheral number that the channel is waiting for: b00000 = DMA channel is waiting for event, or peripheral, 0 b00001 = DMA channel is waiting for event, or peripheral, 1 b00010 = DMA channel is waiting for event, or peripheral, 2 ... b11111 = DMA channel is waiting for event, or peripheral, 31
3:0	RO	0x0	The channel status encoding is: b0000 = Stopped b0001 = Executing b0010 = Cache miss b0011 = Updating PC b0100 = Waiting for event b0101 = At barrier b0110 = reserved b0111 = Waiting for peripheral b1000 = Killing b1001 = Completing b1010-b1101 = reserved b1110 = Faulting completing b1111 = Faulting

**DMAC\_CPC0~DMAC\_CPC7**

Address: Operational Base+0x104

Operational Base+0x10C  
 Operational Base+0x114  
 Operational Base+0x11c  
 Operational Base+0x124  
 Operational Base+0x12C  
 Operational Base+0x134  
 Operational Base+0x13C

Channel Program Counter Registers

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	Program counter for the DMA channel 0 thread

**DMAC\_SAR0~DMAC\_SAR7**

Address: Operational Base+0x400

Operational Base+0x420  
 Operational Base+0x440  
 Operational Base+0x460  
 Operational Base+0x480  
 Operational Base+0x4A0  
 Operational Base+0x4C0  
 Operational Base+0x4E0

Source Address Registers

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	Address of the source data for DMA channel 0

**DMAC\_DAR0~DMAC\_DAR7**

Address: Operational Base+0x404  
 Operational Base+0x424  
 Operational Base+0x444  
 Operational Base+0x464  
 Operational Base+0x484  
 Operational Base+0x4A4  
 Operational Base+0x4C4  
 Operational Base+0x4E4

DestinationAddress Registers

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:0	RO	0x00000000	Address of the Destination data for DMA channel 0

**DMAC\_CCR0~DMAC\_CCR7**

Address: Operational Base+0x408  
 Operational Base+0x428  
 Operational Base+0x448  
 Operational Base+0x468  
 Operational Base+0x488  
 Operational Base+0x4A8  
 Operational Base+0x4C8  
 Operational Base+0x4E8

Channel Control Registers

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:28	RO	0x0	reserved
27:25	RO	0x0	Programs the state of AWCACHE[3,1:0]a when the DMAC writes the destination data. Bit [27] 0 = AWCACHE[3] is LOW 1 = AWCACHE[3] is HIGH. Bit [26] 0 = AWCACHE[1] is LOW 1 = AWCACHE[1] is HIGH. Bit [25] 0 = AWCACHE[0] is LOW 1 = AWCACHE[0] is HIGH
24:22	RO	0x0	Programs the state of AWPROT[2:0]a when the DMAC writes the destination data. Bit [24] 0 = AWPROT[2] is LOW 1 = AWPROT[2] is HIGH. Bit [23] 0 = AWPROT[1] is LOW 1 = AWPROT[1] is HIGH. Bit [22] 0 = AWPROT[0] is LOW 1 = AWPROT[0] is HIGH

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
21:18	RO	0x0	<p>For each burst, these bits program the number of data transfers that the DMAC performs when it writes the destination data:</p> <p>b0000 = 1 data transfer      b0001 = 2 data transfers      b0010 = 3 data transfers      ...      b1111 = 16 data transfers.</p> <p>The total number of bytes that the DMAC writes out of the MFIFO when it executes a DMAST instruction is the product of dst_burst_len and dst_burst_size.</p>
17:15	RO	0x0	<p>For each beat within a burst, it programs the number of bytes that the DMAC writes to the destination:</p> <p>b000 = writes 1 byte per beat      b001 = writes 2 bytes per beat      b010 = writes 4 bytes per beat      b011 = writes 8 bytes per beat      b100 = writes 16 bytes per beat      b101-b111 = reserved.</p> <p>The total number of bytes that the DMAC writes out of the MFIFO when it executes a DMAST instruction is the product of dst_burst_len and dst_burst_size.</p>
14	RO	0x0	<p>Programs the burst type that the DMAC performs when it writes the destination data:</p> <p>0 = Fixed-address burst. The DMAC signals AWBURST[0] LOW.      1 = Incrementing-address burst. The DMAC signals AWBURST[0] HIGH.</p>
13:11	RO	0x0	<p>Set the bits to control the state of ARCACHE[2:0]a when the DMAC reads the source data.</p> <p>Bit [13] 0 = ARCACHE[2] is LOW      1 = ARCACHE[2] is HIGH.      Bit [12] 0 = ARCACHE[1] is LOW      1 = ARCACHE[1] is HIGH.      Bit [11] 0 = ARCACHE[0] is LOW      1 = ARCACHE[0] is HIGH.</p>
10:8	RO	0x0	<p>Programs the state of ARPROT[2:0]a when the DMAC reads the source data.</p> <p>Bit [10] 0 = ARPROT[2] is LOW      1 = ARPROT[2] is HIGH.      Bit [9] 0 = ARPROT[1] is LOW      1 = ARPROT[1] is HIGH.      Bit [8] 0 = ARPROT[0] is LOW      1 = ARPROT[0] is HIGH.</p>

Bit	Attr	Reset Value	Description
7:4	RO	0x0	<p>For each burst, these bits program the number of data transfers that the DMAC performs when it reads the source data:</p> <p>b0000 = 1 data transfer      b0001 = 2 data transfers      b0010 = 3 data transfers      ...      b1111 = 16 data transfers.</p> <p>The total number of bytes that the DMAC reads into the MFIFO when it executes a DMA LD instruction is the product of src_burst_len and src_burst_size</p>
3:1	RO	0x0	<p>For each beat within a burst, it programs the number of bytes that the DMAC reads from the source:</p> <p>b000 = reads 1 byte per beat      b001 = reads 2 bytes per beat      b010 = reads 4 bytes per beat      b011 = reads 8 bytes per beat      b100 = reads 16 bytes per beat      b101-b111 = reserved.</p> <p>The total number of bytes that the DMAC reads into the MFIFO when it executes a DMA LD instruction is the product of src_burst_len and src_burst_size</p>
0	RO	0x0	<p>Programs the burst type that the DMAC performs when it reads the source data:</p> <p>0 = Fixed-address burst. The DMAC signals ARBURST[0] LOW.      1 = Incrementing-address burst. The DMAC signals ARBURST[0] HIGH</p>

**DMAC\_LC0\_0~DMAC\_LC0\_7**

Address: Operational Base+0x40c  
 Operational Base+0x42C  
 Operational Base+0x44C  
 Operational Base+0x46C  
 Operational Base+0x48C  
 Operational Base+0x4AC  
 Operational Base+0x4CC  
 Operational Base+0x4EC

Loop Counter 0 Registers

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	RO	0x00	Loop counter 0 iterations

**DMAC\_LC1\_0~DMAC\_LC1\_7**

Address: Operational Base+0x410  
 Operational Base+0x430  
 Operational Base+0x450  
 Operational Base+0x470  
 Operational Base+0x490  
 Operational Base+0x4B0  
 Operational Base+0x4D0

Operational Base+0x4F0  
Loop Counter 1 Registers

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:8	RO	0x0	reserved
7:0	RO	0x00	Loop counter 1 iterations

**DMAC\_DBGSTATUS**

Address: Operational Base + offset (0x0d00)

Debug Status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:2	RO	0x0	reserved
1:0	RO	0x0	The debug encoding is as follows: b00 = execute the instruction that the DBGINST [1:0] Registers contain b01 = reserved b10 = reserved b11 = reserved.

**DMAC\_DBGCMD**

Address: Operational Base + offset (0x0d04)

Debug Command Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:2	RO	0x0	reserved
1:0	WO	0x0	The debug encoding is as follows: b00 = execute the instruction that the DBGINST [1:0] Registers contain b01 = reserved b10 = reserved b11 = reserved

**DMAC\_DBGINST0**

Address: Operational Base + offset (0x0d08)

Debug Instruction-0 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	WO	0x00	Instruction byte 1
23:16	WO	0x00	Instruction byte 0
15:11	RO	0x0	reserved
10:8	WO	0x0	DMA channel number: b000 = DMA channel 0 b001 = DMA channel 1 b010 = DMA channel 2 ... b111 = DMA channel 7
7:1	RO	0x0	reserved
0	WO	0x0	The debug thread encoding is as follows: 0 = DMA manager thread 1 = DMA channel.

**DMAC\_DBGINST1**

Address: Operational Base + offset (0x0d0c)

Debug Instruction-1 Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:24	WO	0x00	Instruction byte 5
23:16	WO	0x00	Instruction byte 4
15:8	WO	0x00	Instruction byte 3
7:0	WO	0x00	Instruction byte 2

**DMAC\_CRO**

Address: Operational Base + offset (0x0e00)

Configuration Register 0

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:22	RO	0x0	reserved
21:17	RO	0x02	Number of interrupt outputs that the DMAC provides: b00000 = 1 interrupt output, irq[0] b00001 = 2 interrupt outputs, irq[1:0] b00010 = 3 interrupt outputs, irq[2:0] ... b11111 = 32 interrupt outputs, irq[31:0].
16:12	RO	0x07	Number of peripheral request interfaces that the DMAC provides: b00000 = 1 peripheral request interface b00001 = 2 peripheral request interfaces b00010 = 3 peripheral request interfaces ... b11111 = 32 peripheral request interfaces.
11:7	RO	0x0	reserved
6:4	RO	0x5	Number of DMA channels that the DMAC supports: b000 = 1 DMA channel b001 = 2 DMA channels b010 = 3 DMA channels ... b111 = 8 DMA channels.
3	RO	0x0	reserved
2	RO	0x0	Indicates the status of the boot_manager_ns signal when the DMAC exited from reset: 0 = boot_manager_ns was LOW 1 = boot_manager_ns was HIGH.
1	RO	0x0	Indicates the status of the boot_from_pc signal when the DMAC exited from reset: 0 = boot_from_pc was LOW 1 = boot_from_pc was HIGH
0	RO	0x1	Supports peripheral requests: 0 = the DMAC does not provide a peripheral request interface 1 = the DMAC provides the number of peripheral request interfaces that the num_periph_req field specifies.

