

Rockchip
RK3036
Technical Reference Manual
Peripheral and Interface

Revision 1.0
Sep. 2015

Revision History

Date	Revision	Description
2015-9-7	1.0	Update-Part2 Peripheral and Interface

Rockchip Confidential

Table of Content

Table of Content	3
Figure Index	6
Table Index.....	8
Warranty Disclaimer.....	9
Chapter 1 Mobile Storage Host Controller.....	10
1.1 Overview	10
1.2 Block Diagram	10
1.3 Function Description	11
1.4 Register Description.....	25
1.5 Interface Description	45
1.6 Application Notes	46
Chapter 2 USB OTG 2.0	68
2.1 Overview	68
2.2 Block Diagram	68
2.3 USB OTG2.0 Controller.....	70
2.4 USB OTG2.0 PHY.....	73
2.5 UART BYPASS FUNCITON	76
2.6 Register Description	78
2.7 Interface description.....	186
2.8 Application Note.....	186
Chapter 3 USB Host 2.0.....	187
3.1 Overview	187
3.2 Block Diagram	187
3.3 USB Host2.0 Controller	187
3.4 USB Host2.0 PHY	187
3.5 Register Description.....	187
3.6 Interface description.....	187
3.7 Application Note.....	188
Chapter 4 HDMI TX	189
4.1 Overview	189
4.2 Block Diagram	189
4.3 Function Description	189
4.4 Register Description.....	193
4.5 Interface Description	234
4.6 Application Notes	235
Chapter 5 Pulse Width Modulation (PWM).....	246
5.1 Overview	246
5.2 Block Diagram	246
5.3 Function Description	246
5.4 Register Description.....	248
5.5 Interface Description	260
5.6 Application Notes	260
Chapter 6 UART	262
6.1 Overview	262
6.2 Block Diagram	262
6.3 Function Description	263
6.4 Register Description.....	265
6.5 Interface Description	282
6.6 Application Notes	283
6.7 None FIFO Mode Transfer Flow	283

6.8 FIFO Mode Transfer Flow	283
6.9 Baud Rate Calculation	284
Chapter 7 GPIO	286
7.1 Overview	286
7.2 Block Diagram	286
7.3 Function Description	286
7.4 Register Description.....	288
7.5 Interface Description	291
7.6 Application Notes	292
Chapter 8 I2C Interface	293
8.1 Overview	293
8.2 Block Diagram	293
8.3 Function Description	293
8.4 Register Description	296
8.5 Interface Description	304
8.6 Application Notes	305
Chapter 9 I2S/PCM Controller (8 channel).....	308
9.1 Overview	308
9.2 Block Diagram	308
9.3 Function description	309
9.4 Register Description	312
9.5 Interface description.....	320
9.6 Application Notes	321
Chapter 10 Serial Peripheral Interface (SPI)	323
10.1 Overview.....	323
10.2 Block Diagram	323
10.3 Function Description	324
10.4 Register Description.....	326
10.5 Interface Description.....	336
10.6 Application Notes.....	336
Chapter 11 SPDIF transmitter.....	339
11.1 Overview.....	339
11.2 Block Diagram	339
11.3 Function description.....	340
11.4 Register description	342
11.5 Interface description	347
11.6 Application Notes.....	348
Chapter 12 MAC Ethernet Interface	350
12.1 Overview.....	350
12.2 Block Diagram	350
12.3 Register Description.....	352
12.4 Interface Description.....	361
12.5 Application Notes.....	361
Chapter 13 Audio Codec	367
13.1 Overview.....	367
13.2 Block Diagram	367
13.3 Function Description	367
13.4 Register Description.....	370
Chapter 14 SFC (Serial Flash Controller).....	377
14.1 Overview.....	377
14.2 Block Diagram	377

14.3 Function Description	377
14.4 Register Description.....	378
14.5 Interface Description.....	384
14.6 Application Notes.....	385

Rockchip Confidential

Figure Index

Fig. 1-1 Host Controller Block Diagram	11
Fig. 1-2 SD/MMC Card-Detect Signal	14
Fig. 1-3 Host Controller Command Path State Machine.....	16
Fig. 1-4 Host Controller Data Transmit State Machine	18
Fig. 1-5 Host Controller Data Receive State Machine	20
Fig. 1-6 SD/MMC Card-Detect and Write-Protect	46
Fig. 1-7 SD/MMC Card Termination.....	47
Fig. 1-8 Host Controller Initialization Sequence	49
Fig. 1-9 Voltage Switching Command Flow Diagram.....	58
Fig. 1-10 ACMD41 Argument.....	59
Fig. 1-11 ACMD41 Response(R3)	59
Fig. 1-12 Voltage Switch Normal Scenario	59
Fig. 1-13 Voltage Switch Error Scenario	60
Fig. 1-14 CASES for eMMC 4.5 START bit	62
Fig. 1-15 Clock Generation Unit	64
Fig. 1-16 Card Detection Method 2	66
Fig. 1-17 Card Detection Method 4	67
Fig. 2-1 USB OTG 2.0 Architecture	68
Fig. 2-2 UTMI interface – Transmit timing for a data packe	69
Fig. 2-3 UTMI interface – Receive timing for a data packet.....	70
Fig. 2-4 USB OTG2.0 Controller Architecture.....	70
Fig. 2-5 DFIFO single-port synchronous SRAM interface	71
Fig. 2-6 USB OTG 2.0 Controller host mode FIFO address mapping	72
Fig. 2-7 USB OTG 2.0 Controller device mode FIFO address mapping	73
Fig. 2-8 usb phy architecture.....	74
Fig. 2-9 UART Application	77
Fig. 2-10 UART Timing Sequence	77
Fig. 3-1 USB HOST 2.0 Architecture	187
Fig. 4-1 HDMI TX Block Diagram.....	189
Fig. 4-2 HDMI TX Video Data Processing	190
Fig. 4-3 HDMI TX Video Processing Timing	190
Fig. 4-4 HDMI TX Audio Data Processing Diagram	191
Fig. 4-5 HDMI TX Audio Clock Regeneration Model.....	192
Fig. 4-6 VOP Connect to HDMI TX Diagram.....	234
Fig. 4-7 I2S Connect to HDMI TX Diagram.....	235
Fig. 4-8 SPDIF Connect to HDMI TX Diagram	235
Fig. 5-1 PWM Block Diagram	246
Fig. 5-2 PWM Capture Mode	247
Fig. 5-3 PWM Continuous Left-aligned Output Mode	247
Fig. 5-4 PWM Continuous Center-aligned Output Mode	247
Fig. 5-5 PWM One-shot Center-aligned Output Mode.....	248
Fig. 6-1 UART Architecture.....	262
Fig. 6-2 UART Serial protocol.....	263
Fig. 6-3 IrDA 1.0	263
Fig. 6-4 UART baud rate	263
Fig. 6-5 UART Auto flow control block diagram	264
Fig. 6-6 UART AUTO RTS TIMING	265
Fig. 6-7 UART AUTO CTS TIMING	265
Fig. 6-8 UART none fifo mode.....	283
Fig. 6-9 UART fifo mode.....	284
Fig. 6-10 UART clock generation	284
Fig. 7-1 GPIO block diagram	286
Fig. 7-2 GPIO Interrupt RTL Block Diagram	287
Fig. 8-1 I2C architecture.....	293
Fig. 8-2 I2C DATA Validity	295

Fig. 8-3 I2C Start and stop conditions	295
Fig. 8-4 I2C Acknowledge	296
Fig. 8-5 I2C byte transfer	296
Fig. 8-6 I2C Flow chat for transmit only mode	305
Fig. 8-7 I2C Flow chat for receive only mode	306
Fig. 8-8 I2C Flow chat for mix mode.....	307
Fig. 9-1 I2S/PCM controller (8 channel) Block Diagram	308
Fig. 9-2 I2S transmitter-master & receiver-slave condition	309
Fig. 9-3 I2S transmitter-slave& receiver-master condition	309
Fig. 9-4 I2S normal mode timing format	310
Fig. 9-5 I2S left justified mode timing format	310
Fig. 9-6 I2S right justified mode timing format	310
Fig. 9-7 PCM early mode timing format.....	310
Fig. 9-8 PCM late1 mode timing format.....	311
Fig. 9-9 PCM late2 mode timing format.....	311
Fig. 9-10 PCM late3 mode timing format	311
Fig. 9-11 I2S/PCM controller transmit operation flow chart	321
Fig. 9-12 I2S/PCM controller receive operation flow chart	322
Fig. 10-1 SPI Controller Block diagram	324
Fig. 10-2 SPI Master and Slave Interconnection	324
Fig. 10-3 SPI Format (SCPH=0 SCPOL=0).....	325
Fig. 10-4 SPI Format (SCPH=0 SCPOL=1).....	325
Fig. 10-5 SPI Format (SCPH=1 SCPOL=0).....	326
Fig. 10-6 SPI Format (SCPH=1 SCPOL=1).....	326
Fig. 10-7 SPI Master transfer flow diagram.....	337
Fig. 10-8 SPI Slave transfer flow diagram.....	338
Fig. 11-1 SPDIF transmitter Block Diagram.....	339
Fig. 11-2 SPDIF Frame Format	340
Fig. 11-3 SPDIF Sub-frame Format.....	340
Fig. 11-4 SPDIF Channel Coding	341
Fig. 11-5 SPDIF Preamble	341
Fig. 11-6 Format of Data-burst	342
Fig. 11-7 SPDIF transmitter operation flow chart.....	348
Fig. 12-1 VMAC Block Diagram	350
Fig. 12-2 VMAC Block Diagram	351
Fig. 12-3 RMII transmission in 100Mb/s mode	351
Fig. 12-4 RMII reception with no errors in 100Mb/s mode.....	351
Fig. 12-5 VMAC buffer chain.....	362
Fig. 12-6 VMAC transmit buffer descriptor written by CPU	362
Fig. 12-7 VMAC transmit buffer descriptor written by VMAC	363
Fig. 12-8 VMAC receive buffer descriptor written by VMAC.....	364
Fig. 12-9 VMAC receive buffer descriptor written by CPU	365
Fig. 13-1 Audio Codec Block Diagram	367
Fig. 13-2 Left Justified Mode (assuming n-bit word length)	368
Fig. 13-3 Right Justified Mode (assuming n-bit word length)	368
Fig. 13-4 I2S Mode (assuming n-bit word length).....	369
Fig. 13-5 DSP/PCM Mode A (assuming n-bit word length)	369
Fig. 13-6 DSP/PCM Mode B (assuming n-bit word length)	370
Fig. 13-7 DC-blocking capacitor	370
Fig. 13-8 DAC Channel Relationship	370
Fig. 14-1 SFC architecture	377
Fig. 14-2 idle cycles	378
Fig. 14-3 SPI mode.....	378
Fig. 14-4 slave mode read	385
Fig. 14-5 master mode flow	386

Table Index

Table 1-1 Bits in Interrupt Status Register	13
Table 1-2 Auto-Stop Generation	21
Table 1-3 Non-data Transfer Commands and Requirements	22
Table 1-4 SDMMC Interface Description	45
Table 1-5 SDIO Interface Description	45
Table 1-6 EMMC Interface Description	45
Table 1-7 Recommended Usage of use_hold_reg	48
Table 1-8 Command Settings for No-Data Command	51
Table 1-9 Command Setting for Single or Multiple-Block Read	53
Table 1-10 Command Settings for Single or Multiple-Block Write	54
Table 1-11 PBL and Watermark Levels	63
Table 1-12 Configuration for SDMMC Clock Generation	64
Table 1-13 Configuration for SDIO Clock Generation	64
Table 1-14 Configuration for EMMC Clock Generation	65
Table 1-15 Register for SDMMC Card Detection Method 3	67
Table 2-1 USB OTG 2.0 Interface Description	186
Table 3-1 USB HOST 2.0 Interface Description	187
Table 4-1 HDMI TX Supported Input Video Formats	190
Table 4-2 HDMI TX I2S 2 Channel Audio Sampling Frequency at Each Video Format	191
Table 4-3 HDMI TX I2S 8 Channel Audio Sampling Frequency at Each Video Format	191
Table 4-4 HDMI TX SPDIF Sampling Frequency at Each Video Format	191
Table 5-1 PWM Interface Description	260
Table 6-1 UART Interface Description	282
Table 6-2 UART baud rate configuration	285
Table 7-1 GPIO interface description	291
Table 8-1 I2C Interface Description	304
Table 9-1 I2S Interface Description	320
Table 10-1 1SPI interface description	336
Table 11-1 SPDIF Interface Description	347
Table 12-1 IOMUX Settings for RMII/MII Interface	361
Table 12-2 VMAC-tx buffer descriptor for CPU	363
Table 12-3 VMAC-tx buffer descriptor for VMAC	363
Table 12-4 VMAC-rx buffer descriptor for VMAC	364
Table 12-5 VMAC-rx buffer descriptor for CPU	365
Table 13-1 Supported Data Formats in Different Modes	367
Table 13-2 Signals between I2S and ACODEC	370
Table 14-1 1SPI interface description	384

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 Mobile Storage Host Controller

1.1 Overview

The Mobile Storage Host Controller is designed to support Secure Digital memory (SD- max version 3.01) with 1 bits or 4 bits data width, Multimedia Card(MMC-max version 4.51) with 1 bits or 4 bits or 8 bits data width.

The Host Controller is instantiated for SDMMC, SDIO EMMC in RK3036. The interface difference between these instances is shown in "Interface Description".

The Host Controller supports following features:

- Bus Interface Features:
 - Support AMBA AHB interface for master and slave
 - In addition to AMBA slave interface, support external DMA controller for data transfers
 - Support combined single FIFO for both transmit and receive operations
 - Support FIFO size of 256x32
 - Support FIFO over-run and under-run prevention by stopping card clock
- Card Interface Features:
 - Support Secure Digital memory protocol commands
 - Support Secure Digital I/O protocol commands
 - Support Multimedia Card protocol commands
 - Support Command Completion Signal and interrupts to host
 - Support CRC generation and error detection
 - Support programmable baud rate
 - Support power management and power switch
 - Support card detection
 - Support write protection
 - Support hardware reset
 - Support SDIO interrupts in 1-bit and 4-bit modes
 - Support 4-bit mode in SDIO3.0
 - Support SDIO suspend and resume operation
 - Support SDIO read wait
 - Support block size of 1 to 65,535 bytes
 - Support 1-bit, 4-bit and 8-bit SDR modes
 - Support 4-bit DDR,8-bit DDR, as defined by SD3.0 and MMC4.41
 - Support boot in 1-bit, 4-bit and 8-bit SDR modes
 - Support Packed Commands, CMD21, CMD49
- Clock Interface Features:
 - Support 0/90/180/270-degree phase shift operation for sample clock(cclk_in_sample) and drive clock(cclk_in_drv) relative to function clock(cclk_in) respectively
 - Support phase tuning using delay line for sample clock(cclk_in_sample) and drive clock(cclk_in_drv) relative to function clock (cclk_in) respectively. The max number of delay element number is 256.

1.2 Block Diagram

The Host Controller consists of the following main functional blocks.

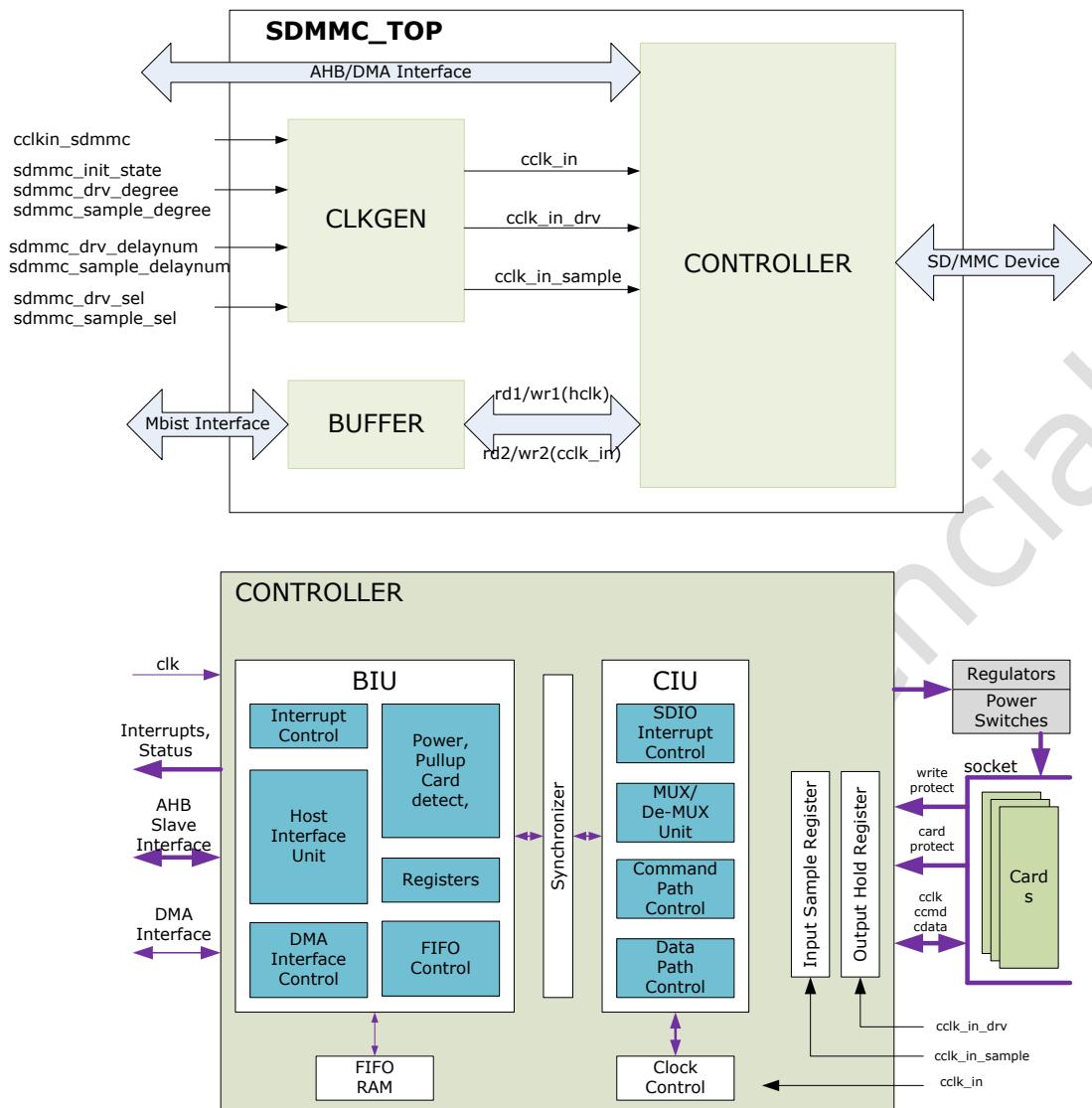


Fig. 1-1 Host Controller Block Diagram

- Clock Generate Unit(CLKGEN): generates card interface clock **cclk_in**/
cclk_sample/**cclk_drv** based on **cclkin_sdmmc** and configuration information.
- Asynchronous dual-port memory(BUFFER): Uses a two-clock synchronous read and
synchronous write dual-port RAM. One of the ports is connected to the host clock, and the
second port is connected to the card clock.
- Bus Interface Unit (BIU): Provides AMBA AHB interfaces for register and data read/writes.
- Card Interface Unit (CIU): Takes care of the SD/MMC protocols and provides clock
management.

1.3 Function Description

1.3.1 Bus Interface Unit

The Bus Interface Unit provides the following functions:

- Host interface
- Interrupt control
- Register access
- External FIFO access
- Power control and card detection

1. Host Interface Unit

The Host Interface Unit is an AHB slave interface, which provides the interface between the SD/MMC card and the host bus.

2. Register Unit

The register unit is part of the bus interface unit; it provides read and write access to the registers.

All registers reside in the Bus Interface Unit clock domain. When a command is sent to a card by setting the start_bit, which is bit[31] of the CMD register, all relevant registers needed for the CIU operation are transferred to the CIU block. During this time, the registers that are transferred from the BIU to the CIU should not be written. The software should wait for the hardware to clear the start bit before writing to these registers again. The register unit has a hardware locking feature to prevent illegal writes to registers. The lock is necessary in order to avoid metastability violations, both because the host and card clock domains are different and to prevent illegal software operations.

Once a command start is issued by setting the start_bit of the CMD register, the following registers cannot be reprogrammed until the command is accepted by the card interface unit:

- CMD – Command
- CMDARG – Command Argument
- BYTCNT – Byte Count
- BLKSIZ – Block Size
- CLKDIV – Clock Divider
- CLKENA – Clock Enable
- CLKSRC – Clock Source
- TMOUT – Timeout
- CTYPE – Card Type

The hardware resets the start_bit once the CIU accepts the command. If a host write to any of these registers is attempted during this locked time, then the write is ignored and the hardware lock error bit is set in the raw interrupt status register. Additionally, if the interrupt is enabled and not masked for a hardware lock error, then an interrupt is sent to the host.

When the Card Interface Unit is in an idle state, it typically takes the following number of clocks for the command handshake, where clk is the BIU clock and cclk_in is the CIU clock:
3 (clk) + 3 (cclk_in)

Once a command is accepted, you can send another command to the CIU-which has a one-deep command queue-under the following conditions:

- If the previous command was not a data transfer command, the new command is sent to the SD/MMC card once the previous command completes.
- If the previous command is a data transfer command and if wait_prvdata_complete (bit[13]) of the Command register is set for the new command, the new command is sent to the SD/MMC card only when the data transfer completes.
- If the wait_prvdata_complete is 0, then the new command is sent to the SD/MMC card as soon as the previous command is sent. Typically, you should use this only to stop or abort a previous data transfer or query the card status in the middle of a data transfer.

3. Interrupt Controller Unit

The interrupt controller unit generates an interrupt that depends on the controller raw interrupt status, the interrupt-mask register, and the global interrupt-enable register bit.

Once an interrupt condition is detected, it sets the corresponding interrupt bit in the raw interrupt status register. The raw interrupt status bit stays on until the software clears the bit by writing a 1 to the interrupt bit; a 0 leaves the bit untouched.

The interrupt port, int, is an active-high, level-sensitive interrupt. The interrupt port is active only when any bit in the raw interrupt status register is active, the corresponding interrupt mask bit is 1, and the global interrupt enable bit is 1. The interrupt port is registered in order to avoid any combinational glitches.

The int_enable is reset to 0 on power-on, and the interrupt mask bits are set to 32'h0, which masks all the interrupts.

Notes:

Before enabling the interrupt, it is always recommended that you write 32'hffff_ffff to the raw interrupt status register in order to clear any pending unserviced interrupts. When clearing interrupts during normal operation, ensure that you clear only the interrupt bits that you serviced.

The SDIO Interrupts, Receive FIFO Data Request (RXDR), and Transmit FIFO Data Request (TXDR) are set by level-sensitive interrupt sources. Therefore, the interrupt source should be first cleared before you can clear the interrupt bit of the Raw Interrupt register. For example, on seeing the Receive FIFO Data Request (RXDR) interrupt, the FIFO should be emptied so that the "FIFO count greater than the RX-Watermark" condition, which triggers the interrupt, becomes inactive. The rest of the interrupts are triggered by a single

clock-pulse-width source.

Table 1-1 Bits in Interrupt Status Register

Bits	Interrupt	Description
24	sdio_interrupt	Interrupt from SDIO card. In MMC-Ver3.3-only mode, these bits are always 0
16	Card no-busy	If card exit busy status, the interrupt happened
15	End Bit Error (read) /Write no CRC (EBE)	Error in end-bit during read operation, or no data CRC or negative CRC received during write operation. <i>Notes: For MMC CMD19, there may be no CRC status returned by the card. Hence, EBE is set for CMD19. The application should not treat this as an error.</i>
14	Auto Command Done (ACD)	Stop/abort commands automatically sent by card unit and not initiated by host; similar to Command Done (CD) interrupt.
13	Start Bit Error (SBE)	Error in data start bit when data is read from a card. In 4-bit mode, if all data bits do not have start bit, then this error is set.
12	Hardware Locked write Error (HLE)	During hardware-lock period, write attempted to one of locked registers.
11	FIFO Underrun/ Overrun Error (FRUN)	Host tried to push data when FIFO was full, or host tried to read data when FIFO was empty. Typically this should not happen, except due to error in software. Card unit never pushes data into FIFO when FIFO is full, and pop data when FIFO is empty.
10	Data Starvation by Host Timeout (HTO)	To avoid data loss, card clock out (cclk_out) is stopped if FIFO is empty when writing to card, or FIFO is full when reading from card. Whenever card clock is stopped to avoid data loss, data-starvation timeout counter is started with data-timeout value. This interrupt is set if host does not fill data into FIFO during write to card, or does not read from FIFO during read from card before timeout period. Even after timeout, card clock stays in stopped state, with CIU state machines waiting. It is responsibility of host to push or pop data into FIFO upon interrupt, which automatically restarts cclk_out and card state machines. Even if host wants to send stop/abort command, it still needs to ensure it has to push or pop FIFO so that clock starts in order for stop/abort command to send on cmd signal along with data that is sent or received on data line.
9	Data Read Timeout (DRTO)	Data timeout occurred. Data Transfer Over (DTO) also set if data timeout occurs.
8	Response Timeout (RTO)	Response timeout occurred. Command Done (CD) also set if response timeout occurs. If command involves data transfer and when response times out, no data transfer is attempted by Host Controller.
7	Data CRC Error (DCRC)	Received Data CRC does not match with locally-generated CRC in CIU.
6	Response CRC Error (RCRC)	Response CRC does not match with locally-generated CRC in CIU.
5	Receive FIFO Data Request (RXDR)	Interrupt set during read operation from card when FIFO level is greater than Receive-Threshold level.
4	Transmit FIFO Data Request (TXDR)	Interrupt set during write operation to card when FIFO level reaches less than or equal to Transmit-Threshold level.

Bits	Interrupt	Description
3	Data Transfer Over (DTO)	Data transfer completed, even if there is Start Bit Error or CRC error. This bit is also set when "read data-timeout" occurs. <i>Notes: DTO bit is set at the end of the last data block, even if the device asserts MMC busy after the last data block.</i>
2	Command Done(CD)	Command sent to card and got response from card, even if Response Error or CRC error occurs. Also set when response timeout occurs
1	Response Error (RE)	Error in received response set if one of following occurs: <ul style="list-style-type: none">● Transmission bit != 0● Command index mismatch● End-bit != 1
0	Card-Detect (CDT)	When card inserted or removed, this interrupt occurs. Software should read card-detect register (CDETECT, 0x50) to determine current card status.

4. FIFO Controller Unit

The FIFO controller interfaces the external FIFO to the host interface and the card controller unit. When FIFO overrun and under-run conditions occur, the card clock stops in order to avoid data loss.

The FIFO uses a two-clock synchronous read and synchronous write dual-port RAM. One of the ports is connected to the host clock, clk, and the second port is connected to the card clock, cclk_in.

Notes: The FIFO controller does not support simultaneous read/write access from the same port. For debugging purposes, the software may try to write into the FIFO and read back the data; results are indeterminate, since the design does not support read/write access from the same port.

5. Power Control and Card Detection Unit

The register unit has registers that control the power. Power to each card can be selectively turned on or off.

The card detection unit looks for any changes in the card-detect signals for card insertion or card removal. It filters out the debounces associated with mechanical insertion or removal, and generates one interrupt to the host. You can program the debounce filter value.

On power-on, the controller should read in the card_detect port and store the value in the memory. Upon receiving a card-detect interrupt, it should again read the card_detect port and XOR with the previous card-detect status to find out which card has interrupted. If more than one card is simultaneously removed or inserted, there is only one card-detect interrupt; the XOR value indicates which cards have been disturbed. The memory should be updated with the new card-detect value.

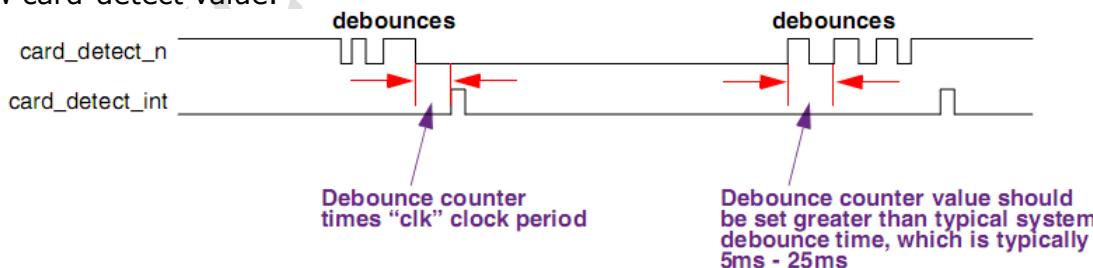


Fig. 1-2 SD/MMC Card-Detect Signal

6. DMA Interface Unit

DMA signals interface the Host Controller to an external DMA controller to reduce the software overhead during FIFO data transfers. The DMA request/acknowledge handshake is used for only data transfers. The DMA interface provides a connection to the DMA Controller.

On seeing the DMA request, the DMA controller initiates accesses through the host interface to read or write into the data FIFO. The Host Controller has FIFO transmit/receive watermark registers that you can set, depending on system latency. The DMA interface asserts the request in the following cases:

- Read from a card when the data FIFO word count exceeds the Rx-Watermark level
- Write to a card when the FIFO word count is less than or equal to the Tx-Watermark level

When the DMA interface is enabled, you can use normal host read/write to access the data FIFO.

1.3.2 Card Interface Unit

The Card Interface Unit (CIU) interfaces with the Bus Interface Unit (BIU) and the devices. The host writes command parameters to the BIU control registers, and these parameters are then passed to the CIU. Depending on control register values, the CIU generates SD/MMC command and data traffic on a selected card bus according to SD/MMC protocol. The Host Controller accordingly controls the command and data path.

The following software restrictions should be met for proper CIU operation:

- Only one data transfer command can be issued at a time.
- During an open-ended card write operation, if the card clock is stopped because the FIFO is empty, the software must first fill the data into the FIFO and start the card clock. It can then issue only a stop/abort command to the card.
- When issuing card reset commands (CMD0, CMD15 or CMD52_reset) while a card data transfer is in progress, the software must set the stop_abort_cmd bit in the Command register so that the Host Controller can stop the data transfer after issuing the card reset command.
- When the data end bit error is set in the RINTSTS register, the Host Controller does not guarantee SDIO interrupts. The software should ignore the SDIO interrupts and issue the stop/abort command to the card, so that the card stops sending the read data.
- If the card clock is stopped because the FIFO is full during a card read, the software should read at least two FIFO locations to start the card clock.

The CIU block consists of the following primary functional blocks:

- Command path
- Data path
- SDIO interrupt control
- Clock control
- Mux/demux unit

1. Command Path

The command path performs the following functions:

- Loads clock parameters
- Loads card command parameters
- Sends commands to card bus (ccmd_out line)
- Receives responses from card bus (ccmd_in line)
- Sends responses to BIU
- Drives the P-bit on command line

A new command is issued to the Host Controller by programming the BIU registers and setting the start_cmd bit in the Command register. The BIU asserts start_cmd, which indicates that a new command is issued to the SD/MMC device. The command path loads this new command (command, command argument, timeout) and sends acknowledge to the BIU by asserting cmd_taken.

Once the new command is loaded, the command path state machine sends a command to the device bus-including the internally generated CRC7-and receives a response, if any. The state machine then sends the received response and signals to the BIU that the command is done, and then waits for eight clocks before loading a new command.

Load Command Parameters

One of the following commands or responses is loaded in the command path:

- New command from BIU – When start_cmd is asserted, then the start_cmd bit is set in the Command register.
- Internally-generated auto-stop command – When the data path ends, the stop command request is loaded.
- IRQ response with RCA 0x000 – When the command path is waiting for an IRQ response from the MMC card and a “send irq response” request is signaled by the BIU, then the send_irq_response bit is set in the control register.

Loading a new command from the BIU in the command path depends on the following Command register bit settings:

- update_clock_registers_only – If this bit is set in the Command register, the command

path updates only the clock enable, clock divider, and clock source registers. If this bit is not set, the command path loads the command, command argument, and timeout registers; it then starts processing the new command.

- **wait_prvdata_complete** – If this bit is set, the command path loads the new command under one of the following conditions:
 - Immediately, if the data path is free (that is, there is no data transfer in progress), or if an open-ended data transfer is in progress (byte_count = 0).
 - After completion of the current data transfer, if a predefined data transfer is in progress.

Send Command and Receive Response

Once a new command is loaded in the command path, update_clock_registers_only bit is unset – the command path state machine sends out a command on the device bus; the command path state machine is illustrated in following figure.

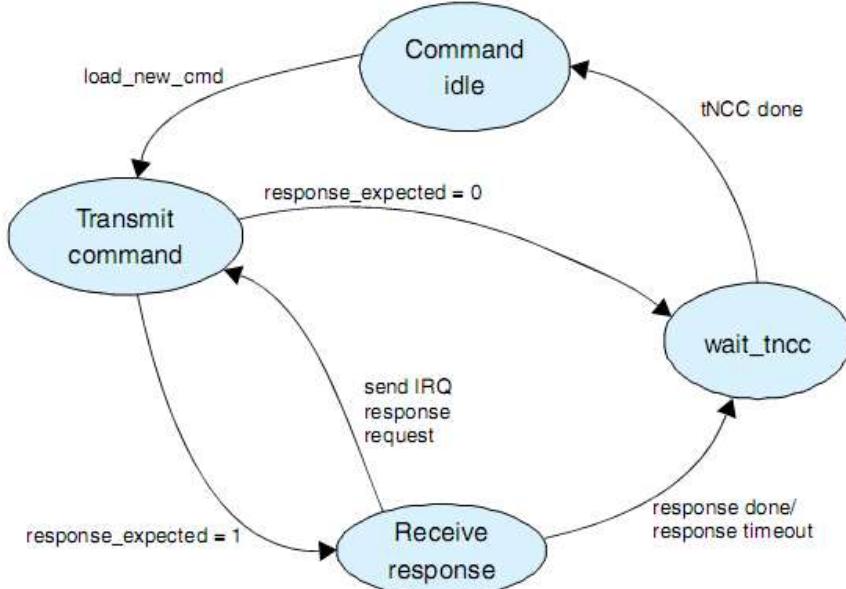


Fig. 1-3 Host Controller Command Path State Machine

The command path state machine performs the following functions, according to Command register bit values:

- **send_initialization** – Initialization sequence of 80 clocks is sent before sending the command.
- **response_expected** – Response is expected for the command. After the command is sent out, the command path state machine receives a 48-bit or 136-bit response and sends it to the BIU. If the start bit of the card response is not received within the number of clocks programmed in the timeout register, then the response timeout and command done bit is set in the Raw Interrupt Status register as a signal to the BIU. If the response-expected bit is not set, the command path sends out a command and signals a response done to the BIU; that is, the command done bit is set in the Raw Interrupt Status register.
- **response_length** – If this bit is set, a 136-bit response is received; if it is not set, a 48-bit response is received.
- **check_response_crc** – If this bit is set, the command path compares CRC7 received in the response with the internally-generated CRC7. If the two do not match, the response CRC error is signaled to the BIU; that is, the response CRC error bit is set in the Raw Interrupt Status register.

Send Response to BIU

If the response_expected bit is set in the Command register, the received response is sent to the BIU. The Response0 register is updated for a short response, and the Response3, Response2, Response1, and Response0 registers are updated on a long response, after which the Command Done bit is set. If the response is for an auto_stop command sent by the CIU, the response is saved in the Response1 register, after which the Auto Command Done bit is set.

Additionally, the command path checks for the following:

- Transmission bit = 0
- Command index matches command index of the sent command
- End bit = 1 in received card response

The command index is not checked for a 136-bit response or if the check_response_crc bit is unset. For a 136-bit response and reserved CRC 48-bit responses, the command index is reserved—that is, 111111.

Polling Command Completion Signal

The device generates the Command Completion Signal in order to notify the host controller of the normal command completion or command termination.

Command Completion Signal Detection and Interrupt to Host Processor

If the ccs_expected bit is set in the Command register, the Command Completion Signal (CCS) from the device is indicated by setting the Data Transfer Over (DTO) bit in the RINTSTS register. The Host Controller generates a Data Transfer Over (DTO) interrupt if this interrupt is not masked.

Command Completion Signal Timeout

If the command expects a CCS from the device—if the ccs_expected bit is set in the Command register—the command state machine waits for the CCS and remains in a wait_CCSS state. If the device fails to send out the CCS, the host software should implement a timeout mechanism to free the command and data path. The host controller does not implement a hardware timer; it is the responsibility of the host software to maintain a software timer. In the event of a CCS timeout, the host should issue a CCSD by setting the send_ccsd bit in the CTRL register. The host controller command state machine sends the CCSD to the device and exits to an idle state. After sending the CCSD, the host should also send a CMD12 to the device in order to abort the outstanding command.

Send Command Completion Signal Disable

If the send_ccsd bit is set in the CTRL register, the host sends a Command Completion Signal Disable (CCSD) pattern on the CMD line. The host can send the CCSD while waiting for the CCS or after a CCS timeout happens.

After sending the CCSD pattern, the host sets the Command Done (CD) bit in RINTSTS and also generates an interrupt to the host if the Command Done interrupt is not masked.

2. Data Path

The data path block pops the data FIFO and transmits data on cdata_out during a write data transfer, or it receives data on cdata_in and pushes it into the FIFO during a read data transfer. The data path loads new data parameters—that is, data expected, read/write data transfer, stream/block transfer, block size, byte count, card type, timeout registers—whenever a data transfer command is not in progress.

If the data_expected bit is set in the Command register, the new command is a data transfer command and the data path starts one of the following:

- Transmit data if the read/write bit = 1
- Data receive if read/write bit = 0

Data Transmit

The data transmit state machine, illustrated in following figure, starts data transmission two clocks after a response for the data write command is received; this occurs even if the command path detects a response error or response CRC error. If a response is not received from the card because of a response timeout, data is not transmitted. Depending upon the value of the transfer_mode bit in the Command register, the data transmit state machine puts data on the card data bus in a stream or in block(s).

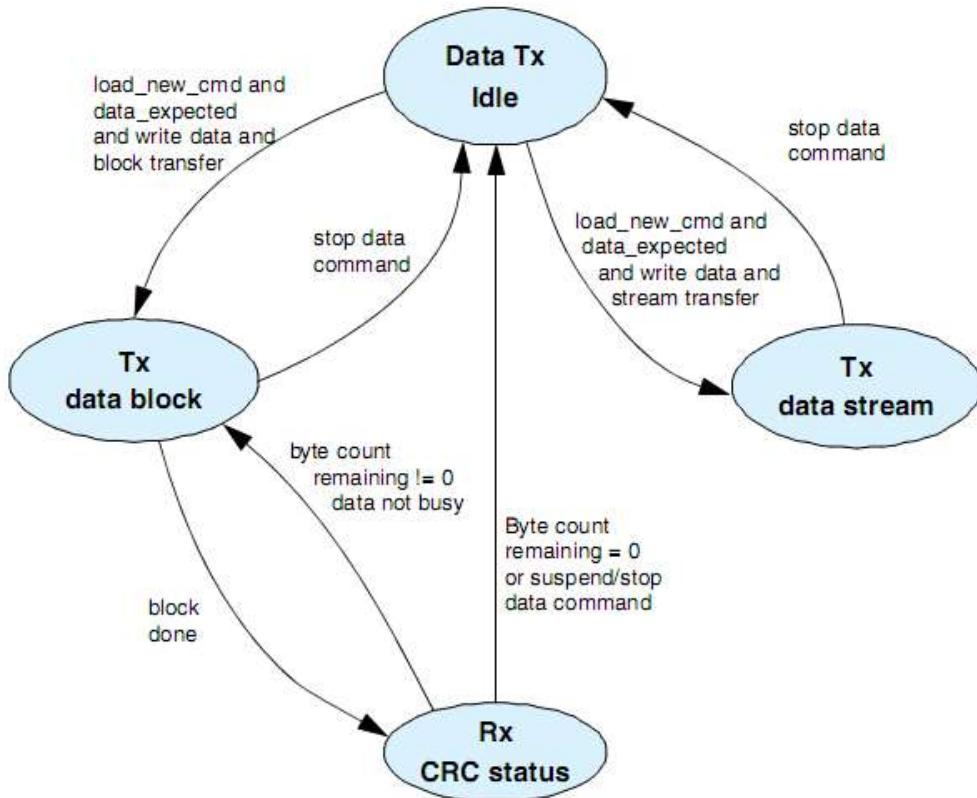


Fig. 1-4 Host Controller Data Transmit State Machine

Stream Data Transmit

If the transfer_mode bit in the Command register is set to 1, it is a stream-write data transfer. The data path pops the FIFO from the BIU and transmits in a stream to the card data bus. If the FIFO becomes empty, the card clock is stopped and restarted once data is available in the FIFO.

If the byte_count register is programmed to 0, it is an open-ended stream-write data transfer. During this data transfer, the data path continuously transmits data in a stream until the host software issues a stop command. A stream data transfer is terminated when the end bit of the stop command and end bit of the data match over two clocks.

If the byte_count register is programmed with a non-zero value and the send_auto_stop bit is set in the Command register, the stop command is internally generated and loaded in the command path when the end bit of the stop command occurs after the last byte of the stream write transfer matches.

This data transfer can also terminate if the host issues a stop command before all the data bytes are transferred to the card bus.

Single Block Data

If the transfer_mode bit in the Command register is set to 0 and the byte_count register value is equal to the value of the block_size register, a single-block write-data transfer occurs. The data transmit state machine sends data in a single block, where the number of bytes equals the block size, including the internally-generated CRC16.

If the CTYPE register bit for the selected card – indicated by the card_num value in the Command register – is set for a 1-bit, 4-bit, or 8-bit data transfer, the data is transmitted on 1, 4, or 8 data lines, respectively, and CRC16 is separately generated and transmitted for 1, 4, or 8 data lines, respectively.

After a single data block is transmitted, the data transmit state machine receives the CRC status from the card and signals a data transfer to the BIU; this happens when the data-transfer-over bit is set in the RINTSTS register.

If a negative CRC status is received from the card, the data path signals a data CRC error to the BIU by setting the data CRC error bit in the RINTSTS register.

Additionally, if the start bit of the CRC status is not received by two clocks after the end of the data block, a CRC status start bit error is signaled to the BIU by setting the write-no-CRC bit in the RINTSTS register.

Multiple Block Data

A multiple-block write-data transfer occurs if the transfer_mode bit in the Command register is set to 0 and the value in the byte_count register is not equal to the value of the block_size register. The data transmit state machine sends data in blocks, where the number of bytes in a block equals the block size, including the internally-generated CRC16.

If the CTYPE register bit for the selected card – indicated by the card_num value in the Command register – is set to 1-bit, 4-bit, or 8-bit data transfer, the data is transmitted on 1, 4, or 8 data lines, respectively, and CRC16 is separately generated and transmitted on 1, 4, or 8 data lines, respectively.

After one data block is transmitted, the data transmit state machine receives the CRC status from the card. If the remaining byte_count becomes 0, the data path signals to the BIU that the data transfer is done; this happens when the data-transfer-over bit is set in the RINTSTS register.

If the remaining data bytes are greater than 0, the data path state machine starts to transmit another data block.

If a negative CRC status is received from the card, the data path signals a data CRC error to the BIU by setting the data CRC error bit in the RINTSTS register, and continues further data transmission until all the bytes are transmitted.

Additionally, if the CRC status start bit is not received by two clocks after the end of a data block, a CRC status start bit error is signaled to the BIU by setting the write-no-CRC bit in the RINTSTS register; further data transfer is terminated.

If the send_auto_stop bit is set in the Command register, the stop command is internally generated during the transfer of the last data block, where no extra bytes are transferred to the card. The end bit of the stop command may not exactly match the end bit of the CRC status in the last data block.

If the block size is less than 4, 16, or 32 for card data widths of 1 bit, 4 bits, or 8 bits, respectively, the data transmit state machine terminates the data transfer when all the data is transferred, at which time the internally generated stop command is loaded in the command path.

If the byte_count is 0 – the block size must be greater than 0 – it is an open-ended block transfer. The data transmit state machine for this type of data transfer continues the block-write data transfer until the host software issues a stop or abort command.

Data Receive

The data-receive state machine, illustrated in following figure, receives data two clock cycles after the end bit of a data read command, even if the command path detects a response error or response CRC error. If a response is not received from the card because a response timeout occurs, the BIU does not receive a signal that the data transfer is complete; this happens if the command sent by the Host Controller is an illegal operation for the card, which keeps the card from starting a read data transfer.

If data is not received before the data timeout, the data path signals a data timeout to the BIU and an end to the data transfer done. Based on the value of the transfer_mode bit in the Command register, the data-receive state machine gets data from the card data bus in a stream or block(s).

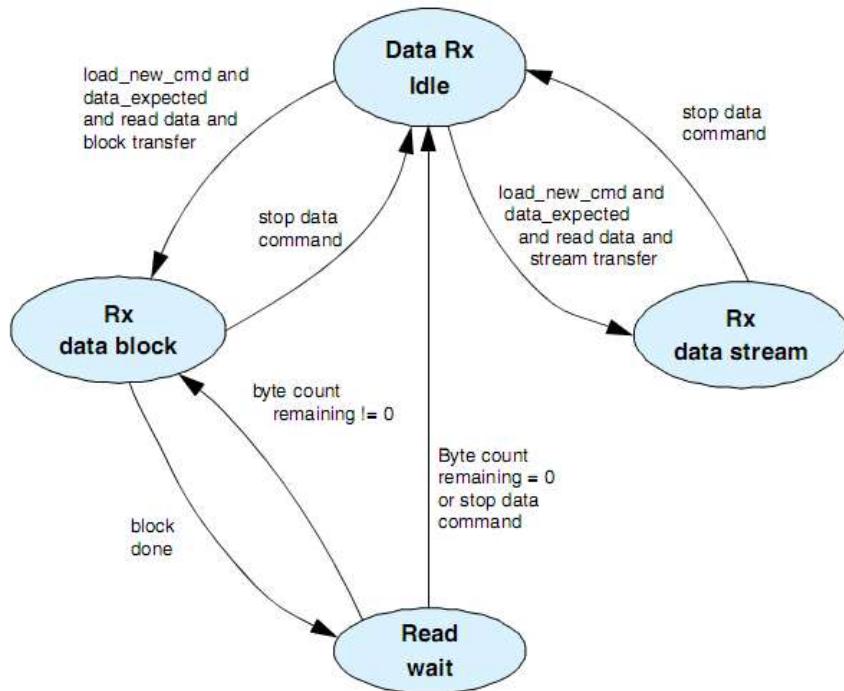


Fig. 1-5 Host Controller Data Receive State Machine

Stream Data Read

A stream-read data transfer occurs if the transfer_mode bit in the Command register equals 1, at which time the data path receives data from the card and pushes it to the FIFO. If the FIFO becomes full, the card clock stops and restarts once the FIFO is no longer full.

An open-ended stream-read data transfer occurs if the byte_count register equals 0. During this type of data transfer, the data path continuously receives data in a stream until the host software issues a stop command. A stream data transfer terminates two clock cycles after the end bit of the stop command.

If the byte_count register contains a non-zero value and the send_auto_stop bit is set in the Command register, a stop command is internally generated and loaded into the command path, where the end bit of the stop command occurs after the last byte of the stream data transfer is received. This data transfer can terminate if the host issues a stop or abort command before all the data bytes are received from the card.

Single-Block Data Read

A single-block read-data transfer occurs if the transfer_mode bit in the Command register is set to 0 and the value of the byte_count register is equal to the value of the block_size register. When a start bit is received before the data times out, data bytes equal to the block size and CRC16 are received and checked with the internally-generated CRC16.

If the CTYPE register bit for the selected card – indicated by the card_num value in the Command register – is set to a 1-bit, 4-bit, or 8-bit data transfer, data is received from 1, 4, or 8 data lines, respectively, and CRC16 is separately generated and checked for 1, 4, or 8 data lines, respectively. If there is a CRC16 mismatch, the data path signals a data CRC error to the BIU. If the received end bit is not 1, the BIU receives an end-bit error.

Multiple-Block Data Read

If the transfer_mode bit in the Command register is set to 0 and the value of the byte_count register is not equal to the value of the block_size register, it is a multiple-block read-data transfer. The data-receive state machine receives data in blocks, where the number of bytes in a block is equal to the block size, including the internally-generated CRC16.

If the CTYPE register bit for the selected card – indicated by the card_num value in the Command register – is set to a 1-bit, 4-bit, or 8-bit data transfer, data is received from 1, 4, or 8 data lines, respectively, and CRC16 is separately generated and checked for 1, 4, or 8 data lines, respectively.

After a data block is received, if the remaining byte_count becomes 0, the data path signals a data transfer to the BIU.

If the remaining data bytes are greater than 0, the data path state machine causes another data block to be received. If CRC16 of a received data block does not match the

internally-generated CRC16, a data CRC error to the BIU and data reception continue further data transmission until all bytes are transmitted.

Additionally, if the end of a received data block is not 1, data on the data path signals terminate the bit error to the CIU and the data-receive state machine terminates data reception, waits for data timeout, and signals to the BIU that the data transfer is complete. If the send_auto_stop bit is set in the Command register, the stop command is internally generated when the last data block is transferred, where no extra bytes are transferred from the card; the end bit of the stop command may not exactly match the end bit of the last data block.

If the requested block size for data transfers to cards is less than 4, 16, or 32 bytes for 1-bit, 4-bit, or 8-bit data transfer modes, respectively, the data-transmit state machine terminates the data transfer when all data is transferred, at which point the internally-generated stop command is loaded in the command path. Data received from the card after that are then ignored by the data path.

If the byte_count is 0—the block size must be greater than 0—it is an open-ended block transfer. For this type of data transfer, the data-receive state machine continues the block-read data transfer until the host software issues a stop or abort command.

Auto-Stop

The Host Controller internally generates a stop command and is loaded in the command path when the send_auto_stop bit is set in the Command register.

The software should set the send_auto_stop bit according to details listed in following table.

Table 1-2 Auto-Stop Generation

Card type	Transfer type	Byte Count	send_auto_stop bit set	Comments
MMC	Stream read	0	No	Open-ended stream
MMC	Stream read	>0	Yes	Auto-stop after all bytes transfer
MMC	Stream write	0	No	Open-ended stream
MMC	Stream write	>0	Yes	Auto-stop after all bytes transfer
MMC	Single-block read	>0	No	Byte count =0 is illegal
MMC	Single-block write	>0	No	Byte count =0 is illegal
MMC	Multiple-block read	0	No	Open-ended multiple block
MMC	Multiple-block read	>0	Yes①	Pre-defined multiple block
MMC	Multiple-block write	0	No	Open-ended multiple block
MMC	Multiple-block write	>0	Yes①	Pre-defined multiple block
SDMEM	Single-block read	>0	No	Byte count =0 is illegal
SDMEM	Single-block write	>0	No	Byte count =0 illegal
SDMEM	Multiple-block read	0	No	Open-ended multiple block
SDMEM	Multiple-block read	>0	Yes	Auto-stop after all bytes transfer
SDMEM	Multiple-block write	0	No	Open-ended multiple block
SDMEM	Multiple-block write	>0	Yes	Auto-stop after all bytes transfer
SDIO	Single-block read	>0	No	Byte count =0 is illegal
SDIO	Single-block write	>0	No	Byte count =0 illegal
SDIO	Multiple-block read	0	No	Open-ended multiple block
SDIO	Multiple-block read	>0	No	Pre-defined multiple block
SDIO	Multiple-block write	0	No	Open-ended multiple block
SDIO	Multiple-block	>0	No	Pre-defined multiple block

Card type	Transfer type	Byte Count	send_auto_stop bit set	Comments
	write			

¹:The condition under which the transfer mode is set to block transfer and byte_count is equal to block size is treated as a single-block data transfer command for both MMC and SD cards. If byte_count = n*block_size (n = 2, 3, ...), the condition is treated as a predefined multiple-block data transfer command. In the case of an MMC card, the host software can perform a predefined data transfer in two ways: 1) Issue the CMD23 command before issuing CMD18/CMD25 commands to the card – in this case, issue MD18/CMD25 commands without setting the send_auto_stop bit. 2) Issue CMD18/CMD25 commands without issuing CMD23 command to the card, with the send_auto_stop bit set. In this case, the multiple-block data transfer is terminated by an internally-generated auto-stop command after the programmed byte count.

The following list conditions for the auto-stop command.

- Stream read for MMC card with byte count greater than 0 – The Host Controller generates an internal stop command and loads it into the command path so that the end bit of the stop command is sent out when the last byte of data is read from the card and no extra data byte is received. If the byte count is less than 6 (48 bits), a few extra data bytes are received from the card before the end bit of the stop command is sent.
- Stream write for MMC card with byte count greater than 0 - The Host Controller generates an internal stop command and loads it into the command path so that the end bit of the stop command is sent when the last byte of data is transmitted on the card bus and no extra data byte is transmitted. If the byte count is less than 6 (48 bits), the data path transmits the data last in order to meet the above condition.
- Multiple-block read memory for SD card with byte count greater than 0 – If the block size is less than 4 (single-bit data bus), 16 (4-bit data bus), or 32 (8-bit data bus), the auto-stop command is loaded in the command path after all the bytes are read. Otherwise, the top command is loaded in the command path so that the end bit of the stop command is sent after the last data block is received.
- Multiple-block write memory for SD card with byte count greater than 0 – If the block size is less than 3 (single-bit data bus), 12 (4-bit data bus), or 24 (8-bit data bus), the auto-stop command is loaded in the command path after all data blocks are transmitted. Otherwise, the stop command is loaded in the command path so that the end bit of the stop command is sent after the end bit of the CRC status is received.
- Precaution for host software during auto-stop – Whenever an auto-stop command is issued, the host software should not issue a new command to the SD/MMC device until the auto-stop is sent by the Host Controller and the data transfer is complete. If the host issues a new command during a data transfer with the auto-stop in progress, an auto-stop command may be sent after the new command is sent and its response is received; this can delay sending the stop command, which transfers extra data bytes. For a stream write, extra data bytes are erroneous data that can corrupt the card data. If the host wants to terminate the data transfer before the data transfer is complete, it can issue a stop or abort command, in which case the Host Controller does not generate an auto-stop command.

3. Non-Data Transfer Commands that Use Data Path

Some non-data transfer commands (non-read/write commands) also use the data path.

Following table lists the commands and register programming requirements for them.

Table 1-3 Non-data Transfer Commands and Requirements

Base Address [12:8]	CMD 27	CMD 30	CMD 42	ACMD 13	ACMD 22	ACMD 51
Command register programming						
cmd_index	6'h1B	6'h1E	6'h2A	6'h0D	6'h16	6'h33
response_expect	1	1	1	1	1	1
rResponse_length	0	0	0	0	0	0
check_response_crc	1	1	1	1	1	1
data_expected	1	1	1	1	1	1
read/write	1	0	1	0	0	0
transfer_mode	0	0	0	0	0	0
send_auto_stop	0	0	0	0	0	0
wait_prevdata_complete	0	0	0	0	0	0

Base Address [12:8]	CMD 27	CMD 30	CMD 42	ACMD 13	ACMD 22	ACMD 51
stop_abort_cmd	0	0	0	0	0	0
Command Argument register programming						
	stuff bits	32-bit write protect data address	stuff bits	stuff bits	stuff bits	stuff bits
Block Size register programming						
	16	4	Num_bytes①	64	4	8
Byte Count register programming						
	16	4	Num_bytes①	64	4	8

①: Num_bytes = No. of bytes specified as per the lock card data structure (Refer to the SD specification and the MMC specification)

4. SDIO Interrupt Control

Interrupts for SD cards are reported to the BIU by asserting an interrupt signal for two clock cycles. SDIO cards signal an interrupt by asserting cdata_in low during the interrupt period; an interrupt period for the selected card is determined by the interrupt control state machine. An interrupt period is always valid for non-active or non-selected cards, and 1-bit data mode for the selected card. An interrupt period for a wide-bus active or selected card is valid for the following conditions:

- Card is idle
- Non-data transfer command in progress
- Third clock after end bit of data block between two data blocks
- From two clocks after end bit of last data until end bit of next data transfer command

Bear in mind that, in the following situations, the controller does not sample the SDIO interrupt of the selected card when the card data width is 4 bits. Since the SDIO interrupt is level-triggered, it is sampled in a further interrupt period and the host does not lose any SDIO interrupt from the card.

- Read/Write Resume – The CIU treats the resume command as a normal data transfer command. SDIO interrupts during the resume command are handled similarly to other data commands. According to the SDIO specification, for the normal data command the interrupt period ends after the command end bit of the data command; for the resume command, it ends after the response end bit. In the case of the resume command, the Controller stops the interrupt sampling period after the resume command end bit, instead of stopping after the response end bit of the resume command.
- Suspend during read transfer – If the read data transfer is suspended by the host, the host sets the abort_read_data bit in the controller to reset the data state machine. In the CIU, the SDIO interrupts are handled such that the interrupt sampling starts after the abort_read_data bit is set by the host. In this case the controller does not sample SDIO interrupts between the period from response of the suspend command to setting the abort_read_data bit, and starts sampling after setting the abort_read_data bit.

5. Clock Control

The clock control block provides different clock frequencies required for SD/MMC cards. The cclk_in signal is the source clock ($cclk_in \geq$ card max operating frequency) for clock divider of the clock control block. This source clock (cclk_in) is used to generate different card clock frequencies (cclk_out). The card clock can have different clock frequencies, since the card can be a low-speed card or a full-speed card. The Host Controller provides one clock signal (cclk_out).

The clock frequency of a card depends on the following clock control registers:

- Clock Divider register – Internal clock dividers are used to generate different clock frequencies required for card. The division factor for each clock divider can be programmed by writing to the Clock Divider register. The clock divider is an 8-bit value that provides a clock division factor from 1 to 510; a value of 0 represents a clock-divider bypass, a value of 1 represents a divide by 2, a value of 2 represents a divide by 4, and so on.

- Clock Control register – cclk_out can be enabled or disabled for each card under the following conditions:
 - clk_enable – cclk_out for a card is enabled if the clk_enable bit for a card in the Clock Control register is programmed (set to 1) or disabled (set to 0).
 - Low-power mode – Low-power mode of a card can be enabled by setting the low-power mode bit of the Clock Control register to 1. If low-power mode is enabled to save card power, the cclk_out is disabled when the card is idle for at least 8 card clock cycles. It is enabled when a new command is loaded and the command path goes to a non-idle state.

Additionally, cclk_out is disabled when an internal FIFO is full – card read (no more data can be received from card) – or when the FIFO is empty – card write (no data is available for transmission). This helps to avoid FIFO overrun and underrun conditions. It is used by the command and data path to qualify cclk_in for driving outputs and sampling inputs at the programmed clock frequency for the selected card, according to the Clock Divider and Clock Source register values.

Under the following conditions, the card clock is stopped or disabled, along with the active clk_en, for the selected card:

- Clock can be disabled by writing to Clock Enable register (clk_en bit = 1).
- If low-power mode is selected and card is idle, or not selected for 8 clocks.
- FIFO is full and data path cannot accept more data from the card and data transfer is incomplete –to avoid FIFO overrun.
- FIFO is empty and data path cannot transmit more data to the card and data transfer is incomplete – to avoid FIFO underrun.

6. Error Detection

- Response
 - Response timeout – Response expected with response start bit is not received within programmed number of clocks in timeout register.
 - Response CRC error – Response is expected and check response CRC requested; response CRC7 does not match with the internally-generated CRC7.
 - Response error – Response transmission bit is not 0, command index does not match with the command index of the send command, or response end bit is not 1.
- Data transmit
 - No CRC status – During a write data transfer, if the CRC status start bit is not received two clocks after the end bit of the data block is sent out, the data path does the following:
 - ◆ Signals no CRC status error to the BIU
 - ◆ Terminates further data transfer
 - ◆ Signals data transfer done to the BIU
 - Negative CRC – If the CRC status received after the write data block is negative (that is, not 010), a data CRC error is signaled to the BIU and further data transfer is continued.
 - Data starvation due to empty FIFO – If the FIFO becomes empty during a write data transmission, or if the card clock is stopped and the FIFO remains empty for data timeout clocks, then a data-starvation error is signaled to the BIU and the data path continues to wait for data in the FIFO.
- Data receive
 - Data timeout – During a read-data transfer, if the data start bit is not received before the number of clocks that were programmed in the timeout register, the data path does the following:
 - ◆ Signals data-timeout error to the BIU
 - ◆ Terminates further data transfer
 - ◆ Signals data transfer done to BIU
 - Data start bit error – During a 4-bit or 8-bit read-data transfer, if the all-bit data line does not have a start bit, the data path signals a data start bit error to the BIU and waits for a data timeout, after which it signals that the data transfer is done.
 - Data CRC error – During a read-data-block transfer, if the CRC16 received does not match with the internally generated CRC16, the data path signals a data CRC error to

- the BIU and continues further data transfer.
- Data end-bit error – During a read-data transfer, if the end bit of the received data is not 1, the data path signals an end-bit error to the BIU, terminates further data transfer, and signals to the BIU that the data transfer is done.
- Data starvation due to FIFO full – During a read data transmission and when the FIFO becomes full, the card clock is stopped. If the FIFO remains full for data timeout clocks, a data starvation error is signaled to the BIU (Data Starvation by Host Timeout bit is set in RINTSTS Register) and the data path continues to wait for the FIFO to start to empty.

1.4 Register Description

1.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
SDMMC_CTRL	0x0000	W	0x01000000	Control register
SDMMC_PWREN	0x0004	W	0x00000000	Power-enable register
SDMMC_CLKDIV	0x0008	W	0x00000000	Clock-divider register
SDMMC_CLKSRC	0x000c	W	0x00000000	SD Clock Source Register
SDMMC_CLKENA	0x0010	W	0x00000000	Clock-enable register
SDMMC_TMOUT	0x0014	W	0xfffffff40	Time-out register
SDMMC_CTYPE	0x0018	W	0x00000000	Card-type register
SDMMC_BLKSIZ	0x001c	W	0x000000200	Block-size register
SDMMC_BYTCNT	0x0020	W	0x000000200	Byte-count register
SDMMC_INTMASK	0x0024	W	0x00000000	Interrupt-mask register
SDMMC_CMDARG	0x0028	W	0x00000000	Command-argument register
SDMMC_CMD	0x002c	W	0x00000000	Command register
SDMMC_RESP0	0x0030	W	0x00000000	Response-0 register
SDMMC_RESP1	0x0034	W	0x00000000	Response-1 register
SDMMC_RESP2	0x0038	W	0x00000000	Response-2 register
SDMMC_RESP3	0x003c	W	0x00000000	Response-3 register
SDMMC_MINTSTS	0x0040	W	0x00000000	Masked interrupt-status register
SDMMC_RINTSTS	0x0044	W	0x00000000	Raw interrupt-status register
SDMMC_STATUS	0x0048	W	0x00000406	Status register
SDMMC_FIFOTH	0x004c	W	0x00000000	FIFO threshold register
SDMMC_CDETECT	0x0050	W	0x00000000	Card-detect register
SDMMC_WRTPRT	0x0054	W	0x00000000	Write-protect register
SDMMC_TCBCNT	0x005c	W	0x00000000	Transferred CIU card byte count
SDMMC_TBBCNT	0x0060	W	0x00000000	Transferred host/DMA to/from BIU-FIFO byte count
SDMMC_DEBNCE	0x0064	W	0x00ffff	Card detect debounce register
SDMMC_USRID	0x0068	W	0x07967797	User ID register
SDMMC_VERID	0x006c	W	0x5342270a	Synopsys version ID register
SDMMC_UHS_REG	0x0074	W	0x00000000	UHS-1 register
SDMMC_RST_N	0x0078	W	0x00000001	Hardware reset register
SDMMC_PLDMND	0x0084	W	0x00000000	Poll Demand Register
SDMMC_CARDTHRCTL	0x0100	W	0x00000000	Card read threshold enable

Name	Offset	Size	Reset Value	Description
SDMMC_BACK_END_POWER	0x0104	W	0x00000000	Back-end power
SDMMC_UHS_REG_EXT	0x0108	W	0x00000000	UHS Register
SDMMC_EMMC_DDR_REG	0x010c	W	0x00000000	eMMC 4.5 DDR START Bit Detection Control Register
SDMMC_ENABLE_SHIFT	0x0110	W	0x00000000	Enable Phase Shift Register
SDMMC_FIFO_BASE	0x0200	W	0x00000000	FIFO Base Address

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

1.4.2 Detail Register Description

SDMMC_CTRL

Address: Operational Base + offset (0x0000)

Control register

Bit	Attr	Reset Value	Description
31:26	RO	0x0	reserved
25	RW	0x0	reserved
24	RW	0x1	reserved
23:20	RW	0x0	reserved
19:16	RW	0x0	reserved
15:12	RO	0x0	reserved
11	RW	0x0	<p>ceata_device_interrupt_status 0: Interrupts not enabled 1: Interrupts are enabled</p> <p>Software should appropriately write to this bit after power-on reset or any other reset to CE-ATA device. After reset, usually CE-ATA device interrupt is disabled (nIEN = 1). If the host enables CE-ATA device interrupt, then software should set this bit.</p>
10	RW	0x0	<p>send_auto_stop_ccsd 0: Clear bit if Host controller does not reset the bit. 1: Send internally generated STOP after sending CCSD to CE-ATA device.</p> <p>NOTE: Always set send_auto_stop_ccsd and send_ccsd bits together send_auto_stop_ccsd should not be set independent of send_ccsd.</p> <p>When set, Host controller automatically sends internally-generated STOP command (CMD12) to CE-ATA device. After sending internally-generated STOP command, Auto Command Done (ACD) biin RINTSTS is set and generates interrupt to host if Auto Command Done interrupt is not masked. After sending the CCSD, Host controller automatically clears send_auto_stop_ccsd bit.</p>

Bit	Attr	Reset Value	Description
9	RW	0x0	<p>send_ccsd 0: Clear bit if Host controller does not reset the bit. 1: Send Command Completion Signal Disable (CCSD) to CE-ATA device When set, Host controller sends CCSD to CE-ATA device. Software sets this bit only if current command is expecting CCS (that is, RW_BLK) and interrupts are enabled in CE-ATA device. Once the CCSD pattern is sent to device, Host controller automatically clears send_ccsd bit. It also sets Command Done (CD) bit in RINTSTS register and generates interrupt to host if Command Done interrupt is not masked.</p> <p>NOTE: Once send_ccsd bit is set, it takes two card clock cycles to drive the CCSD on the CMD line. Due to this, during the boundary conditions it may happen that CCSD is sent to the CE-ATA device, even if the device signalled CCS</p>
8	RW	0x0	<p>abort_read_data 0: no change 1: after suspend command is issued during read-transfer, software polls card to find when suspend happened. Once suspend occurs, software sets bit to reset data state-machine, which is waiting for next block of data. Bit automatically clears once data state machine resets to idle.</p> <p>Used in SDIO card suspend sequence.</p>
7	RW	0x0	<p>send_irq_response 0: no change 1: send auto IRQ response Bit automatically clears once response is sent.</p> <p>To wait for MMC card interrupts, host issues CMD40, and SDMMC Controller waits for interrupt response from MMC card(s). In meantime, if host wants SDMMC Controller to exit waiting for interrupt state, it can set this bit, at which time SDMMC Controller command state-machine sends CMD40 response on bus and returns to idle state.</p>
6	RW	0x0	<p>read_wait 0: clear read wait 1: assert read wait For sending read-wait to SDIO cards</p>
5	RW	0x0	<p>dma_enable 0: disable DMA transfer mode 1: enable DMA transfer mode</p> <p>Even when DMA mode is enabled, host can still push/pop data into or from FIFO; this should not happen during the normal operation. If there is simultaneous FIFO access from host/DMA, the data coherency is lost. Also, there is no arbitration inside SDMMC Controller to prioritize simultaneous host/DMA access.</p>

Bit	Attr	Reset Value	Description
4	RW	0x0	<p>int_enable Global interrupt enable/disable bit: 0: disable interrupts 1: enable interrupts The int port is 1 only when this bit is 1 and one or more unmasked interrupts are set.</p>
3	RO	0x0	reserved
2	W1C	0x0	<p>dma_reset 0: no change 1: reset internal DMA interface control logic To reset DMA interface, firmware should set bit to 1. This bit is auto-cleared after two AHB clocks.</p>
1	W1C	0x0	<p>fifo_reset 0: no change 1: reset to data FIFO To reset FIFO pointers To reset FIFO, firmware should set bit to 1. This bit is auto-cleared after completion of reset operation</p>
0	W1C	0x0	<p>controller_reset 0: no change 1: reset SDMMC controller To reset controller, firmware should set bit to 1. This bit is auto-cleared after two AHB and two cclk_in clock cycles. This resets:<ul style="list-style-type: none"> ● BIU/CIU interface ● CIU and state machines ● abort_read_data, send_irq_response, and read_wait bits of Control register ● start_cmd bit of Command register Does not affect any registers or DMA interface, or FIFO or host interrupts</p>

SDMMC_PWREN

Address: Operational Base + offset (0x0004)

Power-enable register

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	<p>power_enable Power on/off switch for the card. Once power is turned on, firmware should wait for regulator/switch ramp-up time before trying to initialize card. 0: power off 1: power on Bit values output to card_power_en port.</p>

SDMMC_CLKDIV

Address: Operational Base + offset (0x0008)

Clock-divider register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	RW	0x00	clk_divider0 Clock divider-0 value. Clock division is 2^n . For example, value of 0 means divide by $2^0 = 1$ (no division, bypass), value of 1 means divide by $2^1 = 2$, value of "ff" means divide by $2^{255} = 510$, and so on. Note: divider>3 is not recommended.

SDMMC_CLKSRC

Address: Operational Base + offset (0x000c)

SD Clock Source Register

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

SDMMC_CLKENA

Address: Operational Base + offset (0x0010)

Clock-enable register

Bit	Attr	Reset Value	Description
31:17	RO	0x0	reserved
16	RW	0x0	cclk_low_power Low-power control for SD card clock and MMC card clock supported. 0: non-low-power mode 1: low-power mode; stop clock when card in IDLE (should be normally set to only MMC and SD memory cards; for SDIO cards, if interrupts must be detected, clock should not be stopped).
15:1	RO	0x0	reserved
0	RW	0x0	cclk_enable Clock-enable control for SD card clock and MMC card clock supported. 0: clock disabled 1: clock enabled

SDMMC_TMOUT

Address: Operational Base + offset (0x0014)

Time-out register

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

Bit	Attr	Reset Value	Description
31:8	RW	0xffffffff	<p>data_timeout Value for card Data Read Timeout; same value also used for Data Starvation by Host timeout.</p> <p>Value is in number of card output clocks cclk_out of selected card. Note: The software timer should be used if the timeout value is in the order of 100 ms. In this case, read data timeout interrupt needs to be disabled.</p>
7:0	RW	0x40	<p>response_timeout Response timeout value.</p> <p>Value is in number of card output clocks -cclk_out.</p>

SDMMC_CTYPE

Address: Operational Base + offset (0x0018)

Card-type register

Bit	Attr	Reset Value	Description
31:17	RO	0x0	reserved
16	RW	0x0	<p>card_width_8 Indicates if card is 8-bit: 0: non 8-bit mode 1: 8-bit mode</p>
15:1	RO	0x0	reserved
0	RW	0x0	<p>card_width Indicates if card is 1-bit or 4-bit: 0: 1-bit mode 1: 4-bit mode</p>

SDMMC_BLKSIZ

Address: Operational Base + offset (0x001c)

Block-size register

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0200	block_size

SDMMC_BYTCNT

Address: Operational Base + offset (0x0020)

Byte-count register

Bit	Attr	Reset Value	Description
31:0	RW	0x000000200	<p>byte_count Number of bytes to be transferred; should be integer multiple of Block Size for block transfers. For undefined number of byte transfers, byte count should be set to 0. When byte count is set to 0, it is responsibility of host to explicitly send stop/abort command to terminate data transfer.</p>

SDMMC_INTMASK

Address: Operational Base + offset (0x0024)

Interrupt-mask register

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24	RW	0x0	sdio_int_mask Mask SDIO interrupts. When masked, SDIO interrupt detection for that card is disabled. A 0 masks an interrupt, and 1 enables an interrupt.
23:17	RO	0x0	reserved
16	RW	0x0	data_nobusy_int_mask 0: data no busy interrupt not masked 1: data no busy interrupt masked
15:0	RW	0x0000	int_mask Bits used to mask unwanted interrupts. Value of 0 masks interrupt; value of 1 enables interrupt. [15]: End-bit error (read)/Write no CRC (EBE) [14]: Auto command done (ACD) [13]: Start-bit error (SBE) [12]: Hardware locked write error (HLE) [11]: FIFO underrun/overrun error (FRUN) [10]: Data starvation-by-host timeout (HTO) /Volt_switch_int [9]: Data read timeout (DRTO) [8]: Response timeout (RTO) [7]: Data CRC error (DCRC) [6]: Response CRC error (RCRC) [5]: Receive FIFO data request (RXDR) [4]: Transmit FIFO data request (TXDR) [3]: Data transfer over (DTO) [2]: Command done (CD) [1]: Response error (RE) [0]: Card detect (CD)

SDMMC_CMDARG

Address: Operational Base + offset (0x0028)

Command-argument register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	cmd_arg Value indicates command argument to be passed to card.

SDMMC_CMD

Address: Operational Base + offset (0x002c)

Command register

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31	RW	0x0	<p>start_cmd Start command. Once command is taken by CIU, bit is cleared. When bit is set, host should not attempt to write to any command registers. If write is attempted, hardware lock error is set in raw interrupt register.</p> <p>Once command is sent and response is received from SD_MMC cards, Command Done bit is set in raw interrupt register.</p>
30	RO	0x0	reserved
29	RW	0x0	<p>use_hold_reg Use Hold Register 0: CMD and DATA sent to card bypassing HOLD Register 1: CMD and DATA sent to card through the HOLD Register</p> <p>Note:</p> <ul style="list-style-type: none"> ● Set to 1'b1 for SDR12 and SDR25 (with non-zero phase-shifted cclk_in_drv); zero phase shift is not allowed in these modes. ● Set to 1'b0 for SDR50, SDR104, and DDR50 (with zero phase-shifted cclk_in_drv) ● Set to 1'b1 for SDR50, SDR104, and DDR50 (with non-zero phase-shifted cclk_in_drv)
28	RW	0x0	<p>volt_switch Voltage switch bit. 0: no voltage switching 1: voltage switching enabled; must be set for CMD11 only</p>
27	RW	0x0	<p>boot_mode 0: mandatory Boot operation 1: alternate Boot operation</p>
26	RW	0x0	<p>disable_boot Disable Boot. When software sets this bit along with start_cmd, CIU terminates the boot operation. Do NOT set disable_boot and enable_boot together.</p>
25	RW	0x0	<p>expect_boot_ack Expect Boot Acknowledge. When Software sets this bit along with enable_boot, CIU expects a boot acknowledge start pattern of 0-1-0 from the selected card.</p>
24	RW	0x0	<p>enable_boot Enable Boot—this bit should be set only for mandatory boot mode. When Software sets this bit along with start_cmd, CIU starts the boot sequence for the corresponding card by asserting the CMD line low. Do NOT set disable_boot and enable_boot together.</p>

Bit	Attr	Reset Value	Description
23	RW	0x0	<p>ccs_expected 0: Interrupts are not enabled in CE-ATA device (nIEN = 1 in ATA control register), or command does not expect CCS from device 1: Interrupts are enabled in CE-ATA device (nIEN = 0), and RW_BLK command expects command completion signal from CE-ATA device.</p> <p>If the command expects Command Completion Signal (CCS) from the CE-ATA device, the software should set this control bit. Host controller sets Data Transfer Over (DTO) bit in RINTSTS register and generates interrupt to host if Data Transfer Over interrupt is not masked.</p>
22	RW	0x0	<p>read_ceata_device 0: Host is not performing read access (RW_REG or RW_BLK) towards CE-ATA device 1: Host is performing read access (RW_REG or RW_BLK) towards CE-ATA device</p> <p>Software should set this bit to indicate that CE-ATA device is being accessed for read transfer. This bit is used to disable read data timeout indication while performing CE-ATA read transfers. Maximum value of I/O transmission delay can be no less than 10 seconds. Host controller should not indicate read data timeout while waiting for data from CE-ATA device.</p>
21	RW	0x0	<p>update_clock_registers_only 0: normal command sequence 1: do not send commands, just update clock register value into card clock domain</p> <p>Following register values transferred into card clock domain: CLKDIV, CLRSRC, CLKENA. Changes card clocks (change frequency, truncate off or on, and set low-frequency mode); provided in order to change clock frequency or stop clock without having to send command to cards.</p> <p>During normal command sequence, when update_clock_registers_only = 0, following control registers are transferred from BIU to CIU: CMD, CMDARG, TMOUT, CTYPE, BLKSIZ, BYTCNT. CIU uses new register values for new command sequence to card.</p> <p>When bit is set, there are no Command Done interrupts because no command is sent to SD_MMC_CEATA cards.</p>
20:16	RW	0x00	<p>card_number Card number in use. Represents physical slot number of card being accessed. The recommend number is 1.</p>

Bit	Attr	Reset Value	Description
15	RW	0x0	<p>send_initialization 0: do not send initialization sequence (80 clocks of 1) before sending this command 1: send initialization sequence before sending this command After power on, 80 clocks must be sent to card for initialization before sending any commands to card. Bit should be set while sending first command to card so that controller will initialize clocks before sending command to card. This bit should not be set for either of the boot modes (alternate or mandatory).</p>
14	RW	0x0	<p>stop_abort_cmd 0: neither stop nor abort command to stop current data transfer in progress. If abort is sent to function-number currently selected or not in data-transfer mode, then bit should be set to 0. 1: stop or abort command intended to stop current data transfer in progress. When open-ended or predefined data transfer is in progress, and host issues stop or abort command to stop data transfer, bit should be set so that command/data state-machines of CIU can return correctly to idle state. This is also applicable for Boot mode transfers. To Abort boot mode, this bit should be set along with CMD[26] = disable_boot.</p>
13	RW	0x0	<p>wait_prvdata_complete 0: send command at once, even if previous data transfer has not completed 1: wait for previous data transfer completion before sending command The wait_prvdata_complete = 0 option typically used to query status of card during data transfer or to stop current data transfer; card_number should be same as in previous command.</p>
12	RW	0x0	<p>send_auto_stop 0: no stop command sent at end of data transfer 1: send stop command at end of data transfer When set, SDMMC Controller sends stop command to SD_MMC cards at end of data transfer. <ul style="list-style-type: none"> ● when send_auto_stop bit should be set, since some data transfers do not need explicit stop commands ● open-ended transfers that software should explicitly send to stop command Additionally, when "resume" is sent to resume -suspended memory access of SD-Combo card -bit should be set correctly if suspended data transfer needs send_auto_stop. Don't care if no data expected from card. </p>
11	RW	0x0	<p>transfer_mode 0: block data transfer command 1: stream data transfer command Don't care if no data expected.</p>

Bit	Attr	Reset Value	Description
10	RW	0x0	wr 0: read from card 1: write to card Don't care if no data expected from card.
9	RW	0x0	data_expected 0: no data transfer expected (read/write) 1: data transfer expected (read/write)
8	RW	0x0	check_response_crc 0: do not check response CRC 1: check response CRC Some of command responses do not return valid CRC bits. Software should disable CRC checks for those commands in order to disable CRC checking by controller
7	RW	0x0	response_length 0: short response expected from card 1: long response expected from card
6	RW	0x0	response_expect 0: no response expected from card 1: response expected from card
5:0	RW	0x00	cmd_index Command index

SDMMC_RESP0

Address: Operational Base + offset (0x0030)

Response-0 register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	response0 Bit[31:0] of response

SDMMC_RESP1

Address: Operational Base + offset (0x0034)

Response-1 register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	response Register represents bit[63:32] of long response. When CIU sends auto-stop command, then response is saved in register. Response for previous command sent by host is still preserved in Response 0 register. Additional auto-stop issued only for data transfer commands, and response type is always "short" for them.

SDMMC_RESP2

Address: Operational Base + offset (0x0038)

Response-2 register

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

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	response2 Bit[95:64] of long response

SDMMC_RESP3

Address: Operational Base + offset (0x003c)

Response-3 register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	response3 Bit[127:96] of long response

SDMMC_MINTSTS

Address: Operational Base + offset (0x0040)

Masked interrupt-status register

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24	RO	0x0	sdio_interrupt Interrupt from SDIO card; SDIO interrupt for card enabled only if corresponding sdio_int_mask bit is set in Interrupt mask register (mask bit 1 enables interrupt; 0 masks interrupt). 0: no SDIO interrupt from card 1: SDIO interrupt from card
23:17	RO	0x0	reserved
16	RW	0x0	data_nobusy_int_status Data no busy Interrupt Status
15:0	RO	0x0000	int_status Interrupt enabled only if corresponding bit in interrupt mask register is set. [15]: End-bit error (read)/Write no CRC (EBE) [14]: Auto command done (ACD) [13]: Start-bit error (SBE) [12]: Hardware locked write error (HLE) [11]: FIFO underrun/overrun error (FRUN) [10]: Data starvation-by-host timeout (HTO) /Volt_switch_int [9]: Data read timeout (DRTO) [8]: Response timeout (RTO) [7]: Data CRC error (DCRC) [6]: Response CRC error (RCRC) [5]: Receive FIFO data request (RXDR) [4]: Transmit FIFO data request (TXDR) [3]: Data transfer over (DTO) [2]: Command done (CD) [1]: Response error (RE) [0]: Card detect (CD)

SDMMC_RINTSTS

Address: Operational Base + offset (0x0044)

Raw interrupt-status register

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24	RO	0x0	sdio_interrupt Interrupt from SDIO card; Writes to these bits clear them. Value of 1 clears bit and 0 leaves bit intact. 0: no SDIO interrupt from card 1: SDIO interrupt from card
23:17	RO	0x0	reserved
16	RW	0x0	data_nobusy_int_status Data no busy interrupt status
15:0	RO	0x0000	int_status Writes to bits clear status bit. Value of 1 clears status bit, and value of 0 leaves bit intact. Bits are logged regardless of interrupt mask status. [15]: End-bit error (read)/Write no CRC (EBE) [14]: Auto command done (ACD) [13]: Start-bit error (SBE) [12]: Hardware locked write error (HLE) [11]: FIFO underrun/overrun error (FRUN) [10]: Data starvation-by-host timeout (HTO) /Volt_switch_int [9]: Data read timeout (DRTO) [8]: Response timeout (RTO) [7]: Data CRC error (DCRC) [6]: Response CRC error (RCRC) [5]: Receive FIFO data request (RXDR) [4]: Transmit FIFO data request (TXDR) [3]: Data transfer over (DTO) [2]: Command done (CD) [1]: Response error (RE) [0]: Card detect (CD)

SDMMC_STATUS

Address: Operational Base + offset (0x0048)

Status register

Bit	Attr	Reset Value	Description
31	RO	0x0	dma_req DMA request signal state
30	RO	0x0	dma_ack DMA acknowledge signal state
29:17	RO	0x0000	fifo_count Number of filled locations in FIFO
16:11	RO	0x00	response_index Index of previous response, including any auto-stop sent by core
10	RO	0x1	data_state_mc_busy Data transmit or receive state-machine is busy

Bit	Attr	Reset Value	Description
9	RO	0x0	<p>data_busy Inverted version of raw selected card_data[0] 0: card data not busy 1: card data busy default value is 1 or 0 depending on cdata_in</p>
8	RO	0x0	<p>data_3_status Raw selected card_data[3]; checks whether card is present 0: card not present 1: card present default value is 1 or 0 depending on cdata_in</p>
7:4	RO	0x0	<p>command_fsm_states Command FSM states: 0: idle 1: send init sequence 2: Tx cmd start bit 3: Tx cmd tx bit 4: Tx cmd index + arg 5: Tx cmd crc7 6: Tx cmd end bit 7: Rx resp start bit 8: Rx resp IRQ response 9: Rx resp tx bit 10: Rx resp cmd idx 11: Rx resp data 12: Rx resp crc7 13: Rx resp end bit 14: Cmd path wait NCC 15: Wait; CMD-to-response turnaround The command FSM state is represented using 19 bits. The STATUS Register[7:4] has 4 bits to represent the command FSM states. Using these 4 bits, only 16 states can be represented. Thus three states cannot be represented in the STATUS[7:4] register. The three states that are not represented in the STATUS Register[7:4] are: 16: Wait for CCS 17: Send CCSD 18: Boot Mode Due to this, while command FSM is in "Wait for CCS state" or "Send CCSD" or "Boot Mode", the Status register indicates status as 0 for the bit field [7:4].</p>
3	RO	0x0	fifo_full FIFO is full status
2	RO	0x1	fifo_empty FIFO is empty status

Bit	Attr	Reset Value	Description
1	RO	0x1	fifo_tx_watermark FIFO reached Transmit watermark level; not qualified with data transfer
0	RO	0x0	fifo_rx_watermark FIFO reached Receive watermark level; not qualified with data transfer

SDMMC_FIFOTH

Address: Operational Base + offset (0x004c)

FIFO threshold register

Bit	Attr	Reset Value	Description
31	RO	0x0	reserved
30:28	RW	0x0	<p>dma_multiple_transaction_size Burst size of multiple transaction; should be programmed same as DMA controller multiple-transaction-size SRC/DEST_MSIZE.</p> <p>0: 1 transfers 1: 4 2: 8 3: 16 4: 32 5: 64 6: 128 7: 256</p> <p>The unit for transfer is the H_DATA_WIDTH parameter. A single transfer would be signalled based on this value.</p> <p>Value should be sub-multiple of $(RX_WMark + 1) * (F_DATA_WIDTH/H_DATA_WIDTH)$ and $(FIFO_DEPTH - TX_WMark) * (F_DATA_WIDTH/ H_DATA_WIDTH)$</p> <p>For example, if FIFO_DEPTH = 16, FDATA_WIDTH == H_DATA_WIDTH</p> <p>Allowed combinations for MSize and TX_WMark are:</p> <ul style="list-style-type: none"> MSize = 1, TX_WMARK = 1-15 MSize = 4, TX_WMark = 8 MSize = 4, TX_WMark = 4 MSize = 4, TX_WMark = 12 MSize = 8, TX_WMark = 8 MSize = 8, TX_WMark = 4 <p>Allowed combinations for MSize and RX_WMark are:</p> <ul style="list-style-type: none"> MSize = 1, RX_WMARK = 0-14 MSize = 4, RX_WMark = 3 MSize = 4, RX_WMark = 7 MSize = 4, RX_WMark = 11 MSize = 8, RX_WMark = 7 <p>Recommended:</p> <ul style="list-style-type: none"> MSize = 8, TX_WMark = 8, RX_WMark = 7

Bit	Attr	Reset Value	Description
27:16	RW	0x000	<p>rx_wmark FIFO threshold watermark level when receiving data to card. When FIFO data count reaches greater than this number, DMA/FIFO request is raised. During end of packet, request is generated regardless of threshold programming in order to complete any remaining data.</p> <p>In non-DMA mode, when receiver FIFO threshold (RXDR) interrupt is enabled, then interrupt is generated instead of DMA request. During end of packet, interrupt is not generated if threshold programming is larger than any remaining data. It is responsibility of host to read remaining bytes on seeing Data Transfer Done interrupt.</p> <p>In DMA mode, at end of packet, even if remaining bytes are less than threshold, DMA request does single transfers to flush out any remaining bytes before Data Transfer Done interrupt is set. 12 bits-1 bit less than FIFO-count of status register, which is 13 bits.</p> <p>Limitation: RX_WMark <= FIFO_DEPTH-2 Recommended: (FIFO_DEPTH/2) - 1; (means greater than (FIFO_DEPTH/2) - 1)</p> <p>NOTE: In DMA mode during CCS time-out, the DMA does not generate the request at the end of packet, even if remaining bytes are less than threshold. In this case, there will be some data left in the FIFO. It is the responsibility of the application to reset the FIFO after the CCS timeout.</p>
15:12	RO	0x0	reserved
11:0	RW	0x000	<p>tx_wmark FIFO threshold watermark level when transmitting data to card. When FIFO data count is less than or equal to this number, DMA/FIFO request is raised. If Interrupt is enabled, then interrupt occurs. During end of packet, request or interrupt is generated, regardless of threshold programming.</p> <p>In non-DMA mode, when transmit FIFO threshold (TXDR) interrupt is enabled, then interrupt is generated instead of DMA request. During end of packet, on last interrupt, host is responsible for filling FIFO with only required remaining bytes (not before FIFO is full or after CIU completes data transfers, because FIFO may not be empty).</p> <p>In DMA mode, at end of packet, if last transfer is less than burst size, DMA controller does single cycles until required bytes are transferred.</p> <p>12 bits -1 bit less than FIFO-count of status register, which is 13 bits.</p> <p>Limitation: TX_WMark >= 1; Recommended: FIFO_DEPTH/2; (means less than or equal to FIFO_DEPTH/2)</p>

SDMMC_CDETECT

Address: Operational Base + offset (0x0050)

Card-detect register

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RO	0x0	card_detect_n Value on card_detect_n input ports; read-only bits. 0 represents presence of card.

SDMMC_WRTPRT

Address: Operational Base + offset (0x0054)

Write-protect register

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	write_protect Value on card_write_prt input port. 1 represents write protection.

SDMMC_TCBCNT

Address: Operational Base + offset (0x005c)

Transferred CIU card byte count

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	trans_card_byte_count Number of bytes transferred by CIU unit to card. In 32-bit or 64-bit AMBA data-bus-width modes, register should be accessed in full to avoid read-coherency problems. In 16-bit AMBA data-bus-width mode, internal 16-bit coherency register is implemented. User should first read lower 16 bits and then higher 16 bits. When reading lower 16 bits, higher 16 bits of counter are stored in temporary register. When higher 16 bits are read, data from temporary register is supplied. Both TCBCNT and TBBCNT share same coherency register. When AREA_OPTIMIZED parameter is 1, register should be read only after data transfer completes; during data transfer, register returns 0.

SDMMC_TBBCNT

Address: Operational Base + offset (0x0060)

Transferred host/DMA to/from BIU-FIFO byte count

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	<p>trans_fifo_byte_count Number of bytes transferred between Host/DMA memory and BIU FIFO.</p> <p>In 32-bit or 64-bit AMBA data-bus-width modes, register should be accessed in full to avoid read-coherency problems. In 16-bit AMBA data-bus-width mode, internal 16-bit coherency register is implemented. User should first read lower 16 bits and then higher 16 bits. When reading lower 16 bits, higher 16 bits of counter are stored in temporary register. When higher 16 bits are read, data from temporary register is supplied.</p> <p>Both TCBCNT and TBBCNT share same coherency register.</p>

SDMMC_DEBNCE

Address: Operational Base + offset (0x0064)

Card detect debounce register

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved
23:0	RW	0xffffffff	<p>debounce_count Number of host clocks (clk) used by debounce filter logic; typical debounce time is 5-25 ms.</p>

SDMMC_USRID

Address: Operational Base + offset (0x0068)

User ID register

Bit	Attr	Reset Value	Description
31:0	RW	0x07967797	<p>usrid User identification register</p>

SDMMC_VERID

Address: Operational Base + offset (0x006c)

Synopsys version ID register

Bit	Attr	Reset Value	Description
31:0	RO	0x5342270a	<p>verid Version identification register</p>

SDMMC_UHS_REG

Address: Operational Base + offset (0x0074)

UHS-1 register

Bit	Attr	Reset Value	Description
31:17	RO	0x0	reserved

Bit	Attr	Reset Value	Description
16	RW	0x0	ddr_reg DDR mode. Determines the voltage fed to the buffers by an external voltage regulator. 0: non-DDR mode 1: DDR mode UHS_REG [16] should be set for card.
15:1	RO	0x0	reserved
0	RW	0x0	reserved

SDMMC_RST_n

Address: Operational Base + offset (0x0078)

Hardware reset register

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x1	card_reset Hardware reset. 0: active mode 1: reset These bits cause the cards to enter pre-idle state, which requires them to be re-initialized. It should be set to 1'b1 to reset card.

SDMMC_PLDMND

Address: Operational Base + offset (0x0084)

Poll Demand Register

Bit	Attr	Reset Value	Description
31:0	WO	0x00000000	PD Poll Demand. If the OWN bit of a descriptor is not set, the FSM goes to the Suspend state.

SDMMC_CARDTHRCTL

Address: Operational Base + offset (0x0100)

Card read threshold enable

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27:16	RW	0x000	CardRdThreshold Card Read Threshold size
15:2	RO	0x0	reserved

Bit	Attr	Reset Value	Description
1	RW	0x0	BsyClrIntEn Busy Clear Interrupt generation: 0: Busy Clear Interrupt disabled 1: Busy Clear Interrupt enabled Note: The application can disable this feature if it does not want to wait for a Busy Clear Interrupt. For example, in a multi-card scenario, the application can switch to the other card without waiting for a busy to be completed. In such cases, the application can use the polling method to determine the status of busy. By default this feature is disabled and backward-compatible to the legacy drivers where polling is used.
0	RW	0x0	CardRdThrEn Card Read Threshold Enable. 0: Card Read Threshold disabled 1: Card Read Threshold enabled. Host Controller initiates Read Transfer only if CardRdThreshold amount of space is available in receive FIFO.