**DMAC\_CR1**

## RK3228 TRM

Address: Operational Base + offset (0x0e04)

Configuration Register 1

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:4	RO	0x5	[7:4] num_i-cache_lines Number of i-cache lines: b0000 = 1 i-cache line b0001 = 2 i-cache lines b0010 = 3 i-cache lines ... b1111 = 16 i-cache lines.
3	RO	0x0	reserved
2:0	RO	0x7	The length of an i-cache line: b000-b001 = reserved b010 = 4 bytes b011 = 8 bytes b100 = 16 bytes b101 = 32 bytes b110-b111 = reserved

### DMAC\_CR2

Address: Operational Base + offset (0x0e08)

Configuration Register 2

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	Provides the value of boot_addr[31:0] when the DMAC exited from reset

### DMAC\_CR3

Address: Operational Base + offset (0x0e0c)

Configuration Register 3

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	Provides the security state of an event-interrupt resource: Bit [N] = 0 Assigns event<N> or irq[N] to the Secure state. Bit [N] = 1 Assigns event<N> or irq[N] to the Non-secure state.

### DMAC\_CR4

Address: Operational Base + offset (0x0e10)

Configuration Register 4

Bit	Attr	Reset Value	Description
31:0	RO	0x00000006	Provides the security state of the peripheral request interfaces: Bit [N] = 0 Assigns peripheral request interface N to the Secure state. Bit [N] = 1 Assigns peripheral request interface N to the Non-secure state

### DMAC\_CRDn

Address: Operational Base + offset (0x0e14)

DMA Configuration Register

Bit	Attr	Reset Value	Description
31:30	RO	0x0	reserved

Bit	Attr	Reset Value	Description
29:20	RO	0x020	The number of lines that the data buffer contains: b000000000 = 1 line b000000001 = 2 lines ... b111111111 = 1024 lines
19:16	RO	0x9	The depth of the read queue: b0000 = 1 line b0001 = 2 lines ... b1111 = 16 lines.
15	RO	0x0	reserved
14:12	RO	0x4	Read issuing capability that programs the number of outstanding read transactions: b000 = 1 b001 = 2 ... b111 = 8
11:8	RO	0x7	The depth of the write queue: b0000 = 1 line b0001 = 2 lines ... b1111 = 16 lines.
7	RO	0x0	reserved
6:4	RO	0x3	Write issuing capability that programs the number of outstanding write transactions: b000 = 1 b001 = 2 ... b111 = 8
3	RO	0x0	reserved
2:0	RO	0x3	The data bus width of the AXI interface: b000 = reserved b001 = reserved b010 = 32-bit b011 = 64-bit b100 = 128-bit b101-b111 = reserved.

**DMAC\_WD**

Address: Operational Base + offset (0x0e80)

DMA Watchdog Register

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved

Bit	Attr	Reset Value	Description
0	RW	0x0	Controls how the DMAC responds when it detects a lock-up condition: 0 = the DMAC aborts all of the contributing DMA channels and sets irq_abort HIGH 1 = the DMAC sets irq_abort HIGH.

## 13.5 Timing Diagram

Following picture shows the relationship between dma\_req and dma\_ack.

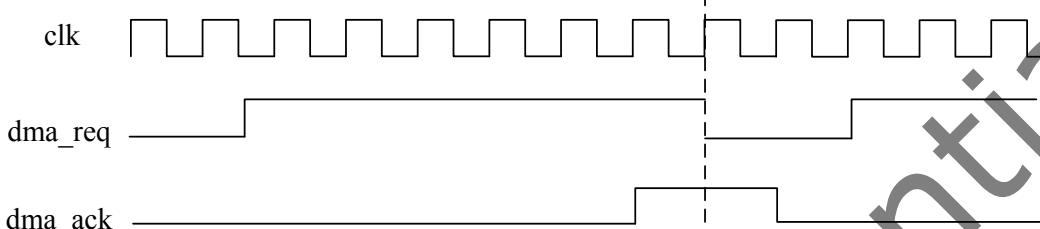


Fig. 13-3 DMAC request and acknowledge timing

## 13.6 Interface Description

DMAC has the following tie-off signals. It can be configured by GRF register. (Please refer to the chapter to find how to configure)

Table 13-2 DMAC boot interface

Interface	Reset value	Control source
boot_addr	0x0	GRF
boot_from_pc	0x0	GRF
boot_manager_ns	0x1	
boot_irq_ns	0xffff	
boot_periph_ns	0x3fff	

Notes: boot\_manager\_ns, boot\_irq\_ns and boot\_periph\_ns can't be configured, so dmac can work under non-secure state only

### boot\_addr

Configures the address location that contains the first instruction the DMAC executes, when it exits from reset.

### boot\_from\_pc

Controls the location in which the DMAC executes its initial instruction, after it exits from reset:

0 = DMAC waits for an instruction from either APB interface

1 = DMA manager thread executes the instruction that is located at the address that

### boot\_manager\_ns

When the DMAC exits from reset, this signal controls the security state of the DMA manager thread:

0 = assigns DMA manager to the Secure state

1 = assigns DMA manager to the Non-secure state.

### boot\_irq\_ns

Controls the security state of an event-interrupt resource, when the DMAC exits from reset:

boot\_irq\_ns[x] is LOW

The DMAC assigns event<x> or irq[x] to the Secure state.

boot\_irq\_ns[x] is HIGH

The DMAC assigns event<x> or irq[x] to the Non-secure state.

### boot\_periph\_ns

Controls the security state of a peripheral request interface, when the DMAC exits from reset:

boot\_periph\_ns[x] is LOW

The DMAC assigns peripheral request interface x to the Secure state.

boot\_periph\_ns[x] is HIGH

The DMAC assigns peripheral request interface x to the Non-secure state.

## 13.7 Application Notes

### 13.7.1 Using the APB slave interfaces

DMAC can work under non-secure state only, and the secure APB interface is not used. So only the non-secure APB interface can be used to start or restart a DMA channel.

The necessary steps to start a DMA channel thread using the debug instruction registers as following:

1. Create a program for the DMA channel.
2. Store the program in a region of system memory.
3. Poll the DBGSTATUS Register to ensure that debug is idle, that is, the dbgstatus bit is 0.
4. Write to the DBGINST0 Register and enter the:
  - Instruction byte 0 encoding for DMAGO.
  - Instruction byte 1 encoding for DMAGO.
  - Debug thread bit to 0. This selects the DMA manager thread.
5. Write to the DBGINST1 Register with the DMAGO instruction byte [5:2] data, see Debug Instruction-1 Register o. You must set these four bytes to the address of the first instruction in the program, that was written to system memory in step 2.
6. Writing zero to the DBGCMD Register. The DMAC starts the DMA channel thread and sets the dbgstatus bit to 1.

### 13.7.2 Security usage

#### DMA manager thread is in the Non-secure state

If the DNS bit is 1, the DMA manager thread operates in the Non-secure state, and it only performs non-secure instruction fetches. When a DMA manager thread in the Non-secure state processes:

##### DMAGO

The DMAC uses the status of the ns bit, to control if it starts a DMA channel thread. If:

ns = 0

The DMAC does not start a DMA channel thread and instead it:

1. Executes a NOP.
2. Sets the FSRD Register, see Fault Status DMA Manager
3. Sets the dmago\_err bit in the FTRD Register, see Fault Type DMA Manager Register.
4. Moves the DMA manager to the Faulting state.

ns = 1

The DMAC starts a DMA channel thread in the Non-secure state and programs the CNS bit to be non-secure.

##### DMAWFE

The DMAC uses the status of the corresponding INS bit, in the CR3 Register, to control if it waits for the event. If:

INS = 0

The event is in the Secure state. The DMAC:

1. Executes a NOP.
2. Sets the FSRD Register, see Fault Status DMA Manager Register.
3. Sets the mgr\_evnt\_err bit in the FTRD Register, see Fault Type DMA Manager Register.
4. Moves the DMA manager to the Faulting state.

INS = 1

The event is in the Non-secure state. The DMAC halts execution of the thread and waits for the event to occur.

##### DMASEV

The DMAC uses the status of the corresponding INS bit, in the CR3 Register, to control if it creates the event-interrupt. If:

INS = 0

The event-interrupt resource is in the Secure state. The DMAC:

1. Executes a NOP.

2. Sets the FSRD Register, see Fault Status DMA Manager Register.
3. Sets the mgr\_evnt\_err bit in the FTRD Register, see Fault Type DMA Manager Register.
4. Moves the DMA manager to the Faulting state.

INS = 1

The event-interrupt resource is in the Non-secure state. The DMAC creates the event-interrupt.

#### **DMA channel thread is in the Non-secure state**

When the CNS bit is 1, the DMA channel thread is programmed to operate in the Non-secure state and it only performs non-secure instruction fetches.

When a DMA channel thread in the Non-secure state processes the following instructions:

#### **DMAWFE**

The DMAC uses the status of the corresponding INS bit, in the CR3 Register, to control if it waits for the event. If:

INS = 0

The event is in the Secure state. The DMAC:

1. Executes a NOP.
2. Sets the appropriate bit in the FSRC Register that corresponds to the DMA channel number. See Fault Status DMA Channel Register.
3. Sets the ch\_evnt\_err bit in the FTRn Register, see Fault Type DMA Channel Registers.
4. Moves the DMA channel to the Faulting completing state.

INS = 1

The event is in the Non-secure state. The DMAC halts execution of the thread and waits for the event to occur.

#### **DMASEV**

The DMAC uses the status of the corresponding INS bit, in the CR3 Register, to control if it creates the event. If:

INS = 0

The event-interrupt resource is in the Secure state. The DMAC:

1. Executes a NOP.
2. Sets the appropriate bit in the FSRC Register that corresponds to the DMA channel number. See Fault Status DMA Channel Register.
3. Sets the ch\_evnt\_err bit in the FTRn Register, see Fault Type DMA Channel Registers .
4. Moves the DMA channel to the Faulting completing state.

INS = 1

The event-interrupt resource is in the Non-secure state. The DMAC creates the event-interrupt.

#### **DMAWFP**

The DMAC uses the status of the corresponding PNS bit, in the CR4 Register, to control if it waits for the peripheral to signal a request. If:

PNS = 0

The peripheral is in the Secure state. The DMAC:

1. Executes a NOP.
2. Sets the appropriate bit in the FSRC Register that corresponds to the DMA channel number. See Fault Status DMA Channel Register.
3. Sets the ch\_periph\_err bit in the FTRn Register, see Fault Type DMA Channel Registers.
4. Moves the DMA channel to the Faulting completing state.

PNS = 1

The peripheral is in the Non-secure state. The DMAC halts execution of the thread and waits for the peripheral to signal a request.

#### **DMALDP, DMASTP**

The DMAC uses the status of the corresponding PNS bit, in the CR4 Register, to control if it sends an acknowledgement to the peripheral. If:

PNS = 0

The peripheral is in the secure state. The DMAC:

1. Executes a NOP.
2. Sets the appropriate bit in the FSRC Register that corresponds to the DMA channel number. See Fault Status DMA Channel Register.

3. Sets the ch\_periph\_err bit in the FTRn Register, see Fault Type DMA Channel Registers.
4. Moves the DMA channel to the Faulting completing state.

PNS = 1

The peripheral is in the Non-secure state. The DMAC sends a message to the peripheral to communicate when the data transfer is complete.

#### **DMAFLUSHP**

The DMAC uses the status of the corresponding PNS bit, in the CR4 Register, to control if it sends a flush request to the peripheral. If:

PNS = 0

The peripheral is in the secure state. The DMAC:

1. Executes a NOP.
2. Sets the appropriate bit in the FSRC Register that corresponds to the DMA channel number. See Fault Status DMA Channel Register.
3. Sets the ch\_periph\_err bit in the FTRn Register, see Fault Type DMA Channel Registers.
4. Moves the DMA channel to the Faulting completing state.

PNS = 1

The peripheral is in the Non-secure state. The DMAC clears the state of the peripheral and sends a message to the peripheral to resend its level status.

When a DMA channel thread is in the Non-secure state, and a DMAMOV CCR instruction attempts to program the channel to perform a secure AXI transaction, the DMAC:

1. Executes a DMANOP.
2. Sets the appropriate bit in the FSRC Register that corresponds to the DMA channel number. See Fault Status DMA Channel Register.
3. Sets the ch\_rdwr\_err bit in the FTRn Register, see Fault Type DMA Channel Registers.
4. Moves the DMA channel thread to the Faulting completing state.

### **13.7.3 Programming restrictions**

#### **Fixed unaligned bursts**

The DMAC does not support fixed unaligned bursts. If you program the following conditions, the DMAC treats this as a programming error:

Unaligned read

- src\_inc field is 0 in the CCRn Register
- the SARn Register contains an address that is not aligned to the size of data that the src\_burst\_size field contain

Unaligned write

- dst\_inc field is 0 in the CCRn Register
- the DARn Register contains an address that is not aligned to the size of data that the dst\_burst\_size field contains

#### **Endian swap size restrictions**

If you program the endian\_swap\_size field in the CCRn Register, to enable a DMA channel to perform an endian swap then you must set the corresponding SARn Register and the corresponding DARn Register to contain an address that is aligned to the value that the endian\_swap\_size field contains.

#### **Updating DMA channel control registers during a DMA cycle restrictions**

Prior to the DMAC executing a sequence of DMA LD and DMA ST instructions, the values you program in to the CCRn Register, SARn Register, and DARn Register control the data byte lane manipulation that the DMAC performs when it transfers the data from the source address to the destination address. You'd better not update these registers during a DMA cycle.

#### **Resource sharing between DMA channels**

DMA channel programs share the MFIFO data storage resource. You must not start a set of concurrently running DMA channel programs with a resource requirement that exceeds the configured size of the MFIFO. If you exceed this limit then the DMAC might lock up and generate a Watchdog abort.

### **13.7.4 Unaligned transfers may be corrupted**

For a configuration with more than one channel, if any of channels 1 to 7 is performing transfers between certain types of misaligned source and destination addresses, then the output data may be corrupted by the action of channel 0.

Data corruption might occur if all of the following are true:

1. Two beats of AXI read data are received for one of channels 1 to 7.
2. Source and destination address alignments mean that each read data beat is split across two lines in the data buffer (see Splitting data, below).
3. There is one idle cycle between the two read data beats.
4. Channel 0 performs an operation that updates channel control information during this idle cycle (see Updates to channel control information, below)

### **Splitting data**

Depending upon the programmed values for the DMA transfer, one beat of read data from the AXI interface need to be split across two lines in the internal data buffer. This occurs when the read data beat contains data bytes which will be written to addresses that wrap around at the AXI interface data width, so that these bytes could not be transferred by a single AXI write data beat of the full interface width.

Most applications of DMA-330 do not split data in this way, so are NOT vulnerable to data corruption from this defect.

The following cases are NOT vulnerable to data corruption because they do not split data:

- Byte lane offset between source and destination addresses is 0 when source and destination addresses have the same byte lane alignment, the offset is 0 and a wrap operation that splits data cannot occur.
- Byte lane offset between source and destination addresses is a multiple of source size

Table 13-3 Source size in CCRn

<b>Source size in CCRn</b>	<b>Allowed offset between SARn and DARn</b>
SS8	any offset allowed.
SS16	0,2,4,6,8,10,12,14
SS32	0,4,8,12
SS64	0,8

### **13.7.5 Interrupt shares between channel**

As the DMAC does not record which channel (or list of channels) have asserted an interrupt. So it will depend on your program and whether any of the visible information for that program can be used to determine progress, and help identify the interrupt source.

There are 4 likely information sources that can be used to determine the progress made by a program:

- Program counter (PC)
- Source address
- Destination address
- Loop counters (LC)

For example, a program might emit an interrupt each time that it iterates around a loop. In this case, the interrupt service routine (ISR) would need to store the loop value of each channel when it is called, and then compare against the new value when it is next called. A change in value would indicate that the program has progressed.

The ISR must be carefully written to ensure that no interrupts are lost. The sequence of operations is as follows:

1. Disable interrupts
2. Immediately clear the interrupt in DMA-330
3. Check the relevant registers for both channels to determine which must be serviced
4. Take appropriate action for the channels
5. Re-enable interrupts and exit ISR

### **13.7.6 Instruction sets**

Table 13-4 DMAC Instruction sets

<b>Mnemonic</b>	<b>Instruction</b>	<b>Thread usage</b>
DMAADDH	Add Halfword	C
DMAEND	End	M/C
DMAFLUSHP	Flush and notify Peripheral	C
DMAGO	Go	M
DMAKILL	Kill	C

Mnemonic	Instruction	Thread usage
DMALD	Load	C
DMALDP	Load Peripheral	C
DMALP	Loop	C
DMALPEND	Loop End	C
DMALPFE	Loop Forever	C
DMAMOV	Move	C
DMANOP	No operation	M/C
DMARMB	Read Memory Barrier	C
DMASEV	Send Event	M/C
DMAST	Store	C
DMASTP	Store and notify Peripheral	C
DMASTZ	Store Zero	C
DMAWFE	Wait For Event M	M/C
DMAWFP	Wait For Peripheral	C
DMAWMB	Write Memory Barrier	C
DMAADNH	Add Negative Halfword	C

Notes: Thread usage: C=DMA channel, M=DMA manager

### 13.7.7 Assembler directives

In this document, only DMMADNH instruction is took as an example to show the way the instruction assembled. For the other instructions, please refer to pl330\_trm.pdf.

#### DMAADNH

Add Negative Halfword adds an immediate negative 16-bit value to the SARn Register or DARn Register, for the DMA channel thread. This enables the DMAC to support 2D DMA operations, or reading or writing an area of memory in a different order to naturally incrementing addresses. See Source Address Registers and Destination Address Registers.

The immediate unsigned 16-bit value is one-extended to 32 bits, to create a value that is the two's complement representation of a negative number between -65536 and -1, before the DMAC adds it to the address using 32-bit addition. The DMAC discards the carry bit so that addresses wrap from 0xFFFFFFFF to 0x00000000. The net effect is to subtract between 65536 and 1 from the current value in the Source or Destination Address Register.

Following table shows the instruction encoding.

Table 13-5 DMAC instruction encoding

Imm[15:8]	Imm[7:0]	0	1	0	1	1	1	ra	0
-----------	----------	---	---	---	---	---	---	----	---

#### Assembler syntax

DMAADNH <address\_register>, <16-bit immediate>

where:

<address\_register>

Selects the address register to use. It must be either:

SAR

SARn Register and sets ra to 0.

DAR

DARn Register and sets ra to 1.

<16-bit immediate>

The immediate value to be added to the <address\_register>.

You should specify the 16-bit immediate as the number that is to be represented in the instruction encoding. For example, DMAADNH DAR, 0xFFFF causes the value 0xFFFFFFF0 to be added to the current value of the Destination Address Register, effectively subtracting 16 from the DAR.

You can only use this instruction in a DMA channel thread.

### 13.7.8 MFIFO usage

For MFIFO usage, please refer to pl330\_trm.pdf

## Chapter 14 System Security

### 14.1 Overview

The RK3228 support the system security application requirement base on the TrustZone access control scheme. The following secure feature are supported

- Secure control of JTAG access
- Secure boot
- eight secure address space in DDR device, the start address and end address for each address scope is configurable, maximum 2GB secure address are supported
- 32K secure internal SRAM
- Many devices are configurable to act as secure or non-secure

### 14.2 Block Diagram

The following figure shows the system security architecture. All the devices which support security access are demonstrated in this figure.