SDMMC_BACK_END_POWER

Address: Operational Base + offset (0x0104)

Back-end power

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	back_end_power Back end power 0: Off; Reset 1: Back-end Power supplied to card application

SDMMC_EMMC_DDR_REG

Address: Operational Base + offset (0x010c)

eMMC 4.5 DDR START Bit Detection Control Register

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	HALF_START_BIT Control for start bit detection mechanism inside Host controller based on duration of start bit; each bit refers to one slot. For eMMC 4.5, start bit can be: 0: Full cycle (HALF_START_BIT = 0) 1: Less than one full cycle (HALF_START_BIT = 1) Set HALF_START_BIT=1 for eMMC 4.5 and above; set to 0 for SD applications.

SDMMC_ENABLE_SHIFT

Address: Operational Base + offset (0x0110)

Enable Phase Shift Register

Bit	Attr	Reset Value	Description
31:0	RO	0x0	reserved
0	RW	0x0	enable_shift Control for the amount of phase shift provided on the default enables in the design. Two bits are assigned for each card/slot. For example, bits[1:0] control slot0 and indicate the following. 0: Default phase shift 1: Enables shifted to next immediate positive edge 2: Enables shifted to next immediate negative edge 3: Reserved

SDMMC_FIFO_BASE

Address: Operational Base + offset (0x0200)

FIFO Base Address

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	fifo_base_addr fifo base addr

1.5 Interface Description

The interface and IOMUX setting for SDMMC, SDIO, EMMC are shown as follows.

Table 1-4 SDMMC Interface Description

Module Pin	Direction	Pad Name	IOMUX Setting
sdmmc_cclk	O	IO_MMC0clkout_GPIO1c0	GRF_GPIO1C_IOMUX[1:0]=2'b01
sdmmc_ccmd	I/O	IO_MMC0cmd_GPIO1b7	GRF_GPIO1B_IOMUX[15:14]=2'b01
sdmmc_cdata0	I/O	IO_MMC0d0_UART2sin_GPIO1c2	GRF_GPIO1C_IOMUX[5:4]=2'b01
sdmmc_cdata1	I/O	IO_MMC0d1_UART2sout_GPIO1c3	GRF_GPIO1C_IOMUX[7:6]=2'b01
sdmmc_cdata2	I/O	IO_MMC0d2_JTAGtck_GPIO1c4	GRF_GPIO1C_IOMUX[9:8]=2'b01
sdmmc_cdata3	I/O	IO_MMC0d3_JTAGtms_GPIO1c5	GRF_GPIO1C_IOMUX[11:10]=2'b01
sdmmc_cdetectn	I	IO_MMC0detn_GPIO1c1	GRF_GPIO1C_IOMUX[3:2]=2'b01

Notes: I=input, O=output, I/O=input/output, bidirectional

Table 1-5 SDIO Interface Description

Module Pin	Direction	Pad Name	IOMUX Setting
sdio_cclk	O	IO_MMC1clkout_I2S1mclk_GPIO0b1	GRF_GPIO0B_IOMUX[3:2]=2'b01
sdio_ccmd	I/O	IO_MMC1cmd_I2S1sdo_GPIO0b0	GRF_GPIO0B_IOMUX[1:0]=2'b01
sdio_cdata0	I/O	IO_MMC1d0_I2S1lrckrx_GPIO0b3	GRF_GPIO0B_IOMUX[7:6]=2'b01
sdio_cdata1	I/O	IO_MMC1d1_I2S1lrcktx_GPIO0b4	GRF_GPIO0B_IOMUX[9:8]=2'b01
sdio_cdata2	I/O	IO_MMC1d2_I2S1sdi_GPIO0b5	GRF_GPIO0B_IOMUX[11:10]=2'b01
sdio_cdata3	I/O	IO_MMC1d3_I2S1sclk_GPIO0b6	GRF_GPIO0B_IOMUX[13:12]=2'b01

Notes: I=input, O=output, I/O=input/output, bidirectional

Table 1-6 EMMC Interface Description

Module Pin	Direction	Pad Name	IOMUX Setting
emmc_cclk	O	IO_NANDcle_EMMCclkout_GPIO2a1	GRF_GPIO2A_IOMUX[3:2]=2'b10
emmc_ccmd	I/O	IO_NANDrdy_EMMCCmd_SFCclk_GPIO2a4	GRF_GPIO2A_IOMUX[9:8]=2'b10
emmc_cdata0	I/O	IO_NANDd0_EMMCd0_SFCsio0_GP1d0	GRF_GPIO1D_IOMUX[1:0]=2'b10

Module Pin	Direction	Pad Name	IOMUX Setting
emmc_cdata1	I/O	IO_NANDd1_EMMCd1_SFCsio1_GP IO1d1	GRF_GPIO1D_IOMUX[3:2]=2'b10
emmc_cdata2	I/O	IO_NANDd2_EMMCd2_SFCsio2_GP IO1d2	GRF_GPIO1D_IOMUX[5:4]=2'b10
emmc_cdata3	I/O	IO_NANDd3_EMMCd3_SFCsio3_GP IO1d3	GRF_GPIO1D_IOMUX[7:6]=2'b10
emmc_cdata4	I/O	IO_NANDd4_EMMCd4_SPIrxd_GPI O1d4	GRF_GPIO1D_IOMUX[9:8]=2'b10
emmc_cdata5	I/O	IO_NANDd5_EMMCd5_SPItxd_GPI O1d5	GRF_GPIO1D_IOMUX[11:10]=2'b10
emmc_cdata6	I/O	IO_NANDd6_EMMCd6_SPIcsn0_GPI IO1d6	GRF_GPIO1D_IOMUX[13:12]=2'b10
emmc_cdata7	I/O	IO_NANDd7_EMMCd7_SPIcsn1_GPI IO1d7	GRF_GPIO1D_IOMUX[15:14]=2'b10

Notes: I=input, O=output, I/O=input/output, bidirectional

1.6 Application Notes

1.6.1 Card-Detect and Write-Protect Mechanism

Following figure illustrates how the SD/MMC card detection and write-protect signals are connected. Most of the SD/MMC sockets have card-detect pins. When no card is present, card_detect_n is 1 due to the pull-up. When the card is inserted, the card-detect pin is shorted to ground, which makes card_detect_n go to 0. Similarly in SD cards, when the write-protect switch is toward the left, it shorts the write_protect port to ground.

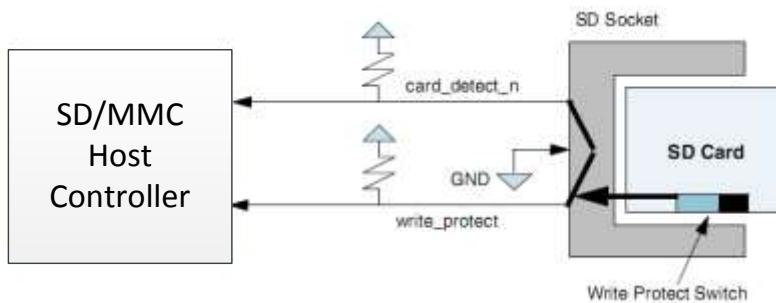


Fig. 1-6 SD/MMC Card-Detect and Write-Protect

1.6.2 SD/MMC Termination Requirement

Following Figure illustrates the SD/MMC termination requirements, which is required to pull up ccmd and cdata lines on the device bus. The recommended specification for pull-up on the ccmd line (Rcmd) is 4.7K - 100K for MMC, and 10K - 100K for an SD. The recommended pull-up on the cdata line (Rdat) is 50K - 100K.

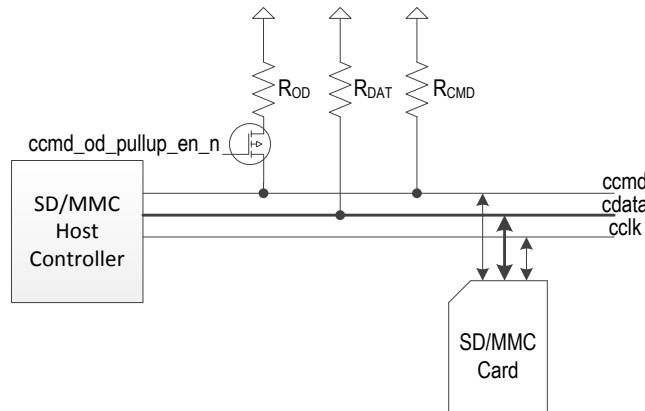


Fig. 1-7 SD/MMC Card Termination

1. Rcmd and Rod Calculation

The SD/MMC card enumeration happens at a very low frequency – 100-400KHz. Since the MMC bus is a shared bus between multiple cards, during enumeration open-drive mode is used to avoid bus conflict. Cards that drive 0 win over cards that drive “z”. The pull-up in the command line pulls the bus to 1 when all cards drive “z”. During normal data transfer, the host chooses only one card and the card driver switches to push-pull mode.

For example, if enumeration is done at 400KHz and the total bus capacitance is 200 pf, the pull-up needed during enumeration is:

$$2.2 \text{ RC} = \text{rise-time} = 1/400\text{KHz}$$

$$\begin{aligned} R &= 1/(2.2 * C * 100\text{KHz}) \\ &= 1/(2.2 \times 200 \times 10^{-12} \times 400 \times 10^3) \\ &= 1/(17.6 \times 10^{-5}) \\ &= 5.68\text{K} \end{aligned}$$

The ROD and RCMD should be adjusted in such a way that the effective pull-up is at the maximum 5.68K during enumeration. If there are only a few cards in the bus, a fixed RCMD register is sufficient and there is no need for an additional ROD pull-up during enumeration. You should also ensure the effective pull-up will not violate the I_{OL} rating of the drivers.

In SD mode, since each card has a separate bus, the capacitance is less, typically in the order of 20-30pf (host capacitance + card capacitance + trace + socket capacitance). For example, if enumeration is done at 400KHz and the total bus capacitance is 20pf, the pull-up needed during enumeration is:

$$2.2 \text{ RC} = \text{rise-time} = 1/400\text{KHz}$$

$$\begin{aligned} R &= 1/(2.2 * C * 100\text{KHz}) \\ &= 1/(2.2 \times 20 \times 10^{-12} \times 400 \times 10^3) \\ &= 1/(1.76 \times 10^{-5}) \\ &= 56.8\text{K} \end{aligned}$$

Therefore, a fixed 56.8K permanent Rcmd is sufficient in SD mode to enumerate the cards. The driver of the SD/MMC on the “command” port needs to be only a push-pull driver. During enumeration, the SD/MMC emulates an open-drain driver by driving only a 0 or a “z” by controlling the cc当地_out and cc当地_out_en signals.

1.6.3 Software/Hardware Restriction

Before issuing a new data transfer command, the software should ensure that the card is not busy due to any previous data transfer command. Before changing the card clock frequency, the software must ensure that there are no data or command transfers in progress.

If the card is enumerated in SDR50, or DDR50 mode, then the application must program the use_hold_reg bit[29] in the CMD register to 1'b0 (phase shift of cclk_in_drv = 0) or 1'b1 (phase shift of cclk_in_drv>0). If the card is enumerated in SDR12 or SDR25 mode, the application must program the use_hold_reg bit[29] in the CMD register to 1'b1.

This programming should be done for all data transfer commands and non-data commands that are sent to the card. When the use_hold_reg bit is programmed to 1'b0, the Host Controller bypasses the Hold Registers in the transmit path. The value of this bit should not be changed when a Command or Data Transfer is in progress. For more details on using use_hold_reg and the implementation requirements for meeting the Card input hold time,

refer to "Recommended Usage" in following table.

Table 1-7 Recommended Usage of use_hold_reg

No.	Speed Mode	use_hold_reg	cclk_in (MHz)	clk_in_drv (MHz)	clk_divider	Phase shift
1	SDR104	1'b0	200	200	0	0
2	SDR104	1'b1	200	200	0	Tunable > 0
3	SDR50	1'b0	100	100	0	0
4	SDR50	1'b1	100	100	0	Tunable > 0
5	DDR50 (8bit)	1'b0	100	100	1	0
6	DDR50 (8bit)	1'b1	100	100	1	Tunable > 0
7	DDR50 (4bit)	1'b0	50	50	0	0
8	DDR50 (4bit)	1'b1	50	50	0	Tunable > 0
9	SDR25	1'b1	50	50	0	Tunable > 0
10	SDR12	1'b1	50	50	1	Tunable > 0

To avoid glitches in the card clock outputs, the software should use the following steps when changing the card clock frequency:

- 1) Before disable the clocks, ensure that the card is not busy due to any previous data command. To determine this, check for 0 in bit9 of STATUS register.
- 2) Update the Clock Enable register to disable all clocks. To ensure completion of any previous command before this update, send a command to the CIU to update the clock registers by setting:

- start_cmd bit
- "update clock registers only" bits
- "wait_previous data complete" bit

Wait for the CIU to take the command by polling for 0 on the start_cmd bit.

- 3) Set the start_cmd bit to update the Clock Divider and/or Clock Source registers, and send a command to the CIU in order to update the clock registers; wait for the CIU to take the command.

- 4) Set start_cmd to update the Clock Enable register in order to enable the required clocks and send a command to the CIU to update the clock registers; wait for the CIU to take the command.

In non-DMA mode, while reading from a card, the Data Transfer Over (RINTSTS[3]) interrupt occurs as soon as the data transfer from the card is over. There still could be some data left in the FIFO, and the RX_WMark interrupt may or may not occur, depending on the remaining bytes in the FIFO. Software should read any remaining bytes upon seeing the Data Transfer Over (DTO) interrupt. While using the external DMA interface for reading from a card, the DTO interrupt occurs only after all the data is flushed to memory by the DMA interface unit.

While writing to a card in external DMA mode, if an undefined-length transfer is selected by setting the Byte Count Register to 0, the DMA logic will likely request more data than it will send to the card, since it has no way of knowing at which point the software will stop the transfer. The DMA request stops as soon as the DTO is set by the CIU.

If the software issues a controller_reset command by setting control register bit[0] to 1, all the CIU state machines are reset; the FIFO is not cleared. The DMA sends all remaining bytes to the host. In addition to a card-reset, if a FIFO reset is also issued, then:

- Any pending DMA transfer on the bus completes correctly
- DMA data read is ignored
- Write data is unknown(x)

Additionally, if dma_reset is also issued, any pending DMA transfer is abruptly terminated. When the DMA is used, the DMA controller channel should also be reset and reprogrammed. If any of the previous data commands do not properly terminate, then the software should issue the FIFO reset in order to remove any residual data, if any, in the FIFO. After asserting the FIFO reset, you should wait until this bit is cleared.

One data-transfer requirement between the FIFO and host is that the number of transfers should be a multiple of the FIFO data width (32bits). For example, you want to write only 15 bytes to an SD/MMC card (BYTCNT), the host should write 16 bytes to the FIFO or program the

DMA to do 16-byte transfers. The software can still program the Byte Count register to only 15, at which point only 15 bytes will be transferred to the card. Similarly, when 15 bytes are read from a card, the host should still read all 16 bytes from the FIFO.

It is recommended that you not change the FIFO threshold register in the middle of data transfers.

1.6.4 Programming Sequence

1. Initialization

Following figure illustrates the initialization flow.

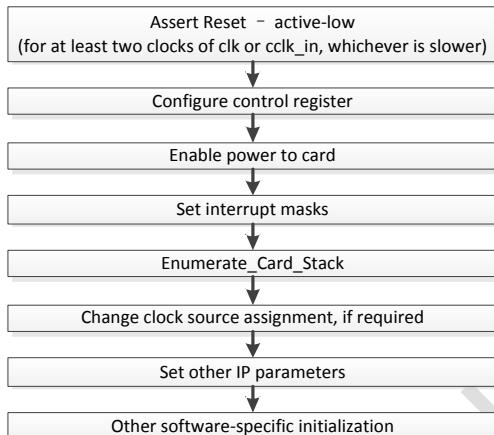


Fig. 1-8 Host Controller Initialization Sequence

Once the power and clocks are stable, reset_n should be asserted(active-low) for at least two clocks of clk or cclk_in, whichever is slower. The reset initializes the registers, ports, FIFO-pointers, DMA interface controls, and state-machines in the design. After power-on reset, the software should do the following:

- 1) Configure control register – For MMC mode, enable the open-drain pullup by setting enable_OD_pullup(bit24) in the control register.
- 2) Enable power to cards – Before enabling the power, confirm that the voltage setting to the voltage regulators is correct. Enable power to the connected cards by setting the corresponding bit to 1 in the Power Enable register. Wait for the power ramp-up time.
- 3) Set masks for interrupts by clearing appropriate bits in the Interrupt Mask register. Set the global int_enable bit of the Control register. It is recommended that you write 0xffff_ffff to the Raw Interrupt register in order to clear any pending interrupts before setting the int_enable bit.
- 4) Enumerate card stack – Each card is enumerated according to card type; for details, refer to “Enumerated Card Stack”. For enumeration, you should restrict the clock frequency to 400KHz.
- 5) Changing clock source assignment – set the card frequency using the clock-divider and clock-source registers; for details, refer to “Clock Programming”. MMC cards operate at a maximum of 20MHz (at maximum of 52MHz in high-speed mode). SD mode operates at a maximum of 25MHz (at maximum of 50MHz in high-speed mode).
- 6) Set other parameters, which normally do not need to be changed with every command, with a typical value such as timeout values in cclk_out according to SD/MMC specifications.
 - ResponseTimeOut = 0x64
 - DataTimeOut = highest of one of the following:
 - (10*((TAAC*Fop)+(100*NSAC))
 - Host FIFO read/write latency from FIFO empty/full
 - Set the debounce value to 25ms(default:0x0fffff) in host clock cycle units in the DEBNCE register.
 - FIFO threshold value in bytes in the FIFOTH register.

2. Enumerated Card Stack

The card stack does the following:

- Enumerates all connected cards
- Sets the RCA for the connected cards
- Reads card-specific information
- Stores card-specific information locally

Enumeration depends on the operating mode of the SD/MMC card; the card type is first identified and the appropriate card enumeration routine is called.

- 1) Check if the card is connected.
- 2) Clear the card type register to set the card width as a single bit. For the given card number, clear the corresponding bits in the card_type register. Clear the register bit for a 1-bit, 4-bit bus width. For example, for card number=1, clear bit 0 and bit 16 of the card_type register.
- 3) Set clock frequency to FOD=400KHz, maximum – Program clock divider0 (bits 0-7 in the CLKDIV register) value to one-half of the cclk_in frequency divided by 400KHz. For example, if cclk_in is 20MHz, then the value is $20,000/(2*400)=25$.
- 4) Identify the card type; that is, SD, MMC, or SDIO.
 - a. Send CMD5 first. If a response is received, then the card is SDIO
 - b. If not, send CMD8 with the following Argument

```
Bit[31:12] = 20'h0 //reserved bits  
Bit[11:8] = 4'b0001 //VHS value  
Bit[7:0] = 8'b10101010 //Preferred Check Pattern by SD2.0
```
 - c. If Response is received the card supports High Capacity SD2.0 then send ACMD41 with the following Argument

```
Bit[31] = 1'b0; //Reserved bits  
Bit[30] = 1'b1; //High Capacity Status  
Bit[29:24] = 6'h0; //Reserved bits  
Bit[23:0] = Supported Voltage Range
```
 - d. If Response is received for ACMD41 then the card is SD. Otherwise the card is MMC.
 - e. If response is not received for initial CMD8 then card does not support High Capacity SD2.0, then issue CMD0 followed by ACMD41 with the following Argument

```
Bit[31] = 1'b0; //Reserved bits  
Bit[30] = 1'b0; //High Capacity Status  
Bit[29:24] = 6'h0; //Reserved bits  
Bit[23:0] = Supported Voltage Range
```
- 5) Enumerate the card according to the card type.
- 6) Use a clock source with a frequency = Fod (that is, 400KHz) and use the following enumeration command sequence:
 - SD card – Send CMD0, CMD8, ACMD41, CMD2, CMD3.
 - MMC – Send CMD0, CMD1, CMD2, CMD3.

3. Power Control

You can implement power control using the following registers, along with external circuitry:

- Control register bits card_voltage_a and card_voltage_b – Status of these bits is reflected at the IO pins. The bits can be used to generate or control the supply voltage that the memory cards require.
- Power enable register – Control power to individual cards.

Programming these two register depends on the implemented external circuitry. While turning on or off the power enable, you should confirm that power supply settings are correct. Power to all cards usually should be disabled while switching off the power.

4. Clock Programming

The Host Controller supports one clock sources. The clock to an individual card can be enabled or disabled. Registers that support this are:

- CLKDIV – Programs individual clock source frequency. CLKDIV limited to 0 or 1 is recommended.
- CLKSRC – Assign clock source for each card.
- CLKENA – Enables or disables clock for individual card and enables low-power mode, which automatically stops the clock to a card when the card is idle for more than 8 clocks.

The Host Controller loads each of these registers only when the start_cmd bit and the Update_clk_regs_only bit in the CMD register are set. When a command is successfully loaded, the Host Controller clears this bit, unless the Host Controller already has another command in the queue, at which point it gives an HLE(Hardware Locked Error).

Software should look for the start_cmd and the Update_clk_regs_only bits, and should also set the wait_prvdata_complete bit to ensure that clock parameters do not change during data

transfer. Note that even though start_cmd is set for updating clock registers, the Host Controller does not raise a command_done signal upon command completion.

The following shows how to program these registers:

- 1) Confirm that no card is engaged in any transaction; if there is a transaction, wait until it finishes.
- 2) Stop all clocks by writing xxxx0000 to the CLKENA register. Set the start_cmd, Update_clk_regs_only, and wait_prvdata_complete bits in the CMD register. Wait until start_cmd is cleared or an HLE is set; in case of an HLE, repeat the command.
- 3) Program the CLKDIV and CLKSRC registers, as required. Set the start_cmd, Update_clk_regs_only, and wait_prvdata_complete bits in the CMD register. Wait until start_cmd is cleared or an HLE is set; in case of an HLE, repeat the command.
- 4) Re-enable all clocks by programming the CLKENA register. Set the start_cmd, Update_clk_regs_only, and wait_prvdata_complete bits in the CMD register. Wait until start_cmd is cleared or an HLE is set; in case of an HLE, repeat the command.

5. No-Data Command With or Without Response Sequence

To send any non-data command, the software needs to program the CMD register @0x2C and the CMDARG register @0x28 with appropriate parameters. Using these two registers, the Host Controller forms the command and sends it to the command bus. The Host Controller reflects the errors in the command response through the error bits of the RINTSTS register. When a response is received – either erroneous or valid – the Host Controller sets the command_done bit in the RINTSTS register. A short response is copied in Response Register0, while long response is copied to all four response registers @0x30, 0x34, 0x38, and 0x3C. The Response3 register bit 31 represents the MSB, and the Response0 register bit 0 represents the LSB of a long response.

For basic commands or non-data commands, follow these steps:

- 1) Program the Command register @0x28 with the appropriate command argument parameter.
- 2) Program the Command register @0x2C with the settings in following table.

Table 1-8 Command Settings for No-Data Command

Parameter	Value	Description
Default		
start_cmd	1	-
use_hold_reg	1/0	Choose value based on speed mode being used; ref to "use_hold_reg" on CMD register
update_clk_regs_only	0	No clock parameters update command
data_expected	0	No data command
card number	0	Actual card number (one controller only connect one card, the num is No. 0)
cmd_index	command-index	-
send_initialization	0	Can be 1, but only for card reset commands, such as CMD0
stop_abort_cmd	0	Can be 1 for commands to stop data transfer, such as CMD12
response_length	0	Can be 1 for R2(long) response
response_expect	1	Can be 0 for commands with no response; for example, CMD0, CMD4, CMD15, and so on
User-selectable		
wait_prvdata_complete	1	Before sending command on command line, host should wait for completion of any data command in process, if any

Parameter	Value	Description
		(recommended to always set this bit, unless the current command is to query status or stop data transfer when transfer is in progress)
check_response_crc	1	If host should crosscheck CRC of response received

- 1) Wait for command acceptance by host. The following happens when the command is loaded into the Host Controller:
 - Host Controller accepts the command for execution and clears the start_cmd bit in the CMD register, unless one command is in process, at which point the Host Controller can load and keep the second command in the buffer.
 - If the Host Controller is unable to load the command – that is, a command is already in progress, a second command is in the buffer, and a third command is attempted – then it generates an HLE (hardware-locked error).
- 2) Check if there is an HLE.
- 3) Wait for command execution to complete. After receiving either a response from a card or response timeout, the Host Controller sets the command_done bit in the RINTSTS register. Software can either poll for this bit or respond to a generated interrupt.
- 4) Check if response_timeout error, response_CRC error, or response error is set. This can be done either by responding to an interrupt raised by these errors or by polling bits 1, 6, and 8 from the RINTSTS register @0x44. If no response error is received, then the response is valid. If required, the software can copy the response from the response registers @0x30-0x3C.

Software should not modify clock parameters while a command is being executed.

6. Data Transfer Commands

Data transfer commands transfer data between the memory card and the Host Controller. To send a data command, the Host Controller needs a command argument, total data size, and block size. Software can receive or send data through the FIFO.

Before a data transfer command, software should confirm that the card is not busy and is in a transfer state, which can be done using the CMD13 and CMD7 commands, respectively.

For the data transfer commands, it is important that the same bus width that is programmed in the card should be set in the card type register @0x18.

The Host Controller generates an interrupt for different conditions during data transfer, which are reflected in the RINTSTS register @0x44 as:

- 1) Data_Transfer_Over (bit 3) – When data transfer is over or terminated. If there is a response timeout error, then the Host Controller does not attempt any data transfer and the “Data Transfer Over” bit is never set.
- 2) Transmit_FIFO_Data_request (bit 4) – FIFO threshold for transmitting data was reached; software is expected to write data, if available, in FIFO.
- 3) Receive_FIFO_Data_request (bit 5) – FIFO threshold for receiving data was reached; software is expected to read data from FIFO.
- 4) Data starvation by Host timeout (bit 10) – FIFO is empty during transmission or is full during reception. Unless software writes data for empty condition or reads data for full condition, the Host Controller cannot continue with data transfer. The clock to the card has been stopped.
- 5) Data read timeout error (bit 9) – Card has not sent data within the timeout period.
- 6) Data CRC error (bit 7) – CRC error occurred during data reception.
- 7) Start bit error (bit 13) – Start bit was not received during data reception.
- 8) End bit error (bit 15) – End bit was not received during data reception or for a write operation; a CRC error is indicated by the card.

Conditions 6, 7, and 8 indicate that the received data may have errors. If there was a response timeout, then no data transfer occurred.

7. Single-Block or Multiple-Block Read

Steps involved in a single-block or multiple-block read are:

- 1) Write the data size in bytes in the BYTCNT register @0x20.
- 2) Write the block size in bytes in the BLKSIZ register @0x1C. The Host Controller expects data from the card in blocks of size BLKSIZ each.
- 3) Program the CMDARG register @0x28 with the data address of the beginning of a data read.
- 4) Program the Command register with the parameters listed in following table. For SD and MMC cards, use CMD17 for a single-block read and CMD18 for a multiple-block read. For SDIO cards, use CMD53 for both single-block and multiple-block transfers.

Table 1-9 Command Setting for Single or Multiple-Block Read

Parameter	Value	Description
Default		
start_cmd	1	-
use_hold_reg	1/0	Choose value based on speed mode being used; ref to "use_hold_reg" on CMD register
update_clk_regs_only	0	No clock parameters update command
card number	0	Actual card number (one controller only connect one card, the num is No.0)
send_initialization	0	Can be 1, but only for card reset commands, such as CMD0
stop_abort_cmd	0	Can be 1 for commands to stop data transfer, such as CMD12
send_auto_stop	0/1	-
transfer_mode	0	Block transfer
read_write	0	Read from card
data_expected	1	Data command
response_length	0	Can be 1 for R2(long) response
response_expect	1	Can be 0 for commands with no response; for example, CMD0, CMD4, CMD15, and so on
User-selectable		
cmd_index	command-index	-
wait_prvdata_complete	1	0- Sends command immediately 1- Sends command after previous data transfer ends
check_response_crc	1	0- Host Controller should not check response CRC 1- Host Controller should check response CRC

After writing to the CMD register, the Host Controller starts executing the command; when the command is sent to the bus, the command_done interrupt is generated.

- Software should look for data error interrupts; that is, bits 7, 9, 13, and 15 of the RINTSTS register. If required, software can terminate the data transfer by sending a STOP command.
- Software should look for Receive_FIFO_Data_request and/or data starvation by host timeout conditions. In both cases, the software should read data from the FIFO and make space in the FIFO for receiving more data.
- When a Data_Transfer_Over interrupt is received, the software should read the remaining data from the FIFO.

8. Single-Block or Multiple-Block Write

Steps involved in a single-block or multiple-block write are:

- 1) Write the data size in bytes in the BYTCNT register @0x20.

- 2) Write the block size in bytes in the BLKSIZ register @0x1C; the Host Controller sends data in blocks of size BLKSIZ each.
- 3) Program CMDARG register @0x28 with the data address to which data should be written.
- 4) Write data in the FIFO; it is usually best to start filling data the full depth of the FIFO.
- 5) Program the Command register with the parameters listed in following table.

Table 1-10 Command Settings for Single or Multiple-Block Write

Parameter	Value	Description
Default		
start_cmd	1	-
use_hold_reg	1/0	Choose value based on speed mode being used; ref to "use_hold_reg" on CMD register
update_clk_regs_only	0	No clock parameters update command
card number	0	Actual card number (one controller only connect one card, the num is No. 0)
send_initialization	0	Can be 1, but only for card reset commands, such as CMD0
stop_abort_cmd	0	Can be 1 for commands to stop data transfer, such as CMD12
send_auto_stop	0/1	-
transfer_mode	0	Block transfer
read_write	1	Write to card
data_expected	1	Data command
response_length	0	Can be 1 for R2(long) response
response_expect	1	Can be 0 for commands with no response; for example, CMD0, CMD4, CMD15, and so on
User-selectable		
cmd_index	command-index	-
wait_prvdata_complete	1	0- Sends command immediately 1- Sends command after previous data transfer ends
check_response_crc	1	0- Host Controller should not check response CRC 1- Host Controller should check response CRC

After writing to the CMD register, Host Controller starts executing a command; when the command is sent to the bus, a command_done interrupt is generated.

- Software should look for data error interrupts; that is, for bits 7, 9, and 15 of the RINTSTS register. If required, software can terminate the data transfer by sending the STOP command.
- Software should look for Transmit_FIFO_Data_Request and/or timeout conditions from data starvation by the host. In both cases, the software should write data into the FIFO.
- When a Data_Transfer_Over interrupt is received, the data command is over. For an open-ended block transfer, if the byte count is 0, the software must send the STOP command. If the byte count is not 0, then upon completion of a transfer of a given number of bytes, the Host Controller should send the STOP command, if necessary. Completion of the AUTO-STOP command is reflected by the Auto_command_done interrupt – bit 14 of the RINTSTS register. A response to AUTO_STOP is stored in RESP1 @0x34.

9. Stream Read

A stream read is like the block read mentioned in "Single-Block or Multiple-Block Read", except for the following bits in the Command register:

```
transfer_mode = 1; //Stream transfer  
cmd_index = CMD20;
```

A stream transfer is allowed for only a single-bit bus width.

10. Stream Write

A stream write is exactly like the block write mentioned in "Single-Block or Multiple-Block Write", except for the following bits in the Command register:

```
transfer_mode = 1;//Stream transfer  
cmd_index = CMD11;
```

In a stream transfer, if the byte count is 0, then the software must send the STOP command. If the byte count is not 0, then when a given number of bytes completes a transfer, the Host Controller sends the STOP command. Completion of this AUTO_STOP command is reflected by the Auto_command_done interrupt. A response to an AUTO_STOP is stored in the RESP1 register@0x34.

A stream transfer is allowed for only a single-bit bus width.

11. Packed Commands

In order to reduce overhead, read and write commands can be packed in groups of commands—either all read or all write—that transfer the data for all commands in the group in one transfer on the bus.

Packed commands can be of two types:

- Packed Write: CMD23 →CMD25
- Packed Read: CMD23 → CMD25 → CMD23 → CMD18

Packed commands are put in packets by the application software and are transparent to the core.

12. Sending Stop or Abort in Middle of Transfer

The STOP command can terminate a data transfer between a memory card and the Controller, while the ABORT command can terminate an I/O data transfer for only the SDIO_IOONLY and SDIO_COMBO cards.

- Send STOP command – Can be sent on the command line while a data transfer is in progress; this command can be sent at any time during a data transfer.

You can also use an additional setting for this command in order to set the Command register bits (5-0) to CMD12 and set bit 14 (stop_abort_cmd) to 1. If stop_abort_cmd is not set to 1, the Controller does not know that the user stopped a data transfer. Reset bit 13 of the Command register (wait_prvdata_complete) to 0 in order to make the Controller send the command at once, even though there is a data transfer in progress.

- Send ABORT command – Can be used with only an SDIO_IOONLY or SDIO_COMBO card. To abort the function that is transferring data, program the function number in ASx bits (CCCR register of card, address 0x06, bits (0-2) using CMD52.

13. Suspend or Resume Sequence

In an SDIO card, the data transfer between an I/O function and the Controller can be temporarily halted using the SUSPEND command; this may be required in order to perform a high-priority data transfer with another function. When desired, the data transfer can be resumed using the RESUME command.

The following functions can be implemented by programming the appropriate bits in the CCCR register (Function 0) of the SDIO card. To read from or write to the CCCR register, use the CMD52 command.

- SUSPEND data transfer – Non-data command
- 1) Check if the SDIO card supports the SUSPEND/RESUME protocol; this can be done through the SBS bit in the CCCR register @0x08 of the card.
- 2) Check if the data transfer for the required function number is in process; the function number that is currently active is reflected in bits 0-3 of the CCCR register @0x0D. Note that if the BS bit (address 0xc::bit 0) is 1, then only the function number given by the FSx bits is valid.
- 3) To suspend the transfer, set BR (bit 2) of the CCCR register @0x0C.
- 4) Poll for clear status of bits BR (bit 1) and BS (bit 0) of the CCCR @0x0C. The BS (Bus Status) bit is 1 when the currently-selected function is using the data bus; the BR (Bus Release) bit

remains 1 until the bus release is complete. When the BR and BS bits are 0, the data transfer from the selected function has been suspended.

- RESUME data transfer – This is a data command
- 1) Check that the card is not in a transfer state, which confirms that the bus is free for data transfer.
 - 2) If the card is in a disconnect state, select it using CMD7. The card status can be retrieved in response to CMD52/CMD53 commands.
 - 3) Check that a function to be resumed is ready for data transfer; this can be confirmed by reading the RFx flag in CCCR @0x0F. If RF = 1, then the function is ready for data transfer.
 - 4) To resume transfer, use CMD52 to write the function number at FSx bits (0-3) in the CCCR register @0x0D. Form the command argument for CMD52 and write it in CMDARG @0x28.
 - 5) Write the block size in the BLKSIZ register @0x1C; data will be transferred in units of this block size.
 - 6) Write the byte count in the BYTCNT register @0x20. This is the total size of the data; that is, the remaining bytes to be transferred. It is the responsibility of the software to handle the data.
 - 7) Program Command register; similar to a block transfer.
 - 8) When the Command register is programmed, the command is sent and the function resumes data transfer. Read the DF flag (Resume Data Flag). If it is 1, then the function has data for the transfer and will begin a data transfer as soon as the function or memory is resumed. If it is 0, then the function has no data for the transfer.
 - 9) If the DF flag is 0, then in case of a read, the Host Controller waits for data. After the data timeout period, it gives a data timeout error.

6. Read_Wait Sequence

Read_wait is used with only the SDIO card and can temporarily stall the data transfer—either from function or memory—and allow the host to send commands to any function within the SDIO device. The host can stall this transfer for as long as required. The Host Controller provides the facility to signal this stall transfer to the card. The steps for doing this are:

- 1) Check if the card supports the read_wait facility; read SRW (bit 2) of the CCCR register @0x08. If this bit is 1, then all functions in the card support the read_wait facility. Use CMD52 to read this bit.
- 2) If the card supports the read_wait signal, then assert it by setting the read_wait (bit 6) in the CTRL register @0x00.
- 3) Clear the read_wait bit in the CTRL register.

14. Controller/DMA/FIFO Reset Usage

- Controller reset – Resets the controller by setting the controller_reset bit (bit 0) in the CTRL register; this resets the CIU and state machines, and also resets the BIU-to-CIU interface. Since this reset bit is self-clearing, after issuing the reset, wait until this bit is cleared.
- FIFO reset - Resets the FIFO by setting the fifo_reset bit (bit 1) in the CTRL register; this resets the FIFO pointers and counters of the FIFO. Since this reset bit is self-clearing, after issuing the reset, wait until this bit is cleared.

In external DMA transfer mode, even when the FIFO pointers are reset, if there is a DMA transfer in progress, it could push or pop data to or from the FIFO; the DMA itself completes correctly. In order to clear the FIFO, the software should issue an additional FIFO reset and clear any FIFO underrun or overrun errors in the RAWINTS register caused by the DMA transfers after the FIFO was reset.

15. Card Read Threshold

When an application needs to perform a Single or Multiple Block Read command, the application must program the CardThrCtl register with the appropriate Card Read Threshold size (CardRdThreshold) and set the Card Read Threshold Enable (CardRdThrEnable) bit to 1'b1. This additional programming ensures that the Host controller sends a Read Command only if there is space equal to the CardRDThreshold available in the Rx FIFO. This in turn ensures that the card clock is not stopped in the middle of a block of data being transmitted from the card. The Card Read Threshold can be set to the block size of the transfer, which guarantees that there is a minimum of one block size of space in the RxFIFO before the controller enables the card clock. The Card Read Threshold is required when the Round Trip

Delay is greater than 0.5cclk_in period.

16. Error Handling

The Host Controller implements error checking; errors are reflected in the RAWINTS register@0x44 and can be communicated to the software through an interrupt, or the software can poll for these bits. Upon power-on, interrupts are disabled (int_enable in the CTRL register is 0), and all the interrupts are masked (bits 0-31 of the INTMASK register; default is 0).

Error handling:

- Response and data timeout errors – For response timeout, software can retry the command. For data timeout, the Host Controller has not received the data start bit – either for the first block or the intermediate block – within the timeout period, so software can either retry the whole data transfer again or retry from a specified block onwards. By reading the contents of the TCBCNT later, the software can decide how many bytes remain to be copied.
- Response errors – Set when an error is received during response reception. In this case, the response that copied in the response registers is invalid. Software can retry the command.
- Data errors – Set when error in data reception are observed; for example, data CRC, start bit not found, end bit not found, and so on. These errors could be set for any block-first block, intermediate block, or last block. On receipt of an error, the software can issue a STOP or ABORT command and retry the command for either whole data or partial data.
- Hardware locked error – Set when the Host Controller cannot load a command issued by software. When software sets the start_cmd bit in the CMD register, the Host Controller tries to load the command. If the command buffer is already filled with a command, this error is raised. The software then has to reload the command.
- FIFO underrun/overrun error – If the FIFO is full and software tries to write data in the FIFO, then an overrun error is set. Conversely, if the FIFO is empty and the software tries to read data from the FIFO, an underrun error is set. Before reading or writing data in the FIFO, the software should read the fifo_empty or fifo_full bits in the Status register.
- Data starvation by host timeout – Raised when the Host Controller is waiting for software intervention to transfer the data to or from the FIFO, but the software does not transfer within the stipulated timeout period. Under this condition and when a read transfer is in process, the software should read data from the FIFO and create space for further data reception. When a transmit operation is in process, the software should fill data in the FIFO in order to start transferring data to the card.
- CRC Error on Command – If a CRC error is detected for a command, the CE-ATA device does not send a response, and a response timeout is expected from the Host Controller. The ATA layer is notified that an MMC transport layer error occurred.

Notes: During a multiple-block data transfer, if a negative CRC status is received from the device, the data path signals a data CRC error to the BIU by setting the data CRC error bit in the RINTSTS register. It then continues further data transmission until all the bytes are transmitted.

1.6.5 Voltage Switching

The Host Controller supports SD 3.0 Ultra High Speed (UHS-1) and is capable of voltage switching in SD-mode, which can be applied to SD High-Capacity (SDHC) and SD Extended Capacity (SDXC) cards. UHS-1 supports only 4-bit mode.

However, whether the IO voltage of 1.8v supported or not is depended on the SoC design. SD 3.0 UHS-1 supports the following transfer speed modes for UHS-50 and/or UHS-104 cards:

- DS – default-speed up to 25MHz, 3.3V signaling
- HS – high-speed up to 50MHz, 3.3V signaling
- SDR12 – SDR up to SDR 25MHz, 1.8V signaling
- SDR25 – SDR up to 50MHz, 1.8V signaling
- SDR50 – SDR up to 100MHz, 1.8V signaling
- DDR50 – DDR up to 50MHz, 1.8V signaling

Voltage selection can be done in only SD mode. The first CMD0 selects the bus mode-either SD mode or SPI mode. The card must be in SD mode in order for 1.8V signaling mode to apply, during which time the card cannot be switched to SPI mode or 3.3V signaling without a power

cycle.

If the System BIOS in an embedded system already knows that it is connected to an SD 3.0 card, then the driver programs the Controller to initiate ACMD41. The software knows from the response of ACMD41 whether or not the card supports voltage switching to 1.8V.

- If bit 32 of ACMD41 response is 1'b1: card supports voltage switching and next command-CMD11-invokes voltage switching sequence. After CMD11 is started, the software must program the IO voltage selection register based on the soc architecture.
- If bit 32 of ACMD41 response is 1'b0: card does not support voltage switching and CMD11 should not be started.

If the card and host controller accept voltage switching, then they support UHS-1 modes of data transfer. After the voltage switch to 1.8V, SDR12 is the default speed.

Since the UHS-1 can be used in only 4-bit mode, the software must start ACMD6 and change the card data width to 4-bit mode; ACMD6 is driven in any of the UHS-1 speeds. If the host wants to select the DDR mode of data transfer, then the software must program the DDR_REG register in the CSR space with the appropriate card number.

To choose from any of the SDR or DDR modes, appropriate values should be programmed in the CLKDIV register.

1. Voltage Switch Operation

The Voltage Switch operation must be performed in SD mode only.

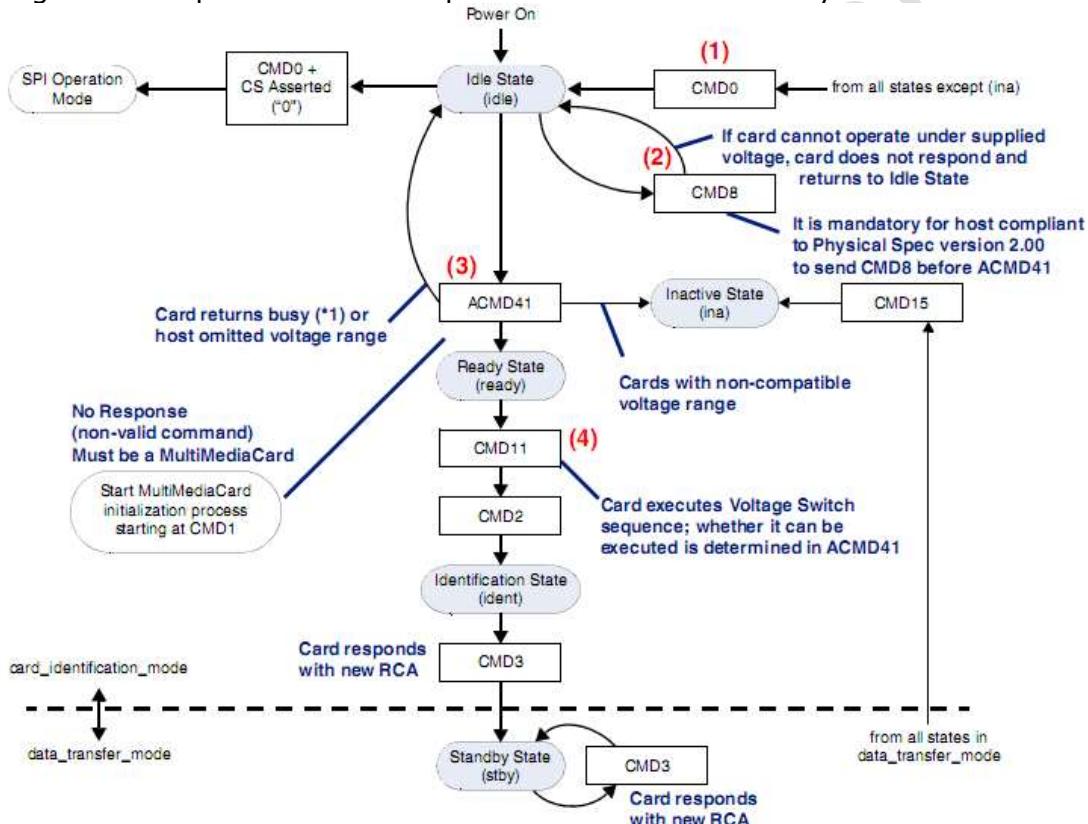


Fig. 1-9 Voltage Switching Command Flow Diagram

The following outlines the steps for the voltage switch programming sequence

- 1) Software Driver starts CMD0, which selects the bus mode as SD.
- 2) After the bus is in SD card mode, CMD8 is started in order to verify if the card is compatible with the SD Memory Card Specification, Version 2.00. CMD8 determines if the card is capable of working within the host supply voltage specified in the VHS (19:16) field of the CMD; the card supports the current host voltage if a response to CMD8 is received.
- 3) ACMD 41 is started. The response to this command informs the software if the card supports voltage switching; bits 38, 36, and 32 are checked by the card argument of ACMD41; refer to following figure.

47	46	45-40	39	38	37	36	35-33	32	31-16	15-08	07-01	00
S	D	Index	Busy 31	HCS 30	(FB) 29	XPC 28	Reserved 27-25	S18R 24	OCR 23-08	Reserved 07-00	CRC7	E
0	1	101001	0	X	0	X	000	X	xxxxh	0000000	xxxxxx	1

Host Capacity Support
0b: SDSC-only Host
1b: SDHC or SDXC supported

SCXC Power Control
0b: Power saving
1b: Maximum performance

S18R: Switching to 1.8V Request
0b: Use current signal voltage
1b: Switch to 1.8V signal voltage

Fig. 1-10 ACMD41 Argument

- Bit 30 informs the card if host supports SDHC/SDXC or not; this bit should be set to 1'b1.
- Bit 28 can be either 1 or 0.
- Bit 24 should be set to 1'b1, indicating that the host is capable of voltage switching; refer to following figure.

47	46	45-40	39	38	37	36-33	32	31-16	15-08	07-01	00
S	D	Index	Busy 31	CCS 30	Rsvd 29	Reserved 28-25	S18R 24	OCR 23-08	Reserved 07-00	CRC7	E
0	0	111111	X	X	0	0000	X	xxxxh	0000000	1111111	1

Busy Status
0b: On Initialization
1b: Initialization complete

Card Capacity Status
0b: SDSC
1b: SDHC or SDXC

S18R: Switching to 1.8V Accepted
0b: Continues current voltage signalling
1b: Ready for switching signal voltage

Fig. 1-11 ACMD41 Response(R3)

- Bit 30 – If set to 1'b1, card supports SDHC/SDXC; if set to 1'b0, card supports only SDSC
- Bit 24 – If set to 1'b1, card supports voltage switching and is ready for the switch
- Bit 31 – If set to 1'b1, initialization is over; if set to 1'b0, means initialization in process
- If the card supports voltage switching, then the software must perform the steps discussed for either the "Voltage Switch Normal Scenario" or the "Voltage Switch Error Scenario".

2. Voltage Switch Normal Scenario

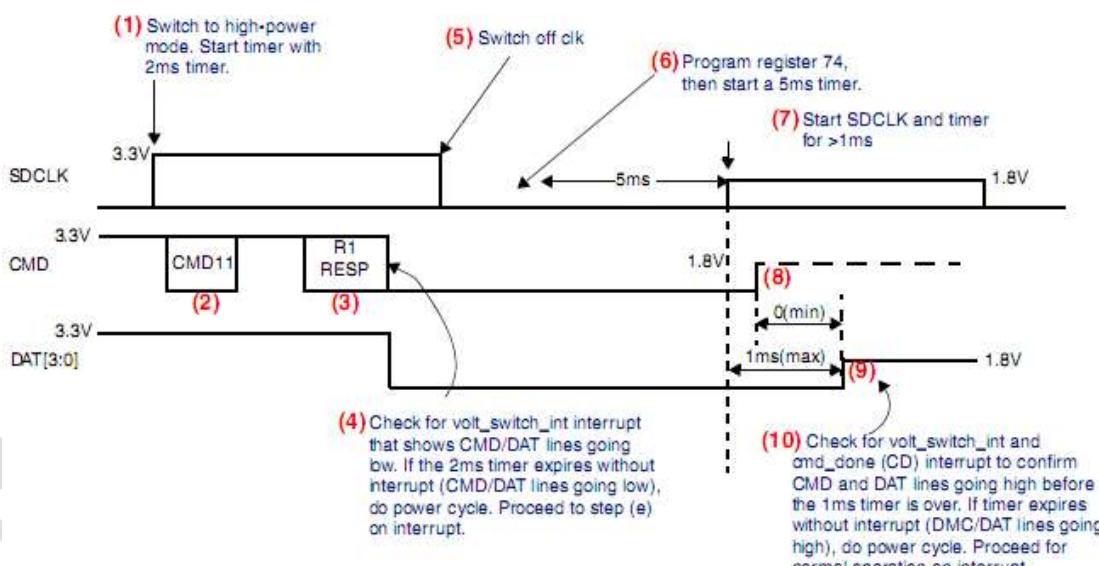


Fig. 1-12 Voltage Switch Normal Scenario

- The host programs CLKENA—cclk_low_power register—with zero (0) for the corresponding card, which makes the host controller move to high-power mode. The application should start a timer with a recommended value of 2ms; this value of 2 ms is determined as below:
Total clk required for CMD11 = 48 clks
Total clk required for RESP R1 = 48 clks
Maximum clk delay between MCD11 end to start of RESP1 = 60 clks
Total = 48+48 + 60 = 160
Minimum frequency during enumeration is 100 KHz; that is, 10us

Total time = $160 * 10\mu s = 1.6ms \sim 2ms$

- 2) The host issues CMD11 to start the voltage switch sequence. Set bit 28 to 1'b1 in CMD when setting CMD11; for more information on setting bits, refer to "Boot Operation".
- 3) The card returns R1 response; the host controller does not generate cmd_done interrupt on receiving R1 response.
- 4) The card drives CMD and DAT [3:0] to low immediately after the response. The host controller generates interrupt (VOLT_SWITCH_INT) once the CMD or DAT [3:0] line goes low. The application should wait for this interrupt. If the 2ms timer expires without an interrupt (CMD/DAT lines going low), do a power cycle.

Note: Before doing a power cycle, switch off the card clock by programming CLKENA register

Proceed to step (5) on getting an interrupt (VOLT_SWITCH_INT).

Note: This interrupt must be cleared once this interrupt is received. Additionally, this interrupt should not be masked during the voltage switch sequence.

If the timer expires without interrupt (CMD/DAT lines going low), perform a power cycle.

Proceed to step (5) on interrupt.

- 1) Program the CLKENA, cclk_enable register, with 0 for the corresponding card; the host stops supplying SDCLK.
- 2) Program Voltage register to the required values for the corresponding card. The application should start a timer > 5ms.
- 3) After the 5ms timer expires, the host voltage regulator is stable. Program CLKENA, cclk_enable register, with 1 for the corresponding card; the host starts providing SDCLK at 1.8V; this can be at zero time after Voltage register has been programmed. When the CLKENA register is programmed, the application should start another timer > 1ms.
- 4) By detecting SDCLK, the card drives CMD to high at 1.8V for at least one clock and then stops driving (tri-state); CMD is triggered by the rising edge of SDCLK (SDR timing).
- 5) If switching to 1.8V signaling is completed successfully, the card drives DAT [3:0] to high at 1.8V for at least one clock and then stops driving (tri-state); DAT [3:0] is triggered by the rising edge of SDCLK (SDR timing). DAT[3:0] must be high within 1ms from the start of SDCLK.
- 6) The host controller generates a voltage switch interrupt (VOLT_SWITCH_INT) and a command done (CD) interrupt once the CMD and DAT[3:0] lines go high. The application should wait for this interrupt to confirm CMD and DAT lines going high before the 1ms timer is done.

If the timer expires without the voltage switch interrupt (VOLT_SWITCH_INT), a power cycle should be performed. Program the CLKENA register to stop the clock for the corresponding card number. Wait for the cmd_done (CD) interrupt. Proceed for normal operation on interrupt. After the sequence is completed, the host and the card start communication in SDR12 timing.

3. Voltage Switch Error Scenario

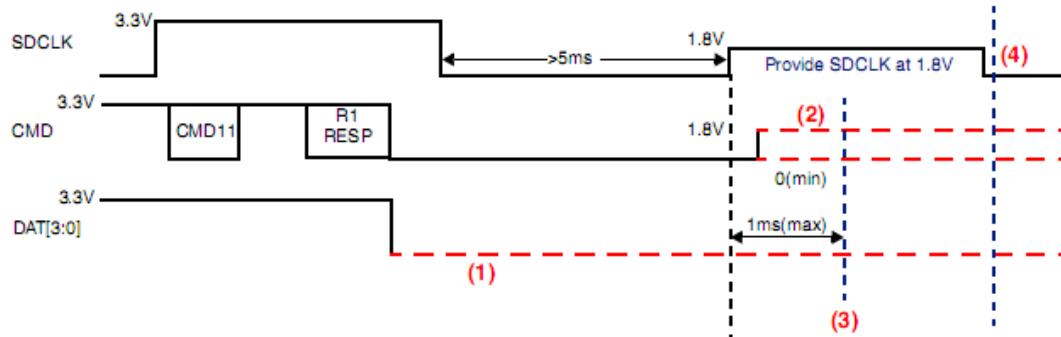


Fig. 1-13 Voltage Switch Error Scenario

- 1) If the interrupt (VOLT_SWITCH_INT) does not come, then the 2 ms timer should time out and a power cycle should be initiated.

Note: Before performing a power cycle, switch off the card clock by programming CLKENA register; no cmd_done (CD) interrupt is generated.

Additionally, if the card detects a voltage error at any point in between steps (5) and (7) in the card keeps driving DAT[3:0] to low until card power off.

- 2) CMD can be low or tri-state.
- 3) The host controller generates a voltage switch interrupt once the CMD and DAT[3:0] lines go high. The application should check for an interrupt to confirm CMD and DAT lines going

high before the 1 ms timer is done.

If the 1 ms timer expires without interrupt (VOLT_SWITCH_INT) and cmd_done (CD), a power cycle should be performed. Program the CLKENA register to stop SDCLK of the corresponding card. Wait for the cmd_done interrupt. Proceed for normal operation on interrupt.

4) If DAT[3:0] is low, the host drives SDCLK to low and then stops supplying the card power.

Note: The card checks voltages of its own regulator output and host signals to ensure they are less than 2.5V. Errors are indicated by (1) and (2).

- If voltage switching is accepted by the card, the default speed is SDR12.
- Command Done is given:
 - If voltage switching is properly done, CMD and DAT line goes high.
 - If switching is not complete, the 1ms timer expires, and the card clk is switched off.

Note: No other CMD should be driven before the voltage switching operation is completed and Command Done is received.

- The application should use CMD6 to check and select the particular function; the function appropriate-speed should be selected.

After the function switches, the application should program the correct value in the CLKDIV register, depending on the function chosen. Additionally, if Function 0x4 of the Access mode is chosen—that is, DDR50, then the application should also program 1'b1 in DDR_REG for the card number that has been selected for DDR50 mode.

1.6.6 Back-End Power

Each device needs one bit to control the back-end power supply for an embedded device; this bit does not control the VDDH of the host controller. A back_end_power register enables software programming for back-end power. The value on this register is output to the back_end_power signal, which can be used to switch power on and off the embedded device.

1.6.7 DDR Operation

1. 4-bit DDR Programming Sequence

DDR programming should be done only after the voltage switch operation has completed. The following outlines the steps for the DDR programming sequence:

- 1) Once the voltage switch operation is complete, the user must program voltage selection register to the required values for the corresponding card.
- To start a card to work in DDR mode, the application must program a bit of the newly defined UHS_REG[16] register with a value of 1'b1.
- The bit that the user programs depends on which card is to be accessed in DDR mode.
- 2) To move back to SDR mode, a power cycle should be run on the card—putting the card in SDR12 mode—and only then should UHS_REG[16] be set back to 1'b0 for the appropriate card.

2. 8-bit DDR Programming Sequence

The following outlines the steps for the 8-bit DDR programming sequence:

- 1) The cclk_in signal should be twice the speed of the required cclk_out. Thus, if the cclk_out signal is required to be 50 MHz, the cclk_in signal should be 100 MHz.
- 2) The CLKDIV register should always be programmed with a value higher than zero (0); that is, a clock divider should always be used for 8-bit DDR mode.
- 3) The application must program the UHS_REG[16] register (DDR_REG bits) by assigning it with a value of 1 for the bit corresponding to the card number; this causes the selected card to start working in DDR mode.
- 4) Depending on the card number, the CTYPE [31:16] bits should be set in order to make the host work in the 8-bit mode.

3. eMMC4.5 DDR START Bit

The eMMC4.5 changes the START bit definition in the following manner:

- 1) Receiver samples the START bit on the rising edge.
- 2) On the next rising edge after sampling the START bit, the receiver must sample the data.
- 3) Removes requirement of the START bit and END bit to be high for one full cycle.

Notes: The Host Controller does not support a START bit duration higher than one clock cycle. START bit durations of one or less than one clock cycle are supported and can be defined at the time of startup by programming the EMMC_DDR_REG register.

Following figure illustrates cases for the definition change of the START bit with eMMC4.5; it also illustrates how some of these cases can fail in sampling when higher-value delays are

considered for I/O PADS.

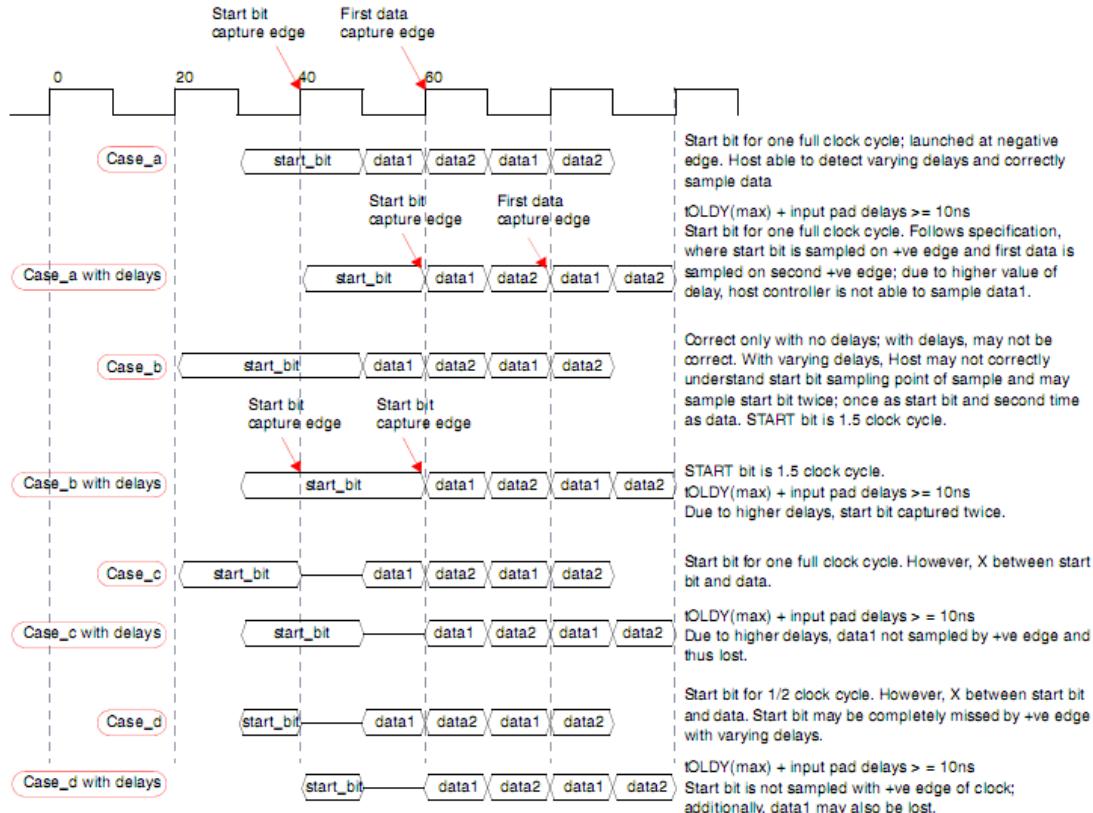


Fig. 1-14 CASES for eMMC 4.5 START bit

4. Reset Command/Moving from DDR50 to SDR12

To reset the mode of operation from DDR50 to SDR12, the following sequence of operations has to be done by the application:

- 1) Issue CMD0.

When CMD0 is received, the card changes from DDR50 to SDR12.

- 2) Program the CLKDIV register with an appropriate value.
- 3) Set DDR_REG to 0.

Note: The Voltage register should not be programmed to 0 while switching from DDR50 to SDR12, since the card is still operating in 1.8V mode after receiving CMD0.

1.6.8 H/W Reset Operation

When the RST_n signal goes low, the card enters a pre-idle state from any state other than the inactive state.

H/W Reset Programming Sequence

The following outlines the steps for the H/W reset programming sequence:

- 1) Program CMD12 to end any transfer in process.
- 2) Wait for DTO, even if no response is sent back by the card.
- 3) Set the following resets:
 - DMA reset- CTRL[2]
 - FIFO reset – CTRL[1] bits

Note: The above steps are required only if a transfer is in process.

- 4) Program the CARD_RESET register with a value of 0; this can be done at any time when the card is connected to the controller. This programming asserts the RST_n signal and resets the card.
- 5) Wait for minimum of 1 μ s or cclk_in period, whichever is greater
- 6) After a minimum of 1 μ s, the application should program a value of 0 into the CARD_RESET register. This de-asserts the RST_n signal and takes the card out of reset.
- 7) The application can program a new CMD only after a minimum of 200 μ s after the de-assertion of the RST_n signal, as per the MMC 4.41 standard.

Note: For backward compatibility, the RST_n signal is temporarily disabled in the card by default. The host may need to set the signal as either permanently enabled or permanently disabled before it uses the card.

1.6.9 FBE Scenarios

An FBE occurs due to an AHB error response on the AHB bus. This is a system error, so the software driver should not perform any further programming to the Host. The only recovery mechanism from such scenarios is to do one of the following:

- Issue a hard reset by asserting the `reset_n` signal
- Do a program controller reset by writing to the `CTRL[0]` register

1. FIFO Overflow and Underflow

During normal data transfer conditions, FIFO overflow and underflow will not occur. However if there is a programming error, then FIFO overflow/underflow can result. For example, consider the following scenarios.

- For transmit: PBL=4, Tx watermark = 1. For the above programming values, if the FIFO has only one location empty, it issues a `dma_req` to IDMAC FSM. Due to PBL value=4, the IDMAC FSM performs 4 pushes into the FIFO. This will result in a FIFO overflow interrupt.
- For receive: PBL=4, Rx watermark = 1. For the above programming values, if the FIFO has only one location filled, it issues a `dma_req` to IDMAC FSM. Due to PBL value=4, the IDMAC FSM performs 4 pops to the FIFO. This will result in a FIFO underflow interrupt.

The driver should ensure that the number of bytes to be transferred as indicated in the descriptor should be a multiple of 4bytes with respect to `H_DATA_WIDTH=32`. For example, if the `BYTCNT = 13`, the number of bytes indicated in the descriptor should be 16 for `H_DATA_WIDTH=32`.

2. Programming of PBL and Watermark Levels

The DMAC performs data transfers depending on the programmed PBL and threshold values.

Table 1-11 PBL and Watermark Levels

PBL (Number of transfers)	Tx/Rx Watermark Value
1	greater than or equal to 1
4	greater than or equal to 4
8	greater than or equal to 8
16	greater than or equal to 16
32	greater than or equal to 32
64	greater than or equal to 64
128	greater than or equal to 128
256	greater than or equal to 256

1.6.10 Variable Delay/Clock Generation

Variable delay mechanism for the `cclk_in_drv` is optional, but it can be useful in order to meet a range of hold-time requirements across modes. Variable delay mechanism for the `cclk_in_sample` is mandatory and is required to achieve the correct sampling point for data. `cclk_in/cclk_in_sample/ cclk_in_drv` is generated by Clock Generation Unit (CLKGEN) with variable delay mechanism, which includes Phase Shift Unit and Delay Line Unit selectable. The Phase Shift Unit can shift `cclk_in_sample/cclk_in_drv` by 0/90/180/270-degree relative to `cclk_in`, controlled by `sample_degree/drv_degree`.

The Delay Line Unit can shift `cclk_in_sample/cclk_in_drv` in the unit of 40ps~80ps for every delay element. The delay unit number is determined by `sample_delaynum/drv_delaynum`, and enabled by `sample_sel/drv_sel`.

`cclk_in` is generated by `cclkin` divided by 2. `cclk_in_drv` and `cclk_in_sample` clocks are phase-shifted with delayed versions of `cclk_in`. All clocks are recommended to have a 50% duty cycle; DDR modes must have 50% duty cycles.

The architecture is as follows.

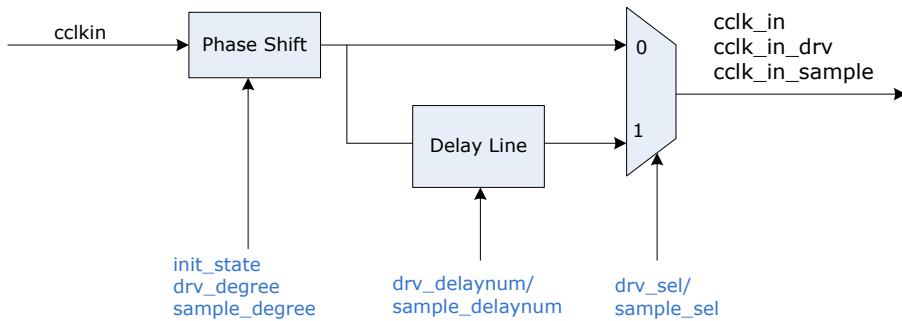


Fig. 1-15 Clock Generation Unit

The control signals for different Host Controller instance are shown as follows:

Table 1-12 Configuration for SDMMC Clock Generation

Signal Name	Source	Default	Description
init_state	CRU_SDMMC_CON0[0]	0	Soft initial state for phase shift.
drv_degree [1:0]	CRU_SDMMC_CON0[2:1]	2	Phase shift for cclk_in_drv. 0: 0-degree 1: 90-degree 2: 180-degree 3: 270-degree
drv_delaynum [7:0]	CRU_SDMMC_CON0[10:3]	0	Element number in delay line for cclk_in_drv
drv_sel	CRU_SDMMC_CON0[11]	0	cclk_in_drv source selection: 0: use clock after phase_shift 1: use clock after phase_shift and delay line
sample_degree [1:0]	CRU_SDMMC_CON1[1:0]	0	Phase shift for cclk_in_sample. 0: 0-degree 1: 90-degree 2: 180-degree 3: 270-degree
sample_delaynum [7:0]	CRU_SDMMC_CON1[9:2]	0	Element number in delay line for cclk_in_sample
sample_sel	CRU_SDMMC_CON1[10]	0	cclk_in_sample source selection: 0: use clock after phase_shift 1: use clock after phase_shift and delay line

Table 1-13 Configuration for SDIO Clock Generation

Signal Name	Source	Default	Description
init_state	CRU_SDIO0_CON0[0]	0	Soft initial state for phase shift.
drv_degree [1:0]	CRU_SDIO0_CON0[2:1]	2	Phase shift for cclk_in_drv. 0: 0-degree 1: 90-degree 2: 180-degree 3: 270-degree
drv_delaynum [7:0]	CRU_SDIO0_CON0[10:3]	0	Element number in delay line for cclk_in_drv
drv_sel	CRU_SDIO0_CON0[11]	0	cclk_in_drv source selection: 0: use clock after phase_shift 1: use clock after phase_shift and delay line
sample_degree [1:0]	CRU_SDIO0_CON1[1:0]	0	Phase shift for cclk_in_sample. 0: 0-degree 1: 90-degree 2: 180-degree 3: 270-degree

Signal Name	Source	Default	Description
sample_delaynum [7:0]	CRU_SDIO0_CON1[9:2]	0	Element number in delay line for cclk_in_sample
sample_sel	CRU_SDIO0_CON1[10]	0	cclk_in_sample source selection: 0: use clock after phase_shift 1: use clock after phase_shift and delay line

Table 1-14 Configuration for EMMC Clock Generation

Signal Name	Source	Default	Description
init_state	CRU_EMMC_CON0[0]	0	Soft initial state for phase shift.
drv_degree [1:0]	CRU_EMMC_CON0[2:1]	2	Phase shift for cclk_in_drv. 0: 0-degree 1: 90-degree 2: 180-degree 3: 270-degree
drv_delaynum [7:0]	CRU_EMMC_CON0[10:3]	0	Element number in delay line for cclk_in_drv
drv_sel	CRU_EMMC_CON0[11]	0	cclk_in_drv source selection: 0: use clock after phase_shift 1: use clock after phase_shift and delay line
sample_degree [1:0]	CRU_EMMC_CON1[1:0]	0	Phase shift for cclk_in_sample. 0: 0-degree 1: 90-degree 2: 180-degree 3: 270-degree
sample_delaynum [7:0]	CRU_EMMC_CON1[9:2]	0	Element number in delay line for cclk_in_sample
sample_sel	CRU_EMMC_CON1[10]	0	cclk_in_sample source selection: 0: use clock after phase_shift 1: use clock after phase_shift and delay line

The following outlines the steps for clock generation sequence:

- 1) Assert init_state to soft reset the CLKGEN.
- 2) Configure drv_degree/sample_degree.
- 3) If fine adjustment required, delay line can be used by configuring drv_delaynum/sample_delaynum and drv_sel/sample_sel.
- 4) Dis-assert init_state to start CLKGEN.

1.6.11 Variable Delay Tuning

Tuning is defined by SD and MMC cards to determine the correct sampling point required for the host, especially for the speed modes SDR104 and HS200 where the output delays from the cards can be up to 2 UI. Tuning is required for other speed modes-such as DDR50-even though the output delay from the card is less than one cycle.

Command for tuning is different for different cards.

- SD Memory Card:
 - CMD19 – SD card for SDR50 and SDR104 speed modes. Tuning data is defined by card specifications.
 - CMD6 – SD card for speed modes not supporting CMD19. Tuning data is the 64byte SD status.
- Multimedia Card:
 - CMD21 – MMC card for HS200 speed mode. Tuning data is defined by card specifications.

- CMD8 – MMC card for speed modes not supporting CMD21. Tuning data is 512 byte ExtCSD data.

The following is the procedure for variable delay tuning:

- 1) Set a phase shift of 0-degree on cclk_in_sample.
- 2) Send the Tuning command to the card; the card in turn sends an R1 response on the CMD line and tuning data on the DAT line.
- 3) If the host sees any of the errors—start bit error, data crc error, end bit error, data read time-out, response crc error, response error—then the sampling point is incorrect.
- 4) Send CMD12 to bring the host controller state machines to idle.
- The card may treat CMD12 as an invalid command because the card has successfully sent the tuning data, and it cannot send a response.
- The host controller may generate a response time-out interrupt that must be cleared by software.
- 5) Repeat steps 2) to 4) by increasing the phase shift value or delay element number on cclk_in_sample until the correct sampling point is received such that the host does not see any of the errors.
- 6) Mark this phase shift value as the starting point of the sampling window.
- 7) Repeat steps 2 to 4 by increasing the phase shift value or delay element number on cclk_in_sample until the host sees the errors starting to come again or the phase shift value reaches 360-degree.
- 8) Mark the last successful phase shift value as the ending point of the sampling window.

A window is established where the tuning block is matched. For example, for a scenario where the tuning block is received correctly for a phase shift window of 90-degree and 180-degree, then an appropriate sampling point is established as 135-degree. Once a sampling point is established, no errors should be visible in the tuning block.

1.6.12 Package Command

In order to reduce overhead, read and write commands can be packed in groups of commands—either all read or all write—that transfer the data for all commands in the group in one transfer on the bus.

Packed commands can be of two types:

- Packed Write: CMD23 → CMD25
- Packed Read: CMD23 → CMD25 → CMD23 → CMD18

Packed commands are put in packets by the application software and are transparent to the core. For more information on packed commands, refer to the eMMC specification.

1.6.13 Card Detection Method

There are many methods for SDMMC/SDIO card detection.

- Method1: Using CDETECT register, which is value on card_detect_n input port. 0 represents presence of card.
- Method2: Using card detection unit, outputting host interrupt (IRQ_ID[46]). The card detection unit looks for any changes in the card-detect signals for card insertion or card removal. It filters out the debounces associated with mechanical insertion or removal, and generates one interrupt to the host. You can program the debounce filter value in DEBNCE[23:0]. Following figure illustrates the timing for card-detect signals.

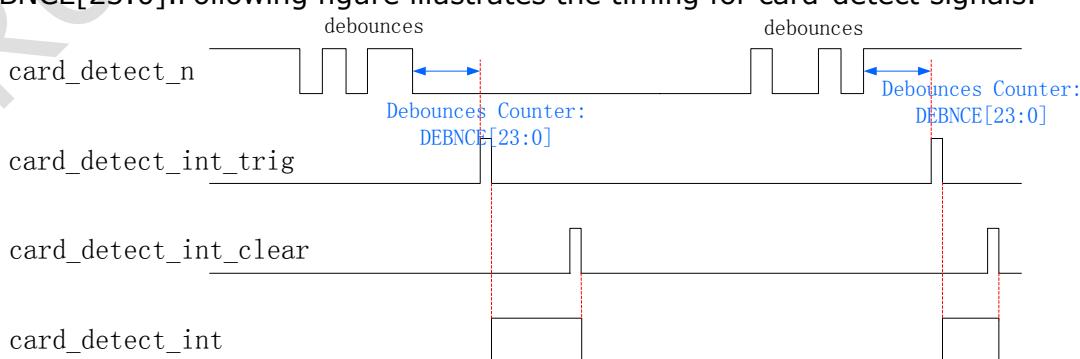


Fig. 1-16 Card Detection Method 2

- Method3: Using card detection unit in GRF, outputting sdmmc_detect_dual

edge_int(IRQ_ID[86]), only available for SDMMC. Similar to Method2, except that the debounce is selecting from 5ms/15ms/35ms/50ms; and the insertion/removal detection interrupt can be enabled or cleared respectively. The detailed register information is:

Table 1-15 Register for SDMMC Card Detection Method 3

Signal Name	Source	Default	Description
sd_detectn_rise_edge_irq_en	GRF_SOC_CON0[2]	0	sdmmc detect_n signal rise edge interrupt enable. 1'b1: enable 1'b0: disable
sd_detectn_rise_edge_irq_pd	GRF_SOC_CON0[0]	0	sdmmc detect_n rise edge interrupt pending status. Write 1 to clear the status.
sd_detectn_fall_edge_irq_en	GRF_SOC_CON0[3]	0	sdmmc detect_n signal fall edge interrupt enable. 1'b1: enable 1'b0: disable
sd_detectn_fall_edge_irq_pd	GRF_SOC_CON0[1]	0	sdmmc detect_n fall edge interrupt pending status. Write 1 to clear the status.
grf_filter_cnt_sel	GRF_SOC_CON0[5:4]	0	the counter select for sd card detect filter: 2'b00: 5ms 2'b01: 15ms 2'b10: 35ms 2'b11: 50ms

- Method4: Using card_detect_n for interrupt source, connecting to IRQ_ID[78] directly.

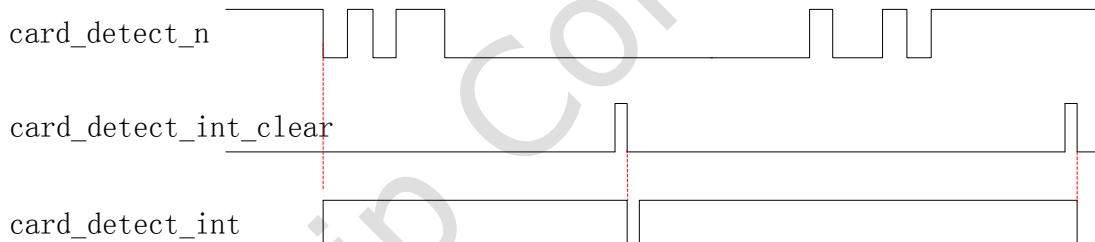


Fig. 1-17 Card Detection Method 4

1.6.14 SDMMC IOMUX With JTAG

The IO for sdmmc_cdata2/sdmmc_cdata3 is shared with jtag_tck/jtag_tms. The condition of usage for SDMMC or JTAG usage is as follows.

- If GRF_SOC_CON0[11](grf_force_jtag) is equal to 1 and sdmmc card is not detected within detection time(defined in GRF_SDMMC_DET_CNT, in the unit of XIN24M clock), the GPIOs are used for JTAG.
- Otherwise, the GPIOs' usage is defined by IOMUX configuration.

Chapter 2 USB OTG 2.0

2.1 Overview

USB OTG 2.0 is a Dual-Role Device controller, which supports both device and host functions and is fully compliant with OTG Supplement to USB2.0 specification, and support high-speed (480Mbps), full-speed (12Mbps), low-speed (1.5Mbps) transfer.

USB OTG 2.0 is optimized for portable electronic devices, point-to-point applications (no hub, direct connection to device) and multi-point applications to devices.

2.1.1 Features

- Compliant with the OTG Supplement to the USB2.0 Specification
- Operates in High-Speed and Full-Speed mode
- Support 9 channels in host mode
- 9 Device mode endpoints in addition to control endpoint 0, 4 in, 3 out and 2 IN/OUT
- Built-in one 1024x35 bits FIFO
- Internal DMA with scatter/gather function
- Supports packet-based, dynamic FIFO memory allocation for endpoints for flexible, efficient use of RAM
- Support dynamic FIFO sizing

2.2 Block Diagram

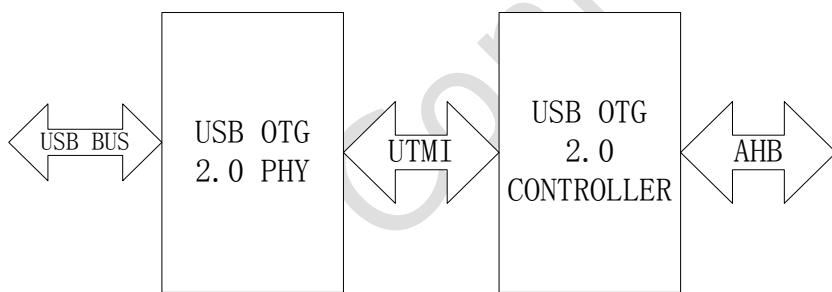


Fig. 2-1 USB OTG 2.0 Architecture

The Fig shows the architecture of USB OTG 2.0. It is broken up into two separate units: USB OTG 2.0 controller and USB OTG 2.0 PHY. The two units are interconnected with UTMI interface.

2.2.1 USB OTG 2.0 Controller Function

The USB OTG 2.0 Controller controls SIE (Serial Interface Engine) logic, the endpoint logic, the channel logic and the internal DMA logic.

The SIE logic contains the USB PID and address recognition logic, and other sequencing and state machine logic to handle USB packets and transactions. Generally the SIE Logic is required for any USB implementation while the number and types of endpoints will vary as function of application and performance requirements.

The endpoint logic contains the endpoint specific logic: endpoint number recognition, FIFOs and FIFO control, etc.

The channel Logic contains the channel tasks schedule, FIFOs and FIFO control, etc.

The internal DMA logic controls data transaction between system memory and USB FIFOs.

2.2.2 USB OTG 2.0 PHY Function

The USB OTG 2.0 PHY handles the low level USB protocol and signaling. This includes features such as data serialization and deserialization, bit stuffing and clock recovery and synchronization. The primary focus of this block is to shift the clock domain of the data from the USB 2.0 rate to the frequency of UTMI clock which is 30MHz.

2.2.3 UTMI Interface

● Transmit

Transmit must be asserted to enable any transmissions.

The USB OTG2.0 CONTROLLER asserts TXValid to begin a transmission and negates TXValid to end a transmission. After the USB OTG2.0 CONTROLLER asserts TXValid it can assume that the transmission has started when it detects TXReady asserted.

The USB OTG2.0 CONTROLLER assumes that the USB OTG2.0 PHY has consumed a data byte if TXReady and TXValid are asserted.

The USB OTG2.0 CONTROLLER must have valid packet information (PID) asserted on the Data In bus coincident with the assertion of TXValid. Depending on the USB OTG2.0 PHY implementation, TXReady may be asserted by the Transmit State Machine as soon as one CLK after the assertion of TXValid. TXValid and TXReady are sampled on the rising edge of CLK. The Transmit State Machine does NOT automatically generate Packet ID's (PIDs) or CRC. When transmitting, the USB OTG2.0 CONTROLLER is always expected to present a PID as the first byte of the data stream and if appropriate, CRC as the last bytes of the data stream. The USB OTG2.0 CONTROLLER must use LineState to verify a Bus Idle condition before asserting TXValid in the TX Wait state.

The state of TXReady in the TX Wait and Send SYNC states is undefined. An MTU implementation may prepare for the next transmission immediately after the Send EOP state and assert TXReady in the TX Wait state. An MTU implementation may also assert TXReady in the Send SYNC state. The first assertion of TXReady is Macrocell implementation dependent. The USB OTG2.0 CONTROLLER must prepare DataIn for the first byte to be transmitted before asserting TXValid.

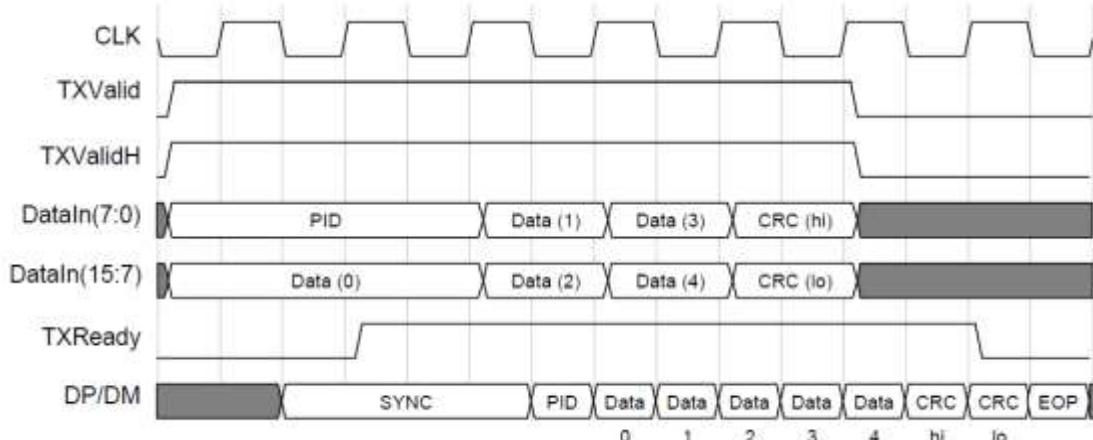


Fig. 2-2 UTMI interface – Transmit timing for a data packe

● Receive

RXActive and RXValid are sampled on the rising edge of CLK.

In the RX Wait state the receiver is always looking for SYNC.

The USB OTG 2.0 PHY asserts RXActive when SYNC is detected (Strip SYNC state).

The USB OTG 2.0 PHY negates RXActive when an EOP is detected (Strip EOP state).

When RxActive is asserted, RXValid will be asserted if the RX Holding Register is full.

RXValid will be negated if the RX Holding Register was not loaded during the previous byte time.

This will occur if 8 stuffed bits have been accumulated.

The USB OTG2.0 Controller must be ready to consume a data byte if RXActive and RXValid are asserted (RX Data state).

In FS mode, if a bit stuff error is detected then the Receive State Machine will negate RXActive and RXValid, and return to the RX Wait state.

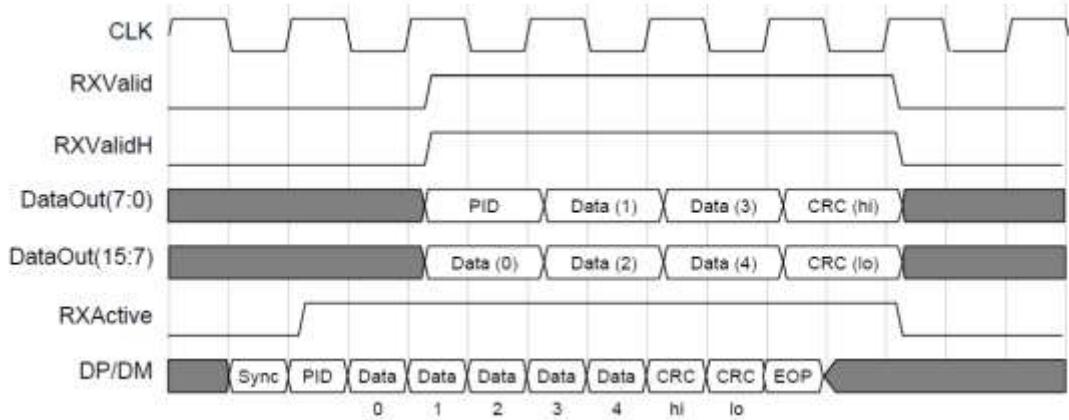


Fig. 2-3 UTMI interface – Receive timing for a data packet

2.3 USB OTG2.0 Controller

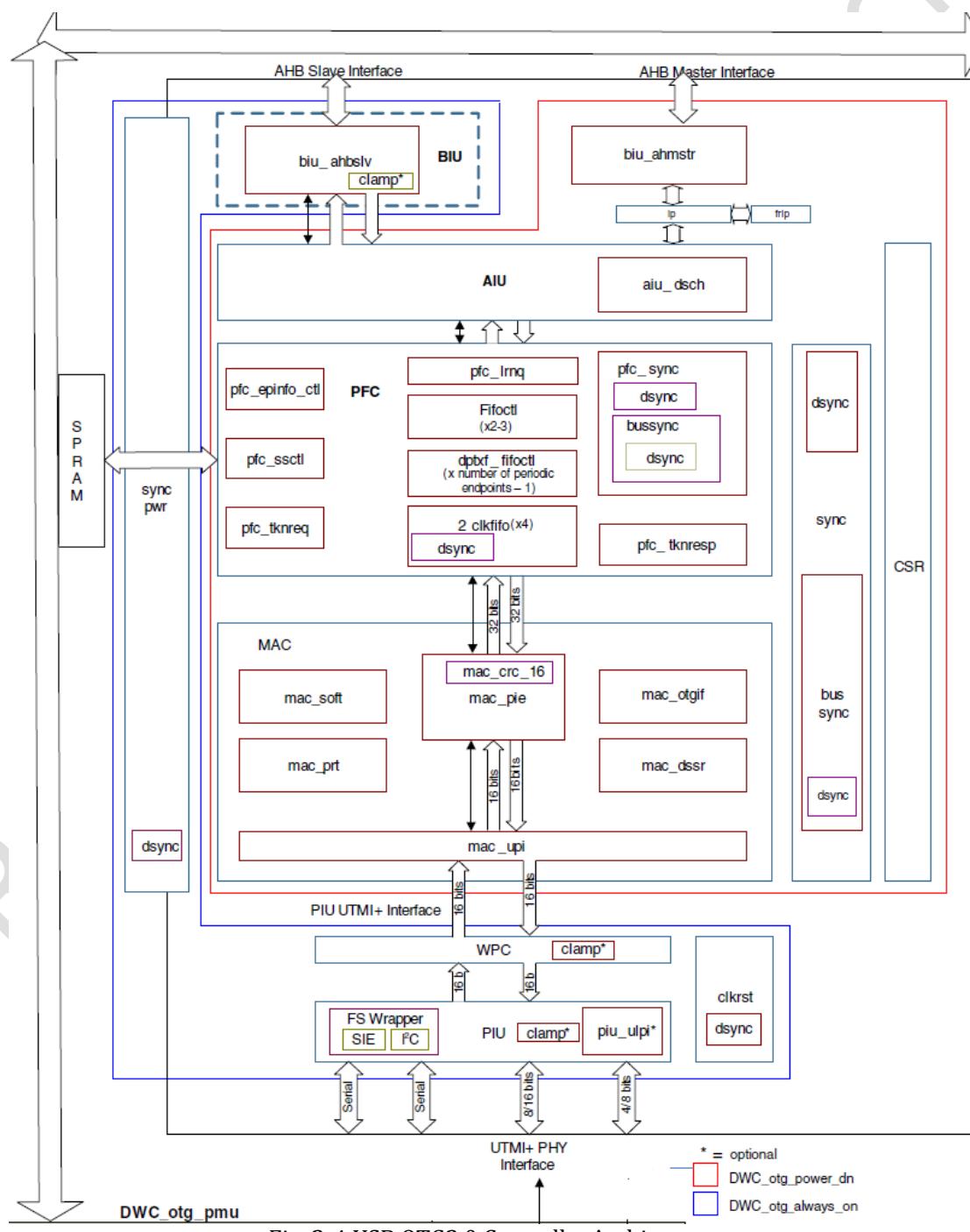


Fig. 2-4 USB OTG2.0 Controller Architecture

1). AHB Slave Bus Interface Unit (BIUS)

The AHB Slave interface unit converts AHB cycles to CSR write/read, Data-FIFO read/write, and DFIFO push/pop signals.

2) Control and Status Registers (CSR)

The CSR block resides in the AHB clock domain, and contains all registers except the Power and Clock Gating Control Register (PCGCCTL) and bits 31:29 of the Core Interrupt register (GINTSTS).

3) Application Interface Unit (AIU)

The application Interface Unit (AIU) consists of the following interfaces:

AHB Master

AHB Slave

Packet FIFO Controller

Control and Status registers

4). DMA Scheduler (DSCH)

This block is used only in DMA mode. It controls the transfer of data packets between the system memory and the USB OTG 2.0 Controller for both Internal and External DMA.

5). Packet FIFO Controller (PFC)

Several FIFOs are used in Device and Host modes to store data inside the core before transmitting it on either the AHB or the USB. PFC connect the Data FIFO interface to an industry-standard, single-port synchronous SRAM. Address, write data, and control outputs are driven late by the USB OTG 2.0 Controller, but in time to meet the SRAM setup requirements. Input read data is expected late from the SRAM and registered inside the core before being used.

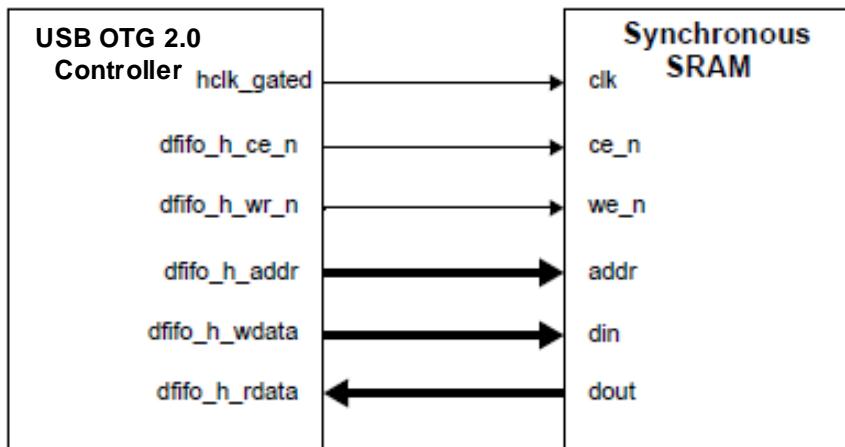


Fig. 2-5 DFIFO single-port synchronous SRAM interface

6).Media Access Controller (MAC)

The Media Access Controller (MAC) module handles USB transactions, and device, host, and OTG protocols.

7) PHY Interface Unit (PIU)

The core uses 16-bit UTMI+ Interface.

8) Wakeup and Power Controller (WPC)

When the USB is suspended or the session is not valid, the PHY is driven into Suspend mode and the PHY clock is stopped to reduce PHY and the core power consumption. To reduce power consumption further, the core also supports AHB clock gating .

2.3.1 Host Architecture

The host uses one transmit FIFO for all non-periodic OUT transactions and one transmit FIFO for all periodic OUT transactions. These transmit FIFOs are used as transmit buffers to hold the data (payload of the transmit packet) to be transmitted over USB.

The host pipes the USB transactions through Request queues (one for periodic and one for non-periodic). Each entry in the Request - queue holds the IN or OUT channel number along with other information to perform a transaction on the USB. The order in which the requests

are written into the queue determines the sequence of transactions on the USB. The host processes the periodic Request queue first, followed by the non-periodic Request queue, at the beginning of each (micro) frame.