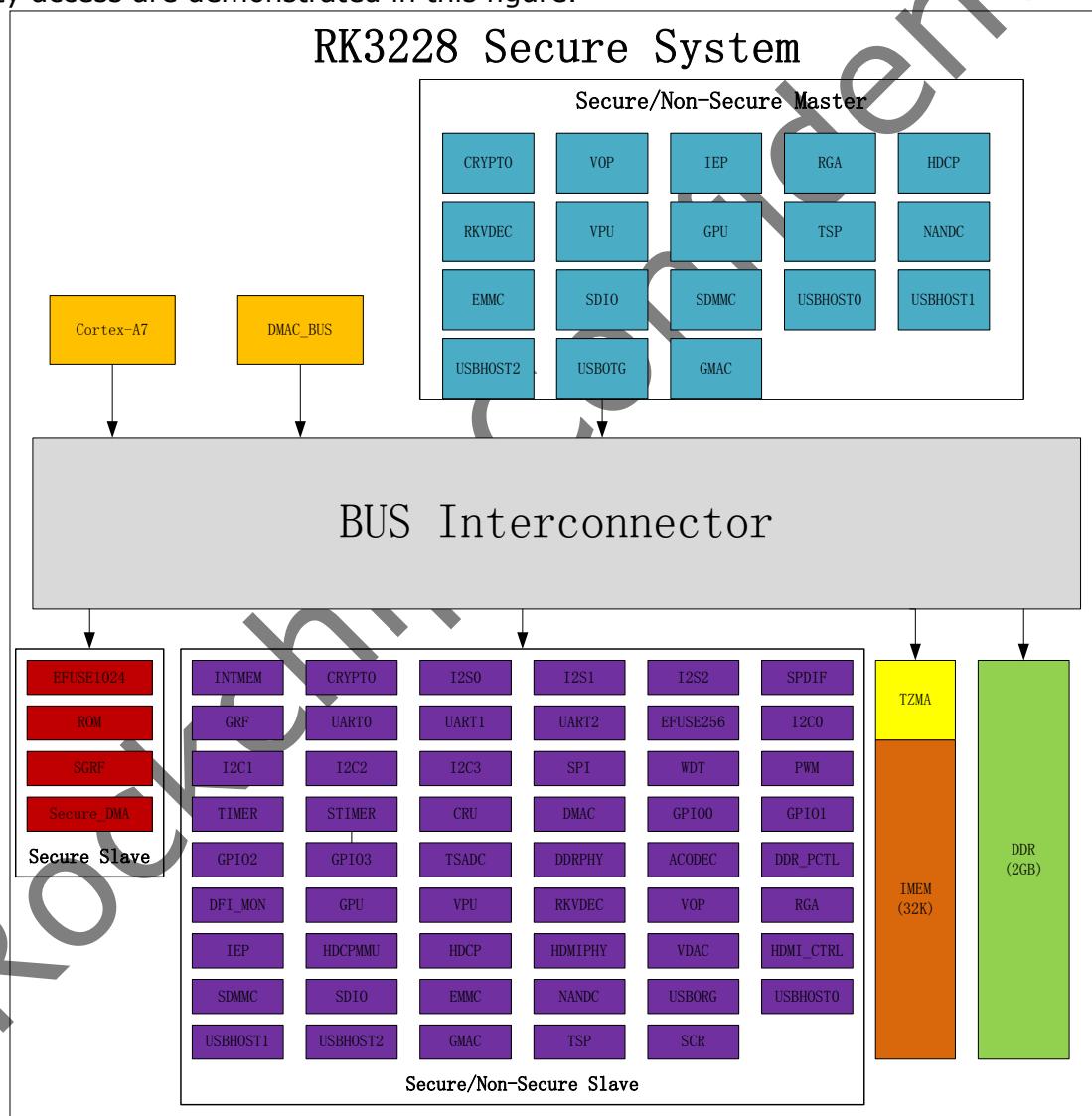


Fig. 14-1 RK3228 security architecture

### 14.3 Function Description

#### 14.3.1 Cortex-A7 Security Extension architecture

The processor implements the TrustZone Security Extensions architecture to facilitate the development of secure applications.

Security Extensions are based on these fundamental principles:

- The extensions define a class of core operation that you can switch between secure and non-secure state. Most code runs in non-secure state. Only trusted code runs in secure state
- The extensions define some memory as secure memory. When the core is in secure state, it can access secure memory
- Entry into secure state is strictly controlled
- Exit from secure state can only occur at programmed points
- Debug is strictly controlled
- The processor enters secure state on reset

### 14.3.2 TZMA

The TZMA (TrustZone Memory Adapter) is a bridge between AXI bus and Embedded SRAM, which support the flexible secure access by controlling R0SIZE port.

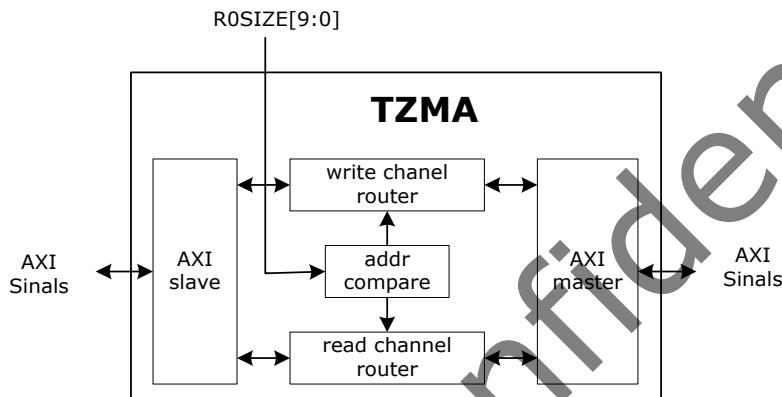


Fig. 14-2 TZMA block diagram

TZMA can support 0KB, 4KB, 8KB, 12KB... up to 32KB by 4KB step(the whole Embedded SRAM space) secure access by setting SGRF\_SOC\_CON0[9:0].

### 14.3.3 DMAC\_BUS secure access

The DMAC\_BUS is an AMBA compliant peripheral, which provides an AXI interface to perform the DMA transfer and two APB interfaces that control its operation. The DMAC\_BUS implements TrustZone secure technology with one APB interface operating in the secure state and the other operating in the Non-secure state. For the detailed description for DMAC\_BUS, please refer to Chapter 10.

The following diagram shows the interface of DMAC\_BUS.

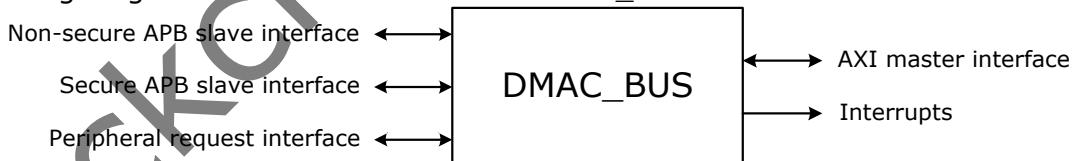


Fig. 14-3 DMAC\_BUS interface

The DMAC\_BUS support the secure feature in the following section:

- DMA manager thread
- DMA channel thread
- Event and interrupts
- Peripheral request interface

1. The security of DMA manager thread is controlled by input port boot\_manager\_ns.

0=assign DMA manager to the Secure state

1=assign DMA manager to the Non-secure state

2. The security of DMA channel thread is controlled by instruction DMAGO ns bit. If ns is present, DMA channel operation is in the non-secure state. Otherwise, the execution of the instruction depends on the security of the state of DMA manager:

DMA manager is in the secure state, DMA channel operates in the secure state

DMA manager is in the non-secure state, DMA abort

3. The security state of the event-interrupt source is controlled by the input port

boot\_irq\_ns[x:0], if boot\_irq\_ns[x] is LOW, the DMAC\_BUS assign event<x> or irq[x] to the secure state, otherwise the DMAC\_BUS assign event<x> or irq[x] to the non-secure state.

4. The security state of peripheral request interface is controlled by the input port boot\_irq\_ns[x], if boot\_irq\_ns[x] is IOW, the DMAC\_BUS assign peripheral interface x to secure state, otherwise the DMAC\_BUS assign peripheral interface x to non-secure state.

#### 14.3.4 Bus components security setting

The following table describes the security support on bus components which have the master interface or slave interface.

Table 14-1 bus components security setting

AXI master	secure	All transactions originating from this master interface are flagged as secure transaction and can access both secure and non-secure component.
	non-secure	All transactions originating from this master interface are flagged as non-secure transactions and cannot access secure component
	per access	The AxPROTx signal determines the security setting of each transaction, and the slave that it can access
	per configure	Can be configured act as secure or non-secure by Secure GRF register
AHB-Lite master	secure	All transactions originating from this master interface are flagged as secure transaction and can access both secure and non-secure component.
	non-secure	All transactions originating from this master interface are flagged as non-secure transactions and cannot access secure component
	per configure	Can be configured act as secure or non-secure by Secure GRF register
AXI slave	secure	Only secure transactions can access this component
	non-secure	Both secure and non-secure transactions can access this components
	boot-secure	You can use software to configure whether it permits secure and non-secure transactions to access component. When boot up, this component only can be accessed by secure transactions.
	per configure	Can be configured act as secure or non-secure by Secure GRF register
AHB slave	secure	Only secure transactions can access this components
	non-secure	Both secure and non-secure transactions can access this components
	boot-secure	You can use software to configure whether it permit secure and non-secure transactions to access this components. When boot up, this component only can be accessed by secure transactions.
	per configure	Can be configured act as secure or non-secure by Secure GRF register
APB slave	secure	Only secure transactions can access this components
	non-secure	Both secure and non-secure transactions can access this components

	boot-secure	You can use software to configure whether it permit secure and non-secure transactions to access this components. When boot up, this component only can be accessed by secure transactions.
	per configure	Can be configured act as secure or non-secure by Secure GRF register

Notes: When a non-secure master tries to access a secure slave, an error response will be returned. In RK3228, Cortex-A7 non-secure access a secure slave will cause a data-abort, and dmac\_bus non-secure access a secure slave will cause an interrupt for access error.

### 14.3.5 DDR region secure setting

RK3228 manage DDR secure for 8 regions. Each region can be configured as a couple of addresses: start address and end address. The granule of DDR region is 1MB(1MB aligned), and software can configure an enable bit for each region.

Then DDR region setting is shown below:

	<b>Start Address</b>	<b>End Address</b>	<b>Enable</b>
Region0	SGRF_DDR_CON0[12:0]	SGRF_DDR_CON1[12:0]	SGRF_DDR_CON0[14]
Region1	SGRF_DDR_CON2[12:0]	SGRF_DDR_CON3[12:0]	SGRF_DDR_CON2[14]
Region2	SGRF_DDR_CON4[12:0]	SGRF_DDR_CON5[12:0]	SGRF_DDR_CON4[14]
Region3	SGRF_DDR_CON6[12:0]	SGRF_DDR_CON7[12:0]	SGRF_DDR_CON6[14]
Region4	SGRF_DDR_CON8[12:0]	SGRF_DDR_CON9[12:0]	SGRF_DDR_CON8[14]
Region5	SGRF_DDR_CON10[12:0]	SGRF_DDR_CON11[12:0]	SGRF_DDR_CON10[14]
Region6	SGRF_DDR_CON12[12:0]	SGRF_DDR_CON13[12:0]	SGRF_DDR_CON12[14]
Region7	SGRF_DDR_CON14[12:0]	SGRF_DDR_CON15[12:0]	SGRF_DDR_CON14[14]

For certain master, a secure bit is correspond for each region, setting this bit 1 means the region is secure for this master and setting this bit 0 means non-secure. The secure bit for each master is shown below:

<b>Master</b>	<b>Secure bits</b>
Cortex-A7	SGRF_DDR_CON16[7:0]
GPU	SGRF_DDR_CON16[15:8]
BUS_DMAC	SGRF_DDR_CON17[7:0]
CRYPTO	SGRF_DDR_CON17[15:8]
PERI_MASTERS(EMMC/SDIO/SDMMC/USB)	SGRF_DDR_CON18[7:0]
GMAC	SGRF_DDR_CON18[15:8]
NANDC	SGRF_DDR_CON19[7:0]
VOP	SGRF_DDR_CON19[15:8]
IEP	SGRF_DDR_CON20[7:0]
HDCP	SGRF_DDR_CON20[15:8]
RGA	SGRF_DDR_CON21[7:0]
VPU	SGRF_DDR_CON21[15:8]
RKVDEC	SGRF_DDR_CON22[7:0]

### 14.3.6 RK3228 secure device setting

Table 14-2 RK3228 secure device setting

<b>Cortex-A7</b>	<b>AXI master</b>	<b>Per access</b>
secure_grf	APB slave	Secure
eFuse1024	APB slave	Secure
Embedded SRAM	AXI slave	Controlled by TZMA, the secure space can be set to 0, 4KB, 8KB, 12KB ...up to 32KB
DMAC_BUS	AXI master	Per access
	Secure APB slave	Secure

Cortex-A7	AXI master	Per access
	Non-secure APB slave	Non-secure
GPU	AXI slave	per configure
	AXI master	per configure
Crypto	AHB slave	per configure
	AHB master	per configure
IEP	AXI master	per configure
	AHB slave	per configure
VPU	AHB slave	per configure
	AXI master	per configure
RKVDEC	AHB slave	per configure
	AXI master	per configure
VOP	AHB slave	per configure
	AXI master	per configure
RGA	AHB slave	per configure
	AXI master	per configure
HDCP	AHB slave	per configure
	AHB master	per configure
NANDC	AHB slave	per configure
	AHB master	per configure
TSP	AHB slave	per configure
	AHB master	per configure
EMMC	AHB slave	per configure
	AHB master	per configure
SDIO	AHB slave	per configure
	AHB master	per configure
SDMMC	AHB slave	per configure
	AHB master	per configure
USBOTG	AHB slave	per configure
	AHB master	per configure
USBHOST0/1/2	AHB slave	per configure
	AHB master	per configure
GMAC	AHB slave	per configure
	AHB master	per configure
I2S0/1/2	AHB slave	per configure
SPDIF	AHB slave	per configure
GRF	APB slave	per configure
UART0/1/2	APB slave	per configure
EFUSE256	APB slave	per configure
I2C0/1/2	APB slave	per configure
SPI	APB slave	per configure
WDT	APB slave	per configure
PWM	APB slave	per configure
TIMER/STIMER	APB slave	per configure
CRU	APB slave	per configure
GPIO0/1/2/3	APB slave	per configure
TSADC	APB slave	per configure
DDRPHY	APB slave	per configure
ACODEC	APB slave	per configure
DDR_PCTL	APB slave	per configure
DFI_MON	APB slave	per configure
SCR	APB slave	per configure
DDR	AXI slave	Support eight secure address scope, the start address and end address for

Cortex-A7	AXI master	Per access
		each address scope is configurable

### 14.3.7 RK3228 device secure input port setting

The following table lists all the secure input ports for the secure device. These secure input ports could be set by configuring SGRF registers.

Table 14-3 RK3228 device secure input port setting

Input Port	Module	Function description
boot_manager_ns	DMAC_BUS	When the DMAC exits from reset, this signal controls the security state of the DMAManager thread: 1'b0: assigns DMA manager to the secure state 1'b1: assigns DMA manager to the non-secure state
boot_periph_ns[2:0]	DMAC_BUS	Controls the security state of a peripheral request interface, when the DMAC exits from reset: boot_periph_ns[x] is LOW The DMAC assigns peripheral request interface x to the secure state boot_periph_ns[x] is HIGH The DMAC assigns peripheral request interface x to the non-secure state.
boot_irq_ns[7:0]	DMAC_BUS	Controls the security state of an event-interrupt resource, when the DMAC exits fromreset: boot_irq_ns[x] is LOW The DMAC assigns event<x> or irq[x] to the secure state. boot_irq_ns[x] is HIGH The DMAC assigns event<x> or irq[x] to the non-secure state.

### 14.3.8 Secure JTAG

The JTAG access is controlled by system secure control register, the following table show the detailed information about the secure control to JTAG.

JTAG Type	Register
CA7 JTAG	SGRF_SOC_CON3[2] SGRF_SOC_CON3[3] SGRF_SOC_CON3[4] SGRF_SOC_CON3[5]

## 14.4 Application Notes

### Secure software conception

The basis of the security extensions model is that the computing environment splits into two isolated states, the secure state and the non-secure state, with no leakage of secure data to the non-secure state. Software secure monitor code, running in the monitor mode, links the two states and acts as a gatekeeper to manage program flow. The system can have both secure and non-secure peripherals that are suitable to secure and non-secure device driver control. Following figure shows the relationship between the secure and non-secure states. The operating system (OS) splits into the secure OS, that includes the secure kernel, and the non-secure OS, that includes the non-secure kernel.

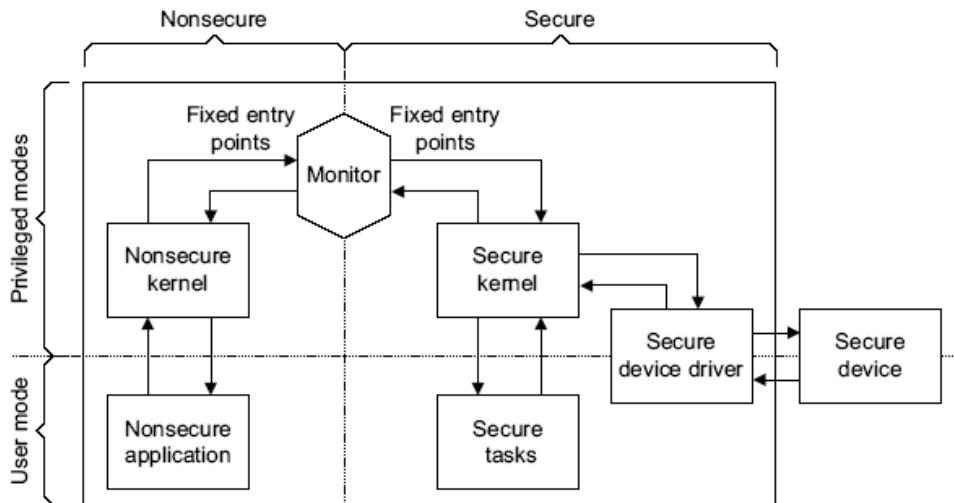


Fig. 14-4 Software Diagram of Secure and Non-secure

In normal non-secure operation, the OS runs tasks in the usual way. When a user process requires secure execution it makes a request to the secure kernel, that operates in privileged mode. This then calls the secure monitor to transfer execution to the secure state.

This approach to secure systems means that the platform OS that works in the non-secure state, has only a few fixed entry points into the secure state through the secure monitor. The trusted code base for the secure state, that includes the secure kernel and secure device drivers, is small and therefore much easier to maintain and verify.

#### Secure/Non-secure memory space for Embedded SRAM

The following figure gives an example of embedded SRAM secure/non-secure memory space setting. The software configures 4KB secure space. The bottom 4KB space will act as secure space and the other 28K space will be non-secure space.

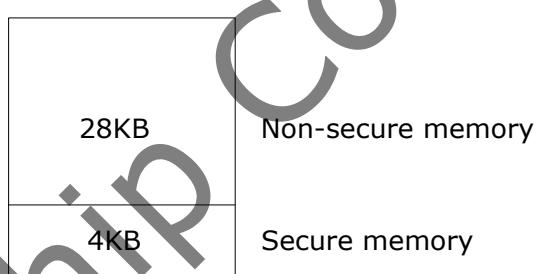


Fig. 14-5 Embedded SRAM secure memory space setting

## Chapter 15 EFUSE

### 15.1 Overview

In RK3228, there are two eFuse. One is organized as 32 bits by 8 one-time programmable electrical fuses with random access interface, and the other is organized as 32bits by 32 one-time programmable electrical fuses.

The eFuse can only be accessed by APB bus at secure mode.

It is a type of non-volatile memory fabricated in standard CMOS logic process. The main features are as follows:

- Programming condition :  $VQPS\_EFUSE = 1.5V \pm 10\%$
- Program time :  $10\mu s \pm 0.2\mu s$  .
- Read condition :  $VQPS\_EFUSE = 0V$
- Provide standby mode

### 15.2 Block Diagram

In the following diagram, all the signals except power supply VDD\_EFUSE, VSS\_EFUSE and VQPS\_EFUSE are controlled by registers. For detailed description, please refer to detailed register descriptions.