The host uses one Receive-FIFO for all periodic and non-periodic transactions. The FIFO is used as a Receive-buffer to hold the received data (payload of the received packet) from the USB until it is transferred to the system memory. The status of each packet received also goes into the FIFO. The status entry holds the IN channel number along with other information, such as received byte count and validity status, to perform a transaction on the AHB.

2.3.2 Device Architecture

The core uses Dedicated Transmit FIFO Operation. In this mode, there are individual transmit FIFOs for each IN endpoint.

The OTG device uses a single receive FIFO to receive the data for all the OUT endpoints. The receive FIFO holds the status of the received data packet, such as byte count, data PID and the validity of the received data. The DMA or the application reads the data out of the receive FIFO as it is received.

2.3.3 FIFO Mapping

- FIFO mapping in Host mode.

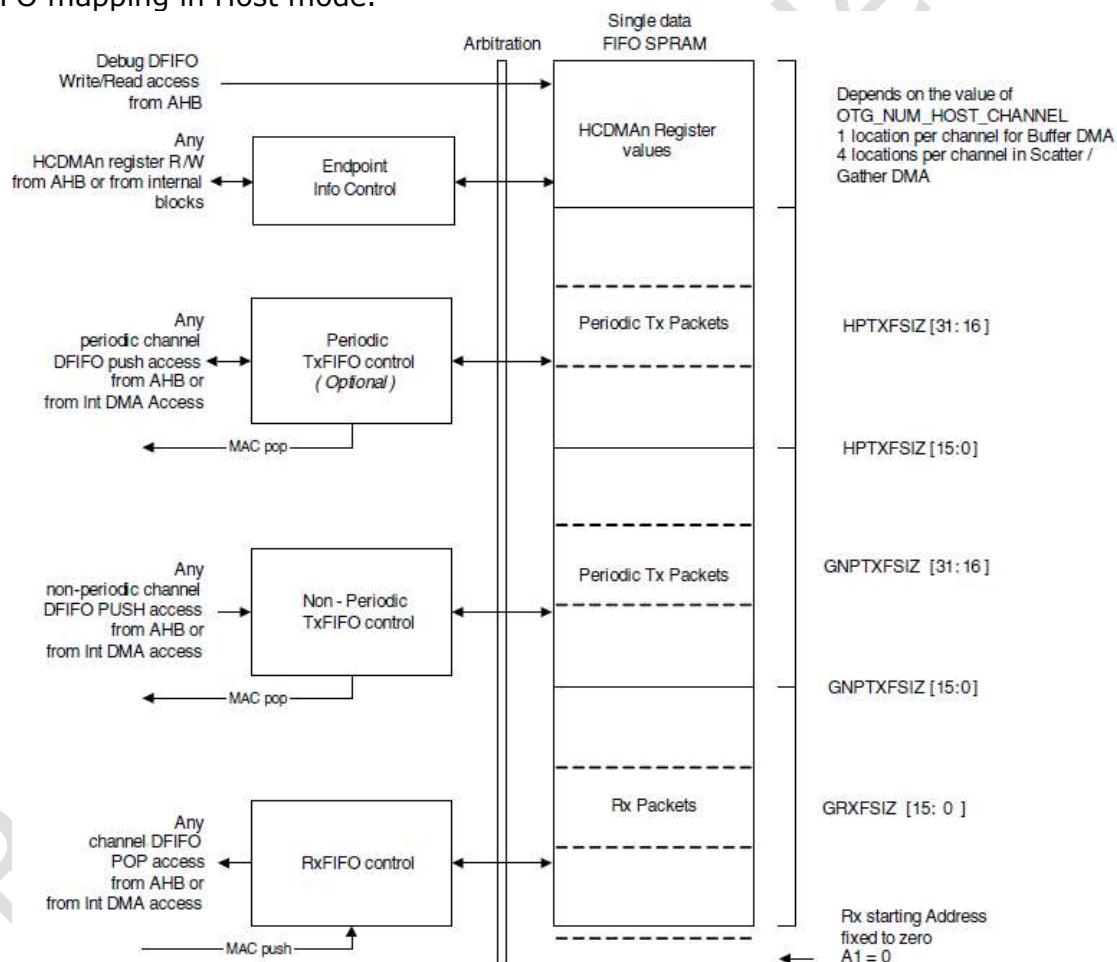


Fig. 2-6 USB OTG 2.0 Controller host mode FIFO address mapping

Note: When the device is operating in Internal DMA mode, the last locations of the SPRAM are used to store the DMAADDR values for each channel.

- FIFO mapping in Device mode.

When the device is operating in non-Descriptor Internal DMA mode, the last locations of the SPRAM are used to store the DMAADDR values for each channel. When the device is operating in Descriptor mode, then the last locations of the SPRAM store the Base Descriptor address, Current Descriptor address, Current Buffer address, and status quad let information for each

endpoint direction.

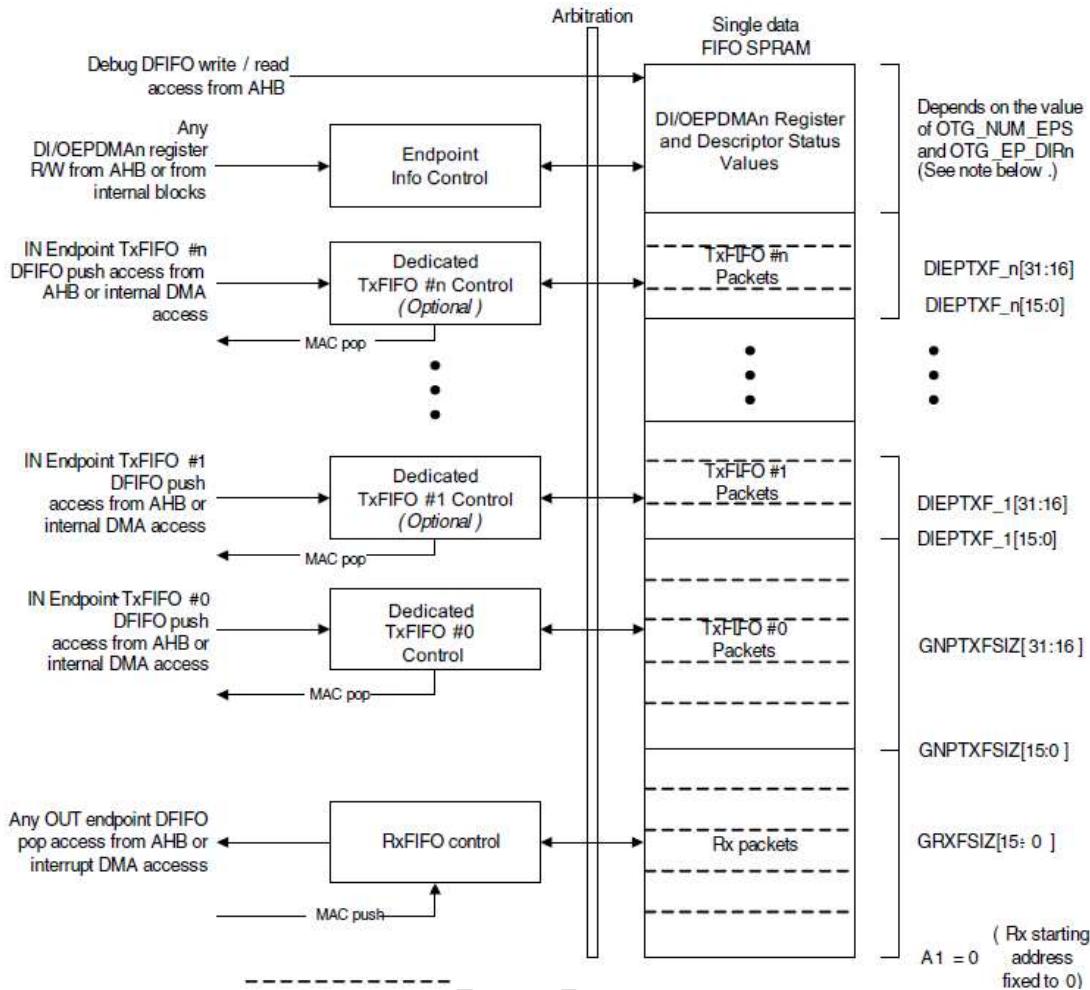


Fig. 2-7 USB OTG 2.0 Controller device mode FIFO address mapping

Note: When the device is operating in non-Scatter Gather Internal DMA mode, the last locations of the SPRAM are used to store the DMAADDR values for each Endpoint (1 location per endpoint). When the device is operating in Scatter Gather mode, then the last locations of the SPRAM store the Base Descriptor address, Current Descriptor address, Current Buffer address, and status quadlet information for each endpoint direction (4 locations per Endpoint). If an Endpoint is bidirectional, then 4 locations will be used for IN, and another 4 for OUT).

2.4 USB OTG2.0 PHY

USB PHY supports dual OTG ports' functions and is fully compliant with USB2.0 specification, and support High-speed (480Mbps), full-speed (12Mbps), low-speed (1.5Mbps) transfer. It provides a complete on-chip transceiver physical solution with ESD protection. A minimum number of external components are needed, which include a 45 ohm resistor for resistance calibration purpose. Its feature contains:

- provide dual UTMI ports
- OTG0 Support UART Bypass Function
- Fully compliant with USB specifications Rev 2.0, 1.1 HOST/Device and OTG V1.2.
- Supports 480Mbps (HS), 12Mbps (FS) & 1.5Mbps(LS) serial data transmission
- Supports low latency hub mode with 40 bit time round trip delay
- 8 bit or 16 bit UTMI interface compliant with UTMI+ specification level 3 Rev 1.
- Loop back BIST mode supported
- Built-in I/O and ESD structure
- On-die self-calibrated HS/FS/LS termination
- 12MHz crystal oscillator with integrated phase-locked loop (PLL) oscillator
- Dual 3.3V / 1.2V supply

2.4.1 Block Diagram

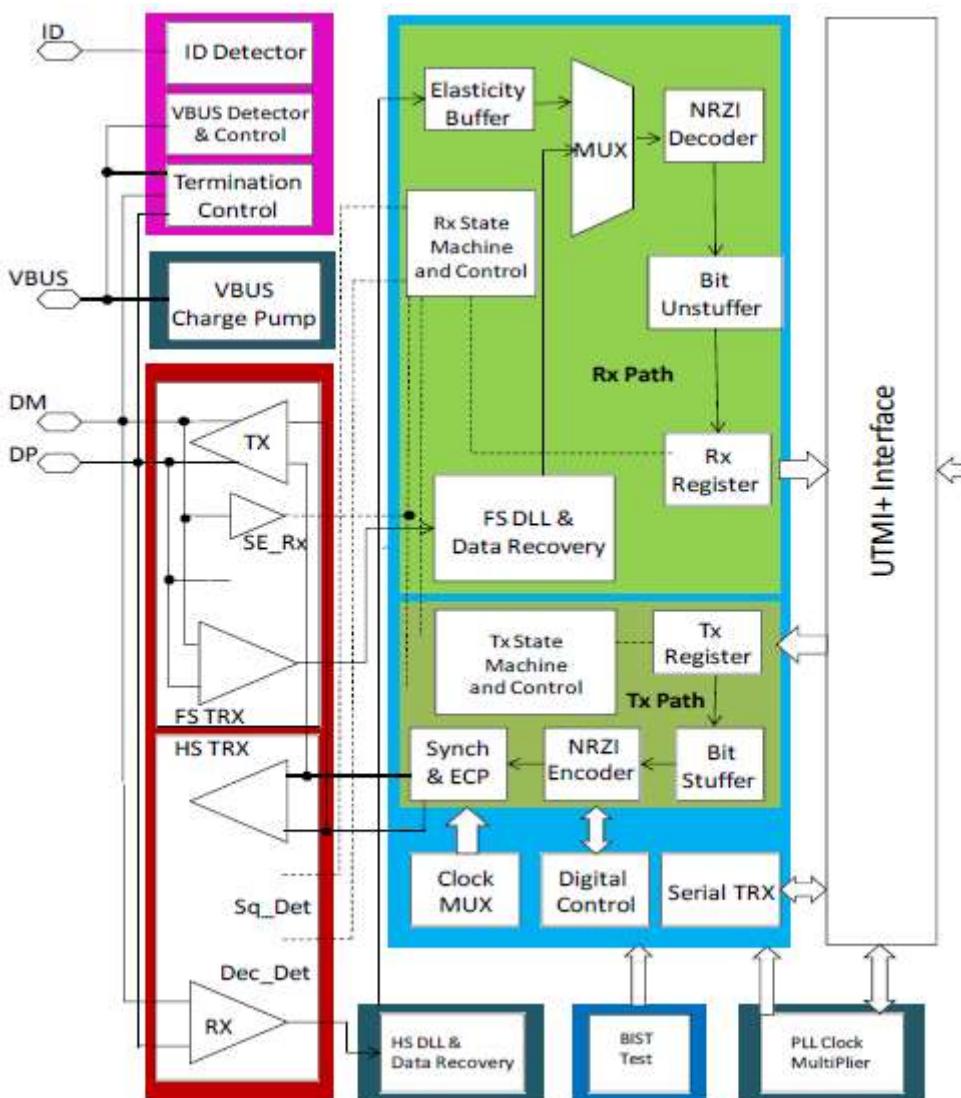


Fig. 2-8 usb phy architecture

HS AFE

The HS AFE contains the low-level analog circuitry, and also the HS differential data transmitter and receiver, to perform HS transmission envelope detection and host disconnection detection. It works in HS mode only.

HS Transmit driver

The HS transmit driver is active only when transmit is asserted. In HS transceiver enabled mode and the transmit state machine has data to send, the XCVR selects input. Data from transmit data path will be driven onto the DP/DM signal lines when enabled.

HS Differential Receiver

When enabled, received HS data will be multiplexed through the receive data path to the receive shift and hold registers. It is active only in HS mode.

transmission envelope detector (Squelch detector)

When the amplitude of the differential signal at a receiver's inputs falls below the squelch threshold, the envelope detector will indicate the invalid data. It must indicate squelch when the signal drops below 100mV differential amplitude, and also, it must indicate that the line is not in the squelch state when the signal exceeds 150mV differential amplitude.

Disconnection envelope detector

In host mode, this envelope detector is active to detect the high speed disconnect state on the line. Disconnection must be indicated when the amplitude of the differential signal at the

downstream facing driver's connector is more than 625 mV, and it must not be indicated when the signal amplitude is less than 525 mV.

FS/LS AFE

In FS or LS mode, the FS/LS AFE is active to send and receive the FS or LS data on the USB bus. Also it supports the reset, suspend and resume detection through the data line single ended receivers.

FS/LS Transmitter

The FS/LHS transmitter is active only when transmit is asserted. In FS or LS transceiver enabled mode and the transmit state machine has data to send, the XCVR selects input. Data from transmit data path will be driven onto the DP/DM signal lines when enabled.

FS/LS Differential Receiver

When enabled, received FS or LS data will be multiplexed through the receive data path to the receive shift and hold registers. It is active only in FS or LS modes.

Single ended receivers

The single ended receivers are used for low-speed and full-speed signaling detection.

Digital Core TX Path

The digital core TX path has some blocks responsible for SYNC and EOP generation, data encoding, bit stuffing and data serialization. And meanwhile, also a TX state machine is involved to manage the communication with the controller.

TX Shift/Hold Register

The TX shift/Hold register module consists of an 8-bit primary shift register for parallel/serial conversion and 8-bit hold register used to buffer the next data to serialize. This module is responsible for reading parallel data from the parallel application bus interface upon command and serializing for transmission over USB.

Bit stuffer

To ensure adequate signal transitions, when sending a packet on USB, a bit stuffer is employed by the transmitter. A '0' has to be inserted after every six consecutive ones in the data stream before the data is NRZI encoded, to force a transition in the NRZI data stream.

NRZI Encoder

The High speed, Full speed or low speed serial transmitted data are encoded by The NRZI encoder. As a state transition, a '0' is encoded, and as no state transition, a '1' is encoded.

Transmit state machine

The communication between the controller and the PHY in TX path is controlled by the transmit state machine, which synchronizes the Data with the Sync and the EOP, and also supports the LS, FS and HS Modes.

Digital Core RX Path

The digital core RX path includes blocks responsible for SYNC and EOP detection and stripping, data decoding, bit un-stuffing and data de-serialization. Also a RX state machine is involved to manage the communication with the controller. FS/LS data and clock is recovered in this section.

Elasticity buffer

To compensate for differences between transmitting and receiving clocks, the Elasticity Buffer is used to synchronize the HS extracted data with the PLL internal clock.

Mux

The Mux block allows the data from the HS or FS/LS receivers to be routed to the shared receive logic. The state of the Mux is determined by the Xcvr Select input.

NRZI Decoder

The NRZI is responsible for decoding the High speed or Full speed received NRZI encoded data. A change in level is decoded as '0' and no change in level is decoded as '1'.

Bit Un-stuffer

The Bit Un-stuffer not only recognizes the stuffed bits from the data stream, but also discards them. Also it detects bit stuff error, which is interpreted as HS EOP.

RX Shift/Hold Register

This module de-serializes received data and transmits 8-bit parallel data to the application bus

interface. It consists of an 8-bit primary shift register for serial to parallel conversion and an 8-bit hold register for buffering the last de-serialized data byte.

Receiver state machine

The receiver state machine controls the communication between the controller and the PHY in the RX path, strips the SYNC and the EOP from the Data and supports the LS, FS and HS Modes.

PLL Clock Multiplier

This module is composed of the off-chip crystal and the on-chip clock multiplier. It generates the appropriate internal clocks for the UTM and the CLK output signal. All data transfer signals are synchronized with the CLK signal.

External Crystal

The external crystal is composed of a precise resonance frequency crystal and a crystal oscillator. It is optional to have this crystal oscillator integrated on-chip or have it off-chip. This crystal/crystal oscillator provides a very precise clock of 12 MHz with deviation of ± 100 ppm. The oscillator is not a part of the PHY, but external.

Clock Multiplier

The UTM interface is described as an un-directional/bi-directional 8-bit/16-bit parallel interface and the CLK signal is a 60/30 MHz signal. All data transfer signals should be synchronized with the CLK signal. CLK usable signal is internally implemented which blocks any transitions of CLK until it is usable. Meanwhile, the clock multiplier provides another three clocks in addition to the CLK signal. That is a 480 MHz and 7.5 MHz clock signals.

Clock MUX

The Clock Multiplexer supplies both the transmitter and receiver paths with the adequate bit clock depending on the XcvrSelect signal and to ensure smooth clock switching. It also includes clock gating and power-down features.

Control Logic Block

This block is responsible for controlling, enabling and disabling the different blocks in the system.

OTG Circuitry (optional)

With the OTG circuitry, the system has the capability to dynamically switch between host and peripheral, enable dual role device behavior and point-to-point communication. The OTG circuitry functions as VBUS generation and detection. Both ID detection and terminations control are implemented in it.

ID Detector (optional)

To provide the ID signal that is used to indicate the state of the ID pin on the USB mini receptacle. This pin makes it able to determine which kind of plug is connected and to confirm if the device default state is A device or B device.

VBU S Detector and termination control

The VBUS detector is a set of comparators, functions to monitor and sense the voltage on USB bus power line. For VBUS signaling and discharging, VBUS pull up and pull-down resistors are also implemented.

Automatic Test Functions

Loop-back test to address all IP components.

In loop-back test mode, all transmitted data packets are received back in an internal loop to check IP functional integrity. There are some digital components that cannot be tested with the scan technique due to the high-speed nature of the digital part. To be regarded as a good idea, Loop-back allows testing full design paths at speed. It should complement the testing suite for digital core to achieve the highest coverage possible. According to the UTMI specification Section 5.18, version 1.05 Page 34, the 8 bits un-directional data bus can be implemented as 8 bits bi-directional one. This implementation will hinder the loop-back test functionality.

2.5 UART BYPASS FUNCITON

When in UART bypass mode, UART2 is connect to USB interface; Otherwise, UART2 use normal UART interface.

Signal	CONNECT	I/O	Description
BYPASSDMDATA0	uart2_sout	I	Data for DM0 Transmitter Digital Bypass
BYPASSDMEN0	grf_uoc1_con4[11]	I	DM0 Transmitter Digital Bypass Enable
BYPASSEL0	grf_uoc1_con4[13]	I	Transmitter Digital Bypass mode Enable
FSVPLUS0	uart2_sin	O	Single-Ended D- Indicator The controller signal indicates the state of the DP during normal operation or UART data reception
OTGDISABLE0	grf_uoc1_con4[10]	I	1'b1: OTG0 disable; 1'b0: OTG0 normal mode
COMMONONNN	grf_uoc0_con5[11]	I	Common Block Power-Down Control This signal controls the power-down signals in PLL blocks when the USB PHY is in Suspend Mode. 1: PLL blocks are powered down. 0: PLL blocks remain powered This signal is a strapping option that must be set prior to a power-on reset and remain static during normal operation.

Note: USB OTG2.0 PHY support UART Bypass Function.

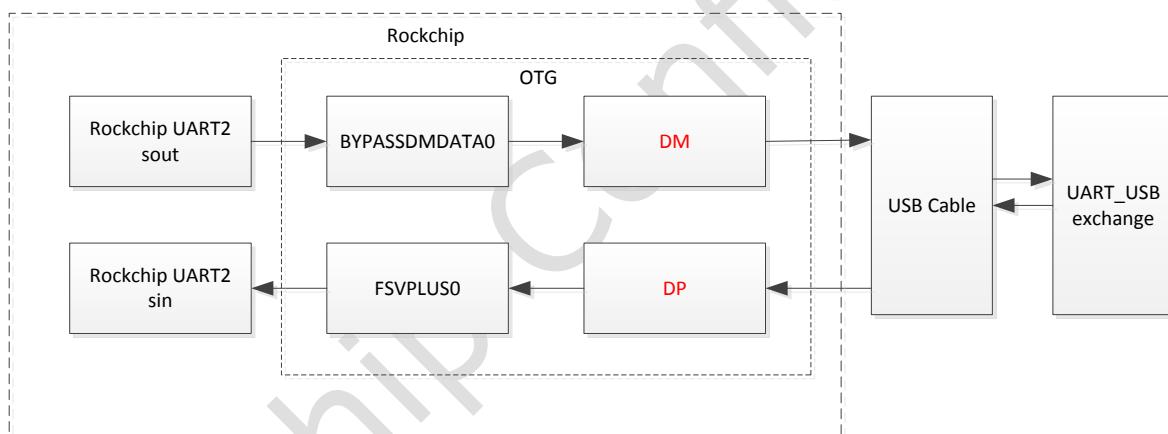


Fig. 2-9 UART Application

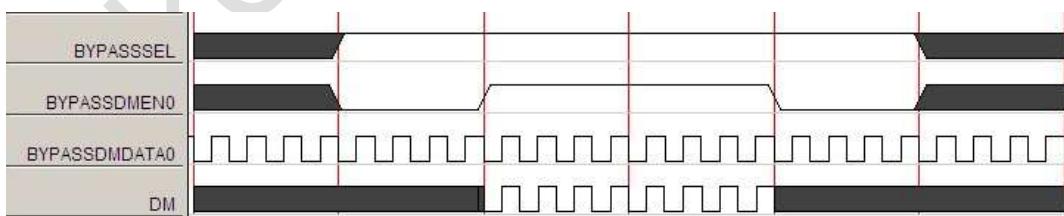


Fig. 2-10 UART Timing Sequence

To use UART and Auto resume functions:

1. Disable the OTG block by setting OTGDISABLE0 to 1'b1.
 2. Disable the pull-up resistance on the DP line by setting OPMODE0[1:0] to 2'b01.
 3. To ensure that the XO, Bias, and PLL blocks are powered down in Suspend mode, set COMMONONNN to 1'b1.
 4. Place the USB PHY in Suspend mode by setting SUSPENDDM0 to 1'b0.
 5. Set BYPASSEL0 to 1'b1.
 6. To transmit data, controls BYPASSDMEN0, and BYPASSDMDATA0.
- To receive data, monitor FSVPLUS0.

To return to normal operating mode:

1. Ensure that there is no activity on the USB.
2. Set BYPASSSEL0 to 1'b0.
3. Set SUSPENDM0 to 1'b1. Resume the USB PHY.
4. Set COMMONONN to 1'b0.
5. set OTGDISABLE0 to 1'b0.

2.6 Register Description

2.6.1 Register Summary

Name	Offset	Size	Reset Value	Description
USBOTG_GOTGCTL	0x0000	W	0x00000000	Control and Status Register
USBOTG_GOTGINT	0x0004	W	0x00000000	Interrupt Register
USBOTG_GAHBCFG	0x0008	W	0x00000000	AHB Configuration Register
USBOTG_GUSBCFG	0x000c	W	0x00001400	USB Configuration Register
USBOTG_GRSTCTL	0x0010	W	0x80000000	Reset Register
USBOTG_GINTSTS	0x0014	W	0x00000000	Interrupt Register
USBOTG_GINTMSK	0x0018	W	0x00000000	Interrupt Mask Register
USBOTG_GRXSTSR	0x001c	W	0x00000000	Receive Status Debug Read Register
USBOTG_GRXSTSP	0x0020	W	0x00000000	Receive Status Read and Pop Register
USBOTG_GRXFSIZ	0x0024	W	0x00000000	Receive FIFO Size Register
USBOTG_GNPTXFSIZ	0x0028	W	0x00000000	Non-Periodic Transmit FIFO Size Register
USBOTG_GNPTXSTS	0x002c	W	0x00000000	Non-Periodic Transmit FIFO/Queue Status Register
USBOTG_GI2CCTL	0x0030	W	0x11000000	I2C Address Register
USBOTG_GPVNDCTL	0x0034	W	0x00000000	PHY Vendor Control Register
USBOTG_GGPIO	0x0038	W	0x00000000	General Purpost Input/Output Register
USBOTG_GUID	0x003c	W	0x00000000	User ID Register
USBOTG_GSNPSID	0x0040	W	0x00004f54	Core ID Register
USBOTG_GHWCFG1	0x0044	W	0x00000000	User HW Config1 Register
USBOTG_GHWCFG2	0x0048	W	0x00000000	User HW Config2 Register
USBOTG_GHWCFG3	0x004c	W	0x00000000	User HW Config3 Register
USBOTG_GHWCFG4	0x0050	W	0x00000000	User HW Config4 Register
USBOTG_GLPMCFG	0x0054	W	0x00000000	Core LPM Configuration Register
USBOTG_GPWRDN	0x0058	W	0x00000000	Global Power Down Register
USBOTG_GDFIFO CFG	0x005c	W	0x00000000	Global DFIFO Software Config Register
USBOTG_GADPCTL	0x0060	W	0x00000000	ADP Timer,Control and Status Register
USBOTG_HPTXFSIZ	0x0100	W	0x00000000	Host Periodic Transmit FIFO Size Register
USBOTG_DIEPTXFn	0x0104	W	0x00000000	Device Periodic Transmit FIFO-n Size Register
USBOTG_HCFG	0x0400	W	0x00000000	Host Configuration Register

Name	Offset	Size	Reset Value	Description
USBOTG_HFIR	0x0404	W	0x00000000	Host Frame Interval Register
USBOTG_HFNUM	0x0408	W	0x0000ffff	Host Frame Number/Frame Time Remaining Register
USBOTG_HPTXSTS	0x0410	W	0x00000000	Host Periodic Transmit FIFO/Queue Status Register
USBOTG_HAINT	0x0414	W	0x00000000	Host All Channels Interrupt Register
USBOTG_HAINTMSK	0x0418	W	0x00000000	Host All Channels Interrupt Mask Register
USBOTG_HPRT	0x0440	W	0x00000000	Host Port Control and Status Register
USBOTG_HCCHARn	0x0500	W	0x00000000	Host Channel-n Characteristics Register
USBOTG_HCSPLTn	0x0504	W	0x00000000	Host Channel-n Split Control Register
USBOTG_HCINTn	0x0508	W	0x00000000	Host Channel-n Interrupt Register
USBOTG_HCINTMSKn	0x050c	W	0x00000000	Host Channel-n Interrupt Mask Register
USBOTG_HCTSIZn	0x0510	W	0x00000000	Host Channel-n Transfer Size Register
USBOTG_HCDMAAn	0x0514	W	0x00000000	Host Channel-n DMA Address Register
USBOTG_HCDMABn	0x051c	W	0x00000000	Host Channel-n DMA Buffer Address Register
USBOTG_DCFG	0x0800	W	0x08200000	Device Configuration Register
USBOTG_DCTL	0x0804	W	0x00002000	Device Control Register
USBOTG_DSTS	0x0808	W	0x00000000	Device Status Register
USBOTG_DIEPMSK	0x0810	W	0x00000000	Device IN Endpoint common interrupt mask register
USBOTG_DOEPMSK	0x0814	W	0x00000000	Device OUT Endpoint common interrupt mask register
USBOTG_DAINT	0x0818	W	0x00000000	Device All Endpoints interrupt register
USBOTG_DAINTMSK	0x081c	W	0x00000000	Device All Endpoint interrupt mask register
USBOTG_DTKNQR1	0x0820	W	0x00000000	Device IN token sequence learning queue read register1
USBOTG_DTKNQR2	0x0824	W	0x00000000	Device IN token sequence learning queue read register2
USBOTG_DVBUUSDIS	0x0828	W	0x00000b8f	Device VBUS discharge time register
USBOTG_DVBUSPULSE	0x082c	W	0x00000000	Device VBUS Pulsing Timer Register
USBOTG_DTHRCTL	0x0830	W	0x08100020	Device Threshold Control Register

Name	Offset	Size	Reset Value	Description
USBOTG_DIEPEMPMSK	0x0834	W	0x00000000	Device IN endpoint FIFO empty interrupt mask register
USBOTG_DEACHINT	0x0838	W	0x00000000	Device each endpoint interrupt register
USBOTG_DEACHINTMSK	0x083c	W	0x00000000	Device each endpoint interrupt register mask
USBOTG_DIEPEACHMSKn	0x0840	W	0x00000000	Device each IN endpoint -n interrupt Register
USBOTG_DOEPEACHMSKn	0x0880	W	0x00000000	Device each out endpoint-n interrupt register
USBOTG_DIEPCTL0	0x0900	W	0x00008000	Device control IN endpoint 0 control register
USBOTG_DIEPINTn	0x0908	W	0x00000000	Device Endpoint-n Interrupt Register
USBOTG_DIEPTSIzn	0x0910	W	0x00000000	Device endpoint n transfer size register
USBOTG_DIEPDMAAn	0x0914	W	0x00000000	Device endpoint-n DMA address register
USBOTG_DTXFSTSn	0x0918	W	0x00000000	Device IN endpoint transmit FIFO status register
USBOTG_DIEPDMABn	0x091c	W	0x00000000	Device endpoint-n DMA buffer address register
USBOTG_DIEPCTLn	0x0920	W	0x00000000	Device endpoint-n control register
USBOTG_DOEPCTL0	0x0b00	W	0x00000000	Device control OUT endpoint 0 control register
USBOTG_DOEPINTn	0x0b08	W	0x00000000	Device endpoint-n control register
USBOTG_DOEPTSIzn	0x0b10	W	0x00000000	Device endpoint n transfer size register
USBOTG_DOEPDMAAn	0x0b14	W	0x00000000	Device Endpoint-n DMA Address Register
USBOTG_DOEPDMABn	0x0b1c	W	0x00000000	Device endpoint-n DMA buffer address register
USBOTG_DOEPCTLn	0x0b20	W	0x00000000	Device endpoint-n control register
USBOTG_PCGCR	0x0b24	W	0x200b8000	Power and clock gating control register
USBOTG_EPBUFO	0x1000	W	0x00000000	Device endpoint 0 / host out channel 0 address
USBOTG_EPBUF1	0x2000	W	0x00000000	Device endpoint 1 / host out channel 1 address
USBOTG_EPBUF2	0x3000	W	0x00000000	Device endpoint 2 / host out channel 2 address
USBOTG_EPBUF3	0x4000	W	0x00000000	Device endpoint 3 / host out channel 3 address
USBOTG_EPBUF4	0x5000	W	0x00000000	Device endpoint 4 / host out channel 4 address

Name	Offset	Size	Reset Value	Description
USBOTG_EPBUF5	0x6000	W	0x00000000	Device endpoint 5 / host out channel 5 address
USBOTG_EPBUF6	0x7000	W	0x00000000	Device endpoint 6 / host out channel 6 address
USBOTG_EPBUF7	0x8000	W	0x00000000	Device endpoint 7 / host out channel 7 address

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

2.6.2 Detail Register Description

USBOTG_GOTGCTL

Address: Operational Base + offset (0x0000)

Control and Status Register

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27	RW	0x0	ChirpEn Chirp on enable This bit when programmed to 1'b1 results in the core asserting chirp_on before sending an actual Chirp "K" signal on USB. This bit is present only if OTG_BC_SUPPORT = 1. If OTG_BC_SUPPORT != 1, this bit is a reserved bit.
26:22	RO	0x00	MultValidBc Multi Valued ID pin Battery Charger ACA inputs in the following order: Bit 26 - rid_float. Bit 25 - rid_gnd Bit 24 - rid_a Bit 23 - rid_b Bit 22 - rid_c These bits are present only if OTG_BC_SUPPORT = 1. Otherwise, these bits are reserved and will read 5'h0.
21	RO	0x0	reserved
20	RW	0x0	OTGVer OTG version Indicates the OTG revision. 0: OTG Version 1.3. In this version the core supports Data line pulsing and VBus pulsing for SRP. 1: OTG Version 2.0. In this version the core supports only Data line pulsing for SRP.

Bit	Attr	Reset Value	Description
19	RO	0x0	<p>BSesVld B-session valid Indicates the Device mode transceiver status. 0: B-session is not valid. 1: B-session is valid.</p> <p>In OTG mode, you can use this bit to determine if the device is connected or disconnected. Note: If you do not enable OTG features (such as SRP and HNP), the read reset value will be 1. The vbus assigns the values internally for non-SRP or non-HNP configurations.</p>
18	RO	0x0	<p>ASesVld A-session valid Indicates the Host mode transceiver status. 0: A-session is not valid 1: A-session is valid</p> <p>Note: If you do not enable OTG features (such as SRP and HNP), the read reset value will be 1. The vbus assigns the values internally for non-SRP or non-HNP configurations.</p>
17	RO	0x0	<p>DbnTime Long/short debounce time Indicates the debounce time of a detected connection. 0: Long debounce time, used for physical connections (100 ms + 2.5 us) 1: Short debounce time, used for soft connections (2.5 us)</p>
16	RO	0x0	<p>ConIDSts Connector ID Status Indicates the connector ID status on a connect event. 0: The core is in A-Device mode 1: The core is in B-Device mode</p>
15:12	RO	0x0	reserved
11	RW	0x0	<p>DevHNPEn Device HNP Enable The application sets this bit when it successfully receives a SetFeature. SetHNPEnable command from the connected USB host. 0: HNP is not enabled in the application 1: HNP is enabled in the application</p>
10	RW	0x0	<p>HstSetHNPEn Host set HNP enable The application sets this bit when it has successfully enabled HNP (using the SetFeature. SetHNPEnable command) on the connected device. 0: Host Set HNP is not enabled 1: Host Set HNP is enabled</p>

Bit	Attr	Reset Value	Description
9	RW	0x0	<p>HNPReq HNP request</p> <p>The application sets this bit to initiate an HNP request to the connected USB host. The application can clear this bit by writing a 0 when the Host Negotiation Success Status Change bit in the OTG Interrupt register (GOTGINT.HstNegSucStsChng) is set. The core clears this bit when the HstNegSucStsChng bit is cleared.</p> <p>0: No HNP request 1: HNP request</p>
8	RO	0x0	<p>HstNegScs Host Negotiation Success</p> <p>The core sets this bit when host negotiation is successful. The core clears this bit when the HNP Request (HNPReq) bit in this register is set.</p> <p>0: Host negotiation failure 1: Host negotiation success</p>
7:2	RO	0x0	reserved
1	RW	0x0	<p>SesReq Session Request</p> <p>The application sets this bit to initiate a session request on the USB. The application can clear this bit by writing a 0 when the Host Negotiation Success Status Change bit in the OTG Interrupt register (GOTGINT.HstNegSucStsChng) is set. The core clears this bit when the HstNegSucStsChng bit is cleared. If you use the USB 1.1 Full-Speed Serial Transceiver interface to initiate the session request, the application must wait until the VBUS discharges to 0.2 V, after the B-Session Valid bit in this register (GOTGCTL.BSesVld) is cleared. This discharge time varies between different PHYs and can be obtained from the PHY vendor.</p> <p>0: No session request 1: Session request</p>
0	RO	0x0	<p>SesReqScs Session Request Success</p> <p>The core sets this bit when a session request initiation is successful.</p> <p>0: Session request failure 1: Session request success</p>

USBOTG_GOTGINT

Address: Operational Base + offset (0x0004)
Interrupt Register

Bit	Attr	Reset Value	Description
31:21	RO	0x0	reserved

Bit	Attr	Reset Value	Description
20	W1C	0x0	MultiValueChg Multi-Valued input changed This bit when set indicates that there is a change in the value of at least one ACA pin value. This bit is present only if OTG_BC_SUPPORT = 1, otherwise it is reserved.
19	W1C	0x0	DbnceDone Debounce Done The core sets this bit when the debounce is completed after the device connect. The application can start driving USB reset after seeing this interrupt. This bit is only valid when the HNP Capable or SRP Capable bit is set in the Core USB Configuration register (GUSBCFG.HNPCap or GUSBCFG.SRPCap, respectively).
18	W1C	0x0	ADevTOUTChg A-Device Timeout Change The core sets this bit to indicate that the A-device has timed out while waiting for the B-device to connect.
17	W1C	0x0	HstNegDet Host Negotiation Detected The core sets this bit when it detects a host negotiation request on the USB
16:10	RO	0x0	reserved
9	W1C	0x0	HstNegSucStsChng Host Negotiation Success Status Change The core sets this bit on the success or failure of a USB host negotiation request. The application must read the Host Negotiation Success bit of the OTG Control and Status register (GOTGCTL.HstNegScs) to check for success or failure
8	W1C	0x0	SesReqSucStsChng Session Request Success Status Change The core sets this bit on the success or failure of a session request. The application must read the Session Request Success bit in the OTG Control and Status register (GOTGCTL.SesReqScs) to check for success or failure.
7:3	RO	0x0	reserved
2	W1C	0x0	SesEndDet Session End Detected The core sets this bit when the utmisrp_bvalid signal is deasserted
1:0	RO	0x0	reserved

USBOTG_GAHBCFG

Address: Operational Base + offset (0x0008)

AHB Configuration Register

Bit	Attr	Reset Value	Description
31:23	RO	0x0	reserved

Bit	Attr	Reset Value	Description
22	RW	0x0	<p>NotiAllDmaWrit Notify All Dma Write Transactions This bit is programmed to enable the System DMA Done functionality for all the DMA write Transactions corresponding to the Channel/Endpoint. This bit is valid only when GAHBCFG.RemMemSupp is set to 1. GAHBCFG.NotiAllDmaWrit = 1.</p> <p>HSOTG core asserts int_dma_req for all the DMA write transactions on the AHB interface along with int_dma_done, chep_last_transact and chep_number signal informations. The core waits for sys_dma_done signal for all the DMA write transactions in order to complete the transfer of a particular Channel/Endpoint. GAHBCFG.NotiAllDmaWrit = 0.</p> <p>HSOTG core asserts int_dma_req signal only for the last transaction of DMA write transfer corresponding to a particular Channel/Endpoint. Similarly, the core waits for sys_dma_done signal only for that transaction of DMA write to complete the transfer of a particular Channel/Endpoint.</p>
21	RW	0x0	<p>RemMemSupp Remote Memory Support This bit is programmed to enable the functionality to wait for the system DMA Done Signal for the DMA Write Transfers. GAHBCFG.RemMemSupp=1.</p> <p>The int_dma_req output signal is asserted when HSOTG DMA starts write transfer to the external memory. When the core is done with the Transfers it asserts int_dma_done signal to flag the completion of DMA writes from HSOTG. The core then waits for sys_dma_done signal from the system to proceed further and complete the Data Transfer corresponding to a particular Channel/Endpoint. GAHBCFG.RemMemSupp=0.</p> <p>The int_dma_req and int_dma_done signals are not asserted and the core proceeds with the assertion of the XferComp interrupt as soon as the DMA write transfer is done at the HSOTG Core Boundary and it does not wait for the sys_dma_done signal to complete the DATA transfers.</p>
20:9	RO	0x0	reserved
8	RW	0x0	<p>PTxFEmpLvl Periodic TxFIFO Empty Level Indicates when the Periodic TxFIFO Empty Interrupt bit in the Core Interrupt register (GINTSTS.PTxFEmp) is triggered. This bit is used only in Slave mode.</p> <p>0: GINTSTS.PTxFEmp interrupt indicates that the Periodic TxFIFO is half empty</p> <p>1: GINTSTS.PTxFEmp interrupt indicates that the Periodic TxFIFO is completely empty</p>

Bit	Attr	Reset Value	Description
7	RW	0x0	<p>NPTxFEmpLvl Non-Periodic TxFIFO Empty Level This bit is used only in Slave mode. In host mode and with Shared FIFO with device mode, this bit indicates when the Non-Periodic TxFIFO Empty Interrupt bit in the Core Interrupt register GINTSTS.NPTxFEmp) is triggered. With dedicated FIFO in device mode, this bit indicates when IN endpoint Transmit FIFO empty interrupt (DIEPINTn.TxFEmp) is triggered.</p> <p>Host mode and with Shared FIFO with device mode:</p> <p>1'b0: GINTSTS.NPTxFEmp interrupt indicates that the Non-Periodic TxFIFO is half empty</p> <p>1'b1: GINTSTS.NPTxFEmp interrupt indicates that the Non-Periodic TxFIFO is completely empty</p> <p>Dedicated FIFO in device mode:</p> <p>1'b0: DIEPINTn.TxFEmp interrupt indicates that the IN Endpoint TxFIFO is half empty</p> <p>1'b1: DIEPINTn.TxFEmp interrupt indicates that the IN Endpoint TxFIFO is completely empty</p>
6	RO	0x0	reserved
5	RW	0x0	<p>DMAEn DMA Enable 0: Core operates in Slave mode 1: Core operates in a DMA mode This bit is always 0 when Slave-Only mode has been selected.</p>
4:1	RW	0x0	<p>HBstLen Burst Length/Type This field is used in both External and Internal DMA modes. In External DMA mode, these bits appear on dma_burst[3:0] ports, External DMA Mode defines the DMA burst length in terms of 32-bit words:</p> <p>4'b0000: 1 word 4'b0001: 4 words 4'b0010: 8 words 4'b0011: 16 words 4'b0100: 32 words 4'b0101: 64 words 4'b0110: 128 words 4'b0111: 256 words Others: Reserved</p> <p>Internal DMA Mode AHB Master burst type:</p> <p>4'b0000: Single 4'b0001: INCR 4'b0011: INCR4 4'b0101: INCR8 4'b0111: INCR16 Others: Reserved</p>

Bit	Attr	Reset Value	Description
0	RW	0x0	<p>GlblIntrMsk Global Interrupt Mask</p> <p>The application uses this bit to mask or unmask the interrupt line assertion to itself. Irrespective of this bit's setting, the interrupt status registers are updated by the core.</p> <p>1'b0: Mask the interrupt assertion to the application. 1'b1: Unmask the interrupt assertion to the application.</p>

USBOTG_GUSBCFG

Address: Operational Base + offset (0x000c)

USB Configuration Register

Bit	Attr	Reset Value	Description
31	RW	0x0	<p>CorruptTxpacket Corrupt Tx packet</p> <p>This bit is for debug purposes only. Never set this bit to 1.</p>
30	RW	0x0	<p>ForceDevMode Force Device Mode</p> <p>Writing a 1 to this bit forces the core to device mode irrespective of utmiotg_iddig input pin.</p> <p>1'b0: Normal Mode 1'b1: Force Device Mode</p> <p>After setting the force bit, the application must wait at least 25 ms before the change to take effect. When the simulation is in scale down mode, waiting for 500 us is sufficient. This bit is valid only when OTG_MODE = 0, 1 or 2. In all other cases, this bit reads 0.</p>
29	RW	0x0	<p>ForceHstMode Force Host Mode</p> <p>Writing a 1 to this bit forces the core to host mode irrespective of utmiotg_iddig input pin.</p> <p>1'b0: Normal Mode 1'b1: Force Host Mode</p> <p>After setting the force bit, the application must wait at least 25 ms before the change to take effect. When the simulation is in scale down mode, waiting for 500 us is sufficient. This bit is valid only when OTG_MODE = 0, 1 or 2. In all other cases, this bit reads 0.</p>
28	RW	0x0	<p>TxEndDelay Tx End Delay</p> <p>Writing a 1 to this bit enables the TxEndDelay timers in the core.</p> <p>1'b0: Normal mode 1'b1: Introduce Tx end delay timers</p>

Bit	Attr	Reset Value	Description
27	RW	0x0	<p>IC_USBTrfCtl IC_USB TrafficPullRemove Control When this bit is set, pullup/pulldown resistors are detached from the USB during traffic signaling. This bit is valid only when configuration parameter OTG_ENABLE_IC_USB = 1 and register field GUSBCFG.IC_USBCap is set to 1.</p>
26	RW	0x0	<p>IC_USBCap IC_USB-Capable The application uses this bit to control the IC_USB capabilities. 1'b0: IC_USB PHY Interface is not selected. 1'b1: IC_USB PHY Interface is selected. This bit is writable only if OTG_ENABLE_IC_USB=1 and OTG_FSPHY_INTERFACE!=0. The reset value depends on the configuration parameter OTG_SELECT_IC_USB when OTG_ENABLE_IC_USB = 1. In all other cases, this bit is set to 1'b0 and the bit is read only.</p>
25	RW	0x0	<p>ULPIIfDis ULPI Interface Protect Disable Controls circuitry built into the PHY for protecting the ULPI interface when the link tri-states STP and data. Any pull-ups or pull-downs employed by this feature can be disabled. Please refer to the ULPI Specification for more detail. 1'b0: Enables the interface protect circuit 1'b1: Disables the interface protect circuit</p>
24	RW	0x0	<p>IndPassThrough Indicator Pass Through Controls whether the Complement Output is qualified with the Internal Vbus Valid comparator before being used in the Vbus State in the RX CMD. Please refer to the ULPI Specification for more detail. 1'b0: Complement Output signal is qualified with the Internal VbusValid comparator. 1'b1: Complement Output signal is not qualified with the Internal VbusValid comparator.</p>
23	RW	0x0	<p>IndComple Indicator Complement Controls the PHY to invert the ExternalVbusIndicator input signal, generating the Complement Output. Please refer to the ULPI Specification for more detail 1'b0: PHY does not invert ExternalVbusIndicator signal 1'b1: PHY does invert ExternalVbusIndicator signal</p>

Bit	Attr	Reset Value	Description
22	RW	0x0	<p>TermSelDLPulse TermSel DLine Pulsing Selection This bit selects utmi_termselect to drive data line pulse during SRP. 1'b0: Data line pulsing using utmi_txvalid (default). 1'b1: Data line pulsing using utmi_termsel.</p>
21	RW	0x0	<p>ULPIExtVbusIndicator ULPI External VBUS Indicator This bit indicates to the ULPI PHY to use an external VBUS over-current indicator. 1'b0: PHY uses internal VBUS valid comparator. 1'b1: PHY uses external VBUS valid comparator. (Valid only when RTL parameter OTG_HSPHY_INTERFACE = 2 or 3)</p>
20	RW	0x0	<p>ULPIExtVbusDrv ULPI External VBUS Drive This bit selects between internal or external supply to drive 5V on VBUS,in ULPI PHY. 1'b0: PHY drives VBUS using internal charge pump (default). 1'b1: PHY drives VBUS using external supply. (Valid only when RTL parameter OTG_HSPHY_INTERFACE = 2 or 3)</p>
19	RW	0x0	<p>ULPIClkSusM ULPI Clock SuspendM This bit sets the ClockSuspendM bit in the Interface Control register on the ULPI PHY. This bit applies only in serial or carkit modes. 1'b0: PHY powers down internal clock during suspend. 1'b1: PHY does not power down internal clock. (Valid only when RTL parameter OTG_HSPHY_INTERFACE = 2 or 3)</p>
18	RW	0x0	<p>ULPIAutoRes ULPI Auto Resume This bit sets the AutoResume bit in the Interface Control register on the ULPI PHY. 1'b0: PHY does not use AutoResume feature. 1'b1: PHY uses AutoResume feature. (Valid only when RTL parameter OTG_HSPHY_INTERFACE = 2 or 3)</p>
17	RW	0x0	<p>ULPIFsLs ULPI FS/LS Select The application uses this bit to select the FS/LS serial interface for the ULPI PHY. This bit is valid only when the FS serial transceiver is selected on the ULPI PHY. 1'b0: ULPI interface 1'b1: ULPI FS/LS serial interface (Valid only when RTL parameters OTG_HSPHY_INTERFACE = 2 or 3 and OTG_FSPHY_INTERFACE = 1, 2, or 3)</p>

Bit	Attr	Reset Value	Description
16	RW	0x0	<p>OtgI2CSel UTMIFS or I2C Interface Select The application uses this bit to select the I2C interface. 1'b0: UTMI USB 1.1 Full-Speed interface for OTG signals 1'b1: I2C interface for OTG signals This bit is writable only if I2C and UTMIFS were specified for Enable I2C Interface? (parameter OTG_I2C_INTERFACE = 2). Otherwise, reads return 0.</p>
15	RW	0x0	<p>PhyLPwrClkSel PHY Low-Power Clock Select Selects either 480-MHz or 48-MHz (low-power) PHY mode. In FS and LS modes, the PHY can usually operate on a 48-MHz clock to save power. 1'b0: 480-MHz Internal PLL clock 1'b1: 48-MHz External Clock In 480 MHz mode, the UTMI interface operates at either 60 or 30-MHz, depending upon whether 8- or 16-bit data width is selected. In 48-MHz mode, the UTMI interface operates at 48 MHz in FS and LS modes. This bit drives the utmi_fs_ls_low_power core output signal, and is valid only for UTMI+ PHYs.</p>
14	RO	0x0	reserved
13:10	RW	0x5	<p>USBTrdTim USB Turnaround Time Sets the turnaround time in PHY clocks. Specifies the response time for a MAC request to the Packet FIFO Controller (PFC) to fetch data from the DFIF(SPRAM). This must be programmed to 4'h5: When the MAC interface is 16-bit UTMI+. 4'h9: When the MAC interface is 8-bit UTMI+. Note: The values above are calculated for the minimum AHB frequency of 30 MHz. USB turnaround time is critical for certification where long cables and 5-Hubs are used, so if you need the AHB to run at less than 30 MHz, and if USB turnaround time is not critical, these bits can be programmed to a larger value.</p>
9	RW	0x0	<p>HNPcap HNP-Capable The application uses this bit to control the DWC_otg core's HNP capabilities. 0: HNP capability is not enabled. 1: HNP capability is enabled. This bit is writable only if an HNP mode was specified for Mode of Operation (parameter OTG_MODE). Otherwise, reads return 0.</p>

Bit	Attr	Reset Value	Description
8	RW	0x0	<p>SRPCap SRP-Capable</p> <p>The application uses this bit to control the DWC_otg core SRP capabilities. If the core operates as a non-SRP-capable B-device, it cannot request the connected A-device (host) to activate VBUS and start a session.</p> <p>0: SRP capability is not enabled. 1: SRP capability is enabled.</p> <p>This bit is writable only if an SRP mode was specified for Mode of Operation in coreConsultant (parameter OTG_MODE). Otherwise, reads return 0.</p>
7	RW	0x0	<p>DDRSel ULPI DDR Select</p> <p>The application uses this bit to select a Single Data Rate (SDR) or Double Data Rate (DDR) or ULPI interface.</p> <p>0: Single Data Rate ULPI Interface, with 8-bit-wide data bus 1: Double Data Rate ULPI Interface, with 4-bit-wide data bus</p>
6	RW	0x0	<p>PHYSel USB 2.0 High-Speed PHY or USB 1.1 Full-Speed Serial Transceiver</p> <p>The application uses this bit to select either a high-speed UTMI+ or ULPI PHY, or a full-speed transceiver.</p> <p>0: USB 2.0 high-speed UTMI+ or ULPI PHY 1: USB 1.1 full-speed serial transceiver</p> <p>If a USB 1.1 Full-Speed Serial Transceiver interface was not selected (parameter OTG_FSPHY_INTERFACE = 0), this bit is always 0, with Write Only access. If a high-speed PHY interface was not selected (parameter OTG_HSPHY_INTERFACE = 0), this bit is always 1, with Write Only access.</p> <p>If both interface types were selected in coreConsultant (parameters have non-zero values), the application uses this bit to select which interface is active, and access is Read and Write.</p>
5	RW	0x0	<p>FSIntf Full-Speed Serial Interface Select</p> <p>The application uses this bit to select either a unidirectional or bidirectional USB 1.1 full-speed serial transceiver interface.</p> <p>0: 6-pin unidirectional full-speed serial interface 1: 3-pin bidirectional full-speed serial interface</p> <p>If a USB 1.1 Full-Speed Serial Transceiver interface was not selected (parameter OTG_FSPHY_INTERFACE = 0), this bit is always 0, with Write Only access. If a USB 1.1 FS interface was selected (parameter OTG_FSPHY_INTERFACE! = 0), then the application can set this bit to select between the 3- and 6-pin interfaces, and access is Read and Write.</p>

Bit	Attr	Reset Value	Description
4	RW	0x0	<p>ULPI_UTMI_Sel ULPI or UTMI+ Select The application uses this bit to select either a UTMI+ interface or ULPI Interface. 0: UTMI+ Interface 1: ULPI Interface This bit is writable only if UTMI+ and ULPI was specified for High-Speed PHY Interface(s) in coreConsultant configuration (parameter OTG_HSPHY_INTERFACE = 3). Otherwise, reads return either 0 or 1, depending on the interface selected using the OTG_HSPHY_INTERFACE parameter.</p>
3	RW	0x0	<p>PHYIf PHY Interface The application uses this bit to configure the core to support a UTMI+ PHY with an 8- or 16-bit interface. When a ULPI PHY is chosen, this must be set to 8-bit mode. 0: 8 bits 1: 16 bits This bit is writable only if UTMI+ and ULPI were selected (parameter OTG_HSPHY_DWIDTH = 3). Otherwise, this bit returns the value for the power-on interface selected during configuration.</p>
2:0	RW	0x0	<p>TOutCal HS/FS Timeout Calibration The number of PHY clocks that the application programs in this field is added to the high-speed/full-speed interpacket timeout duration in the core to account for any additional delays introduced by the PHY. This can be required, because the delay introduced by the PHY in generating the line state condition can vary from one PHY to another. The USB standard timeout value for high-speed operation is 736 to 816 (inclusive) bit times. The USB standard timeout value for full-speed operation is 16 to 18 (inclusive) bit times. The application must program this field based on the speed of enumeration. The number of bit times added per PHY clock are: High-speed operation: One 30-MHz PHY clock = 16 bit times One 60-MHz PHY clock = 8 bit times Full-speed operation: One 30-MHz PHY clock = 0.4 bit times One 60-MHz PHY clock = 0.2 bit times One 48-MHz PHY clock = 0.25 bit times</p>

USBOTG_GRSTCTL

Address: Operational Base + offset (0x0010)
Reset Register

Bit	Attr	Reset Value	Description
31	RO	0x1	AHBIdle AHB Master Idle Indicates that the AHB Master State Machine is in the IDLE condition.
30	RO	0x0	DMAReq DMA Request Signal Indicates that the DMA request is in progress. Used for debug.
29:11	RO	0x0	reserved
10:6	RW	0x00	TxFNum Tx FIFO Number This is the FIFO number that must be flushed using the Tx FIFO Flush bit. This field must not be changed until the core clears the Tx FIFO Flush bit. 5'h0: Non-periodic Tx FIFO flush in Host mode; Non-periodic Tx FIFO flush in device mode when in shared FIFO operation. Tx FIFO 0 flush in device mode when in dedicated FIFO mode. 5'h1: Periodic Tx FIFO flush in Host mode: Periodic Tx FIFO 1 flush in Device mode when in shared FIFO operation; TX FIFO 1 flush in device mode when in dedicated FIFO mode. 5'h2: Periodic Tx FIFO 2 flush in Device mode when in shared FIFO operation: TX FIFO 2 flush in device mode when in dedicated FIFO mode. ... 5'hF: Periodic Tx FIFO 15 flush in Device mode when in shared FIFO operation: TX FIFO 15 flush in device mode when in dedicated FIFO mode. 5'h10: Flush all the transmit FIFOs in device or host mode.
5	R/W SC	0x0	TxFFlsh Tx FIFO Flush This bit selectively flushes a single or all transmit FIFOs, but cannot do so if the core is in the midst of a transaction. The application must write this bit only after checking that the core is neither writing to the Tx FIFO nor reading from the Tx FIFO. Verify using these registers: Read NAK Effective Interrupt ensures the core is not reading from the FIFO. Write GRSTCTL.AHBIdle ensures the core is not writing anything to the FIFO. Flushing is normally recommended when FIFOs are re-configured or when switching between Shared FIFO and Dedicated Transmit FIFO operation. FIFO flushing is also recommended during device endpoint disable. The application must wait until the core clears this bit before performing any operations. This bit takes eight clocks to clear, using the slower clock of phy_clk or hclk.

Bit	Attr	Reset Value	Description
4	R/W SC	0x0	RxFFlsh Rx FIFO Flush The application can flush the entire Rx FIFO using this bit, but must first ensure that the core is not in the middle of a transaction. The application must only write to this bit after checking that the core is neither reading from the Rx FIFO nor writing to the Rx FIFO. The application must wait until the bit is cleared before performing any other operations. This bit requires 8 clocks (slowest of PHY or AHB clock) to clear.
3	R/W SC	0x0	INTknQFlsh IN Token Sequence Learning Queue Flush This bit is valid only if OTG_ENDED_TX_FIFO = 0. The application writes this bit to flush the IN Token Sequence Learning Queue.
2	W1C	0x0	FrmCntrRst Host Frame Counter Reset The application writes this bit to reset the (micro)frame number counter inside the core. When the (micro)frame counter is reset, the subsequent SOF sent out by the core has a (micro)frame number of 0.
1	R/W SC	0x0	Reset A write to this bit issues a soft reset to the DWC_otg_power_dn module of the core.

Bit	Attr	Reset Value	Description
0	R/W SC	0x0	<p>CSftRst Core Soft Reset Resets the hclk and phy_clock domains as follows: Clears the interrupts and all the CSR registers except the following register bits:</p> <ul style="list-style-type: none"> CGCCTL.RstPdwnModule PCGCCTL.GateHclk PCGCCTL.PwrClmp PCGCCTL.StopPPhyLPwrClkSelClk GUSBCFG.PhyLPwrClkSel GUSBCFG.DDRSel GUSBCFG.PHYSel GUSBCFG.FSIntf GUSBCFG.ULPI_UTMI_Sel GUSBCFG.PHYIf HCFG.FSLSPclkSel DCFG.DevSpd GGPIO GPWRDN GADPCTL <p>All module state machines (except the AHB Slave Unit) are reset to the IDLE state, and all the transmit FIFOs and the receive FIFO are flushed. Any transactions on the AHB Master are terminated as soon as possible, after gracefully completing the last data phase of an AHB transfer. Any transactions on the USB are terminated immediately. When Hibernation or ADP feature is enabled, the PMU module is not reset by the Core Soft Reset. The application can write to this bit any time it wants to reset the core. This is a self-clearing bit and the core clears this bit after all the necessary logic is reset in the core, which can take several clocks, depending on the current state of the core. Once this bit is cleared software must wait at least 3 PHY clocks before doing any access to the PHY domain (synchronization delay). Software must also check that bit 31 of this register is 1 (AHB Master is IDLE) before starting any operation. Typically software reset is used during software development and also when you dynamically change the PHY selection bits in the USB configuration registers listed above. When you change the PHY, the corresponding clock for the PHY is selected and used in the PHY domain. Once a new clock is selected, the PHY domain has to be reset for proper operation.</p>

USBOTG_GINTSTS

Address: Operational Base + offset (0x0014)

Interrupt Register

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

Bit	Attr	Reset Value	Description
31	W1C	0x0	<p>WkUpInt Resume/Remote Wakeup Detected Interrupt Wakeup Interrupt during Suspend(L2) or LPM(L1) state.</p> <p>During Suspend(L2): Device Mode: This interrupt is asserted only when Host Initiated Resume is detected on USB. Host Mode: This interrupt is asserted only when Device Initiated Remote Wakeup is detected on USB.</p> <p>During LPM(L1): Device Mode: This interrupt is asserted for either Host Initiated Resume or Device Initiated Remote Wakeup on USB. Host Mode: This interrupt is asserted for either Host Initiated Resume or Device Initiated Remote Wakeup on USB.</p>
30	W1C	0x0	<p>SessReqInt Session Request/New Session Detected Interrupt In Host mode, this interrupt is asserted when a session request is detected from the device. In Host mode, this interrupt is asserted when a session request is detected from the device. In Device mode, this interrupt is asserted when the utmisrp_bvalid signal goes high.</p>
29	W1C	0x0	<p>DisconnInt Disconnect Detected Interrupt This interrupt is asserted when a device disconnect is detected.</p>
28	W1C	0x0	<p>ConIDStsChng Connector ID Status Change This interrupt is asserted when there is a change in connector ID status.</p>
27	W1C	0x0	<p>LPM_Int LPM Transaction Received Interrupt Device Mode : This interrupt is asserted when the device receives an LPM transaction and responds with a non-ERRORed response. Host Mode : This interrupt is asserted when the device responds to an LPM transaction with a non-ERRORed response or when the host core has completed LPM transactions for the programmed number of times (GLPMCFG.RetryCnt). This field is valid only if the Core LPM Configuration register's LPMCapable (LPMCap) field is set to 1.</p>
26	RO	0x0	<p>PTxFEmp Periodic TxFIFO Empty This interrupt is asserted when the Periodic Transmit FIFO is either half or completely empty and there is space for at least one entry to be written in the Periodic Request Queue. The half or completely empty status is determined by the Periodic TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.PTxFEmpLvl).</p>

Bit	Attr	Reset Value	Description
25	RO	0x0	HChInt Host Channels Interrupt The core sets this bit to indicate that an interrupt is pending on one of the channels of the core (in Host mode). The application must read the Host All Channels Interrupt (HAINT) register to determine the exact number of the channel on which the interrupt occurred, and then read the corresponding Host Channel-n Interrupt (HCINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the HCINTn register to clear this bit.
24	RO	0x0	PrtInt Host Port Interrupt The core sets this bit to indicate a change in port status of one of the DWC_otg core ports in Host mode. The application must read the Host Port Control and Status (HPRT) register to determine the exact event that caused this interrupt. The application must clear the appropriate status bit in the Host Port Control and Status register to clear this bit.
23	RW	0x0	ResetDet Reset Detected Interrupt The core asserts this interrupt in Device mode when it detects a reset on the USB in Partial Power-Down mode when the device is in Suspend. This interrupt is not asserted in Host mode.
22	W1C	0x0	FetSusp Data Fetch Suspended This interrupt is valid only in DMA mode. This interrupt indicates that the core has stopped fetching data for IN endpoints due to the unavailability of TxFIFO space or Request Queue space. This interrupt is used by the application for an endpoint mismatch algorithm. For example, after detecting an endpoint mismatch, the application: Sets a global non-periodic IN NAK handshake, Disables In endpoints, Flushes the FIFO, Determines the token sequence from the IN Token Sequence Learning Queue, Re-enables the endpoints, Clears the global non-periodic IN NAK handshake. If the global non-periodic IN NAK is cleared, the core has not yet fetched data for the IN endpoint, and the IN token is received: the core generates an "IN token received when FIFO empty" interrupt. The OTG then sends the host a NAK response. To avoid this scenario, the application can check the GINTSTS.FetSusp interrupt, which ensures that the FIFO is full before clearing a global NAK handshake. Alternatively, the application can mask the IN token received when FIFO empty interrupt when clearing a global IN NAK handshake.

Bit	Attr	Reset Value	Description
21	W1C	0x0	<p>incomlP Incomplete Periodic Transfer</p> <p>In Host mode, the core sets this interrupt bit when there are incomplete periodic transactions still pending which are scheduled for the current microframe. Incomplete Isochronous OUT Transfer (incompISOOUT) The Device mode, the core sets this interrupt to indicate that there is at least one isochronous OUT endpoint on which the transfer is not completed in the current microframe. This interrupt is asserted along with the End of Periodic Frame Interrupt (EOPF) bit in this register.</p>
20	W1C	0x0	<p>incompISOIN Incomplete Isochronous IN Transfer</p> <p>The core sets this interrupt to indicate that there is at least one isochronous IN endpoint on which the transfer is not completed in the current microframe. This interrupt is asserted along with the End of Periodic Frame Interrupt (EOPF) bit in this register. Note: This interrupt is not asserted in Scatter/Gather DMA mode.</p>
19	RO	0x0	<p>OEPInt OUT Endpoints Interrupt</p> <p>The core sets this bit to indicate that an interrupt is pending on one of the OUT endpoints of the core (in Device mode). The application must read the Device All Endpoints Interrupt (DAINT) register to determine the exact number of the OUT endpoint on which the interrupt occurred, and then read the corresponding Device OUT Endpoint-n Interrupt (DOEPINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the corresponding DOEPINTn register to clear this bit.</p>
18	RO	0x0	<p>IEPInt IN Endpoints Interrupt</p> <p>The core sets this bit to indicate that an interrupt is pending on one of the IN endpoints of the core (in Device mode). The application must read the Device All Endpoints Interrupt (DAINT) register to determine the exact number of the IN endpoint on which the interrupt occurred, and then read the corresponding Device IN Endpoint-n Interrupt (DIEPINTn) register to determine the exact cause of the interrupt. The application must clear the appropriate status bit in the corresponding DIEPINTn register to clear this bit.</p>
17	W1C	0x0	<p>EPMIs Endpoint Mismatch Interrupt</p> <p>Note: This interrupt is valid only in shared FIFO operation.</p> <p>Indicates that an IN token has been received for a non-periodic endpoint, but the data for another endpoint is present in the top of the Non-periodic Transmit FIFO and the IN endpoint mismatch count programmed by the application has expired.</p>

Bit	Attr	Reset Value	Description
16	W1C	0x0	RstrDoneInt Restore Done Interrupt The core sets this bit to indicate that the restore command after Hibernation was completed by the core. The core continues from Suspended state into the mode dictated by PCGCCTL.RestoreMode field. This bit is valid only when Hibernation feature is enabled.
15	W1C	0x0	EOPF End of Periodic Frame Interrupt Indicates that the period specified in the Periodic Frame Interval field of the Device Configuration register (DCFG.PerFrInt) has been reached in the current microframe.
14	W1C	0x0	ISOOutDrop Isochronous OUT Packet Dropped Interrupt The core sets this bit when it fails to write an isochronous OUT packet into the Rx FIFO because the Rx FIFO does not have enough space to accommodate a maximum packet size packet for the isochronous OUT endpoint.
13	W1C	0x0	EnumDone Enumeration Done The core sets this bit to indicate that speed enumeration is complete. The application must read the Device Status (DSTS) register to obtain the enumerated speed.
12	W1C	0x0	USBRst USB Reset The core sets this bit to indicate that a reset is detected on the USB.
11	W1C	0x0	USBSusp USB Suspend The core sets this bit to indicate that a suspend was detected on the USB. The core enters the Suspended state when there is no activity on the utmi_linestate signal for an extended period of time.
10	W1C	0x0	ErlySusp Early Suspend The core sets this bit to indicate that an Idle state has been detected on the USB for 3 ms.
9	W1C	0x0	I2CINT I2C Interrupt The core sets this interrupt when I2C access is completed on the I2C interface. This field is used only if the I2C interface was enabled . Otherwise, reads return 0.

Bit	Attr	Reset Value	Description
8	W1C	0x0	<p>ULPICKINT ULPI Carkit Interrupt This field is used only if the Carkit interface was enabled . Otherwise, reads return 0. The core sets this interrupt when a ULPI Carkit interrupt is received. The core's PHY sets ULPI Carkit interrupt in UART or Audio mode. I2C Carkit Interrupt (I2CCKINT) This field is used only if the I2C interface was enabled . Otherwise, reads return 0.The core sets this interrupt when a Carkit interrupt is received. The core's PHY sets the I2C Carkit interrupt in Audio mode.</p>
7	RO	0x0	<p>GOUTNakEff Global OUT NAK Effective Indicates that the Set Global OUT NAK bit in the Device Control register DCTL.SGOUTNak), set by the application, has taken effect in the core. This bit can be cleared by writing the Clear Global OUT NAK bit in the Device Control register (DCTL.CGOUTNak).</p>
6	RO	0x0	<p>GINNakEff Global IN Non-Periodic NAK Effective Indicates that the Set Global Non-periodic IN NAK bit in the Device Control register (DCTL.SGNPInNak), set by the application, has taken effect in the core. That is, the core has sampled the Global IN NAK bit set by the application. This bit can be cleared by clearing the Clear Global Nonperiodic IN NAK bit in the Device Control register (DCTL.CGNPInNak). This interrupt does not necessarily mean that a NAK handshake is sent out on the USB. The STALL bit takes precedence over the NAK bit.</p>
5	RO	0x0	<p>NPTxFEmp Non-Periodic TxFIFO Empty This interrupt is valid only when OTG_ENDED_TX_FIFO = 0. This interrupt is asserted when the Non-periodic TxFIFO is either half or completely empty, and there is space for at least one entry to be written to the Non-periodic Transmit Request Queue. The half or completely empty status is determined by the Non-periodic TxFIFO Empty Level bit in the Core AHB Configuration register(GAHBCFG.NPTxFEmpLvl).</p>
4	RO	0x0	<p>RxFLvl RxFIFO Non-Empty Indicates that there is at least one packet pending to be read from the RxFIFO.</p>

Bit	Attr	Reset Value	Description
3	W1C	0x0	Sof Start of (micro)Frame In Host mode, the core sets this bit to indicate that an SOF (FS), micro-SOF(HS), or Keep-Alive (LS) is transmitted on the USB. The application must write a 1 to this bit to clear the interrupt. In Device mode, in the core sets this bit to indicate that an SOF token has been received on the USB. The application can read the Device Status register to get the current (micro)frame number. This interrupt is seen only when the core is operating at either HS or FS.
2	RO	0x0	OTGInt OTG Interrupt The core sets this bit to indicate an OTG protocol event. The application must read the OTG Interrupt Status (GOTGINT) register to determine the exact event that caused this interrupt. The application must clear the appropriate status bit in the GOTGINT register to clear this bit.
1	W1C	0x0	ModeMis Mode Mismatch Interrupt The core sets this bit when the application is trying to access: A Host mode register, when the core is operating in Device mode ; A Device mode register, when the core is operating in Host mode. The register access is completed on the AHB with an OKAY response, but is ignored by the core internally and does not affect the operation of the core.
0	RO	0x0	CurMod Current Mode of Operation Indicates the current mode. 1'b0: Device mode 1'b1: Host mode

USBOTG_GINTMSK

Address: Operational Base + offset (0x0018)

Interrupt Mask Register

Bit	Attr	Reset Value	Description
31	RW	0x0	WkUpIntMsk Resume/Remote Wakeup Detected Interrupt Mask
30	RW	0x0	SessReqIntMsk Session Request/New Session Detected Interrupt Mask
29	RW	0x0	DisconnIntMsk Disconnect Detected Interrupt Mask
28	RW	0x0	ConIDStsChngMsk Connector ID Status Change Mask
27	RW	0x0	LPM_IntMsk LPM Transaction Received Interrupt Mask

Bit	Attr	Reset Value	Description
26	RW	0x0	PTxFEmpMsk Periodic TxFIFO Empty Mask
25	RW	0x0	HChIntMsk Host Channels Interrupt Mask
24	RW	0x0	PrtIntMsk Host Port Interrupt Mask
23	RW	0x0	ResetDetMsk Reset Detected Interrupt Mask
22	RW	0x0	FetSuspMsk Data Fetch Suspended Mask
21	RW	0x0	incomlPMsk_incompISOOUTMsk Incomplete Periodic Transfer Mask(Host only) Incomplete Isochronous OUT Transfer Mask(Device only)
20	RW	0x0	incompISOINMsk Incomplete Isochronous IN Transfer Mask
19	RW	0x0	OEPIntMsk OUT Endpoints Interrupt Mask
18	RW	0x0	IEPIntMsk IN Endpoints Interrupt Mask
17	RW	0x0	EPMisMsk Endpoint Mismatch Interrupt Mask
16	RW	0x0	RstrDoneIntMsk Restore Done Interrupt Mask This field is valid only when Hibernation feature is enabled.
15	RW	0x0	EOPFMsk End of Periodic Frame Interrupt Mask
14	RW	0x0	ISOOutDropMsk Isochronous OUT Packet Dropped Interrupt Mask
13	RW	0x0	EnumDoneMsk Enumeration Done Mask
12	RW	0x0	USBRstMsk USB Reset Mask
11	RW	0x0	USBSuspMsk USB Suspend Mask
10	RW	0x0	ErlySuspMsk Early Suspend Mask
9	RW	0x0	I2CIntMsk I2C Interrupt Mask
8	RW	0x0	ULPICKINTMsk_I2CCKINTMsk ULPI Carkit Interrupt Mask (ULPICKINTMsk) I2C Carkit Interrupt Mask (I2CCKINTMsk)
7	RW	0x0	GOUTNakEffMsk Global OUT NAK Effective Mask
6	RW	0x0	GINNakEffMsk Global Non-periodic IN NAK Effective Mask

Bit	Attr	Reset Value	Description
5	RW	0x0	NPTxFEmpMsk Non-periodic TxFIFO Empty Mask
4	RW	0x0	RxFLvIMsk Receive FIFO Non-Empty Mask
3	RW	0x0	SofMsk Start of (micro)Frame Mask
2	RW	0x0	OTGIntMsk OTG Interrupt Mask
1	RW	0x0	ModeMisMsk Mode Mismatch Interrupt Mask
0	RO	0x0	reserved

USBOTG_GRXSTSR

Address: Operational Base + offset (0x001c)

Receive Status Debug Read Register

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24:21	RO	0x0	FN Frame Number (Device Only)This is the least significant 4 bits of the (micro)frame number in which the packet is received on the USB. This field is supported only when isochronous OUT endpoints are supported.
20:17	RO	0x0	PktSts Packet Status Indicates the status of the received packet(Host Only) 4'b0010: IN data packet received 4'b0011: IN transfer completed (triggers an interrupt) 4'b0101: Data toggle error (triggers an interrupt) 4'b0111: Channel halted (triggers an interrupt) Others: Reserved Indicates the status of the received packet(Device only) 4'b0001: Global OUT NAK (triggers an interrupt) 4'b0010: OUT data packet received 4'b0011: OUT transfer completed (triggers an interrupt) 4'b0100: SETUP transaction completed (triggers an interrupt) 4'b0110: SETUP data packet received Others: Reserved
16:15	RO	0x0	DPID Data PID Indicates the Data PID of the received packet 2'b00: DATA0 2'b10: DATA1 2'b01: DATA2 2'b11: MDATA

Bit	Attr	Reset Value	Description
14:4	RW	0x000	BCnt Byte Count Indicates the byte count of the received data packet.
3:0	RO	0x0	ChNum_EPNum Channel Number(Host) Endpoint Number(Device) (Host Only) Indicates the channel number to which the current received packet belongs. (Device Only) Indicates the endpoint number to which the current received packet belongs.

USBOTG_GRXSTSP

Address: Operational Base + offset (0x0020)

Receive Status Read and Pop Register

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24:21	RO	0x0	FN Frame Number (Device Only) This is the least significant 4 bits of the (micro)frame number in which the packet is received on the USB. This field is supported only when isochronous OUT endpoints are supported.
20:17	RO	0x0	PktSts Packet Status Indicates the status of the received packet(Host Only) 4'b0010: IN data packet received 4'b0011: IN transfer completed (triggers an interrupt) 4'b0101: Data toggle error (triggers an interrupt) 4'b0111: Channel halted (triggers an interrupt) Others: Reserved Indicates the status of the received packet(Device only) 4'b0001: Global OUT NAK (triggers an interrupt) 4'b0010: OUT data packet received 4'b0011: OUT transfer completed (triggers an interrupt) 4'b0100: SETUP transaction completed (triggers an interrupt) 4'b0110: SETUP data packet received Others: Reserved
16:15	RO	0x0	DPID Data PID Indicates the Data PID of the received OUT data packet 2'b00: DATA0 2'b10: DATA1 2'b01: DATA2 2'b11: MDATA
14:4	RO	0x000	BCnt Byte Count Indicates the byte count of the received data packet.

Bit	Attr	Reset Value	Description
3:0	RO	0x0	ChNum_EPNum Channel Number(Host) Endpoint Number(Device) (Host Only) Indicates the channel number to which the current received packet belongs. (Device Only) Indicates the endpoint number to which the current received packet belongs.

USBOTG_GRXFSIZ

Address: Operational Base + offset (0x0024)

Receive FIFO Size Register

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0000	RxFDep Rx FIFO Depth This value is in terms of 32-bit words. Minimum value is 16, Maximum value is 32,768. The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth. If Enable Dynamic FIFO Sizing? was deselected, these flops are optimized, and reads return the power-on value. If Enable Dynamic FIFO Sizing? was selected , you can write a new value in this field. You can write a new value in this field. Programmed values must not exceed the power-on value.

USBOTG_GNPTXFSIZ

Address: Operational Base + offset (0x0028)

Non-Periodic Transmit FIFO Size Register

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

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>NPTxFDep Non-periodic TxFIFO For host mode, this field is always valid. For Device mode, this field is valid only when OTG_ENDED_TX_FIFO==0. This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768 This field is determined by Enable Dynamic FIFO Sizing. OTG_DFIFO_DYNAMIC = 0: These flops are optimized, and reads return the power-on value.</p> <p>OTG_DFIFO_DYNAMIC = 1: The application can write a new value in this field. Programmed values must not exceed the power-on value.</p> <p>The power-on reset value of this field is specified by OTG_ENDED_TX_FIFO: OTG_ENDED_TX_FIFO = 0: The reset value is the Largest Non-periodic Tx Data FIFO Depth parameter, OTG_TX_NPERIO_DFIFO_DEPTH. OTG_ENDED_TX_FIFO = 1: The reset value is parameter OTG_TX_HNPERIO_DFIFO_DEPTH.</p>
15:0	RW	0x0000	<p>NPTxFStAddr Non-periodic Transmit RAM For host mode, this field is always valid. This field contains the memory start address for Non-periodic Transmit FIFO RAM. This field is determined by Enable Dynamic FIFO Sizing?(OTG_DFIFO_DYNAMIC): OTG_DFIFO_DYNAMIC = 0 :These flops are optimized, and reads return the power-on value. OTG_DFIFO_DYNAMIC = 1 :The application can write a new value in this field. Programmed values must not exceed the power-on value. The power-on reset value of this field is specified by Largest Rx Data FIFO Depth (parameter OTG_RX_DFIFO_DEPTH).</p>

USBOTG_GNPTXSTS

Address: Operational Base + offset (0x002c)

Non-Periodic Transmit FIFO/Queue Status Register

Bit	Attr	Reset Value	Description
31	RO	0x0	reserved

Bit	Attr	Reset Value	Description
30:24	RO	0x00	<p>NPTxQTop Top of the Non-periodic Transmit Request Queue Entry in the Non-periodic Tx Request Queue that is currently being processed by the MAC.</p> <p>Bits [30:27]: Channel/endpoint number Bits [26:25]: 2'b00: IN/OUT token 2'b01: Zero-length transmit packet (device IN/host OUT) 2'b10: PING/CSPLIT token 2'b11: Channel halt command Bit [24]: Terminate (last entry for selected channel/endpoint)</p>
23:16	RO	0x00	<p>NPTxQSpAvail Non-periodic Transmit Request Queue Space Available Indicates the amount of free space available in the Non-periodic Transmit Request Queue. This queue holds both IN and OUT requests in Host mode. Device mode has only IN requests.</p> <p>8'h0: Non-periodic Transmit Request Queue is full 8'h1: 1 location available 8'h2: 2 locations available n: n locations available (0 <=n <= 8) Others: Reserved</p>
15:0	RO	0x0000	<p>NPTxFSpAvail Non-periodic TxFIFO Space Avail Indicates the amount of free space available in the Non-periodic TxFIFO. Values are in terms of 32-bit words.</p> <p>16'h0: Non-periodic TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 <=n <=32,768) 16'h8000: 32,768 words available Others: Reserved</p>

USBOTG_GI2CCTL

Address: Operational Base + offset (0x0030)

I2C Address Register

Bit	Attr	Reset Value	Description
31	R/W SC	0x0	<p>BsyDne I2C Busy/Done</p> <p>The application sets this bit to 1'b1 to start a request on the I2C interface. When the transfer is complete, the core deasserts this bit to 1'b0. As long as the bit is set, indicating that the I2C interface is busy, the application cannot start another request on the interface.</p>

Bit	Attr	Reset Value	Description
30	RW	0x0	RW Read/Write Indicator Indicates whether a read or write register transfer must be performed on the interface. Read/write bursting is not supported for registers. 1'b1: Read 1'b0: Write
29	RO	0x0	reserved
28	RW	0x1	I2CDatSe0 I2C DatSe0 USB Mode Selects the FS interface USB mode. 1'b1: VP_VM USB mode 1'b0: DAT_SE0 USB mode
27:26	RW	0x0	I2CDevAdr I2C Device Address Selects the address of the I2C Slave on the USB 1.1 full-speed serial transceiver that the core uses for OTG signaling. 2'b00: 7'h2C 2'b01: 7'h2D 2'b10: 7'h2E 2'b11: 7'h2F
25	RW	0x0	I2CSuspCtl I2C Suspend Control Selects how Suspend is connected to a full-speed transceiver in I2C mode. 1'b0: Use the dedicated utmi_suspend_n pin 1'b1: Use an I2C write to program the Suspend bit in the PHY register
24	RO	0x1	Ack I2C ACK Indicates whether an ACK response was received from the I2C Slave. This bit is valid when BsyDne is cleared by the core, after application has initiated an I2C access. 1'b0: NAK 1'b1: ACK
23	RW	0x0	I2CEn I2C Enable Enables the I2C Master to initiate I2C transactions on the I2C interface
22:16	RW	0x00	Addr I2C Address This is the 7-bit I2C device address used by software to access any external I2C Slave, including the I2C Slave on a USB 1.1 OTG full-speed serial transceiver. Software can change this address to access different I2C Slaves.

Bit	Attr	Reset Value	Description
15:8	RW	0x00	RegAddr I2C Register Addr This field programs the address of the register to be read from or written to.
7:0	RW	0x00	RWData I2C Read/Write Data After a register read operation, this field holds the read data for the application. During a write operation, the application can use this register to program the write data to be written to a register. During writes, this field holds the write data.

USBOTG_GPVNDCTL

Address: Operational Base + offset (0x0034)

PHY Vendor Control Register

Bit	Attr	Reset Value	Description
31	R/W SC	0x0	DisUlpiDrv Disable ULPI Drivers This field is used only if the Carkit interface was enabled in coreConsultant (parameter OTG_ULPI_CARKIT = 1). Otherwise, reads return 0. The application sets this bit when it has finished processing the ULPI Carkit Interrupt (GINTSTS.ULPICKINT). When set, the DWC_otg core disables drivers for output signals and masks input signal for the ULPI interface. DWC_otg clears this bit before enabling the ULPI interface.
30:28	RO	0x0	reserved
27	R/W SC	0x0	VStsDone VStatus Done The core sets this bit when the vendor control access is done. This bit is cleared by the core when the application sets the New Register Request bit (bit 25).
26	RO	0x0	VStsBsy VStatus Busy The core sets this bit when the vendor control access is in progress and clears this bit when done.
25	R/W SC	0x0	NewRegReq New Register Request The application sets this bit for a new vendor control access.
24:23	RO	0x0	reserved
22	RW	0x0	RegWr Register Write Set this bit for register writes, and clear it for register reads.

Bit	Attr	Reset Value	Description
21:16	RW	0x00	RegAddr Register Address The 6-bit PHY register address for immediate PHY Register Set access. Set to 6'h2F for Extended PHY Register Set access.
15:8	RW	0x00	VCtrl UTMI+ Vendor Control Register Address The 4-bit register address a vendor defined 4-bit parallel output bus. Bits 11:8 of this field are placed on utmi_vcontrol[3:0]. ULPI Extended Register Address (ExtRegAddr) The 6-bit PHY extended register address.
7:0	RW	0x00	RegData Register Data Contains the write data for register write. Read data for register read, valid when VStatus Done is set.

USBOTG_GGPIO

Address: Operational Base + offset (0x0038)

General Purpose Input/Output Register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	GPO General Purpose Output This field is driven as an output from the core, gp_o[15:0]. The application can program this field to determine the corresponding value on the gp_o[15:0] output.
15:0	RO	0x0000	GPI General Purpose Input This field's read value reflects the gp_i[15:0] core input value.

USBOTG_GUID

Address: Operational Base + offset (0x003c)

User ID Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	UserID Application-programmable ID field.

USBOTG_GSNPSID

Address: Operational Base + offset (0x0040)

Core ID Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00004f54	CoreID Release number of the core being used

USBOTG_GHWCFG1

Address: Operational Base + offset (0x0044)

User HW Config1 Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	<p>epdir Endpoint Direction This 32-bit field uses two bits per endpoint to determine the endpoint direction.</p> <p>Endpoint Bits [31:30]: Endpoint 15 direction Bits [29:28]: Endpoint 14 direction ... Bits [3:2]: Endpoint 1 direction Bits[1:0]: Endpoint 0 direction (always BIDIR)</p> <p>Direction 2'b00: BIDIR (IN and OUT) endpoint 2'b01: IN endpoint 2'b10: OUT endpoint 2'b11: Reserved</p>

USBOTG_GHWCFG2

Address: Operational Base + offset (0x0048)

User HW Config2 Register

Bit	Attr	Reset Value	Description
31	RO	0x0	<p>OTG_ENABLE_IC_USB IC_USB mode specified for mode of operation (parameter OTG_ENABLE_IC_USB). To choose IC_USB_MODE, both OTG_FSPHY_INTERFACE and OTG_ENABLE_IC_USB must be 1.</p>
30:26	RO	0x00	<p>TknQDepth Device Mode IN Token Sequence Learning Queue Depth Range: 0-30</p>
25:24	RO	0x0	<p>PTxQDepth Host Mode Periodic Request Queue Depth 2'b00: 2 2'b01: 4 2'b10: 8 Others: Reserved</p>
23:22	RO	0x0	<p>NPTxQDepth Non-periodic Request Queue Depth 2'b00: 2 2'b01: 4 2'b10: 8 Others: Reserved</p>
21	RO	0x0	reserved

Bit	Attr	Reset Value	Description
20	RO	0x0	MultiProcIntrpt Multi Processor Interrupt Enabled 1'b0: No 1'b1: Yes
19	RO	0x0	DynFifoSizing Dynamic FIFO Sizing Enabled 1'b0: No 1'b1: Yes
18	RO	0x0	PerioSupport Periodic OUT Channels Supported in Host Mode 1'b0: No 1'b1: Yes
17:14	RO	0x0	NumHstChnl Number of Host Channels Indicates the number of host channels supported by the core in Host mode. The range of this field is 0-15: 0 specifies 1 channel, 15 specifies 16 channels.
13:10	RO	0x0	NumDevEps Number of Device Endpoints Indicates the number of device endpoints supported by the core in Device mode in addition to control endpoint 0. The range of this field is 1-15.
9:8	RO	0x0	FSPhyType Full-Speed PHY Interface Type 2'b00: Full-speed interface not supported 2'b01: Dedicated full-speed interface 2'b10: FS pins shared with UTMI+ pins 2'b11: FS pins shared with ULPI pins
7:6	RO	0x0	HSPhyType High-Speed PHY Interface Type 2'b00: High-Speed interface not supported 2'b01: UTMI+ 2'b10: ULPI 2'b11: UTMI+ and ULPI
5	RO	0x0	SingPnt Point-to-Point 1'b0: Multi-point application (hub and split support) 1'b1: Single-point application (no hub and no split support)
4:3	RO	0x0	OtgArch Architecture 2'b00: Slave-Only 2'b01: External DMA 2'b10: Internal DMA Others: Reserved

Bit	Attr	Reset Value	Description
2:0	RO	0x0	<p>OtgMode Mode of Operation</p> <p>3'b000: HNP- and SRP-Capable OTG (Host and Device)</p> <p>3'b001: SRP-Capable OTG (Host and Device)</p> <p>3'b010: Non-HNP and Non-SRP Capable OTG (Host and Device)</p> <p>3'b011: SRP-Capable Device</p> <p>3'b100: Non-OTG Device</p> <p>3'b101: SRP-Capable Host</p> <p>3'b110: Non-OTG Host</p> <p>Others: Reserved</p>

USBOTG_GHWCFG3

Address: Operational Base + offset (0x004c)

User HW Config3 Register

Bit	Attr	Reset Value	Description
31:16	RO	0x0000	<p>DfifoDepth DFIFO Depth</p> <p>This value is in terms of 32-bit words.</p> <p>Minimum value is 32</p> <p>Maximum value is 32,768</p>
15	RO	0x0	<p>OTG_ENABLE_LPM</p> <p>LPM mode specified for Mode of Operation (parameter OTG_ENABLE_LPM).</p>
14	RO	0x0	<p>OTG_BC_SUPPORT</p> <p>This bit indicates the HS OTG controller support for Battery Charger.</p> <p>0 : No Battery Charger Support</p> <p>1 : Battery Charger support present.</p>
13	RO	0x0	<p>OTG_ENABLE_HSIC</p> <p>HSIC mode specified for Mode of Operation (parameter OTG_ENABLE_HSIC).</p> <p>Value Range: 0-1</p> <p>1: HSIC-capable with shared UTMI PHY interface</p> <p>0: Non-HSIC-capable</p>
12	RO	0x0	<p>OTG_AD_P_SUPPORT</p> <p>This bit indicates whether ADP logic is present within or external to the HS OTG controller</p> <p>0: No ADP logic present with HSOTG controller</p> <p>1: ADP logic is present along with HSOTG controller.</p>
11	RO	0x0	<p>RstType</p> <p>Reset Style for Clocked always Blocks in RTL</p> <p>1'b0: Asynchronous reset is used in the core</p> <p>1'b1: Synchronous reset is used in the core</p>

Bit	Attr	Reset Value	Description
10	RO	0x0	OptFeature Optional Features Removed Indicates whether the User ID register, GPIO interface ports, and SOF toggle and counter ports were removed for gate count optimization by enabling Remove Optional Features? 1'b0: No 1'b1: Yes
9	RO	0x0	VndctlSupt Vendor Control Interface Support 1'b0: Vendor Control Interface is not available on the core. 1'b1: Vendor Control Interface is available.
8	RO	0x0	I2CIntSel I2C Selection 1'b0: I2C Interface is not available on the core. 1'b1: I2C Interface is available on the core.
7	RO	0x0	OtgEn OTG Function Enabled The application uses this bit to indicate the DWC_otg core's OTG capabilities. 1'b0: Not OTG capable 1'b1: OTG Capable
6:4	RO	0x0	PktSizeWidth Width of Packet Size Counters 3'b000: 4 bits 3'b001: 5 bits 3'b010: 6 bits 3'b011: 7 bits 3'b100: 8 bits 3'b101: 9 bits 3'b110: 10 bits Others: Reserved
3:0	RO	0x0	XferSizeWidth Width of Transfer Size Counters 4'b0000: 11 bits 4'b0001: 12 bits ... 4'b1000: 19 bits Others: Reserved

USBOTG_GHWCFG4

Address: Operational Base + offset (0x0050)

User HW Config4 Register

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

Bit	Attr	Reset Value	Description
31	RO	0x0	SGDMA Scatter/Gather DMA 1'b1: Dynamic configuration
30	RO	0x0	SGDMACon Scatter/Gather DMA configuration 1'b0: Non-Scatter/Gather DMA configuration 1'b1: Scatter/Gather DMA configuration
29:26	RO	0x0	INEndps Number of Device Mode IN Endpoints Including Control Endpoint Range 0 -15 0:1 IN Endpoint 1:2 IN Endpoints 15:16 IN Endpoints
25	RW	0x0	DedFifoMode Enable Dedicated Transmit FIFO for device IN Endpoints 1'b0: Dedicated Transmit FIFO Operation not enabled. 1'b1: Dedicated Transmit FIFO Operation enabled.
24	RW	0x0	SessEndFltr session_end Filter Enabled 1'b0: No filter 1'b1: Filter
23	RW	0x0	BValidFltr "b_valid" Filter Enabled 1'b0: No filter 1'b1: Filter
22	RO	0x0	AValidFltr "a_valid" Filter Enabled 1'b0: No filter 1'b1: Filter
21	RO	0x0	VBusValidFltr "vbus_valid" Filter Enabled 1'b0: No filter 1'b1: Filter
20	RO	0x0	IddgFltr "iddig" Filter Enable 1'b0: No filter 1'b1: Filter
19:16	RO	0x0	NumCtlEndps Number of Device Mode Control Endpoints in Addition to Endpoint Range: 0-15

Bit	Attr	Reset Value	Description
15:14	RO	0x0	<p>PhyDataWidth UTMI+ PHY/ULPI-to-Internal UTMI+ Wrapper Data Width When a ULPI PHY is used, an internal wrapper converts ULPI to UTMI+.</p> <p>2'b00: 8 bits 2'b01: 16 bits 2'b10: 8/16 bits, software selectable Others: Reserved</p>
13:7	RO	0x0	reserved
6	RO	0x0	<p>EnHiber Enable Hibernation 1'b0: Hibernation feature not enabled 1'b1: Hibernation feature enabled</p>
5	RO	0x0	<p>AhbFreq Minimum AHB Frequency Less Than 60 MHz 1'b0: No 1'b1: Yes</p>
4	RO	0x0	<p>EnParPwrDown Enable Partial Power Down 1'b0: Partial Power Down Not Enabled 1'b1: Partial Power Down Enabled</p>
3:0	RO	0x0	<p>NumDevPerioEps Number of Device Mode Periodic IN Endpoints Range: 0-15</p>

USBOTG_GLPMCFG

Address: Operational Base + offset (0x0054)

Core LPM Configuration Register

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

Bit	Attr	Reset Value	Description
31	RW	0x0	<p>InvSelHsic HSIC-Invert Select HSIC</p> <p>The application uses this bit to control the DWC_otg core HSIC enable/disable. This bit overrides and functionally inverts the if_sel_hsic input port signal. If the core operates as non-HSIC-capable, it can only connect to non-HSIC-capable PHYs. If the core operates as HSIC-capable, it can only connect to HSICcapable PHYs. If the if_sel_hsic input signal is 1:</p> <ul style="list-style-type: none"> 1'b1: HSIC capability is not enabled. 1'b0: HSIC capability is enabled, <p>If InvSelHsic = 1'b0: HSIC capability is enabled. If the if_sel_hsic input signal is 0:</p> <ul style="list-style-type: none"> 1'b1: HSIC capability is enabled, 1'b0: HSIC capability is not enabled. <p>This bit is writable only if an HSIC mode was specified for Mode of Operation (parameter OTG_ENABLE_HSIC). This bit is valid only if OTG_ENABLE_HSIC is enabled.</p>
30	RW	0x0	<p>HSICCon HSIC-Connect</p> <p>The application must use this bit to initiate the HSIC Attach sequence. Host Mode: Once this bit is set, the host core configures to drive the HSIC Idle state (STROBE = 1 & DATA = 0) on the bus. It then waits for the device to initiate the Connect sequence. Device Mode: Once this bit is set, the device core waits for the HSIC Idle line state on the bus. Upon receiving the Idle line state, it initiates the HSIC Connect sequence. This bit is valid only if OTG_ENABLE_HSIC is 1, if_sel_hsic = 1 and InvSelHSIC is 0. Otherwise, it is read-only.</p>
29:28	RO	0x0	reserved
27:25	RO	0x0	<p>LPM_RetryCnt_Sts LPM Retry Count Status</p> <p>Number of LPM host retries remaining to be transmitted for the current LPM sequence.</p>
24	RW	0x0	<p>SndLPM Send LPM Transaction</p> <p>Host Mode: When the application software sets this bit, an LPM transaction containing two tokens, EXT and LPM, is sent. The hardware clears this bit once a valid response (STALL, NYET, or ACK) is received from the device or the core has finished transmitting the programmed number of LPM retries. Note: This bit must only be set when the host is connected to a local port.</p>
23:21	R/W SC	0x0	<p>LPM_Retry_Cnt LPM Retry Count</p> <p>When the device gives an ERROR response, this is the number of additional LPM retries that the host performs until a valid device response (STALL, NYET, or ACK) is received.</p>

Bit	Attr	Reset Value	Description
20:17	RW	0x0	<p>LPM_Chnl_Indx LPM Channel Index The channel number on which the LPM transaction must be applied while sending an LPM transaction to the local device. Based on the LPM channel index, the core automatically inserts the device address and endpoint number programmed in the corresponding channel into the LPM transaction.</p>
16	RO	0x0	<p>L1ResumeOK Sleep State Resume OK Indicates that the application or host can start a resume from the Sleep state. This bit is valid in the LPM Sleep (L1) state. It is set in Sleep mode after a delay of 50 us (TL1Residency). The bit is reset when SlpSts is 0 1'b1: The application/core can start resume from the Sleep state 1'b0: The application/core cannot start resume from the Sleep state</p>
15	RO	0x0	<p>SlpSts Port Sleep Status Device Mode: This bit is set as long as a Sleep condition is present on the USB bus. The core enters the Sleep state when an ACK response is sent to an LPM transaction and the timer TL1TokenRetry. has expired. To stop the PHY clock, the application must set the Port Clock Stop bit, which asserts the PHY Suspend input pin. The application must rely on SlpSts and not ACK in CoreL1Res to confirm transition into sleep. The core comes out of sleep: When there is any activity on the USB line_state When the application writes to the Remote Wakeup Signaling bit in the Device Control register (DCTL.RmtWkUpSig) or when the application resets or soft-disconnects the device. Host Mode: The host transitions to the Sleep (L1) state as a side effect of a successful LPM transaction by the core to the local port with an ACK response from the device. The read value of this bit reflects the port's current sleep status. The core clears this bit after: The core detects a remote L1 Wakeup signal The application sets the Port Reset bit or the Port L1Resume bit in the HPRT register or The application sets the L1Resume/ Remote Wakeup Detected Interrupt bit or Disconnect Detected Interrupt bit in the Core Interrupt register (GINTSTS.L1WkUpInt or GINTSTS.DisconnInt, respectively). Values: 1'b0: Core not in L1 1'b1: Core in L1</p>

Bit	Attr	Reset Value	Description																																																			
14:13	RO	0x0	<p>CoreL1Res LPM Response Device Mode: The core's response to the received LPM transaction is reflected in these two bits. Host Mode: The handshake response received from the local device for LPM transaction.</p> <p>11: ACK 10: NYET 01: STALL 00: ERROR (No handshake response)</p>																																																			
12:8	RW	0x00	<p>HIRD_Thres HIRD Threshold Device Mode: The core asserts L1SuspendM to put the PHY into Deep Low-Power mode in L1 when the HIRD value is greater than or equal to the value defined in this field (GLPMCFG.HIRD_Thres[3:0]), and HIRD_Thres[4] is set to 1'b1. Host Mode: The core asserts L1SuspendM to put the PHY into Deep Low-Power mode in L1 when HIRD_Thres[4] is set to 1'b1. HIRD_Thres[3:0] specifies the time for which resume signaling is to be reflected by the host TL1HubDrvResume2) on the USB when it detects device-initiated resume. HIRD_Thres must not be programmed with a value greater than 4'b1100 in Host mode, because this exceeds maximum TL1HubDrvResume2.</p> <table> <thead> <tr> <th>SI. No</th> <th>HIRD_Thres[3:0]</th> <th>Host mode resume time(us)</th> </tr> </thead> <tbody> <tr><td>1</td><td>4'b0000</td><td>60</td></tr> <tr><td>2</td><td>4'b0001</td><td>135</td></tr> <tr><td>3</td><td>4'b0010</td><td>210</td></tr> <tr><td>4</td><td>4'b0011</td><td>285</td></tr> <tr><td>5</td><td>4'b0100</td><td>360</td></tr> <tr><td>6</td><td>4'b0101</td><td>435</td></tr> <tr><td>7</td><td>4'b0110</td><td>510</td></tr> <tr><td>8</td><td>4'b0111</td><td>585</td></tr> <tr><td>9</td><td>4'b1000</td><td>660</td></tr> <tr><td>10</td><td>4'b1001</td><td>735</td></tr> <tr><td>11</td><td>4'b1010</td><td>810</td></tr> <tr><td>12</td><td>4'b1011</td><td>885</td></tr> <tr><td>13</td><td>4'b1100</td><td>960</td></tr> <tr><td>14</td><td>4'b1101</td><td>invalid</td></tr> <tr><td>15</td><td>4'b1110</td><td>invalid</td></tr> <tr><td>16</td><td>4'b1111</td><td>invalid</td></tr> </tbody> </table>	SI. No	HIRD_Thres[3:0]	Host mode resume time(us)	1	4'b0000	60	2	4'b0001	135	3	4'b0010	210	4	4'b0011	285	5	4'b0100	360	6	4'b0101	435	7	4'b0110	510	8	4'b0111	585	9	4'b1000	660	10	4'b1001	735	11	4'b1010	810	12	4'b1011	885	13	4'b1100	960	14	4'b1101	invalid	15	4'b1110	invalid	16	4'b1111	invalid
SI. No	HIRD_Thres[3:0]	Host mode resume time(us)																																																				
1	4'b0000	60																																																				
2	4'b0001	135																																																				
3	4'b0010	210																																																				
4	4'b0011	285																																																				
5	4'b0100	360																																																				
6	4'b0101	435																																																				
7	4'b0110	510																																																				
8	4'b0111	585																																																				
9	4'b1000	660																																																				
10	4'b1001	735																																																				
11	4'b1010	810																																																				
12	4'b1011	885																																																				
13	4'b1100	960																																																				
14	4'b1101	invalid																																																				
15	4'b1110	invalid																																																				
16	4'b1111	invalid																																																				

Bit	Attr	Reset Value	Description
7	RW	0x0	<p>EnbISlpM Enable utmi_sleep_n For ULPI interface: The application uses this bit to write to the function control [7] in the L1 state, to enable the PHY to go into Low Power mode. For the host, this bit is valid only in Local Device mode.</p> <p>1'b0: Writes to the ULPI Function Control Bit[7] are disabled. 1'b1: The core is enabled to write to the ULPI Function Control Bit[7], which enables the PHY to enter Low-Power mode.</p> <p>Note: When a ULPI interface is configured, enabling this bit results in a write to Bit 7 of the ULPI Function Control register. The ULPI PHY supports writing to this bit, and in the L1 state asserts SleepM when utmi_l1_suspend_n cannot be asserted. When a ULPI interface is configured, this bit must always be set if you are using the ULPI PHY. Note: For ULPI interfaces, do not clear this bit during the resume. For all other interfaces: The application uses this bit to control utmi_sleep_n assertion to the PHY in the L1 state. For the host, this bit is valid only in Local Device mode.</p> <p>1'b0: utmi_sleep_n assertion from the core is not transferred to the external PHY. 1'b1: utmi_sleep_n assertion from the core is transferred to the external PHY when utmi_l1_suspend_n cannot be asserted.</p>
6	RW	0x0	<p>bRemoteWake RemoteWakeEnable Host Mode: The remote wakeup value to be sent in the LPM transaction's wIndex field. Device Mode: This field is updated with the received bRemoteWake LPM token's bmAttribute when an ACK/NYET/STALL response is sent to an LPM transaction.</p>

Bit	Attr	Reset Value	Description																																																			
5:2	RW	0x0	<p>HIRD Host-Initiated Resume Duration Host Mode: The value of HIRD to be sent in an LPM transaction. This value is also used to initiate resume for a duration TL1HubDrvResume1 for host initiated resume. Device Mode: This field is updated with the Received LPM Token HIRD bmAttribute when an ACK/NYET/STALL response is sent to an LPM transaction</p> <table> <thead> <tr> <th>Sl. No</th> <th>HIRD[3:0]</th> <th>THIRD (us)</th> </tr> </thead> <tbody> <tr><td>1</td><td>4'b0000</td><td>50</td></tr> <tr><td>2</td><td>4'b0001</td><td>125</td></tr> <tr><td>3</td><td>4'b0010</td><td>200</td></tr> <tr><td>4</td><td>4'b0011</td><td>275</td></tr> <tr><td>5</td><td>4'b0100</td><td>350</td></tr> <tr><td>6</td><td>4'b0101</td><td>425</td></tr> <tr><td>7</td><td>4'b0110</td><td>500</td></tr> <tr><td>8</td><td>4'b0111</td><td>575</td></tr> <tr><td>9</td><td>4'b1000</td><td>650</td></tr> <tr><td>10</td><td>4'b1001</td><td>725</td></tr> <tr><td>11</td><td>4'b1010</td><td>800</td></tr> <tr><td>12</td><td>4'b1011</td><td>875</td></tr> <tr><td>13</td><td>4'b1100</td><td>950</td></tr> <tr><td>14</td><td>4'b1101</td><td>1025</td></tr> <tr><td>15</td><td>4'b1110</td><td>1100</td></tr> <tr><td>16</td><td>4'b1111</td><td>1175</td></tr> </tbody> </table>	Sl. No	HIRD[3:0]	THIRD (us)	1	4'b0000	50	2	4'b0001	125	3	4'b0010	200	4	4'b0011	275	5	4'b0100	350	6	4'b0101	425	7	4'b0110	500	8	4'b0111	575	9	4'b1000	650	10	4'b1001	725	11	4'b1010	800	12	4'b1011	875	13	4'b1100	950	14	4'b1101	1025	15	4'b1110	1100	16	4'b1111	1175
Sl. No	HIRD[3:0]	THIRD (us)																																																				
1	4'b0000	50																																																				
2	4'b0001	125																																																				
3	4'b0010	200																																																				
4	4'b0011	275																																																				
5	4'b0100	350																																																				
6	4'b0101	425																																																				
7	4'b0110	500																																																				
8	4'b0111	575																																																				
9	4'b1000	650																																																				
10	4'b1001	725																																																				
11	4'b1010	800																																																				
12	4'b1011	875																																																				
13	4'b1100	950																																																				
14	4'b1101	1025																																																				
15	4'b1110	1100																																																				
16	4'b1111	1175																																																				
1	RW	0x0	<p>AppL1Res LPM response programmed by application Handshake response to LPM token pre-programmed by device application software. The response depends on GLPMCFG.LPMCap. If GLPMCFG.LPMCap is 1'b0, the core always responds with a NYET. If GLPMCFG.LPMCap is 1'b1, the core responds as follows: 1: ACK. Even though an ACK is pre-programmed, the core responds with an ACK only on a successful LPM transaction. The LPM transaction is successful if: There are no PID/CRC5 errors in both the EXT token and the LPM token (else ERROR); A valid bLinkState = 0001B (L1) is received in the LPM transaction (else STALL); No data is pending in the Transmit queue (else NYET) 0: NYET. The pre-programmed software bit is overridden for response to LPM token when:(1)The received bLinkState is not L1 (STALL response); (2)An error is detected in either of the LPM token packets due to corruption (ERROR response).</p>																																																			

Bit	Attr	Reset Value	Description
0	RW	0x0	<p>LPMCap LPM-Capable</p> <p>The application uses this bit to control the DWC_otg core LPM capabilities. If the core operates as a non-LPM-capable host, it cannot request the connected device/hub to activate LPM mode. If the core operates as a non-LPM-capable device, it cannot respond to any LPM transactions.</p> <p>1'b0: LPM capability is not enabled. 1'b1: LPM capability is enabled.</p> <p>This bit is writable only if an LPM mode was specified for Mode of Operation (parameter OTG_ENABLE_LPM). Otherwise, reads return 0.</p>

USBOTG_GPWRDN

Address: Operational Base + offset (0x0058)

Global Power Down Register

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:24	RO	0x00	<p>MultValIdBC Multi Valued ID pin Battery Charger ACA inputs in the following order: Bit 26 - rid_float. Bit 25 - rid_gnd Bit 24 - rid_a Bit 23 - rid_b Bit 22 - rid_c</p> <p>These bits are present only if OTG_BC_SUPPORT = 1. Otherwise, these bits are reserved and will read 5'h0.</p>
23	W1C	0x0	<p>ADPInt ADP Interupt This bit is set whenever there is a ADP event.</p>
22	RO	0x0	<p>BSessVld B Session Valid This field reflects the B session valid status signal from the PHY. 1'b0: B-Valid is 0. 1'b1: B-Valid is 1. This bit is valid only when GPWRDN.PMUActv is 1.</p>
21	RO	0x0	<p>IDDIG This bit indicates the status of the signal IDDIG. The application must read this bit after receiving GPWRDN.StsChngInt and decode based on the previous value stored by the application. Indicates the current mode. 1'b1: Device mode 1'b0: Host mode This bit is valid only when GPWRDN.PMUActv is 1.</p>

Bit	Attr	Reset Value	Description
20:19	RO	0x0	<p>LineState This field indicates the current linestate on USB as seen by the PMU module.</p> <p>2'b00: DM = 0, DP = 0. 2'b01: DM = 0, DP = 1. 2'b10: DM = 1, DP = 0. 2'b11: Not-defined.</p> <p>This bit is valid only when GPWRDN.PMUActv is 1.</p>
18	RW	0x0	<p>StsChngIntMsk Mask For StsChng Interrupt</p>
17	W1C	0x0	<p>StsChngInt This field indicates a status change in either the IDDIG or BSessVld signal.</p> <p>1'b0: No Status change 1'b1: status change detected</p> <p>After receiving this interrupt the application should read the GPWRDN register and interpret the change in IDDIG or BSesVld with respect to the previous value stored by the application.</p>
16	RW	0x0	<p>SRPDetectMsk Mask For SRPDetect Interrupt</p>
15	W1C	0x0	<p>SRPDetect This field indicates that SRP has been detected by the PMU. This field generates an interrupt. After detecting SRP during hibernation the application should not restore the core. The application should get into the initialization process.</p> <p>1'b0: SRP not detected 1'b1: SRP detected</p>
14	RW	0x0	<p>ConnDetMsk Mask for ConnectDet interrupt</p> <p>This bit is valid only when OTG_EN_PWRLOPT = 2.</p>
13	W1C	0x0	<p>ConnectDet This field indicates that a new connect has been detected</p> <p>1'b0: Connect not detected 1'b1: Connect detected</p> <p>This bit is valid only when OTG_EN_PWRLOPT = 2.</p>
12	RW	0x0	<p>DisconnectDetectMsk Mask For DisconnectDetect Interrupt</p> <p>This bit is valid only when OTG_EN_PWRLOPT = 2.</p>

Bit	Attr	Reset Value	Description
11	W1C	0x0	<p>DisconnectDetect</p> <p>This field indicates that Disconnect has been detected by the PMU. This field generates an interrupt. After detecting disconnect during hibernation the application must not restore the core, but instead start the initialization process.</p> <p>1'b0: Disconnect not detected 1'b1: Disconnect detected</p> <p>This bit is valid only when OTG_EN_PWR0PT = 2.</p>
10	RW	0x0	<p>ResetDetMsk</p> <p>Mask For ResetDetected interrupt. This bit is valid only when OTG_EN_PWR0PT = 2.</p>
9	W1C	0x0	<p>ResetDetected</p> <p>This field indicates that Reset has been detected by the PMU module. This field generates an interrupt.</p> <p>1'b0: Reset Not Detected 1'b1: Reset Detected</p> <p>This bit is valid only when OTG_EN_PWR0PT = 2.</p>
8	RW	0x0	<p>LineStageChangeMsk</p> <p>Mask For LineStateChange interrupt</p> <p>This bit is valid only when OTG_EN_PWR0PT = 2.</p>
7	W1C	0x0	<p>LnStsChng</p> <p>Line State Change</p> <p>This interrupt is asserted when there is a Linestate Change detected by the PMU. The application should read GPWRDN.Linestate to determine the current linestate on USB.</p> <p>1'b0: No LineState change on USB 1'b1: LineState change on USB</p> <p>This bit is valid only when GPWRDN.PMUActv is 1. This bit is valid only when OTG_EN_PWR0PT = 2.</p>
6	RW	0x0	<p>DisableVBUS</p> <p>The application should program this bit if HPRT0.PrtPwr was programmed to 0 before entering Hibernation. This is to indicate PMU whether session was ended before entering Hibernation.</p> <p>1'b0: HPRT0.PrtPwr was not programed to 0. 1'b1: HPRT0.PrtPwr was programmed to 0.</p>
5	RW	0x0	<p>PwrDnSwtch</p> <p>Power Down Switch</p> <p>This bit indicates to the DWC_otg core VDD switch is in ON/OFF state</p> <p>1'b0: DWC_otg is in ON state 1'b1: DWC_otg is in OFF state</p> <p><i>Note: This bit must not be written to during normal mode of operation.</i></p>

Bit	Attr	Reset Value	Description
4	RW	0x0	<p>PwrDnRst_n Power Down ResetN The application must program this bit to reset the DWC OTG core during the Hibernation exit process or during ADP when powering up the core (in case the DWC OTG core was powered off during ADP process). 1'b1: DWC_otg is in normal operation 1'b0: reset DWC_otg <i>Note: This bit must not be written to during normal mode of operation.</i></p>
3	RW	0x0	<p>PwrDnClmp Power Down Clamp The application must program this bit to enable or disable the clamps to all the outputs of the DWC OTG core module to prevent the corruption of other active logic. 1'b0: Disable PMU power clamp 1'b1: Enable PMU power clamp</p>
2	RW	0x0	<p>Restore The application should program this bit to enable or disable restore mode from the PMU module. 1'b0: DWC_otg in normal mode of operation 1'b1: DWC_otg in restore mode <i>Note: This bit must not be written to during normal mode of operation. This bit is valid only when OTG_EN_PWR_OPT = 2.</i></p>
1	RW	0x0	<p>PMUActv PMU Active This is bit is to enable or disable the PMU logic. 1'b0: Disable PMU module 1'b1: Enable PMU module <i>Note: This bit must not be written to during normal mode of operation.</i></p>
0	RW	0x0	<p>PMUIntSel PMU Interrupt Select When the hibernation functionality is selected using the configuration option OTG_EN_PWR_OPT = 2, a write to this bit with 1'b1 enables the PMU to generate interrupts to the application. During this state all interrupts from the core module are blocked to the application. Note: This bit must be set to 1'b1 before the core is put into hibernation 1'b0: Internal DWC_otg_core interrupt is selected 1'b1: the external DWC_otg_pmu interrupt is selected <i>Note: This bit must not be written to during normal mode of operation.</i></p>

USBOTG_GDFIFOCFG

Address: Operational Base + offset (0x005c)

Global DFIFO Software Config Register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	EPIInfoBaseAddr This field provides the start address of the EP info controller.
15:0	RW	0x0000	GDFIFOCfg This field is for dynamic programming of the DFIFO Size. This value takes effect only when the application programs a non zero value to this register. The core does not have any corrective logic if the FIFO sizes are programmed incorrectly.

USBOTG_GADPCTL

Address: Operational Base + offset (0x0060)

ADP Timer,Control and Status Register

Bit	Attr	Reset Value	Description
31:29	RO	0x0	reserved
28:27	R/W SC	0x0	AR Access Request 2'b00 Read/Write Valid (updated by the core) 2'b01 Read 2'b10 Write 2'b11 Reserved
26	RW	0x0	AdpTmoutMsk ADP Timeout Interrupt Mask When this bit is set, it unmasks the interrupt because of AdpTmoutInt. This bit is valid only if OTG_Ver = 1'b1(GOTGCTL[20]).
25	RW	0x0	AdpSnsIntMsk ADP Sense Interrupt Mask When this bit is set, it unmasks the interrupt due to AdpSnsInt. This bit is valid only if OTG_Ver = 1'b1(GOTGCTL[20]).
24	RW	0x0	AdpPrbIntMsk ADP Probe Interrupt Mask When this bit is set, it unmasks the interrupt due to AdpPrbInt. This bit is valid only if OTG_Ver = 1'b1(GOTGCTL[20]).
23	W1C	0x0	AdpTmoutInt ADP Timeout Interrupt This bit is relevant only for an ADP probe. When this bit is set, it means that the ramp time has completed (GADPCTL.RTIM has reached its terminal value of 0x7FF). This is a debug feature that allows software to read the ramp time after each cycle. This bit is valid only if OTG_Ver = 1'b1.

Bit	Attr	Reset Value	Description
22	W1C	0x0	<p>AdpSnsInt ADP Sense Interrupt When this bit is set, it means that the VBUS voltage is greater than VadpSns value or VadpSns is reached. This bit is valid only if OTG_Ver = 1'b1 (GOTGCTL[20]).</p>
21	W1C	0x0	<p>AdpPrbInt ADP Probe Interrupt When this bit is set, it means that the VBUS voltage is greater than VadpPrb or VadpPrb is reached. This bit is valid only if OTG_Ver = 1'b1 (GOTGCTL[20]).</p>
20	RW	0x0	<p>ADPEn ADP Enable When set, the core performs either ADP probing or sensing based on EnaPrb or EnaSns. This bit is valid only if OTG_Ver = 1'b1 (GOTGCTL[20]).</p>
19	R/W SC	0x0	<p>ADPRes ADP Reset When set, ADP controller is reset. This bit is auto-cleared after the reset procedure is complete in ADP controller. This bit is valid only if OTG_Ver = 1'b1 (GOTGCTL[20]).</p>
18	RW	0x0	<p>EnaSns Enable Sense When programmed to 1'b1, the core performs a sense operation. This bit is valid only if OTG_Ver = 1'b1 (GOTGCTL[20]).</p>
17	RW	0x0	<p>EnaPrb Enable Probe When programmed to 1'b1, the core performs a probe operation. This bit is valid only if OTG_Ver = 1'b1 (GOTGCTL[20]).</p>

Bit	Attr	Reset Value	Description
16:6	RO	0x000	<p>RTIM RAMP TIME</p> <p>These bits capture the latest time it took for VBUS to ramp from VADP_SINK to VADP_PRB. The bits are defined in units of 32 kHz clock cycles as follows:</p> <ul style="list-style-type: none"> 0x000 - 1 cycles 0x001 - 2 cycles 0x002 - 3 cycles and so on till 0x7FF - 2048 cycles <p>A time of 1024 cycles at 32 kHz corresponds to a time of 32 msec. (Note for scaledown ramp_timeout = prb_delta == 2'b00 => 200 cycles prb_delta == 2'b01 => 100 cycles prb_delta == 2'b01 => 50 cycles prb_delta == 2'b01 => 25 cycles.)</p>
5:4	RW	0x0	<p>PrbPer Probe Period</p> <p>These bits sets the TadpPrd as follows:</p> <ul style="list-style-type: none"> 2'b00 - 0.625 to 0.925 sec (typical 0.775 sec) 2'b01 - 1.25 to 1.85 sec (typical 1.55 sec) 2'b10 - 1.9 to 2.6 sec (typical 2.275 sec) 2'b11 - Reserved <p>(PRB_PER is also scaledown prb_per== 2'b00 => 400 ADP clocks prb_per== 2'b01 => 600 ADP clocks prb_per== 2'b10 => 800 ADP clocks prb_per==2'b11 => 1000 ADP clocks)</p>
3:2	RW	0x0	<p>PrbDelta Probe Delta</p> <p>These bits set the resolution for RTIM value. The bits are defined in units of 32 kHz clock cycles as follows:</p> <ul style="list-style-type: none"> 2'b00 - 1 cycles 2'b01 - 2 cycles 2'b10 - 3 cycles 2'b11 - 4 cycles <p>For example if this value is chosen to 2'b01, it means that RTIM increments for every three 32Khz clock cycles.</p>

Bit	Attr	Reset Value	Description
1:0	RW	0x0	<p>PrbDschg Probe Discharge</p> <p>These bits set the times for TadpDschg. These bits are defined as follows:</p> <ul style="list-style-type: none"> 2'b00 4 msec (Scaledown 2 32Khz clock cycles) 2'b01 8 msec (Scaledown 4 32Khz clock cycles) 2'b10 16 msec (Scaledown 8 32Khz clock cycles) 2'b11 32 msec (Scaledown 16 32Khz clock cycles)

USBOTG_HPTXFSIZ

Address: Operational Base + offset (0x0100)

Host Periodic Transmit FIFO Size Register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>PTxFSize Host Periodic TxFIFO Depth</p> <p>This value is in terms of 32-bit words.</p> <p>Minimum value is 16</p> <p>Maximum value is 32,768</p> <p>The power-on reset value of this register is specified as the Largest Host Mode Periodic Tx Data FIFO Depth (parameter OTG_TX_HPERIO_DFIFO_DEPTH). If Enable Dynamic FIFO Sizing? was deselected (parameter OTG_DFIFO_DYNAMIC = 0), these flops are optimized, and reads return the power-on value. If Enable Dynamic FIFO Sizing? was selected (parameter OTG_DFIFO_DYNAMIC = 1), you can write a new value in this field. Programmed values must not exceed the power-on value set .</p>
15:0	RW	0x0000	<p>PTxFSAddr Host Periodic TxFIFO Start Address</p> <p>The power-on reset value of this register is the sum of the Largest Rx Data FIFO Depth and Largest Non-periodic Tx Data FIFO Depth specified. These parameters are:</p> <p>In shared FIFO operation: OTG_RX_DFIFO_DEPTH + OTG_TX_NPERIO_DFIFO_DEPTH.</p> <p>In dedicated FIFO mode: OTG_RX_DFIFO_DEPTH + OTG_TX_HNPERIO_DFIFO_DEPTH.</p> <p>If Enable Dynamic FIFO Sizing? was deselected (parameter OTG_DFIFO_DYNAMIC = 0), these flops are optimized, and reads return the power-on value.</p> <p>If Enable Dynamic FIFO Sizing? was selected (parameter OTG_DFIFO_DYNAMIC = 1), you can write a new value in this field. Programmed values must not exceed the power-on value.</p>

USBOTG_DIEPTXFn

Address: Operational Base + offset (0x0104)

Device Periodic Transmit FIFO-n Size Register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	<p>INEP1TxDep IN Endpoint TxFIFO Depth This value is in terms of 32-bit words. Minimum value is 16 Maximum value is 32,768</p> <p>The power-on reset value of this register is specified as the Largest IN Endpoint FIFO number Depth (parameter OTG_TX_DINEP_DFIFO_DEPTH_n)(0 < n <= 15). If Enable Dynamic FIFO Sizing? was deselected (parameter OTG_DFIFO_DYNAMIC = 0), these flops are optimized, and reads return the power-on value.</p> <p>If Enable Dynamic FIFO Sizing? was selected (parameter OTG_DFIFO_DYNAMIC = 1), you can write a new value in this field. Programmed values must not exceed the power-on value .</p>
15:0	RW	0x0000	<p>INEP1TxFStAddr IN Endpoint FIFO1 Transmit RAM Start Address This field contains the memory start address for IN endpoint Transmit FIFOOn (0 < n <= 15). The power-on reset value of this register is specified as the Largest Rx Data FIFO Depth (parameter OTG_RX_DFIFO_DEPTH). OTG_RX_DFIFO_DEPTH + SUM 0 to n-1 (OTG_DINEP_TXFIFO_DEPTH_n)</p> <p>For example start address of IN endpoint FIFO 1 is OTG_RX_DFIFO_DEPTH + OTG_DINEP_TXFIFO_DEPTH_0. The start address of IN endpoint FIFO 2 is OTG_RX_DFIFO_DEPTH + OTG_DINEP_TXFIFO_DEPTH_0 + OTG_DINEP_TXFIFO_DEPTH_1. If Enable Dynamic FIFO Sizing? was deselected (parameter OTG_DFIFO_DYNAMIC = 0), these flops are optimized, and reads return the power-on value. If Enable Dynamic FIFO Sizing? was selected (parameter OTG_DFIFO_DYNAMIC = 1), and you have programmed a new value for RxFIFO depth, you can write that value in this field. Programmed values must not exceed the power-on value set .</p>

USBOTG_HCFG

Address: Operational Base + offset (0x0400)

Host Configuration Register

Bit	Attr	Reset Value	Description
31:27	RO	0x0	reserved

Bit	Attr	Reset Value	Description
26	RW	0x0	<p>PerSchedEna Enable Periodic Scheduling Applicable in Scatter/Gather DMA mode only. Enables periodic scheduling within the core. Initially, the bit is reset. The core will not process any periodic channels. As soon as this bit is set, the core will get ready to start scheduling periodic channels and sets HCFG.PerSchedStat. The setting of HCFG.PerSchedStat indicates the core has enabled periodic scheduling. Once HCFG.PerSchedEna is set, the application is not supposed to again reset the bit unless HCFG.PerSchedStat is set. As soon as this bit is reset, the core will get ready to stop scheduling periodic channels and resets HCFG.PerSchedStat. In non Scatter/Gather DMA mode, this bit is reserved.</p>
25:24	RW	0x0	<p>FrListEn Frame List Entries The value in the register specifies the number of entries in the Frame list. This field is valid only in Scatter/Gather DMA mode.</p>
23	RW	0x0	<p>DescDMA Enable Scatter/gather DMA in Host mode When the Scatter/Gather DMA option selected during configuration of the RTL, the application can set this bit during initialization to enable the Scatter/Gather DMA operation. NOTE: This bit must be modified only once after a reset. The following combinations are available for programming: GAHBCFG.DMAEn=0, HCFG.DescDMA=0 => Slave mode GAHBCFG.DMAEn=0, HCFG.DescDMA=1 => Invalid GAHBCFG.DMAEn=1, HCFG.DescDMA=0 => Buffered DMA mode GAHBCFG.DMAEn=1, HCFG.DescDMA=1 => Scatter/Gather DMA mode In non Scatter/Gather DMA mode, this bit is reserved.</p>
22:16	RO	0x0	reserved
15:8	RW	0x00	<p>ResValid Resume Validation Period This field is effective only when HCFG.Ena32KHzS is set. It controls the resume period when the core resumes from suspend. The core counts the ResValid number of clock cycles to detect a valid resume when this is set.</p>
7	RW	0x0	<p>Ena32KHzS Enable 32-KHz Suspend Mode This bit can only be set if the USB 1.1 Full-Speed Serial Transceiver Interface has been selected. If USB 1.1 Full-Speed Serial Transceiver Interface has not been selected, this bit must be zero. When the USB 1.1 Full-Speed Serial Transceiver Interface is chosen and this bit is set, the core expects the 48-MHz PHY clock to be switched to 32 KHz during a suspend.</p>

Bit	Attr	Reset Value	Description
6:3	RO	0x0	reserved
2	RW	0x0	<p>FSLSSupp FS- and LS-Only Support The application uses this bit to control the core enumeration speed. Using this bit, the application can make the core enumerate as a FS host, even if the connected device supports HS traffic. Do not make changes to this field after initial programming.</p> <p>1'b0: HS/FS/LS, based on the maximum speed supported by the connected device 1'b1: FS/LS-only, even if the connected device can support HS</p>
1:0	RW	0x0	<p>FSLSPclkSel FS/LS PHY Clock Select 2'b00: PHY clock is running at 30/60 MHz 2'b01: PHY clock is running at 48 MHz Others: Reserved</p>

USBOTG_HFIR

Address: Operational Base + offset (0x0404)

Host Frame Interval Register

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0000	<p>FrInt Frame Interval The value that the application programs to this field specifies the interval between two consecutive SOFs (FS) or micro-SOFs (HS) or Keep-Alive tokens (HS). This field contains the number of PHY clocks that constitute the required frame interval. The default value set in this field for a FS operation when the PHY clock frequency is 60 MHz. The application can write a value to this register only after the Port Enable bit of the Host Port Control and Status register (HPRT.PrtEnaPort) has been set. If no value is programmed, the core calculates the value based on the PHY clock specified in the FS/LS PHY Clock Select field of the Host Configuration register (HCFG.FSLSPclkSel). Do not change the value of this field after the initial configuration.</p> <p>125 us * (PHY clock frequency for HS) 1 ms * (PHY clock frequency for FS/LS)</p>

USBOTG_HFNUM

Address: Operational Base + offset (0x0408)

Host Frame Number/Frame Time Remaining Register

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

Bit	Attr	Reset Value	Description
31:16	RO	0x0000	FrRem Frame Time Remaining Indicates the amount of time remaining in the current microframe (HS) or frame (FS/LS), in terms of PHY clocks. This field decrements on each PHY clock. When it reaches zero, this field is reloaded with the value in the Frame Interval register and a new SOF is transmitted on the USB.
15:0	RO	0xfffff	FrNum Frame Number This field increments when a new SOF is transmitted on the USB, and is reset to 0 when it reaches 16'h3FFF. This field is writable only if Remove Optional Features? was not selected (OTG_RM_OTG_FEATURES = 0). Otherwise, reads return the frame number value.

USBOTG_HPTXSTS

Address: Operational Base + offset (0x0410)
 Host Periodic Transmit FIFO/Queue Status Register

Bit	Attr	Reset Value	Description
31:24	RO	0x00	PTxQTop Top of the Periodic Transmit Request Queue This indicates the entry in the Periodic Tx Request Queue that is currently being processed by the MAC. This register is used for debugging. Bit [31]: Odd/Even (micro)frame 1'b0: send in even (micro)frame 1'b1: send in odd (micro)frame Bits [30:27]: Channel/endpoint number Bits [26:25]: Type 2'b00: IN/OUT 2'b01: Zero-length packet 2'b10: CSPLIT 2'b11: Disable channel command Bit [24]: Terminate (last entry for the selected channel/endpoint)
23:16	RO	0x00	PTxQSpAvail Periodic Transmit Request Queue Space Available Indicates the number of free locations available to be written in the Periodic Transmit Request Queue. This queue holds both IN and OUT requests. 8'h0: Periodic Transmit Request Queue is full 8'h1: 1 location available 8'h2: 2 locations available n: n locations available (0 <=n <= 16) Others: Reserved

Bit	Attr	Reset Value	Description
15:0	RW	0x0000	PTxFSpAvail Periodic Transmit Data FIFO Space Available Indicates the number of free locations available to be written to in the Periodic TxFIFO. Values are in terms of 32-bit words 16'h0: Periodic TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 . n . 32,768) 16'h8000: 32,768 words available Others: Reserved

USBOTG_HAINT

Address: Operational Base + offset (0x0414)

Host All Channels Interrupt Register

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RO	0x0000	HAINT Channel Interrupts One bit per channel: Bit 0 for Channel 0, bit 15 for Channel 15

USBOTG_HINTMSK

Address: Operational Base + offset (0x0418)

Host All Channels Interrupt Mask Register

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0000	HAINTMsk Channel Interrupt Mask One bit per channel: Bit 0 for channel 0, bit 15 for channel 15

USBOTG_HPRT

Address: Operational Base + offset (0x0440)

Host Port Control and Status Register

Bit	Attr	Reset Value	Description
31:19	RO	0x0	reserved
18:17	RO	0x0	PrtSpd Port Speed Indicates the speed of the device attached to this port. 2'b00: High speed 2'b01: Full speed 2'b10: Low speed 2'b11: Reserved

Bit	Attr	Reset Value	Description
16:13	RW	0x0	<p>PrtTstCtl Port Test Control The application writes a nonzero value to this field to put the port into a Test mode, and the corresponding pattern is signaled on the port.</p> <p>4'b0000: Test mode disabled 4'b0001: Test_J mode 4'b0010: Test_K mode 4'b0011: Test_SE0_NAK mode 4'b0100: Test_Packet mode 4'b0101: Test_Force_Enable Others: Reserved</p>
12	R/W SC	0x0	<p>PrtPwr Port Power The application uses this field to control power to this port (write 1'b1 to set to 1'b1 and write 1'b0 to set to 1'b0), and the core can clear this bit on an over current condition.</p> <p>1'b0: Power off 1'b1: Power on</p>
11:10	RO	0x0	<p>PrtLnSts Port Line Status Indicates the current logic level USB data lines Bit [10]: Logic level of D+ Bit [11]: Logic level of D</p>
9	RO	0x0	reserved
8	RW	0x0	<p>PrtRst Port Reset When the application sets this bit, a reset sequence is started on this port. The application must time the reset period and clear this bit after the reset sequence is complete.</p> <p>1'b0: Port not in reset 1'b1: Port in reset</p> <p>To start a reset on the port, the application must leave this bit set for at least the minimum duration mentioned below, as specified in the USB 2.0 specification, Section 7.1.7.5. The application can leave it set for another 10 ms in addition to the required minimum duration, before clearing the bit, even though there is no maximum limit set by the USB standard.</p> <p>High speed: 50 ms Full speed/Low speed: 10 ms</p>

Bit	Attr	Reset Value	Description
7	R/W SC	0x0	<p>PrtSusp Port Suspend</p> <p>The application sets this bit to put this port in Suspend mode. The core only stops sending SOFs when this is set. To stop the PHY clock, the application must set the Port Clock Stop bit, which asserts the suspend input pin of the PHY.</p> <p>The read value of this bit reflects the current suspend status of the port. This bit is cleared by the core after a remote wakeup signal is detected or the application sets the Port Reset bit or Port Resume bit in this register or the Resume/Remote Wakeup Detected Interrupt bit or Disconnect Detected Interrupt bit in the Core Interrupt register (GINTSTS.WkUpInt or GINTSTS.DisconnInt, respectively).</p> <p>1'b0: Port not in Suspend mode 1'b1: Port in Suspend mode</p>

Bit	Attr	Reset Value	Description
6	R/W SC	0x0	<p>PrtRes Port Resume The application sets this bit to drive resume signaling on the port. The core continues to drive the resume signal until the application clears this bit. If the core detects a USB remote wakeup sequence, as indicated by the Port Resume/Remote Wakeup Detected Interrupt bit of the Core Interrupt register (GINTSTS.WkUpInt), the core starts driving resume signaling without application intervention and clears this bit when it detects a disconnect condition. The read value of this bit indicates whether the core is currently driving resume signaling.</p> <p>1'b0: No resume driven 1'b1: Resume driven When LPM is enabled and the core is in the L1 (Sleep) state, setting this bit results in the following behavior: The core continues to drive the resume signal until a pre-determined time specified in the GLPMCFG.HIRD_Thres[3:0] field. If the core detects a USB remote wakeup sequence, as indicated by the Port L1 Resume/Remote L1 Wakeup Detected Interrupt bit of the Core Interrupt register (GINTSTS.L1WkUpInt), the core starts driving resume signaling without application intervention and clears this bit at the end of the resume. The read value of this bit indicates whether the core is currently driving resume signaling.</p> <p>1'b0: No resume driven 1'b1: Resume driven</p>
5	W1C	0x0	<p>PrtOvrCurrChng Port Overcurrent Change The core sets this bit when the status of the Port Overcurrent Active bit (bit 4) in this register changes.</p>
4	RO	0x0	<p>PrtOvrCurrAct Port Overcurrent Active Indicates the overcurrent condition of the port. 1'b0: No overcurrent condition 1'b1: Overcurrent condition</p>

Bit	Attr	Reset Value	Description
3	W1C	0x0	PrtEnChng Port Enable/Disable Change The core sets this bit when the status of the Port Enable bit [2] of this register changes.
2	W1C	0x0	PrtEna Port Enable A port is enabled only by the core after a reset sequence, and is disabled by an overcurrent condition, a disconnect condition, or by the application clearing this bit. The application cannot set this bit by a register write. It can only clear it to disable the port. This bit does not trigger any interrupt to the application. 1'b0: Port disabled 1'b1: Port enabled
1	W1C	0x0	PrtConnDet Port Connect Detected The core sets this bit when a device connection is detected to trigger an interrupt to the application using the Host Port Interrupt bit of the Core Interrupt register (GINTSTS.PrtInt). The application must write a 1 to this bit to clear the interrupt.
0	RO	0x0	PrtConnSts Port Connect Status 0: No device is attached to the port. 1: A device is attached to the port.

USBOTG_HCCHARn

Address: Operational Base + offset (0x0500)

Host Channel-n Characteristics Register

Bit	Attr	Reset Value	Description
31	R/W SC	0x0	ChEna Channel Enable When Scatter/Gather mode is enabled 1'b0: Indicates that the descriptor structure is not yet ready. 1'b1: Indicates that the descriptor structure and data buffer with data is setup and this channel can access the descriptor. When Scatter/Gather mode is disabled, This field is set by the application and cleared by the OTG host. 1'b0: Channel disabled 1'b1: Channel enabled

Bit	Attr	Reset Value	Description
30	R/W SC	0x0	<p>ChDis Channel Disable</p> <p>The application sets this bit to stop transmitting/receiving data on a channel, even before the transfer for that channel is complete. The application must wait for the Channel Disabled interrupt before treating the channel as disabled.</p>
29	RW	0x0	<p>OddFrm Odd Frame</p> <p>This field is set (reset) by the application to indicate that the OTG host must perform a transfer in an odd (micro)frame. This field is applicable for only periodic (isochronous and interrupt) transactions.</p> <p>1'b0: Even (micro)frame 1'b1: Odd (micro)frame</p> <p>This field is not applicable for Scatter/Gather DMA mode and need not be programmed by the application and is ignored by the core.</p>
28:22	RW	0x00	<p>DevAddr Device Address</p> <p>This field selects the specific device serving as the data source or sink.</p>
21:20	RW	0x0	<p>MC_EC Multi Count (MC) / Error Count (EC)</p> <p>When the Split Enable bit of the Host Channel-n Split Control register (HCSPLTn.SpltnEna) is reset (1'b0), this field indicates to the host the number of transactions that must be executed per microframe for this periodic endpoint. For non periodic transfers, this field is used only in DMA mode, and specifies the number packets to be fetched for this channel before the internal DMA engine changes arbitration.</p> <p>2'b00: Reserved This field yields undefined results. 2'b01: 1 transaction 2'b10: 2 transactions to be issued for this endpoint per microframe 2'b11: 3 transactions to be issued for this endpoint per microframe</p> <p>When HCSPLTn.SpltnEna is set (1'b1), this field indicates the number of immediate retries to be performed for a periodic split transactions on transaction errors. This field must be set to at least 2'b01.</p>

Bit	Attr	Reset Value	Description
19:18	RW	0x0	EPType Endpoint Type Indicates the transfer type selected. 2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt
17	RW	0x0	LSpdDev Low-Speed Device This field is set by the application to indicate that this channel is communicating to a low-speed device.
16	RO	0x0	reserved
15	RW	0x0	EPDir Endpoint Direction Indicates whether the transaction is IN or OUT. 1'b0: OUT 1'b1: IN
14:11	RW	0x0	EPNum Endpoint Number Indicates the endpoint number on the device serving as the data source or sink.
10:0	RW	0x000	MPS Maximum Packet Size Indicates the maximum packet size of the associated endpoint.

USBOTG_HCSPLTn

Address: Operational Base + offset (0x0504)

Host Channel-n Split Control Register

Bit	Attr	Reset Value	Description
31	RW	0x0	SplEna Split Enable The application sets this field to indicate that this channel is enabled to perform split transactions.
30:17	RO	0x0	reserved
16	RW	0x0	CompSplt Do Complete Split The application sets this field to request the OTG host to perform a complete split transaction.

Bit	Attr	Reset Value	Description
15:14	RW	0x0	<p>XactPos Transaction Position This field is used to determine whether to send all, first, middle, or last payloads with each OUT transaction.</p> <p>2'b11: All. This is the entire data payload is of this transaction (which is less than or equal to 188 bytes).</p> <p>2'b10: Begin. This is the first data payload of this transaction (which is larger than 188 bytes).</p> <p>2'b00: Mid. This is the middle payload of this transaction (which is larger than 188bytes).</p> <p>2'b01: End. This is the last payload of this transaction (which is larger than 188 bytes).</p>
13:7	RW	0x00	<p>HubAddr Hub Address This field holds the device address of the transaction translator's hub.</p>
6:1	RO	0x0	reserved
0	RW	0x0	<p>PrtAddr Port Address This field is the port number of the recipient transaction translator.</p>

USBOTG_HCINTn

Address: Operational Base + offset (0x0508)

Host Channel-n Interrupt Register

Bit	Attr	Reset Value	Description
31:14	RO	0x0	reserved
13	W1C	0x0	<p>DESC_LST_ROLLIntr Descriptor rollover interrupt This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when the corresponding channel's descriptor list rolls over. For non Scatter/Gather DMA mode, this bit is reserved.</p>
12	W1C	0x0	<p>XCS_XACT_ERR Excessive Transaction Error This bit is valid only when Scatter/Gather DMA mode is enabled. The core sets this bit when 3 consecutive transaction errors occurred on the USB bus. XCS_XACT_ERR will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved.</p>

Bit	Attr	Reset Value	Description
11	W1C	0x0	BNAIntr BNA (Buffer Not Available) Interrupt This bit is valid only when Scatter/Gather DMA mode is enabled. The core generates this interrupt when the descriptor accessed is not ready for the Core to process. BNA will not be generated for Isochronous channels. For non Scatter/Gather DMA mode, this bit is reserved.
10	W1C	0x0	DataTglErr Data Toggle Error In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.
9	W1C	0x0	FrmOvrun Frame Overrun In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core
8	W1C	0x0	BblErr Babble Error In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.
7	W1C	0x0	XactErr Transaction Error Indicates one of the following errors occurred on the USB: CRC check failure, Timeout, Bit stuff error, False EOP. In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.
6	WO	0x0	NYET NYET Response Received Interrupt In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.
5	W1C	0x0	ACK ACK Response Received/Transmitted Interrupt In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.
4	W1C	0x0	NAK NAK Response Received Interrupt In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.
3	W1C	0x0	STALL STALL Response Received Interrupt In Scatter/Gather DMA mode, the interrupt due to this bit is masked in the core.

Bit	Attr	Reset Value	Description
2	W1C	0x0	AHBErr AHB Error This is generated only in DMA mode when there is an AHB error during AHB read/write. The application can read the corresponding channel's DMA address register to get the error address.
1	W1C	0x0	ChHltd Channel Halted In non Scatter/Gather DMA mode, it indicates the transfer completed abnormally either because of any USB transaction error or in response to disable request by the application or because of a completed transfer. In Scatter/Gather DMA mode, this indicates that transfer completed due to any of the following: EOL being set in descriptor, AHB error, Excessive transaction errors, In response to disable request by the application, Babble, Stall, Buffer Not Available (BNA)
0	W1C	0x0	XferCompl Transfer Completed For Scatter/Gather DMA mode, it indicates that current descriptor processing got completed with IOC bit set in its descriptor. In non Scatter/Gather DMA mode, it indicates that Transfer completed normally without any errors.

USBOTG_HCINTMSKn

Address: Operational Base + offset (0x050c)

Host Channel-n Interrupt Mask Register

Bit	Attr	Reset Value	Description
31:14	RO	0x0	reserved
13	RW	0x0	DESC_LST_ROLLIntrMsk Descriptor rollover interrupt Mask register This bit is valid only when Scatter/Gather DMA mode is enabled. In non Scatter/Gather DMA mode, this bit is reserved.
12	RO	0x0	reserved
11	RW	0x0	BNAIntrMsk BNA (Buffer Not Available) Interrupt mask register This bit is valid only when Scatter/Gather DMA mode is enabled. In non Scatter/Gather DMA mode, this bit is reserved.
10	RW	0x0	DataTglErrMsk Data Toggle Error Mask This bit is not applicable in Scatter/Gather DMA mode.
9	RW	0x0	FrmOvrunMsk Frame Overrun Mask This bit is not applicable in Scatter/Gather DMA mode.

Bit	Attr	Reset Value	Description
8	RW	0x0	BblErrMsk Babble Error Mask This bit is not applicable in Scatter/Gather DMA mode.
7	RW	0x0	XactErrMsk Transaction Error Mask This bit is not applicable in Scatter/Gather DMA mode
6	RW	0x0	NyetMsk NYET Response Received Interrupt Mask This bit is not applicable in Scatter/Gather DMA mode.
5	RW	0x0	AckMsk ACK Response Received/Transmitted Interrupt Mask This bit is not applicable in Scatter/Gather DMA mode.
4	RW	0x0	NakMsk NAK Response Received Interrupt Mask This bit is not applicable in Scatter/Gather DMA mode.
3	RW	0x0	StallMsk STALL Response Received Interrupt Mask This bit is not applicable in Scatter/Gather DMA mode.
2	RW	0x0	AHBErrMsk AHB Error Mask Note: This bit is only accessible when OTG_ARCHITECTURE = 2
1	RW	0x0	ChHltdMsk Channel Halted Mask
0	RW	0x0	XferComplMsk Transfer Completed Mask This bit is valid only when Scatter/Gather DMA mode is enabled. In non Scatter/Gather DMA mode, this bit is reserved.

USBOTG_HCTSIZn

Address: Operational Base + offset (0x0510)

Host Channel-n Transfer Size Register

Bit	Attr	Reset Value	Description
31	RW	0x0	DoPng Do Ping This bit is used only for OUT transfers. Setting this field to 1 directs the host to do PING protocol. Note: Do not set this bit for IN transfers. If this bit is set for IN transfers it disables the channel.

Bit	Attr	Reset Value	Description
30:29	RW	0x0	<p>Pid PID</p> <p>The application programs this field with the type of PID to use for the initial transaction. The host maintains this field for the rest of the transfer.</p> <p>2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA (non-control)/SETUP (control)</p>
28:19	RW	0x000	<p>PktCnt Packet Count</p> <p>This field is programmed by the application with the expected number of packets to be transmitted (OUT) or received (IN). The host decrements this count on every successful transmission or reception of an OUT/IN packet. Once this count reaches zero, the application is interrupted to indicate normal completion. The width of this counter is specified as Width of Packet Counters (parameter OTG_PACKET_COUNT_WIDTH).</p>
18:0	RW	0x00000	<p>XferSize Transfer Size</p> <p>For an OUT, this field is the number of data bytes the host sends during the transfer. For an IN, this field is the buffer size that the application has Reserved for the transfer. The application is expected to program this field as an integer multiple of the maximum packet size for IN transactions (periodic and non-periodic). The width of this counter is specified as Width of Transfer Size Counters (parameter OTG_TRANS_COUNT_WIDTH).</p>

USBOTG_HCDMAN

Address: Operational Base + offset (0x0514)

Host Channel-n DMA Address Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	<p>DMAAddr DMA Address</p> <p>This field holds the start address in the external memory from which the data for the endpoint must be fetched or to which it must be stored. This register is incremented on every AHB transaction.</p>

USBOTG_HCDMABn

Address: Operational Base + offset (0x051c)

Host Channel-n DMA Buffer Address Register

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

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	HCDMABn Holds the current buffer address This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.

USBOTG_DCFG

Address: Operational Base + offset (0x0800)

Device Configuration Register

Bit	Attr	Reset Value	Description
31:26	RW	0x02	ResValid Resume Validation Period This field controls the period when the core resumes from a suspend. When this bit is set, the core counts for the ResValid number of clock cycles to detect a valid resume. This field is effective only when DCFG.Ena32KHzSusp is set.
25:24	RW	0x0	PerSchIntvl Periodic Scheduling Interval PerSchIntvl must be programmed only for Scatter/Gather DMA mode. Description: This field specifies the amount of time the Internal DMA engine must allocate for fetching periodic IN endpoint data. Based on the number of periodic endpoints, this value must be specified as 25,50 or 75% of (micro)frame. When any periodic endpoints are active, the internal DMA engine allocates the specified amount of time in fetching periodic IN endpoint data. When no periodic endpoints are active, then the internal DMA engine services nonperiodic endpoints, ignoring this field. After the specified time within a (micro)frame, the DMA switches to fetching for nonperiodic endpoints. 2'b00: 25% of (micro)frame. 2'b01: 50% of (micro)frame. 2'b10: 75% of (micro)frame. 2'b11: Reserved.

Bit	Attr	Reset Value	Description
23	RW	0x0	<p>DescDMA Enable Scatter/Gather DMA in Device mode When the Scatter/Gather DMA option selected during configuration of the RTL, the application can set this bit during initialization to enable the Scatter/Gather DMA operation. NOTE: This bit must be modified only once after a reset. The following combinations are available for programming: GAHBCFG.DMAEn=0,DCFG.DescDMA=0 => Slave mode GAHBCFG.DMAEn=0,DCFG.DescDMA=1 => Invalid GAHBCFG.DMAEn=1,DCFG.DescDMA=0 => Buffered DMA mode GAHBCFG.DMAEn=1,DCFG.DescDMA=1 => Scatter/Gather DMA mode</p>
22:18	RW	0x08	<p>EPMisCnt IN Endpoint Mismatch Count This field is valid only in shared FIFO operation. The application programs this field with a count that determines when the core generates an Endpoint Mismatch interrupt (GINTSTS.EPMis). The core loads this value into an internal counter and decrements it. The counter is reloaded whenever there is a match or when the counter expires. The width of this counter depends on the depth of the Token Queue.</p>
17:13	RO	0x0	reserved
12:11	RW	0x0	<p>PerFrInt Periodic Frame Interval Indicates the time within a (micro)frame at which the application must be notified using the End Of Periodic Frame Interrupt. This can be used to determine if all the isochronous traffic for that (micro)frame is complete. 2'b00: 80% of the (micro)frame interval 2'b01: 85% 2'b10: 90% 2'b11: 95%</p>
10:4	RW	0x00	<p>DevAddr Device Address The application must program this field after every SetAddress control command.</p>
3	RW	0x0	<p>Ena32KHzS Enable 32-KHz Suspend Mode When the USB 1.1 Full-Speed Serial Transceiver Interface is chosen and this bit is set, the core expects the 48-MHz PHY clock to be switched to 32 KHz during a suspend. This bit can only be set if USB 1.1 Full-Speed Serial Transceiver Interface has been selected. If USB 1.1 Full-Speed Serial Transceiver Interface has not been selected, this bit must be zero.</p>

Bit	Attr	Reset Value	Description
2	RW	0x0	<p>NZStsOUTHShk Non-Zero-Length Status OUT Handshake The application can use this field to select the handshake the core sends on receiving a nonzero-length data packet during the OUT transaction of a control transfer's Status stage.</p> <p>1'b1: Send a STALL handshake on a nonzero-length status OUT transaction and do not send the received OUT packet to the application.</p> <p>1'b0: Send the received OUT packet to the application (zero-length or nonzerolength) and send a handshake based on the NAK and STALL bits for the endpoint in the Device Endpoint Control register.</p>
1:0	RW	0x0	<p>DevSpd Device Speed Indicates the speed at which the application requires the core to enumerate, or the maximum speed the application can support. However, the actual bus speed is determined only after the chirp sequence is completed, and is based on the speed of the USB host to which the core is connected.</p> <p>2'b00: High speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) 2'b01: Full speed (USB 2.0 PHY clock is 30 MHz or 60 MHz) 2'b10: Reserved 2'b11: Full speed (USB 1.1 transceiver clock is 48 MHz)</p>

USBOTG_DCTL

Address: Operational Base + offset (0x0804)

Device Control Register

Bit	Attr	Reset Value	Description
31:17	RO	0x0	reserved
16	RW	0x0	<p>NakOnBble Set NAK automatically on babble The core sets NAK automatically for the endpoint on which babble is received.</p>

Bit	Attr	Reset Value	Description
15	RW	0x0	<p>IgnrFrmNum Ignore frame number for isochronous endpoints in case of Scatter/Gather DMA mode. Note: When Scatter/Gather DMA mode is enabled this feature is not applicable to highspeed, high-bandwidth transfers. When this bit is enabled, there must be only one packet per descriptor.</p> <p>0: The core transmits the packets only in the frame number in which they are intended to be transmitted.</p> <p>1: The core ignores the frame number, sending packets immediately as the packets are ready.</p> <p>Scatter/Gather: In Scatter/Gather DMA mode, when this bit is enabled, the packets are not flushed when an ISOC IN token is received for an elapsed frame. When Scatter/Gather DMA mode is disabled, this field is used by the application to enable periodic transfer interrupt. The application can program periodic endpoint transfers for multiple microframes.</p> <p>0: Periodic transfer interrupt feature is disabled; the application must program transfers for periodic endpoints every (micro)frame.</p> <p>1: Periodic transfer interrupt feature is enabled; the application can program transfers for multiple (micro)frames for periodic endpoints.</p> <p>In non-Scatter/Gather DMA mode, the application receives transfer complete interrupt after transfers for multiple (micro) frames are completed.</p>
14:13	RW	0x1	<p>GMC Global Multi Count GMC must be programmed only once after initialization. Applicable only for Scatter/Gather DMA mode. This indicates the number of packets to be serviced for that end point before moving to the next end point. It is only for nonperiodic end points.</p> <p>2'b00: Invalid. 2'b01: 1 packet. 2'b10: 2 packets. 2'b11: 3 packets.</p> <p>When Scatter/Gather DMA mode is disabled, this field is reserved. and reads 2'b00.</p>
12	RO	0x0	reserved
11	RW	0x0	<p>PWROnPrgDone Power-On Programming Done The application uses this bit to indicate that register programming is completed after a wake-up from Power Down mode.</p>

Bit	Attr	Reset Value	Description
10	WO	0x0	CGOUTNak Clear Global OUT NAK A write to this field clears the Global OUT NAK.
9	WO	0x0	SGOUTNak Set Global OUT NAK A write to this field sets the Global OUT NAK. The application uses this bit to send a NAK handshake on all OUT endpoints. The application must set the this bit only after making sure that the Global OUT NAK Effective bit in the Core Interrupt Register (GINTSTS.GOUTNakEff) is cleared.
8	WO	0x0	CGNPInNak Clear Global Non-periodic IN NAK A write to this field clears the Global Non-periodic IN NAK.
7	WO	0x0	SGNPInNak Set Global Non-periodic IN NAK A write to this field sets the Global Non-periodic IN NAK. The application uses this bit to send a NAK handshake on all non-periodic IN endpoints. The core can also set this bit when a timeout condition is detected on a non-periodic endpoint in shared FIFO operation. The application must set this bit only after making sure that the Global IN NAK Effective bit in the Core Interrupt Register (GINTSTS.GINNakEff) is cleared.
6:4	RW	0x0	TstCtl Test Control 3'b000: Test mode disabled 3'b001: Test_J mode 3'b010: Test_K mode 3'b011: Test_SE0_NAK mode 3'b100: Test_Packet mode 3'b101: Test_Force_Enable Others: Reserved
3	RO	0x0	GOUTNaksts Global OUT NAK Status 1'b0: A handshake is sent based on the FIFO Status and the NAK and STALL bit settings. 1'b1: No data is written to the RxFIFO, irrespective of space availability. Sends a NAK handshake on all packets, except on SETUP transactions. All isochronous OUT packets are dropped
2	RO	0x0	GNPINNaksts Global Non-periodic IN NAK Status 1'b0: A handshake is sent out based on the data availability in the transmit FIFO. 1'b1: A NAK handshake is sent out on all non-periodic IN endpoints, irrespective of the data availability in the transmit FIFO.

Bit	Attr	Reset Value	Description
1	RW	0x0	<p>SftDiscon Soft Disconnect</p> <p>The application uses this bit to signal the DWC_otg core to do a soft disconnect. As long as this bit is set, the host does not see that the device is connected, and the device does not receive signals on the USB. The core stays in the disconnected state until the application clears this bit.</p> <p>1'b0: Normal operation. When this bit is cleared after a soft disconnect, the core drives the phy_opmode_o signal on the UTMI+ to 2'b00, which generates a device connect event to the USB host. When the device is reconnected, the USB host restarts device enumeration.</p> <p>1'b1: The core drives the phy_opmode_o signal on the UTMI+ to 2'b01, which generates a device disconnect event to the USB host.</p>
0	RW	0x0	<p>RmtWkUpSig Remote Wakeup Signaling</p> <p>When the application sets this bit, the core initiates remote signaling to wake the USB host. The application must set this bit to instruct the core to exit the Suspend state. As specified in the USB 2.0 specification, the application must clear this bit 1ms after setting it. If LPM is enabled and the core is in the L1 (Sleep) state, when the application sets this bit, the core initiates L1 remote signaling to wake up the USB host. The application must set this bit to instruct the core to exit the Sleep state. As specified in the LPM specification, the hardware automatically clears this bit 50 us (TL1DevDrvResume) after being set by the application. The application must not set this bit when GLPMCFG bRemoteWake from the previous LPM transaction is zero.</p>

USBOTG_DSTS

Address: Operational Base + offset (0x0808)

Device Status Register

Bit	Attr	Reset Value	Description
31:22	RO	0x0	reserved
21:8	RW	0x0000	<p>SOFFN</p> <p>Frame or Microframe Number of the Received SOF</p> <p>When the core is operating at high speed, this field contains a microframe number. When the core is operating at full or low speed, this field contains a frame number.</p>
7:4	RO	0x0	reserved

Bit	Attr	Reset Value	Description
3	RW	0x0	<p>ErrticErr Erratic Error The core sets this bit to report any erratic errors (phy_rxvalid_i/phy_rxvldh_i or phy_rxactive_i is asserted for at least 2 ms, due to PHY error) seen on the UTMI+. Due to erratic errors, the DWC_otg core goes into Suspended state and an interrupt is generated to the application with Early Suspend bit of the Core Interrupt register (GINTSTS.ErlySusp). If the early suspend is asserted due to an erratic error, the application can only perform a soft disconnect recover.</p>
2:1	RW	0x0	<p>EnumSpd Enumerated Speed Indicates the speed at which the DWC_otg core has come up after speed detection through a chirp sequence. 2'b00: High speed (PHY clock is running at 30 or 60 MHz) 2'b01: Full speed (PHY clock is running at 30 or 60 MHz) 2'b10: Low speed (PHY clock is running at 48 MHz, internal phy_clk at 6 MHz) 2'b11: Full speed (PHY clock is running at 48 MHz) Low speed is not supported for devices using a UTMI+ PHY.</p>
0	RW	0x0	<p>SuspSts Suspend Status In Device mode, this bit is set as long as a Suspend condition is detected on the USB. The core enters the Suspended state when there is no activity on the utmi_linestate signal for an extended period of time. The core comes out of the suspend: When there is any activity on the utmi_linestate signal, When the application writes to the Remote Wakeup Signaling bit in the Device Control register (DCTL.RmtWkUpSig).</p>

USBOTG_DIEPMSK

Address: Operational Base + offset (0x0810)

Device IN Endpoint common interrupt mask register

Bit	Attr	Reset Value	Description
31:14	RO	0x0	reserved
13	RW	0x0	NAKMsk NAK interrupt Mask
12:10	RO	0x0	reserved
9	RW	0x0	BNAInIntrMsk BNA Interrupt Mask
8	RW	0x0	TxfifoUndrnMsk Fifo Underrun Mask
7	RO	0x0	reserved

Bit	Attr	Reset Value	Description
6	RW	0x0	INEPNakEffMsk IN Endpoint NAK Effective Mask
5	RW	0x0	INTknEPMisMsk IN Token received with EP Mismatch Mask
4	RW	0x0	INTknTxFEmpMsk IN Token Received When TxFIFO Empty Mask
3	RW	0x0	TimeOUTMsk Timeout Condition Mask
2	RW	0x0	AHBErrMsk AHB Error Mask
1	RW	0x0	EPDisbldMsk Endpoint Disabled Interrupt Mask
0	RW	0x0	XferComplMsk Transfer Completed Interrupt Mask

USBOTG_DOEPMSK

Address: Operational Base + offset (0x0814)

Device OUT Endpoint common interrupt mask register

Bit	Attr	Reset Value	Description
31:15	RO	0x0	reserved
14	RW	0x0	NYETMsk NYET Interrupt Mask
13	RW	0x0	NAKMsk NAK Interrupt Mask
12	RW	0x0	BbleErrMsk Babble Interrupt Mask
11:10	RO	0x0	reserved
9	RW	0x0	BnaOutIntrMsk BNA interrupt Mask
8	RW	0x0	OutPktErrMsk OUT Packet Error Mask
7	RO	0x0	reserved
6	RW	0x0	Back2BackSETup Back-to-Back SETUP Packets Received Mask Applies to control OUT endpoints only.
5	RO	0x0	reserved
4	RW	0x0	OUTTknEPdisMsk OUT Token Received when Endpoint Disabled Mask Applies to control OUT endpoints only.
3	RW	0x0	SetUPMsk SETUP Phase Done Mask Applies to control endpoints only.
2	RW	0x0	AHBErrMsk AHB Error

Bit	Attr	Reset Value	Description
1	RW	0x0	EPDisbldMsk Endpoint Disabled Interrupt Mask
0	RW	0x0	XferComplMsk Transfer Completed Interrupt Mask

USBOTG_DAINT

Address: Operational Base + offset (0x0818)

Device All Endpoints interrupt register

Bit	Attr	Reset Value	Description
31:16	RO	0x0000	OutEPInt OUT Endpoint Interrupt Bits One bit per OUT endpoint: Bit 16 for OUT endpoint 0, bit 31 for OUT endpoint 15
15:0	RO	0x0000	InEpInt IN Endpoint Interrupt Bits One bit per IN Endpoint: Bit 0 for IN endpoint 0, bit 15 for endpoint 15

USBOTG_DAINTMSK

Address: Operational Base + offset (0x081c)

Device All Endpoint interrupt mask register

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	OutEpMsk OUT EP Interrupt Mask Bits One per OUT Endpoint: Bit 16 for OUT EP 0, bit 31 for OUT EP 15
15:0	RW	0x0000	InEpMsk IN EP Interrupt Mask Bits One bit per IN Endpoint: Bit 0 for IN EP 0, bit 15 for IN EP 15

USBOTG_DTKNQR1

Address: Operational Base + offset (0x0820)

Device IN token sequence learning queue read register1

Bit	Attr	Reset Value	Description
31:8	RO	0x0000000	EPTkn Endpoint Token Four bits per token represent the endpoint number of the token: Bits [31:28]: Endpoint number of Token 5 Bits [27:24]: Endpoint number of Token 4 Bits [15:12]: Endpoint number of Token 1 Bits [11:8]: Endpoint number of Token 0
7	RO	0x0	WrapBit Wrap Bit This bit is set when the write pointer wraps. It is cleared when the learning queue is cleared.

Bit	Attr	Reset Value	Description
6:5	RO	0x0	reserved
4:0	RO	0x00	INTknWPtr IN Token Queue Write Pointer

USBOTG_DTKNQR2

Address: Operational Base + offset (0x0824)

Device IN token sequence learning queue read register2

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	EPTkn Endpoint Token Four bits per token represent the endpoint number of the token: Bits [31:28]: Endpoint number of Token 13 Bits [27:24]: Endpoint number of Token 12 Bits [7:4]: Endpoint number of Token 7 Bits [3:0]: Endpoint number of Token 6

USBOTG_DVBUSDIS

Address: Operational Base + offset (0x0828)

Device VBUS discharge time register

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0b8f	DVBUSDis Device VBUS Discharge Time Specifies the VBUS discharge time after VBUS pulsing during SRP. This value equals: VBUS discharge time in PHY clocks / 1,024. The value you use depends whether the PHY is operating at 30 MHz (16-bit data width) or 60 MHz (8-bit data width). Depending on your VBUS load, this value can need adjustment.

USBOTG_DVBUSPULSE

Address: Operational Base + offset (0x082c)

Device VBUS Pulsing Timer Register

Bit	Attr	Reset Value	Description
31:12	RO	0x0	reserved
11:0	RW	0x000	DVBUSPulse Device VBUS Pulsing Time Specifies the VBUS pulsing time during SRP. This value equals: VBUS pulsing time in PHY clocks / 1,024. The value you use depends whether the PHY is operating at 30 MHz (16-bit data width) or 60 MHz (8-bit data width).

USBOTG_DTHRCTL

Address: Operational Base + offset (0x0830)

Device Threshold Control Register

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27	RW	0x1	<p>ArbPrkEn Arbiter Parking Enable This bit controls internal DMA arbiter parking for IN endpoints. When thresholding is enabled and this bit is set to one, then the arbiter parks on the IN endpoint for which there is a token received on the USB. This is done to avoid getting into underrun conditions. By default the parking is enabled.</p>
26	RO	0x0	reserved
25:17	RW	0x008	<p>RxThrLen Receive Threshold Length This field specifies Receive thresholding size in DWORDS. This field also specifies the amount of data received on the USB before the core can start transmitting on the AHB. The threshold length has to be at least eight DWORDS. The recommended value for ThrLen is to be the same as the programmed AHB Burst Length (GAHBCFG.HBstLen).</p>
16	RW	0x0	<p>RxThrEn Receive Threshold Enable When this bit is set, the core enables thresholding in the receive direction.</p>
15:13	RO	0x0	reserved
12:11	RW	0x0	<p>AHBThrRatio AHB Threshold Ratio These bits define the ratio between the AHB threshold and the MAC threshold for the transmit path only. The AHB threshold always remains less than or equal to the USB threshold, because this does not increase overhead. Both the AHB and the MAC threshold must be DWORD-aligned. The application needs to program TxThrLen and the AHBThrRatio to make the AHB Threshold value DWORD aligned. If the AHB threshold value is not DWORD aligned, the core might not behave correctly. When programming the TxThrLen and AHBThrRatio, the application must ensure that the minimum AHB threshold value does not go below 8 DWORDS to meet the USB turnaround time requirements. 2'b00: AHB threshold = MAC threshold 2'b01: AHB threshold = MAC threshold / 2 2'b10: AHB threshold = MAC threshold / 4 2'b11: AHB threshold = MAC threshold / 8 </p>

Bit	Attr	Reset Value	Description
10:2	RW	0x008	<p>TxThrLen Transmit Threshold Length This field specifies Transmit thresholding size in DWORDS. This field also forms the MAC threshold and specifies the amount of data, in bytes, to be in the corresponding endpoint transmit FIFO before the core can start a transmit on the USB. When the value of AHBThrRatio is 2'h00, the threshold length must be at least 8 DWORDS. If the AHBThrRatio is nonzero, the application must ensure that the AHB threshold value does not go below the recommended 8 DWORDs.</p> <p>This field controls both isochronous and non-isochronous IN endpoint thresholds.</p> <p>The recommended value for ThrLen is to be the same as the programmed AHB Burst Length (GAHBCFG.HBstLen).</p>
1	RW	0x0	<p>ISOThrEn ISO IN Endpoints Threshold Enable When this bit is set, the core enables thresholding for isochronous IN endpoints.</p>
0	RW	0x0	<p>NonISOThrEn Non-ISO IN Endpoints Threshold Enable When this bit is set, the core enables thresholding for Non Isochronous IN endpoints.</p>

USBOTG_DIEPEMPMSK

Address: Operational Base + offset (0x0834)

Device IN endpoint FIFO empty interrupt mask register

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0000	<p>InEpTxfEmpMsk IN EP Tx FIFO Empty Interrupt Mask Bits These bits acts as mask bits for DIEPINTn. TxFTEmp interrupt One bit per IN Endpoint: Bit 0 for IN endpoint 0 ... Bit 15 for endpoint 15</p>

USBOTG_DEACHINT

Address: Operational Base + offset (0x0838)

Device each endpoint interrupt register

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

Bit	Attr	Reset Value	Description
31:16	RO	0x0000	EchOutEPInt OUT Endpoint Interrupt Bits One bit per OUT endpoint: Bit 16 for OUT endpoint 0 ... Bit 31 for OUT endpoint 15
15:0	RO	0x0000	EchInEpInt IN Endpoint Interrupt Bits One bit per IN Endpoint: Bit 0 for IN endpoint 0 ... Bit 15 for endpoint 15

USBOTG_DEACHINTMSK

Address: Operational Base + offset (0x083c)

Device each endpoint interrupt register mask

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	EchOutEpMsk OUT EP Interrupt Mask Bits One per OUT Endpoint: Bit 16 for IN endpoint 0 ... Bit 31 for endpoint 15
15:0	RW	0x0000	EchInEpMsk IN EP Interrupt Mask Bits One bit per IN Endpoint: Bit 0 for IN endpoint 0 ... Bit 15 for endpoint 15

USBOTG_DIEPEACHMSKn

Address: Operational Base + offset (0x0840)

Device each IN endpoint -n interrupt Register

Bit	Attr	Reset Value	Description
31:14	RO	0x0	reserved
13	RW	0x0	NAKMsk NAK interrupt Mask
12:10	RO	0x0	reserved
9	RW	0x0	BNAInIntrMsk BNA interrupt Mask
8	RW	0x0	TxfifoUndrnMsk Fifo Under run Mask
7	RO	0x0	reserved
6	RW	0x0	INEPNakEffMsk IN Endpoint NAK Effective Mask

Bit	Attr	Reset Value	Description
5	RW	0x0	INTknEPMisMsk IN Token received with EP Mismatch Mask
4	RW	0x0	INTknTxFEmpMsk IN Token Received When TxFIFO Empty Mask
3	RW	0x0	TimeOUTMsk Timeout Condition Mask(Non-isochronous endpoints)
2	RW	0x0	AHBErrMsk AHB Error Mask
1	RW	0x0	EPDisbldMsk Endpoint Disabled Interrupt Mask
0	RW	0x0	XferComplMsk Transfer Completed Interrupt Mask

USBOTG_DOEPEACHMSKn

Address: Operational Base + offset (0x0880)

Device each out endpoint-n interrupt register

Bit	Attr	Reset Value	Description
31:15	RO	0x0	reserved
14	RW	0x0	NYETMsk NYET interrupt Mask
13	RW	0x0	NAKMsk NAK interrupt Mask
12	RW	0x0	BbleErrMsk Babble interrupt Mask
11:10	RO	0x0	reserved
9	RW	0x0	BnaOutIntrMsk BNA interrupt Mask
8	RW	0x0	OutPktErrMsk OUT Packet Error Mask
7	RO	0x0	reserved
6	RW	0x0	Back2BackSETup Back-to-Back SETUP Packets Received Mask Applies to control OUT endpoints only.
5	RO	0x0	reserved
4	RW	0x0	OUTTknEPdisMsk OUT Token Received when Endpoint Disabled Mask Applies to control OUT endpoints only.
3	RW	0x0	SetUPMsk SETUP Phase Done Mask Applies to control endpoints only.
2	RW	0x0	AHBErrMsk AHB Error
1	RW	0x0	EPDisbldMsk Endpoint Disabled Interrupt Mask

Bit	Attr	Reset Value	Description
0	RW	0x0	XferComplMsk Transfer Completed Interrupt Mask

USBOTG_DIEPCTL0

Address: Operational Base + offset (0x0900)

Device control IN endpoint 0 control register

Bit	Attr	Reset Value	Description
31	R/W SC	0x0	EPEna Endpoint Enable When Scatter/Gather DMA mode is enabled, for IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. When Scatter/Gather DMA mode is disabled-such as in buffer-pointer based DMA mode-this bit indicates that data is ready to be transmitted on the endpoint. The core clears this bit before setting the following interrupts on this endpoint: Endpoint Disabled; Transfer Completed.
30	R/W SC	0x0	EPDis Endpoint Disable The application sets this bit to stop transmitting data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled Interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint.
29:28	RO	0x0	reserved
27	WO	0x0	SNAK Set NAK A write to this bit sets the NAK bit for the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also set this bit for an endpoint after a SETUP packet is received on that endpoint.
26	WO	0x0	CNAK Clear NAK A write to this bit clears the NAK bit for the endpoint.
25:23	RO	0x0	reserved
22	RW	0x0	TxFNum TxFIFO Number For Shared FIFO operation, this value is always set to 0, indicating that control IN endpoint 0 data is always written in the Non-Periodic Transmit FIFO. For Dedicated FIFO operation, this value is set to the FIFO number that is assigned to IN Endpoint 0.

Bit	Attr	Reset Value	Description
21	R/W SC	0x0	<p>Stall STALL Handshake</p> <p>The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority.</p>
20	RO	0x0	reserved
19:18	RO	0x0	<p>EPType Endpoint Type</p> <p>Hardcoded to 00 for control</p>
17	RO	0x0	<p>NAKsts NAK Status</p> <p>Indicates the following:</p> <p>1'b0: The core is transmitting non-NAK handshakes based on the FIFO status</p> <p>1'b1: The core is transmitting NAK handshakes on this endpoint. When this bit is set, either by the application or core, the core stops transmitting data, even if there is data available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake.</p>
16	RO	0x0	reserved
15	RO	0x1	<p>USBActEP USB Active Endpoint</p> <p>This bit is always set to 1, indicating that control endpoint 0 is always active in all configurations and interfaces.</p>
14:11	RW	0x0	<p>NextEp Next Endpoint</p> <p>Applies to non-periodic IN endpoints only. Indicates the endpoint number to be fetched after the data for the current endpoint is fetched. The core can access this field, even when the Endpoint Enable (EPEna) bit is not set. This field is not valid in Slave mode.</p> <p>Note: This field is valid only for Shared FIFO operations.</p>
10:2	RO	0x0	reserved
1:0	RW	0x0	<p>MPS Maximum Packet Size</p> <p>Applies to IN and OUT endpoints. The application must program this field with the maximum packet size for the current logical endpoint.</p> <p>2'b00: 64 bytes 2'b01: 32 bytes 2'b10: 16 bytes 2'b11: 8 bytes</p>

USBOTG_DIEPINTn

Address: Operational Base + offset (0x0908)

Device Endpoint-n Interrupt Register

Bit	Attr	Reset Value	Description
31:15	RO	0x0	reserved
14	W1C	0x0	NYETIntrpt NYET interrupt The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint.
13	W1C	0x0	NAKIntrpt NAK interrupt The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo.
12	W1C	0x0	BbleErrIntrpt BbleErr (Babble Error) interrupt The core generates this interrupt when babble is received for the endpoint.
11	W1C	0x0	PktDrpSts Packet Dropped Status This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected.
10	RO	0x0	reserved
9	W1C	0x0	BNAIntr BNA (Buffer Not Available) Interrupt The core generates this interrupt when the descriptor accessed is not ready for the Core to process, such as Host busy or DMA done Dependency: This bit is valid only when Scatter/Gather DMA mode is enabled.
8	W1C	0x0	TxfifoUndrn FIFO Underrun Applies to IN endpoints only. The core generates this interrupt when it detects a transmit FIFO underrun condition for this endpoint. Dependency: This interrupt is valid only when both of the following conditions are true: Parameter OTG_ENDED_TX_FIFO==1; Thresholding is enabled; OUT Packet Error(OutPktErr). Applies to OUT endpoints only. This interrupt is asserted when the core detects an overflow or a CRC error for an OUT packet. Dependency: This interrupt is valid only when both of the following conditions are true: Parameter OTG_ENDED_TX_FIFO==1; Thresholding is enabled.

Bit	Attr	Reset Value	Description
7	W1C	0x0	<p>TxFEmp Transmit FIFO Empty This bit is valid only for IN Endpoints. This interrupt is asserted when the TxFIFO for this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register (GAHBCFG.NPTxFEmpLvl).</p>
6	W1C	0x0	<p>INEPNakEff IN Endpoint NAK Effective Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. This bit is applicable only when the endpoint is enabled. Back-to-Back SETUP Packets Received (Back2BackSETup) Applies to Control OUT endpoints only. This bit indicates that the core has received more than three back-to-back SETUP packets for this particular endpoint.</p>
5	W1C	0x0	<p>INTknEPMis IN Token Received with EP Mismatch Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one for which the IN token was received. This interrupt is asserted on the endpoint for which the IN token was received. Status Phase Received For Control Write (StsPhseRcvd) This interrupt is valid only for Control OUT endpoints and only in Scatter Gather DMA mode. This interrupt is generated only after the core has transferred all the data that the host has sent during the data phase of a control write transfer, to the system memory buffer. The interrupt indicates to the application that the host has switched from data phase to the status phase of a Control Write transfer. The application can use this interrupt to ACK or STALL the Status phase, after it has decoded the data phase. This is applicable only in case of Scatter Gather DMA mode.</p>

Bit	Attr	Reset Value	Description
4	W1C	0x0	INTknTxFEmp IN Token Received When TxFIFO is Empty Indicates that an IN token was received when the associated TxFIFO periodic/nonperiodic) was empty. This interrupt is asserted on the endpoint for which the IN token was received. OUT Token Received When Endpoint Disabled (OUTTknEPdis) Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint for which the OUT token was received.
3	W1C	0x0	TimeOUT Timeout Condition In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB for the last IN token on this endpoint. SETUP Phase Done (SetUp) Applies to control OUT endpoints only. Indicates that the SETUP phase for the control endpoint is complete and no more back-to-back SETUP packets were received for the current control transfer. On this interrupt, the application can decode the received SETUP data packet.
2	W1C	0x0	AHBErr AHB Error Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address.
1	W1C	0x0	EPDisbld Endpoint Disabled Interrupt Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request.
0	W1C	0x0	XferCompl Transfer Completed Interrupt Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled: For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit for the corresponding descriptor is set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, for this endpoint.

USBOTG_DIEPTSIZEn

Address: Operational Base + offset (0x0910)

Device endpoint n transfer size register

Bit	Attr	Reset Value	Description
31	RO	0x0	reserved
30:29	RW	0x0	<p>MC Multi Count Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints.</p> <p>2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets</p> <p>For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetch for an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp). Received Data PID (RxDPID)</p> <p>Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint.</p> <p>2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA</p> <p>SETUP Packet Count (SUPCnt).Applies to control OUT Endpoints only.This field specifies the number of back-to-back SETUP data packets the endpoint can receive.</p> <p>2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets</p>
28:19	RW	0x000	<p>PktCnt Packet Count Indicates the total number of USB packets that constitute the Transfer Size amount of data for this endpoint. The power-on value is specified for Width of Packet Counters during coreConsultant configuration (parameter OTG_PACKET_COUNT_WIDTH). IN Endpoints: This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO. OUT Endpoints: This field is decremented every time a packet (maximum size or short packet) is written to the RxFIFO.</p>

Bit	Attr	Reset Value	Description
18:0	RW	0x000000	XferSize Transfer Size This field contains the transfer size in bytes for the current endpoint. The power-on value is specified for Width of Transfer Size Counters during coreConsultant configuration (parameter OTG_TRANS_COUNT_WIDTH). The core only interrupts the application after it has exhausted the transfer size amount of data. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. IN Endpoints: The core decrements this field every time a packet from the external memory is written to the TxFIFO. OUT Endpoints: The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.

USBOTG_DIEPDMan

Address: Operational Base + offset (0x0914)

Device endpoint-n DMA address register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	DMAAddr DMA Address Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.

USBOTG_DTXFSTS_n

Address: Operational Base + offset (0x0918)

Device IN endpoint transmit FIFO status register

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved

Bit	Attr	Reset Value	Description
15:0	RW	0x0000	<p>INEPTxFSpcAvail IN Endpoint TxFIFO Space Avail Indicates the amount of free space available in the Endpoint TxFIFO. Values are in terms of 32-bit words.</p> <p>16'h0: Endpoint TxFIFO is full 16'h1: 1 word available 16'h2: 2 words available 16'hn: n words available (where 0 . n . 32,768) 16'h8000: 32,768 words available Others: Reserved</p>

USBOTG_DIEPDMAFn

Address: Operational Base + offset (0x091c)

Device endpoint-n DMA buffer address register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	<p>DMABufferAddr DMA Buffer Address Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.</p>

USBOTG_DIEPCTLn

Address: Operational Base + offset (0x0920)

Device endpoint-n control register

Bit	Attr	Reset Value	Description
31	R/W SC	0x0	<p>EPEna Endpoint Enable Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabled-such as for buffer-pointer based DMA mode: For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint ; For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done, Endpoint Disabled, Transfer Completed. Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory.</p>

Bit	Attr	Reset Value	Description
30	R/W SC	0x0	<p>EPDis Endpoint Disable</p> <p>Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint.</p>
29	WO	0x0	<p>SetD1PID Set DATA1 PID</p> <p>Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr). Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode.</p>
28	WO	0x0	<p>SetD0PID Set DATA0 PID</p> <p>Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro)frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure.</p>
27	WO	0x0	<p>SNAK Set NAK</p> <p>Applies to IN and OUT endpoints. A write to this bit sets the NAK bit for the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also set this bit for OUT endpoints on a Transfer Completed interrupt, or after a SETUP is received on the endpoint.</p>
26	WO	0x0	<p>CNAK Clear NAK</p> <p>Applies to IN and OUT endpoints. A write to this bit clears the NAK bit for the endpoint.</p>

Bit	Attr	Reset Value	Description
25:22	RW	0x0	<p>TxFNum Tx FIFO Number</p> <p>Shared FIFO Operation: non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic Tx FIFO number. 4'h0: Non-Periodic Tx FIFO; Others: Specified Periodic Tx FIFO number. Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint using coreConsultant, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area.</p> <p>Dedicated FIFO Operation: these bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.</p>
21	RW	0x0	<p>Stall STALL Handshake</p> <p>Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core.</p> <p>Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority.</p> <p>Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake.</p>
20	RW	0x0	<p>Snp Snoop Mode</p> <p>Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory.</p>
19:18	RW	0x0	<p>EPType Endpoint Type</p> <p>Applies to IN and OUT endpoints. This is the transfer type supported by this logical endpoint.</p> <p>2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt</p>

Bit	Attr	Reset Value	Description
17	RO	0x0	<p>NAKSts NAK Status Applies to IN and OUT endpoints. Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet. For non-isochronous IN endpoints: The core stops transmitting any data on an IN endpoint, even if there data is available in the TxFIFO. For isochronous IN endpoints: The core sends out a zero-length data packet, even if there data is available in the TxFIFO. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake.</p>
16	RO	0x0	<p>DPID Endpoint Data PID Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID. 1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Even/Odd (Micro)Frame (EO_FrNum) In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro) frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register. 1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure.</p>

Bit	Attr	Reset Value	Description
15	R/W SC	0x0	USBActEP USB Active Endpoint Applies to IN and OUT endpoints. Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program endpoint registers accordingly and set this bit.
14:11	RW	0x0	NextEp Next Endpoint Applies to non-periodic IN endpoints only. Indicates the endpoint number to be fetched after the data for the current endpoint is fetched. The core can access this field, even when the Endpoint Enable (EPEna) bit is low. This field is not valid in Slave mode operation. Note: This field is valid only for Shared FIFO operations.
10:0	RW	0x000	MPS Maximum Packet Size Applies to IN and OUT endpoints. The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.

USBOTG_DOEPCtl0

Address: Operational Base + offset (0x0b00)

Device control OUT endpoint 0 control register

Bit	Attr	Reset Value	Description
31	R/W SC	0x0	EPEna Endpoint Enable When Scatter/Gather DMA mode is enabled, for OUT endpoints this bit indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is disabled? such as for buffer-pointer based DMA mode)-this bit indicates that the application has allocated the memory to start receiving data from the USB. The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done, Endpoint Disabled, Transfer Completed. <i>Note: In DMA mode, this bit must be set for the core to transfer SETUP data packets into memory.</i>
30	WO	0x0	EPDis Endpoint Disable The application cannot disable control OUT endpoint 0.
29:28	RO	0x0	reserved

Bit	Attr	Reset Value	Description
27	WO	0x0	SNAK Set NAK A write to this bit sets the NAK bit for the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also set bit on a Transfer Completed interrupt, or after a SETUP is received on the endpoint.
26	WO	0x0	CNAK Clear NAK A write to this bit clears the NAK bit for the endpoint.
25:22	RO	0x0	reserved
21	R/W SC	0x0	Stall STALL Handshake The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit or Global OUT NAK is set along with this bit, the STALL bit takes priority. Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake.
20	RW	0x0	Snp Snoop Mode This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory.
19:18	RO	0x0	EPType Endpoint Type Hardcoded to 2'b00 for control.
17	RO	0x0	NAKsts NAK Status Indicates the following: 1'b0: The core is transmitting non-NAK handshakes based on the FIFO status. 1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit, the core stops receiving data, even if there is space in the RxFIFO to accommodate the incoming packet. Irrespective of this bit setting, the core always responds to SETUP data packets with an ACK handshake.
16	RO	0x0	reserved
15	RO	0x0	USBActEP USB Active Endpoint This bit is always set to 1, indicating that a control endpoint 0 is always active in all configurations and interfaces.
14:2	RO	0x0	reserved

Bit	Attr	Reset Value	Description
1:0	RO	0x0	MPS Maximum Packet Size The maximum packet size for control OUT endpoint 0 is the same as what is programmed in control IN Endpoint 0. 2'b00: 64 bytes 2'b01: 32 bytes 2'b10: 16 bytes 2'b11: 8 bytes

USBOTG_DOEPINTn

Address: Operational Base + offset (0x0b08)

Device endpoint-n control register

Bit	Attr	Reset Value	Description
31:15	RO	0x0	reserved
14	W1C	0x0	NYETInrpt NYET interrupt The core generates this interrupt when a NYET response is transmitted for a non isochronous OUT endpoint.
13	W1C	0x0	NAKInrpt NAK interrupt The core generates this interrupt when a NAK is transmitted or received by the device. In case of isochronous IN endpoints the interrupt gets generated when a zero length packet is transmitted due to un-availability of data in the TXFifo.
12	W1C	0x0	BbleErrInrpt BbleErr (Babble Error) interrupt The core generates this interrupt when babble is received for the endpoint.
11	W1C	0x0	PktDrpSts Packet Dropped Status This bit indicates to the application that an ISOC OUT packet has been dropped. This bit does not have an associated mask bit and does not generate an interrupt. Dependency: This bit is valid in non Scatter/Gather DMA mode when periodic transfer interrupt feature is selected.
10	RO	0x0	reserved
9	W1C	0x0	BNAInrpt BNA (Buffer Not Available) Interrupt The core generates this interrupt when the descriptor accessed is not ready for the Core to process, such as Host busy or DMA done. Dependency: This bit is valid only when Scatter/Gather DMA mode is enabled.