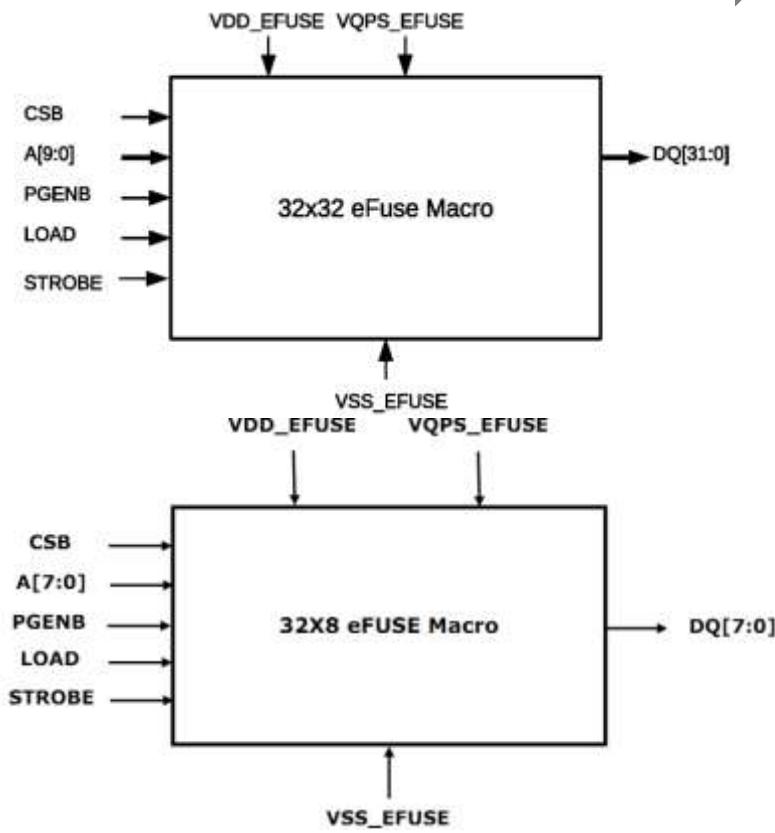


Fig. 15-1 RK3228 eFuse block diagram

### 15.3 Function Description

eFuse has three operation modes. They are defined as standby, read and programming.

#### Program (PGM) Mode

In order to enter programming mode, the following conditions need to be satisfied:  
 VQPS\_EFUSE is at high voltage, LOAD signal is low, PGGENB signal is low, and CSB signal is low.  
 All bits can be individually programmed (one at a time) with the proper address selected, the STROBE signal high and the address bits satisfying setup and hold time with respect to STROBE.

#### Read Mode

In order to enter read mode the following conditions need to be satisfied: VQPS\_EFUSE is at

ground, the LOAD signal is high, the PGENB signal is high, and the CSB is low. An entire 8-bit word of data can be read in one read operation with STROBE being high and a proper address selected (address signals A5~A7/A9 are “don’t cares”).

### Standby Mode

Standby is defined when the macro is not being programmed or read. The conditions for standby mode are: the LOAD signal is low, the STROBE signal is low, the CSB signal is high and PGENB is high.

Table 15-1 list of allowed modes

Signals/Supplies					Mode
VQPS_EFUSE	CSB	PGENB	LOAD	STROBE*	
High	Low	Low	Low	Low to High	Programming
Low	Low	High	High	Low to High	Read
Low	High	High	Low	Low	Standby

## 15.4 Register Description

### 15.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
EFUSE_CTRL	0x0000	W	0x00000000	efuse control register
EFUSE_DOUT	0x0004	W	0x00000000	efuse data out register

Notes:Size: **B**- Byte (8 bits) access, **HW**- Half WORD (16 bits) access, **W**-WORD (32 bits) access

### 15.4.2 Detail Register Description

#### EFUSE\_CTRL

Address: Operational Base + offset (0x0000)  
eFuse control register

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:6	RW	0x00	efuse_addr efuse address pins : A[7:0] / A[9:0]
5:4	RO	0x0	reserved
3	RW	0x0	efuse_pgenb efuse program enable (active low) : PGENB
2	RW	0x0	efuse_load efuse turn on sense amplifier and load data into latch (active high) : LOAD
1	RW	0x0	efuse_strobe efuse turn on the array for read or program access (active high) : STROBE
0	RW	0x0	efuse_csb efuse chip select enable signal, active low : CSB

#### EFUSE\_DOUT

Address: Operational Base + offset (0x0004)  
eFuse data out register

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
[31:0]/[7:0]	RO	0x00	efuse_dout efuse data output

## 15.5 Timing Diagram

- When efuse32×8 is in program(PGM) mode.

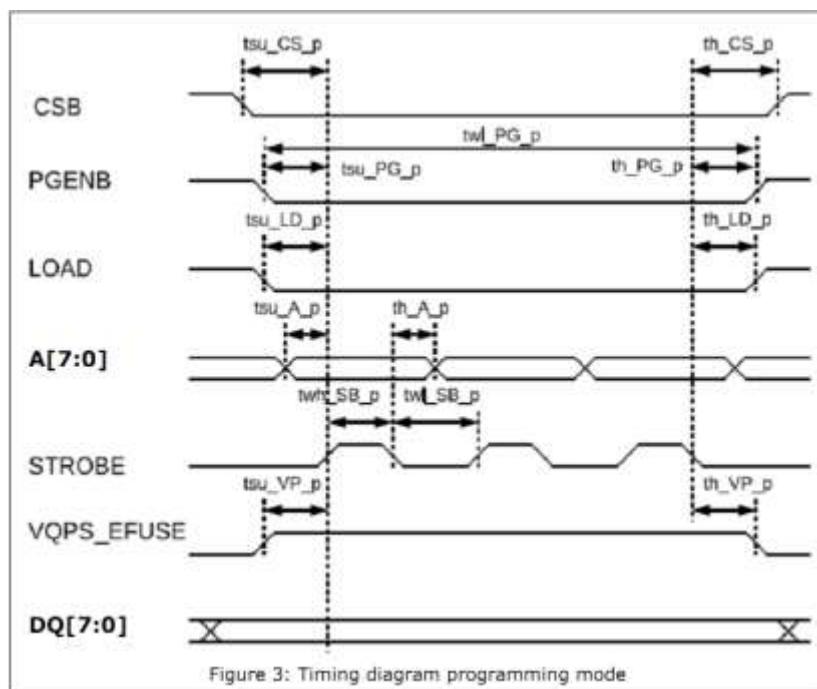


Fig. 15-2 RK3228 efuse32×8 timing diagram in program mode

- When efuse32×8 is in read mode.

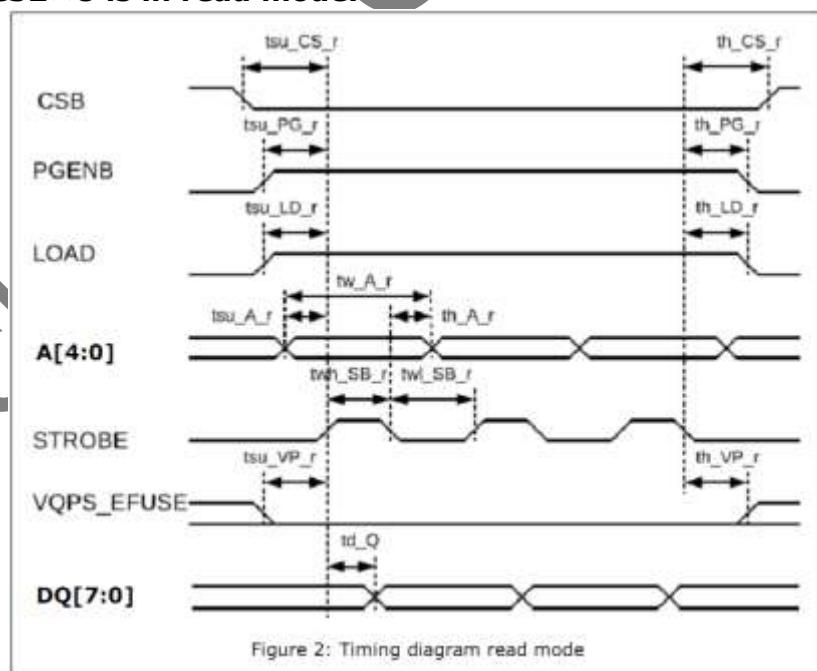


Fig. 15-3 RK3228 efuse32×8 timing diagram in read mode

- When efuse32×32 is in program(PGM) mode.

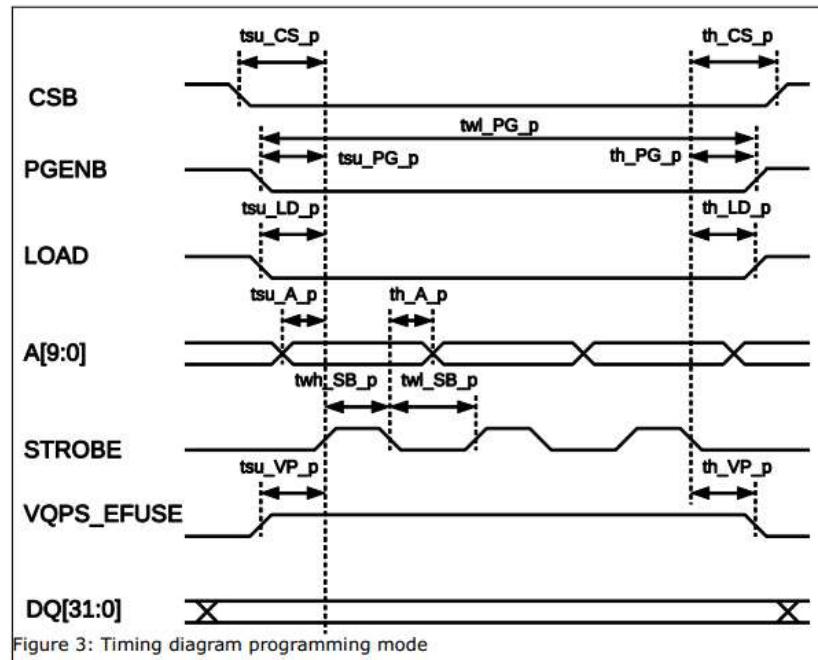


Fig. 15-4 RK3228 efuse32×32 timing diagram in program mode

- When efuse32×32 is in read mode.