Bit	Attr	Reset Value	Description
8	W1C	0x0	<p>TxfifoUndrn FIFO Underrun</p> <p>Applies to IN endpoints only. The core generates this interrupt when it detects a transmit FIFO underrun condition for this endpoint. Dependency: This interrupt is valid only when both of the following conditions are true: Parameter OTG_EN_DED_TX_FIFO==1, Thresholding is enabled, OUT Packet Error (OutPktErr). Applies to OUT endpoints only . This interrupt is asserted when the core detects an overflow or a CRC error for an OUT packet. Dependency: This interrupt is valid only when both of the following conditions are true: Parameter OTG_EN_DED_TX_FIFO==1, Thresholding is enabled.</p>
7	W1C	0x0	<p>TxFEmp Transmit FIFO Empty</p> <p>This bit is valid only for IN Endpoints. This interrupt is asserted when the TxFIFO for this endpoint is either half or completely empty. The half or completely empty status is determined by the TxFIFO Empty Level bit in the Core AHB Configuration register(GAHBCFG.NPTxFEmpLvl)).</p>
6	W1C	0x0	<p>INEPNakEff IN Endpoint NAK Effective</p> <p>Applies to periodic IN endpoints only. This bit can be cleared when the application clears the IN endpoint NAK by writing to DIEPCTLn.CNAK. This interrupt indicates that the core has sampled the NAK bit set (either by the application or by the core). The interrupt indicates that the IN endpoint NAK bit set by the application has taken effect in the core. This interrupt does not guarantee that a NAK handshake is sent on the USB. A STALL bit takes priority over a NAK bit. This bit is applicable only when the endpoint is enabled. Back-to-Back SETUP Packets Received (Back2BackSETUp) Applies to Control OUT endpoints only.</p> <p>This bit indicates that the core has received more than three back-to-back SETUP packets for this particular endpoint.</p>
5	W1C	0x0	<p>INTknEPMIs IN Token Received with EP Mismatch</p> <p>Applies to non-periodic IN endpoints only. Indicates that the data in the top of the non-periodic TxFIFO belongs to an endpoint other than the one for which the IN token was received. This interrupt is asserted on the endpoint for which the IN token was received.</p> <p>Status Phase Received For Control Write (StsPhseRcvd)</p> <p>This interrupt is valid only for Control OUT endpoints and only in Scatter Gather DMA mode.</p>

Bit	Attr	Reset Value	Description
4	W1C	0x0	INTknTxFEmp IN Token Received When TxFIFO is Empty Indicates that an IN token was received when the associated TxFIFO periodic/nonperiodic) was empty. This interrupt is asserted on the endpoint for which the IN token was received. OUT Token Received When Endpoint Disabled (OUTTknEPdis) Indicates that an OUT token was received when the endpoint was not yet enabled. This interrupt is asserted on the endpoint for which the OUT token was received.
3	W1C	0x0	TimeOUT Timeout Condition In shared TX FIFO mode, applies to non-isochronous IN endpoints only. In dedicated FIFO mode, applies only to Control IN endpoints. In Scatter/Gather DMA mode, the TimeOUT interrupt is not asserted. Indicates that the core has detected a timeout condition on the USB for the last IN token on this endpoint. SETUP Phase Done (SetUp). Applies to control OUT endpoints only. Indicates that the SETUP phase for the control endpoint is complete and no more back-to-back SETUP packets were received for the current control transfer. On this interrupt, the application can decode the received SETUP data packet.
2	W1C	0x0	AHBErr AHB Error Applies to IN and OUT endpoints. This is generated only in Internal DMA mode when there is an AHB error during an AHB read/write. The application can read the corresponding endpoint DMA address register to get the error address.
1	W1C	0x0	EPDisbld Endpoint Disabled Interrupt Applies to IN and OUT endpoints. This bit indicates that the endpoint is disabled per the application's request.
0	W1C	0x0	XferCompl Transfer Completed Interrupt Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled. For IN endpoint this field indicates that the requested data from the descriptor is moved from external system memory to internal FIFO. For OUT endpoint this field indicates that the requested data from the internal FIFO is moved to external system memory. This interrupt is generated only when the corresponding endpoint descriptor is closed, and the IOC bit for the corresponding descriptor is set. When Scatter/Gather DMA mode is disabled, this field indicates that the programmed transfer is complete on the AHB as well as on the USB, for this endpoint.

USBOTG_DOEPTSIZn

Address: Operational Base + offset (0x0b10)

Device endpoint n transfer size register

Bit	Attr	Reset Value	Description
31	RO	0x0	reserved
30:29	RW	0x0	<p>MC Multi Count Applies to IN endpoints only. For periodic IN endpoints, this field indicates the number of packets that must be transmitted per microframe on the USB. The core uses this field to calculate the data PID for isochronous IN endpoints.</p> <p>2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets</p> <p>For non-periodic IN endpoints, this field is valid only in Internal DMA mode. It specifies the number of packets the core must fetch for an IN endpoint before it switches to the endpoint pointed to by the Next Endpoint field of the Device Endpoint-n Control register (DIEPCTLn.NextEp). Received Data PID (RxDPID)</p> <p>Applies to isochronous OUT endpoints only. This is the data PID received in the last packet for this endpoint.</p> <p>2'b00: DATA0 2'b01: DATA2 2'b10: DATA1 2'b11: MDATA</p> <p>SETUP Packet Count (SUPCnt).Applies to control OUT Endpoints only. This field specifies the number of back-to-back SETUP data packets the endpoint can receive.</p> <p>2'b01: 1 packet 2'b10: 2 packets 2'b11: 3 packets</p>
28:19	RW	0x000	<p>PktCnt Packet Count Indicates the total number of USB packets that constitute the Transfer Size amount of data for this endpoint. The power-on value is specified for Width of Packet Counters during coreConsultant configuration (parameter OTG_PACKET_COUNT_WIDTH).</p> <p>IN Endpoints: This field is decremented every time a packet (maximum size or short packet) is read from the TxFIFO.</p> <p>OUT Endpoints: This field is decremented every time a packet (maximum size or short packet) is written to the RxFIFO.</p>

Bit	Attr	Reset Value	Description
18:0	RW	0x000000	XferSize Transfer Size This field contains the transfer size in bytes for the current endpoint. The power-on value is specified for Width of Transfer Size Counters during coreConsultant configuration (parameter OTG_TRANS_COUNT_WIDTH). The core only interrupts the application after it has exhausted the transfer size amount of data. The transfer size can be set to the maximum packet size of the endpoint, to be interrupted at the end of each packet. IN Endpoints: The core decrements this field every time a packet from the external memory is written to the TxFIFO. OUT Endpoints: The core decrements this field every time a packet is read from the RxFIFO and written to the external memory.

USBOTG_DOEPDMAn

Address: Operational Base + offset (0x0b14)

Device Endpoint-n DMA Address Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	DMAAddr DMA Address Holds the start address of the external memory for storing or fetching endpoint data. Note: For control endpoints, this field stores control OUT data packets as well as SETUP transaction data packets. When more than three SETUP packets are received back-to-back, the SETUP data packet in the memory is overwritten. This register is incremented on every AHB transaction. The application can give only a DWORD-aligned address. When Scatter/Gather DMA mode is not enabled, the application programs the start address value in this field. When Scatter/Gather DMA mode is enabled, this field indicates the base pointer for the descriptor list.

USBOTG_DOEPDMABn

Address: Operational Base + offset (0x0b1c)

Device endpoint-n DMA buffer address register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	DMABufferAddr DMA Buffer Address Holds the current buffer address. This register is updated as and when the data transfer for the corresponding end point is in progress. This register is present only in Scatter/Gather DMA mode. Otherwise this field is reserved.

USBOTG_DOEPCCTLn

Address: Operational Base + offset (0x0b20)

Device endpoint-n control register

Bit	Attr	Reset Value	Description
31	R/W SC	0x0	<p>EPEna Endpoint Enable</p> <p>Applies to IN and OUT endpoints. When Scatter/Gather DMA mode is enabled, For IN endpoints this bit indicates that the descriptor structure and data buffer with data ready to transmit is setup. For OUT endpoint it indicates that the descriptor structure and data buffer to receive data is setup. When Scatter/Gather DMA mode is enabled-such as for buffer-pointer based DMA mode: For IN endpoints, this bit indicates that data is ready to be transmitted on the endpoint; For OUT endpoints, this bit indicates that the application has allocated the memory to start receiving data from the USB. The core clears this bit before setting any of the following interrupts on this endpoint: SETUP Phase Done, Endpoint Disabled, Transfer Completed. Note: For control endpoints in DMA mode, this bit must be set to be able to transfer SETUP data packets in memory.</p>
30	R/W SC	0x0	<p>EPDis Endpoint Disable</p> <p>Applies to IN and OUT endpoints. The application sets this bit to stop transmitting/receiving data on an endpoint, even before the transfer for that endpoint is complete. The application must wait for the Endpoint Disabled interrupt before treating the endpoint as disabled. The core clears this bit before setting the Endpoint Disabled interrupt. The application must set this bit only if Endpoint Enable is already set for this endpoint.</p>
29	RO	0x0	<p>SetD1PID Field0001 Abstract</p> <p>Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA1. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Set Odd (micro)frame (SetOddFr). Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to odd (micro)frame. This field is not applicable for Scatter/Gather DMA mode.</p>

Bit	Attr	Reset Value	Description
28	WO	0x0	<p>SetD0PID Set DATA0 PID Applies to interrupt/bulk IN and OUT endpoints only. Writing to this field sets the Endpoint Data PID (DPID) field in this register to DATA0. This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. In non-Scatter/Gather DMA mode: Set Even (micro)frame (SetEvenFr) Applies to isochronous IN and OUT endpoints only. Writing to this field sets the Even/Odd (micro)frame (EO_FrNum) field to even (micro)frame. When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is in the transmit descriptor structure. The frame in which to receive data is updated in receive descriptor structure.</p>
27	WO	0x0	<p>SNAK Set NAK Applies to IN and OUT endpoints. A write to this bit sets the NAK bit for the endpoint. Using this bit, the application can control the transmission of NAK handshakes on an endpoint. The core can also set this bit for OUT endpoints on a Transfer Completed interrupt, or after a SETUP is received on the endpoint.</p>
26	WO	0x0	<p>CNAK Clear NAK Applies to IN and OUT endpoints. A write to this bit clears the NAK bit for the endpoint.</p>
25:22	RW	0x0	<p>TxFNum Tx FIFO Number Shared FIFO Operation: non-periodic endpoints must set this bit to zero. Periodic endpoints must map this to the corresponding Periodic Tx FIFO number. 4'h0: Non-Periodic Tx FIFO; Others: Specified Periodic Tx FIFO number. Note: An interrupt IN endpoint can be configured as a non-periodic endpoint for applications such as mass storage. The core treats an IN endpoint as a non-periodic endpoint if the TxFNum field is set to 0. Otherwise, a separate periodic FIFO must be allocated for an interrupt IN endpoint using coreConsultant, and the number of this FIFO must be programmed into the TxFNum field. Configuring an interrupt IN endpoint as a non-periodic endpoint saves the extra periodic FIFO area. Dedicated FIFO Operation: these bits specify the FIFO number associated with this endpoint. Each active IN endpoint must be programmed to a separate FIFO number. This field is valid only for IN endpoints.</p>

Bit	Attr	Reset Value	Description
21	RW	0x0	<p>Stall STALL Handshake</p> <p>Applies to non-control, non-isochronous IN and OUT endpoints only. The application sets this bit to stall all tokens from the USB host to this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority. Only the application can clear this bit, never the core.</p> <p>Applies to control endpoints only. The application can only set this bit, and the core clears it, when a SETUP token is received for this endpoint. If a NAK bit, Global Non-periodic IN NAK, or Global OUT NAK is set along with this bit, the STALL bit takes priority.</p> <p>Irrespective of this bit's setting, the core always responds to SETUP data packets with an ACK handshake.</p>
20	RW	0x0	<p>Snp Snoop Mode</p> <p>Applies to OUT endpoints only. This bit configures the endpoint to Snoop mode. In Snoop mode, the core does not check the correctness of OUT packets before transferring them to application memory.</p>
19:18	RW	0x0	<p>EPType Endpoint Type</p> <p>Applies to IN and OUT endpoints. This is the transfer type supported by this logical endpoint.</p> <p>2'b00: Control 2'b01: Isochronous 2'b10: Bulk 2'b11: Interrupt</p>
17	RO	0x0	<p>NAKsts NAK Status</p> <p>Applies to IN and OUT endpoints. Indicates the following:</p> <p>1'b0: The core is transmitting non-NAK handshakes based on the FIFO status.</p> <p>1'b1: The core is transmitting NAK handshakes on this endpoint. When either the application or the core sets this bit: The core stops receiving any data on an OUT endpoint, even if there is space in the RxFIFO to accommodate the incoming packet.</p>

Bit	Attr	Reset Value	Description
16	RO	0x0	<p>DPID Endpoint Data PID Applies to interrupt/bulk IN and OUT endpoints only. Contains the PID of the packet to be received or transmitted on this endpoint. The application must program the PID of the first packet to be received or transmitted on this endpoint, after the endpoint is activated. The applications use the SetD1PID and SetD0PID fields of this register to program either DATA0 or DATA1 PID.</p> <p>1'b0: DATA0 1'b1: DATA1 This field is applicable both for Scatter/Gather DMA mode and non-Scatter/Gather DMA mode. Even/Odd (Micro)Frame (EO_FrNum). In non-Scatter/Gather DMA mode: Applies to isochronous IN and OUT endpoints only. Indicates the (micro)frame number in which the core transmits/receives isochronous data for this endpoint. The application must program the even/odd (micro) frame number in which it intends to transmit/receive isochronous data for this endpoint using the SetEvnFr and SetOddFr fields in this register.</p> <p>1'b0: Even (micro)frame 1'b1: Odd (micro)frame When Scatter/Gather DMA mode is enabled, this field is reserved. The frame number in which to send data is provided in the transmit descriptor structure. The frame in which data is received is updated in receive descriptor structure.</p>
15	R/W SC	0x0	<p>USBActEP USB Active Endpoint Applies to IN and OUT endpoints. Indicates whether this endpoint is active in the current configuration and interface. The core clears this bit for all endpoints (other than EP 0) after detecting a USB reset. After receiving the SetConfiguration and SetInterface commands, the application must program endpoint registers accordingly and set this bit.</p>
14:11	RW	0x0	<p>NextEp Next Endpoint Applies to non-periodic IN endpoints only. Indicates the endpoint number to be fetched after the data for the current endpoint is fetched. The core can access this field, even when the Endpoint Enable (EPEna) bit is low. This field is not valid in Slave mode operation. Note: This field is valid only for Shared FIFO operations.</p>
10:0	RW	0x000	<p>MPS Maximum Packet Size Applies to IN and OUT endpoints. The application must program this field with the maximum packet size for the current logical endpoint. This value is in bytes.</p>

USBOTG_PCGCR

Address: Operational Base + offset (0x0b24)

Power and clock gating control register

Bit	Attr	Reset Value	Description
31:14	RW	0x0802e	<p>RestoreValue Restore Value (Applicable only when Hibernation is enabled (OTG_EN_PWROPT=2). Defines port clock select for different speeds.</p> <p>[31] if_dev_mode</p> <ul style="list-style-type: none"> - 1: Device mode, core restored as device - 0: Host mode, core restored as host <p>[30:29] p2hd_prt_spd (PRT speed)</p> <ul style="list-style-type: none"> - 00: HS - 01: FS - 10: LS - 11: Reserved <p>[28:27] p2hd_dev_enum_spd (Device enumerated speed)</p> <ul style="list-style-type: none"> - 00: HS - 01: FS (30/60 MHz clk) - 10: LS - 11: FS (48 MHz clk) <p>[26:20] mac_dev_addr (MAC device address) Device address</p> <p>[19] mac_termselect (Termination selection)</p> <ul style="list-style-type: none"> - 0: HS_TERM (Program for High Speed) - 1: FS_TERM (Program for Full Speed) <p>[18:17] mac_xcvrselect (Transceiver select)</p> <ul style="list-style-type: none"> - 00: HS_XCVR (High Speed) - 01: FS_XCVR (Full Speed) - 10: LS_XCVR (Low Speed) - 11: LFS_XCVR (Reserved) <p>[16] sh2pl_prt_ctl[0]</p> <ul style="list-style-type: none"> - 1: prt_power enabled - 0: prt_power disabled <p>[15:14] prt_clk_sel (Refer prt_clk_sel table)</p>
13	RW	0x0	<p>EssRegRestored Essential Register Values Restored (Applicable only when Hibernation is enabled (OTG_EN_PWROPT=2). When a value of 1 is written to this field, it indicates that register values of essential registers have been restored.</p>
12:10	RO	0x0	reserved

Bit	Attr	Reset Value	Description
9	RO	0x0	<p>RestoreMode Restore Mode (Applicable only when Hibernation is enabled (OTG_EN_PWROPT=2). The application should program this bit to specify the restore mode during RESTORE POINT before programming PCGCCTL.EssRegRest bit is set.</p> <p>Host Mode: 1'b0: Host Initiated Resume, Host Initiated Reset 1'b1: Device Initiated Remote Wake up</p> <p>Device Mode: 1'b0: Device Initiated Remote Wake up 1'b1: Host Initiated Resume, Host Initiated Reset</p>
8	RW	0x0	<p>ResetAfterSusp Reset After Suspend Applicable in Partial power-down mode. In partial power-down mode of operation, this bit needs to be set in host mode before clamp is removed if the host needs to issue reset after suspend. If this bit is not set, then the host issues resume after suspend. This bit is not applicable in device mode and non-partial power-down mode. In Hibernation mode, this bit needs to be set at RESTORE_POINT before PCGCCTL.EssRegRestored is set. In this case, PCGCCTL.restore_mode needs to be set to wait_restore.</p>
7	RO	0x0	<p>L1Suspended Deep Sleep This bit indicates that the PHY is in deep sleep when in L1 state.</p>
6	RO	0x0	<p>PhySleep PHY in Sleep This bit indicates that the PHY is in the Sleep state.</p>
5	RW	0x0	<p>Enbl_L1Gating Enable Sleep Clock Gating When this bit is set, core internal clock gating is enabled in Sleep state if the core cannot assert utmi_l1_suspend_n. When this bit is not set, the PHY clock is not gated in Sleep state.</p>
4	RO	0x0	reserved
3	RW	0x0	<p>RstPdwnModule Reset Power-Down Modules This bit is valid only in Partial Power-Down mode. The application sets this bit when the power is turned off. The application clears this bit after the power is turned on and the PHY clock is up.</p>

Bit	Attr	Reset Value	Description
2	RW	0x0	PwrClmp Power Clamp This bit is valid only in Partial Power-Down mode (OTG_EN_PWRDPT = 1). The application sets this bit before the power is turned off to clamp the signals between the power-on modules and the power-off modules. The application clears the bit to disable the clamping before the power is turned on.
1	RW	0x0	GateHclk Gate Hclk The application sets this bit to gate hclk to modules other than the AHB Slave and Master and wakeup logic when the USB is suspended or the session is not valid. The application clears this bit when the USB is resumed or a new session starts.
0	RW	0x0	StopPclk Stop Pclk The application sets this bit to stop the PHY clock (phy_clk) when the USB is suspended, the session is not valid, or the device is disconnected. The application clears this bit when the USB is resumed or a new session starts.

USBOTG_EPBUFO

Address: Operational Base + offset (0x1000)

Device endpoint 0 / host out channel 0 address

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	EPBUFO From 0x1000 to 0x2000, EPBUF for endport0

USBOTG_EPBUF1

Address: Operational Base + offset (0x2000)

Device endpoint 1 / host out channel 1 address

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	EPBUF1 From 0x2000 to 0x3000, EPBUF for endport1

USBOTG_EPBUF2

Address: Operational Base + offset (0x3000)

Device endpoint 2 / host out channel 2 address

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved

Bit	Attr	Reset Value	Description
0	RW	0x0	EPBUF2 From0x3000 to 0x4000, EPBUF for endport2

USBOTG_EPBUF3

Address: Operational Base + offset (0x4000)

Device endpoint 3 / host out channel 3 address

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	EPBUF3 From0x4000 to 0x5000, EPBUF for endport3

USBOTG_EPBUF4

Address: Operational Base + offset (0x5000)

Device endpoint 4 / host out channel 4 address

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	EPBUF4 From0x5000 to 0x6000, EPBUF for endport4

USBOTG_EPBUF5

Address: Operational Base + offset (0x6000)

Device endpoint 5 / host out channel 5 address

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	EPBUF5 From0x6000 to 0x7000, EPBUF for endport5

USBOTG_EPBUF6

Address: Operational Base + offset (0x7000)

Device endpoint 6 / host out channel 6 address

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	EPBUF6 From0x7000 to 0x8000, EPBUF for endport6

USBOTG_EPBUF7

Address: Operational Base + offset (0x8000)

Device endpoint 7 / host out channel 7 address

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	EPBUF7 From0x8000 to 0x9000, EPBUF for endport7

2.7 Interface description

Table 2-1 USB OTG 2.0 Interface Description

Module Pin	Direction	Pad Name	pinmux
VSSA	AG	VSSA	-
VCCA3P3	AP	VCCA3P3	-
VCCCORE1P2	AP	VCCCORE1P2	-
USB0PN	A	USB0PN	-
USBRBIAS	A	USBRBIAS	-
USB0PP	A	USB0PP	-
VBUS_0	A	VBUS_0	-
USB0ID	A	USB0ID	-

Note: **A**—Analog pad ; **AP**—Analog power; **AG**—Analog ground ;**DP**—Digital power ;**DG**—Digital ground;

2.8 Application Note

2.8.1 Suspend Mode

When PHY is in suspend state

- COMMONONN = 1'b1, 480M clock invalid
- COMMONONN = 1'b0, 480M clock output available.

Please refer to "Chapter GRF" for configuration details

2.8.2 Reset a port

Please refer to "Chapter CRU" for more details.

2.8.3 Relative GRF Registers

USBPHY contains 256 bit registers to configure USB PHY. These bits are used to adjust DP/DM SI. Please refer to "Chapter GRF" for more details.

Chapter 3 USB Host 2.0

3.1 Overview

USB HOST2.0 supports Non_OTG Host functions and is fully compliant with USB2.0 specification, and support high-speed (480Mbps), full-speed (12Mbps), low-speed (1.5Mbps) transfer. It is optimized for point-to-point applications (no hub, direct connection to device).

3.1.1 Feature

- Compliant with the USB2.0 Specification
- Operates in Non_OTG Host mode
- Operates in High-Speed, Full-Speed, Low-speed mode
- Supports 16 channels in host mode
- Built-in one 840x35 bits FIFO
- Internal DMA with no scatter/gather function

3.2 Block Diagram

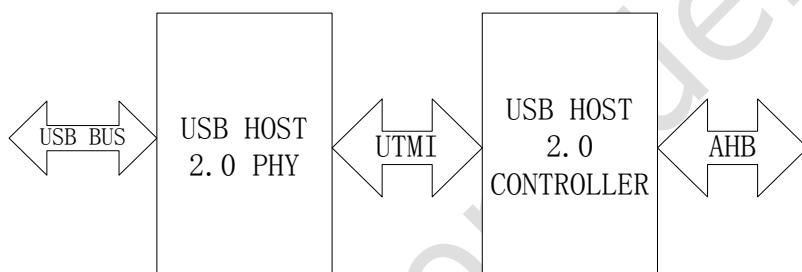


Fig. 3-1 USB HOST 2.0 Architecture

USB HOST 2.0 is broken up into two separate units: USB HOST 2.0 controller and USB HOST 2.0 PHY. The two units are interconnected with 16-bits UTMI interface.

3.3 USB Host2.0 Controller

Much the same as USB OTG with no Device Mode supported. See Chapter OTG for more information.

3.4 USB Host2.0 PHY

Much the same as USB OTG with no Device Mode supported. See Chapter OTG for more information.

USB Host2.0 PHY doesn't support UART-DEBUG function.

3.5 Register Description

The Registers are much the same as OTG with no Device-Mode supported. The registers of Device are not available. See Chapter OTG for more information.

3.6 Interface description

Table 3-1 USB HOST 2.0 Interface Description

Module Pin	Direction	Pad Name	pinmux
VSSA	AG	VSSA	-
VCCA3P3	AP	VCCA3P3	-
VCCCORE1P1	AP	VCCCORE1P1	-
USB1PN	A	USB1PN	-

Module Pin	Direction	Pad Name	pinmux
USBRBIAS	A	USBRBIAS	-
USB1PP	A	USB1PP	-

Note: **A**—Analog pad ; **AP**—Analog power; **AG**—Analog ground ;**DP**—Digital power ;**DG**—Digital ground;

3.7 Application Note

See Chapter OTG for more information.

Reset a port

CRU_SOFTRST4_CON contains HOST reset signal description. Please refer to "Chapter CRU" for more details.

Relative GRF Registers

GRF_UOC0_CON0 ~ GRF_UOC0_CON2 is OTG PHY register.

GRF_UOC0_CON3 is OTG Controller register.

Please refer to "Chapter GRF" for more details.

Chapter 4 HDMI TX

4.1 Overview

HDMI TX is fully compliant with HDMI 1.4a specification. It offers a simple implementation for consumer electronics like DVD/player/recorder and camcorder. HDMI TX consists of one HDMI transmitter controller and one HDMI transmitter PHY.

It supports the following features:

- HDMI 1.4a/b/1.3/1.2/1.1, HDCP 1.2 and DVI 1.0 standard compliant transmitter
- Supports data rate from 25MHz, 1.65bps up to 3.4Gbps over a Single channel HDMI
- TMDS Tx Drivers with programmable output swing, resister values and pre-emphasis
- Supports all DTV resolutions including 480p/576p/720p/1080p
- Digital video interface supports a pixel size of 24bits color depth in RGB
- S/PDIF output supports PCM, Dolby Digital, DTS digital audio transmission (32-192kHz Fs) using IEC60958 and IEC 61937
- Multiphase 4MHz fixed bandwidth PLL with low jitter
- DDC Bus I2C master interface at 3.3V
- HDCP encryption and decryption engine contains all the necessary logic to encrypt the incoming audio and video data
- Lower power operation with optimal power management feature
- The EDID and CEC function are also supported by HDMI Transmitter Controller
- Monitor Detection supported through Hot Plug
- 3.3V high speed I/O and 1.2V core power supply

4.2 Block Diagram

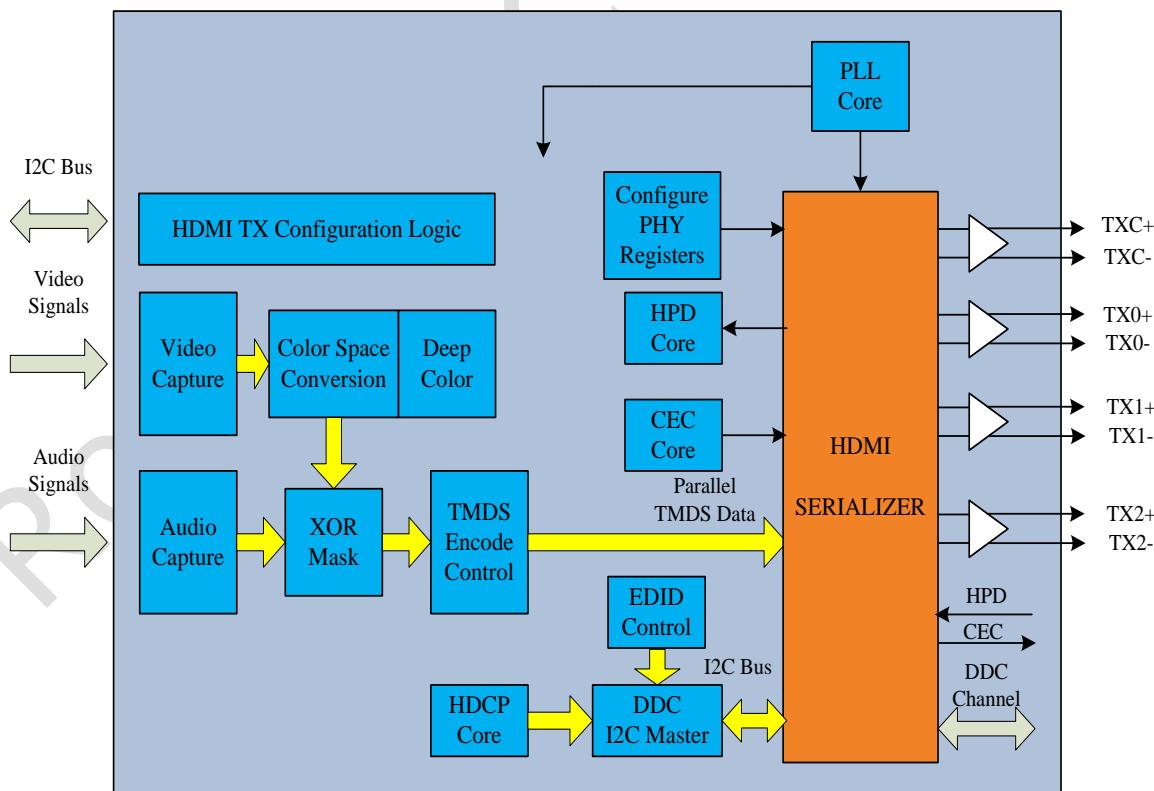


Fig. 4-1 HDMI TX Block Diagram

4.3 Function Description

4.3.1 Video Data Processing

The video processing contain video format timings, pixel encodings(RGB to YCbCr, or YCbCr to RGB), colorimetry and corresponding requirements. This function is implemented by some functional blocks, Video Capture block, Color Space Conversion block, and Deep Color block.

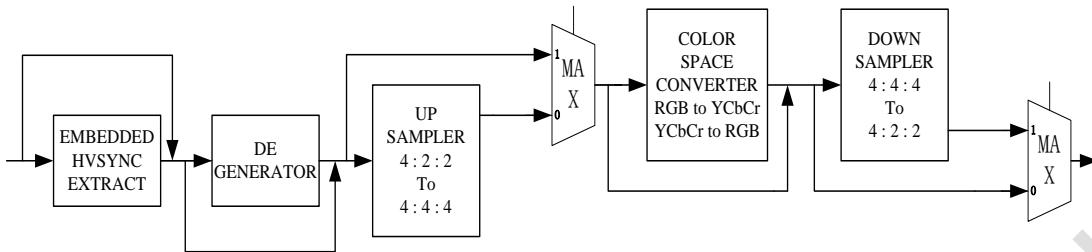


Fig. 4-2 HDMI TX Video Data Processing

The input video pixels can be encoded in either RGB, YCBCR 4:4:4 or YCBCR 4:2:2 formats by Color Space Conversion block.

The input Video data can have a pixel size of 24 bits. The deep color block is used to deal with different pixel size. Video at the 24-bit color depth is carried at a TMDS clock rate equal to the pixel clock rate.

The following interface timing diagram outlines the Video interface signal format. 24 bit data in RGB can be captured by the rising edge of VCLK with 1ns setup time and 1ns hold time requirements. Control signals such DE and VSync/HSync/FSync going with the same timing relationship.

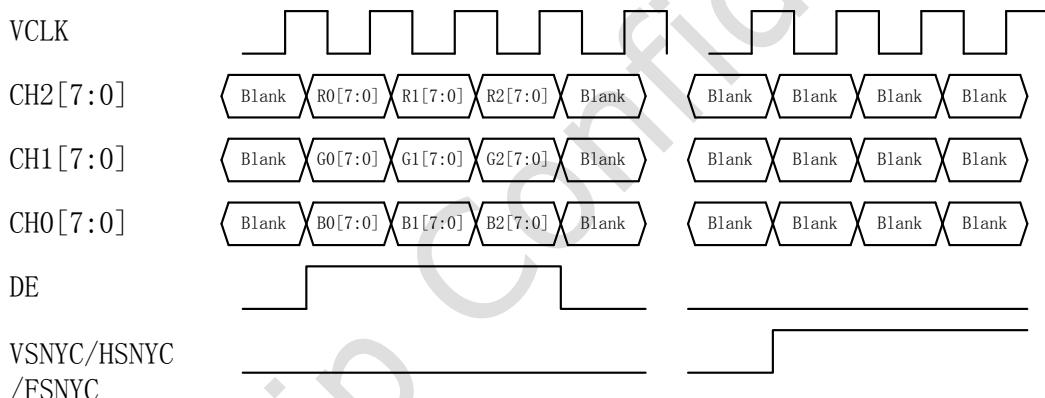


Fig. 4-3 HDMI TX Video Processing Timing

1. Video Data Capture Logic

HDMI TX support input video data related format table is listed below.

Table 4-1 HDMI TX Supported Input Video Formats

Color Space	Pixel Encoding	Sync	Channel Width	Pin Nums
RGB	4:4:4	Separate	8	24

2. Embedded Sync Extraction Module

The module is used to extract Vsync and Hsync signals from input video data stream such as ITU656 format. With setting the relative registers, this functional module can extract correct video sync signals for later process block using.

3. Data Enable (DE) Generator

HDMI Transmitter has DE signal generator by incoming HSYNCs, VSYNCs and Video clock. External DE is optional and selected by appropriate register settings. This feature is particularly useful when interfacing to MPEG decoders that do not provide a specific DE output signal.

4. Color Space Conversion

HDMI Transmitter Color space conversion (CSC) is available to interface for several MPEG decoders like with YCbCr-only outputs, and to provide full DVI backwards compatibility. The function of this module is to convert RGB input Video data to YCbCr Video data.

4.3.2 Audio Data Processing

The HDMI TX audio process contain audio clock regeneration, placement of audio samples within packets, packet timing control, audio sample rates setting, and channel/speaker assignments. This function is implemented by Audio Capture blocks

The Audio Capture support either SPDIF or four channel I2S input. SPDIF input supports audio sampling rates from 32 to 192 KHz. The I2S input supports from 2-channel to 8-channel audio up to 192 KHz.

The scheme of audio processing as shown in the figure below:

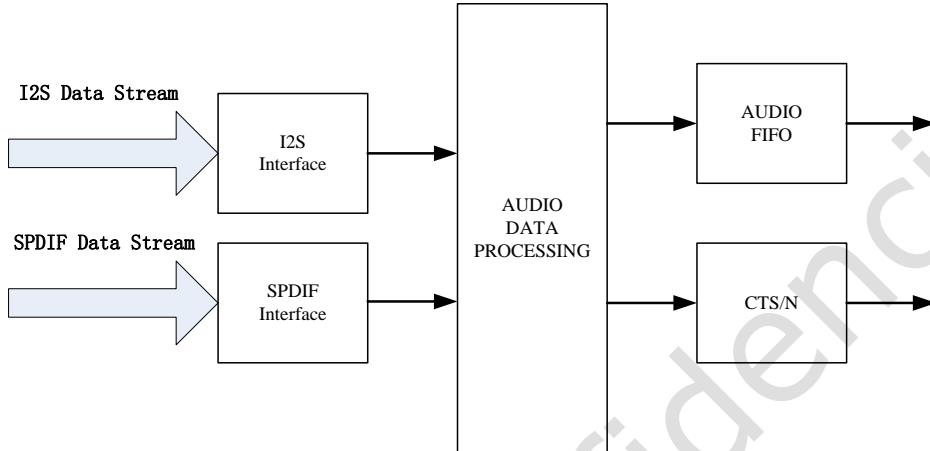


Fig. 4-4 HDMI TX Audio Data Processing Diagram

1. I2S

The function of this module is to implement I2S audio input feature. The incoming audio stream is captured, processed then transmitted into the TMDS link. Four I2S inputs also allow transmission of DVD-Audio and decoded Dolby Digital to A/V Receivers and high-end displays. The interface supports from 2-channel to 8-channel audio up to 192 kHz. The I2S pins must also be coherent with mclk. The appropriate registers must be configured to describe the format of audio being input. This information is passed over the HDMI link in the CEA-861D Audio Info (AI) packets. Table shows the I2S 8 channel audio formats that are supported for each of the video formats.

Table 4-2 HDMI TX I2S 2 Channel Audio Sampling Frequency at Each Video Format

Video Format	32kHz	44.1kHz	48kHz	88.2kHz	96kHz	176.4kHz	192kHz
720x480p /720x576p	Yes	Yes	Yes	Yes	Yes	Yes	Yes
720p	Yes	Yes	Yes	Yes	Yes	Yes	Yes
1080p	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Table 4-3 HDMI TX I2S 8 Channel Audio Sampling Frequency at Each Video Format

Video Format	32kHz	44.1kHz	48kHz	88.2kHz	96kHz	176.4kHz	192kHz
720x480p /720x576p	Yes	Yes	Yes	No	No	No	No
720p	Yes	Yes	Yes	Yes	Yes	Yes	Yes
1080p	Yes	Yes	Yes	Yes	Yes	Yes	Yes

2. SPDIF

The function of this module is to implement SPDIF audio input feature. The incoming audio stream is captured, processed then transmitted into the TMDS link. SPDIF stream can carry 2-channel uncompressed PCM data (IEC 60958) or a compressed bit stream for multi-channel (IEC 61937) formats. The audio data capture logic forms the audio data into packets in accordance with the HDMI specification. SPDIF input supports audio sampling rates from 32 to 192 KHz. The following shows the SPDIF audio formats that are supported for each of the video formats

Table 4-4 HDMI TX SPDIF Sampling Frequency at Each Video Format

Video Format	32kHz	44.1kHz	48kHz	88.2kHz	96kHz	176.4kHz	192kHz
720x480p /720x576p	Yes	Yes	Yes	Yes	Yes	No	No
720p	Yes	Yes	Yes	Yes	Yes	Yes	Yes
1080p	Yes	Yes	Yes	Yes	Yes	Yes	Yes

3. Audio Sample Clock Capture and Regeneration

Audio data being carried across the HDMI link, which is driven by a TMDS clock running at a rate corresponding to the video pixel rate, does not retain the original audio sample clock. The task of recreating this clock at the Sink is called Audio Clock Regeneration.

The HDMI Transmitter determine the fractional relationship between the TMDS clock and an audio reference clock (128 audio sample rate [fs]) and pass the numerator and denominator of that fraction to the HDMI Sink across the HDMI link. The Sink then re-create the audio clock from the TMDS clock by using a clock divider and a clock multiplier.

The exact relationship between the two clocks will be.

$$128 \cdot f_s = f_{TMDS_clock} \cdot N / CTS.$$

The scheme of the Audio Sample Clock Capture and Regeneration as shown below:

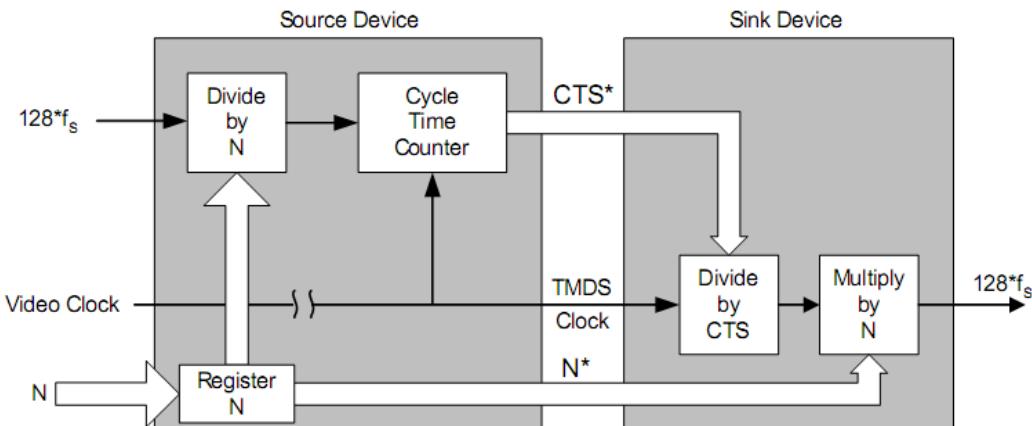


Fig. 4-5 HDMI TX Audio Clock Regeneration Model

4.3.3 DDC

The DDC functional block is used for configuration and status exchange between the HDMI Source and HDMI Sink. HDMI Transmitter Controller has I2C Master Interface for DDC transactions. It enables for host controller to read EDID, HDCP authentication by issuing simple register access. The I2C bus speed is limited by DDC specification. DDC bus access frequency can be controlled.

4.3.4 EDID

Extended Display Identification Data (EDID) was created by VESA to enable plug and play capabilities of monitors. This data, which is stored in the sink device, describes video formats that the DTV Monitor is capable of receiving and rendering. The information is supplied to the source device, over the interface, upon the request of the source device. The source device then chooses its output format, taking into account the format of the original video stream and the formats supported by the DTV Monitor. The function of this module is to implement EDID feature.

4.3.5 HDCP

HDMI Transmitter has a capability for HDCP authentication by hardware. The function of this module is to implement HDCP encryption feature. This feature can be turned on or off depending on register setting.

4.3.6 Hot Plug Detect

HDMI Transmitter has a capability for detecting the Sink plug in or plug out, and launch an interrupt and registers state indicating for software controlling.

4.3.7 TMDS encoder

The TMDS encoder converts the 2/4/8 bits data into the 10 bit DC-balanced TMDS data. HDMI TX put the TMDS encoding on the audio /video /aux data received from the HDCP XOR mask. This data is output onto three TMDS differential data lines along with a TMDS differential clock.

4.3.8 CEC

The CEC functional block provides high-level control functions between all of the various audiovisual products in a user's environment through one line.

4.4 Register Description

4.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
HDMI_reg00	0x0000	B	0x67	Register00
HDMI_reg01	0x0004	B	0x01	Register01
HDMI_reg02	0x0008	B	0x30	Register02
HDMI_reg04	0x0010	B	0x08	Register04
HDMI_reg05	0x0014	B	0x00	Register05
HDMI_reg08	0x0020	B	0x00	Register08
HDMI_reg09	0x0024	B	0x00	Register09
HDMI_reg0a	0x0028	B	0x00	Register0a
HDMI_reg0b	0x002c	B	0x00	Register0b
HDMI_reg0c	0x0030	B	0x00	Register0c
HDMI_reg0d	0x0034	B	0x00	Register0d
HDMI_reg0e	0x0038	B	0x00	Register0e
HDMI_reg0f	0x003c	B	0x00	Register0f
HDMI_reg10	0x0040	B	0x00	Register10
HDMI_reg11	0x0044	B	0x00	Register11
HDMI_reg12	0x0048	B	0x00	Register12
HDMI_reg13	0x004c	B	0x00	Register13
HDMI_reg14	0x0050	B	0x00	Register14
HDMI_reg15	0x0054	B	0x00	Register15
HDMI_reg35	0x00d4	B	0x01	Register35
HDMI_reg37	0x00dc	B	0x00	Register37
HDMI_reg38	0x00e0	B	0x3c	Register38
HDMI_reg39	0x00e4	B	0x00	Register39
HDMI_reg3a	0x00e8	B	0x00	Register3a
HDMI_reg3f	0x00fc	B	0x00	Register3f
HDMI_reg40	0x0100	B	0x18	Register40
HDMI_reg41	0x0104	B	0x00	Register41
HDMI_reg45	0x0114	B	0x00	Register45
HDMI_reg46	0x0118	B	0x00	Register46
HDMI_reg47	0x011c	B	0x00	Register47
HDMI_reg4a	0x0128	B	0x00	Register4a
HDMI_reg4b	0x012c	B	0x40	Register4b
HDMI_reg4c	0x0130	B	0x00	Register4c

Name	Offset	Size	Reset Value	Description
HDMI_reg4d	0x0134	B	0x00	Register4d
HDMI_reg4e	0x0138	B	0x00	Register4e
HDMI_reg4f	0x013c	B	0x00	Register4f
HDMI_reg50	0x0140	B	0x00	Register50
HDMI_reg52	0x0148	B	0x12	Register52
HDMI_reg53	0x014c	B	0x04	Register53
HDMI_reg57	0x015c	B	0x20	Register57
HDMI_reg58	0x0160	B	0x00	Register58
HDMI_reg63	0x018c	B	0x26	Register63
HDMI_reg65	0x0194	B	0x00	Register65
HDMI_reg66	0x0198	B	0x00	Register66
HDMI_reg67	0x019c	B	0x00	Register67
HDMI_reg68	0x01a0	B	0x00	Register68
HDMI_reg69	0x01a4	B	0x00	Register69
HDMI_reg6a	0x01a8	B	0x00	Register6a
HDMI_reg6c	0x01b0	B	0x00	Register6c
HDMI_reg95	0x0254	B	0x00	Register95
HDMI_reg96	0x0258	B	0x00	Register96
HDMI_reg97	0x025c	B	0x00	Register97
HDMI_reg98	0x0260	B	0x03	Register98
HDMI_reg9c	0x0270	B	0x00	Register9c
HDMI_reg9e	0x0278	B	0x01	Register9e
HDMI_reg9f	0x027c	B	0x00	Register9f
HDMI_rega0	0x0280	B	0x00	Registera0
HDMI_rega1	0x0284	B	0x00	Registera1
HDMI_rega2	0x0288	B	0x00	Registera2
HDMI_rega3	0x028c	B	0x00	Registera3
HDMI_rega4	0x0290	B	0x00	Registera4
HDMI_rega5	0x0294	B	0x00	Registera5
HDMI_rega6	0x0298	B	0x00	Registera6
HDMI_rega7	0x029c	B	0x00	Registera7
HDMI_rega8	0x02a0	B	0x00	Registera8
HDMI_rega9	0x02a4	B	0x00	Registera9
HDMI_regaa	0x02a8	B	0x00	Registeraa
HDMI_regab	0x02ac	B	0x00	Registerab
HDMI_regac	0x02b0	B	0x00	Registerac
HDMI_regad	0x02b4	B	0x00	Registerad
HDMI_regae	0x02b8	B	0x00	Registerae
HDMI_regaf	0x02bc	B	0x00	Registeraf
HDMI_regb0	0x02c0	B	0x00	Registerb0
HDMI_regb1	0x02c4	B	0x00	Registerb1
HDMI_regb2	0x02c8	B	0x00	Registerb2
HDMI_regb3	0x02cc	B	0x00	Registerb3
HDMI_regb4	0x02d0	B	0x00	Registerb4

Name	Offset	Size	Reset Value	Description
HDMI_regb5	0x02d4	B	0x00	Registerb5
HDMI_regb6	0x02d8	B	0x00	Registerb6
HDMI_regb7	0x02dc	B	0x00	Registerb7
HDMI_regb8	0x02e0	B	0x00	Registerb8
HDMI_regb9	0x02e4	B	0x00	Registerb9
HDMI_regba	0x02e8	B	0x00	Registerba
HDMI_regbb	0x02ec	B	0x00	Registerbb
HDMI_regbc	0x02f0	B	0x00	Registerbc
HDMI_regbd	0x02f4	B	0x00	Registerbd
HDMI_Regbe	0x02f8	B	0x00	Registerbe
HDMI_regc0	0x0300	B	0xc0	Registerc0
HDMI_regc1	0x0304	B	0x00	Registerc1
HDMI_regc2	0x0308	B	0x78	Registerc2
HDMI_regc3	0x030c	B	0x00	Registerc3
HDMI_regc4	0x0310	B	0x00	Registerc4
HDMI_regc5	0x0314	B	0x00	Registerc5
HDMI_regc8	0x0320	B	0x00	Registerc8
HDMI_regc9	0x0324	B	0x50	Registerc9
HDMI_Regce	0x0338	B	0x01	Registerce
HDMI_regd0	0x0340	B	0x00	Registerd0
HDMI_regd1	0x0344	B	0x00	Registerd1
HDMI_regd2	0x0348	B	0x00	Registerd2
HDMI_regd3	0x034c	B	0x00	Registerd3
HDMI_regd4	0x0350	B	0x03	Registerd4
HDMI_regd5	0x0354	B	0x09	Registerd5
HDMI_regd6	0x0358	B	0x03	Registerd6
HDMI_regd7	0x035c	B	0x00	Registerd7
HDMI_regd8	0x0360	B	0xff	Registerd8
HDMI_regd9	0x0364	B	0xff	Registerd9
HDMI_Regda	0x0368	B	0x00	Registerda
HDMI_Regdb	0x036c	B	0x00	Registerdb
HDMI_Regdc	0x0370	B	0xd0	Registerdc
HDMI_Regdd	0x0374	B	0x20	Registerdd
HDMI_Regde	0x0378	B	0x00	Registerde
HDMI_Rege0	0x0380	B	0x23	Registere0
HDMI_Rege1	0x0384	B	0x0f	Registere1
HDMI_Rege2	0x0388	B	0xaa	Registere2
HDMI_Rege3	0x038c	B	0x0f	Registere3
HDMI_Rege7	0x039c	B	0x1e	Registere7
HDMI_Rege8	0x03a0	B	0x00	Registere8
HDMI_Reged	0x03b4	B	0x03	Registered

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

4.4.2 Detail Register Description

HDMI_reg00

Address: Operational Base + offset (0x0000)

Register00

Bit	Attr	Reset Value	Description
7	RW	0x0	reserved
6	RW	0x1	sw_reset_ana Soft reset for analog part 1'b0: reset 1'b1: not
5	RW	0x1	sw_reset_dig Soft reset for digital part 1'b0: reset 1'b1: not
4	RW	0x0	reserved
3	RW	0x0	vclk_inv Vclk invert select 1'b0: not invert 1'b1: invert
2	RW	0x1	reserved
1	RW	0x1	pd_dig System power down for digital function 1'b0: not 1'b1: power down
0	RW	0x1	interrupt_polarity Interrupt polarity 1'b1: Active High 1'b0: Active low

HDMI_reg01

Address: Operational Base + offset (0x0004)

Register01

Bit	Attr	Reset Value	Description
7:4	RW	0x0	reserved
3:1	RW	0x0	input_video_fmt Input video format 3'b000: RGB and YCbCr 4:4:4 3'b101: DDR RGB 4:4:4 or YCbCr 4:4:4 3'b110: DDR YCbCr 4:2:2
0	RW	0x1	de_sel DE select 1'b0: internal DE 1'b1: external DE

HDMI_reg02

Address: Operational Base + offset (0x0008)

Register02

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
7:6	RW	0x0	video_output_fmt Video output format 2'b00: RGB 4:4:4 2'b01: YCbCr 4:4:4 2'b10: YCbCr 4:2:2
5:4	RW	0x3	data_width_422 Data width for 4:2:2 input 2'b00: 12 bits 2'b01: 10 bits 2'b11: 8 bits
3:1	RW	0x0	reserved
0	RW	0x0	Video input color space 1'b0: RGB 1'b1: YCbCr

HDMI_reg04

Address: Operational Base + offset (0x0010)

Register04

Bit	Attr	Reset Value	Description
7:4	RW	0x0	reserved
3	RW	0x1	sof_sel After 1st SOF(the first edge of the Vsync) for external DE sample 1'b0: after SOF 1'b1: Not
2:1	RW	0x0	reserved
0	RW	0x0	csc_en CSC (Color Space Convert) enable. 1'b0: no CSC 1'b1: enable CSC

HDMI_reg05

Address: Operational Base + offset (0x0014)

Register05

Bit	Attr	Reset Value	Description
7	RW	0x0	clear_avmute Clear avmute
6	RW	0x0	set_avmute Set avmute
5:2	RW	0x0	reserved
1	RW	0x0	audio_mute Audio mute
0	RW	0x0	video_black Video black

HDMI_reg08

Address: Operational Base + offset (0x0020)

Register08

Bit	Attr	Reset Value	Description
7:4	RW	0x0	reserved
3	RW	0x0	vs_polarity VSYNC polarity 1'b0: Negative 1'b1: Positive
2	RW	0x0	hs_polarity HSYNC polarity 1'b0: Negative 1'b1: Positive
1	RW	0x0	interlace_progressiv_sel Interlace/Progressive 1'b0: Progressive 1'b1: Interlace
0	RW	0x0	ext_video_para_set_en External video parameter setting enable 1'b0: disable 1'b1: enable

HDMI_reg09

Address: Operational Base + offset (0x0024)

Register09

Bit	Attr	Reset Value	Description
7:0	RW	0x00	ext_hs_total_part1 External horizontal total part1. It is not need to be set at normal resolution. Just for special resolution or test purpose.

HDMI_reg0a

Address: Operational Base + offset (0x0028)

Register0a

Bit	Attr	Reset Value	Description
7:5	RW	0x0	reserved
4:0	RW	0x00	ext_hs_total_part2 External horizontal total part2. It is not need to be set at normal resolution. Just for special resolution or test purpose.

HDMI_reg0b

Address: Operational Base + offset (0x002c)

Register0b

Bit	Attr	Reset Value	Description
7:0	RW	0x00	ext_hs_blank_part1 External horizontal blank part1. It is not need to be set at normal resolution. Just for special resolution or test purpose.

HDMI_reg0c

Address: Operational Base + offset (0x0030)

Register0c

Bit	Attr	Reset Value	Description
7:3	RW	0x0	reserved

Bit	Attr	Reset Value	Description
2:0	RW	0x0	ext_hs_blank_part2 External horizontal blank part2. It is not need to be set at normal resolution. Just for special resolution or test purpose.

HDMI_reg0d

Address: Operational Base + offset (0x0034)

Register0d

Bit	Attr	Reset Value	Description
7:0	RW	0x00	Ext_hs_hback_part1 External horizontal Hsync plus Hback part1. It is not need to be set at normal resolution. Just for special resolution or test purpose.

HDMI_reg0e

Address: Operational Base + offset (0x0038)

Register0e

Bit	Attr	Reset Value	Description
7:2	RW	0x0	reserved
1:0	RW	0x0	Ext_hs_hback_part2 External horizontal Hsync plus Hback part2. It is not need to be set at normal resolution. Just for special resolution or test purpose.

HDMI_reg0f

Address: Operational Base + offset (0x003c)

Register0f

Bit	Attr	Reset Value	Description
7:0	RW	0x00	ext_hs_sync_part1 External horizontal duration part1. It is not need to be set at normal resolution. Just for special resolution or test purpose

HDMI_reg10

Address: Operational Base + offset (0x0040)

Register10

Bit	Attr	Reset Value	Description
7:2	RW	0x0	reserved
1:0	RW	0x0	ext_hs_sync_part2 External horizontal duration part2. It is not need to be set at normal resolution. Just for special resolution or test purpose

HDMI_reg11

Address: Operational Base + offset (0x0044)

Register11

Bit	Attr	Reset Value	Description
7:0	RW	0x00	ext_vertical_total_part1 External vertical total part1. It is not need to be set at normal resolution. Just for special resolution or test purpose

HDMI_reg12

Address: Operational Base + offset (0x0048)

Register12

Bit	Attr	Reset Value	Description
7:4	RW	0x0	reserved
3:0	RW	0x0	ext_vertical_total_part2 External vertical total part2. It is not need to be set at normal resolution. Just for special resolution or test purpose

HDMI_reg13

Address: Operational Base + offset (0x004c)

Register13

Bit	Attr	Reset Value	Description
7	RW	0x0	reserved
6:0	RW	0x00	ext_vertical_blank External vertical blank. It is not need to be set at normal resolution. Just for special resolution or test purpose.

HDMI_reg14

Address: Operational Base + offset (0x0050)

Register14

Bit	Attr	Reset Value	Description
7	RW	0x0	reserved
6:0	RW	0x00	ext_vertical_vsync_vback External vertical Vsync plus Vback. It is not need to be set at normal resolution. Just for special resolution or test purpose.

HDMI_reg15

Address: Operational Base + offset (0x0054)

Register15

Bit	Attr	Reset Value	Description
7:6	RW	0x0	reserved
5:0	RW	0x00	ext_vertical_duration External vertical duration. It is not need to be set at normal resolution. Just for special resolution or test purpose.

HDMI_reg35

Address: Operational Base + offset (0x00d4)

Register35

Bit	Attr	Reset Value	Description
7	RW	0x0	cts_sel CTS source select 1'b0: internal CTS 1'b1: external CTS
6:5	RW	0x0	reserved
4:3	RW	0x0	audio_type_sel Audio type select 2'b00: I2S 2'b01: S/PDIF
2	RW	0x0	mclk_sel MCLK select

Bit	Attr	Reset Value	Description
1:0	RW	0x1	mclk_ratio MCLK ratio 2'b00: 128 fs 2'b01: 256 fs 2'b10: 384 fs 2'b11: 512 fs

HDMI_reg37

Address: Operational Base + offset (0x00dc)

Register37

Bit	Attr	Reset Value	Description
7:4	RW	0x0	reserved
3:0	RW	0x0	sample_freq_sel Sampling frequency for I2S audio 4'b0011: 32 kHz 4'b0000: 44.1 kHz 4'b0010: 48 kHz 4'b1000: 88.2 kHz 4'b1010: 96 kHz 4'b1100: 176.4 kHz 4'b1110: 192 kHz

HDMI_reg38

Address: Operational Base + offset (0x00e0)

Register38

Bit	Attr	Reset Value	Description
7:6	RW	0x0	reserved
5:2	RW	0xf	i2s_sel I2S enable for the four I2S pins 4'b0001: I2S0 4'b0010: I2S1 4'b0100: I2S2 4'b1000: I2S3
1:0	RW	0x0	i2s_fmt I2S format 2'b00: standard I2S mode 2'b01: right-justified I2S mode 2'b10: left-justified I2S mode

HDMI_reg39

Address: Operational Base + offset (0x00e4)

Register39

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
7:6	RW	0x0	audio_channel3_input_sel Audio channel3 input 2'b00: I2S3 2'b01: I2S2 2'b10: I2S1 2'b11: I2S0
5:4	RW	0x0	audio_channel2_input_sel Audio channel2 input 2'b00: I2S2 2'b01: I2S1 2'b10: I2S0 2'b11: I2S3
3:2	RW	0x0	audio_channel1_input_sel Audio channel1 input 2'b00: I2S1 2'b01: I2S0 2'b10: I2S3 2'b11: I2S2
1:0	RW	0x0	audio_channel0_input_sel Audio channel0 input 2'b00: I2S0 2'b01: I2S3 2'b10: I2S2 2'b11: I2S1

HDMI_reg3a

Address: Operational Base + offset (0x00e8)

Register3a

Bit	Attr	Reset Value	Description
7	RW	0x0	lr_swap_ch7_8 Left/Right data swap for ch7/8
6	RW	0x0	lr_swap_ch5_6 Left/Right data swap for ch5/6
5	RW	0x0	lr_swap_ch3_4 Left/Right data swap for ch3/4
4	RW	0x0	lr_swap_ch1_2 Left/Right data swap for ch1/2
3:0	RW	0x0	spdif_sample_freq S/PDIF sampling frequency 4'b0011: 32 kHz. 4'b0000: 44.1 kHz. 4'b0010: 48 kHz. 4'b1000: 88.2 kHz. 4'b1010: 96 kHz. 4'b1100: 176.4 kHz. 4'b1110: 192 kHz.

HDMI_reg3f

Address: Operational Base + offset (0x00fc)

Register3f

Bit	Attr	Reset Value	Description
7:4	RW	0x0	reserved
3:0	RW	0x0	ncts_part1 20-bit N-CTS Value for audio clock generation at Sink end. N-CTS=reg3f[3:0],reg40[7:0],reg41[7:0] Use this function please refer to the HDMI specification at "Recommended N and Expected CTS Values"

HDMI_reg40

Address: Operational Base + offset (0x0100)

Register40

Bit	Attr	Reset Value	Description
7:0	RW	0x18	ncts_part2 20-bit N-CTS Value for audio clock generation at Sink end. N-CTS=reg3f[3:0],reg40[7:0],reg41[7:0] Use this function please refer to the HDMI specification at "Recommended N and Expected CTS Values"

HDMI_reg41

Address: Operational Base + offset (0x0104)

Register41

Bit	Attr	Reset Value	Description
7:0	RW	0x00	ncts_part3 20-bit N-CTS Value for audio clock generation at Sink end. N-CTS=reg3f[3:0],reg40[7:0],reg41[7:0] Use this function please refer to the HDMI specification at "Recommended N and Expected CTS Values"

HDMI_reg45

Address: Operational Base + offset (0x0114)

Register45

Bit	Attr	Reset Value	Description
7:4	RW	0x00	reserved
3:0	RW	0x00	When use external CTS (reg0x35[7]=1)then set these three regs. CTS={reg45[3:0],reg46[7:0], reg47[7:0]}

HDMI_reg46

Address: Operational Base + offset (0x0118)

Register46

Bit	Attr	Reset Value	Description
7:0	RW	0x00	When use external CTS (reg0x35[7]=1)then set these three regs. CTS={reg45[3:0],reg46[7:0], reg47[7:0]}

HDMI_reg47

Address: Operational Base + offset (0x011c)

Register47

Bit	Attr	Reset Value	Description
7:0	RW	0x00	When use external CTS (reg0x35[7]=1)then set these three regs. CTS={reg45[3:0],reg46[7:0], reg47[7:0]}

HDMI_reg4a

Address: Operational Base + offset (0x0128)

Register4a

Bit	Attr	Reset Value	Description
7:2	RW	0x0	reserved
1	RO	0x0	ddc_sda_bus_statue DDC SDA bus status
0	RO	0x0	ddc_scl_bus_statue DDC SCL bus status

HDMI_reg4b

Address: Operational Base + offset (0x012c)

Register4b

Bit	Attr	Reset Value	Description
7:0	RW	0x40	ddc_bus_access_freq_lsb DDC bus access frequency (LSB)

HDMI_reg4c

Address: Operational Base + offset (0x0130)

Register4c

Bit	Attr	Reset Value	Description
7:0	RW	0x00	ddc_bus_access_freq_msb DDC bus access frequency (MSB)

HDMI_reg4d

Address: Operational Base + offset (0x0134)

Register4d

Bit	Attr	Reset Value	Description
7:0	RW	0x00	edid_segment_pointer EDID segment pointer

HDMI_reg4e

Address: Operational Base + offset (0x0138)

Register4e

Bit	Attr	Reset Value	Description
7:0	RW	0x00	edid_word_addr EDID word address

HDMI_reg4f

Address: Operational Base + offset (0x013c)

Register4f

Bit	Attr	Reset Value	Description
7:0	RW	0x00	edid_fifo_rd_addr EDID FIFO read address

HDMI_reg50

Address: Operational Base + offset (0x0140)

Register50

Bit	Attr	Reset Value	Description
7:0	RO	0x00	edid_rd_access_window EDID reading access window

HDMI_reg52

Address: Operational Base + offset (0x0148)

Register52

Bit	Attr	Reset Value	Description
7	RW	0x0	authentication_start Authentication start
6	RW	0x0	bksv_not_in_blacklist BKSV is not in the blacklist
5	RW	0x0	bksv_in_blacklist BKSV is in the blacklist
4	RW	0x1	encrypte_en current frame encrypted en 1'b0: current frame should not be encrypted 1'b1: current frame should be encrypted
3	RW	0x0	authentication_stop Authentication Stop
2	RW	0x0	advanced_mode_sel 1'b0: do not use advanced cipher mode 1'b1: use advanced cipher mode
1	RW	0x1	mode_sel mode selection 1'b0: DVI mode 1'b1: HDMI mode
0	RW	0x0	hdcp_reset HDCP reset

HDMI_reg53

Address: Operational Base + offset (0x014c)

Register53

Bit	Attr	Reset Value	Description
7	RW	0x0	disable_127_chk Disable 127 check
6	RW	0x0	skip_bksv_blacklist_chk Skip BKSV blacklist check
5	RW	0x0	pj_chk_en Enable Pj check
4	RW	0x0	disable_dev_num_check Disable device number check
3	RW	0x1	dly_ri_chk Delay Ri check by one clock
2	RW	0x0	use_preset Use preset An value

Bit	Attr	Reset Value	Description
1:0	RW	0x0	key_comb Key combination

HDMI_reg57

Address: Operational Base + offset (0x015c)

Register57

Bit	Attr	Reset Value	Description
7	RW	0x0	Authenticated enable 1'b0: Authenticated 1'b1: Not authenticated
6	RW	0x0	1'b0: AV content is not encrypted 1'b1: AV content is encrypted
5	RW	0x0	reserved
4	RW	0x0	HDCP_work 1'b0: HDCP is working 1'b1: HDCP is not working
3	RW	0x0	advanced_cipher_en advanced cipher enable 1'b0: Advanced cipher is not enabled 1'b1: Advanced cipher is enabled
2:0	RW	0x0	reserved

HDMI_reg58

Address: Operational Base + offset (0x0160)

Register58

Bit	Attr	Reset Value	Description
7:0	RW	0x00	rx_bcaps_value RX Bcaps value

HDMI_reg63

Address: Operational Base + offset (0x018c)

Register63

Bit	Attr	Reset Value	Description
7:5	RW	0x00	reserved
4:0	RW	0x06	timer_100ms Registers for timer 100ms

HDMI_reg65

Address: Operational Base + offset (0x0194)

Register65

Bit	Attr	Reset Value	Description
7:4	RW	0x0	reserved
3	RW	0x0	ddc_channel_no_ack DDC channels no acknowledge
2	RW	0x0	pj_mismatch Pj mismatch

Bit	Attr	Reset Value	Description
1	RW	0x0	ri_mismatch Ri mismatch
0	RW	0x0	bksv_wrong Bksv is wrong

HDMI_reg66

Address: Operational Base + offset (0x0198)

Register66

Bit	Attr	Reset Value	Description
7:0	RW	0x00	Bksv value with least significant byte first

HDMI_reg67

Address: Operational Base + offset (0x019c)

Register67

Bit	Attr	Reset Value	Description
7:0	RW	0x00	Bksv value with least significant byte first

HDMI_reg68

Address: Operational Base + offset (0x01a0)

Register68

Bit	Attr	Reset Value	Description
7:0	RW	0x00	Bksv value with least significant byte first

HDMI_reg69

Address: Operational Base + offset (0x01a4)

Register69

Bit	Attr	Reset Value	Description
7:0	RW	0x00	Bksv value with least significant byte first

HDMI_reg6a

Address: Operational Base + offset (0x01a8)

Register6a

Bit	Attr	Reset Value	Description
7:0	RW	0x00	Bksv value with least significant byte first

HDMI_reg6c

Address: Operational Base + offset (0x01b0)

Register6c

Bit	Attr	Reset Value	Description
7:0	RW	0x00	seed_an An seed for generate An

HDMI_reg95

Address: Operational Base + offset (0x0254)

Register95

Bit	Attr	Reset Value	Description
7:0	RW	0x00	hdcp_fifo_first_wr_byte_addr HDCP KEY FIFO first write byte num address

HDMI_reg96

Address: Operational Base + offset (0x0258)

Register96

Bit	Attr	Reset Value	Description
7:0	RW	0x00	hdcp_fifo_first_rd_byte_addr HDCP KEY FIFO first read byte num address

HDMI_reg97

Address: Operational Base + offset (0x025c)

Register97

Bit	Attr	Reset Value	Description
7:2	RO	0x0	reserved
1	RW	0x0	hdcp_fifo_rd_addr8 HDCP KEY FIFO first read byte num address[8]
0	RW	0x0	hdcp_fifo_wr_addr8 HDCP KEY FIFO first write byte num address[8]

HDMI_reg98

Address: Operational Base + offset (0x0260)

Register98

Bit	Attr	Reset Value	Description
7:0	RW	0x03	hdcp_fifo_wr_rd_addr HDCP KEY FIFO write or read point address

HDMI_reg9c

Address: Operational Base + offset (0x0270)

Register9c

Bit	Attr	Reset Value	Description
7	RW	0x0	general_control_packet General control packet 1'b1: enable 1'b0: disable
6	RW	0x0	mpeg_source_frame_packet MPEG source InfoFrame packet 1'b1: enable 1'b0: disable
5	RW	0x0	source_descriptor_frame_packet Source product descriptor InfoFrame packet 1'b1: enable 1'b0: disable
4	RW	0x0	vendor_specific_frame_packet Vendor specific InfoFrame packet 1'b1: enable 1'b0: disable
3	RW	0x0	gamut_metadata_packet Gamut metadata packet 1'b1: enable 1'b0: disable

Bit	Attr	Reset Value	Description
2	RW	0x0	isrc1_packet ISRC1 packet 1'b1: enable 1'b0: disable
1	RW	0x0	acp_packet ACP packet 1'b1: enable 1'b0: disable
0	RW	0x0	generic_packet Generic packet 1'b1: enable 1'b0: disable

HDMI_reg9e

Address: Operational Base + offset (0x0278)

Register9e

Bit	Attr	Reset Value	Description
7:1	RW	0x0	reserved
0	RW	0x1	auto_checksum_frame Auto checksum for InfoFrame 1'b1: enable 1'b0: disable It enables checksum calculation for any InfoFrame packets by hardware.

HDMI_reg9f

Address: Operational Base + offset (0x027c)

Register9f

Bit	Attr	Reset Value	Description
7:0	RW	0x00	packet_offset packet offset register for 0xa0~0xbe 8'h0: Generic packet 8'h1: ACP packet 8'h2: ISRC1 packet 8'h3: ISRC2 packet 8'h4: Gamut metadata packet 8'h5: Vendor specific InfoFrame 8'h6: AVI InfoFrame 8'h7: Source product descriptor InfoFrame packet 8'h8: Audio InfoFrame packet 8'h9: MPEG source InfoFrame

HDMI_Rega0

Address: Operational Base + offset (0x0280)

Registera0

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_rega1

Address: Operational Base + offset (0x0284)

Registera1

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_rega2

Address: Operational Base + offset (0x0288)

Registera2

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_rega3

Address: Operational Base + offset (0x028c)

Registera3

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_rega4

Address: Operational Base + offset (0x0290)

Registera4

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_rega5

Address: Operational Base + offset (0x0294)

Registera5

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_rega6

Address: Operational Base + offset (0x0298)

Registera6

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_rega7

Address: Operational Base + offset (0x029c)

Registera7

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_rega8

Address: Operational Base + offset (0x02a0)

Registera8

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_rega9

Address: Operational Base + offset (0x02a4)

Registera9

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regaa

Address: Operational Base + offset (0x02a8)

Registersaa

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regab

Address: Operational Base + offset (0x02ac)

Registerab

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regac

Address: Operational Base + offset (0x02b0)

Registerac

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regad

Address: Operational Base + offset (0x02b4)

Registerad

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regae

Address: Operational Base + offset (0x02b8)

Registerae

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regaf

Address: Operational Base + offset (0x02bc)

Registeraf

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regb0

Address: Operational Base + offset (0x02c0)

Registerb0

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regb1

Address: Operational Base + offset (0x02c4)

Registerb1

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regb2

Address: Operational Base + offset (0x02c8)

Registerb2

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regb3

Address: Operational Base + offset (0x02cc)

Registerb3

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regb4

Address: Operational Base + offset (0x02d0)

Registerb4

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regb5

Address: Operational Base + offset (0x02d4)

Registerb5

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regb6

Address: Operational Base + offset (0x02d8)

Registerb6

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regb7

Address: Operational Base + offset (0x02dc)

Registerb7

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regb8

Address: Operational Base + offset (0x02e0)

Registerb8

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regb9

Address: Operational Base + offset (0x02e4)

Registerb9

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regba

Address: Operational Base + offset (0x02e8)

Registerba

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regbb

Address: Operational Base + offset (0x02ec)

Registerbb

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regbc

Address: Operational Base + offset (0x02f0)

Registerbc

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_regbd

Address: Operational Base + offset (0x02f4)

Registerbd

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet</p> <p>ACP packet</p> <p>ISRC1 packet</p> <p>ISRC2 packet</p> <p>Gamut metadata packet</p> <p>Vendor specific InfoFrame packet</p> <p>AVI InfoFrame packet</p> <p>Source product descriptor InfoFrame packet</p> <p>Audio InfoFrame packet</p> <p>MPEG source InfoFrame packet</p>

HDMI_Regbe

Address: Operational Base + offset (0x02f8)

Registerbe

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

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>After configure the packet offset register, these registers can configure any packet shown following.</p> <p>Use this function please refer to the HDMI specification at "Data Island Packet Definitions"</p> <p>Generic packet ACP packet ISRC1 packet ISRC2 packet Gamut metadata packet Vendor specific InfoFrame packet AVI InfoFrame packet Source product descriptor InfoFrame packet Audio InfoFrame packet MPEG source InfoFrame packet</p>

HDMI_regc0

Address: Operational Base + offset (0x0300)

Registerc0

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>mask_int Mask for Interrupt 1'b0: masked 1'b1: not mask [5] Interrupt for active Vsync edge [2] Interrupt for EDID Ready</p>

HDMI_regc1

Address: Operational Base + offset (0x0304)

Registerc1

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>int_statu Interrupt status [5] Interrupt for active Vsync edge This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit. [2] Interrupt for EDID Ready This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit.</p>

HDMI_regc2

Address: Operational Base + offset (0x0308)

Registerc2

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
7:3	RW	0x0f	int_mask Mask for interrupt: 1'b0: masked 1'b1: not masked [7] for HDCP error [6] for bksv fifo ready [5] for bksv update [4] for authentication success [3] for ready to start authentication
2:0	RW	0x0	reserved

HDMI_regc3

Address: Operational Base + offset (0x030c)

Registerc3

Bit	Attr	Reset Value	Description
7:3	RW	0x00	int_status Interrupt status [7] HDCP error Interrupt This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit. [6] bksv fifo ready This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit. [5] bksv update This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit. [4] authentication success This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit. [3] ready to start authentication This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit.
2:0	RW	0x0	reserved

HDMI_regc4

Address: Operational Base + offset (0x0310)

Registerc4

Bit	Attr	Reset Value	Description
7	RW	0x00	reserved

Bit	Attr	Reset Value	Description
6:0	RW	0x00	<p>int_mask Mask for interrupt: 1'b0: masked 1'b1: not masked</p> <p>[6] for HDCP authentication M0 ready [5] for first frame arrive [4] for HDCP An ready [3] reserved [2] for HDCP encrypted [1] for HDCP not encrypted (av mute) [0] for HDCP not encrypted (no av mute)</p>

HDMI_regc5

Address: Operational Base + offset (0x0314)

Registerc5

Bit	Attr	Reset Value	Description
7	RW	0x00	reserved
6:0	RW	0x00	<p>int_status_group3 Interrupt status Group3 This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit.</p> <p>[6] HDCP authentication M0 ready This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit.</p> <p>[5] irst frame arrive This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit.</p> <p>[4] HDCP An ready This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit.</p> <p>[2] HDCP encrypted This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit.</p> <p>[1] HDCP not encrypted (av mute) This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit.</p> <p>[0] HDCP not encrypted (no av mute) This bit will active when the mask for this bit to be set 1. When Interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit.</p>

HDMI_regc8

Address: Operational Base + offset (0x0320)

Registerc8

Bit	Attr	Reset Value	Description
7	RO	0x0	hot_plug_pin_status Hot plug pin status 1'b1: Plug in 1'b0: Plug out
6	RO	0x0	reserved
5	RW	0x1	Mask_hpd_int Mask for hot plug detect(HPG) interrupt 1'b0: masked 1'b1: not mask
4:2	RW	0x4	reserved
1	RW	0x0	Int_hpd Interrupt for hot plug detect(HPD) This bit will active when the mask for this bit to be set 1. When interrupt occurs, this bit will be equal to 1. Clear interrupt: write 1 to this bit.
0	RW	0x0	reserved

HDMI_regc9

Address: Operational Base + offset (0x0324)

Registerc9

Bit	Attr	Reset Value	Description
7:6	RW	0x1	video_bist_mode Video BIST mode 2'b00: normal color bar. 2'b01: special color bar. 2'b10: black
5	RW	0x0	flush_en Flush enable. After enable it the screen will flush from color bar to black again and again. 1'b1: enable 1'b0: disable
4	RW	0x1	disable_colorbar_bist_test Disable color bar BIST test 1'b1: disable 1'b0: enable
3:0	RW	0x0	reserved

HDMI_regce

Address: Operational Base + offset (0x0338)

Registerce

Bit	Attr	Reset Value	Description
7:1	RW	0x0	reserved

Bit	Attr	Reset Value	Description
0	RW	0x1	tmds_channel_sync The synchronization for TMDS channels The synchronization for TMDS channels is automatic after Reset the HDMI. But you can synchronization for TMDS channels manually. The operation for this function is: first send 0 to this bit, then send 1, and keep 1 into this bit.

HDMI_regd0

Address: Operational Base + offset (0x0340)

Registerd0

Bit	Attr	Reset Value	Description
7	RW	0x0	reserved
6	RW	0x0	modify_start_bit_timing_hisense_tv Modify the start bit timing for Hisense TV 1'b1: Modify the start bit timing for Hisense TV 1'b0: Not modify
5	RW	0x0	reject_rec_cec_broadcast_message Reject receive CEC broadcast message 1'b1: Reject receive CEC broadcast message 1'b0: Not reject
4:3	RW	0x0	reserved
2	RW	0x0	CEC_bus_free_time_cnt_en Enable count CEC bus free time 1'b1: Enable count CEC bus free time 1'b0: Not enable
1	RW	0x0	forbid_receive_cec_message Forbid receive CEC message 1'b1: Forbid receive CEC message 1'b0: Not forbid
0	RW	0x0	start_transmit_cec_message Start transmit CEC message 1'b1: Start transmit CEC message 1'b0: Not start

HDMI_regd1

Address: Operational Base + offset (0x0344)

Registerd1

Bit	Attr	Reset Value	Description
7:0	RW	0x00	cec_rx_tx_data_in_fifo CEC RX/TX data in FIFO

HDMI_regd2

Address: Operational Base + offset (0x0348)

Registerd2

Bit	Attr	Reset Value	Description
7:5	RW	0x0	reserved

Bit	Attr	Reset Value	Description
4:0	RW	0x00	wr_cec_tx_data_addr Point address for write CEC TX data

HDMI_regd3

Address: Operational Base + offset (0x034c)

Registerd3

Bit	Attr	Reset Value	Description
7:5	RW	0x0	reserved
4:0	RW	0x00	cec_rx_data_addr Point address for CEC RX data

HDMI_regd4

Address: Operational Base + offset (0x0350)

Registerd4

Bit	Attr	Reset Value	Description
7:0	RW	0x03	cec_clk_freq CEC working clock frequency register1

HDMI_regd5

Address: Operational Base + offset (0x0354)

Registerd5

Bit	Attr	Reset Value	Description
7:0	RW	0x09	cec_work_clk_freq CEC working clock frequency register1

HDMI_regd6

Address: Operational Base + offset (0x0358)

Registerd6

Bit	Attr	Reset Value	Description
7:5	RW	0x0	reserved
4:0	RW	0x03	tx_block_length Length of TX blocks

HDMI_regd7

Address: Operational Base + offset (0x035c)

Registed7

Bit	Attr	Reset Value	Description
7:5	RW	0x0	reserved
4:0	RW	0x00	rx_block_length Length of RX blocks

HDMI_regd8

Address: Operational Base + offset (0x0360)

Registerd8

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

Bit	Attr	Reset Value	Description
7:0	RW	0xff	<p>cec_tx_int_mask Mask for CEC TX Interrupts 1'b0: masked 1'b1: not mask</p> <p>[3] Interrupt for TX done [2] Interrupt for TX no ACK from follower [1] Interrupt for TX broadcast rejected [0] Interrupt for CEC line free time not satisfied</p>

HDMI_regd9

Address: Operational Base + offset (0x0364)

Registerd9

Bit	Attr	Reset Value	Description
7:0	RW	0xff	<p>cec_rx_int_mask Mask for CEC RX Interrupts 1'b0: masked 1'b1: not mask</p> <p>[4] Interrupt for RX receive logic address error [3] Interrupt for RX receive glitch error [2] Interrupt for RX ACK detect overtime [1] Interrupt for RX sending broadcast reject [0] Interrupt for RX done</p>

HDMI_regda

Address: Operational Base + offset (0x0368)

Registerda

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>cec_tx_int CEC TX Interrupts</p> <p>[3] Interrupt for TX done [2] Interrupt for TX no ACK from follower [1] Interrupt for TX broadcast rejected [0] Interrupt for CEC line free time not satisfied</p>

HDMI_regdb

Address: Operational Base + offset (0x036c)

Registerdb

Bit	Attr	Reset Value	Description
7:0	RW	0x00	<p>cec_tx_int CEC RX Interrupts</p> <p>[4] Interrupt for RX receive logic address error [3] Interrupt for RX receive glitch error [2] Interrupt for RX ACK detect overtime [1] Interrupt for RX sending broadcast reject [0] Interrupt for RX done</p>

HDMI_refdc

Address: Operational Base + offset (0x0370)

Registerdc

Bit	Attr	Reset Value	Description
7:0	RW	0xd0	signal_free_time_l Signal free time length_L

HDMI_regdd

Address: Operational Base + offset (0x0374)

Registerdd

Bit	Attr	Reset Value	Description
7:0	RW	0x20	signal_free_time_h Signal free time length_H

HDMI_regde

Address: Operational Base + offset (0x0378)

Registerde

Bit	Attr	Reset Value	Description
7:4	RW	0x0	reserved
3:0	RW	0x0	dev_addr Device logic address

HDMI_rege0

Address: Operational Base + offset (0x0380)

Registere0

Bit	Attr	Reset Value	Description
7:5	RW	0x0	reserved
4	RW	0x1	TMDS_clk_sel 1'b0: select TMDS clock from PLL; 1'b1: select TMDS clock generated by serializer
3	RW	0x0	TMDS_phase_sel 1'b0: select default phase for TMDS clock; 1'b1: select synchronized phase for TMDS clock with regard to TMDS data
2	RW	0x0	hdmi_band_gap_pd HDMI Band gap power down. 1'b1: Active
1	RW	0x1	hdmi_pll_pd HDMI PLL power down. 1'b1: Active
0	RW	0x1	tmds_channel_pd TMDS channel power down. 1'b1: Active

HDMI_rege1

Address: Operational Base + offset (0x0384)

Registere1

Bit	Attr	Reset Value	Description
7:4	RW	0x0	reserved

Bit	Attr	Reset Value	Description
3	RW	0x1	clk_channel_driver_en Clock channel driver enable 1'b0: disable 1'b1: enable
2:0	RW	0x7	data_channels_driver_en Three data channels driver enable [2] ch2 [1] ch1 [0] ch0 1'b0: disable 1'b1: enable

HDMI_rege2

Address: Operational Base + offset (0x0388)

Registere2

Bit	Attr	Reset Value	Description
7:4	RW	0xa	clk_driver_strength Clock channel main-driver strength 0000~1111: the strength from low to high
3:0	RW	0xa	data_driver_strength Three data channels main-driver strength 0000~1111: the strength from low to high

HDMI_rege3

Address: Operational Base + offset (0x038c)

Registere3

Bit	Attr	Reset Value	Description
7	RW	0x0	reserved
6:4	RW	0x0	pre_emphasis_strength Pre-emphasis strength 000~111: the strength from 0 to high
3:2	RW	0x3	clk_driver_strength Clock channel pre-driver strength Slew rate from low to high 00~11: the strength from low to high
1:0	RW	0x3	data_driver_strength Three data channels pre-driver strength Slew rate from low to high 00~11: the strength from low to high

HDMI_rege7

Address: Operational Base + offset (0x039c)

Registere7

Bit	Attr	Reset Value	Description
7:0	RW	0x78	feedback_dividing_ratio_part1 The value of feedback dividing ratio

HDMI_rege8

Address: Operational Base + offset (0x03a0)

Registers8

Bit	Attr	Reset Value	Description
7:1	RW	0x0	reserved
0	RW	0x0	feedback_dividing_ratio_part2 The value of feedback dividing ratio

HDMI_Reged

Address: Operational Base + offset (0x03b4)

Registers

Bit	Attr	Reset Value	Description
7:5	RW	0x0	reserved
4:0	RW	0x0c	pre_dividing_ratio The value of pre-dividing ratio

Note1: There are some bits of registers have not description in the registers table above, are reserved for other using, please keep it into default value.

4.5 Interface Description

4.5.1 Video Input Source

The HDMI TX video source comes from VOP.

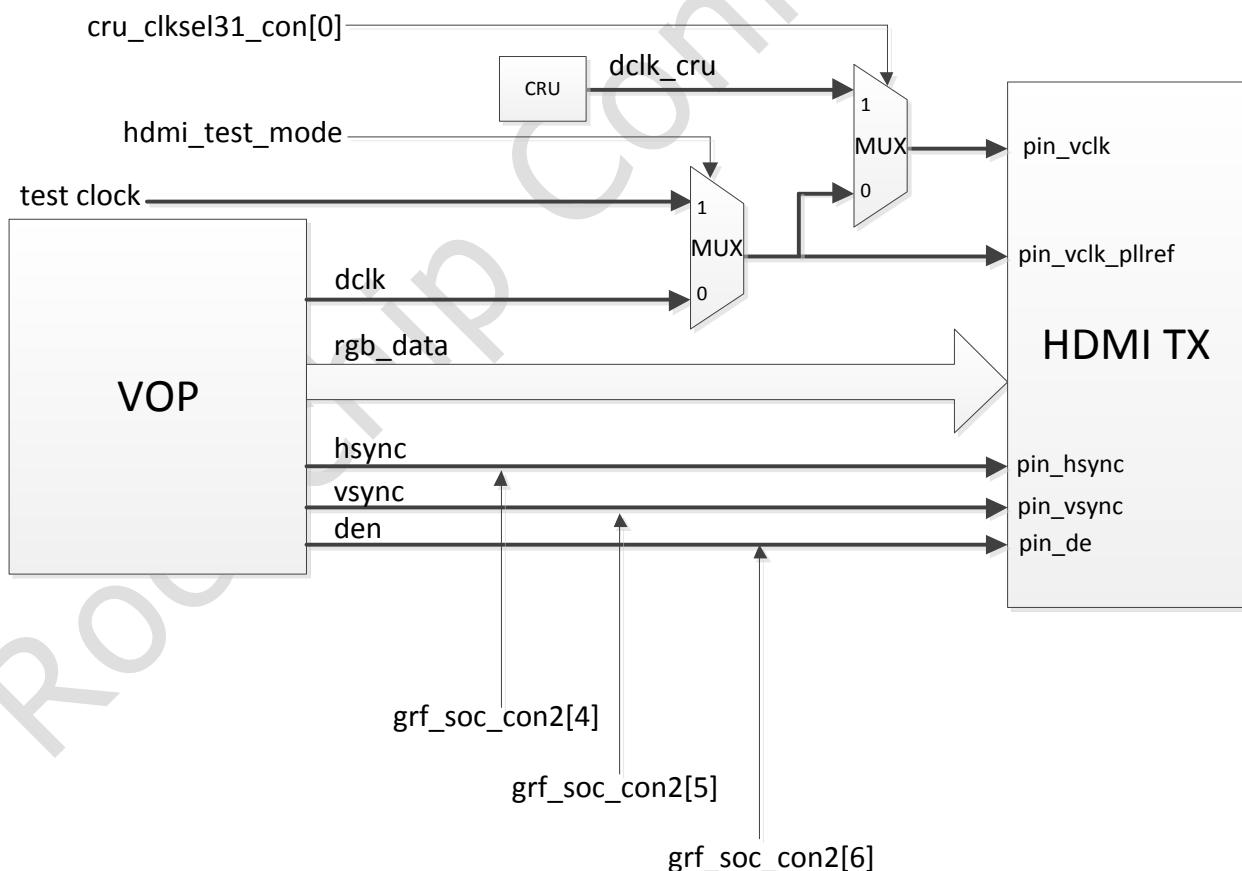


Fig. 4-6 VOP Connect to HDMI TX Diagram

grf_soc_con2[4] is used to configure the polarity of hsync input to HDMI TX. Write 1 to **grf_soc_con2[4]** will invert hsync.

grf_soc_con2[5] is used to configure the polarity of vsync input to HDMI TX. Write 1 to grf_soc_con2[5] will invert vsync.

grf_soc_con2[6] is used to configure the polarity of den input to HDMI TX. Write 1 to grf_soc_con2[6] will invert den.

4.5.2 Audio Input Source

The HDMI TX I2S source comes from I2S_2CH.

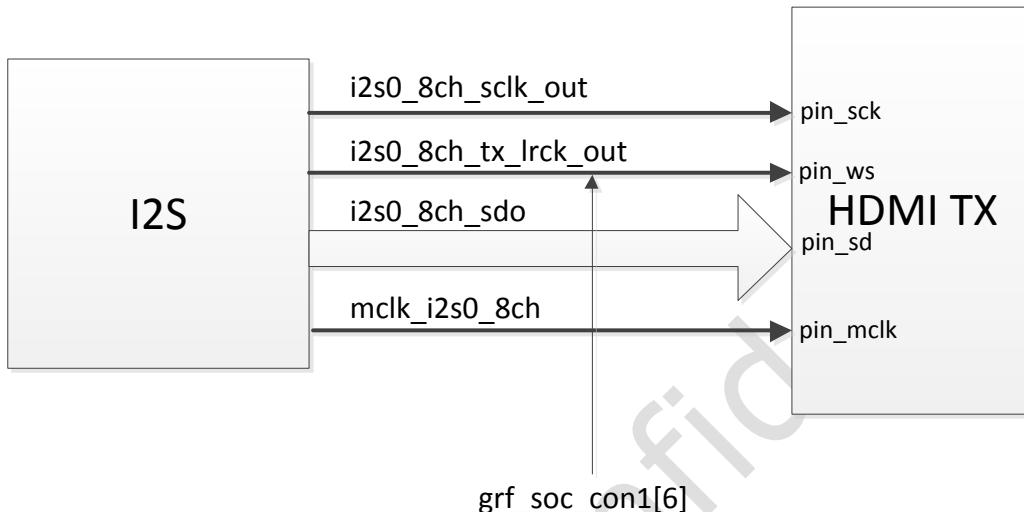


Fig. 4-7 I2S Connect to HDMI TX Diagram

When $\text{grf_soc_con1[6]} = 1$: $\text{pin_ws} = \text{i2s0_8ch_rx_lrck_out_o} | \text{i2s0_8ch_tx_lrck_out_o}$
When $\text{grf_soc_con1[6]} = 0$: $\text{pin_ws} = \text{i2s0_8ch_tx_lrck_out_o}$

The HDMI TX SPDIF source comes from SPDIF.

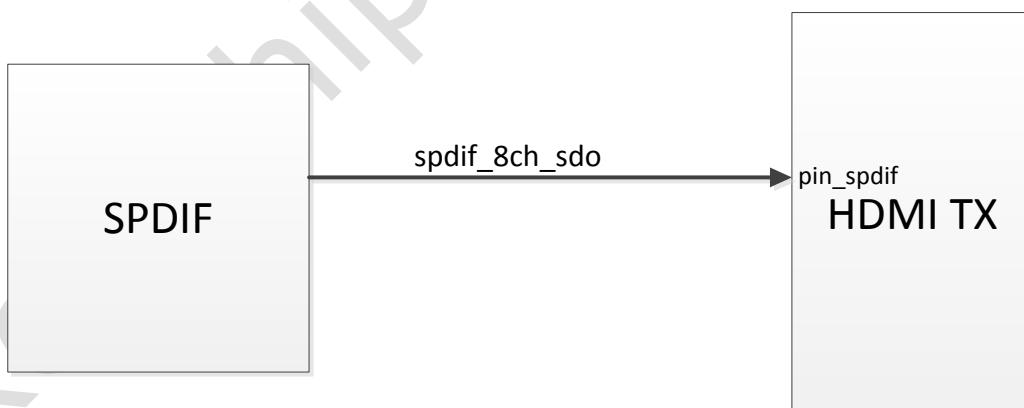


Fig. 4-8 SPDIF Connect to HDMI TX Diagram

4.6 Application Notes

This chapter describes how to bring up HDMI transmitter in your system. As shown few examples below, these introduce the basically HDMI transmitter application, likes, the Hot Plug Detect, EDID read back, multiple audio format input and different video resolution displaying.

You can easily configure these functions with proper registers value setting by HDMI TX APB BUS.

4.6.1 System Configure

Need configure IOMUX before us HDMI.

```
//HDMI hpg in
GRF_BASE + GRF_GPIO0B_IOMUX = (0x1<<30) | (0x4000);

//HDMI ddc in/out
GRF_BASE + GRF_GPIO0A_IOMUX = (0xf<<28) | (0xa000);

//HDMI cec/sda in/out
GRF_BASE + GRF_GPIO0C_IOMUX = (0x1<<24) | (0x0100);
```

4.6.2 Initial Operation

The default HDMI transmitter is configured to 24bit RGB 1080P resolution video with 8 channel 48K sample I2S format audio input. It is easily for customer to turn on HDMI transmitter without doing more complex operation. Just do the step, reset the HDMI TX.

4.6.3 Hot Plug Detection

Hot Plug Detect is a special feature for HDMI transmitter spying the state on the HDMI port. You can control this function by using the interrupt signal and proper registers from the HDMI transmitter with few operations. The following is a step by step instruction for detecting the hot plug in and out.

Hot Plug in Steps:

Step1: Send 0 x63 to register 0x00.

Step2: Plug HDMI receiver in.

Step3: Check the interrupt from signal pin_int.

If the pin_int shows high, that means the HDMI transmitter interrupt have generated.

Step4: Check the interrupt.

Read the bit1 of the register 0 xc8. If bit1=1'b1 that means the hot plug interrupt valid, otherwise there is other interrupt. Then send the 1'b1 to bit1 of register 0xc8 to clean up the hot plug interrupt.

Step5: Check if really HDMI receiver plug in.

Read the bit7 of the register 0xc8. If bit7=1'b1 that means hot plug in was really happen, otherwise there is may be a glitch on the HPD port.

Hot Plug out Steps:

Step1: HDMI transmitter at working state.

Step2: Plug HDMI receiver out.

Step3: Check the interrupt from signal pin_int.

If the pin_int shows high, that means the HDMI transmitter interrupt have generated.

Step4: Check the interrupt.

Read the bit1 of the register 0 xc8. If bit1=1'b1 that means the hot plug interrupt valid, otherwise there is other interrupt. Then send the 1'b1 to bit1 of register 0xc8 to clean up the hot plug interrupt.

Step5: Check if really HDMI receiver plug out.

Read the bit7 of the register 0xc8. If bit7=1'b0 that means hot plug out was really happen, otherwise there is may be a glitch on the HPD port.

4.6.4 Reading EDID

Read EDID is a function that can make the HDMI transmitter to read the HDMI receiver's Extended Display Identification Data (EDID) in order to discover the HDMI receiver's configuration and capabilities. HDMI transmitter can choose the appropriate audio and video format for playing and displaying by the HDMI receiver through the use of the EDID. Besides, HDMI transmitter support the reading Enhanced Extended Display Identification Data (E-EDID) if HDMI receiver have this enhanced structure.

The following describes how to read E-EDID through HDMI transmitter. The total E-EDID is 512bytes data, which is divided into 2 segments. Each segment has 256bytes data. The Read E-EDID function is only read 128bytes data from HDMI receiver at each time. So, you must read 4 times that can read total 512bytes data back.

Prepare read E-EDID 512bytes Steps:

Step1: Send 0 x63 to register 0x00.

System power down mode.

Step2: HDMI transmitter detected Hot Plug in.

Step3: Define DDC clock (SCL) frequency $F_{SCL} = 100K$.

$F_{SCL} = F_{reg_clk}/(4*X)$. X={register 0x4c, register 0x4b}.

Note Freg_clk is the frequency o f the tmds clock.

Assume the tmds clock is 148.5MHz.

Send 0 x73 to register 0x4b, Send 0x01 to register 0x4c.

Step4: Enable EDID Ready interrupt.

Send 0 x04 to register 0xc0

Read E-EDID segment 0x00 256bytes Steps:

Step1: Set EDID FIFO initial address.

Send 0x00 to register 0x4f

Step2: Set EDID first word address, read first 128bytes.

Send 0x00 to register 0x4e

Step3: Set EDID segment address.

Send 0x00 to register 0x4d

Step4: Waiting and check EDID interrupt from pin_int.

Check the EDID Ready state from bit2 o f register 0xc1. If bit2 = 1'b1 that means EDID data ready. Then send the 1'b1 to bit2 of register 0xc1 to clean up the EDID Ready interrupt.

Step5: Read first 128bytes EDID data from EDID FIFO.

for (i = 0, i < 128, i = i + 1)

 Read data from register 0x50.

Step6: Set EDID FIFO initial address again.

Send 0x00 to register 0x4f

Step7: Set EDID first word address, read last 128bytes.

Send 0x80 to register 0x4e

Step8: Set EDID segment address.

Send 0x00 to register 0x4d

Step9: Waiting and check EDID interrupt from pin_int.

Check the EDID Ready state from bit2 of register 0xc1. If bit2 = 1'b1 that means EDID data ready. Then send the 1'b1 to bit2 of register 0xc1 to clean up the EDID Ready interrupt.

Step10: Read last 128bytes EDID data from EDID FIFO.

 for (i = 0, i < 128, i = i + 1)

 Read data from register 0x50.

Read E-EDID segment 0x01 256bytes Steps:

Step1: Set EDID FIFO initial address.

Send 0x00 to register 0x4f

Step2: Set EDID first word address, read first 128bytes.

Send 0x00 to register 0x4e

Step3: Set EDID segment address.

Send 0x01 to register 0x4d

Step4: Waiting and check EDID interrupt from pin_int.

Check the EDID Ready state from bit2 of register 0xc1. If bit2 =1'b1 that means EDID data ready. Then send the 1'b1 to bit2 of register 0xc1 to clean up the EDID Ready interrupt.

Step5: Read first 128bytes EDID data from EDID FIFO.

for (i = 0, i < 128, i = i + 1)

Read data from register 0x50.

Step6: Set EDID FIFO initial address again.

Send 0x00 to register 0x4f

Step7: Set EDID first word address, read last 128bytes.

Send 0x80 to register 0x4e

Step8: Set EDID segment address.

Send 0x01 to register 0x4d

Step9: Waiting and check EDID interrupt from pin_int.

Check the EDID Ready state from bit2 of register 0xc1. If bit2 =1'b1 that means EDID data ready. Then send the 1'b1 to bit2 of register 0xc1 to clean up the EDID Ready interrupt.

Step10: Read last 128bytes EDID data from EDID FIFO.

for (i = 0, i < 128, i = i + 1)

Read data from register 0x50.

Note: The segment address must be sent at each time when read 128bytes E-EDID data.

4.6.5 Audio input configuration

HDMI transmitter audio support either SPDIF or four channel I2S input. SPDIF input supports audio sampling rates from 32 to 192 KHz. The I2S input supports from 2-channel to 8-channel audio up to 192 KHz. The default audio format is I2S input with 8 channels. The audio sample rate is 48K.

The following describes how to configure audio input format into I2S input with 2 channels and the sample rate is 44.1K.

Audio input requirement:

SD0 input (2 Channels I2S input)

WS (fs) = 44.1 KHz

SCK = 64fs

MCLK = 256fs

Configure Audio Input Format Steps:

Step1: Select I2S input.

Send 0x01 to register 0x35

Step2: Select SD0 input.

Send 0x04 to register 0x38

Step3: Select N/CTS for sample rate = 44.1 KHz.

Send 0x18 to register 0x40

Send 0x80 to register 0x41

The following describes how to configure audio input format into I2S input with 8 channels and the sample rate is 44.1K.

Audio input requirement:

SD[3 :0] input (8 Channels I2S input)

WS (fs) = 44.1 KHz

SCK = 64fs

MCLK = 256fs

Configure Audio Input Format Steps :

Step1: Select I2S input.

Send 0x01 to register 0x35

Step2: Select SD[3:0] input.

Send 0x3c to register 0x38

Send 0x00 to register 0x39

Step3: Select N/CTS for sample rate = 44.1 KHz.

Send 0x18 to register 0x40

Send 0x80 to register 0x41

4.6.6 Video input configuration

HDMI transmitter support RGB/YCbCr 24/30/36bit video input with different resolution. The default video format is RGB24bit input at resolution of 1080P@60. The following describes how to configure video input format into RGB24bit input at resolution of 480P@60, 720P@60 or 1080P@60.

VOP dclk cannot get invert.

Video input requirement:

24bit RGB 4:4:4 Source.

Resolution is 480P@60, 720P@60 or 1080P@60.

Configure Video Input Format Steps:

Step1: Select configure for video format.

Send 0x06 to register 0x9f

Step2: Configure the AVI info to HDMI RX.

Send 0x82 to register 0xa0 //HB0

Send 0x02 to register 0xa1 //HB1

Send 0xd to register 0xa2 //HB3

Send 0x00 to register 0xa3 //PB0: checksum is auto set, needn't set this register

Send 0x00 to register 0xa4 //PB1

Send 0x08 to register 0xa5 //PB2

Send 0x70 to register 0xa6 //PB3

Send 0x10 to register 0xa7 //PB4: VID 1920*1080P@60

Send 0x40 to register 0xa8 //PB5

Note: Select correct VID for displaying.

Send 0x02 to register 0xa7 for 720*480P@60.

Send 0x04 to register 0xa7 for 1280*720P@60.

Send 0x10 to register 0xa7 for 1920*1080P@60.

Send 0x11 to register 0xa7 for 720*576P@50.

Send 0x13 to register 0xa7 for 1280*720P@50.

Send 0 x1 f to register 0xa7 for 1920*1080P@50.
The detail configuration for AVI information, please refer to the HDMI specification (8.2.1) and CEA-861-D (6.3).

4.6.7 Low Power Mode

HDMI Transmitter can be configured into Low Power Mode at special customer works mode. The following is a step by step instruction to describe how to configure our HDMI Transmitter into this mode.

Low Power Enter Steps:

- Step1:** HDMI Transmitter working.
- Step2:** Low Power analog module.
Send 0x00 to register 0xe1.//Drive disable
Send 0x17 to register 0xe0.
- Step3:** Assign pin_vclk = 1'b0 or 1'b1.

Low Power Quit Steps:

- Step1: Reset system by pin_rst_n.
- Step2: Wait Hot Plug.
- Step3: Read EDID.
- Step4: Active vclk through p in_vclk
- Step5: Bring up analog module.
Send 0x15 to register 0xe0. //PLL power on
Send 0x14 to register 0xe0. //TX power on
Send 0x10 to register 0xe0. //BG power on
Send 0 x0f to register 0xe1. //driver enable
- Step6: Configuration mode reg at addr 0x00 if needed.
- Step7: Configuration Video format if needed.
- Step8: Configuration Audio format if needed.
- Step9: Configuration mode reg, power on digital part and select tmds_clk for configuration.
Send 0 x61 to register 0x00.
- Step10: Synchronize analog module.
Send 0x00 to register 0xce.
Send 0x01 to register 0xce.

4.6.8 CEC OPERATION

The CEC line is used for high-level user control of HDMI-connected devices.
You can control this function by using the interrupt signal and proper registers from the HDMI transmitter with few operations.
The following is a step by step instruction for CEC TX operations and CEC RX operations.

CEC TX steps

- (Send the Command : Standby (0x36) for example)
- Step1: Set logic address for CEC (logic address = 1).
Send 0x01 to register 0xde.
- Step2: Configure the divide numbers for internal reference clock.
Send 0 x03 to register 0xd4 and 0x09 to register 0 xd5. See Note1 for details.
- Step3: Configure the value for signal free time counter.

Send 0xd0 to register 0xdc and 0x20 to register 0xdd. See Note2 for details.

Step4: Configure point address for write CEC TX data.

Send 0x00 to register 0xd2.

Step5: Configure TX data FIFO (the Command o f Standby).

Send 0 x01 to register 0xd1, then, send 0x36 to register 0xd1.

The data 0x01 is for the Header Block, in which the high 4bits is for the initiator, the low 4bits is for the destination. The data 0x36 is for the Data Block, it indicate the command for standby.

Step6: Configure TX data length of this packet. The length is 2.

Send 0x02 to register 0xd6.

Step7: Enable count CEC b us free time.

Send 0x04 to register 0xd0.

Step8: Waiting 16.8ms+4 us for CEC bus free.

If Interrupt happened for CEC line free time not satisfied. You should change to receive coming data from opposite. After this you can do CEC TX steps again.

Note that the time for 16.8ms+4us should be compute by MCU itself

Step9: Begin TX.

Send 0x05 to register 0xd0.

Step10: Waiting and check interrupt o f C EC TX do ne.

Waiting the interrupt, then, read the value of register 0xda, if the value is 0x08 that means CEC command was transmitted successfully. At last, send 0 x08 to register 0xda to clear up this interrupt.

CEC RX steps:

Step1: Set logic address for CEC (logic address = 1).

Send 0x01 to register 0xde.

Step2: Configure the divide numbers for internal reference clock. See Note1 for details.

Send 0 x03 to register 0xd4 and 0x09 to register 0xd5.

Step3: Configure the point address fo r CEC RX data.

Send 0x00 to register 0xd3.

Step4: Waiting CEC TX send CEC packet to our HDMI Transmitter, then check interrupt of CEC. Read the value of register 0xdb, if the value is 0x80, that means CEC command was received successfully. Then send 0x80 to register 0xdb to clear up this interrupt.

Step5: Read the RX packet length.

Read the value of the register 0xd7.

Step6: Read the received CEC packet.

Read the value of the register 0 xd1 according to the RX packet length. If the length is 2, please read twice from the register 0xd1.

Note1: The system clock (Default value Fsys = 20MHz from pin_sys_clk), register 0xd4 and 0xd5 are used to configure the CEC logic to generate a reference clock (Fref=0.5 MHz, Tref = 2us), which is used to control the CEC signal's generation. The following is the formula for generating reference clock 0.5MHz.

Fref = Fsys / ((register 0xd4 + 1)(register 0xd5 + 1)). Under the default 20MHz system clock, the values of register 0xd4 and 0xd5 are 0x03 and 0x09.*

*Note2: Before attempting to transmit or re-transmit a frame, a device shall ensure that the CEC line has been inactive for a number of bit periods . For example , the signal free time is 7*2.4ms=16.8ms (the bit time*

is 2.4 ms). According to the $16.8ms/2us = \{register 0xdd, register 0xdc\}$, so the register $0xdd=0x20$ and register $0xdc= 0xd0$.

4.6.9 HDCP OPERATION

HDCP is designed to protect the transmission of Audiovisual Content between an HDCP Transmitter and an HDCP Receiver. You can control this function by using the interrupt signal and proper registers from the HDMI transmitter with few operations.

The following is a step by step instruction for HDCP operation.

HDCP operation steps (skip BKSV black list check) :

Example Condition: Video Format : $1920\times 1080p@59.94/60Hz$

Step1: Select the TMDS clock to configure registers.

Send 0x61 to register 0x00.

Step2: Write HDCP transmitter's Akey to the chip. See Note1 for details.

```
for(i =0; i <= 39 ; i = i+ 1 )
```

```
begin
```

```
    hdcpc_wdata_temp = akey[i];
```

```
    for ( j=0 ; j < 7 ; j = j + 1 )
```

```
        begin
```

```
            Send hdcpc_wdata_temp[7:0] to register 0x98;
```

```
            hdcpc_wdata_temp = hdcpc_wdata_temp >>8;
```

```
        end
```

```
    end
```

Step3: Write HDCP transmitter's AKSV1 to the chip. See Note1 for details.

```
hdcpc_wdata_temp = pre_ksv1
```

```
for (i = 0, i < 5, i = i + 1)
```

```
begin
```

```
    Send hdcpc_wdata_temp[7:0] to register 0x98;
```

```
    hdcpc_wdata_temp = hdcpc_wdata_temp >>8;
```

```
end
```

Step4: Write HDCP transmitter's AKSV2 to the chip. See Note1 for details.

```
hdcpc_wdata_temp = pre_ksv2
```

```
for (i = 0, i < 5, i = i + 1)
```

```
begin
```

```
    Send hdcpc_wdata_temp[7:0] to register 0x98;
```

```
    hdcpc_wdata_temp = hdcpc_wdata_temp >>8;
```

```
end
```

Step5: Check the key_ready signal.

Read the value of register 0x54, if the value is 0x01, that means the Akey and AKSV were load into the chip successfully.

Step6: Configure the DDC frequency. See Note2 for details.

Send 0x73 to register 0x4b.

Send 0x01 to register 0x4c.

Step7: Configure the HDCP function.

Send 0x77 to register 0x53.

Step8: Open the mask for HDCP interrupt.

Send 0xff to register 0xc2.

Send 0xff to register 0xc4.

Step9: Start the HDCP authentication.

Send 0x96 to register 0x52.

Step10: Checking HDCP interrupt.

Read the value of register 0xc3 and register 0xc5, if the value of register 0xc3 is 0x10 and the value of register 0xc5 is 0x65, that is means the HDCP authentication was successfully finished.

HDCP operation steps (not skip BKSV black list check) :

Example Condition: Video Format : 1920×1080p@59.94/60Hz

Step1: Select the TMDS clock to configure registers.

Send 0x01 to register 0x00.

Step2: Write HDCP transmitter's Akey to the chip. See Note1 for details.

```
for(i =0; i <= 39 ; i = i+ 1 )
```

```
begin
```

```
hdcp_wdata_temp = akey[i];
```

```
for ( j=0 ; j < 7 ; j = j + 1 )
```

```
begin
```

```
Send hdcp_wdata_temp[7:0] to register 0x98;
```

```
hdcp_wdata_temp = hdcp_wdata_temp >>8;
```

```
end
```

```
end
```

Step3: Write HDCP transmitter's AKSV1 to the chip. See Note1 for details.

```
hdcp_wdata_temp = pre_ksv1
```

```
for (i = 0, i < 5, i = i + 1)
```

```
begin
```

```
Send hdcp_wdata_temp[7:0] to register 0x98;
```

```
hdcp_wdata_temp = hdcp_wdata_temp >>8;
```

```
end
```

Step4: Write HDCP transmitter's AKSV2 to the chip. See Note1 for details.

```
hdcp_wdata_temp = pre_ksv2
```

```
for (i = 0, i < 5, i = i + 1)
```

```
begin
```

```
Send hdcp_wdata_temp[7:0] to register 0x98;
```

```
hdcp_wdata_temp = hdcp_wdata_temp >>8;
```

```
end
```

Step5: Check the key_ready signal.

Read the value of register 0x54, if the value is 0x01, that means the Akey and AKSV were load into the chip successfully.

Step6: Configure the DDC frequency. See Note2 for details.

Send 0x73 to register 0x4b.

Send 0x01 to register 0x4c.

Step7: Configure the HDCP function.

Send 0x37 to register 0x53(do not skip the BKSv blacklist check).

Step8: Open the mask for HDCP interrupt.

Send 0xff to register 0xc2.

Send 0xff to register 0xc4.

Step9: Start the HDCP authentication.

Send 0x96 to register 0x52.

Step10: Check the BKSv.

Wait for INT from registerc3 bit5(bksv_update), than read register66~register6a for BKSv values to check whether the BKSv is in the revocation list. If the BKSv is not in the revocation list, write 1 to register52 bit6(bksv_pass) , and authentication will continue. Else if the BKSv is in the revocation list, write 1 to register52 bit5(bksv_fail), and authentication will stop.

Step11: Checking HDCP interrupt.

Read the value of register 0xc3 and register 0xc5, if the value of register 0xc3 is 0x10 and the value of register 0xc5 is 0x65, that is means the HDCP authentication was successfully finished.

Note1: For HDCP's akey and KSV write, use the TEST KEYS in HDCP specification for example

akey[0] =56'h4da4588f131e69;

akey[1] =56'h1f823558e65009;

akey[2] =56'h8a6a47abb9980d;

akey[3] =56'hf3181b52cbc5ca;

akey[4] =56'hfb147f6896d8b4;

akey[5] =56'he08bc978488f81;

akey[6] =56'ha0d064c8112c41;

```
akey[7] =56'hb39d5a28242044;
akey[8] =56'hb928b2bdad566b;
akey[9] =56'h91a47b4a6ce4f6;
akey[10] =56'h5600f8205e9d58;
akey[11] =56'h8c7fb706ee3fa0;
akey[12] =56'hc02d8c9d7cbc28;
akey[13] =56'h561261e54b9f05;
akey[14] =56'h74f0de8ccac1cb;
akey[15] =56'h3bb8f60efcdb6a;
akey[16] =56'ha02bbb16b22fd7;
akey[17] =56'h482f8e46785498;
akey[18] =56'h66ae2562274738;
akey[19] =56'h3d4952a323ddf2;
akey[20] =56'he2d231767b3a54;
akey[21] =56'h4d581aed66125;
akey[22] =56'h326082bf7b22f7;
akey[23] =56'hf61b463530ce6b;
akey[24] =56'h360409f0d7976b;
akey[25] =56'ha1e105618d49f9;
akey[26] =56'hc98e9dd1053406;
akey[27] =56'h20c36794426190;
akey[28] =56'h964451ceac4fc3;
akey[29] =56'h3e904504e18c8a;
akey[30] =56'h290010579c2dfc;
akey[31] =56'hd7943b69e5b180;
akey[32] =56'h54c7ea5bdd7b43;
akey[33] =56'h74fb5887c790ba;
akey[34] =56'h935cfa364e1de0;
akey[35] =56'h03075e159a11ae;
akey[36] =56'h05d3408a78fb01;
akey[37] =56'h0059a5d7a04db3;
akey[38] =56'h373b634a2c9e40;
akey[39] =56'h2573bbb4562041;
```

```
pre_ksv1 = 40'hb70361f714;
pre_ksv2 = 40'hb70361f714;
```

Note2: For DDC frequency configuration

In the HDCP specification, the frequency value of DDC(F_{ddc}) support only up to 100Kbps, and now we select the TMDS clock as the reference clock of DDC , and under the condition of 1920×1080p@59.94/60Hz, the frequency of TMDS(F_{TMDS}) is 148.5MHz. Pay attention, internally we use the 4 times of DDC frequency to generate the SCL. So

$X = F_{TMDS}/F_{ddc}/4 = 'h173$, So the values of register 0x4b and 0x4c are 0x73 and 0x01.

4.6.10 NOMAL OPERATION EXAMPLE AT 1080P

After the description of HDMI Transmitter configuration above, the following example shows an entire configuration for HDMI Transmitter works on the 1080P mode.

Audio input requirement:

SD0 input (2 Channels I2S input)

WS (fs) = 48 KHz

SCK = 64fs

MCLK = 256fs

Video input requirement:

24bit RGB 4:4:4 Source.

Resolution is 1080P@60.

Setup Steps:

Step1: Power on HDMI Transmitter.

Step2: Configure the input signals.

Assign pin_test_en = 1'b0.

Step3: Reset HDMI Transmitter.

Assign 0 to the signal pin_rst_n and then assign 1.

Step4: System power down.
Send 0x63 to register 0x00.
Step5: Detect Hot Plug In.
Step6: Read EDID.
Step7: Configure video input format.
Step8: Configure audio input format.
Step9: Assign video and audio source to HDMI Transmitter.
Step10: System power on.
Send 0x61 to register 0x00.
Step11: Now, HDMI Transmitter is ready to go. Start your operation.

Rockchip Confidential

Chapter 5 Pulse Width Modulation (PWM)

5.1 Overview

The pulse-width modulator (PWM) feature is very common in embedded systems. It provides a way to generate a pulse periodic waveform for motor control or can act as a digital-to-analog converter with some external components.

The PWM Module supports the following features:

- 4-built-in PWM channels
- Configurable to operate in capture mode
 - Measures the high/low polarity effective cycles of this input waveform
 - Generates a single interrupt at the transition of input waveform polarity
 - 32-bit high polarity capture register
 - 32-bit low polarity capture register
 - 32-bit current value register
- Configurable to operate in continuous mode or one-shot mode
 - 32-bit period counter
 - 32-bit duty register
 - 32-bit current value register
 - Configurable PWM output polarity in inactive state and duty period pulse polarity
 - Period and duty cycle are shadow buffered. Change takes effect when the end of the effective period is reached or when the channel is disabled
 - Programmable center or left aligned outputs, and change takes effect when the end of the effective period is reached or when the channel is disabled
 - 8-bit repeat counter for one-shot operation. One-shot operation will produce $N + 1$ periods of the waveform, where N is the repeat counter value, and generates a single interrupt at the end of operation
 - Continuous mode generates the waveform continuously, and does not generate any interrupts
- pre-scaled operation to bus clock and then further scaled
- Available low-power mode to reduce power consumption when the channel is inactive.

5.2 Block Diagram

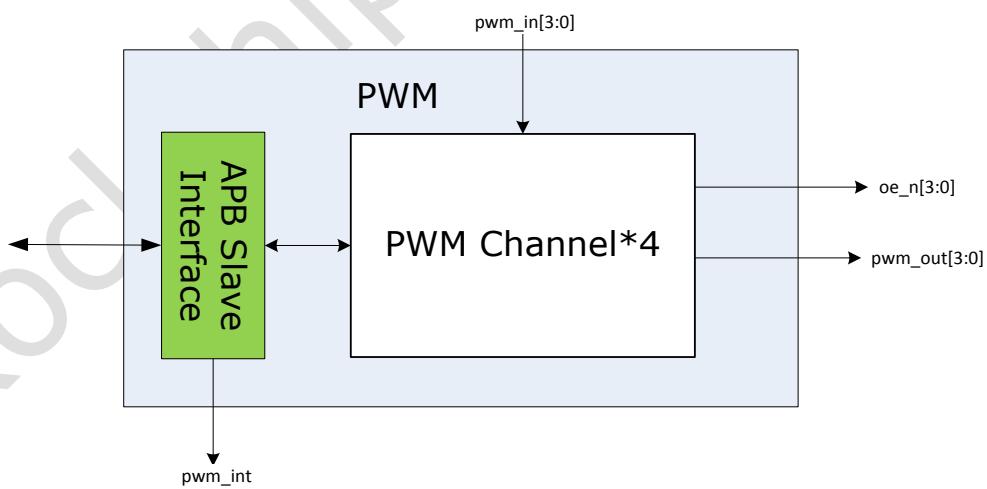


Fig. 5-1 PWM Block Diagram

The host processor gets access to PWM Register Block through the APB slave interface with 32-bit bus width, and asserts the active-high level interrupt.

PWM Channel is the control logic of PWM module, and controls the operation of PWM module according to the configured working mode.

5.3 Function Description

The PWM supports three operation modes: capture mode, one-shot mode and continuous

mode. For the one-shot mode and the continuous mode, the PWM output can be configured as the left-aligned mode or the center-aligned mode.

5.3.1 Capture mode

The capture mode is used to measure the PWM channel input waveform high/low effective cycles with the PWM channel clock, and asserts an interrupt when the polarity of the input waveform changes. The number of the high effective cycles is recorded in the PWMx_PERIOD_HPC register, while the number of the low effective cycles is recorded in the PWMx_DUTY_LPC register.

Notes: the PWM input waveform is doubled buffered when the PWM channel is working in order to filter unexpected shot-time polarity transition, and therefore the interrupt is asserted several cycles after the input waveform polarity changes, and so does the change of the values of PWMx_PERIOD_HPC and PWMx_DUTY_LPC.

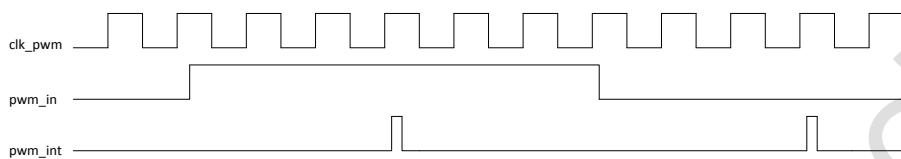


Fig. 5-2 PWM Capture Mode

5.3.2 Continuous mode

The PWM channel generates a series of the pulses continuously as expected once the channel is enabled with continuous mode.

In the continuous mode, the PWM output waveforms can be in one form of the two output mode: left-aligned mode or center-aligned mode.

For the left-aligned output mode, the PWM channel firstly starts the duty cycle with the configured duty polarity (PWMx_CTRL.duty_pol). Once duty cycle number (PWMx_DUTY_LPC) is reached, the output is switched to the opposite polarity. After the period number (PWMx_PERIOD_HPC) is reached, the output is again switched to the opposite polarity to start another period of desired pulse.

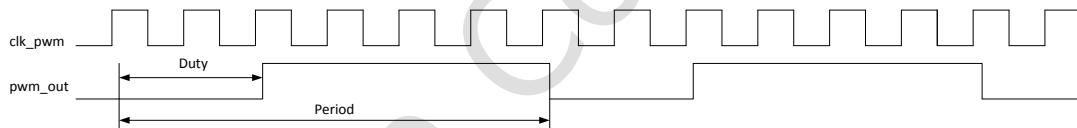


Fig. 5-3 PWM Continuous Left-aligned Output Mode

For the center-aligned output mode, the PWM channel firstly starts the duty cycle with the configured duty polarity (PWMx_CTRL.duty_pol). Once one half of duty cycle number (PWMx_DUTY_LPC) is reached, the output is switched to the opposite polarity. Then if there is one half of duty cycle left for the whole period, the output is again switched to the opposite polarity. Finally after the period number (PWMx_PERIOD_HPC) is reached, the output starts another period of desired pulse.

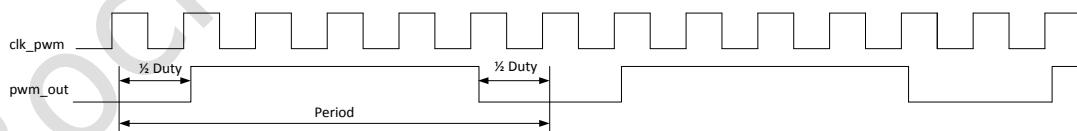


Fig. 5-4 PWM Continuous Center-aligned Output Mode

Once disable the PWM channel, the channel stops generating the output waveforms and output polarity is fixed as the configured inactive polarity (PWMx_CTRL.inactive_pol).

5.3.3 One-shot mode

Unlike the continuous mode, the PWM channel generates the output waveforms within the configured periods (PWM_CTRL.rpt + 1), and then stops. At the same times, an interrupt is asserted to inform that the operation has been finished.

There are also two output modes for the one-shot mode: the left-aligned mode and the center-aligned mode.

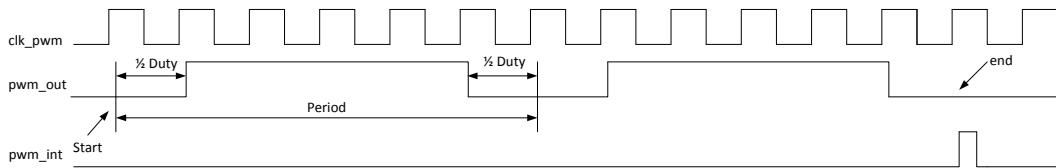


Fig. 5-5 PWM One-shot Center-aligned Output Mode

5.4 Register Description

5.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
PWM_PWM0_CNT	0x0000	W	0x00000000	PWM Channel 0 Counter Register
PWM_PWM0_PERIOD_HPR	0x0004	W	0x00000000	PWM Channel 0 Period Register/High Polarity Capture Register
PWM_PWM0_DUTY_LPR	0x0008	W	0x00000000	PWM Channel 0 Duty Register/Low Polarity Capture Register
PWM_PWM0_CTRL	0x000c	W	0x00000000	PWM Channel 0 Control Register
PWM_PWM1_CNT	0x0010	W	0x00000000	PWM Channel 1 Counter Register
PWM_PWM1_PERIOD_HPR	0x0014	W	0x00000000	PWM Channel 1 Period Register/High Polarity Capture Register
PWM_PWM1_DUTY_LPR	0x0018	W	0x00000000	PWM Channel 1 Duty Register/Low Polarity Capture Register
PWM_PWM1_CTRL	0x001c	W	0x00000000	PWM Channel 1 Control Register
PWM_PWM2_CNT	0x0020	W	0x00000000	PWM Channel 2 Counter Register
PWM_PWM2_PERIOD_HPR	0x0024	W	0x00000000	PWM Channel 2 Period Register/High Polarity Capture Register
PWM_PWM2_DUTY_LPR	0x0028	W	0x00000000	PWM Channel 2 Duty Register/Low Polarity Capture Register
PWM_PWM2_CTRL	0x002c	W	0x00000000	PWM Channel 2 Control Register
PWM_PWM3_CNT	0x0030	W	0x00000000	PWM Channel 3 Counter Register
PWM_PWM3_PERIOD_HPR	0x0034	W	0x00000000	PWM Channel 3 Period Register/High Polarity Capture Register
PWM_PWM3_DUTY_LPR	0x0038	W	0x00000000	PWM Channel 3 Duty Register/Low Polarity Capture Register
PWM_PWM3_CTRL	0x003c	W	0x00000000	PWM Channel 3 Control Register
PWM_INTSTS	0x0040	W	0x00000000	Interrupt Status Register
PWM_INT_EN	0x0044	W	0x00000000	Interrupt Enable Register

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

5.4.2 Detail Register Description

PWM_PWM0_CNT

Address: Operational Base + offset (0x0000)

PWM Channel 0 Counter Register

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

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	CNT Timer Counter The 32-bit indicates current value of PWM Channel 0 counter. The counter runs at the rate of PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM0_PERIOD_HPR

Address: Operational Base + offset (0x0004)

PWM Channel 0 Period Register/High Polarity Capture Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	PERIOD_HPR Output Waveform Period/Input Waveform High Polarity Cycle If PWM is operated at the continuous mode or one-shot mode, this value defines the period of the output waveform. Note that, if the PWM is operated at the center-aligned mode, the period should be an even one, and therefore only the bit [31:1] is taken into account and bit [0] always considered as 0. If PWM is operated at the capture mode, this value indicates the effective high polarity cycles of input waveform. This value is based on the PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM0_DUTY_LPR

Address: Operational Base + offset (0x0008)

PWM Channel 0 Duty Register/Low Polarity Capture Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	DUTY_LPR Output Waveform Duty Cycle/Input Waveform Low Polarity Cycle If PWM is operated at the continuous mode or one-shot mode, this value defines the duty cycle of the output waveform. The PWM starts the output waveform with duty cycle. Note that, if the PWM is operated at the center-aligned mode, the period should be an even one, and therefore only the [31:1] is taken into account. If PWM is operated at the capture mode, this value indicates the effective low polarity cycles of input waveform. This value is based on the PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM0_CTRL

Address: Operational Base + offset (0x000c)

PWM Channel 0 Control Register

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:24	RW	0x00	rpt Repeat Counter This field defines the repeated effective periods of output waveform in one-shot mode. The value N means N+1 repeated effective periods.
23:16	RW	0x00	scale Scale Factor This field defines the scale factor applied to prescaled clock. The value N means the clock is divided by 2*N. If N is 0, it means that the clock is divided by 512(2*256).
15	RO	0x0	reserved
14:12	RW	0x0	prescale Prescale Factor This field defines the prescale factor applied to input clock. The value N means that the input clock is divided by 2^N.
11:10	RO	0x0	reserved
9	RW	0x0	clk_sel Clock Source Select 0: non-scaled clock is selected as PWM clock source. It means that the prescale clock is directly used as the PWM clock source 1: scaled clock is selected as PWM clock source
8	RW	0x0	lp_en Low Power Mode Enable 0: disabled 1: enabled When PWM channel is inactive state and Low Power Mode is enabled, the path to PWM Clock prescale module is blocked to reduce power consumption.
7:6	RO	0x0	reserved
5	RW	0x0	output_mode PWM Output mode 0: left aligned mode 1: center aligned mode
4	RW	0x0	inactive_pol Inactive State Output Polarity This defines the output waveform polarity when PWM channel is in inactive state. The inactive state means that PWM finishes the complete waveform in one-shot mode or PWM channel is disabled. 0: negative 1: positive

Bit	Attr	Reset Value	Description
3	RW	0x0	duty_pol Duty Cycle Output Polarity This defines the polarity for duty cycle. PWM starts the output waveform with duty cycle. 0: negative 1: positive
2:1	RW	0x0	pwm_mode PWM Operation Mode 00: One shot mode. PWM produces the waveform within the repeated times defined by PWMx_CTRL_rpt. 01: Continuous mode. PWM produces the waveform continuously 10: Capture mode. PWM measures the cycles of high/low polarity of input waveform. 11: reserved
0	RW	0x0	pwm_en PWM channel enable 0: disabled 1: enabled. If the PWM is worked in the one-shot mode, this bit will be cleared at the end of operation

PWM_PWM1_CNT

Address: Operational Base + offset (0x0010)

PWM Channel 1 Counter Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	CNT Timer Counter The 32-bit indicates current value of PWM Channel 1 counter. The counter runs at the rate of PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM1_PERIOD_HPR

Address: Operational Base + offset (0x0014)

PWM Channel 1 Period Register/High Polarity Capture Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	PERIOD_HPR Output Waveform Period/Input Waveform High Polarity Cycle If PWM is operated at the continuous mode or one-shot mode, this value defines the period of the output waveform. Note that, if the PWM is operated at the center-aligned mode, the period should be an even one, and therefore only the bit [31:1] is taken into account and bit [0] always considered as 0. If PWM is operated at the capture mode, this value indicates the effective high polarity cycles of input waveform. This value is based on the PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM1_DUTY_LPR

Address: Operational Base + offset (0x0018)

PWM Channel 1 Duty Register/Low Polarity Capture Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	DUTY_LPR Output Waveform Duty Cycle/Input Waveform Low Polarity Cycle If PWM is operated at the continuous mode or one-shot mode, this value defines the duty cycle of the output waveform. The PWM starts the output waveform with duty cycle. Note that, if the PWM is operated at the center-aligned mode, the period should be an even one, and therefore only the [31:1] is taken into account. If PWM is operated at the capture mode, this value indicates the effective low polarity cycles of input waveform. This value is based on the PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM1_CTRL

Address: Operational Base + offset (0x001c)

PWM Channel 1 Control Register

Bit	Attr	Reset Value	Description
31:24	RW	0x00	rpt Repeat Counter This field defines the repeated effective periods of output waveform in one-shot mode. The value N means N+1 repeated effective periods.
23:16	RW	0x00	scale Scale Factor This field defines the scale factor applied to prescaled clock. The value N means the clock is divided by 2^N . If N is 0, it means that the clock is divided by 512(2^{10}).
15	RO	0x0	reserved
14:12	RW	0x0	prescale Prescale Factor This field defines the prescale factor applied to input clock. The value N means that the input clock is divided by 2^N .
11:10	RO	0x0	reserved
9	RW	0x0	clk_sel Clock Source Select 0: non-scaled clock is selected as PWM clock source. It means that the prescale clock is directly used as the PWM clock source 1: scaled clock is selected as PWM clock source

Bit	Attr	Reset Value	Description
8	RW	0x0	lp_en Low Power Mode Enable 0: disabled 1: enabled When PWM channel is inactive state and Low Power Mode is enabled, the path to PWM Clock prescale module is blocked to reduce power consumption.
7:6	RO	0x0	reserved
5	RW	0x0	output_mode PWM Output mode 0: left aligned mode 1: center aligned mode
4	RW	0x0	inactive_pol Inactive State Output Polarity This defines the output waveform polarity when PWM channel is in inactive state. The inactive state means that PWM finishes the complete waveform in one-shot mode or PWM channel is disabled. 0: negative 1: positive
3	RW	0x0	duty_pol Duty Cycle Output Polarity This defines the polarity for duty cycle. PWM starts the output waveform with duty cycle. 0: negative 1: positive
2:1	RW	0x0	pwm_mode PWM Operation Mode 00: One shot mode. PWM produces the waveform within the repeated times defined by PWMx_CTRL_rpt 01: Continuous mode. PWM produces the waveform continuously 10: Capture mode. PWM measures the cycles of high/low polarity of input waveform. 11: reserved
0	RW	0x0	pwm_en PWM channel enable 0: disabled 1: enabled. If the PWM is worked in the one-shot mode, this bit will be cleared at the end of operation

PWM_PWM2_CNT

Address: Operational Base + offset (0x0020)

PWM Channel 2 Counter Register

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

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	CNT Timer Counter The 32-bit indicates current value of PWM Channel 2 counter. The counter runs at the rate of PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM2_PERIOD_HPR

Address: Operational Base + offset (0x0024)

PWM Channel 2 Period Register/High Polarity Capture Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	PERIOD_HPR Output Waveform Period/Input Waveform High Polarity Cycle If PWM is operated at the continuous mode or one-shot mode, this value defines the period of the output waveform. Note that, if the PWM is operated at the center-aligned mode, the period should be an even one, and therefore only the bit [31:1] is taken into account and bit [0] always considered as 0. If PWM is operated at the capture mode, this value indicates the effective high polarity cycles of input waveform. This value is based on the PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM2_DUTY_LPR

Address: Operational Base + offset (0x0028)

PWM Channel 2 Duty Register/Low Polarity Capture Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	DUTY_LPR Output Waveform Duty Cycle/Input Waveform Low Polarity Cycle If PWM is operated at the continuous mode or one-shot mode, this value defines the duty cycle of the output waveform. The PWM starts the output waveform with duty cycle. Note that, if the PWM is operated at the center-aligned mode, the period should be an even one, and therefore only the [31:1] is taken into account. If PWM is operated at the capture mode, this value indicates the effective low polarity cycles of input waveform. This value is based on the PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM2_CTRL

Address: Operational Base + offset (0x002c)

PWM Channel 2 Control Register

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

Bit	Attr	Reset Value	Description
31:24	RW	0x00	rpt Repeat Counter This field defines the repeated effective periods of output waveform in one-shot mode. The value N means N+1 repeated effective periods.
23:16	RW	0x00	scale Scale Factor This field defines the scale factor applied to prescaled clock. The value N means the clock is divided by 2*N. If N is 0, it means that the clock is divided by 512(2*256).
15	RO	0x0	reserved
14:12	RW	0x0	prescale Prescale Factor This field defines the prescale factor applied to input clock. The value N means that the input clock is divided by 2^N.
11:10	RO	0x0	reserved
9	RW	0x0	clk_sel Clock Source Select 0: non-scaled clock is selected as PWM clock source. It means that the prescale clock is directly used as the PWM clock source 1: scaled clock is selected as PWM clock source
8	RW	0x0	lp_en Low Power Mode Enable 0: disabled 1: enabled When PWM channel is inactive state and Low Power Mode is enabled, the path to PWM Clock prescale module is blocked to reduce power consumption.
7:6	RO	0x0	reserved
5	RW	0x0	output_mode PWM Output mode 0: left aligned mode 1: center aligned mode
4	RW	0x0	inactive_pol Inactive State Output Polarity This defines the output waveform polarity when PWM channel is in inactive state. The inactive state means that PWM finishes the complete waveform in one-shot mode or PWM channel is disabled. 0: negative 1: positive

Bit	Attr	Reset Value	Description
3	RW	0x0	duty_pol Duty Cycle Output Polarity This defines the polarity for duty cycle. PWM starts the output waveform with duty cycle. 0: negative 1: positive
2:1	RW	0x0	pwm_mode PWM Operation Mode 00: One shot mode. PWM produces the waveform within the repeated times defined by PWMx_CTRL_rpt. 01: Continuous mode. PWM produces the waveform continuously 10: Capture mode. PWM measures the cycles of high/low polarity of input waveform. 11: reserved
0	RW	0x0	pwm_en PWM channel enable 0: disabled 1: enabled. If the PWM is worked in the one-shot mode, this bit will be cleared at the end of operation

PWM_PWM3_CNT

Address: Operational Base + offset (0x0030)

PWM Channel 3 Counter Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	CNT Timer Counter The 32-bit indicates current value of PWM Channel 3 counter. The counter runs at the rate of PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM3_PERIOD_HPR

Address: Operational Base + offset (0x0034)

PWM Channel 3 Period Register/High Polarity Capture Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	PERIOD_HPR Output Waveform Period/Input Waveform High Polarity Cycle If PWM is operated at the continuous mode or one-shot mode, this value defines the period of the output waveform. Note that, if the PWM is operated at the center-aligned mode, the period should be an even one, and therefore only the bit [31:1] is taken into account and bit [0] always considered as 0. If PWM is operated at the capture mode, this value indicates the effective high polarity cycles of input waveform. This value is based on the PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM3_DUTY_LPR

Address: Operational Base + offset (0x0038)

PWM Channel 3 Duty Register/Low Polarity Capture Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	DUTY_LPR Output Waveform Duty Cycle/Input Waveform Low Polarity Cycle If PWM is operated at the continuous mode or one-shot mode, this value defines the duty cycle of the output waveform. The PWM starts the output waveform with duty cycle. Note that, if the PWM is operated at the center-aligned mode, the period should be an even one, and therefore only the [31:1] is taken into account. If PWM is operated at the capture mode, this value indicates the effective low polarity cycles of input waveform. This value is based on the PWM clock. The value ranges from 0 to ($2^{32}-1$).

PWM_PWM3_CTRL

Address: Operational Base + offset (0x003c)

PWM Channel 3 Control Register

Bit	Attr	Reset Value	Description
31:24	RW	0x00	rpt Repeat Counter This field defines the repeated effective periods of output waveform in one-shot mode. The value N means N+1 repeated effective periods.
23:16	RW	0x00	scale Scale Factor This field defines the scale factor applied to prescaled clock. The value N means the clock is divided by 2^N . If N is 0, it means that the clock is divided by 512(2^{10}).
15	RO	0x0	reserved
14:12	RW	0x0	prescale Prescale Factor This field defines the prescale factor applied to input clock. The value N means that the input clock is divided by 2^N .
11:10	RO	0x0	reserved
9	RW	0x0	clk_sel Clock Source Select 0: non-scaled clock is selected as PWM clock source. It means that the prescale clock is directly used as the PWM clock source 1: scaled clock is selected as PWM clock source

Bit	Attr	Reset Value	Description
8	RW	0x0	lp_en Low Power Mode Enable 0: disabled 1: enabled When PWM channel is inactive state and Low Power Mode is enabled, the path to PWM Clock prescale module is blocked to reduce power consumption.
7:6	RO	0x0	reserved
5	RW	0x0	output_mode PWM Output mode 0: left aligned mode 1: center aligned mode
4	RW	0x0	inactive_pol Inactive State Output Polarity This defines the output waveform polarity when PWM channel is in inactive state. The inactive state means that PWM finishes the complete waveform in one-shot mode or PWM channel is disabled. 0: negative 1: positive
3	RW	0x0	duty_pol Duty Cycle Output Polarity This defines the polarity for duty cycle. PWM starts the output waveform with duty cycle. 0: negative 1: positive
2:1	RW	0x0	pwm_mode PWM Operation Mode 00: One shot mode. PWM produces the waveform within the repeated times defined by PWMx_CTRL_rpt 01: Continuous mode. PWM produces the waveform continuously 10: Capture mode. PWM measures the cycles of high/low polarity of input waveform. 11: reserved
0	RW	0x0	pwm_en PWM channel enable 0: disabled 1: enabled. If the PWM is worked in the one-shot mode, this bit will be cleared at the end of operation

PWM_INTSTS

Address: Operational Base + offset (0x0040)

Interrupt Status Register

Bit	Attr	Reset Value	Description
31:12	RO	0x0	reserved

Bit	Attr	Reset Value	Description
11	RO	0x0	<p>CH3_Pol Channel 3 Interrupt Polarity Flag This bit is used in capture mode in order to identify the transition of the input waveform when interrupt is generated. When bit is 1, please refer to PWM3_PERIOD_HPR to know the effective high cycle of Channel 3 input waveform. Otherwise, please refer to PWM3_PERIOD_LPR to know the effective low cycle of Channel 3 input waveform. Write 1 to CH3_IntSts will clear this bit.</p>
10	RO	0x0	<p>CH2_Pol Channel 2 Interrupt Polarity Flag This bit is used in capture mode in order to identify the transition of the input waveform when interrupt is generated. When bit is 1, please refer to PWM2_PERIOD_HPR to know the effective high cycle of Channel 2 input waveform. Otherwise, please refer to PWM2_PERIOD_LPR to know the effective low cycle of Channel 2 input waveform. Write 1 to CH2_IntSts will clear this bit.</p>
9	RO	0x0	<p>CH1_Pol Channel 1 Interrupt Polarity Flag This bit is used in capture mode in order to identify the transition of the input waveform when interrupt is generated. When bit is 1, please refer to PWM1_PERIOD_HPR to know the effective high cycle of Channel 1 input waveform. Otherwise, please refer to PWM1_PERIOD_LPR to know the effective low cycle of Channel 1 input waveform. Write 1 to CH1_IntSts will clear this bit.</p>
8	RO	0x0	<p>CH0_Pol Channel 0 Interrupt Polarity Flag This bit is used in capture mode in order to identify the transition of the input waveform when interrupt is generated. When bit is 1, please refer to PWM0_PERIOD_HPR to know the effective high cycle of Channel 0 input waveform. Otherwise, please refer to PWM0_PERIOD_LPR to know the effective low cycle of Channel 0 input waveform. Write 1 to CH0_IntSts will clear this bit.</p>
7:4	RO	0x0	reserved
3	RW	0x0	<p>CH3_IntSts Channel 3 Interrupt Status 0: Channel 3 Interrupt not generated 1: Channel 3 Interrupt generated</p>
2	RW	0x0	<p>CH2_IntSts Channel 2 Interrupt Status 0: Channel 2 Interrupt not generated 1: Channel 2 Interrupt generated</p>
1	RW	0x0	<p>CH1_IntSts Channel 1 Interrupt Status 0: Channel 1 Interrupt not generated 1: Channel 1 Interrupt generated</p>

Bit	Attr	Reset Value	Description
0	RW	0x0	CH0_IntSts Channel 0 Raw Interrupt Status 0: Channel 0 Interrupt not generated 1: Channel 0 Interrupt generated

PWM_INT_EN

Address: Operational Base + offset (0x0044)

Interrupt Enable Register

Bit	Attr	Reset Value	Description
31:4	RO	0x0	reserved
3	RW	0x0	CH3_Int_en Channel 3 Interrupt Enable 0: Channel 3 Interrupt disabled 1: Channel 3 Interrupt enabled
2	RW	0x0	CH2_Int_en Channel 2 Interrupt Enable 0: Channel 2 Interrupt disabled 1: Channel 2 Interrupt enabled
1	RW	0x0	CH1_Int_en Channel 1 Interrupt Enable 0: Channel 1 Interrupt disabled 1: Channel 1 Interrupt enabled
0	RW	0x0	CH0_Int_en Channel 0 Interrupt Enable 0: Channel 0 Interrupt disabled 1: Channel 0 Interrupt enabled

5.5 Interface Description

Table 5-1 PWM Interface Description

Module Pin	Direction	Pad Name	IOMUX Setting
pwm0	I/O	IO_PWM0_GPIO0d2	GRF_GPIO0D_IOMUX[4]=1'b1
pwm1	I/O	IO_I2C0pmuscl_PWM1_GPIO0a0	GRF_GPIO0A_IOMUX[1:0]=2'b10
		IO_I2Slrckrx_PWM10_GPIO1a2	GRF_GPIO1A_IOMUX[5:4]=2'b10 &GRF_SOC_CON1[5]=1'b1
pwm2	I/O	IO_I2C0pmusda_PWM2_GPIO0a1	GRF_GPIO0A_IOMUX[3:2]=2'b10

Notes: I=input, O=output, I/O=input/output, bidirectional

5.6 Application Notes

5.6.1 PWM Capture Mode Standard Usage Flow

1. Set PWMx_CTRL.pwm_en to '0' to disable the PWM channel.
2. Choose the prescale factor and the scale factor for pclk by programming PWMx_CTRL.prescale and PWMx_CTRL.scale, and select the clock needed by setting PWMx_CTRL.clk_sel.

3. Configure the channel to work in the capture mode.
4. Enable the INT_EN.chx_int_en to enable the interrupt generation.
5. Enable the channel by writing '1' to PWMx_CTRL.pwm_en bit to start the channel.
6. When an interrupt is asserted, refer to INTSTS register to know the raw interrupt status. If the corresponding polarity flag is set, turn to PWMx_PERIOD_HPC register to know the effective high cycles of input waveforms, otherwise turn to PWMx_DUTY_LPC register to know the effective low cycles.
7. Write '0' to PWMx_CTRL.pwm_en to disable the channel.

5.6.2 PWM One-shot Mode/Continuous Standard Usage Flow

1. Set PWMx_CTRL.pwm_en to '0' to disable the PWM channel.
2. Choose the prescale factor and the scale factor for pclk by programming PWMx_CTRL.prescale and PWMx_CTRL.scale, and select the clock needed by setting PWMx_CTRL.clk_sel.
3. Choose the output mode by setting PWMx_CTRL.output_mode, and set the duty polarity and inactive polarity by programming PWMx_CTRL.duty_pol and PWMx_CTRL.inactive_pol.
4. Set the PWMx_CTRL.rpt if the channel is desired to work in the one-shot mode.
5. Configure the channel to work in the one-shot mode or the continuous mode.
6. Enable the INT_EN.chx_int_en to enable the interrupt generation if if the channel is desired to work in the one-shot mode.
7. If the channel is working in the one-shot mode, an interrupt is asserted after the end of operation, and the PWMx_CTRL.pwm_en is automatically cleared. Whatever mode the channel is working in, write '0' to PWMx_CTRL.pwm_en bit to disable the PWM channel.

5.6.3 Low-power mode

Setting PWMx_CTRL.ip_en to '1' makes the channel enter the low-power mode. When the PWM channel is inactive, the APB bus clock to the clock prescale module is gated in order to reduce the power consumption. It is recommended to disable the channel before entering the low-power mode, and quit the low-power mode before enabling the channel.

5.6.4 Other notes

When the channel is active to produce waveforms, it is free to program the PWMx_PERIOD_HPC and PWMx_DUTY_LPC register. The change will not take effect immediately until the current period ends.

An active channel can be changed to another operation mode without disable the PWM channel. However, during the transition of the operation mode there may be some irregular output waveforms. So does changing the clock division factor when the channel is active.

Chapter 6 UART

6.1 Overview

The Universal Asynchronous Receiver/Transmitter (UART) is used for serial communication with a peripheral, modem (data carrier equipment, DCE) or data set. Data is written from a master (CPU) over the APB bus to the UART and it is converted to serial form and transmitted to the destination device. Serial data is also received by the UART and stored for the master (CPU) to read back.

UART Controller supports the following features:

- Support 3 independent UART controller: UART0, UART1, UART2
- UART0 contains two 64Bytes FIFOs for BT transfer, UART1/UART2 contains two 32Bytes FIFOs for data receive and transmit
- UART0 supports auto flow-control, UART1/UART2 do not support auto flow-control
- Input clock can operate at a higher speed
- Support bit rates 115.2Kbps, 460.8Kbps, 921.6Kbps, 1.5Mbps, 3Mbps, 4Mbps
- Support programmable baud rates, even with non-integer clock divider
- Standard asynchronous communication bits (start, stop and parity)
- Support interrupt-based or DMA-based mode
- Support 5-8 bits width transfer

6.2 Block Diagram

This section provides a description about the functions and behavior under various conditions. The UART Controller comprises with:

- AMBA APB interface
- FIFO controllers
- Register block
- Modem synchronization block and baud clock generation block
- Serial receiver and serial transmitter

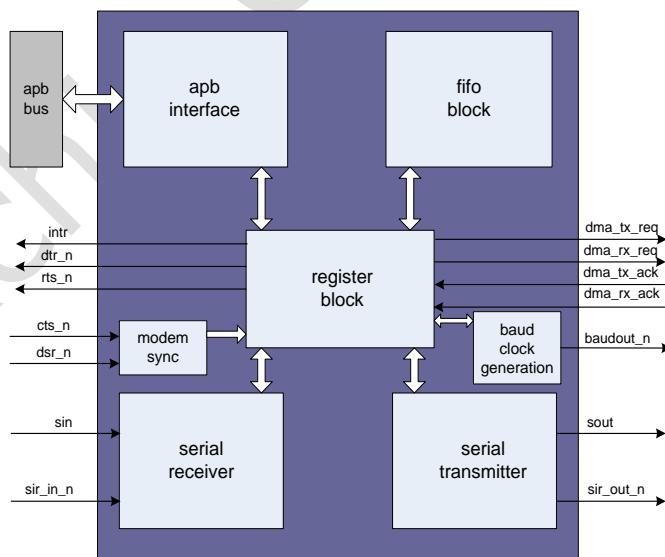


Fig. 6-1 UART Architecture

APB INTERFACE

The host processor accesses data, control, and status information on the UART through the APB interface. The UART supports APB data bus widths of 8, 16, and 32 bits.

Register block

Be responsible for the main UART functionality including control, status and interrupt generation.

Modem Synchronization block

Synchronizes the modem input signal.

FIFO block

Be responsible for FIFO control and storage (when using internal RAM) or signaling to control external RAM (when used).

Baud Clock Generator

Generates the transmitter and receiver baud clock along with the output reference clock signal (baudout_n).

Serial Transmitter

Converts the parallel data, written to the UART, into serial form and adds all additional bits, as specified by the control register, for transmission. This makeup of serial data, referred to as a character can exit the block in two forms, either serial UART format or IrDA 1.0 SIR format.

Serial Receiver

Converts the serial data character (as specified by the control register) received in either the UART or IrDA 1.0 SIR format to parallel form. Parity error detection, framing error detection and line break detection is carried out in this block.

6.3 Function Description

UART (RS232) Serial Protocol

Because the serial communication is asynchronous, additional bits (start and stop) are added to the serial data to indicate the beginning and end. An additional parity bit may be added to the serial character. This bit appears after the last data bit and before the stop bit(s) in the character structure to perform simple error checking on the received data, as shown in Figure.

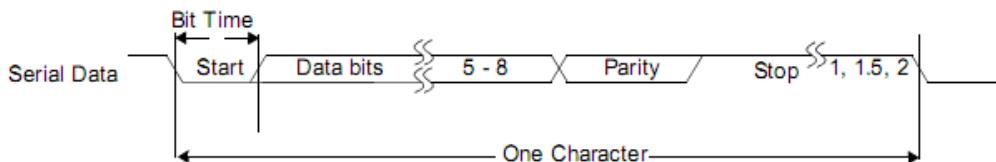


Fig. 6-2 UART Serial protocol

IrDA 1.0 SIR Protocol

The Infrared Data Association (IrDA) 1.0 Serial Infrared (SIR) mode supports bi-directional data communications with remote devices using infrared radiation as the transmission medium. IrDA 1.0 SIR mode specifies a maximum baud rate of 115.2 Kbaud.

Transmitting a single infrared pulse signals a logic zero, while a logic one is represented by not sending a pulse. The width of each pulse is 3/16ths of a normal serial bit time. Data transfers can only occur in half-duplex fashion when IrDA SIR mode is enabled.

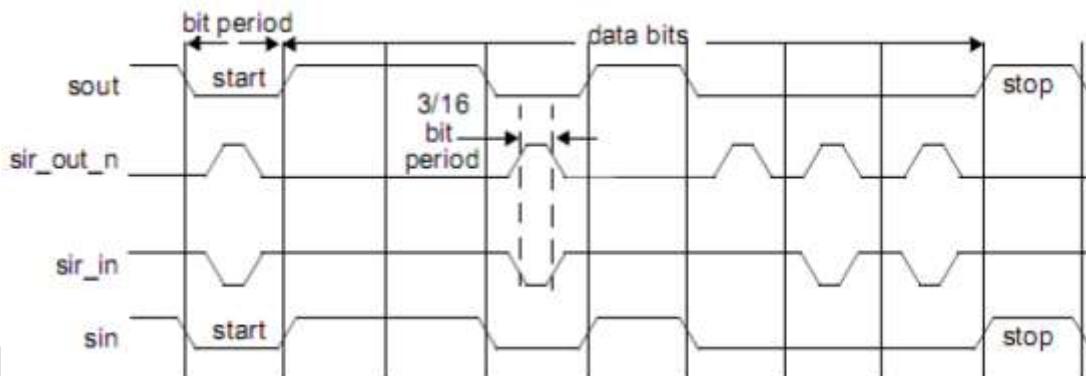


Fig. 6-3 IrDA 1.0

Baud Clock

The baud rate is controlled by the serial clock (sclk or pclk in a single clock implementation) and the Divisor Latch Register (DLH and DLL). As the exact number of baud clocks that each bit was transmitted for is known, calculating the mid-point for sampling is not difficult, that is every 16 baud clocks after the mid-point sample of the start bit.

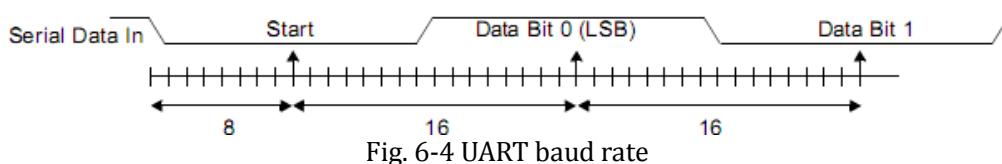


Fig. 6-4 UART baud rate

FIFO Support

1. NONE FIFO MODE

If FIFO support is not selected, then no FIFOs are implemented and only a single receive data byte and transmit data byte can be stored at a time in the RBR and THR.

2. FIFO MODE

The FIFO depth of UART1/UART2 is 32bytes and the FIFO depth of UART0 is 64bytes. The FIFO mode of all the UART is enabled by register FCR[0].

Interrupts

The following interrupt types can be enabled with the IER register.

- Receiver Error
- Receiver Data Available
- Character Timeout (in FIFO mode only)
- Transmitter Holding Register Empty at/below threshold (in Programmable THRE Interrupt mode)
- Modem Status

DMA Support

The UART supports DMA signaling with the use of two output signals (dma_tx_req_n and dma_rx_req_n) to indicate when data is ready to be read or when the transmit FIFO is empty.

The dma_tx_req_n signal is asserted under the following conditions:

- When the Transmitter Holding Register is empty in non-FIFO mode.
- When the transmitter FIFO is empty in FIFO mode with Programmable THRE interrupt mode disabled.
- When the transmitter FIFO is at, or below the programmed threshold with Programmable THRE interrupt mode enabled.

The dma_rx_req_n signal is asserted under the following conditions:

- When there is a single character available in the Receive Buffer Register in non-FIFO mode.
- When the Receiver FIFO is at or above the programmed trigger level in FIFO mode.

Auto Flow Control

The UART can be configured to have a 16750-compatible Auto RTS and Auto CTS serial data flow control mode available. If FIFOs are not implemented, then this mode cannot be selected. When Auto Flow Control mode has been selected, it can be enabled with the Modem Control Register (MCR[5]). Following figure shows a block diagram of the Auto Flow Control functionality.

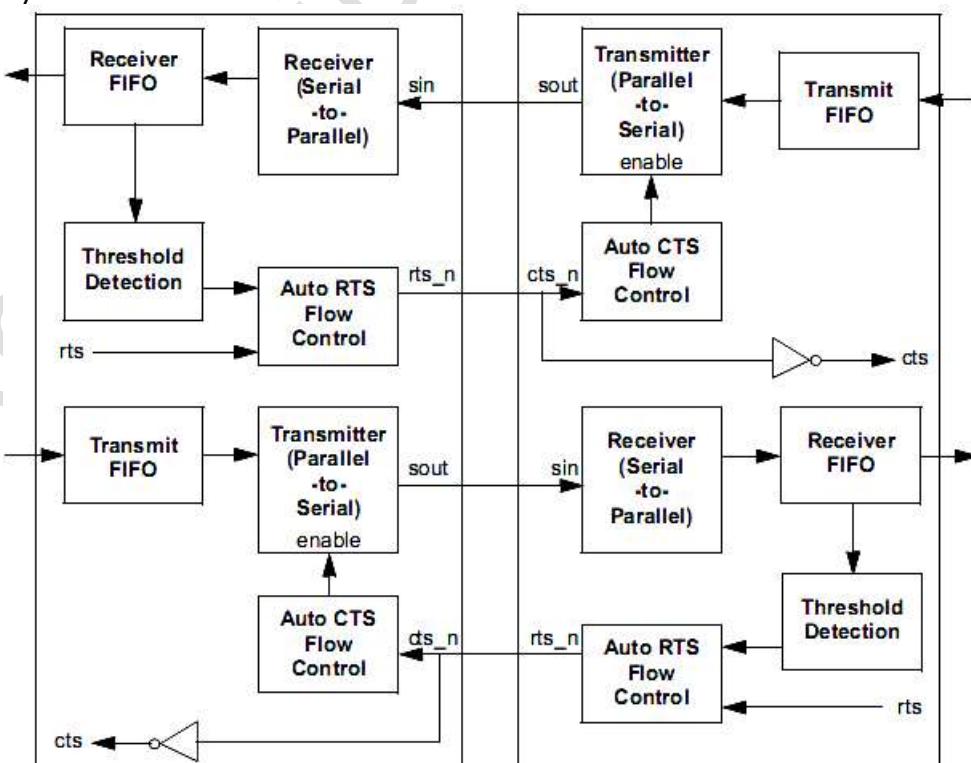


Fig. 6-5 UART Auto flow control block diagram

Auto RTS – Becomes active when the following occurs:

- Auto Flow Control is selected during configuration
- FIFOs are implemented
- RTS (MCR[1] bit and MCR[5]bit are both set)
- FIFOs are enabled (FCR[0]) bit is set)
- SIR mode is disabled (MCR[6] bit is not set)

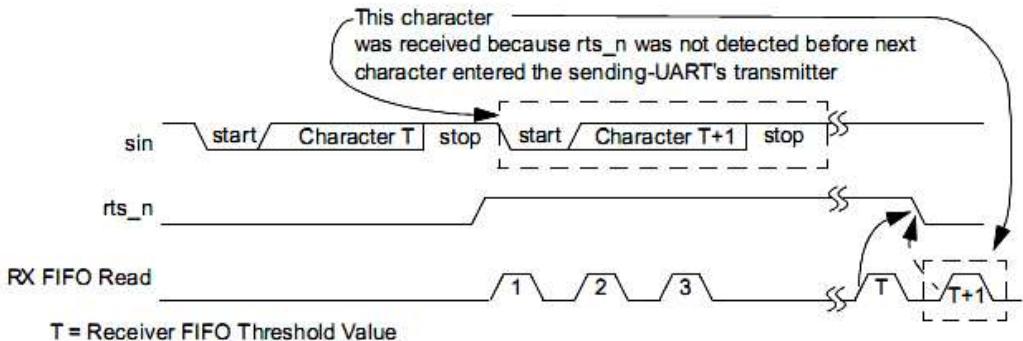


Fig. 6-6 UART AUTO RTS TIMING

Auto CTS – becomes active when the following occurs:

- Auto Flow Control is selected during configuration
- FIFOs are implemented
- AFCE (MCR[5] bit is set)
- FIFOs are enabled through FIFO Control Register FCR[0] bit
- SIR mode is disabled (MCR[6] bit is not set)

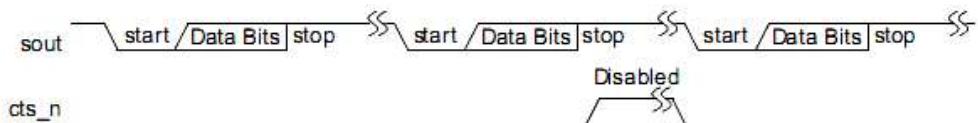


Fig. 6-7 UART AUTO CTS TIMING

6.4 Register Description

This section describes the control/status registers of the design. There are 3 UARTs in RK3036, and each one has its own base address.

6.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
UART_RBR	0x0000	W	0x00000000	Receive Buffer Register
UART_THR	0x0000	W	0x00000000	Transmit Holding Register
UART_DLL	0x0000	W	0x00000000	Divisor Latch (Low)
UART_DLH	0x0004	W	0x00000000	Divisor Latch (High)
UART_IER	0x0004	W	0x00000000	Interrupt Enable Register
UART_IIR	0x0008	W	0x00000000	Interrupt Identification Register
UART_FCR	0x0008	W	0x00000000	FIFO Control Register
UART_LCR	0x000c	W	0x00000000	Line Control Register
UART_MCR	0x0010	W	0x00000000	Modem Control Register
UART_LSR	0x0014	W	0x00000000	Line Status Register
UART_MSR	0x0018	W	0x00000000	Modem Status Register
UART_SCR	0x001c	W	0x00000000	Scratchpad Register
UART_SRBR	0x0030	W	0x00000000	Shadow Receive Buffer Register
UART_STHR	0x006c	W	0x00000000	Shadow Transmit Holding Register
UART_FAR	0x0070	W	0x00000000	FIFO Access Register

Name	Offset	Size	Reset Value	Description
UART_TFR	0x0074	W	0x00000000	Transmit FIFO Read
UART_RFW	0x0078	W	0x00000000	Receive FIFO Write
UART_USR	0x007c	W	0x00000000	UART Status Register
UART_TFL	0x0080	W	0x00000000	Transmit FIFO Level
UART_RFL	0x0084	W	0x00000000	Receive FIFO Level
UART_SRR	0x0088	W	0x00000000	Software Reset Register
UART_SRTS	0x008c	W	0x00000000	Shadow Request to Send
UART_SBCR	0x0090	W	0x00000000	Shadow Break Control Register
UART_SDMAM	0x0094	W	0x00000000	Shadow DMA Mode
UART_SFE	0x0098	W	0x00000000	Shadow FIFO Enable
UART_SRT	0x009c	W	0x00000000	Shadow RCVR Trigger
UART_STET	0x00a0	W	0x00000000	Shadow TX Empty Trigger
UART_HTX	0x00a4	W	0x00000000	Halt TX
UART_DMASA	0x00a8	W	0x00000000	DMA Software Acknowledge
UART_CPR	0x00f4	W	0x00000000	Component Parameter Register
UART_UCV	0x00f8	W	0x0330372a	UART Component Version
UART_CTR	0x00fc	W	0x44570110	Component Type Register

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

6.4.2 Detail Register Description

UART_RBR

Address: Operational Base + offset (0x0000)

Receive Buffer Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	RW	0x00	<p>data_input</p> <p>Data byte received on the serial input port (sin) in UART mode, or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line Status Register (LCR) is set.</p> <p>If in non-FIFO mode (FIFO_MODE == NONE) or FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it is overwritten, resulting in an over-run error.</p> <p>If in FIFO mode (FIFO_MODE != NONE) and FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO.</p> <p>If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO is preserved, but any incoming data are lost and an over-run error occurs.</p>

UART_THR

Address: Operational Base + offset (0x0000)

Transmit Holding Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	RW	0x00	<p>data_output Data to be transmitted on the serial output port (sout) in UART mode or the serial infrared output (sir_out_n) in infrared mode. Data should only be written to the THR when the THR Empty (THRE) bit (LSR[5]) is set.</p> <p>If in non-FIFO mode or FIFOs are disabled (FCR[0] = 0) and THRE is set, writing a single character to the THR clears the THRE. Any additional writes to the THR before the THRE is set again causes the THR data to be overwritten.</p> <p>If in FIFO mode and FIFOs are enabled (FCR[0] = 1) and THRE is set, x number of characters of data may be written to the THR before the FIFO is full. The number x (default=16) is determined by the value of FIFO Depth that you set during configuration. Any attempt to write data when the FIFO is full results in the write data being lost.</p>

UART_DLL

Address: Operational Base + offset (0x0000)

Divisor Latch (Low)

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	RW	0x00	<p>baud_rate_divisor_L Lower 8-bits of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART. This register may only be accessed when the DLAB bit (LCR[7]) is set and the UART is not busy (USR[0] is zero). The output baud rate is equal to the serial clock (sclk) frequency divided by sixteen times the value of the baud rate divisor, as follows: baud rate = (serial clock freq) / (16 * divisor).</p> <p>Note that with the Divisor Latch Registers (DLL and DLH) set to zero, the baud clock is disabled and no serial communications occur. Also, once the DLH is set, at least 8 clock cycles of the slowest UART clock should be allowed to pass before transmitting or receiving data.</p>

UART_DLH

Address: Operational Base + offset (0x0004)

Divisor Latch (High)

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	RW	0x00	<p>baud_rate_divisor_H Upper 8 bits of a 16-bit, read/write, Divisor Latch register that contains the baud rate divisor for the UART.</p>

UART_IER

Address: Operational Base + offset (0x0004)

Interrupt Enable Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7	RW	0x0	prog_thre_int_en Programmable THRE Interrupt Mode Enable This is used to enable/disable the generation of THRE Interrupt. 0 = disabled 1 = enabled
6:4	RO	0x0	reserved
3	RW	0x0	modem_status_int_en Enable Modem Status Interrupt. This is used to enable/disable the generation of Modem Status Interrupt. This is the fourth highest priority interrupt. 0 = disabled 1 = enabled
2	RW	0x0	receive_line_status_int_en Enable Receiver Line Status Interrupt. This is used to enable/disable the generation of Receiver Line Status Interrupt. This is the highest priority interrupt. 0 = disabled 1 = enabled
1	RW	0x0	trans_hold_empty_int_en Enable Transmit Holding Register Empty Interrupt.
0	RW	0x0	receive_data_available_int_en Enable Received Data Available Interrupt. This is used to enable/disable the generation of Received Data Available Interrupt and the Character Timeout Interrupt (if in FIFO mode and FIFOs enabled). These are the second highest priority interrupts. 0 = disabled 1 = enabled

UART_IIR

Address: Operational Base + offset (0x0008)

Interrupt Identification Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:6	RO	0x0	fifos_en FIFOs Enabled. This is used to indicate whether the FIFOs are enabled or disabled. 00 = disabled 11 = enabled
5:4	RO	0x0	reserved

Bit	Attr	Reset Value	Description
3:0	RO	0x0	<p>int_id Interrupt ID This indicates the highest priority pending interrupt which can be one of the following types:</p> <ul style="list-style-type: none"> 0000 = modem status 0001 = no interrupt pending 0010 = THR empty 0100 = received data available 0110 = receiver line status 0111 = busy detect 1100 = character timeout

UART_FCR

Address: Operational Base + offset (0x0008)

FIFO Control Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:6	WO	0x0	<p>rcvr_trigger RCVR Trigger. This is used to select the trigger level in the receiver FIFO at which the Received Data Available Interrupt is generated. In auto flow control mode it is used to determine when the rts_n signal is de-asserted. It also determines when the dma_rx_req_n signal is asserted in certain modes of operation. The following trigger levels are supported:</p> <ul style="list-style-type: none"> 00 = 1 character in the FIFO 01 = FIFO 1/4 full 10 = FIFO 1/2 full 11 = FIFO 2 less than ful
5:4	WO	0x0	<p>tx_empty_trigger TX Empty Trigger. This is used to select the empty threshold level at which the THRE Interrupts are generated when the mode is active. It also determines when the dma_tx_req_n signal is asserted when in certain modes of operation. The following trigger levels are supported:</p> <ul style="list-style-type: none"> 00 = FIFO empty 01 = 2 characters in the FIFO 10 = FIFO 1/4 full 11 = FIFO 1/2 full

Bit	Attr	Reset Value	Description
3	WO	0x0	<p>dma_mode DMA Mode</p> <p>This determines the DMA signalling mode used for the dma_tx_req_n and dma_rx_req_n output signals when additional DMA handshaking signals are not selected .</p> <p>0 = mode 0 1 = mode 11100 = character timeout.</p>
2	WO	0x0	<p>xmit_fifo_reset XMIT FIFO Reset.</p> <p>This resets the control portion of the transmit FIFO and treats the FIFO as empty. This also de-asserts the DMA TX request and single signals when additional DMA handshaking signals are selected .</p> <p>Note that this bit is 'self-clearing'. It is not necessary to clear this bit.</p>
1	WO	0x0	<p>rcvr_fifo_reset RCVR FIFO Reset.</p> <p>This resets the control portion of the receive FIFO and treats the FIFO as empty. This also de-asserts the DMA RX request and single signals when additional DMA handshaking signals are selected.</p> <p>Note that this bit is 'self-clearing'. It is not necessary to clear this bit.</p>
0	WO	0x0	<p>fifo_en FIFO Enable.</p> <p>FIFO Enable. This enables/disables the transmit (XMIT) and receive (RCVR) FIFOs. Whenever the value of this bit is changed both the XMIT and RCVR controller portion of FIFOs is reset.</p>

UART_LCR

Address: Operational Base + offset (0x000c)

Line Control Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7	RW	0x0	<p>div_lat_access Divisor Latch Access Bit.</p> <p>Writeable only when UART is not busy (USR[0] is zero), always readable. This bit is used to enable reading and writing of the Divisor Latch register (DLL and DLH) to set the baud rate of the UART. This bit must be cleared after initial baud rate setup in order to access other registers.</p>

Bit	Attr	Reset Value	Description
6	RW	0x0	<p>break_ctrl Break Control Bit.</p> <p>This is used to cause a break condition to be transmitted to the receiving device. If set to one the serial output is forced to the spacing (logic 0) state. When not in Loopback Mode, as determined by MCR[4], the sout line is forced low until the Break bit is cleared. If MCR[6] set to one, the sir_out_n line is continuously pulsed. When in Loopback Mode, the break condition is internally looped back to the receiver and the sir_out_n line is forced low.</p>
5	RO	0x0	reserved
4	RW	0x0	<p>even_parity_sel Even Parity Select.</p> <p>Writeable only when UART is not busy (USR[0] is zero), always readable. This is used to select between even and odd parity, when parity is enabled (PEN set to one). If set to one, an even number of logic 1s is transmitted or checked. If set to zero, an odd number of logic 1s is transmitted or checked.</p>
3	RW	0x0	<p>parity_en Parity Enable.</p> <p>Writeable only when UART is not busy (USR[0] is zero), always readable. This bit is used to enable and disable parity generation and detection in transmitted and received serial character respectively.</p> <p>0 = parity disabled 1 = parity enabled</p>
2	RW	0x0	<p>stop_bits_num Number of stop bits.</p> <p>Writeable only when UART is not busy (USR[0] is zero), always readable. This is used to select the number of stop bits per character that the peripheral transmits and receives. If set to zero, one stop bit is transmitted in the serial data. If set to one and the data bits are set to 5 (LCR[1:0] set to zero) one and a half stop bits is transmitted. Otherwise, two stop bits are transmitted. Note that regardless of the number of stop bits selected, the receiver checks only the first stop bit.</p> <p>0 = 1 stop bit 1 = 1.5 stop bits when DLS (LCR[1:0]) is zero, else 2 stop bit.</p>

Bit	Attr	Reset Value	Description
1:0	RW	0x0	<p>data_length_sel Data Length Select.</p> <p>Writeable only when UART is not busy (USR[0] is zero), always readable. This is used to select the number of data bits per character that the peripheral transmits and receives. The number of bit that may be selected areas follows:</p> <ul style="list-style-type: none"> 00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits

UART_MCR

Address: Operational Base + offset (0x0010)

Modem Control Register

Bit	Attr	Reset Value	Description
31:7	RO	0x0	reserved
6	RW	0x0	<p>sir_mode_en SIR Mode Enable. SIR Mode Enable.</p> <p>This is used to enable/disable the IrDA SIR Mode .</p> <p>0 = IrDA SIR Mode disabled 1 = IrDA SIR Mode enabled</p>
5	RW	0x0	<p>auto_flow_ctrl_en Auto Flow Control Enable.</p> <p>0 = Auto Flow Control Mode disabled 1 = Auto Flow Control Mode enabled</p>
4	RW	0x0	<p>loopback LoopBack Bit.</p> <p>This is used to put the UART into a diagnostic mode for test purposes.</p>
3	RW	0x0	<p>out2 OUT2.</p> <p>This is used to directly control the user-designated Output2 (out2_n) output. The value written to this location is inverted and driven out on out2_n, that is:</p> <p>0 = out2_n de-asserted (logic 1) 1 = out2_n asserted (logic 0)</p>
2	RW	0x0	<p>out1 OUT1</p> <p>This is used to directly control the user-designated Output2 (out2_n) output. The value written to this location is inverted and driven out on out2_n, that is:</p> <p>1'b0: out2_n de-asserted (logic 1) 1'b1: out2_n asserted (logic 0)</p>

Bit	Attr	Reset Value	Description
1	RW	0x0	<p>req_to_send Request to Send.</p> <p>This is used to directly control the Request to Send (rts_n) output. The Request To Send (rts_n) output is used to inform the modem or data set that the UART is ready to exchange data.</p>
0	RW	0x0	<p>data_terminal_ready Data Terminal Ready.</p> <p>This is used to directly control the Data Terminal Ready (dtr_n) output. The value written to this location is inverted and driven out on dtr_n, that is:</p> <p>0 = dtr_n de-asserted (logic 1) 1 = dtr_n asserted (logic 0)</p>

UART_LSR

Address: Operational Base + offset (0x0014)

Line Status Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7	RO	0x0	<p>receiver_fifo_error Receiver FIFO Error bit.</p> <p>This bit is relevant FIFOs are enabled (FCR[0] set to one). This is used to indicate if there is at least one parity error, framing error, or break indication in the FIFO.</p> <p>0 = no error in RX FIFO 1 = error in RX FIFO</p>
6	RO	0x0	<p>trans_empty Transmitter Empty bit.</p> <p>Transmitter Empty bit. If FIFOs enabled (FCR[0] set to one), this bit is set whenever the Transmitter Shift Register and the FIFO are both empty. If FIFOs are disabled, this bit is set whenever the Transmitter Holding Register and the Transmitter Shift Register are both empty.</p>
5	RO	0x0	<p>trans_hold_reg_empty Transmit Holding Register Empty bit.</p> <p>If THRE mode is disabled (IER[7] set to zero) and regardless of FIFO's being implemented/enabled or not, this bit indicates that the THR or TX FIFO is empty.</p> <p>This bit is set whenever data is transferred from the THR or TX FIFO to the transmitter shift register and no new data has been written to the THR or TX FIFO. This also causes a THRE Interrupt to occur, if the THRE Interrupt is enabled. If IER[7] set to one and FCR[0] set to one respectively, the functionality is switched to indicate the transmitter FIFO is full, and no longer controls THRE interrupts, which are then controlled by the FCR[5:4] threshold setting.</p>

Bit	Attr	Reset Value	Description
4	RO	0x0	break_int Break Interrupt bit. This is used to indicate the detection of a break sequence on the serial input data.
3	RO	0x0	framing_error Framing Error bit. This is used to indicate the occurrence of a framing error in the receiver. A framing error occurs when the receiver does not detect a valid STOP bit in the received data.
2	RO	0x0	parity_error Parity Error bit. This is used to indicate the occurrence of a parity error in the receiver if the Parity Enable (PEN) bit (LCR[3]) is set.
1	RO	0x0	overrun_error Overrun error bit. This is used to indicate the occurrence of an overrun error. This occurs if a new data character was received before the previous data was read.
0	RO	0x0	data_ready Data Ready bit. This is used to indicate that the receiver contains at least one character in the RBR or the receiver FIFO. 0 = no data ready 1 = data ready

UART_MSR

Address: Operational Base + offset (0x0018)

Modem Status Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7	RO	0x0	data_carrier_detect Data Carrier Detect. This is used to indicate the current state of the modem control line dcd_n.
6	RO	0x0	ring_indicator Ring Indicator. This is used to indicate the current state of the modem control line ri_n.
5	RO	0x0	data_set_ready Data Set Ready. This is used to indicate the current state of the modem control line dsr_n.

Bit	Attr	Reset Value	Description
4	RO	0x0	clear_to_send Clear to Send. This is used to indicate the current state of the modem control line cts_n.
3	RO	0x0	delta_data_carrier_detect Delta Data Carrier Detect. This is used to indicate that the modem control line dcd_n has changed since the last time the MSR was read.
2	RO	0x0	trailing_edge_ring_indicator Trailing Edge of Ring Indicator. Trailing Edge of Ring Indicator. This is used to indicate that a change on the input ri_n (from an active-low to an inactive-high state) has occurred since the last time the MSR was read.
1	RO	0x0	delta_data_set_ready Delta Data Set Ready. This is used to indicate that the modem control line dsr_n has changed since the last time the MSR was read.
0	RO	0x0	delta_clear_to_send Delta Clear to Send. This is used to indicate that the modem control line cts_n has changed since the last time the MSR was read.

UART_SCR

Address: Operational Base + offset (0x001c)

Scratchpad Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	RW	0x00	temp_store_space This register is for programmers to use as a temporary storage space.

UART_SRBR

Address: Operational Base + offset (0x0030)

Shadow Receive Buffer Register

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

Bit	Attr	Reset Value	Description
7:0	RO	0x00	<p>shadow_rbr This is a shadow register for the RBR and has been allocated sixteen 32-bit locations so as to accommodate burst accesses from the master. This register contains the data byte received on the serial input port (sin) in UART mode or the serial infrared input (sir_in) in infrared mode. The data in this register is valid only if the Data Ready (DR) bit in the Line status Register (LSR) is set.</p> <p>If FIFOs are disabled (FCR[0] set to zero), the data in the RBR must be read before the next data arrives, otherwise it is overwritten, resulting in an overrun error.</p> <p>If FIFOs are enabled (FCR[0] set to one), this register accesses the head of the receive FIFO. If the receive FIFO is full and this register is not read before the next data character arrives, then the data already in the FIFO are preserved, but any incoming data is lost. An overrun error also occurs.</p>

UART_STHR

Address: Operational Base + offset (0x006c)

Shadow Transmit Holding Register

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	RO	0x00	<p>shadow_thr This is a shadow register for the THR.</p>

UART_FAR

Address: Operational Base + offset (0x0070)

FIFO Access Register

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	<p>fifo_access_test_en This register is use to enable a FIFO access mode for testing, so that the receive FIFO can be written by the master and the transmit FIFO can be read by the master when FIFOs are implemented and enabled. When FIFOs are not enabled it allows the RBR to be written by the master and the THR to be read by the master. 0 = FIFO access mode disabled 1 = FIFO access mode enabled</p>

UART_TFR

Address: Operational Base + offset (0x0074)

Transmit FIFO Read

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

Bit	Attr	Reset Value	Description
7:0	RO	0x00	<p>trans_fifo_read Transmit FIFO Read.</p> <p>These bits are only valid when FIFO access mode is enabled (FAR[0] is set to one). When FIFOs are implemented and enabled, reading this register gives the data at the top of the transmit FIFO. Each consecutive read pops the transmit FIFO and gives the next data value that is currently at the top of the FIFO.</p>

UART_RFW

Address: Operational Base + offset (0x0078)

Receive FIFO Write

Bit	Attr	Reset Value	Description
31:10	RO	0x0	reserved
9	WO	0x0	<p>receive_fifo_framing_error Receive FIFO Framing Error.</p> <p>These bits are only valid when FIFO access mode is enabled (FAR[0] is set to one).</p>
8	WO	0x0	<p>receive_fifo_parity_error Receive FIFO Parity Error.</p> <p>These bits are only valid when FIFO access mode is enabled (FAR[0] is set to one).</p>
7:0	WO	0x00	<p>receive_fifo_write Receive FIFO Write Data.</p> <p>These bits are only valid when FIFO access mode is enabled (FAR[0] is set to one).</p> <p>When FIFOs are enabled, the data that is written to the RFWD is pushed into the receive FIFO. Each consecutive write pushes the new data to the next write location in the receive FIFO.</p> <p>When FIFOs not enabled, the data that is written to the RFWD is pushed into the RBR.</p>

UART_USR

Address: Operational Base + offset (0x007c)

UART Status Register

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4	RO	0x0	<p>receive_fifo_full Receive FIFO Full.</p> <p>This is used to indicate that the receive FIFO is completely full.</p> <p>0 = Receive FIFO not full 1 = Receive FIFO Full</p> <p>This bit is cleared when the RX FIFO is no longer full.</p>

Bit	Attr	Reset Value	Description
3	RO	0x0	<p>receive_fifo_not_empty Receive FIFO Not Empty. This is used to indicate that the receive FIFO contains one or more entries. 0 = Receive FIFO is empty 1 = Receive FIFO is not empty This bit is cleared when the RX FIFO is empty.</p>
2	RO	0x0	<p>transn_fifo_empty Transmit FIFO Empty. This is used to indicate that the transmit FIFO is completely empty. 0 = Transmit FIFO is not empty 1 = Transmit FIFO is empty This bit is cleared when the TX FIFO is no longer empty</p>
1	RO	0x0	<p>trans_fifo_not_full Transmit FIFO Not Full. This is used to indicate that the transmit FIFO is not full. 0 = Transmit FIFO is full 1 = Transmit FIFO is not full This bit is cleared when the TX FIFO is full.</p>
0	RO	0x0	<p>uart_busy UART Busy. UART Busy. This indicates that a serial transfer is in progress, when cleared indicates that the UART is idle or inactive. 0 = UART is idle or inactive 1 = UART is busy (actively transferring data)</p>

UART_TFL

Address: Operational Base + offset (0x0080)

Transmit FIFO Level

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4:0	RW	0x00	<p>trans_fifo_level Transmit FIFO Level. This indicates the number of data entries in the transmit FIFO.</p>

UART_RFL

Address: Operational Base + offset (0x0084)

Receive FIFO Level

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4:0	RO	0x00	<p>receive_fifo_level Receive FIFO Level. This indicates the number of data entries in the receive FIFO.</p>

UART_SRR

Address: Operational Base + offset (0x0088)

Software Reset Register

Bit	Attr	Reset Value	Description
31:3	RO	0x0	reserved
2	WO	0x0	xmit_fifo_reset XMIT FIFO Reset. This is a shadow register for the XMIT FIFO Reset bit (FCR[2]).
1	WO	0x0	rcvr_fifo_reset RCVR FIFO Reset. This is a shadow register for the RCVR FIFO Reset bit (FCR[1]).
0	WO	0x0	uart_reset UART Reset. This asynchronously resets the UART and synchronously removes the reset assertion. For a two clock implementation both pclk and sclk domains are reset.

UART_SRTS

Address: Operational Base + offset (0x008c)

Shadow Request to Send

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	shadow_req_to_send Shadow Request to Send. This is a shadow register for the RTS bit (MCR[1]), this can be used to remove the burden of having to performing a read-modify-write on the MCR.

UART_SBCR

Address: Operational Base + offset (0x0090)

Shadow Break Control Register

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	shadow_break_ctrl Shadow Break Control Bit. This is a shadow register for the Break bit (LCR[6]), this can be used to remove the burden of having to performing a read modify write on the LCR.

UART_SDMAM

Address: Operational Base + offset (0x0094)

Shadow DMA Mode

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved

Bit	Attr	Reset Value	Description
0	RW	0x0	shadow_dma_mode Shadow DMA Mode. This is a shadow register for the DMA mode bit (FCR[3]).

UART_SFE

Address: Operational Base + offset (0x0098)

Shadow FIFO Enable

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	shadow_fifo_en Shadow FIFO Enable. Shadow FIFO Enable. This is a shadow register for the FIFO enable bit (FCR[0]).

UART_SRT

Address: Operational Base + offset (0x009c)

Shadow RCVR Trigger

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	shadow_rcvr_trigger Shadow RCVR Trigger. This is a shadow register for the RCVR trigger bits (FCR[7:6]).

UART_STET

Address: Operational Base + offset (0x00a0)

Shadow TX Empty Trigger

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	shadow_tx_empty_trigger Shadow TX Empty Trigger. This is a shadow register for the TX empty trigger bits (FCR[5:4]).

UART_HTX

Address: Operational Base + offset (0x00a4)

Halt TX

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	halt_tx_en This register is use to halt transmissions for testing, so that the transmit FIFO can be filled by the master when FIFOs are implemented and enabled. 0 = Halt TX disabled 1 = Halt TX enabled

UART_DMASA

Address: Operational Base + offset (0x00a8)

DMA Software Acknowledge

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	WO	0x0	dma_software_ack This register is use to perform a DMA software acknowledge if a transfer needs to be terminated due to an error condition.

UART_CPR

Address: Operational Base + offset (0x00f4)

Component Parameter Register

UART_CPR is UART0's own unique register

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved
23:16	RO	0x00	FIFO_MODE 0x00 = 0 0x01 = 16 0x02 = 32 to 0x80 = 2048 0x81- 0xff = reserved
15:14	RO	0x0	reserved
13	RO	0x0	DMA_EXTRA 0 = FALSE 1 = TRUE
12	RO	0x0	UART_ADD_ENCODED_PARAMS 0 = FALSE 1 = TRUE
11	RO	0x0	SHADOW 0 = FALSE 1 = TRUE
10	RO	0x0	FIFO_STAT 0 = FALSE 1 = TRUE
9	RO	0x0	FIFO_ACCESS 0 = FALSE 1 = TRUE
8	RO	0x0	NEW_FEAT 0 = FALSE 1 = TRUE
7	RO	0x0	SIR_LP_MODE 0 = FALSE 1 = TRUE
6	RO	0x0	SIR_MODE 0 = FALSE 1 = TRUE

Bit	Attr	Reset Value	Description
5	RO	0x0	THRE_MODE 0 = FALSE 1 = TRUE
4	RO	0x0	AFCE_MODE 0 = FALSE 1 = TRUE
3:2	RO	0x0	reserved
1:0	RO	0x0	APB_DATA_WIDTH 00 = 8 bits 01 = 16 bits 10 = 32 bits 11 = reserved

UART_UCV

Address: Operational Base + offset (0x00f8)

UART Component Version

Bit	Attr	Reset Value	Description
31:0	RO	0x0330372a	ver ASCII value for each number in the version

UART_CTR

Address: Operational Base + offset (0x00fc)

Component Type Register

Bit	Attr	Reset Value	Description
31:0	RO	0x44570110	peripheral_id This register contains the peripherals identification code.