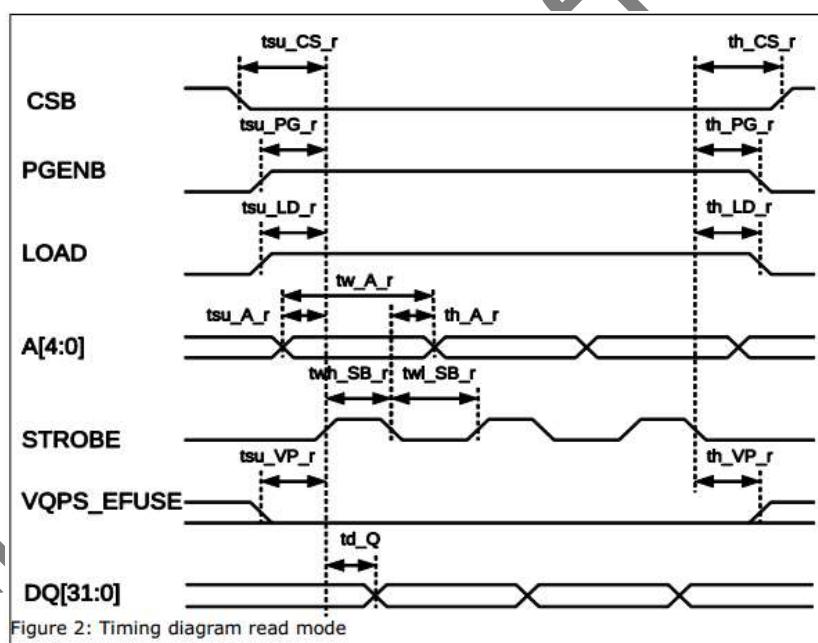


Fig. 15-5 RK3228 efuse32×32 timing diagram in read mode

The following table has shows the detailed value for timing parameters in the above diagram.

Table 15-2 RK3228 eFuse timing parameters list

Mode	Item	Description	Min	Typ	Max	Unit
Read Mode	twh_SB_r	Pulse width high of STROBE read strobe	20	-	-	ns
	twl_SB_r	Pulse width low of STROBE read strobe	15	-	-	ns
	tsu_A_r	A[7:0] to STROBE setup time in read mode	25	-	-	ns
	th_A_r	A[7:0] to STROBE hold time in read mode	3	-	-	ns
	tw_A_r	A[7:0] pulse width while LOAD high in read mode	48	-	100	ns
	tsu_CS_r	CSB to STROBE setup time in read mode	16	-	-	ns
	th_CS_r	CSB to STROBE hold time in read mode	6	-	-	ns
	tsu_PG_r	PGENB to STROBE setup time in read mode	14	-	-	ns
	th_PG_r	PGENB to STROBE hold time in read mode	10	-	-	ns
	tsu_LD_r	LOAD to STROBE setup time in read mode	10	-	-	ns
	th_LD_r	LOAD to STROBE hold time in read mode	7	-	-	ns
	tsu_VP_r	VQPS_EFUSE to STROBE setup time in read mode	20	-	-	ns
	th_VP_r	VQPS_EFUSE to STROBE hold time in read mode	20	-	-	ns
	td_Q	DQ[7:0] delay time after STROBE high	0	-	8	ns
PGM Mode	twh_SB_p	Pulse width high of STROBE PGM strobe	9.8	10	10.2	us
	twl_SB_p	Pulse width low of STROBE PGM strobe	15	-	-	ns
	tsu_A_p	A[7:0] to STROBE setup time in PGM mode	12	-	-	ns
	th_A_p	A[7:0] to STROBE hold time in PGM mode	3	-	-	ns
	tsu_CS_p	CSB to STROBE setup time in PGM mode	16	-	-	ns
	th_CS_p	CSB to STROBE hold time in PGM mode	6	-	-	ns
	tsu_PG_p	PGENB to STROBE setup time in PGM mode	14	-	-	ns
	th_PG_p	PGENB to STROBE hold time in PGM mode	10	-	-	ns
	twl_PG_p	PGENB pulse width low (cumulative) in PGM mode	-	-	100	ms
	tsu_LD_p	LOAD to STROBE setup time in PGM mode	10	-	-	ns
	th_LD_p	LOAD to STROBE hold time in PGM mode	7	-	-	ns
	tsu_VP_p	VQPS_EFUSE to STROBE setup time in PGM mode	20	-	-	ns
	th_VP_p	VQPS_EFUSE to STROBE hold time in PGM mode	20	-	-	ns

## 15.6 Application Notes

During usage of efuse, customers must pay more attention to the following items:

1. In condition of program(PGM) mode, VQPS\_EFUSE=  $1.5V \pm 10\%$ .
2. Q0~Q7/Q31 will be reset to "0" once CSB at high.
3. No data access allowed at the rising edge of CSB.
4. All the program timing for each signal must be more than the value defined in the timing table.
5. When programming the EFUSE32×32, SGRF\_SOC\_CON6[1] (SGRF\_EFUSE\_PRG\_EN) should be set to 1.

## Chapter 16 WatchDog

### 15.1 Overview

Watchdog Timer (WDT) is an APB slave peripheral that can be used to prevent system lockup that may be caused by conflicting parts or programs in a SoC. The WDT would generate interrupt or reset signal when its counter reaches zero, then a reset controller would reset the system.

WDT supports the following features:

- 32 bits APB bus width
- WDT counter's clock is pclk
- 32 bits WDT counter width
- Counter counts down from a preset value to 0 to indicate the occurrence of a timeout
- WDT can perform two types of operations when timeout occurs:
  - Generate a system reset
  - First generate an interrupt and if this is not cleared by the service routine by the time a second timeout occurs then generate a system reset
- Programmable reset pulse length
- Total 16 defined ranges of main timeout period

### 15.2 Block Diagram

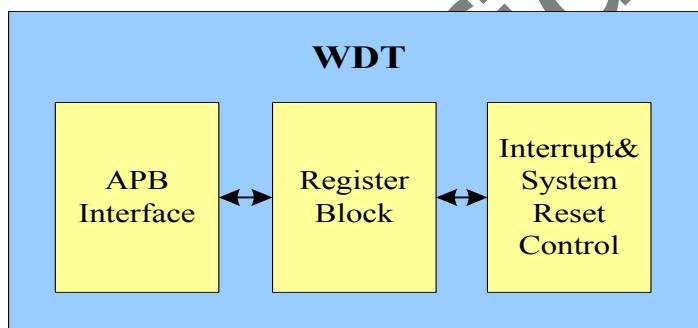


Fig. 16-1 WDT block diagram

#### Block Descriptions:

- APB Interface

The APB Interface implements the APB slave operation. Its data bus width is 32 bits.

- Register Block

A register block that reads coherence for the current count register.

- Interrupt & system reset control

An interrupt/system reset generation block is comprised of a decrementing counter and control logic.

### 15.3 Function Description

#### 15.3.1 Operation

##### Counter

The WDT counts from a preset (timeout) value in descending order to zero. When the counter reaches zero, depending on the output response mode selected, either a system reset or an interrupt occurs. When the counter reaches zero, it wraps to the selected timeout value and continues decrementing. The user can restart the counter to its initial value. This is programmed by writing to the restart register at any time. The process of restarting the watchdog counter is sometimes referred to as kicking the dog. As a safety feature to prevent accidental restarts, the value 0x76 must be written to the Current Counter Value Register (WDT\_CRR).

##### Interrupts

The WDT can be programmed to generate an interrupt (and then a system reset) when a timeout occurs. When a 1 is written to the response mode field (RMOD, bit 1) of the Watchdog

Timer Control Register (WDT\_CR), the WDT generates an interrupt. If it is not cleared by the time a second timeout occurs, then it generates a system reset. If a restart occurs at the same time the watchdog counter reaches zero, an interrupt is not generated.

### System Resets

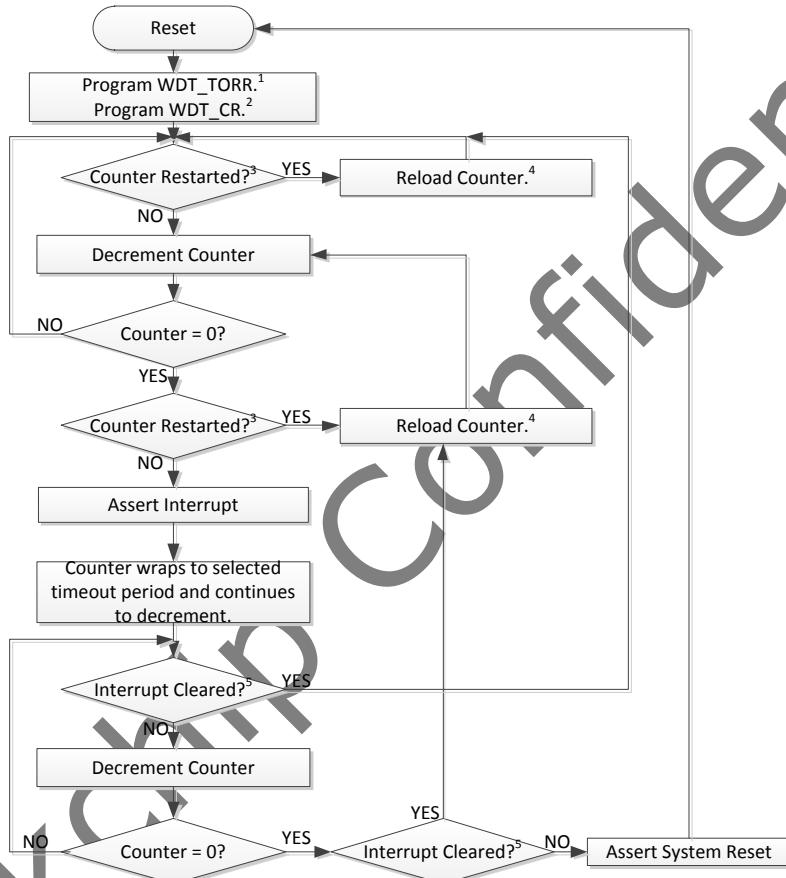
When a 0 is written to the output response mode field (RMOD, bit 1) of the Watchdog Timer Control Register (WDT\_CR), the WDT generates a system reset when a timeout occurs.