6.5 Interface Description

Table 6-1 UART Interface Description

Module Pin	Direction	Pad Name	IOMUX Setting
UART0 Interface			
uart0_sin	I	IO_URT0sin_GPIO0c1	GRF_GPIO0C_IOMUX[2]=1'b1
uart0_sout	O	IO_URT0sout_GPIO0c0	GRF_GPIO0C_IOMUX[0]=1'b1
uart0_cts_n	I	IO_URT0ctsn_GPIO0c3	GRF_GPIO0C_IOMUX[6]=1'b1
uart0_rts_n	O	IO_URT0rtsn_GPIO0c2	GRF_GPIO0C_IOMUX[4]=1'b1
UART1 Interface			
uart1_sin	I	IO_UART1sin_GPIO2c6	GRF_GPIO0C_IOMUX[12]=1'b1
uart1_sout	O	IO_UART1sout_TESTCLKou_t1_GPIO2c7	GRF_GPIO2C_IOMUX[15:14]=2'b01
UART2 Interface			
uart2_sin	I	IO_MMCD0d0_UART2sin_GP IO1c2	GRF_GPIO1C_IOMUX[5:4]=2'b10
uart2_sout	O	IO_MMCD0d1_UART2sout_G	GRF_GPIO1C_IOMUX[7:6]=2'b1

Module Pin	Direction	Pad Name	IOMUX Setting
		PIO1c3	0

6.6 Application Notes

6.7 None FIFO Mode Transfer Flow

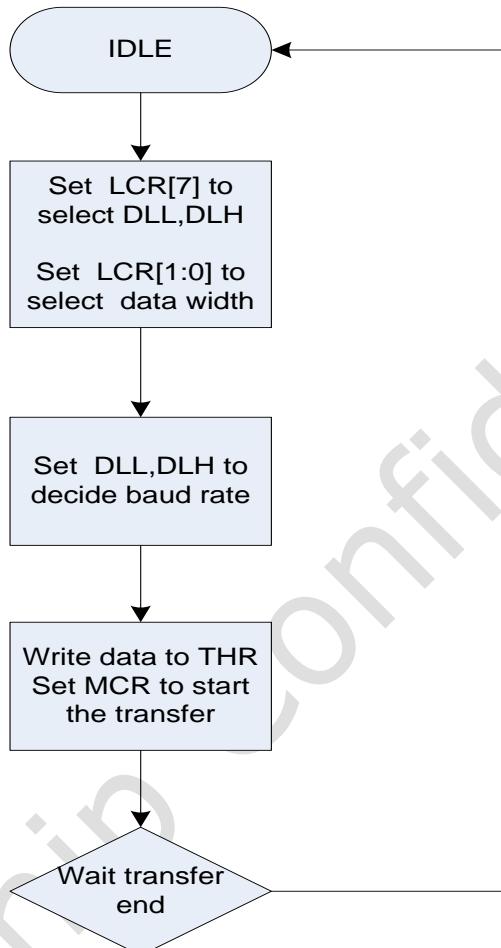


Fig. 6-8 UART none fifo mode

6.8 FIFO Mode Transfer Flow

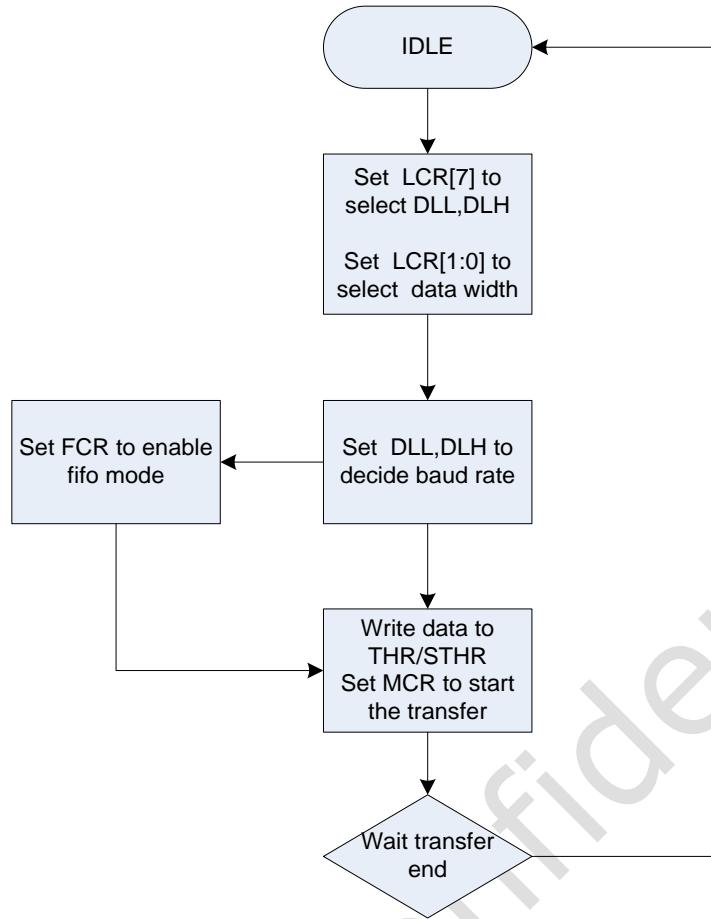


Fig. 6-9 UART fifo mode

The UART is an APB slave performing:

Serial-to-parallel conversion on data received from a peripheral device.

Parallel-to-serial conversion on data transmitted to the peripheral device.

The CPU reads and writes data and control/status information through the APB interface. The transmitting and receiving paths are buffered with internal FIFO memories enabling up to 64-bytes(UART1/UART2 FIFO is 32-bytes) to be stored independently in both transmit and receive modes. A baud rate generator can generate a common transmit and receive internal clock input. The baud rates will depend on the internal clock frequency. The UART will also provide transmit, receive and exception interrupts to system. A DMA interface is implemented for improving the system performance.

6.9 Baud Rate Calculation

UART clock generation

The following figure shows the UART clock generation.

UART source clocks can be selected from ARM_PLL/DDR_PLL/GEBERAL_PLL/480M outputs.

UART clocks can be generated by 1 to 64 division of its source clock, or can be fractionally divided again, or be provided by XIN24M.

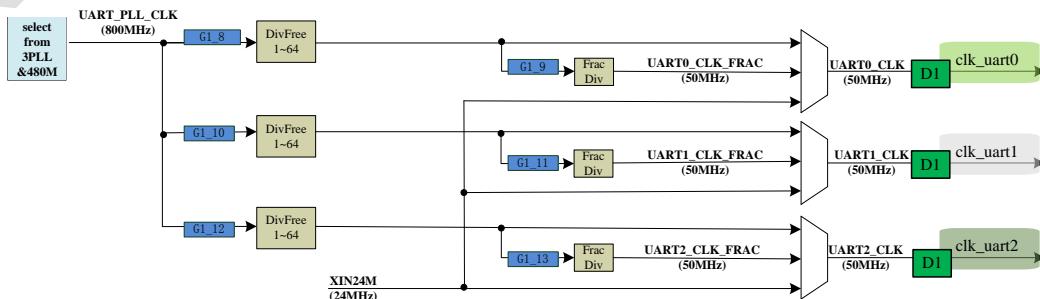


Fig. 6-10 UART clock generation

UART baud rate configuration

The following table provides some reference configuration for different UART baud rates.

Table 6-2 UART baud rate configuration

Baud Rate	Reference Configuration
115.2 Kbps	Configure GENERAL PLL to get 648MHz clock output; Divide 648MHz clock by 1152/50625 to get 14.7456MHz clock; Configure UART_DLL to 8.
460.8 Kbps	Configure GENERAL PLL to get 648MHz clock output; Divide 648MHz clock by 1152/50625 to get 14.7456MHz clock; Configure UART_DLL to 2.
921.6 Kbps	Configure GENERAL PLL to get 648MHz clock output; Divide 648MHz clock by 1152/50625 to get 14.7456MHz clock; Configure UART_DLL to 1.
1.5 Mbps	Choose GENERAL PLL to get 384MHz clock output; Divide 384MHz clock by 16 to get 24MHz clock; Configure UART_DLL to 1
3 Mbps	Choose GENERAL PLL to get 384MHz clock output; Divide 384MHz clock by 8 to get 48MHz clock; Configure UART_DLL to 1
4 Mbps	Configure GENERAL PLL to get 384MHz clock output; Divide 384MHz clock by 6 to get 64MHz clock; Configure UART_DLL to 1

Chapter 7 GPIO

7.1 Overview

GPIO is a programmable General Purpose Programming I/O peripheral. This component is an APB slave device. GPIO controls the output data and direction of external I/O pads. It also can read back the data on external pads using memory-mapped registers.

GPIO supports the following features:

- 32 bits APB bus width
- 32 independently configurable signals
- Separate data registers and data direction registers for each signal
- Software control for each signal, or for each bit of each signal
- Configurable interrupt mode

7.2 Block Diagram

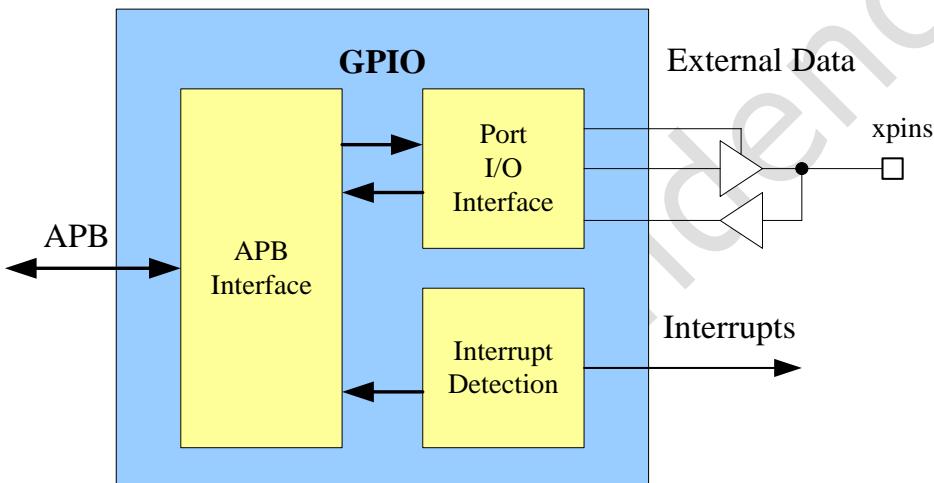


Fig. 7-1 GPIO block diagram

Block descriptions:

APB Interface

The APB Interface implements the APB slave operation. Its data bus width is 32 bits.

Port I/O Interface

External data Interface to or from I/O pads.

Interrupt Detection

Interrupt interface to or from interrupt controller.

7.3 Function Description

7.3.1 Operation

Control Mode (software)

Under software control, the data and direction control for the signal are sourced from the data register (GPIO_SWPORTA_DR) and direction control register (GPIO_SWPORTA_DDR).

The direction of the external I/O pad is controlled by a write to the Porta data direction register (GPIO_SWPORTA_DDR). The data written to this memory-mapped register gets mapped onto an output signal, GPIO_PORTA_DDR, of the GPIO peripheral. This output signal controls the direction of an external I/O pad.

The data written to the Porta data register (GPIO_SWPORTA_DR) drives the output buffer of the I/O pad. External data are input on the external data signal, GPIO_EXT_PORTA. Reading the external signal register (GPIO_EXT_PORTA) shows the value on the signal, regardless of the direction. This register is read-only, meaning that it cannot be written from the APB software interface.

Reading External Signals

The data on the GPIO_EXT_PORTA external signal can always be read. The data on the external GPIO signal is read by an APB read of the memory-mapped register, GPIO_EXT_PORTA.

An APB read to the GPIO_EXT_PORTA register yields a value equal to that which is on the GPIO_EXT_PORTA signal.

Interrupts

Port A can be programmed to accept external signals as interrupt sources on any of the bits of the signal. The type of interrupt is programmable with one of the following settings:

- Active-high and level
- Active-low and level
- Rising edge
- Falling edge

The interrupts can be masked by programming the GPIO_INTMASK register. The interrupt status can be read before masking (called raw status) and after masking.

The interrupts are combined into a single interrupt output signal, which has the same polarity as the individual interrupts. In order to mask the combined interrupt, all individual interrupts have to be masked. The single combined interrupt does not have its own mask bit.

Whenever Port A is configured for interrupts, the data direction must be set to Input. If the data direction register is reprogrammed to Output, then any pending interrupts are not lost. However, no new interrupts are generated.

For edge-detected interrupts, the ISR can clear the interrupt by writing a 1 to the GPIO_PORTA_EOI register for the corresponding bit to disable the interrupt. This write also clears the interrupt status and raw status registers. Writing to the GPIO_PORTA_EOI register has no effect on level-sensitive interrupts. If level-sensitive interrupts cause the processor to interrupt, then the ISR can poll the GPIO_INT_RAWSTATUS register until the interrupt source disappears, or it can write to the GPIO_INTMASK register to mask the interrupt before exiting the ISR. If the ISR exits without masking or disabling the interrupt prior to exiting, then the level-sensitive interrupt repeatedly requests an interrupt until the interrupt is cleared at the source.

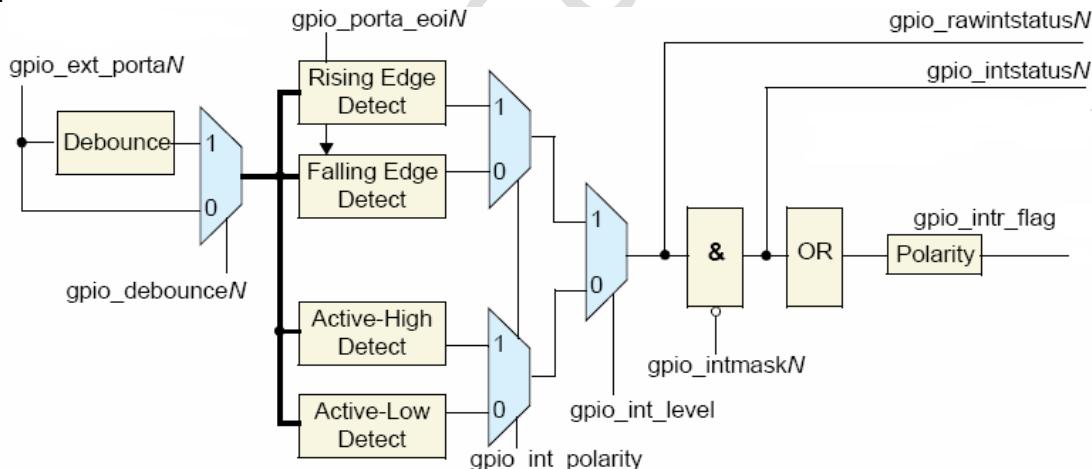


Fig. 7-2 GPIO Interrupt RTL Block Diagram

Debounce operation

Port A has been configured to include the debounce capability interrupt feature. The external signal can be debounced to remove any spurious glitches that are less than one period of the external debouncing clock.

When input interrupt signals are debounced using a debounce clock (pclk), the signals must be active for a minimum of two cycles of the debounce clock to guarantee that they are registered. Any input pulse widths less than a debounce clock period are bounced. A pulse width between one and two debounce clock widths may or may not propagate, depending on its phase relationship to the debounce clock. If the input pulse spans two rising edges of the debounce clock, it is registered. If it spans only one rising edge, it is not registered.

Synchronization of Interrupt Signals to the System Clock

Interrupt signals are internally synchronized to pclk. Synchronization topclk must occur for edge-detect signals. With level-sensitive interrupts, synchronization is optional and under

software control (GPIO_LS_SYNC).

7.3.2 Programming

Programming Considerations

- Reading from an unused location or unused bits in a particular register always returns zeros. There is no error mechanism in the APB.
- Programming the GPIO registers for interrupt capability, edge-sensitive or level-sensitive interrupts, and interrupt polarity should be completed prior to enabling the interrupts on Port A in order to prevent spurious glitches on the interrupt lines to the interrupt controller.
- Writing to the interrupt clear register clears an edge-detected interrupt and has no effect on a level-sensitive interrupt.

9 GPIOs' hierarchy in the chip

GPIO0, GPIO1, GPIO2 are in PD_PERI subsystem.

7.4 Register Description

This section describes the control/status registers of the design. Software should read and write these registers using 32-bits accesses. There are 3 GPIOs (GPIO0 ~ GPIO2), and each of them has same register group. Therefore, 3 GPIOs' register groups have 3 different base addresses.

7.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
GPIO_SWPORTA_DR	0x0000	W	0x00000000	Port A data register
GPIO_SWPORTA_DDR	0x0004	W	0x00000000	Port A data direction register
GPIO_INTEN	0x0030	W	0x00000000	Interrupt enable register
GPIO_INTMASK	0x0034	W	0x00000000	Interrupt mask register
GPIO_INTPTYPE_LEVEL	0x0038	W	0x00000000	Interrupt level register
GPIO_INT_POLARITY	0x003c	W	0x00000000	Interrupt polarity register
GPIO_INT_STATUS	0x0040	W	0x00000000	Interrupt status of port A
GPIO_INT_RAWSTATUS	0x0044	W	0x00000000	Raw Interrupt status of port A
GPIO_DEBOUNCE	0x0048	W	0x00000000	Debounce enable register
GPIO_PORTA_EOI	0x004c	W	0x00000000	Port A clear interrupt register
GPIO_EXT_PORTA	0x0050	W	0x00000000	Port A external port register
GPIO_LS_SYNC	0x0060	W	0x00000000	Level_sensitive synchronization enable register

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

7.4.2 Detail Register Description

GPIO_SWPORTA_DR

Address: Operational Base + offset (0x0000)

Port A data register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	gpio_swporta_dr Values written to this register are output on the I/O signals for Port A if the corresponding data direction bits for Port A are set to Output mode. The value read back is equal to the last value written to this register.

GPIO_SWPORTA_DDR

Address: Operational Base + offset (0x0004)

Port A data direction register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	gpio_swporta_ddr Values written to this register independently control the direction of the corresponding data bit in Port A. 0: Input (default) 1: Output

GPIO_INTEN

Address: Operational Base + offset (0x0030)

Interrupt enable register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	gpio_int_en Allows each bit of Port A to be configured for interrupts. Whenever a 1 is written to a bit of this register, it configures the corresponding bit on Port A to become an interrupt; otherwise, Port A operates as a normal GPIO signal. Interrupts are disabled on the corresponding bits of Port A if the corresponding data direction register is set to Output. 0: Configure Port A bit as normal GPIO signal (default) 1: Configure Port A bit as interrupt

GPIO_INTMASK

Address: Operational Base + offset (0x0034)

Interrupt mask register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	gpio_int_mask Controls whether an interrupt on Port A can create an interrupt for the interrupt controller by not masking it. Whenever a 1 is written to a bit in this register, it masks the interrupt generation capability for this signal; otherwise interrupts are allowed through. 0: Interrupt bits are unmasked (default) 1: Mask interrupt

GPIO_INTPTYPE_LEVEL

Address: Operational Base + offset (0x0038)

Interrupt level register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	gpio_inttype_level Controls the type of interrupt that can occur on Port A. 0: Level-sensitive (default) 1: Edge-sensitive

GPIO_INT_POLARITY

Address: Operational Base + offset (0x003c)

Interrupt polarity register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	gpio_int_polarity Controls the polarity of edge or level sensitivity that can occur on input of Port A. 0: Active-low (default) 1: Active-high

GPIO_INT_STATUS

Address: Operational Base + offset (0x0040)

Interrupt status of port A

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	gpio_int_status Interrupt status of Port A

GPIO_INT_RAWSTATUS

Address: Operational Base + offset (0x0044)

Raw Interrupt status of port A

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	gpio_int_rawstatus Raw interrupt of status of Port A (premasking bits)

GPIO_DEBOUNCE

Address: Operational Base + offset (0x0048)

Debounce enable register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	gpio_debounce Controls whether an external signal that is the source of an interrupt needs to be debounced to remove any spurious glitches. Writing a 1 to a bit in this register enables the debouncing circuitry. A signal must be valid for two periods of an external clock before it is internally processed. 0: No debounce (default) 1: Enable debounce

GPIO_PORTA_EOI

Address: Operational Base + offset (0x004c)

Port A clear interrupt register

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:0	WO	0x00000000	gpio_porta_eoi Controls the clearing of edge type interrupts from Port A. When a 1 is written into a corresponding bit of this register, the interrupt is cleared. All interrupts are cleared when Port A is not configured for interrupts. 0: No interrupt clear (default) 1: Clear interrupt

GPIO_EXT_PORTA

Address: Operational Base + offset (0x0050)

Port A external port register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	gpio_ext_porta When Port A is configured as Input, then reading this location reads the values on the signal. When the data direction of Port A is set as Output, reading this location reads the data register for Port A.

GPIO_LS_SYNC

Address: Operational Base + offset (0x0060)

Level_sensitive synchronization enable register

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	gpio_ls_sync Writing a 1 to this register results in all level-sensitive interrupts being synchronized to pclk_intr. 0: No synchronization to pclk_intr (default) 1: Synchronize to pclk_intr

7.5 Interface Description

Table 7-1 GPIO interface description

Module Pin	Dir	Pad Name	IOMUX Setting
GPIO0 Interface			
gpio0_porta[3:0]	I/O	GPIO0_A[3:0]	GRF_GPIO0A_IOMUX[7:0]=8'h0
gpio0_porta[14:8]	I/O	GPIO0_B[6:0]	GRF_GPIO0B_IOMUX[13:0]=14'h0
gpio0_porta[20:16]	I/O	GPIO0_C[4:0]	GRF_GPIO0C_IOMUX[9:0]=10'h0
gpio0_porta[28:26]	I/O	GPIO0_D[4:2]	GRF_GPIO0D_IOMUX[9:4]=6'h0
GPIO1 Interface			
gpio1_porta[5:0]	I/O	GPIO1_A[5:0]	GRF_GPIO1A_IOMUX[11:0]=12'h0
gpio1_porta[11:8]	I/O	GPIO1_B[3:0]	GRF_GPIO1B_IOMUX[7:0]=8'h0
gpio1_porta[15]		GPIO1_B[7]	GRF_GPIO1B_IOMUX[15:14]=2'h0
gpio1_porta[21:16]	I/O	GPIO1_C[5:0]	GRF_GPIO1C_IOMUX[11:0]=12'h0
gpio1_porta[31:24]	I/O	GPIO1_D[7:0]	GRF_GPIO1D_IOMUX[15:0]=16'h0
GPIO2 Interface			

Module Pin	Dir	Pad Name	IOMUX Setting
gpio2_porta[4:0]	I/O	GPIO2_A[4:0]	GRF_GPIO2A_IOMUX[9:0]=10'h0
gpio2_porta[7:6]		GPIO2_A[7:6]	GRF_GPIO2A_IOMUX[15:12]=4'h0
gpio2_porta[15:8]	I/O	GPIO2_B[7:0]	GRF_GPIO2B_IOMUX[15:0]=16'h0
gpio2_porta[23:16]	I/O	GPIO2_C[7:0]	GRF_GPIO2C_IOMUX[15:0]=16'h0
gpio2_porta[25]	I/O	GPIO2_D[1]	GRF_GPIO2D_IOMUX[3:2]=2'h0
gpio2_porta[30:28]		GPIO2_D[6:4]	GRF_GPIO2D_IOMUX[13:8]=6'h0

7.6 Application Notes

Steps to set GPIO's direction

- Write GPIO_SWPORT_DDR[x] as 1 to set this gpio as output direction and Write GPIO_SWPORT_DDR[x] as 0 to set this gpio as input direction.
- Default GPIO's direction is input direction.

Steps to set GPIO's level

- Write GPIO_SWPORT_DDR[x] as 1 to set this gpio as output direction.
- Write GPIO_SWPORT_DR[x] as v to set this GPIO's value.

Steps to get GPIO's level

- Write GPIO_SWPORT_DDR[x] as 0 to set this gpio as input direction.
- Read from GPIO_EXT_PORT[x] to get GPIO's value

Steps to set GPIO as interrupt source

- Write GPIO_SWPORT_DDR[x] as 0 to set this gpio as input direction.
- Write GPIO_INTPTYPE_LEVEL[x] as v1 and write GPIO_INT_POLARITY[x] as v2 to set interrupt type
- Write GPIO_INTEN[x] as 1 to enable GPIO's interrupt

Note: Please switch iomux to GPIO mode first!

Chapter 8 I2C Interface

8.1 Overview

The Inter-Integrated Circuit (I2C) is a two wired (SCL and SDA), bi-directional serial bus that provides an efficient and simple method of information exchange between devices. This I2C bus controller supports master mode acting as a bridge between AMBA protocol and generic I2C bus system.

I2C Controller supports the following features:

- Item Compatible with I2C-bus
- AMBA APB slave interface
- Supports master mode of I2C bus
- Software programmable clock frequency and transfer rate up to 400Kbit/sec
- Supports 7 bits and 10 bits addressing modes
- Interrupt or polling driven multiple bytes data transfer
- Clock stretching and wait state generation

8.2 Block Diagram

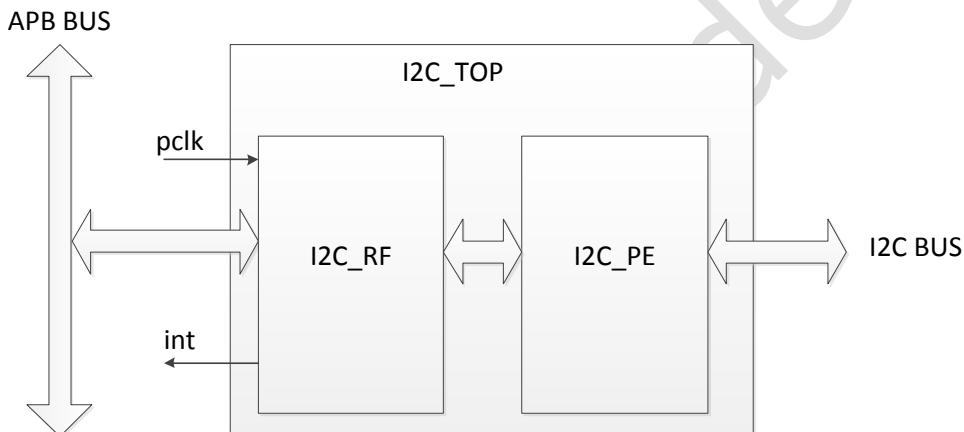


Fig. 8-1 I2C architecture

8.2.1 I2C_RF

I2C_RF module is used to control the I2C controller operation by the host with APB interface. It implements the register set and the interrupt functionality. The CSR component operates synchronously with the pclk clock.

8.2.2 I2C_PE

I2C_PE module implements the I2C master operation for transmit data to and receive data from other I2C devices. The I2C master controller operates synchronously with the pclk.

8.2.3 I2C_TOP

I2C_TOP module is the top module of the I2C controller.

8.3 Function Description

This chapter provides a description about the functions and behavior under various conditions. The I2C controller supports only Master function. It supports the 7-bits/10-bits addressing mode and support general call address. The maximum clock frequency and transfer rate can be up to 400Kbit/sec.

The operations of I2C controller is divided to 2 parts and described separately: initialization and master mode programming.

8.3.1 Initialization

The I2C controller is based on AMBA APB bus architecture and usually is part of a SOC. So before I2C operates, some system setting and configuration must be conformed, which

includes:

- I2C interrupt connection type: CPU interrupt scheme should be considered. If the I2C interrupt is connected to extra Interrupt Controller module, we need decide the INTC vector.
- I2C Clock Rate: The I2C controller uses the APB clock as the working clock so the APB clock will determine the I2C bus clock. The correct register setting is subject to the system requirement.

8.3.2 Master Mode Programming

- SCL Clock

When the I2C controller is programmed in Master mode, the SCL frequency is determined by I2C_CLKDIV register. The SCL frequency is calculated by the following formula:

$$\text{SCL Divisor} = 8 * (\text{CLKDIVL} + 1 + \text{CLKDIVH} + 1)$$

$$\text{SCL} = \text{PCLK} / \text{SCLK Divisor}$$

- Data Receiver Register Access

When the I2C controller received MRXCNT bytes data, CPU can get the data through register RXDATA0 ~ RXDATA7. The controller can receive up to 32 bytes' data in one transaction.

When MRXCNT register is written, the I2C controller will start to drive SCL to receive data.

- Transmit Transmitter Register

Data to transmit are written to TXDATA0~7 by CPU. The controller can transmit up to 32 bytes' data in one transaction. The lower byte will be transmitted first.

When MTXCNT register is written, the I2C controller will start to transmit data.

- Start Command

Write 1 to I2C_CON[3], the controller will send I2C start command.

- Stop Command

Write 1 to I2C_CON[4], the controller will send I2C stop command

- I2C Operation mode

There are four i2c operation modes.

- When I2C_CON[2:1] is 2'b00, the controller transmit all valid data in TXDATA0~TXDATA7 byte by byte. The controller will transmit lower byte first.
- When I2C_CON[2:1] is 2'b01, the controller will transmit device address in MRXADDR first (Write/Read bit = 0) and then transmit device register address in MRXRADDR. After that, the controller will assert restart signal and resend MRXADDR (Write/Read bit = 1). At last, the controller enter receive mode.
- When I2C_CON[2:1] is 2'b10, the controller is in receive mode, it will trigger clock to read MRXCNT byte data.
- When I2C_CON[2:1] is 2'b11, the controller will transmit device address in MRXADDR first (Write/Read bit = 1) and then transmit device register address in MRXRADDR . After that, the controller will assert restart signal and resend MRXADDR (Write/Read bit = 1). At last, the controller enter receive mode.

- Read/Write Command

- When I2C_OPMODE(I2C_CON[2:1]) is 2'b01 or 2'b11, the Read/Write command bit is decided by controller itself.
- In RX only mode (I2C_CON[2:1] is 2'b10), the Read/Write command bit is decided by MRXADDR[0].
- In TX only mode (I2C_CON[[2:1] is 2'b00), the Read/Write command bit is decided by TXDATA[0].

- Master Interrupt Condition
There are 7 interrupt bits in I2C_ISR register related to master mode.
 - Byte transmitted finish interrupt (Bit 0): The bit is asserted when Master completed transmitting a byte.
 - Byte received finish interrupt (Bit 1): The bit is asserted when Master completed receiving a byte.
 - MTXCNT bytes data transmitted finish interrupt (Bit 2): The bit is asserted when Master completed transmitting MTXCNT bytes.
 - MRXCNT bytes data received finish interrupt (Bit 3): The bit is asserted when Master completed receiving MRXCNT bytes.
 - Start interrupt (Bit 4): The bit is asserted when Master finished asserting start command to I2C bus.
 - Stop interrupt (Bit 5): The bit is asserted when Master finished asserting stop command to I2C bus.
 - NAK received interrupt (Bit 6): The bit is asserted when Master received a NAK handshake.
- Last byte acknowledge control
 - If I2C_CON[5] is 1, the I2C controller will transmit NAK handshake to slave when the last byte received in RX only mode.
 - If I2C_CON[5] is 0, the I2C controller will transmit ACK handshake to slave when the last byte received in RX only mode.
- How to handle NAK handshake received
 - If I2C_CON[6] is 1, the I2C controller will stop all transactions when NAK handshake received. And the software should take responsibility to handle the problem.
 - If I2C_CON[6] is 0, the I2C controller will ignore all NAK handshake received.
- I2C controller data transfer waveform
 - Bit transferring
 - Data Validity**
The SDA line must be stable during the high period of SCL, and the data on SDA line can only be changed when SCL is in low state.

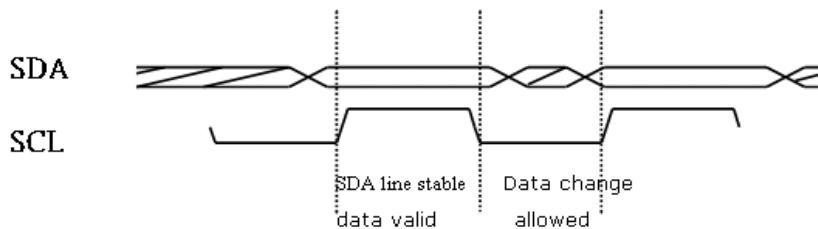


Fig. 8-2 I2C DATA Validity

- ◆ START and STOP conditions

START condition occurs when SDA goes low while SCL is in high period. STOP condition is generated when SDA line goes high while SCL is in high state.

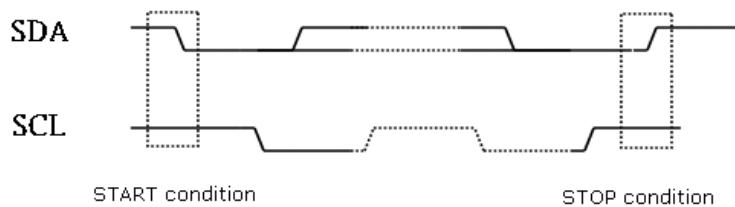


Fig. 8-3 I2C Start and stop conditions

- ◆ Data transfer
 - Acknowledge

After a byte of data transferring (clocks labeled as 1~8), in 9th clock the receiver must assert an ACK signal on SDA line, if the receiver pulls SDA line to low, it means "ACK", on the contrary, it's "NOT ACK".

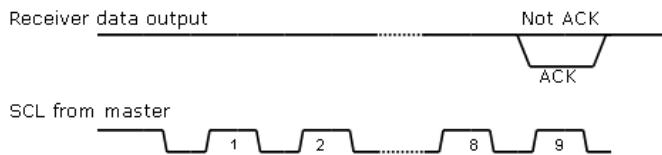


Fig. 8-4 I2C Acknowledge

➤ **Byte transfer**

The master own I2C bus might initiate multi byte to transfer to a slave. The transfer starts from a "START" command and ends in a "STOP" command. After every byte transfer, the receiver must reply an ACK to transmitter.

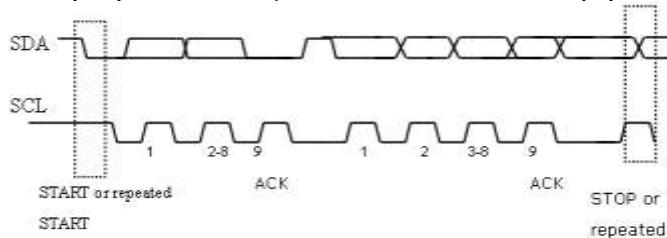


Fig. 8-5 I2C byte transfer

8.4 Register Description

8.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
RKI2C_CON	0x0000	W	0x00000000	control register
RKI2C_CLKDIV	0x0004	W	0x00000001	clock divider register
RKI2C_MRXADDR	0x0008	W	0x00000000	the slave address accessed for master rx mode
RKI2C_MRXRADDR	0x000c	W	0x00000000	the slave register address accessed for master rx mode
RKI2C_MTXCNT	0x0010	W	0x00000000	master transmit count
RKI2C_MRXCNT	0x0014	W	0x00000000	master rx count
RKI2C_IEN	0x0018	W	0x00000000	interrupt enable register
RKI2C_IPD	0x001c	W	0x00000000	interrupt pending register
RKI2C_FCNT	0x0020	W	0x00000000	finished count
RKI2C_TXDATA0	0x0100	W	0x00000000	I2C tx data register 0
RKI2C_TXDATA1	0x0104	W	0x00000000	I2C tx data register 1
RKI2C_TXDATA2	0x0108	W	0x00000000	I2C tx data register 2
RKI2C_TXDATA3	0x010c	W	0x00000000	I2C tx data register 3
RKI2C_TXDATA4	0x0110	W	0x00000000	I2C tx data register 4
RKI2C_TXDATA5	0x0114	W	0x00000000	I2C tx data register 5
RKI2C_TXDATA6	0x0118	W	0x00000000	I2C tx data register 6
RKI2C_TXDATA7	0x011c	W	0x00000000	I2C tx data register 7
RKI2C_RXDATA0	0x0200	W	0x00000000	I2C rx data register 0
RKI2C_RXDATA1	0x0204	W	0x00000000	I2C rx data register 1
RKI2C_RXDATA2	0x0208	W	0x00000000	I2C rx data register 2
RKI2C_RXDATA3	0x020c	W	0x00000000	I2C rx data register 3

Name	Offset	Size	Reset Value	Description
RKI2C_RXDATA4	0x0210	W	0x00000000	I2C rx data register 4
RKI2C_RXDATA5	0x0214	W	0x00000000	I2C rx data register 5
RKI2C_RXDATA6	0x0218	W	0x00000000	I2C rx data register 6
RKI2C_RXDATA7	0x021c	W	0x00000000	I2C rx data register 7

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

8.4.2 Detail Register Description

RKI2C_CON

Address: Operational Base + offset (0x0000)
control register

Bit	Attr	Reset Value	Description
31:7	RO	0x0	reserved
6	RW	0x0	act2nak operation when NAK handshake is received 1'b0: ignored 1'b1: stop transaction
5	RW	0x0	ack last byte acknowledge control in master receive mode 1'b0: ACK 1'b1: NAK
4	RW	0x0	stop stop enable stop enable, when this bit is written to 1, I2C will generate stop signal.
3	RW	0x0	start start enable start enable, when this bit is written to 1, I2C will generate start signal.
2:1	RW	0x0	i2c_mode i2c mode select 2'b00: transmit only 2'b01: transmit address (device + register address) --> restart --> transmit address -> receive only 2'b10: receive only 2'b11: transmit address (device + register address, write/read bit is 1) --> restart --> transmit address (device address) --> receive data
0	RW	0x0	i2c_en i2c module enable 1'b0: not enable 1'b1: enable

RKI2C_CLKDIV

Address: Operational Base + offset (0x0004)
clock divider register

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	CLKDIVH scl high level clock count $T(SCL_HIGH) = T(PCLK) * (CLKDIVH + 1) * 8$
15:0	RW	0x0001	CLKDIVL scl low level clock count $T(SCL_LOW) = T(PCLK) * (CLKDIVL + 1) * 8$

RKI2C_MRXADDR

Address: Operational Base + offset (0x0008)
the slave address accessed for master rx mode

Bit	Attr	Reset Value	Description
31:27	RO	0x0	reserved
26	RW	0x0	addhvld address high byte valid 1'b0:invalid 1'b1:valid
25	RW	0x0	addmvld address middle byte valid 1'b0:invalid 1'b1:valid
24	RW	0x0	addlvld address low byte valid 1'b0:invalid 1'b1:valid
23:0	RW	0x0000000	saddr master address register the lowest bit indicate write or read 24 bits address register

RKI2C_MRXRADDR

Address: Operational Base + offset (0x000c)
the slave register address accessed for master rx mode

Bit	Attr	Reset Value	Description
31:27	RO	0x0	reserved
26	RW	0x0	sraddhvld address high byte valid 1'b0:invalid 1'b1:valid
25	RW	0x0	sraddmvld address middle byte valid 1'b0:invalid 1'b1:valid

Bit	Attr	Reset Value	Description
24	RW	0x0	sraaddrvld address low byte valid 1'b0:invalid 1'b1:valid
23:0	RW	0x000000	sraaddr slave register address accessed 24 bits register address

RKI2C_MTXCNT

Address: Operational Base + offset (0x0010)

master transmit count

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5:0	RW	0x00	mtxcnt master transmit count 6 bits counter

RKI2C_MRXCNT

Address: Operational Base + offset (0x0014)

master rx count

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5:0	RW	0x00	mrxcnt master rx count 6 bits counter

RKI2C_IEN

Address: Operational Base + offset (0x0018)

interrupt enable register

Bit	Attr	Reset Value	Description
31:7	RO	0x0	reserved
6	RW	0x0	nakrcvien NAK handshake received interrupt enable 1'b0:disable 1'b1:enable
5	RW	0x0	stopien stop operation finished interrupt enable 1'b0:disable 1'b1:enable
4	RW	0x0	startien start operation finished interrupt enable 1'b0:disable 1'b1:enable

Bit	Attr	Reset Value	Description
3	RW	0x0	mbrfien MRXCNT data received finished interrupt enable 1'b0:disable 1'b1:enable
2	RW	0x0	mbtfien MTXCNT data transfer finished interrupt enable 1'b0:disable 1'b1:enable
1	RW	0x0	brfien byte rx finished interrupt enable 1'b0:disable 1'b1:enable
0	RW	0x0	btfien byte tx finished interrupt enable 1'b0:disable 1'b1:enable

RKI2C_IPD

Address: Operational Base + offset (0x001c)
interrupt pending register

Bit	Attr	Reset Value	Description
31:7	RO	0x0	reserved
6	W1C	0x0	nakrcvipd NAK handshake received interrupt pending bit 1'b0:no interrupt available 1'b1:NAK handshake received interrupt appear, write 1 to clear
5	W1C	0x0	stopipd stop operation finished interrupt pending bit 1'b0:no interrupt available 1'b1:stop operation finished interrupt appear, write 1 to clear
4	W1C	0x0	startipd start operation finished interrupt pending bit 1'b0:no interrupt available 1'b1:start operation finished interrupt appear, write 1 to clear
3	W1C	0x0	mbrfidp MRXCNT data received finished interrupt pending bit 1'b0:no interrupt available 1'b1:MRXCNT data received finished interrupt appear, write 1 to clear
2	W1C	0x0	mbtfidp MTXCNT data transfer finished interrupt pending bit 1'b0:no interrupt available 1'b1:MTXCNT data transfer finished interrupt appear, write 1 to clear

Bit	Attr	Reset Value	Description
1	W1C	0x0	brfidp byte rx finished interrupt pending bit 1'b0: no interrupt available 1'b1: byte rx finished interrupt appear, write 1 to clear
0	W1C	0x0	btfidp byte tx finished interrupt pending bit 1'b0: no interrupt available 1'b1: byte tx finished interrupt appear, write 1 to clear

RKI2C_FCNT

Address: Operational Base + offset (0x0020)

finished count

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5:0	RO	0x00	fcnt finished count the count of data which has been transmitted or received for debug purpose

RKI2C_TXDATA0

Address: Operational Base + offset (0x0100)

I2C tx data register 0

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata0 data0 to be transmitted 32 bits data

RKI2C_TXDATA1

Address: Operational Base + offset (0x0104)

I2C tx data register 1

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata1 data1 to be transmitted 32 bits data

RKI2C_TXDATA2

Address: Operational Base + offset (0x0108)

I2C tx data register 2

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata2 data2 to be transmitted 32 bits data

RKI2C_TXDATA3

Address: Operational Base + offset (0x010c)

I2C tx data register 3

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata3 data3 to be transmitted 32 bits data

RKI2C_TXDATA4

Address: Operational Base + offset (0x0110)

I2C tx data register 4

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata4 data4 to be transmitted 32 bits data

RKI2C_TXDATA5

Address: Operational Base + offset (0x0114)

I2C tx data register 5

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata5 data5 to be transmitted 32 bits data

RKI2C_TXDATA6

Address: Operational Base + offset (0x0118)

I2C tx data register 6

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata6 data6 to be transmitted 32 bits data

RKI2C_TXDATA7

Address: Operational Base + offset (0x011c)

I2C tx data register 7

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	txdata7 data7 to be transmitted 32 bits data

RKI2C_RXDATA0

Address: Operational Base + offset (0x0200)

I2C rx data register 0

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata0 data0 received 32 bits data

RKI2C_RXDATA1

Address: Operational Base + offset (0x0204)

I2C rx data register 1

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata1 data1 received 32 bits data

RKI2C_RXDATA2

Address: Operational Base + offset (0x0208)

I2C rx data register 2

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata2 data2 received 32 bits data

RKI2C_RXDATA3

Address: Operational Base + offset (0x020c)

I2C rx data register 3

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata3 data3 received 32 bits data

RKI2C_RXDATA4

Address: Operational Base + offset (0x0210)

I2C rx data register 4

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata4 data4 received 32 bits data

RKI2C_RXDATA5

Address: Operational Base + offset (0x0214)

I2C rx data register 5

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata5 data5 received 32 bits data

RKI2C_RXDATA6

Address: Operational Base + offset (0x0218)

I2C rx data register 6

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata6 data6 received 32 bits data

RKI2C_RXDATA7

Address: Operational Base + offset (0x021c)

I2C rx data register 7

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	rxdata7 data7 received 32 bits data

8.5 Interface Description

Table 8-1 I2C Interface Description

Module pin	Direction	Pad name	IOMUX
I2C0 Interface			
i2c0_sda	I/O	IO_I2C0pmusda_PWM2_GPIO0a1	GRF_GPIO0A_IOMUX[3:2]=2'b01
i2c0_scl	I/O	IO_I2C0pmuscl_PWM1_GPIO0a0	GRF_GPIO0A_IOMUX[1:0]=2'b01
I2C1 Interface			
i2c1_sda	I/O	IO_I2C1tpsda_GPIO0a3	GRF_GPIO0A_IOMUX[7:6]=2'b01
i2c1_scl	I/O	IO_I2C1tpsc1_GPIO0a2	GRF_GPIO0A_IOMUX[5:4]=2'b01
I2C2 Interface			
i2c2_sda	I/O	IO_I2C2sda_GPIO2c4	GRF_GPIO0A_IOMUX[9:8]=2'b01
i2c2_scl	I/O	IO_I2C2scl_GPIO2c5	GRF_GPIO0A_IOMUX[11:10]=2'b01

8.6 Application Notes

The I2C controller core operation flow chart below is to describe how the software configures and performs an I2C transaction through this I2C controller core. Descriptions are divided into 3 sections, transmit only mode, receive only mode, and mix mode. Users are strongly advised to follow

- Transmit only mode (I2C_CON[1:0]=2'b00)

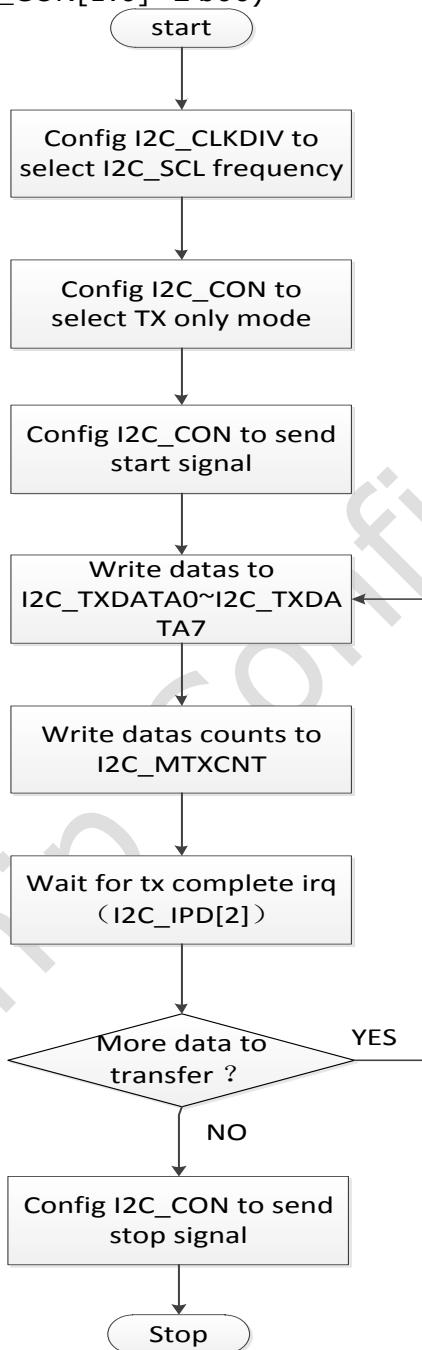


Fig. 8-6 I2C Flow chat for transmit only mode

- Receive only mode (I2C_CON[1:0]=2'b10)

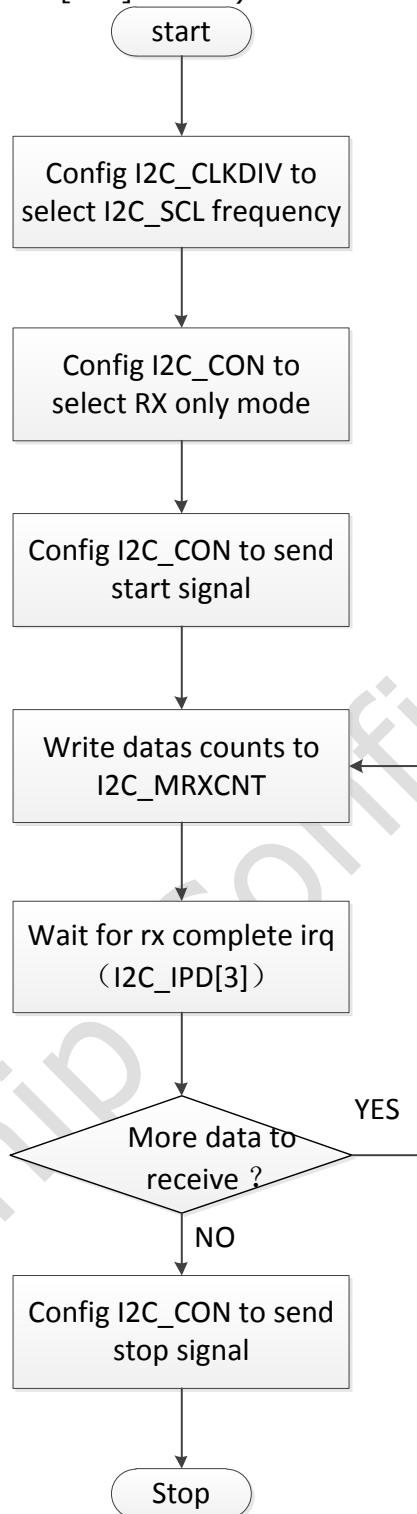


Fig. 8-7 I2C Flow chat for receive only mode

- Mix mode ($I2C_CON[1:0]=2'b01$ or $I2C_CON[1:0]=2'b11$)

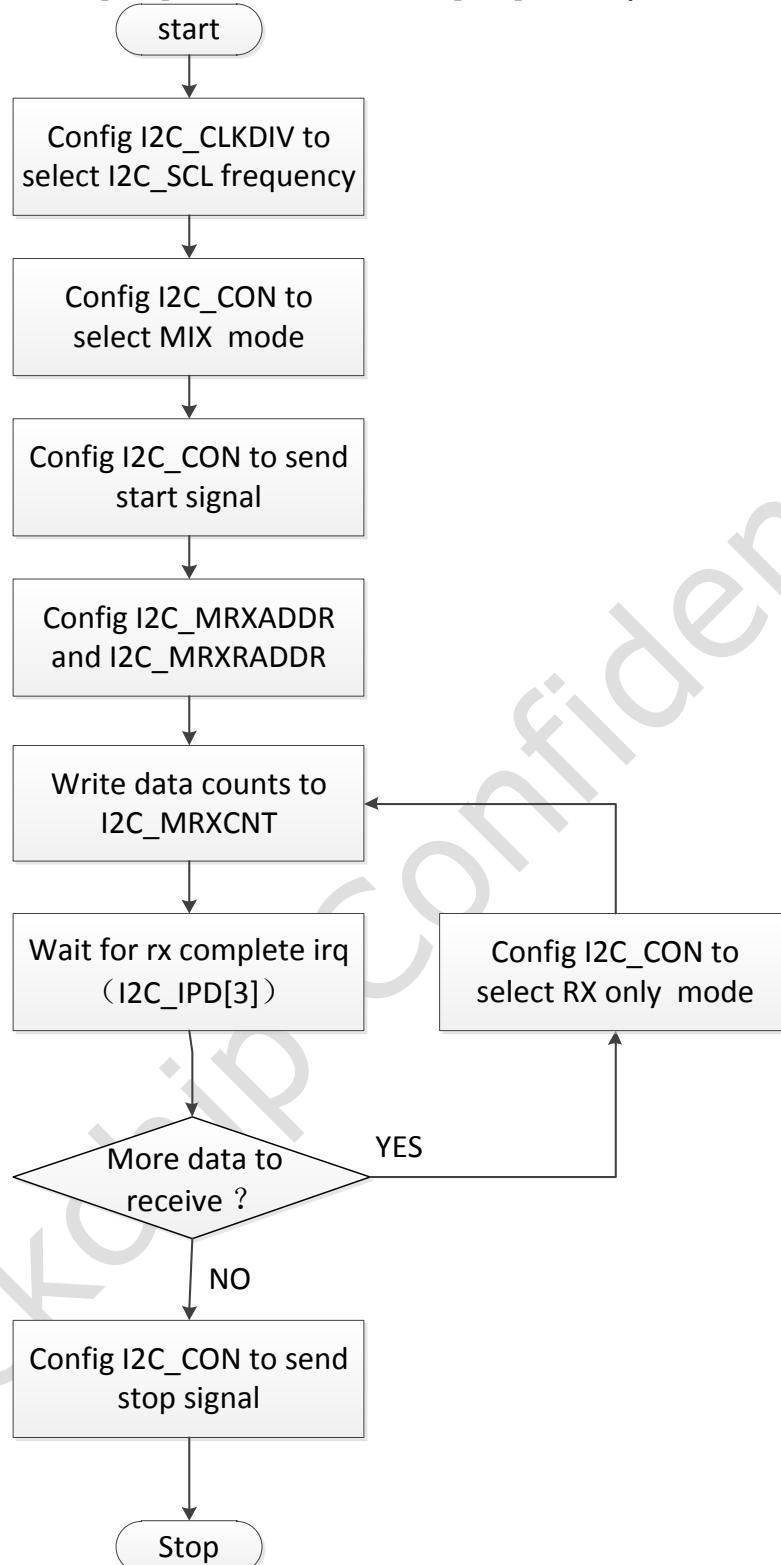


Fig. 8-8 I2C Flow chat for mix mode

Chapter 9 I2S/PCM Controller (8 channel)

9.1 Overview

The I2S/PCM bus is a serial link for digital audio data transfer between devices in the system. Now it is widely used by many semiconductor manufacturers.

Devices often use the I2S bus are ADC, DAC, DSP, CPU, etc. With the I2S interface, we can connect audio devices and the embedded SoC platform together and provide an audio interface solution for the system.

Not only I2S but also PCM mode surround audio output and stereo input are supported in I2S/PCM controller.

- Support five internal 32-bit wide and 32-location deep FIFOs, four for transmitting and one for receiving audio data
- Support AHB bus interface
- Support 16 ~ 32 bits audio data transfer
- Support master and slave mode
- Support DMA handshake interface and configurable DMA water level
- Support transmit FIFO empty, underflow, receive FIFO full, overflow interrupt and all interrupts can be masked
- Support configurable water level of transmit FIFO empty and receive FIFO full interrupt
- Support combine interrupt output
- Support 8channels audio transmitting and 2 channels audio receiving in I2S mode
- Support 2 channels audio receiving in PCM mode
- Support up to 192kHz sample rate
- Support I2S normal, left and right justified mode serial audio data transfer
- Support PCM early, late1, late2, late3 mode serial audio data transfer
- Support MSB or LSB first serial audio data transfer
- Support 16 to 31 bit audio data left or right justified in 32-bit wide FIFO
- Support two 16-bit audio data store together in one 32-bit wide location
- Support 2 independent LRCK signals, one for receiving and one for transmitting audio data
- Support configurable SCLK and LRCK polarity

9.2 Block Diagram

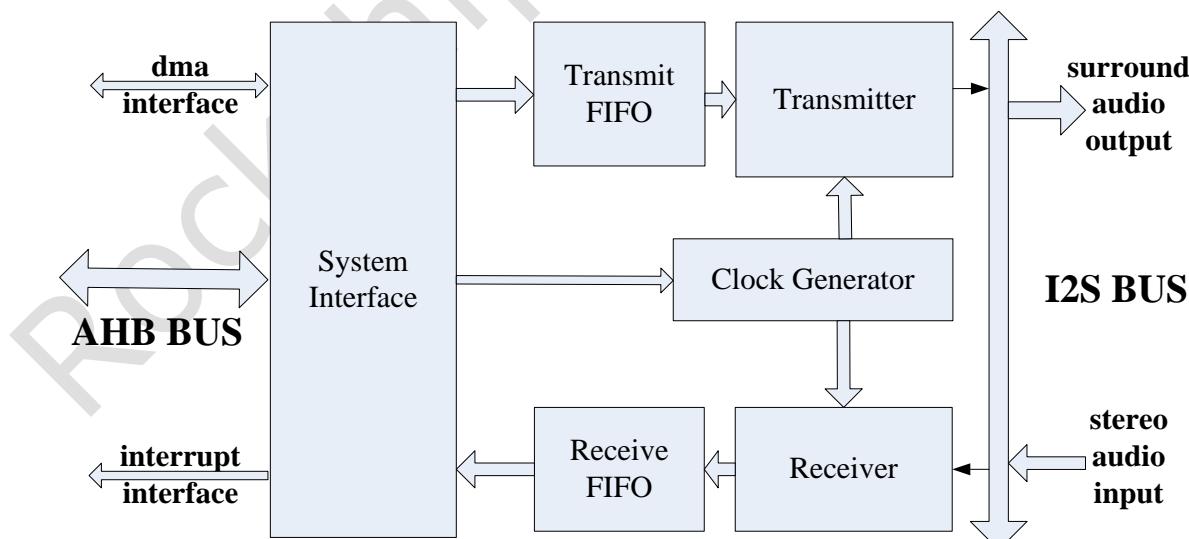


Fig. 9-1 I2S/PCM controller (8 channel) Block Diagram

System Interface

The system interface implements the AHB slave operation. It contains not only control registers of transmitter and receiver inside but also interrupt and DMA handshake interface.

Clock Generator

The Clock Generator implements clock generation function. The input source clock to the

module is MCLK_I2S, and by the divider of the module, the clock generator generates SCLK and LRCK to transmitter and receiver.

Transmitter

The Transmitter implements transmission operation. The transmitter can act as either master or slave, with I2S or PCM mode surround serial audio interface.

Receiver

The Receiver implements receive operation. The receiver can act as either master or slave, with I2S or PCM mode stereo serial audio interface.

Transmit FIFO

There are four transmit fifos in the design. The Transmit FIFO are the buffer to store transmitted audio data. The size of the FIFO is 32bits x 32.

Receive FIFO

There is one receive fifo in the design. The Receive FIFO is the buffer to store received audio data. The size of the FIFO is 32bits x 32.

9.3 Function description

In the I2S/PCM controller, there are four conditions: transmitter-master & receiver-master; transmitter-master & receiver-slave; transmitter-slave & receiver-master; transmitter-slave & receiver-slave.

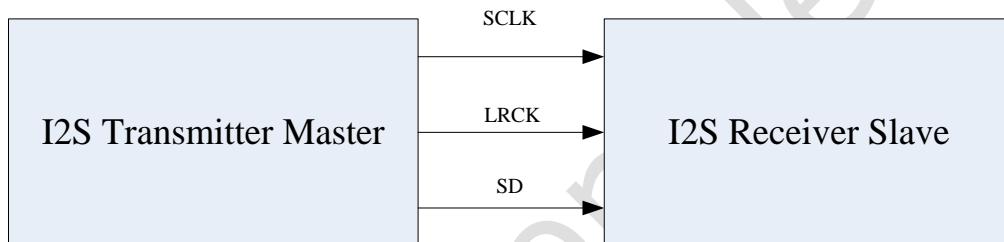


Fig. 9-2 I2S transmitter-master & receiver-slave condition

When transmitter acts as a master, it sends all signals to receiver (slave), and CPU control when to send clock and data to the receiver. When acting as a slave, SD signal still goes from transmitter to receiver, but SCLK and LRCK signals are from receiver (master) to transmitter. Based on three interface specifications, transmitting data should be ready before transmitter receives SCLK and LRCK signals. CPU should know when the receiver to initialize a transaction and when to send data.

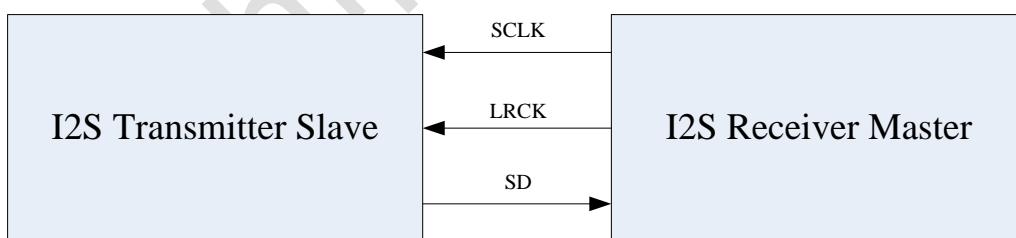


Fig. 9-3 I2S transmitter-slave& receiver-master condition

When the receiver acts as a master, it sends SCLK and LRCK signals to the transmitter (slave) and receives serial data. So CPU must tell the transmitter when to start a transaction for it to prepare transmitting data then the receiver start a transfer and send clock and channel-select signals. When the receiver acts as a slave, CPU should only do initial setting and wait for all signals and then start reading data.

Before transmitting or receiving data, CPU need do initial setting to the I2S register. These includes CPU settings, I2S interface registers settings, and maybe the embedded SoC platform settings. These registers must be set before starting data transfer.

9.3.1 i2s normal mode

This is the waveform of I2S normal mode. For LRCK (i2s_lrck_rx/i2s_lrck_tx) signal, it goes low to indicate left channel and high to right channel. For SD (i2s_sdo,i2s_sdi) signal, it transfers MSB or LSB first and sends the first bit one SCLK clock cycle after LRCK changes. The range of SD signal width is from 16 to 32bits.

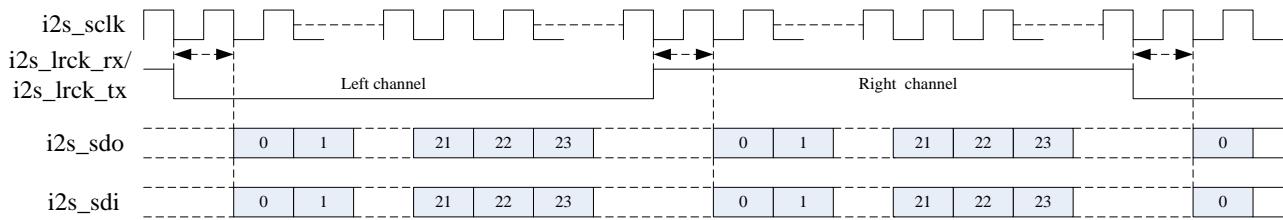


Fig. 9-4 I2S normal mode timing format

9.3.2 i2s left justified mode

This is the waveform of I2S left justified mode. For LRCK (i2s_lrck_rx / i2s_lrck_tx) signal, it goes high to indicate left channel and low to right channel. For SD (i2s_sdo, i2s_sdi) signal, it transfers MSB or LSB first and sends the first bit at the same time when LRCK changes. The range of SD signal width is from 16 to 32bits.

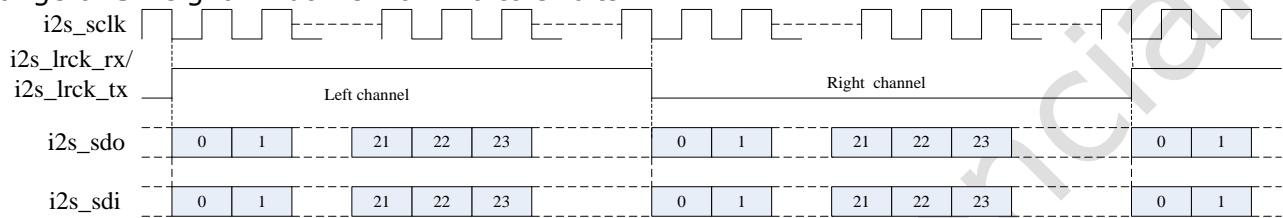


Fig. 9-5 I2S left justified mode timing format

9.3.3 i2s right justified mode

This is the waveform of I2S right justified mode. For LRCK (i2s_lrck_rx / i2s_lrck_tx) signal, it goes high to indicate left channel and low to right channel. For SD (i2s_sdo, i2s_sdi) signal, it transfers MSB or LSB first; but different from I2S normal or left justified mode, its data is aligned to last bit at the edge of the LRCK signal. The range of SD signal width is from 16 to 32bits.

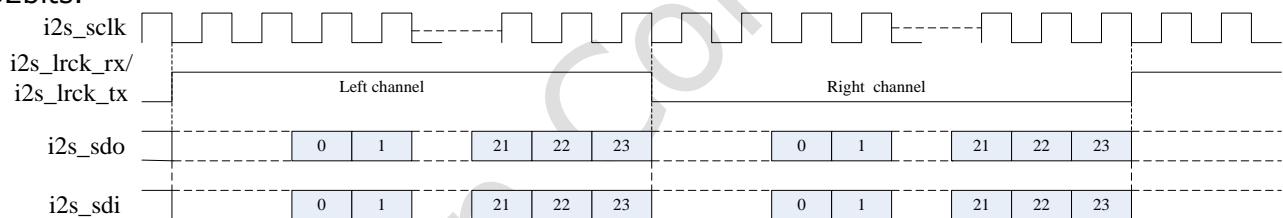


Fig. 9-6 I2S right justified mode timing format

9.3.4 PCM early mode

This is the waveform of PCM early mode. For LRCK (i2s_lrck_rx / i2s_lrck_tx) signal, it goes high to indicate the start of a group of audio channels. For SD (i2s_sdo, i2s_sdi) signal, it transfers MSB or LSB first and sends the first bit at the same time when LRCK goes high. The range of SD signal width is from 16 to 32bits.

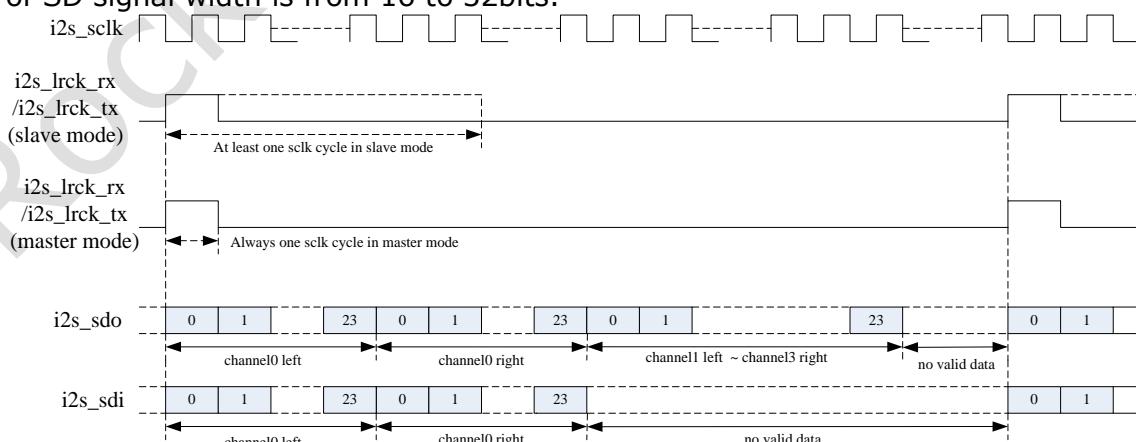


Fig. 9-7 PCM early mode timing format

9.3.5 PCM late1 mode

This is the waveform of PCM late1 mode. For LRCK (i2s_lrck_rx / i2s_lrck_tx) signal, it goes high to indicate the start of a group of audio channels. For SD (i2s_sdo, i2s_sdi) signal, it

transfers MSB or LSB first and sends the first bit one SCLK clock cycle after LRCK goes high. The range of SD signal width is from 16 to 32bits.

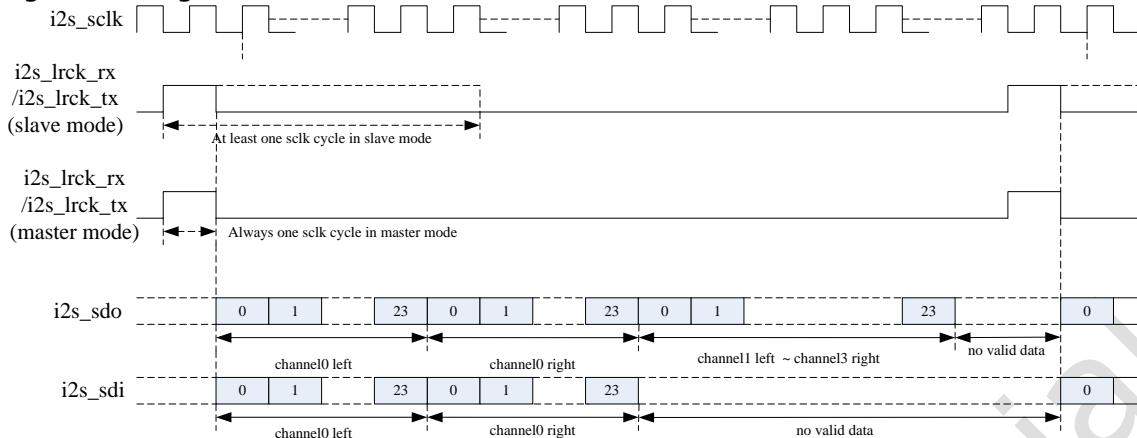


Fig. 9-8 PCM late1 mode timing format

9.3.6 PCM late2 mode

This is the waveform of PCM late2 mode. For LRCK (i2s_lrck_rx / i2s_lrck_tx) signal, it goes high to indicate the start of a group of audio channels. For SD (i2s_sdo, i2s_sdi) signal, it transfers MSB or LSB first and sends the first bit two SCLK clock cycles after LRCK goes high. The range of SD signal width is from 16 to 32bits.

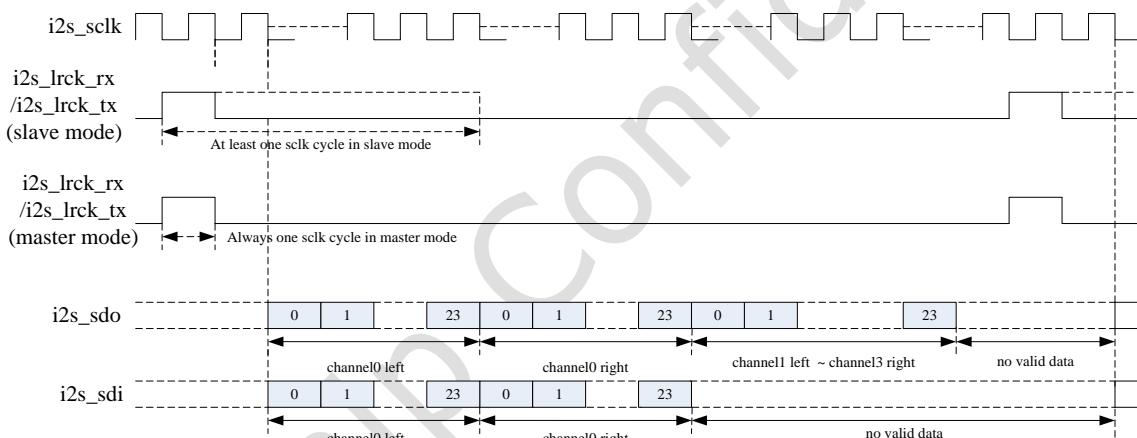


Fig. 9-9 PCM late2 mode timing format

9.3.7 PCM late3 mode

This is the waveform of PCM late3 mode. For LRCK (i2s_lrck_rx / i2s_lrck_tx) signal, it goes high to indicate the start of a group of audio channels. For SD (i2s_sdo, i2s_sdi) signal, it transfers MSB or LSB first and sends the first bit three SCLK clock cycles after LRCK goes high. The range of SD signal width is from 16 to 32bits.

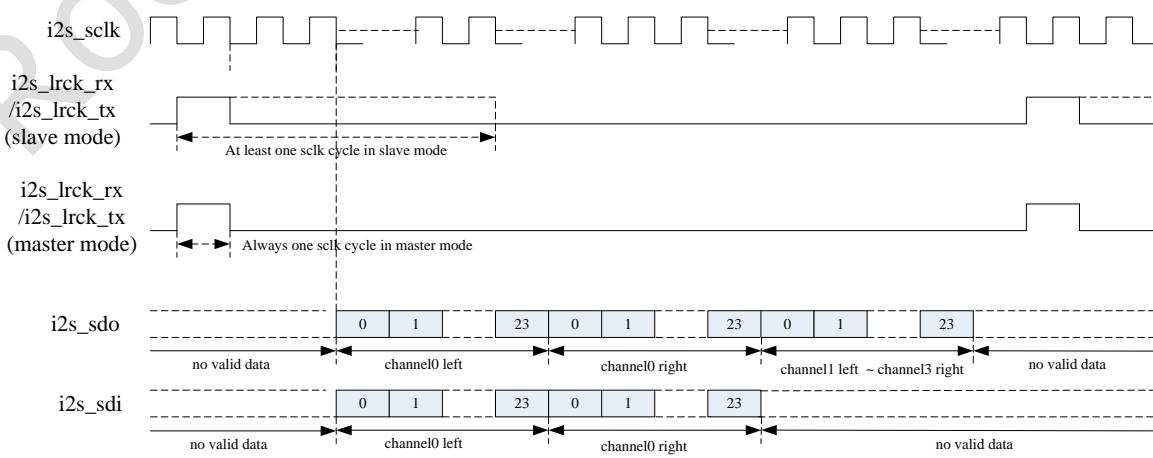


Fig. 9-10 PCM late3 mode timing format

9.4 Register Description

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

9.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
I2S_TXCR	0x0000	W	0x0000000f	transmit operation control register
I2S_RXCR	0x0004	W	0x0000000f	receive operation control register
I2S_CKR	0x0008	W	0x00071f1f	clock generation register
I2S_FIFOLR	0x000c	W	0x00000000	FIFO level register
I2S_DMACR	0x0010	W	0x001f0000	DMA control register
I2S_INTCR	0x0014	W	0x01f00000	interrupt control register
I2S_INTSR	0x0018	W	0x00000000	interrupt status register
I2S_XFER	0x001c	W	0x00000000	Transfer Start Register
I2S_CLR	0x0020	W	0x00000000	SCLK domain logic clear Register
I2S_TXDR	0x0024	W	0x00000000	Transmit FIFO Data Register
I2S_RXDR	0x0028	W	0x00000000	Receive FIFO Data Register

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

9.4.2 Detail Register Description

I2S_TXCR

Address: Operational Base + offset (0x0000)

transmit operation control register

Bit	Attr	Reset Value	Description
31:23	RO	0x0	reserved
22:17	RW	0x00	RCNT right jusitified counter (Can be written only when XFER[0] bit is 0.) Only vailid in I2S Right justified format and slave tx mode is selected. Start to transmit data RCNT sclk cycles after left channel valid.
16:15	RW	0x0	CSR Channel select register 2'b00:2 channel 2'b01:4 channel 2'b10:6 channel 2'b11:8 channel
14	RW	0x0	HWT Halfword word transform (Can be written only when XFER[0] bit is 0.) Only valid when VDW select 16bit data. 0:32 bit data valid from AHB/APB bus. Low 16 bit for left channel and high 16 bit for right channel. 1:low 16bit data valid from AHB/APB bus, high 16 bit data invalid.
13	RO	0x0	reserved

Bit	Attr	Reset Value	Description
12	RW	0x0	SJM Store justified mode (Can be written only when XFER[0] bit is 0.) 16bit~31bit DATA stored in 32 bits width fifo. If VDW select 16bit data, this bit is valid only when HWT select 0.Because if HWT is 1, every fifo unit contain two 16bit data and 32 bit space is full, it is impossible to choose justified mode. 0:right justified 1:left justified
11	RW	0x0	FBM First Bit Mode (Can be written only when XFER[0] bit is 0.) 0:MSB 1:LSB
10:9	RW	0x0	IBM I2S bus mode (Can be written only when XFER[0] bit is 0.) 0:I2S normal 1:I2S Left justified 2:I2S Right justified 3:reserved
8:7	RW	0x0	PBM PCM bus mode (Can be written only when XFER[0] bit is 0.) 0:PCM no delay mode 1:PCM delay 1 mode 2:PCM delay 2 mode 3:PCM delay 3 mode
6	RO	0x0	reserved
5	RW	0x0	TFS Transfer format select (Can be written only when XFER[0] bit is 0.) 0: I2S format 1: PCM format

Bit	Attr	Reset Value	Description
4:0	RW	0x0f	<p>VDW Valid Data width (Can be written only when XFER[0] bit is 0.)</p> <p>0~14:reserved</p> <p>15:16bit</p> <p>16:17bit</p> <p>17:18bit</p> <p>18:19bit</p> <p>.....</p> <p>28:29bit</p> <p>29:30bit</p> <p>30:31bit</p> <p>31:32bit</p>

I2S_RXCR

Address: Operational Base + offset (0x0004)

receive operation control register

Bit	Attr	Reset Value	Description
31:15	RO	0x0	reserved
14	RW	0x0	<p>HWT Halfword word transform (Can be written only when XFER[1] bit is 0.)</p> <p>Only valid when VDW select 16bit data.</p> <p>0:32 bit data valid to AHB/APB bus. Low 16 bit for left channel and high 16 bit for right channel.</p> <p>1:low 16bit data valid to AHB/APB bus, high 16 bit data invalid.</p>
13	RO	0x0	reserved
12	RW	0x0	<p>SJM Store justified mode (Can be written only when XFER[1] bit is 0.)</p> <p>16bit~31bit DATA stored in 32 bits width fifo.</p> <p>If VDW select 16bit data, this bit is valid only when HWT select 0.Because if HWT is 1, every fifo unit contain two 16bit data and 32 bit space is full, it is impossible to choose justified mode.</p> <p>0:right justified 1:left justified</p>
11	RW	0x0	<p>FBM First Bit Mode (Can be written only when XFER[1] bit is 0.)</p> <p>0:MSB 1:LSB</p>

Bit	Attr	Reset Value	Description
10:9	RW	0x0	IBM I2S bus mode (Can be written only when XFER[1] bit is 0.) 0:I2S normal 1:I2S Left justified 2:I2S Right justified 3:reserved
8:7	RW	0x0	PBM PCM bus mode (Can be written only when XFER[1] bit is 0.) 0:PCM no delay mode 1:PCM delay 1 mode 2:PCM delay 2 mode 3:PCM delay 3 mode
6	RO	0x0	reserved
5	RW	0x0	TFS Transfer format select (Can be written only when XFER[1] bit is 0.) 0:i2s 1:pcm
4:0	RW	0x0f	VDW Valid Data width (Can be written only when XFER[1] bit is 0.) 0~14:reserved 15:16bit 16:17bit 17:18bit 18:19bit 28:29bit 29:30bit 30:31bit 31:32bit

I2S_CKR

Address: Operational Base + offset (0x0008)
clock generation register

Bit	Attr	Reset Value	Description
31:28	RO	0x0	reserved
27	RW	0x0	MSS Master/slave mode select (Can be written only when XFER[1] or XFER[0] bit is 0.) 0:master mode(sclk output) 1:slave mode(sclk input)

Bit	Attr	Reset Value	Description
26	RW	0x0	CKP Sclk polarity (Can be written only when XFER[1] or XFER[0] bit is 0.) 0: sample data at posedge sclk and drive data at negedge sclk 1: sample data at negedge sclk and drive data at posedge sclk
25	RW	0x0	RLP Receive lrck polarity (Can be written only when XFER[1] or XFER[0] bit is 0.) 0: normal polartiy (I2S normal: low for left channel, high for right channel I2S left/right just: high for left channel, low for right channel PCM start signal:high valid) 1: oppsite polarity (I2S normal: high for left channel, low for right channel I2S left/right just: low for left channel, high for right channel PCM start signal:low valid)
24	RW	0x0	TLP Transmit lrck polarity (Can be written only when XFER[1] or XFER[0] bit is 0.) 0: normal polartiy (I2S normal: low for left channel, high for right channel I2S left/right just: high for left channel, low for right channel PCM start signal:high valid) 1: oppsite polarity (I2S normal: high for left channel, low for right channel I2S left/right just: low for left channel, high for right channel PCM start signal:low valid)
23:16	RW	0x07	MDIV mclk divider (Can be written only when XFER[1] or XFER[0] bit is 0.) mclk divider = mclk / (sclk-1). For example, if mclk divider is 5, then the frequency of sclk is mclk/6
15:8	RW	0x1f	RSD Receive sclk divider (Can be written only when XFER[1] or XFER[0] bit is 0.) 0~30:reserved 31~255:frequency of rx_lrck= (Receive sclk divider[7:1]+1)*2*frequency of sclk
7:0	RW	0x1f	TSD Transmit sclk divider (Can be written only when XFER[1] or XFER[0] bit is 0.) 0~30:reserved 31~255:frequency of tx_lrck= (Transmit sclk divider[7:1]+1)*2*frequency of sclk

I2S_FIFOLR

Address: Operational Base + offset (0x000c)

FIFO level register

Bit	Attr	Reset Value	Description
31:30	RO	0x0	reserved
29:24	RO	0x00	RFL Receive FIFO Level Contains the number of valid data entries in the receive FIFO.
23:18	RO	0x00	TFL3 Transmit FIFO3 Level Contains the number of valid data entries in the transmit FIFO3.
17:12	RO	0x00	TFL2 Transmit FIFO2 Level Contains the number of valid data entries in the transmit FIFO2.
11:6	RO	0x00	TFL1 Transmit FIFO1 Level Contains the number of valid data entries in the transmit FIFO1.
5:0	RO	0x00	TFL0 Transmit FIFO0 Level Contains the number of valid data entries in the transmit FIFO0.

I2S_DMACR

Address: Operational Base + offset (0x0010)

DMA control register

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24	RW	0x0	RDE Receive DMA Enable 0 : Receive DMA disabled 1 : Receive DMA enabled
23:21	RO	0x0	reserved
20:16	RW	0x1f	RDL Receive Data Level This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or above this field value + 1.
15:9	RO	0x0	reserved
8	RW	0x0	TDE Transmit DMA Enable 0 : Transmit DMA disabled 1 : Transmit DMA enabled
7:5	RO	0x0	reserved

Bit	Attr	Reset Value	Description
4:0	RW	0x00	TDL Transmit Data Level This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the TXFIFO(TXFIFO0 if CSR=00;TXFIFO1 if CSR=01,TXFIFO2 if CSR=10,TXFIFO3 if CSR=11)is equal to or below this field value.

I2S_INTCR

Address: Operational Base + offset (0x0014)

interrupt control register

Bit	Attr	Reset Value	Description
31:25	RO	0x0	reserved
24:20	RW	0x00	RFT Receive FIFO Threshold When the number of receive FIFO entries is more than or equal to this threshold plus 1, the receive FIFO full interrupt is triggered.
19	RO	0x0	reserved
18	WO	0x0	RXOIC RX overrun interrupt clear Write 1 to clear RX overrun interrupt.
17	RW	0x0	RXOIE RX overrun interrupt enable 0:disable 1:enable
16	RW	0x0	RXFIE RX full interrupt enable 0:disable 1:enable
15:9	RO	0x0	reserved
8:4	RW	0x00	TFT Transmit FIFO Threshold When the number of transmit FIFO (TXFIFO0 if CSR=00; TXFIFO1 if CSR=01, TXFIFO2 if CSR=10, TXFIFO3 if CSR=11) entries is less than or equal to this threshold, the transmit FIFO empty interrupt is triggered.
3	RO	0x0	reserved
2	WO	0x0	TXUIC TX underrun interrupt clear Write 1 to clear TX underrun interrupt.
1	RW	0x0	TXUIE TX underrun interrupt enable 0:disable 1:enable

Bit	Attr	Reset Value	Description
0	RW	0x0	TXEIE TX empty interrupt enable 0:disable 1:enable

I2S_INTSR

Address: Operational Base + offset (0x0018)

interrupt status register

Bit	Attr	Reset Value	Description
31:18	RO	0x0	reserved
17	RO	0x0	RXOI RX overrun interrupt 0:inactive 1:active
16	RO	0x0	RXFI RX full interrupt 0:inactive 1:active
15:2	RO	0x0	reserved
1	RO	0x0	TXUI TX underrun interrupt 0:inactive 1:active
0	RO	0x0	TXEI TX empty interrupt 0:inactive 1:active

I2S_XFER

Address: Operational Base + offset (0x001c)

Transfer Start Register

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved
1	RW	0x0	RXS RX Transfer start bit 0:stop RX transfer. 1:start RX transfer
0	RW	0x0	TXS TX Transfer start bit 0:stop TX transfer. 1:start TX transfer

I2S_CLR

Address: Operational Base + offset (0x0020)

SCLK domain logic clear Register

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved
1	RW	0x0	RXC RX logic clear This is a self cleared bit. Write 1 to clear all receive logic.
0	RW	0x0	TXC TX logic clear This is a self cleared bit. Write 1 to clear all transmit logic.

I2S_TXDR

Address: Operational Base + offset (0x0024)

Transmit FIFO Data Register

Bit	Attr	Reset Value	Description
31:0	WO	0x00000000	TXDR Transmit FIFO Data Register When it is written to, data are moved into the transmit FIFO.

I2S_RXDR

Address: Operational Base + offset (0x0028)

Receive FIFO Data Register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	RXDR Receive FIFO Data Register When the register is read, data in the receive FIFO is accessed.

9.5 Interface description

Table 9-1 I2S Interface Description

Module Pin	Direction	Pad Name	IOMUX Setting
GRF_SOC_CON1[4]=0			
i2s_mclk	I	IO_I2Smclk_GPIO1a0	GRF_GPIO1A_IOMUX[0]=1'b1
i2s_sclk	I/O	IO_I2Ssclk_GPIO1a1	GRF_GPIO1A_IOMUX[2]=1'b1
i2s_lrck_rx	I/O	IO_I2Slrckrx_PWM10_GPIO1a2	GRF_GPIO1A_IOMUX[5:4]=2'b01
i2s_lrck_tx	I/O	IO_I2Slrcktx_GPIO1a3	GRF_GPIO1A_IOMUX[6]=1'b1
i2s_sdo0	O	IO_I2Ssdo_GPIO1a4	GRF_GPIO1A_IOMUX[8]=1'b1
i2s_sdo1	O	IO_I2Ssdo1_GPIO2d6	GRF_GPIO2D_IOMUX[12]=1'b1
i2s_sdo2	O	IO_I2Ssdo2_GPIO2d5	GRF_GPIO2D_IOMUX[10]=1'b1
i2s_sdo3	O	IO_I2Ssdo3_GPIO2d4	GRF_GPIO2D_IOMUX[8]=1'b1
i2s_sdi	I	IO_I2Ssdi_GPIO1a5	GRF_GPIO1A_IOMUX[10]=1'b1
GRF_SOC_CON1[4]=1			
i2s1_mclk	I	IO_MMC1clkout_I2S1mclk_GPI00b1	GRF_GPIO0B_IOMUX[3:2]=2'b10
i2s1_sclk	I/O	IO_MMC1d3_I2S1sclk_GPIO0b6	GRF_GPIO0B_IOMUX[13:12]=2'b10
i2s1_lrck_rx	I/O	IO_MMC1d0_I2S1lrckrx_GPIO0b3	GRF_GPIO0B_IOMUX[7:6]=2'b10
i2s1_lrck_tx	I/O	IO_MMC1d1_I2S1lrcktx_GPIO0b4	GRF_GPIO0B_IOMUX[9:8]=2'b10
i2s1_sdi	I	IO_MMC1d2_I2S1sdi_GPIO0b5	GRF_GPIO0B_IOMUX[11:10]=2'b10
i2s1_sdo	O	IO_MMC1cmd_I2S1sdo_GPIO0b0	GRF_GPIO0B_IOMUX[1:0]=2'b10

Notes: I=input, O=output, I/O=input/output, bidirectional

There is only one i2s module in the design, which sets of i2s interface to use depends on the value of GRF_SOC_CON1[4]. For more information of iomux, please refer to GRF module.

9.6 Application Notes

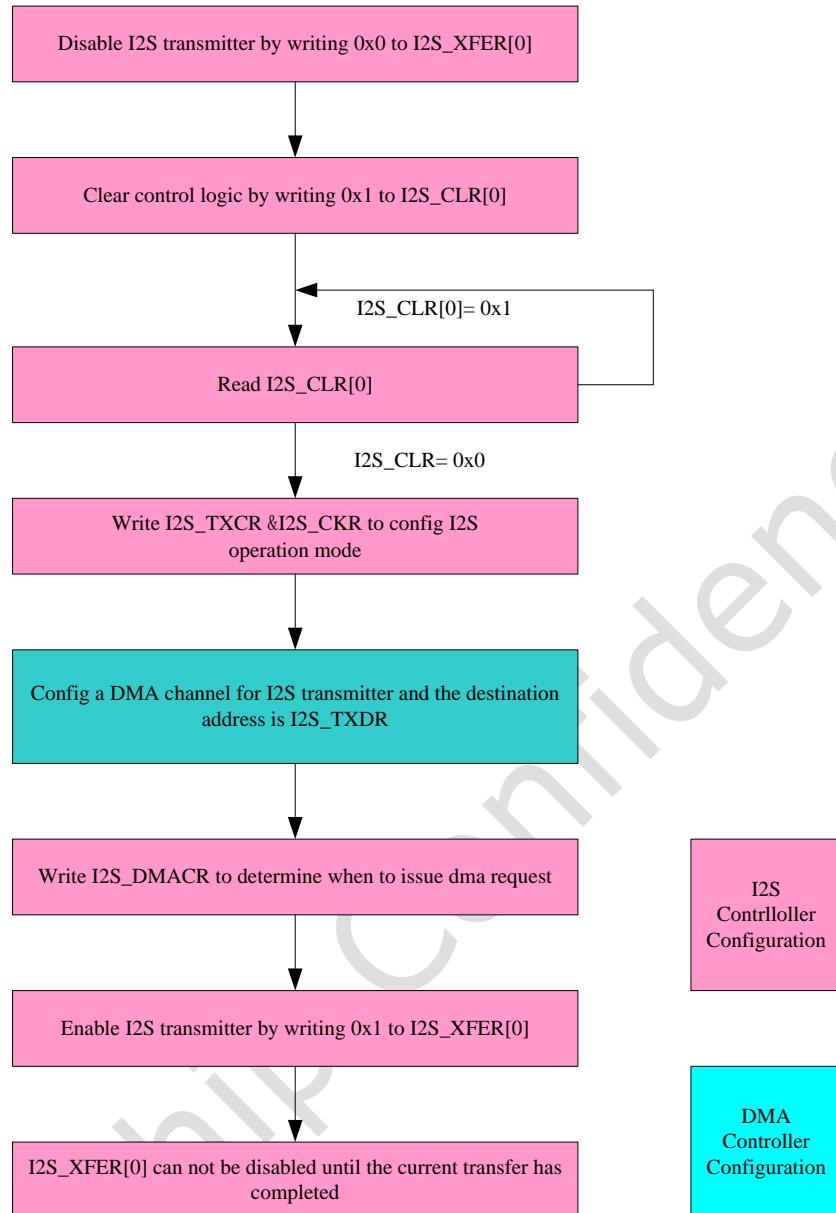


Fig. 9-11 I2S/PCM controller transmit operation flow chart

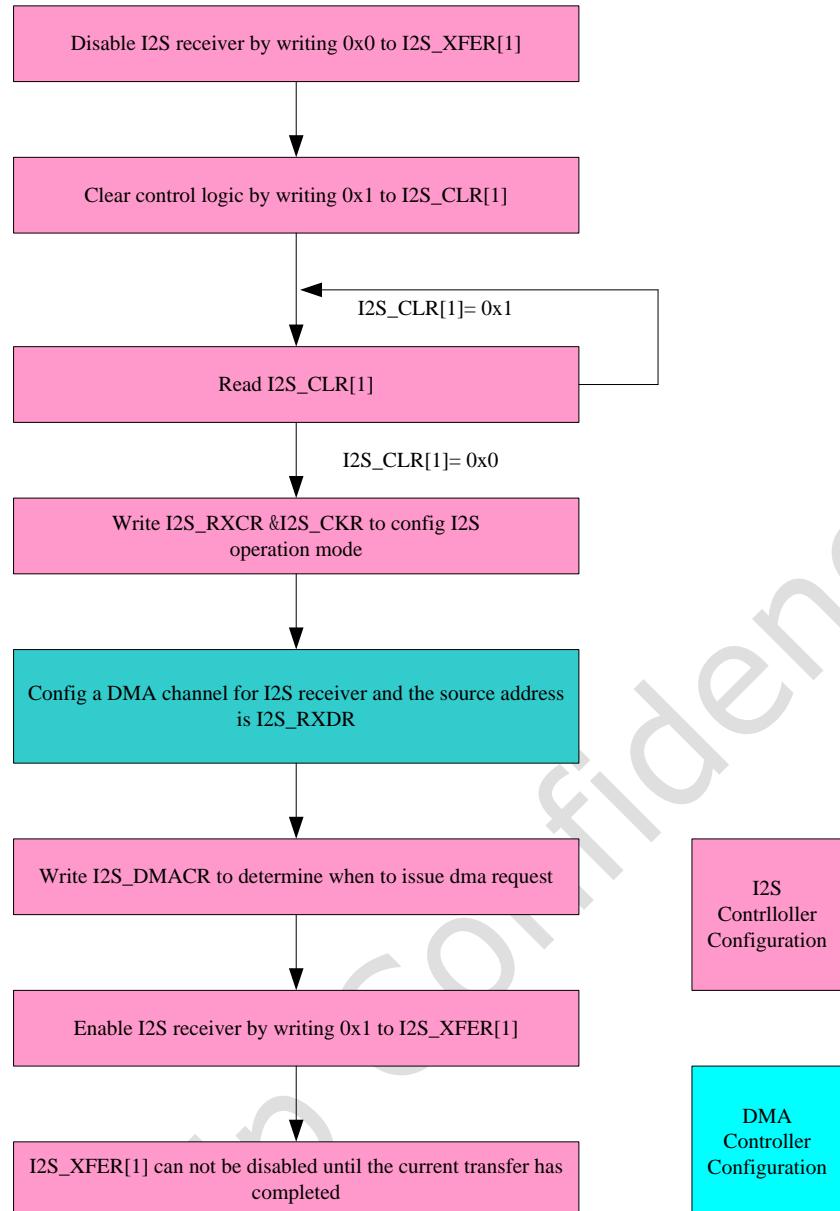


Fig. 9-12 I2S/PCM controller receive operation flow chart

Chapter 10 Serial Peripheral Interface (SPI)

10.1 Overview

The serial peripheral interface is an APB slave device. A four wire full duplex serial protocol from Motorola. There are four possible combinations for the serial clock phase and polarity. The clock phase (SCPH) determines whether the serial transfer begins with the falling edge of slave select signals or the first edge of the serial clock. The slave select line is held high when the SPI is idle or disabled. This SPI controller can work as either master or slave mode.