### Reset Pulse Length

The reset pulse length is the number of pclk cycles for which a system reset is asserted. When a system reset is generated, it remains asserted for the number of cycles specified by the reset pulse length or until the system is reset. A counter restart has no effect on the system reset once it has been asserted.

### 15.3.2 Programming sequence

#### Operation Flow Chart (Response mode=1)



1. Select required timeout period.
2. Set reset pulse length, response mode, and enable WDT.
3. Write 0x76 to WDT\_CRR.
4. Starts back to selected timeout period.
5. Can clear by reading WDT\_EOI or restarting (kicking) the counter by writing 0x76 to WDT\_CRR.

Fig. 16-2 WDT Operation Flow

### 15.4 Register Description

This section describes the control/status registers of the design.

#### 15.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
WDT_CR	0x0000	W	0x0000000a	Control Register
WDT_TORR	0x0004	W	0x00000000	Timeout range Register
WDT_CCVR	0x0008	W	0x00000000	Current counter value Register

Name	Offset	Size	Reset Value	Description
WDT_CRR	0x000c	W	0x00000000	Counter restart Register
WDT_STAT	0x0010	W	0x00000000	Interrupt status Register
WDT_EOI	0x0014	W	0x00000000	Interrupt clear Register

Notes: **S**-ize: **B**- Byte (8 bits) access, **H**W- Half WORD (16 bits) access, **W**-WORD (32 bits) access

### 15.4.2 Detail Register Description

#### WDT\_CR

Address: Operational Base + offset (0x0000)

Control Register

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4:2	RW	0x2	<p>rst_pluse_lenth Reset pulse length. This is used to select the number of pclk cycles for which the system reset stays asserted.</p> <p>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</p>
1	RW	0x1	<p>resp_mode Response mode. Selects the output response generated to a timeout.</p> <p>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.</p>
0	RW	0x0	wdt_en WDT enable 0: WDT disabled; 1: WDT enabled.

#### WDT\_TORR

Address: Operational Base + offset (0x0004)

Timeout range Register

Bit	Attr	Reset Value	Description
31:4	RO	0x0	reserved

Bit	Attr	Reset Value	Description
3:0	RW	0x0	<p>timeout_period Timeout period.</p> <p>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).</p> <p>The range of values available for a 32-bit watchdog counter are:</p> <ul style="list-style-type: none"> <li>0000: 0x0000ffff</li> <li>0001: 0x0001ffff</li> <li>0010: 0x0003ffff</li> <li>0011: 0x0007ffff</li> <li>0100: 0x000fffff</li> <li>0101: 0x001fffff</li> <li>0110: 0x003fffff</li> <li>0111: 0x007fffff</li> <li>1000: 0x00ffffff</li> <li>1001: 0x01ffffff</li> <li>1010: 0x03ffffff</li> <li>1011: 0x07ffffff</li> <li>1100: 0x0fffffff</li> <li>1101: 0x1fffffff</li> <li>1110: 0x3fffffff</li> <li>1111: 0x7fffffff</li> </ul>

**WDT\_CCVR**

Address: Operational Base + offset (0x0008)

Current counter value Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	<p>cur_cnt Current counter value</p> <p>This register, when read, is the current value of the internal counter. This value is read coherently whenever it is read</p>

**WDT\_CRR**

Address: Operational Base + offset (0x000c)

Counter restart Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	W1C	0x00	<p>cnt_restart Counter restart</p> <p>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. Reading this register returns zero.</p>

**WDT\_STAT**

Address: Operational Base + offset (0x0010)

Interrupt status Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:1	RO	0x0	reserved
0	RO	0x0	wdt_status This register shows the interrupt status of the WDT. 1: Interrupt is active regardless of polarity; 0: Interrupt is inactive.

**WDT\_EOI**

Address: Operational Base + offset (0x0014)

Interrupt clear Register

<b>Bit</b>	<b>Attr</b>	<b>Reset Value</b>	<b>Description</b>
31:1	RO	0x0	reserved
0	RC	0x0	wdt_int_clr Clears the watchdog interrupt. This can be used to clear the interrupt without restarting the watchdog counter.

**15.5 Application Notes**

Please refer to the function description section

## Chapter 17 System Debug

### 17.1 Overview

The chip uses the DAPLITE Technology to support real-time debug.

#### 17.1.1 Features

- Invasive debug with core halted
- SW-DP

#### 17.1.2 Debug components address map

The following table shows the debug components address in memory map:

Module	Base Address
DAP_ROM	0x20020000

### 17.2 Block Diagram

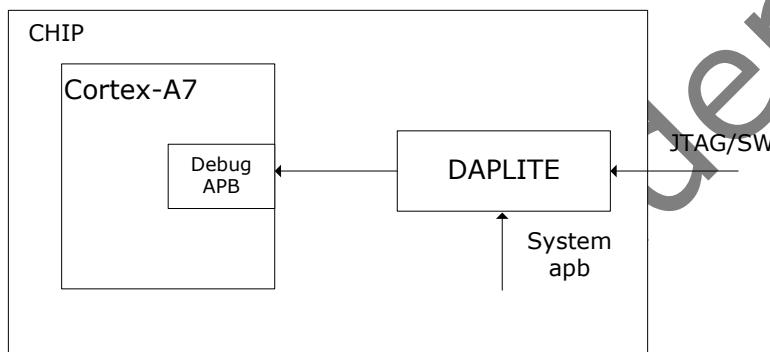


Fig. 17-1 Debug system structure

### 17.3 Function Description

#### 17.3.1 DAP

The DAP has following components:

- Serial Wire JTAG Debug Port(SWJ-DP)
- APB Access Port(APB-AP)
- ROM table

The debug port is the host tools interface to access the DAP-Lite. This interface controls any access ports provided within the DAP-Lite. The DAP-Lite supports a combined debug port which includes both JTAG and Serial Wire Debug(SWD), with a mechanism that supports switching between them.

The APB-AP acts as a bridge between SWJ-DP and APB bus which translate the Debug request to APB bus.

The DAP provides an internal ROM table connected to the master Debug APB port of the APB-Mux. The Debug ROM table is loaded at address 0x00000000 and 0x80000000 of this bus and is accessible from both APB-AP and the system APB input. Bit[31] of the address bus is not connected to the ROM Table, ensuring that both views read the same value. The ROM table stores the locations of the components on the Debug APB.

More information please refer to the documentCoreSight\_DAPLite\_TRM.pdf for the debug detail description.

### 17.4 Register Description

Please refer to the documentCoreSight\_DAPLite\_TRM.pdf for the debug detail description.

## 17.5 Interface Description

### 17.5.1 DAP SWJ-DP Interface

The following figure is the DAP SWJ-DP interface, the SWJ-DP is a combined JTAG-DP and SW-DP that enable you connect either a Serial Wire Debug(SWJ) to JTAG probe to a target.

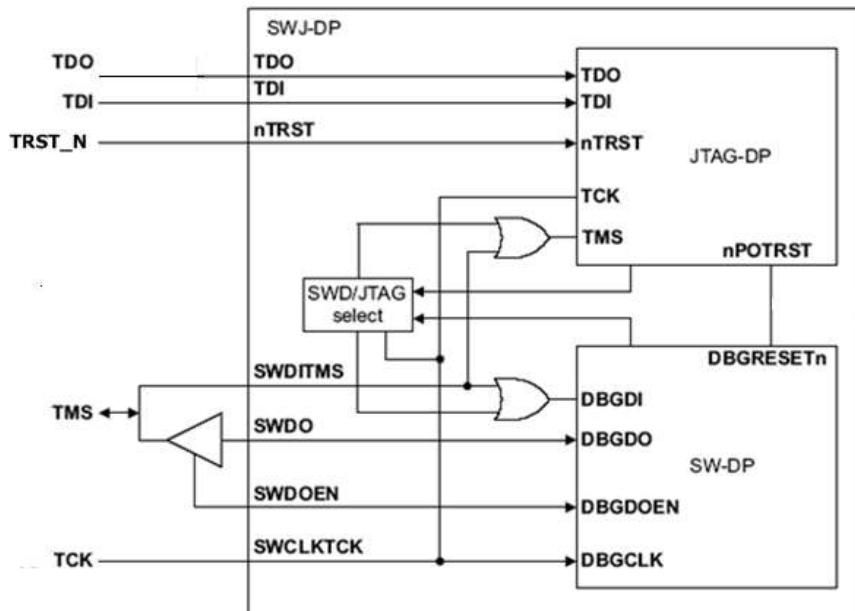


Fig. 17-2 DAP SWJ interface

### 17.5.2 DAP SW-DP Interface

This implementation is taken from ADIv5.1 and operates with a synchronous serial interface. This uses a single bidirectional data signal, and a clock signal.

The figure below describes the interaction between the timing of transactions on the serialwire interface, and the DAP internal bus transfers. It shows when the target responds with a WAIT acknowledgement.

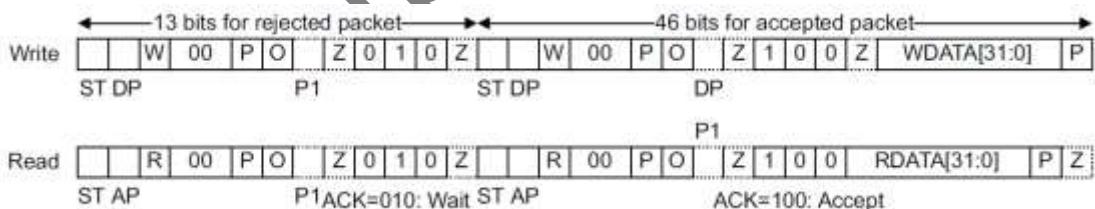


Fig. 17-3 SW-DP acknowledgement timing

Table 17-1 SW-DP Interface Description

Module pin	Direction	Pad name	IOMUX
jtag_tck	I	IO_SDMMCd2_JTAGtck_GPIO1c4	GRF_GPIO1C_IOMUX[9:8]=2'b10& mmc0_detn
jtag_tm s	I/O	IO_SDMMCd3_JTAGtms_GPIO1c5	GRF_GPIO1C_IOMUX[11:10]=2'b10 & mmc0_detn

Note : mmc0\_detn, when high, no sd card is used.