SPI Controller supports the following features:

- Support Motorola SPI,TI Synchronous Serial Protocol and National Semiconductor Micro wire interface
- Support 32-bit APB bus
- Support two internal 16-bit wide and 32-location deep FIFOs, one for transmitting and the other for receiving serial data
- Support two chip select signals in master mode
- Support 4,8,16 bit serial data transfer
- Support configurable interrupt polarity
- Support asynchronous APB bus and SPI clock
- Support master and slave mode
- Support DMA handshake interface and configurable DMA water level
- Support transmit FIFO empty, underflow, receive FIFO full, overflow, interrupt and all interrupts can be masked
- Support configurable water level of transmit FIFO empty and receive FIFO full interrupt
- Support combine interrupt output
- Support up to half of SPI clock frequency transfer in master mode and one sixth of SPI clock frequency transfer in slave mode
- Support full and half duplex mode transfer
- Stop transmitting SCLK if transmit FIFO is empty or receive FIFO is full in master mode
- Support configurable delay from chip select active to SCLK active in master mode
- Support configurable period of chip select inactive between two parallel data in master mode
- Support big and little endian, MSB and LSB first transfer
- Support two 8-bit audio data store together in one 16-bit wide location
- Support sample RXD 0~3 SPI clock cycles later
- Support configurable SCLK polarity and phase
- Support fix and incremental address access to transmit and receive FIFO

10.2 Block Diagram

The SPI Controller comprises with:

- AMBA APB interface and DMA Controller Interface
- Transmit and receive FIFO controllers and an FSM controller
- Register block
- Shift control and interrupt

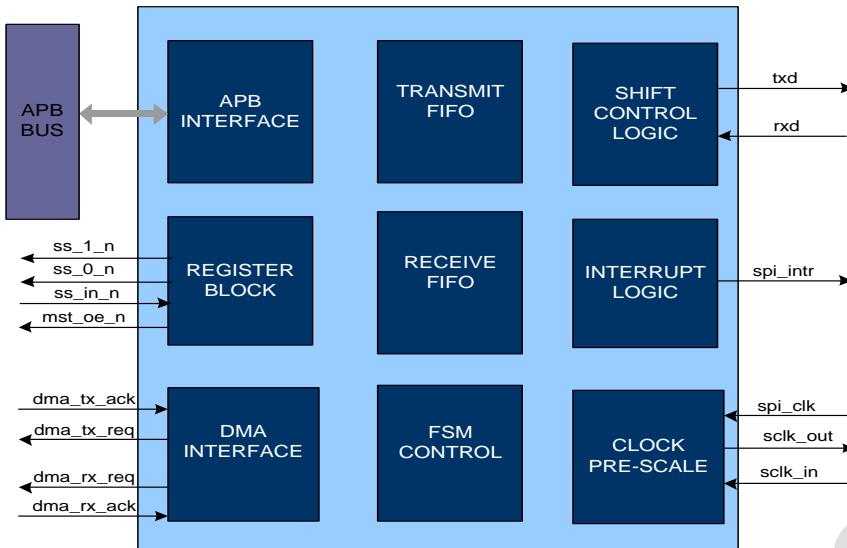


Fig. 10-1 SPI Controller Block diagram

APB INTERFACE

The host processor accesses data, control, and status information on the SPI through the APB interface. The SPI supports APB data bus widths of 32 bits and 8 or 16 bits when reading or writing internal FIFO if data frame size(SPI_CTRL0[1:0]) is set to 8 bits.

DMA INTERFACE

This block has a handshaking interface to a DMA Controller to request and control transfers. The APB bus is used to perform the data transfer to or from the DMA Controller.

FIFO LOGIC

For transmit and receive transfers, data transmitted from the SPI to the external serial device is written into the transmit FIFO. Data received from the external serial device into the SPI is pushed into the receive FIFO. Both fifos are 32x16bits.

FSM CONTROL

Control the state's transformation of the design.

REGISTER BLOCK

All registers in the SPI are addressed at 32-bit boundaries to remain consistent with the APB bus. Where the physical size of any register is less than 32-bits wide, the upper unused bits of the 32-bit boundary are reserved. Writing to these bits has no effect; reading from these bits returns 0.

SHIFT CONTROL

Shift control logic shift the data from the transmit fifo or to the receive fifo. This logic automatically right-justifies receive data in the receive FIFO buffer.

INTERRUPT CONTROL

The SPI supports combined and individual interrupt requests, each of which can be masked. The combined interrupt request is the ORed result of all other SPI interrupts after masking.

10.3 Function Description

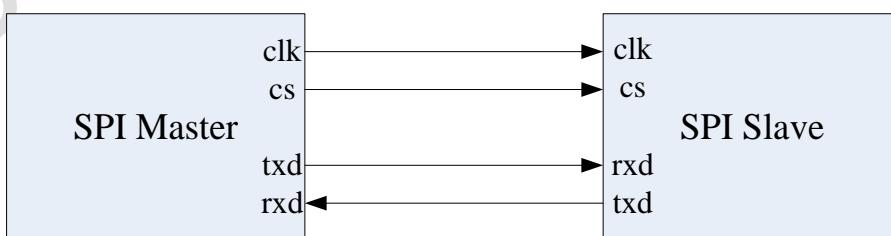


Fig. 10-2 SPI Master and Slave Interconnection

The SPI controller support dynamic switching between master and slave in a system. The diagram show how the SPI controller connects with other SPI devices.

Operation Modes

The SPI can be configured in the following two fundamental modes of operation: Master Mode when SPI_CTRL0 [20] is 1'b0, Slave Mode when SPI_CTRL0 [20] is 1'b1.

Transfer Modes

The SPI operates in the following three modes when transferring data on the serial bus.

1). Transmit and Receive

When SPI_CTRLR0 [19:18]== 2'b00, both transmit and receive logic are valid.

2).Transmit Only

When SPI_CTRLR0 [19:18] == 2'b01, the receive data are invalid and should not be stored in the receive FIFO.

3).Receive Only

When SPI_CTRLR0 [19:18]== 2'b10, the transmit data are invalid.

Clock Ratios

A summary of the frequency ratio restrictions between the bit-rate clock (sclk_out/sclk_in) and the SPI peripheral clock (spi_clk) are described as,

When SPI Controller works as master, the $F_{spi_clk} \geq 2 \times (\text{maximum } F_{sclk_out})$

When SPI Controller works as slave, the $F_{spi_clk} \geq 6 \times (\text{maximum } F_{sclk_in})$

With the SPI, the clock polarity (SCPOL) configuration parameter determines whether the inactive state of the serial clock is high or low. To transmit data, both SPI peripherals must have identical serial clock phase (SCPH) and clock polarity (SCPOL) values. The data frame can be 4/8/16 bits in length.

When the configuration parameter SCPH = 0, data transmission begins on the falling edge of the slave select signal. The first data bit is captured by the master and slave peripherals on the first edge of the serial clock; therefore, valid data must be present on the txd and rxd lines prior to the first serial clock edge. The following two figures show a timing diagram for a single SPI data transfer with SCPH = 0. The serial clock is shown for configuration parameters SCPOL = 0 and SCPOL = 1.

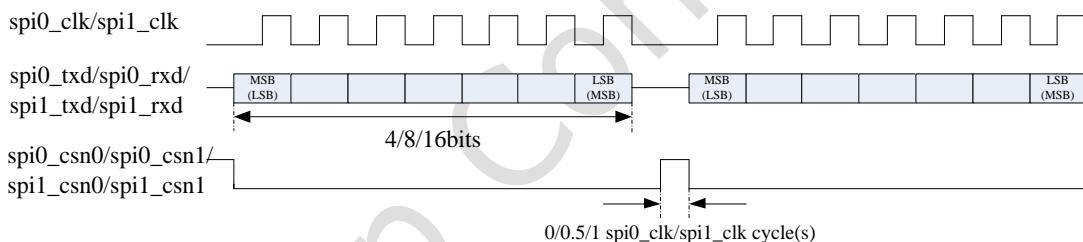


Fig. 10-3 SPI Format (SCPH=0 SCPOL=0)

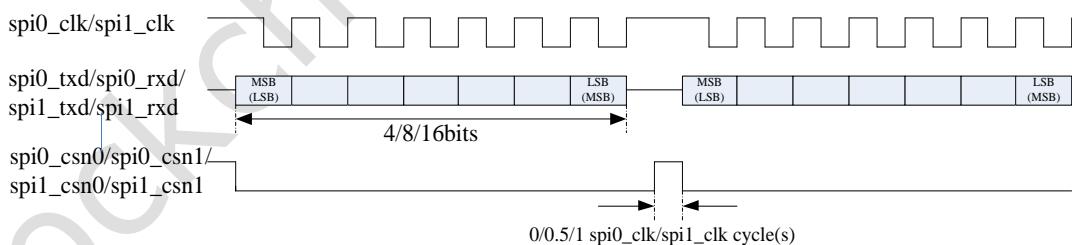


Fig. 10-4 SPI Format (SCPH=0 SCPOL=1)

When the configuration parameter SCPH = 1, both master and slave peripherals begin transmitting data on the first serial clock edge after the slave select line is activated. The first data bit is captured on the second (trailing) serial clock edge. Data are propagated by the master and slave peripherals on the leading edge of the serial clock. During continuous data frame transfers, the slave select line may be held active-low until the last bit of the last frame has been captured. The following two figures show the timing diagram for the SPI format when the configuration parameter SCPH = 1.

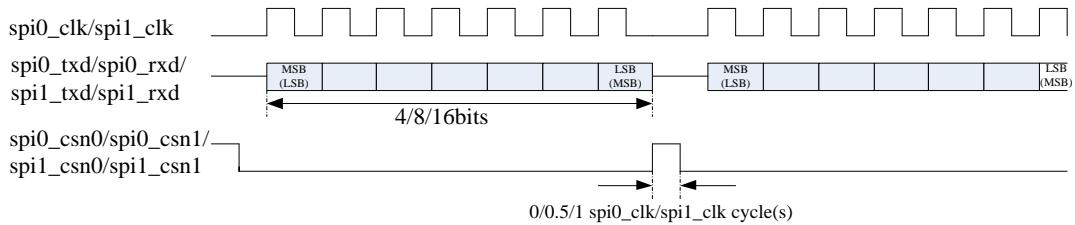


Fig. 10-5 SPI Format (SCPH=1 SCPOL=0)

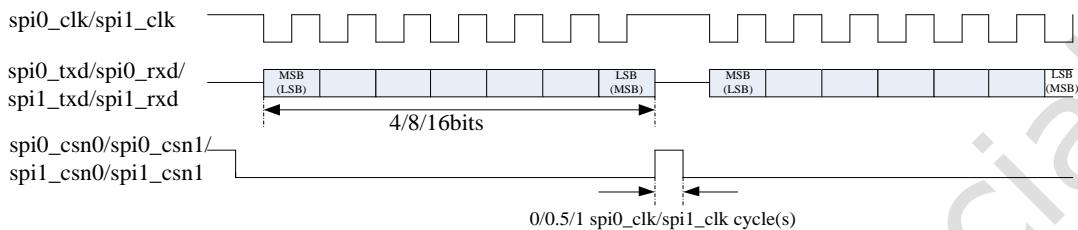


Fig. 10-6 SPI Format (SCPH=1 SCPOL=1)

10.4 Register Description

10.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
SPI_CTRLR0	0x0000	W	0x00000002	Control Register 0
SPI_CTRLR1	0x0004	W	0x00000000	Control Register 1
SPI_ENR	0x0008	W	0x00000000	SPI Enable
SPI_SER	0x000c	W	0x00000000	Slave Enable Register
SPI_BAUDR	0x0010	W	0x00000000	Baud Rate Select
SPI_TXFTLR	0x0014	W	0x00000000	Transmit FIFO Threshold Level
SPI_RXFTLR	0x0018	W	0x00000000	Receive FIFO Threshold Level
SPI_TXFLR	0x001c	W	0x00000000	Transmit FIFO Level
SPI_RXFLR	0x0020	W	0x00000000	Receive FIFO Level
SPI_SR	0x0024	W	0x0000000c	SPI Status
SPI_IPR	0x0028	W	0x00000000	Interrupt Polarity
SPI_IMR	0x002c	W	0x00000000	Interrupt Mask
SPI_ISR	0x0030	W	0x00000000	Interrupt Status
SPI_RISR	0x0034	W	0x00000001	Raw Interrupt Status
SPI_ICR	0x0038	W	0x00000000	Interrupt Clear
SPI_DMACR	0x003c	W	0x00000000	DMA Control
SPI_DMATDLR	0x0040	W	0x00000000	DMA Transmit Data Level
SPI_DMARDLR	0x0044	W	0x00000000	DMA Receive Data Level
SPI_TXDR	0x0048	W	0x00000000	Transmit FIFO Data
SPI_RXDR	0x004c	W	0x00000000	Receive FIFO Data

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

10.4.2 Detail Register Description

SPI_CTRLR0

Address: Operational Base + offset (0x0000)
Control Register 0

Bit	Attr	Reset Value	Description
31:22	RO	0x0	reserved
21	RW	0x0	MTM Microwire Transfer Mode Valid when frame format is set to National Semiconductors Microwire. 1'b0: non-sequential transfer 1'b1: sequential transfer
20	RW	0x0	OPM Operation Mode 1'b0: Master Mode 1'b1: Slave Mode
19:18	RW	0x0	XFM Transfer Mode 2'b00 :Transmit & Receive 2'b01 : Transmit Only 2'b10 : Receive Only 2'b11 :reserved
17:16	RW	0x0	FRF Frame Format 2'b00: Motorola SPI 2'b01: Texas Instruments SSP 2'b10: National Semiconductors Microwire 2'b11 : Reserved
15:14	RW	0x0	RSD Rxd Sample Delay When SPI is configured as a master, if the rxd data cannot be sampled by the sclk_out edge at the right time, this register should be configured to define the number of the spi_clk cycles after the active sclk_out edge to sample rxd data later when SPI works at high frequency. 2'b00:do not delay 2'b01:1 cycle delay 2'b10:2 cycles delay 2'b11:3 cycles delay
13	RW	0x0	BHT Byte and Halfword Transform Valid when data frame size is 8bit. 1'b0:apb 16bit write/read, spi 8bit write/read 1'b1: apb 8bit write/read, spi 8bit write/read
12	RW	0x0	FBM First Bit Mode 1'b0:first bit is MSB 1'b1:first bit is LSB

Bit	Attr	Reset Value	Description
11	RW	0x0	<p>EM Endian Mode Serial endian mode can be configured by this bit. Apb endian mode is always little endian. 1'b0:little endian 1'b1:big endian</p>
10	RW	0x0	<p>SSD ss_n to sclk_out delay Valid when the frame format is set to Motorola SPI and SPI used as a master. 1'b0: the period between ss_n active and sclk_out active is half sclk_out cycles. 1'b1: the period between ss_n active and sclk_out active is one sclk_out cycle.</p>
9:8	RW	0x0	<p>CSM Chip Select Mode Valid when the frame format is set to Motorola SPI and SPI used as a master. 2'b00: ss_n keep low after every frame data is transferred. 2'b01:ss_n be high for half sclk_out cycles after every frame data is transferred. 2'b10: ss_n be high for one sclk_out cycle after every frame data is transferred. 2'b11:reserved</p>
7	RW	0x0	<p>SCPOL Serial Clock Polarity Valid when the frame format is set to Motorola SPI. 1'b0: Inactive state of serial clock is low 1'b1: Inactive state of serial clock is high</p>
6	RW	0x0	<p>SCPH Serial Clock Phase Valid when the frame format is set to Motorola SPI. 1'b0: Serial clock toggles in middle of first data bit 1'b1: Serial clock toggles at start of first data bit</p>

Bit	Attr	Reset Value	Description
5:2	RW	0x0	CFS Control Frame Size Selects the length of the control word for the Microwire frame format. 4'b0000~0010:reserved 4'b0011:4-bit serial data transfer 4'b0100:5-bit serial data transfer 4'b0101:6-bit serial data transfer 4'b0110:7-bit serial data transfer 4'b0111:8-bit serial data transfer 4'b1000:9-bit serial data transfer 4'b1001:10-bit serial data transfer 4'b1010:11-bit serial data transfer 4'b1011:12-bit serial data transfer 4'b1100:13-bit serial data transfer 4'b1101:14-bit serial data transfer 4'b1110:15-bit serial data transfer 4'b1111:16-bit serial data transfer
1:0	RW	0x2	DFS Data Frame Size Selects the data frame length. 2'b00:4bit data 2'b01:8bit data 2'b10:16bit data 2'b11:reserved

SPI_CTRLR1

Address: Operational Base + offset (0x0004)

Control Register 1

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0000	NDM Number of Data Frames When Transfer Mode is receive only, this register field sets the number of data frames to be continuously received by the SPI. The SPI continues to receive serial data until the number of data frames received is equal to this register value plus 1, which enables you to receive up to 64 KB of data in a continuous transfer.

SPI_ENR

Address: Operational Base + offset (0x0008)

SPI Enable

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved

Bit	Attr	Reset Value	Description
0	RW	0x0	<p>ENR SPI Enable</p> <p>1'b1: Enable all SPI operations. 1'b0: Disable all SPI operations</p> <p>Transmit and receive FIFO buffers are cleared when the device is disabled.</p>

SPI_SER

Address: Operational Base + offset (0x000c)

Slave Enable Register

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved
1	RW	0x0	<p>SER1 Slave 1 Select Enable</p> <p>1'b1: Enable chip select 1 1'b0: Disable chip select 1</p> <p>This register is valid only when SPI is configured as a master device.</p>
0	RW	0x0	<p>SERO Slave Select Enable</p> <p>1'b1: Enable chip select 0 1'b0: Disable chip select 0</p> <p>This register is valid only when SPI is configured as a master device.</p>

SPI_BAUDR

Address: Operational Base + offset (0x0010)

Baud Rate Select

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0000	<p>BAUDR Baud Rate Select SPI Clock Divider.</p> <p>This register is valid only when the SPI is configured as a master device.</p> <p>The LSB for this field is always set to 0 and is unaffected by a write operation, which ensures an even value is held in this register.</p> <p>If the value is 0, the serial output clock (sclk_out) is disabled. The frequency of the sclk_out is derived from the following equation:</p> <p>$F_{sclk_out} = F_{spi_clk} / SCKDV$</p> <p>Where SCKDV is any even value between 2 and 65534.</p> <p>For example:</p> <p>for $F_{spi_clk} = 3.6864\text{MHz}$ and $SCKDV = 2$</p> <p>$F_{sclk_out} = 3.6864/2 = 1.8432\text{MHz}$</p>

SPI_TXFTLR

Address: Operational Base + offset (0x0014)

Transmit FIFO Threshold Level

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4:0	RW	0x00	TXFTLR Transmit FIFO Threshold Level When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered.

SPI_RXFTLR

Address: Operational Base + offset (0x0018)

Receive FIFO Threshold Level

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4:0	RW	0x00	RXFTLR Receive FIFO Threshold Level When the number of receive FIFO entries is greater than or equal to this value + 1, the receive FIFO full interrupt is triggered.

SPI_TXFLR

Address: Operational Base + offset (0x001c)

Transmit FIFO Level

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5:0	RO	0x00	TXFLR Transmit FIFO Level Contains the number of valid data entries in the transmit FIFO.

SPI_RXFLR

Address: Operational Base + offset (0x0020)

Receive FIFO Level

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5:0	RO	0x00	RXFLR Receive FIFO Level Contains the number of valid data entries in the receive FIFO.

SPI_SR

Address: Operational Base + offset (0x0024)

SPI Status

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved

Bit	Attr	Reset Value	Description
4	RO	0x0	RFF Receive FIFO Full 1'b0: Receive FIFO is not full 1'b1: Receive FIFO is full
3	RO	0x1	RFE Receive FIFO Empty 1'b0: Receive FIFO is not empty 1'b1: Receive FIFO is empty
2	RO	0x1	TFE Transmit FIFO Empty 1'b0: Transmit FIFO is not empty 1'b1: Transmit FIFO is empty
1	RO	0x0	TFF Transmit FIFO Full 1'b0: Transmit FIFO is not full 1'b1: Transmit FIFO is full
0	RO	0x0	BSF SPI Busy Flag When set, indicates that a serial transfer is in progress; when cleared indicates that the SPI is idle or disabled. 1'b0: SPI is idle or disabled 1'b1: SPI is actively transferring data

SPI_IPR

Address: Operational Base + offset (0x0028)

Interrupt Polarity

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	IPR Interrupt Polarity Interrupt Polarity Register 1'b0: Active Interrupt Polarity Level is HIGH 1'b1: Active Interrupt Polarity Level is LOW

SPI_IMR

Address: Operational Base + offset (0x002c)

Interrupt Mask

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4	RW	0x0	RFFIM Receive FIFO Full Interrupt Mask 1'b0: spi_rxf_intr interrupt is masked 1'b1: spi_rxf_intr interrupt is not masked

Bit	Attr	Reset Value	Description
3	RW	0x0	RFOIM Receive FIFO Overflow Interrupt Mask 1'b0: spi_rxo_intr interrupt is masked 1'b1: spi_rxo_intr interrupt is not masked
2	RW	0x0	RFUIM Receive FIFO Underflow Interrupt Mask 1'b0: spi_rxu_intr interrupt is masked 1'b1: spi_rxu_intr interrupt is not masked
1	RW	0x0	TFOIM Transmit FIFO Overflow Interrupt Mask 1'b0: spi_txo_intr interrupt is masked 1'b1: spi_txo_intr interrupt is not masked
0	RW	0x0	TFEIM Transmit FIFO Empty Interrupt Mask 1'b0: spi_txe_intr interrupt is masked 1'b1: spi_txe_intr interrupt is not masked

SPI_ISR

Address: Operational Base + offset (0x0030)

Interrupt Status

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4	RO	0x0	RFFIS Receive FIFO Full Interrupt Status 1'b0: spi_rxf_intr interrupt is not active after masking 1'b1: spi_rxf_intr interrupt is full after masking
3	RO	0x0	RFOIS Receive FIFO Overflow Interrupt Status 1'b0: spi_rxo_intr interrupt is not active after masking 1'b1: spi_rxo_intr interrupt is active after masking
2	RO	0x0	RFUIS Receive FIFO Underflow Interrupt Status 1'b0: spi_rxu_intr interrupt is not active after masking 1'b1: spi_rxu_intr interrupt is active after masking
1	RO	0x0	TFOIS Transmit FIFO Overflow Interrupt Status 1'b0: spi_txo_intr interrupt is not active after masking 1'b1: spi_txo_intr interrupt is active after masking
0	RO	0x0	TFEIS Transmit FIFO Empty Interrupt Status 1'b0: spi_txe_intr interrupt is not active after masking 1'b1: spi_txe_intr interrupt is active after masking

SPI_RISR

Address: Operational Base + offset (0x0034)

Raw Interrupt Status

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4	RO	0x0	RFFRIS Receive FIFO Full Raw Interrupt Status 1'b0: spi_rxf_intr interrupt is not active prior to masking 1'b1: spi_rxf_intr interrupt is full prior to masking
3	RO	0x0	RFORIS Receive FIFO Overflow Raw Interrupt Status 1'b0 = spi_rxo_intr interrupt is not active prior to masking 1'b1 = spi_rxo_intr interrupt is active prior to masking
2	RO	0x0	RFURIS Receive FIFO Underflow Raw Interrupt Status 1'b0: spi_rxu_intr interrupt is not active prior to masking 1'b1: spi_rxu_intr interrupt is active prior to masking
1	RO	0x0	TFORIS Transmit FIFO Overflow Raw Interrupt Status 1'b0: spi_txo_intr interrupt is not active prior to masking 1'b1: spi_txo_intr interrupt is active prior to masking
0	RO	0x1	TFERIS Transmit FIFO Empty Raw Interrupt Status 1'b0: spi_txe_intr interrupt is not active prior to masking 1'b1: spi_txe_intr interrupt is active prior to masking

SPI_ICR

Address: Operational Base + offset (0x0038)

Interrupt Clear

Bit	Attr	Reset Value	Description
31:4	RO	0x0	reserved
3	WO	0x0	CTFOI Clear Transmit FIFO Overflow Interrupt Write 1 to Clear Transmit FIFO Overflow Interrupt
2	WO	0x0	CRFOI Clear Receive FIFO Overflow Interrupt Write 1 to Clear Receive FIFO Overflow Interrupt
1	WO	0x0	CRFUI Clear Receive FIFO Underflow Interrupt Write 1 to Clear Receive FIFO Underflow Interrupt
0	WO	0x0	CCI Clear Combined Interrupt Write 1 to Clear Combined Interrupt

SPI_DMACR

Address: Operational Base + offset (0x003c)

DMA Control

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:2	RO	0x0	reserved
1	RW	0x0	TDE Transmit DMA Enable 1'b0: Transmit DMA disabled 1'b1: Transmit DMA enabled
0	RW	0x0	RDE Receive DMA Enable 1'b0: Receive DMA disabled 1'b1: Receive DMA enabled

SPI_DMATDLR

Address: Operational Base + offset (0x0040)

DMA Transmit Data Level

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4:0	RW	0x00	TDL Transmit Data Level This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the transmit FIFO is equal to or below this field value, and Transmit DMA Enable (DMACR[1]) = 1.

SPI_DMARDLR

Address: Operational Base + offset (0x0044)

DMA Receive Data Level

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4:0	RW	0x00	RDL Receive Data Level This bit field controls the level at which a DMA request is made by the receive logic. The watermark level = DMARDL+1; that is, dma_rx_req is generated when the number of valid data entries in the receive FIFO is equal to or above this field value + 1, and Receive DMA Enable(DMACR[0])=1.

SPI_TXDR

Address: Operational Base + offset (0x0048)

Transmit FIFO Data

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	WO	0x0000	TXDR Transimt FIFO Data Register. When it is written to, data are moved into the transmit FIFO.

SPI_RXDR

Address: Operational Base + offset (0x004c)

Receive FIFO Data

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0000	RXDR Receive FIFO Data Register. When the register is read, data in the receive FIFO is accessed.

10.5 Interface Description

Table 10-1 1SPI interface description

Module Pin	Direction	Pad Name	IOMUX Setting
spi_clk	I/O	IO_NANDale_SPIclk_GPIO2a0	GRF_GPIO2A_IOMUX[1:0]=2'b10
spi_rxd	I	IO_NANDd4_EMMCd4_SPIrxd_GPIO1d4	GRF_GPIO1D_IOMUX[9:8]=2'b11
spi_txd	O	IO_NANDd5_EMMCd5_SPItxd_GPIO1d5	GRF_GPIO1D_IOMUX[11:10]=2'b11
spi_csn0	I/O	IO_NANDd6_EMMCd6_SPIcsn0_GPIO1d6	GRF_GPIO1D_IOMUX[13:12]=2'b11
spi_csn1	O	IO_NANDd7_EMMCd7_SPIcsn1_GPIO1d7	GRF_GPIO1D_IOMUX[15:14]=2'b11

Notes: I=input, O=output, I/O=input/output, bidirectional. spi_csn1 can only be used in master mode

10.6 Application Notes**Clock Ratios**

A summary of the frequency ratio restrictions between the bit-rate clock (sclk_out/sclk_in) and the SPI peripheral clock (spi_clk) are described as,

When SPI Controller works as master, the $F_{spi_clk} \geq 2 \times (\text{maximum } F_{sclk_out})$ When SPI Controller works as slave, the $F_{spi_clk} \geq 6 \times (\text{maximum } F_{sclk_in})$ **Master Transfer Flow**

When configured as a serial-master device, the SPI initiates and controls all serial transfers. The serial bit-rate clock, generated and controlled by the SPI, is driven out on the sclk_out line. When the SPI is disabled (SPI_ENR = 0), no serial transfers can occur and sclk_out is held in "inactive" state, as defined by the serial protocol under which it operates.

Slave Transfer Flow

When the SPI is configured as a slave device, all serial transfers are initiated and controlled by the serial bus master.

When the SPI serial slave is selected during configuration, it enables its txd data onto the serial bus. All data transfers to and from the serial slave are regulated on the serial clock line (sclk_in), driven from the serial-master device. Data are propagated from the serial slave on one edge of the serial clock line and sampled on the opposite edge.

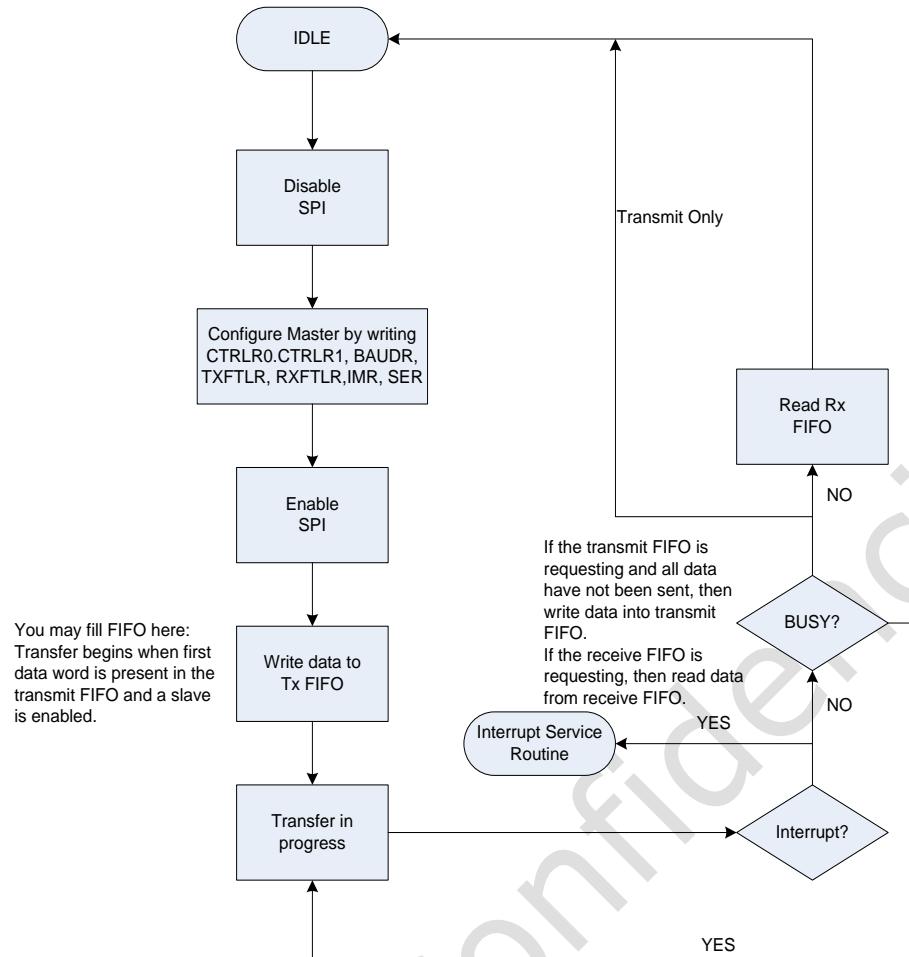


Fig. 10-7 SPI Master transfer flow diagram

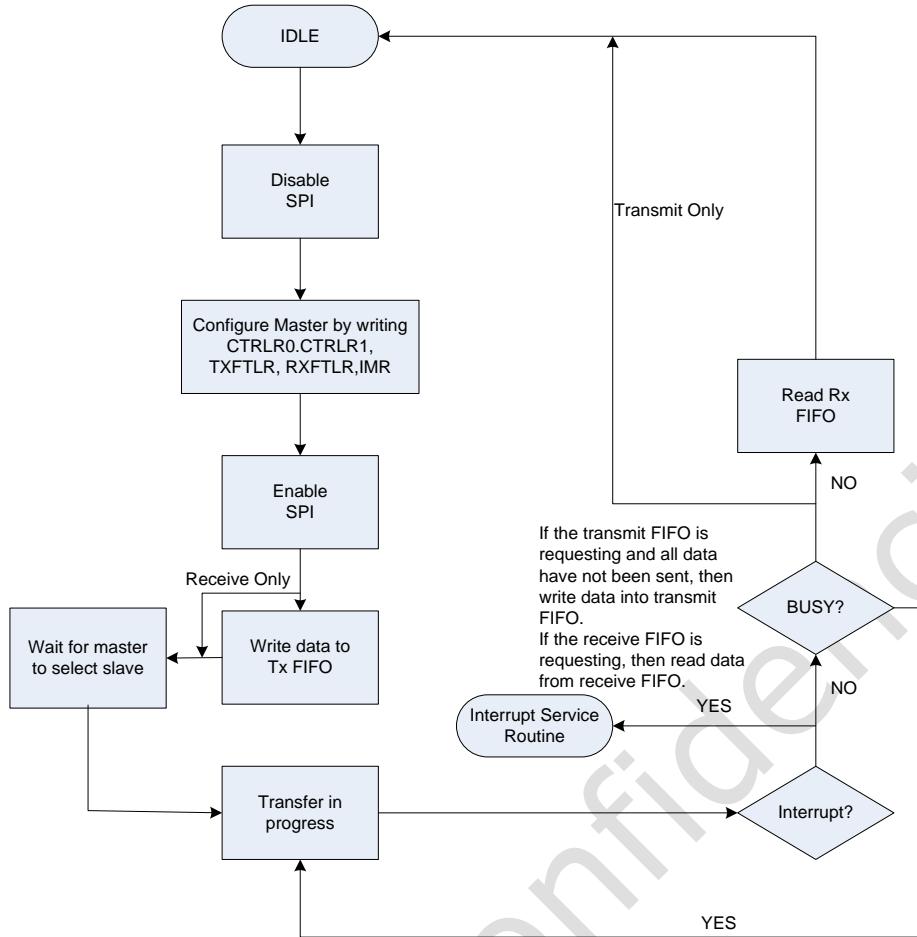


Fig. 10-8 SPI Slave transfer flow diagram

Chapter 11 SPDIF transmitter

11.1 Overview

The SPDIF transmitter is a self-clocking, serial and unidirectional interface for the interconnection of digital audio equipment in consumer and professional applications which uses linear PCM coded audio samples.

When used in professional application, the interface is primarily intended to carry monophonic or stereophonic programmes at a 48 kHz sampling frequency with a resolution of up to 24bits per sample. It may alternatively be used to carry signals sampled at 32 kHz or 44.1 kHz.

When used in consumer application, the interface is primarily intended to carry stereophonic programmes with a resolution of up to 20 bits per sample, an extension to 24 bits per sample being possible.

When used for other purposes, the interface is primarily intended to carry audio data coded other than linear PCM coded audio samples. Provision is also made to allow the interface to carry data related to computer software or signals coded using non-linear PCM. The maximum sample frequency can be up to 192 kHz for the non-linear PCM mode.

In all cases, the clock references and auxiliary information are transmitted along with the programme.

- Supports one internal 32-bit wide and 32-location deep sample data buffer
- Supports two 16-bit audio data store together in one 32-bit wide location
- Supports AHB bus interface
- Supports biphasic format stereo audio data output
- Supports DMA handshake interface and configurable DMA water level
- Supports sample data buffer empty, block terminate and user data interrupt
- Supports combine interrupt output
- Supports 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

11.2 Block Diagram

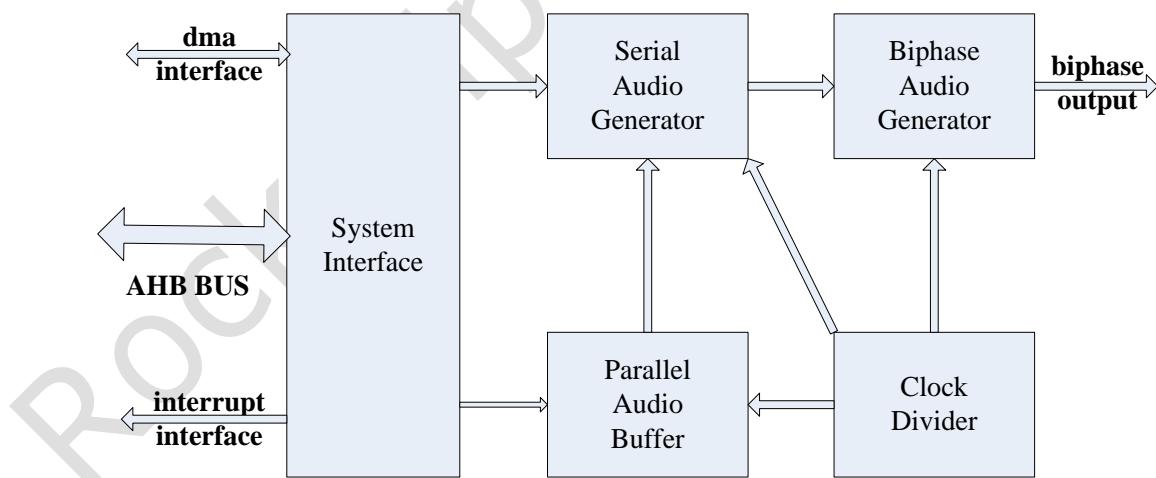


Fig. 11-1 SPDIF transmitter Block Diagram

System Interface

The system interface implements the AHB slave operation. It contains not only control registers of transmitters and receiver inside but also interrupt and DMA handshake interface.

Clock Divider

The Clock Divider implements clock generation function. The input source clock to the module is MCLK, and by the divider of the module, the clock divider generates work clock for digital audio data transformation.

Parallel Audio Buffer

The Parallel Audio Buffer is the buffer to store transmitted audio data. The size of the FIFO is

32bits x 32.

Serial Audio Converter

The Serial Audio Converter reads parallel audio data from the Parallel Audio Buffer and converts it to serial audio data.

Biphase Audio Generator

The Biphase Audio Generator reads serial audio data from the Serial Audio Converter and generates biphase audio data based on IEC-60958 standard.

11.3 Function description

11.3.1 Frame Format

A frame is uniquely composed of two sub-frames. For linear coded audio applications, the rate of transmission of frames corresponds exactly to the source sampling frequency.

In the 2-channel operation mode, the samples taken from both channels are transmitted by time multiplexing in consecutive sub-frames. The first sub-frame (left channel in stereophonic operation and primary channel in monophonic operation) normally use preamble M. However, the preamble is changed to preamble B once every 192 frame to identify the start of the block structure used to organize the channel status information. The second sub-frame (right in stereophonic operation and secondary channel in monophonic operation) always use preamble W.

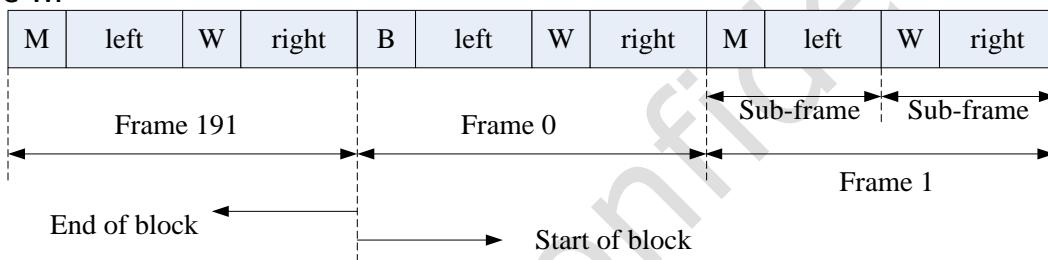


Fig. 11-2 SPDIF Frame Format

In the single channel operation mode in a professional application, the frame format is the same as in the 2-channel mode. Data is carried only in the first sub-frame and may be duplicated in the second sub-frame. If the second sub-frame is not carrying duplicate data, then time slot 28 (validity flag) shall be set to logical '1' (not valid).

11.3.2 Sub-frame Format

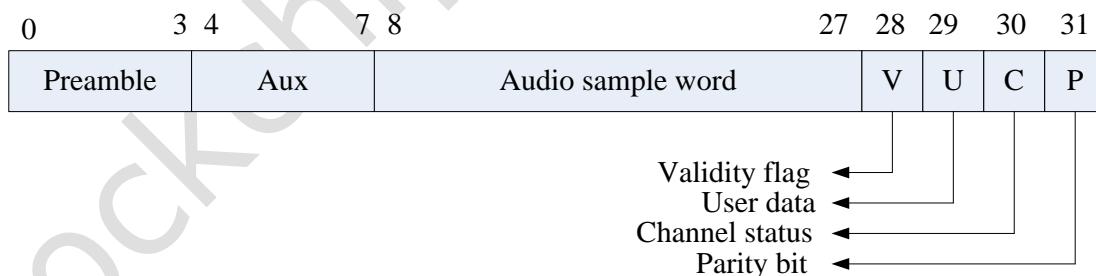


Fig. 11-3 SPDIF Sub-frame Format

Each sub-frame is divided into 32 time slots, numbered from 0 to 31. Time slot 0 to 3 carries one of the three permitted preambles. Time slot 4 to 27 carry the audio sample word in linear 2's complement representation. The MSB is carried by time slot 27. When a 24-bit coding range is used, the LSB is in time slot 4. When a 20-bit coding range is used, time slot 8 to 27 carry the audio sample word with the LSB in time slot 8. Time slot 4 to 7 may be used for other application. Under these circumstances, the bits in the time slot 4 to 7 are designated auxiliary sample bits.

If the source provides fewer bits than the interface allows (either 24 or 20), the unused LSBs are set to a logical '0'. For a non-linear PCM audio application or a data application the main data field may carry any other information. Time slot 28 carries the validity flag associated with the main data field. Time slot 29 carries 1 bit of the user data associated with the audio channel transmitted in the same sub-frame. Time slot 30 carries one bit of the channel status

words associated with the main data field channel transmitted in the same sub-frame. Time slot 31 carries a parity bit such that time slots 4 to 31 inclusive carries an even number of ones and an even number of zeros.

11.3.3 Channel Coding

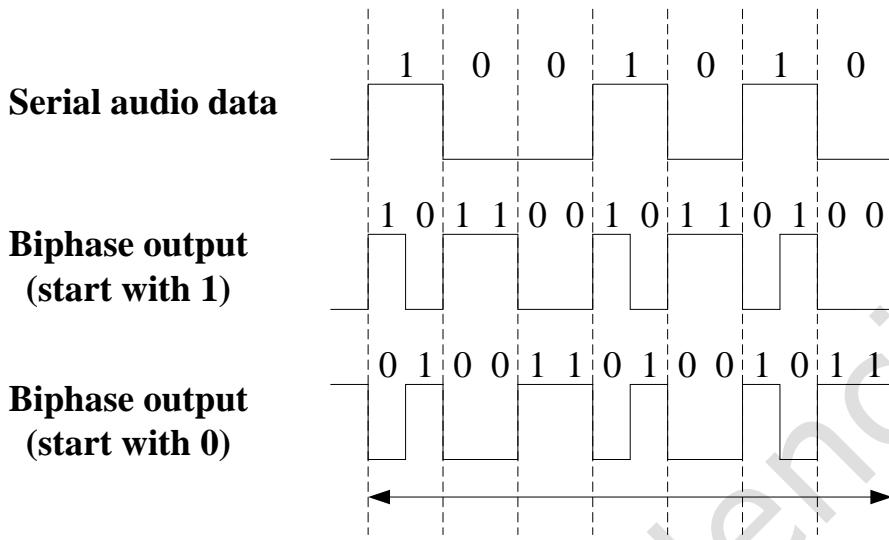


Fig. 11-4 SPDIF Channel Coding

To minimize the direct current component on the transmission line, to facilitate clock recovery from the data stream and to make the interface insensitive to the polarity of connections, time slots 4 to 31 are encoded in biphase-mark.

Each bit to be transmitted is represented by a symbol comprising two consecutive binary states. The first state of a symbol is always different from the second state of the previous symbol. The second state of the symbol is identical to the first if the bit to be transmitted is logical '0'. However, it is different from the first if the bit is logical '1'.

11.3.4 Preamble

Preambles are specific patterns providing synchronization and identification of the sub-frames and blocks.

To achieve synchronization within one sampling period and to make this process completely reliable, these patterns violate the biphase-mark code rules, thereby avoiding the possibility of data imitating the preambles.

A set of three preambles is used. These preambles are transmitted in the time allocated to four time slots (time slots 0 to 3) and are represented by eight successive states. The first state of the preamble is always different from the second state of the previous symbol.

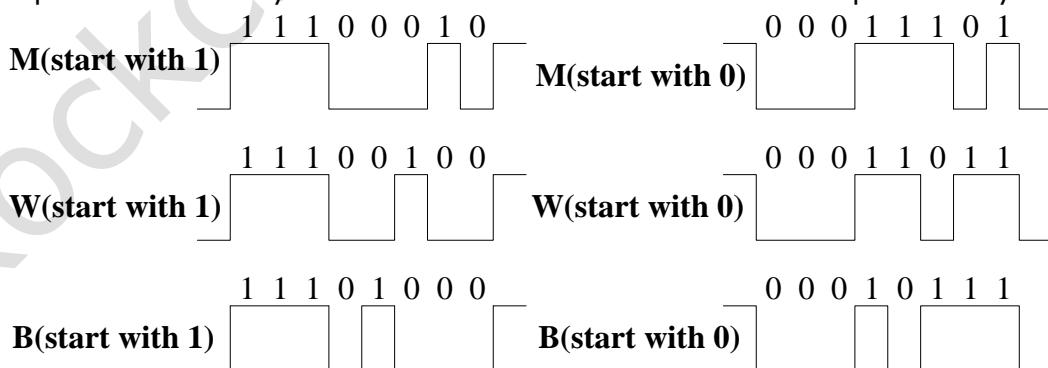


Fig. 11-5 SPDIF Preamble

Like biphase code, these preambles are dc free and provide clock recovery. They differ in at least two states from any valid biphase sequence.

11.3.5 NON-LINEAR PCM ENCODED SOURCE (IEC 61937)

The non-linear PCM encoded audio bitstream is transferred using the basic 16-bit data area of the IEC 60958 subframes, i.e. in time slots 12 to 27. Each IEC 60958 frame transfers 32-bit of the non-PCM data in consumer application mode.

If the SPDIF bitstream conveys linear PCM audio, the symbol frequency is 64 times the PCM

sampling frequency (32 time slots per PCM sample times two channels). If a non-linear PCM encoded audio bitstream is conveyed by the interface, the symbol frequency is 64 times the sampling rate of the encoded audio within that bitstream. But in the case where a non-linear PCM encoded audio bitstream is conveyed by the interface containing audio with low sampling frequency, the symbol frequency is 128 times the sampling rate of the encoded audio within that bitstream.

Each data burst contains a burst-preamble consisting of four 16-bit words (Pa, Pb, Pc, Pd), followed by the burst payload which contains data of an encoded audio frame.

The burst-preamble consists of four mandatory fields. Pa and Pb represent a synchronization word. Pc gives information about the type of data and some information/control for the receiver. Pd gives the length of the burst payload, the number of bits or number of bytes according to data-type.

The four preamble words are contained in two sequential SPDIF frames. The frame beginning the data-burst contains preamble word Pa in subframe 0 and Pb in subframe 1. The next frame contains Pc in subframe 0 and Pd in subframe 1. When placed into a SPDIF subframe, the MSB of a 16-bit burst-preamble is placed into timeslot 27 and the LSB is placed into time slot 12.

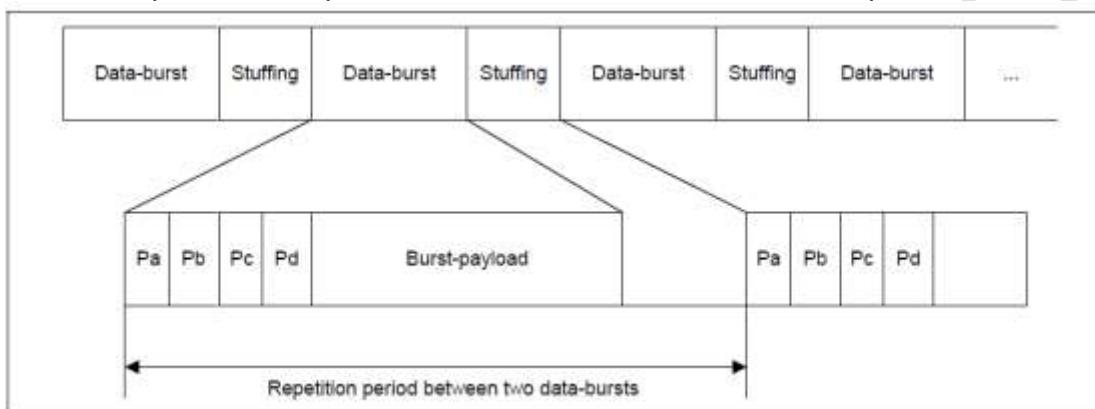


Fig. 11-6 Format of Data-burst

11.4 Register description

11.4.1 Register Summary

Name	Offset	Size	Reset Value	Description
SPDIF_CFGR	0x0000	W	0x00000000	Transfer Configuration Register
SPDIF_SDBLR	0x0004	W	0x00000000	Sample Date Buffer Level Register
SPDIF_DMACR	0x0008	W	0x00000000	DMA Control Register
SPDIF_INTCR	0x000c	W	0x00000000	Interrupt Control Register
SPDIF_INTSR	0x0010	W	0x00000000	Interrupt Status Register
SPDIF_XFER	0x0018	W	0x00000000	Transfer Start Register
SPDIF_SMPDR	0x0020	W	0x00000000	Sample Data Register
SPDIF_VLDFRN	0x0060	W	0x00000000	Validity Flag Register n
SPDIF_USRDRN	0x0090	W	0x00000000	User Data Register n
SPDIF_CHNSRN	0x00c0	W	0x00000000	Channel Status Register n
SPDIF_BURTSINFO	0x0100	W	0x00000000	Channel Burst Info Register
SPDIF_REPETITION	0x0104	W	0x00000000	Channel Repetition Register
SPDIF_BURTSINFO_SHD	0x0108	W	0x00000000	Shadow Channel Burst Info Register
SPDIF_REPETITION_SHD	0x010c	W	0x00000000	Shadow Channel Repetition Register
SPDIF_USRDR_SHDn	0x0190	W	0x00000000	Shadow User Data Register n

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

11.4.2 Detail Register Description

SPDIF_CFGR

Address: Operational Base + offset (0x0000)

Transfer Configuration Register

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved
23:16	RW	0x00	MCD mclk divider Fmclk/Fsdo This parameter can be calculated by Fmclk/(Fs*128). Fs=the sample frequency be wanted
15:9	RO	0x0	reserved
8	RW	0x0	PCMTYPE PCM type 0: linear PCM 1: non-linear PCM
7	WO	0x0	CLR mclk domain logic clear Write 1 to clear mclk domain logic. Read return zero.
6	RW	0x0	CSE Channel status enable 0: disable 1: enable The bit should be set to 1 when the channel conveys non-linear PCM
5	RW	0x0	UDE User data enable 0: disable 1: enable
4	RW	0x0	VFE Validity flag enable 0: disable 1: enable
3	RW	0x0	ADJ audio data justified 0: Right justified 1: Left justified
2	RW	0x0	HWT Halfword word transform enable 0: disable 1: enable It is valid only when the valid data width is 16bit.
1:0	RW	0x0	VDW Valid data width 00: 16bit 01: 20bit 10: 24bit 11: reserved The valid data width is 16bit only for non-linear PCM

SPDIF_SDBLR

Address: Operational Base + offset (0x0004)

Sample Date Buffer Level Register

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5:0	RW	0x00	SDBLR Sample Date Buffer Level Register Contains the number of valid data entries in the sample data buffer.

SPDIF_DMACR

Address: Operational Base + offset (0x0008)

DMA Control Register

Bit	Attr	Reset Value	Description
31:6	RO	0x0	reserved
5	RW	0x0	TDE Transmit DMA Enable 0: Transmit DMA disabled 1: Transmit DMA enabled
4:0	RW	0x00	TDL Transmit Data Level This bit field controls the level at which a DMA request is made by the transmit logic. It is equal to the watermark level; that is, the dma_tx_req signal is generated when the number of valid data entries in the Sample Date Buffer is equal to or below this field value

SPDIF_INTCR

Address: Operational Base + offset (0x000c)

Interrupt Control Register

Bit	Attr	Reset Value	Description
31:18	RO	0x0	reserved
17	W1C	0x0	UDTIC User Data Interrupt Clear Write '1' to clear the user data interrupt.
16	W1C	0x0	BTTIC Block/Data burst transfer finish interrupt clear Write 1 to clear the interrupt.
15:10	RO	0x0	reserved
9:5	RW	0x00	SDBT Sample Date Buffer Threshold Sample Date Buffer Threshold for empty interrupt
4	RW	0x0	SDBEIE Sample Date Buffer empty interrupt enable 0: disable 1: enable

Bit	Attr	Reset Value	Description
3	RW	0x0	BTTIE Block transfer/repetition period end interrupt enable When enabled, an interrupt will be asserted when the block transfer is finished if the channel conveys linear PCM or when the repetition period is reached if the channel conveys non-linear PCM. 0: disable 1: enable
2	RW	0x0	UDTIE User Data Interrupt 0: disable 1: enable If enabled, an interrupt will be asserted when the content of the user data register is fed into the corresponding shadow register
1:0	RO	0x0	reserved

SPDIF_INTSR

Address: Operational Base + offset (0x0010)

Interrupt Status Register

Bit	Attr	Reset Value	Description
31:5	RO	0x0	reserved
4	RW	0x0	SDBEIS Sample Date Buffer empty interrupt status 0: inactive 1: active
3	RW	0x0	BTTIS Block/Data burst transfer interrupt status 0: inactive 1: active
2	RW	0x0	UDTIS User Data Interrupt Status 0: inactive 1: active
1:0	RO	0x0	reserved

SPDIF_XFER

Address: Operational Base + offset (0x0018)

Transfer Start Register

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	XFER Transfer Start Register Transfer Start Register

SPDIF_SMPDR

Address: Operational Base + offset (0x0020)

Sample Data Register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	SMPDR Sample Data Register Sample Data Register

SPDIF_VLDFRn

Address: Operational Base + offset (0x0060)

Validity Flag Register n

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	VLDFR_SUB_1 Validity Flag Subframe 1 Validity Flag Register 0
15:0	RW	0x0000	VLDFR_SUB_0 Validity Flag Subframe 0 Validity Flag for Subframe 0

SPDIF_USRDRn

Address: Operational Base + offset (0x0090)

User Data Register n

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	USR_SUB_1 User Data Subframe 1 User Data Bit for Subframe 1
15:0	RW	0x0000	USR_SUB_0 User Data Subframe 0 User Data Bit for Subframe 0

SPDIF_CHNSRn

Address: Operational Base + offset (0x00c0)

Channel Status Register n

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	CHNSR_SUB_1 Channel Status Subframe 1 Channel Status Bit for Subframe 1
15:0	RW	0x0000	CHNSR_SUB_0 Channel Status Subframe 0 Channel Status Bit for Subframe 0

SPDIF_BURTSINFO

Address: Operational Base + offset (0x00d0)

Channel Burst Info Register

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

Bit	Attr	Reset Value	Description
31:16	RW	0x0000	PD pd Preamble Pd for non-linear pcm, indicating the length of burst payload in unit of bytes or bits.
15:13	RW	0x0	BSNUM Bitstream Number This field indicates the bitstream number. Usually the birstream number is 0.
12:8	RW	0x00	DATAINFO Data-type-dependent info This field gives the data-type-dependent info
7	RW	0x0	ERRFLAG Error Flag 0: indicates a valid burst-payload 1: indicates that the burst-payload may contain errors

11.5 Interface description

Table 11-1 SPDIF Interface Description

Module Pin	Direction	Pad Name	IOMUX Setting
spdif_8ch_sdo	O	SPDIF_SDO	GRF_GPIO0D_IOMUX[8]=1'b1

The output of SPDIF module which signals as spdif_8ch_sdo is also connected to the audio interface of HDMI.

11.6 Application Notes

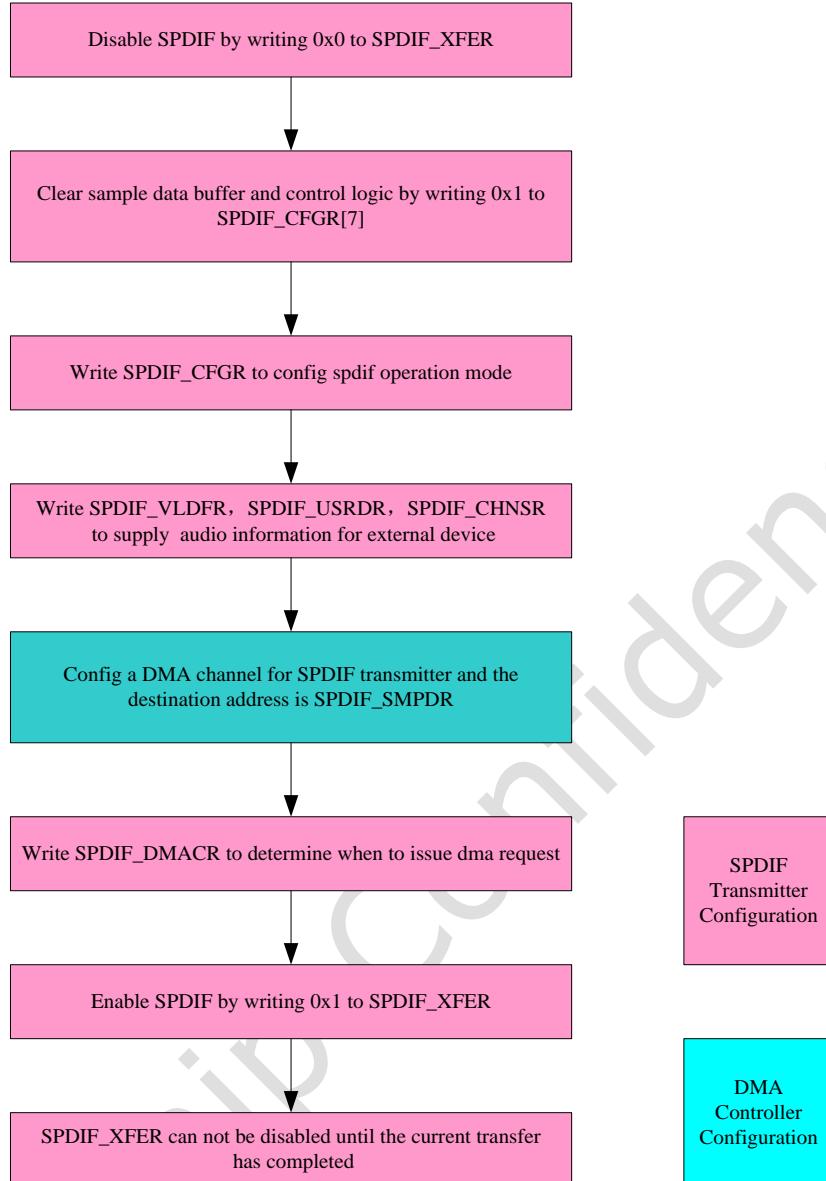


Fig. 11-7 SPDIF transmitter operation flow chart

11.6.1 Channel Status Bit and Validity Flag Bit

Normally the channel status bits and validity flag bits are not necessarily updated frequently. If it is desired to change the channel status bits or validity flag, please write to the corresponding register after a block termination interrupt is asserted. The new value will take effect immediately.

11.6.2 User Data Bit

As the user data bits are updated frequently, the design takes use of the shadow register mechanism to store and convey the user data bit. When the SPDIF interface is disabled, the values of the shadow user data registers keeps the same with the corresponding user data registers. After the SPDIF starts, any change of the user data register will not go to the corresponding shadow user data registers until an user data interrupt is asserted.

Therefore before the SPDIF transfer starts, prepare the first 384 user data bits by writing them to the SPDIF_USRDR registers. After the SPDIF transfer starts, writing the second 384 user data bits to the SPDIF_USRDR registers. Then wait for the assertion of user data interrupt. The second 384 user data bits goes to the shadow registers, and then third 384 user bits are written to SPDIF_USRDR.

11.6.3 Burst Info and Repetition

The shadow register mechanism is also applied to the data of burst info and repetition as the user data. The difference is that the update of shadow register will be taken after assertion of the block termination interrupt.

It is important to note that the repetition defined in the design is a little different from the repetition defined in IEC-61957. The repetition is always defined as the length (measured in IEC-60958 frame) between Pa of two consecutive data-bursts. Therefore the user needs to calculate the new repetition value if the definition of the repetition is different for some audio formats such as AC-3.

Rockchip Confidential

Chapter 12 MAC Ethernet Interface

12.1 Overview

The VMAC Ethernet Controller provides a complete Ethernet interface from processor to a Reduced Media Independent Interface (RMII) compliant Ethernet PHY.

The VMAC includes a DMAC controller. The DMAC controller efficiently moves packet data from microprocessor's RAM, format the data for an IEEE 802.3 compliant packet and transmit the data to an Ethernet Physical Interface (PHY). It also efficiently moves packet data from RXFIFO to microprocessor's RAM.

It supports the following features:

- IEEE 802.3u compliant Ethernet Media Access Controller
- 10Mbps and 100Mbps compatible
- Automatic retry and automatic collision frame deletion
- Reduced Media Independent Interface (RMII) for PHY connection
- Management Interface (MDIO) state machine for easy real-time communication with the PHY
- Full Duplex Support
- Pause full-duplex flow-control support
- Address filtering – Broadcast/Multicast/Logic/Physical
- Complete DMA buffer management controller for minimal processor overhead
- Wake-On-LAN low-power mode support
- AHB interface to any CPU or memory

12.2 Block Diagram

12.2.1 Architecture

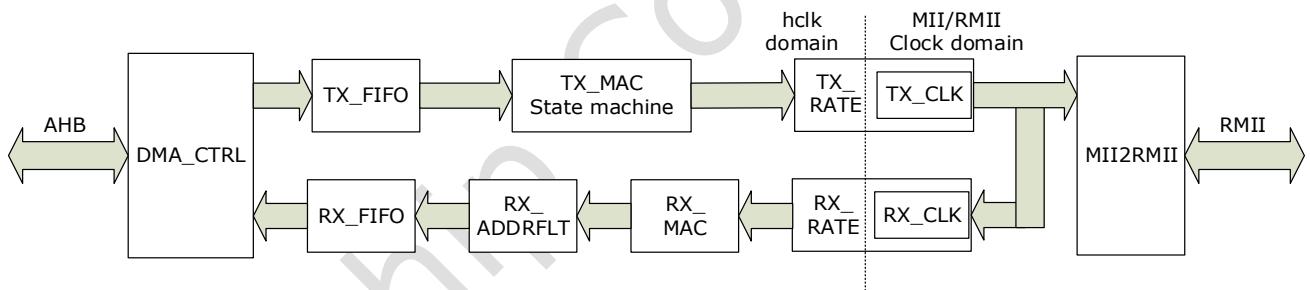


Fig. 12-1 VMAC Block Diagram

The VMAC is broken up into nine separate functional units. These nine blocks are interconnected in the VMAC module. The block diagram shows the general flow of data and control signals between these blocks.

The DMA_CTRL controller moves data between system memory and the respective TX or RX pipelines and manages the buffer descriptors. A single state machine handles the Buffer Descriptors as well as both the RX and TX DMAs. Transmit and Receive DMAs may be interleaved as well as polling of the buffer descriptor rings. The DMA controller also includes all of the configuration, control and status registers of the VMAC as well as the MDIO state machine for communicating to the PHY.

The transmit FIFO and its controller are integrated in the TX_FIFO block. The FIFO is typically built of a 512x8bit dual-port RAM.

The TX_MAC provides all of the logic necessary to build and transmit a frame that meets the IEEE 802.3 Ethernet LAN standard. When a start of frame is detected on the data/status bus, then the TX_MAC block starts to transmit preamble data. TX_MAC delays the transmission if the receive path is active in Half-Duplex mode and ensures that the interframe period is met. In Full-Duplex mode only the interframe period is observed and no deferring process takes place.

The TX_RATE block transfers the raw packet data from the HCLK to the CLK_TX clock domains. In addition to its main function, the TX_RATE block monitors the collision signal COL from the PHY and the preamble field from TX_RATE. In case of a collision, the TX_RATE jams the data

on TXD bus and notifies TX_MAC about the collision.

The RX_FIFO block is composed with a 1024x9 bit dual-port ram.

The RX_ADDRFLT block determines if the destination address matches under any of the currently-active addressing modes. While the Destination Address field is being verified, the incoming bytes are stored in a small FIFO. If the Destination address does not match, then the RX_ADDRFLT resets its FIFO and disregards incoming data.

The RX_MAC block provides all of the logic necessary to meet the IEEE 802.3 Ethernet LAN standard for frame reception. The RX_MAC detects the SFD pattern, verifies FCS field, senses framing errors (odd number of nibbles) and monitors the RX_ER signal, which indicates any other errors received from an external PHY.

The RX_RATE block synchronizes signals from MII CLK_RX domain to hclk domain.

The MII2RMII block transfers MII signals to RMII signals.

12.2.2 Frame Structure

Data frames transmitted shall have the frame format shown in Fig. 25-2.



Fig. 12-2 VMAC Block Diagram

The preamble <preamble> begins a frame transmission. The bit value of the preamble field consists of 7 octets with the following bit values:

10101010 10101010 10101010 10101010 10101010 10101010 10101010

The SFD (start frame delimiter) <sfd> indicates the start of a frame and follows the preamble.

The bit value is 10101011.

The data in a well formed frame shall consist of N octet's data.

12.2.3 RMII Interface timing diagram

1. Transmission diagram

Following figure shows the 100Mb/s Transmission diagram. The REF_CLK frequency is 50MHz in RMII interface.

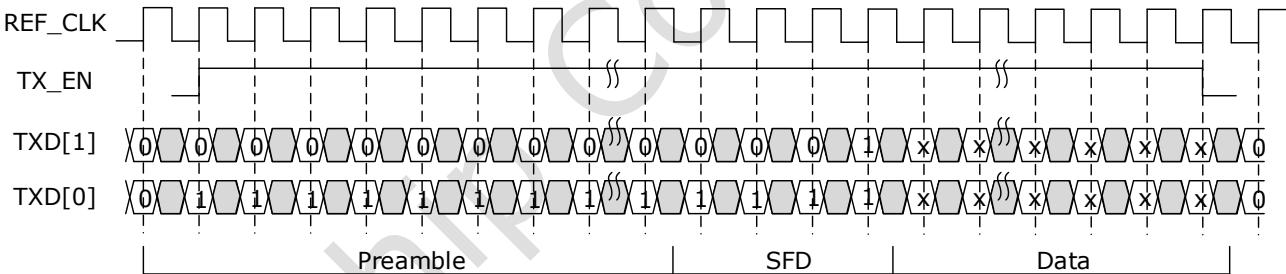


Fig. 12-3 RMII transmission in 100Mb/s mode

In 10Mb/s mode, as the REF_CLK frequency is 10 times as the data rate, the value on TXD[1:0] shall be valid such that TXD[1:0] may be sampled every 10th cycle, regard-less of the starting cycle within the group and yield the correct frame data.

2. Reception diagram

Following figure shows the 100Mb/s reception diagram. The REF_CLK frequency is 50MHz in RMII interface.

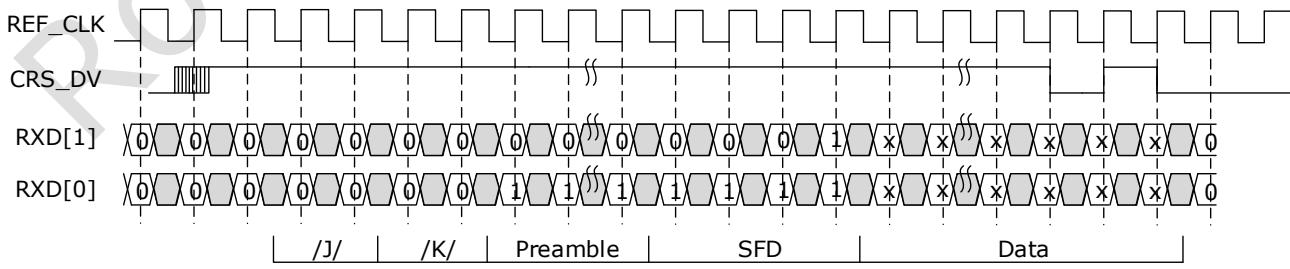


Fig. 12-4 RMII reception with no errors in 100Mb/s mode

In 10Mb/s mode, as the REF_CLK frequency is 10 times the data rate, the value on RXD[1:0] shall be valid such that RXD[1:0] may be sampled every 10th cycle, regardless of the starting cycle within the group and yield the correct frame data.

12.2.4 Management Interface

The MII management interface provides a simple, two-wire, serial interface to connect the

VMAC and a managed PHY, for the purposes of controlling the PHY and gathering status from the PHY. The management interface consists of a pair of signals that transport the management information across the MII bus: MDIO and MDC.

12.3 Register Description

12.3.1 Registers Summary

Name	Offset	Size	Reset Value	Description
EMAC_ID	0x0000	W	0x00053d02	Hardware Version
EMAC_STAT	0x0004	W	0x00000000	Interrupt status register
EMAC_ENABLE	0x0008	W	0x00000000	Interrupt enable register
EMAC_CONTROL	0x000c	W	0x00000000	CONTROL Register
EMAC_POLLRATE	0x0010	W	0x00000000	Poll Rate register
EMAC_RXERR	0x0014	W	0x00000000	Receive Error Counters
EMAC_MISS	0x0018	W	0x00000000	Missed Packet Counter
EMAC_TXRINGPTR	0x001c	W	0x00000000	Transmit Ring Pointer address register
EMAC_RXRINGPTR	0x0020	W	0x00000000	Receive Ring Pointer address register
EMAC_ADDR	0x0024	W	0x00000000	Ethernet MAC Address, low 32 bits
EMAC_ADDRH	0x0028	W	0x00000000	Ethernet MAC Address, high 16 bits
EMAC_LAFL	0x002c	W	0x00000000	Logical Address filter Register Low
EMAC_LAFH	0x0030	W	0x00000000	Logical Address filter Register High
EMAC_MDIO_DATA	0x0034	W	0x00000000	MDIO access register
EMAC_TXRINGPTR_READ	0x0038	W	0x00000000	Transmit Ring Pointer read-back register
EMAC_RXRINGPTR_READ	0x003c	W	0x00000000	Receive Ring Pointer read-back register
EMAC_XTRA_CON	0x0040	W	0x00000000	Extra control register
EMAC_XTRA_STATUS	0x0044	W	0x00000000	Extra status register

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

12.3.2 Detail Register Description

EMAC_ID

Address: Operational Base + offset (0x0000)

Hardware Version

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved
23:16	RO	0x05	Revision Revision number (currently Rev 5).
15:14	RO	0x0	reserved
13:8	RO	0x3d	NOTID Ones-Complement of the ID, (6'b111101).
7:6	RO	0x0	reserved
5:0	RO	0x02	ID Identification number, always 2 for VMAC.

EMAC_STAT

Address: Operational Base + offset (0x0004)

Interrupt status register

Bit	Attr	Reset Value	Description
31	RW	0x0	<p>TXPOLL Writing a one forces a poll of the transmit descriptors. Always write this bit with a one after adding a packet to the transmit BDT. Always read as zero.</p>
30:13	RO	0x0	reserved
12	RW	0x0	<p>MDIO MDIO Complete. The register access to the PHY has completed.</p>
11	RO	0x0	reserved
10	RW	0x0	<p>RXFL The RXOFLVERR counter has rolled over.</p>
9	RW	0x0	<p>RXFRAME The RXFRAMEERR counter has rolled over.</p>
8	RW	0x0	<p>RXCRC The RXCRCERR counter has rolled over.</p>
7:5	RO	0x0	reserved
4	RW	0x0	<p>MISSERR Missed packet counter has rolled over.</p>
3	RW	0x0	<p>TXCH TX Chaining Error A bad combination of FIRST and LAST bits has been encountered. The VMAC asserts this error bit and disables the TXRN bit in the CONTROL register.</p>
2	RO	0x0	<p>ERR Error interrupt pending It is the logical OR of all of the other error bits in the status register (all interrupts except TXINT, RXINT and MDIO). This is a read-only bit.</p>
1	RW	0x0	<p>RXINT Receive Interrupt Pending RXINT is set when the VMAC clears the OWN bit of an RX BDT and LAST=1. Note that multiple RX BDTs could be cleared before the RXINT interrupt is serviced. Thus ALL RX BDTs must be processed within the interrupt. This also results in the possibility that there will be a RXINT without any corresponding BDTs with their OWN bits cleared. This is normal and the software driver must handle this case.</p>

Bit	Attr	Reset Value	Description
0	RW	0x0	<p>TXINT Transmit Interrupt Pending</p> <p>TXINT is set when the OWN bit of a TX BDT is cleared to 0 by the VMAC and LAST=1. Note that multiple TX BDTs could be cleared before the TXINT interrupt is serviced. Thus ALL TX BDTs must be processed within the interrupt. This also results in the possibility that there will be a TXINT without any corresponding BDTs with their OWN bits cleared. This is normal and the software driver must handle this case.</p>

EMAC_ENABLE

Address: Operational Base + offset (0x0008)

Interrupt enable register

Bit	Attr	Reset Value	Description
31	RW	0x0	<p>TXPL TXPOLL - Undefined.</p>
30:13	RO	0x0	reserved
12	RW	0x0	<p>MDIO MDIO Complete enable.</p>
11	RO	0x0	reserved
10	RW	0x0	<p>RXFL RXOFLWERR counter rolled over error enable.</p>
9	RW	0x0	<p>RXFR RXFRAMEERR counter rolled over error enable.</p>
8	RW	0x0	<p>RXCR RXFRAMEERR counter rolled over error enable.</p>
7:5	RO	0x0	reserved
4	RW	0x0	<p>MSER Missed packet counter error enable.</p>
3	RW	0x0	<p>TXCH TX Chaining Error enable.</p>
2	RW	0x0	<p>ERR Error Interrupt Pending Enable.</p>
1	RW	0x0	<p>RXINT Receive interrupt pending enable.</p>
0	RW	0x0	<p>TXINT Transmit interrupt pending enable.</p>

EMAC_CONTROL

Address: Operational Base + offset (0x000c)

CONTROL Register

Bit	Attr	Reset Value	Description
31:24	RW	0x00	<p>RXBDTLEN</p> <p>Number of BDTs in the RX Ring. 1-255 allowed.</p>

Bit	Attr	Reset Value	Description
23:16	RW	0x00	TXBDTLEN Number of BDTs in the TX Ring. 1-255 allowed.
15	RW	0x0	DISADDRCRC 1'b1: disable adding the 4byte CRC(FCS) on every packet. Instead, use the ADDCRC bit in the INFO word of the Transmit Buffer descriptor to add the FCS on a packet by packet basis. 1'b0: always add CRC and ignore the ADDCRC bit in the info field.
14	RW	0x0	DISRETRY 1'b1: disable retries, tx will be attempted only once.
13	RW	0x0	TEST Used for silicon testing -always set to zero.
12	RW	0x0	DISABLE2PART 1'b1: Disable two part deferral. Disabling 2-part deferral disables a fast Inter-packet Gap time as described in the 802.3 specification. See ANSI/IEEE Std802.3-1993 Edition, 4.2.3.2.1.
11	RW	0x0	PROM 1'b0: normal mode. 1'b1: promiscuous mode = accepts all packets.
10	RW	0x0	ENBFULL 1'b1: enable full duplex mode. This bit needs to be set to the corresponding duplex mode of the PHY chip the VMAC is connected to. The duplex mode of the PHY needs to be polled periodically to keep the VMAC duplex setting in line with the PHY.
9	RO	0x0	reserved
8	RW	0x0	DISBDCST 1'b1: disable receive broadcast packets.
7:5	RO	0x0	reserved
4	RW	0x0	RXRUN 1'b0: disable receive operation. If a packet is currently being received, it will complete before the receive operation is disabled. Like TXRN, RXRN can be used to safely disable packet reception.
3	RW	0x0	TXRUN 1'b0: disable transmit operation. This bit is only tested when the TXBDT state machine completes the current transmit operation and just before it checks the next BDT for new packet. This bit can be used to safely stop transmitting packets without affecting packet reception. A TX packet that is already in process will complete before TXRN is recognized.
2:1	RO	0x0	reserved
0	RW	0x0	ENABLE 1'b0: stop all activity. 1'b1: enable ethernet traffic. All registers should be initialized before setting this bit. Clearing this bit to zero resets BDT rings to their first BD in the table.

EMAC_POLLRATE

Address: Operational Base + offset (0x0010)

Poll Rate register

Bit	Attr	Reset Value	Description
31:15	RO	0x0	reserved
14:0	RW	0x0000	POLLRATE The value programmed into this register is the number of clocks between polls times 1024. A value of 1 will cause a poll every 1024 clocks. 2=2048 clocks. 0 is not valid. A CPU clock frequency of 100MHz would typically want to program the POLLRATE register with a value of 100 which will cause a poll to occur about once every millisecond (10ns * 1024 * 100 = 1ms).

EMAC_RXERR

Address: Operational Base + offset (0x0014)

Receive Error Counters

Bit	Attr	Reset Value	Description
31:24	RO	0x0	reserved
23:16	RW	0x00	RXOFLOW Overflow Errors Number of receive packets dropped due to FIFO overflows.
15:8	RW	0x00	RXFRAM RXFRAME Errors Number of receive packets dropped due to framing errors.
7:0	RW	0x00	RXCRC CRC Errors Number of receive packets dropped due to CRC errors.

EMAC_MISS

Address: Operational Base + offset (0x0018)

Missed Packet Counter

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	RW	0x00	MISSCNTR Missed packet counter When the counter counts up to the maximum value, it sets the MISSERR bit in the INTER register. This counter counts the number of packets that were dropped because a BD was not available. This counter is auto-zeroed when read.

EMAC_TXRINGPTR

Address: Operational Base + offset (0x001c)

Transmit Ring Pointer address register

Bit	Attr	Reset Value	Description

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	TXRINGPTR Address of the start of the Transmit RING of Buffer Descriptors. Must be on 8-byte boundaries. (ID: Bits 2-0 must be zero).

EMAC_RXRINGPTR

Address: Operational Base + offset (0x0020)

Receive Ring Pointer address register

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	RXRINGPTR Address of the start of the Receive RING of BDs. Must be on 8-byte boundaries (IE: Bits 2-0 must be zero).

EMAC_ADDRL

Address: Operational Base + offset (0x0024)

Ethernet MAC Address, low 32 bits

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	ADDRL Lower 32 bits of the ethernet MAC address The address is little-endian. Thus the first byte transferred on the Ethernet wire must match bits 7:0. The second byte transferred is in bits 15:8 and so on. Thus for a Physical address transmitted in byte order of: 8'h00: 1st byte transmitted 8'h11: 2nd byte transmitted 8'h22: 3rd byte transmitted 8'h33: 4th byte transmitted 8'h44: 5th byte transmitted 8'h55: 6th byte transmitted The ADDRL register should be programmed with 0x33221100. Bit 0 is the multi-cast / Broadcast bit of the address. Bit 0 MUST be programmed with a zero.

EMAC_ADDRH

Address: Operational Base + offset (0x0028)

Ethernet MAC Address, high 16 bits

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved
15:0	RW	0x0000	ADDRH Upper 16 bits of the ethernet MAC address. See ADDRL for more details. In the example given in ADDRL, this register would be programmed with 0x5544.

EMAC_LAFL

Address: Operational Base + offset (0x002c)

Logical Address filter Register Low

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	LAFL Low 32 bits for the Logical Address Filter. Each bit corresponds to a hash function of the destination address of a packet. If the bit is set to a 1, then the packet is accepted, otherwise it is filtered out.

EMAC_LAFH

Address: Operational Base + offset (0x0030)

Logical Address filter Register High

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	LAFH High 32 bits for the Logical Address Filter. Each bit corresponds to a hash function of the destination address of a packet. If the bit is set to a 1, then the packet is accepted, otherwise it is filtered out.

EMAC_MDIO_DATA

Address: Operational Base + offset (0x0034)

MDIO access register

Bit	Attr	Reset Value	Description
31:30	RW	0x0	SFD Start of Frame Delimiter, must be set to 2'b01.
29:28	RW	0x0	OP Operation Code, set to 2'b10 for a read and 2'b01 for a write.
27:23	RW	0x00	PHY PHY address (0-31).
22:18	RW	0x00	REG Register to access (0-31).
17:16	RW	0x0	TA Bus Turn-Around, must be set to 2'b10.
15:0	RW	0x0000	DATA DATA to be written to or read from the PHY register.

EMAC_TXRINGPTR_READ

Address: Operational Base + offset (0x0038)

Transmit Ring Pointer read-back register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	TXRINGPTR Transmit Ring Pointer read-back register This read-only register gives the address of the current Transmit Buffer Descriptor being polled or processed by the VMAC. This allows the software to determine where in the BDT ring the VMAC is currently processing. Note that this is a hardware register and undergoes rapid changes. The value in this register should ONLY be read when TXRN is cleared to zero so that the VMAC will stop processing buffers.

EMAC_RXRINGPTR_READ

Address: Operational Base + offset (0x003c)

Receive Ring Pointer read-back register

Bit	Attr	Reset Value	Description
31:0	RO	0x00000000	RXRINGPTR Receive Ring Pointer read-back register This read-only register gives the address of the current Receive Buffer Descriptor being polled or processed by the VMAC. This allows the software to determine where in the BDT ring the VMAC is currently processing buffers.

EMAC_XTRA_CON

Address: Operational Base + offset (0x0040)

Extra control register

Bit	Attr	Reset Value	Description
31:27	RO	0x0	reserved
26	RW	0x0	lockup_RST 1'b1: reset rxbf_unload_lockout.
25	RW	0x0	lockup_drain_enb 1'b1: Drains RX FIFO to next SOP if EOP-DATA sequence seen.
24	RW	0x0	lockup_missbuf_dis 1'b1: Decouples rx_missed_inc from any RX_FIFO effect.
23	RW	0x0	lockup_fix_enb Detects lockup, issues rx_done, sets rx_lockerr info field.
22	RW	0x0	xtractl_en Only when xtractl_en is 1, the XTRA_CONTROL[31:22] and XTRA_STATUS can be read.
21	RW	0x0	pause_dis 1'b1: disable pause frame reception.
20	RW	0x0	paddr_dis 1'b1: disable pad for RX management & broadcast.
19	RW	0x0	bvci_fix 1'b0: old bvci design. 1'b1: new bvci design.
18	RW	0x0	txc_apad 1'b1: add pad, will pad out packet to be 64 bytes if less than 64 bytes long.
17	RW	0x0	rx_burst8 1'b0: AHB master use burst 4 to write rxdata to system memory. 1'b1: AHB master use burst 8 to write rxdata to system memory.
16	RW	0x0	tx_burst8 1'b0: AHB master use burst 4 to read txdata from system memory. 1'b1: AHB master use burst 8 to read txdata from system memory.
15:10	RO	0x0	reserved
9:0	RW	0x000	txf_threshold Tx FIFO threshold.

EMAC_XTRA_STATUS

Address: Operational Base + offset (0x0044)

Extra status register

Bit	Attr	Reset Value	Description
31:24	RO	0x00	rxbf_data rbxf data
23	RO	0x0	rxbf_tag rbxf tag
22	RO	0x0	rxbf_valid rbxf valid
21:20	RO	0x0	reserved
19	RO	0x0	rx_bufflen_pre0 Status of rx_bufflen_pre[0].
18	RO	0x0	rx_bufflen_pre_zero 1'b1: rx_bufflen_pre = 0.
17	RO	0x0	rx_fifo0_valid_zero 1'b1: rx_fifo0_valid = 0.
16	RO	0x0	rx_byte_ctr_zero 1'b1: rx_byte_ctr = 0.
15	RO	0x0	rx_fifo_empty 1'b1: RX fifo empty.
14	RO	0x0	rx_own rx own
13	RO	0x0	rx_bufferr rx bufferr
12	RO	0x0	rx_last rx last
11	RO	0x0	rx_first rx first
10	RO	0x0	burst_rspval_done Burst read done.
9	RO	0x0	rxbf_unload_lockout At the end of a packet, do not unload any more data from the rxbf buffer (even the start signal) until the current packet descriptor has finished being processed and the next descriptor is loaded this prevents the packet being loaded before the new descriptor is loaded, which will corrupt the byte count in the info field.
8	RO	0x0	rx_start_pending Start tag is at the output of buffer, but unload signal not active.
7	RO	0x0	rx_info_poll Active when we are loading the INFO value from the Buffer Descriptor.

Bit	Attr	Reset Value	Description
6	RO	0x0	rx_dma_inprogress Indicates that a receive DMA is currently operating set when the RX_MAC says we start a packet and cleared when the last of the RX data has been transferred.
5	RO	0x0	rx_recycle_desc Recycle RX descriptor if the packet ends but has errors or the packet is a runt (less than 64 bytes).
4:0	RO	0x00	state State machine status of dma_ctrlr.

12.4 Interface Description

Table 12-1 IOMUX Settings for RMII/MII Interface

Module Pin	Direction	Pad Name	IOMUX Setting
RMII Interface			
rmii_clkout	O	IO_MACClkout_MACClkin_GPIO2b6	GRF_GPIO2B_IOMUX[13:12]=2'b01
rmii_clkin	I	IO_MACClkout_MACClkin_GPIO2b6	GRF_GPIO2B_IOMUX[13:12]=2'b10
rmii_tx_en	O	IO_MACTxen_GPIO2b5	GRF_GPIO2B_IOMUX[11:10]=2'b01
rmii_txd1	O	IO_MACtxd1_GPIO2c2	GRF_GPIO2C_IOMUX[5:4]=2'b01
rmii_txd0	O	IO_MACtxd0_GPIO2c3	GRF_GPIO2C_IOMUX[7:6]=2'b01
rmii_rx_err	I	IO_MACRxer_GPIO2b7	GRF_GPIO2B_IOMUX[15:14]=2'b01
rmii_crs_dvalid	I	IO_MACcrs_GPIO2b2	GRF_GPIO2B_IOMUX[5:4]=2'b01
rmii_rxd1	I	IO_MACRxd1_GPIO2c0	GRF_GPIO2C_IOMUX[1:0]=2'b01
rmii_rxd0	I	IO_MACRxd0_GPIO2c1	GRF_GPIO2C_IOMUX[3:2]=2'b01
Management Interface			
mii_md	I/O	IO_MACmdio_GPIO2b4	GRF_GPIO2B_IOMUX[9:8]=2'b01
mii_mdclk	O	IO_MACmdc_GPIO2d1	GRF_GPIO2D_IOMUX[3:2]=2'b01

Notes: I=input, O=output, I/O=input/output, bidirectional

12.5 Application Notes

12.5.1 Buffer Descriptors

Data is sent and received via buffers that are built in system memory. These buffers are pointed to by the Buffer Descriptor Rings that are also located in memory. There are two Buffer Descriptor Rings: the transmit (TX) and receive (RX) rings. The Transmit Ring Pointer (TXRINGPTR) is a pointer to the start of the Transmit Descriptor Ring. The RXRINGPTR register similarly is a pointer to the start of the Receive Descriptor Ring. The VMAC processes each descriptor in each ring sequentially until it reaches the last descriptor. Then the VMAC will automatically cycle back to the first descriptor in the ring.

Each Buffer Descriptor Ring in turn consists of a number of Buffer Descriptors. The Buffer Descriptor Rings must occupy a contiguous area of memory and must be aligned on 8-byte boundaries. The number of Buffer Descriptors in the Ring is programmed by fields in the CONTROL register.

Each Buffer Descriptor (BD) within the Ring in turn points to a buffer in memory that contains the packet data. Each Buffer Descriptor consists of eight bytes of data formatted as two 32-bit words. The first word (INFO) contains various status information on the buffer itself. The most important field is the OWN bit which indicates whether the VMAC "owns" the buffer, or the processor "owns" the buffer. The OWN bit is a semaphore that indicates who is allowed access to the buffer and the buffer descriptor. If the VMAC owns the buffer, the processor must not make any changes to either the buffer descriptor or the contents of the buffer. If the processor owns the buffer then the VMAC will ignore the buffer and wait for the processor to release it. Once a buffer and the Buffer Descriptor have been prepared, the processor toggles the OWN bit and releases the buffer to the VMAC. The second word of the BD is the PTR field which is a 32-bit byte addressable pointer to the packet data. The data can start on any byte boundary and need not be aligned. Any number of bytes of data can be contained in the buffer, every zero bytes. The length of the buffer is defined in the LENGTH field of the INFO word. A buffer

typically contains all of the bytes of a single packet.

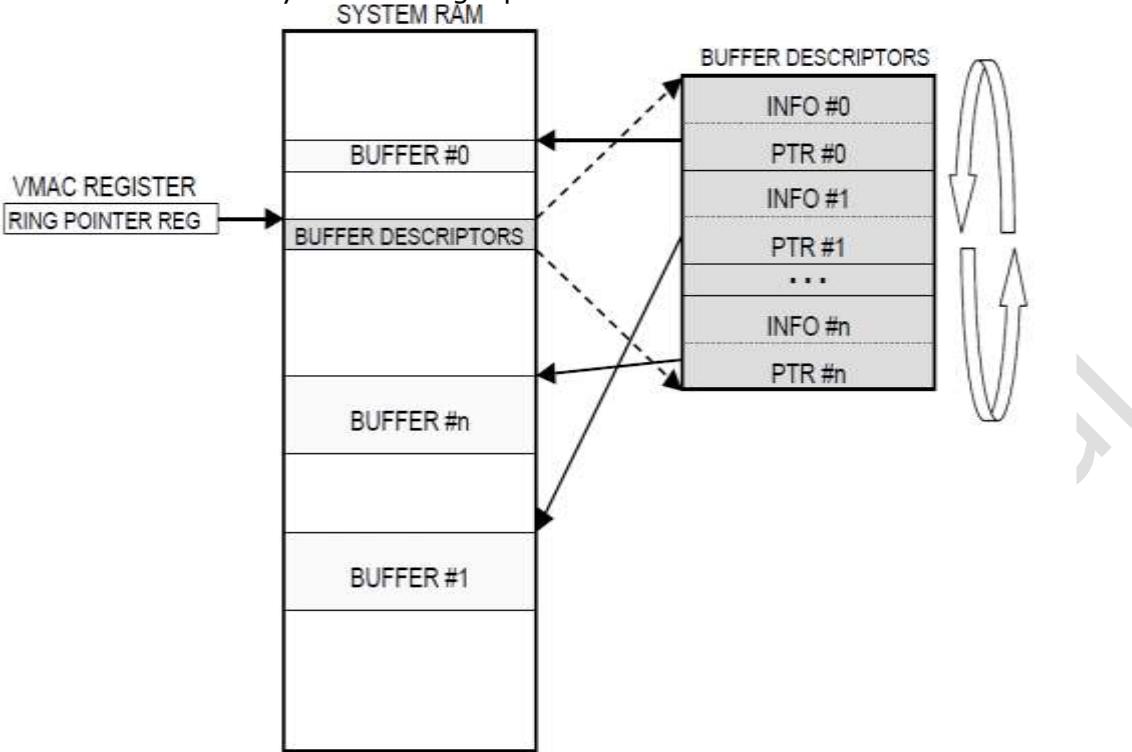


Fig. 12-5 VMAC buffer chain

Previous figure shows the relationship of the VMAC registers and the Buffer Descriptors. The RINGPTR register is shown on the left which points to the starting address of the Buffer Descriptor Ring (either TX or RX) in system memory. The Buffer Descriptor Ring takes up a very small amount of the system memory and is expanded on the right of the diagram. The Buffer Descriptor Ring has a number of Buffer Descriptors in it each containing the two 32-bit words, INFO and PTR. The PTR in turn points to the start of buffer data which can be in any location in system memory. The VMAC processes one Buffer Descriptor after the next in the ring. When the last BD has been processed, it circles back to the beginning of the ring and continues processing from there.

12.5.2 Transmit Buffer Descriptor

The Transmit Buffer descriptor consists of two 32-bit words, the INFO and PTR words. The INFO word is broken up into a number of smaller fields described on the following page. Note that the processor fills the INFO field with one set of data, and then release the buffer to the VMAC by setting the OWN bit. Once the VMAC has completed processing this buffer, it will fill the INFO word with different data and clear the OWN bit.

The processor must completely set up the Buffer Descriptor and completely fill the buffer with data before setting the OWN bit. Once the OWN bit is set, the processor must not alter the BD or buffer data. Once the OWN bit is set the VMAC will clear the OWN bit once the buffer has been sent or error condition has occurred.

1. Transmit Buffer Descriptor written by CPU

Address: 0x00

31	30~19	18	17	16	15~11	10~0
OWN	Reserved	ADCS	LAST	FRST	Reserved	TXLN

Address: 0x04

30~0

POINTER

Fig. 12-6 VMAC transmit buffer descriptor written by CPU

The field descriptions for these registers are shown in the following table:

Table 12-2 VMAC-tx buffer descriptor for CPU

Bit	Attr	Reset Value	Description
63:32	RW	0x0	POINTER 32 bit physical address to the start of the buffer data. Does not have to be word aligned. The DMA controller will DMA bytes until it becomes word aligned, then it will transfer Words. Unchanged by VMAC.
31	RW	0x0	OWN 1'b0: buffer owned by the CPU 1'b1: buffer owned by the VMAC
30:19	N/A	0x0	Reserved
18	RW	0x0	ADCR ADDCRC 1'b1: VMAC will compute and add the 4 byte CRC to the end of the packet. 1'b0: don't add the CRC (FCS) to the end of the packet
17	RW	0x0	LAST This bit must be set to one when it is the last buffer in a packet. Note that both FIRST and LAST are set at the same time if the buffer is large enough to hold the entire packet
16	RW	0x0	FIRST This bit must be set to one when it is the first buffer in a packet
15:11	N/A	0x0	Reserved
10:0	RW	0x0	TXLEN length of data in this buffer to be transmitted. Can be zero length in the case of packet-chaining, but the minimum length for a packet is 64 bytes.

2. Transmit Buffer Descriptor Written by VMAC

Address: 0x00

31	30	29	28	27~24	23	22	21	20~19	18	17	16	15~11	10~0
OWN	Rese rved	UFLO	LTCL	RTRY	DRP	DEFR	CRLS	Rese rved	ADCR	LAST	FRST	Rese rved	TXLN

Address: 0x04

30~0

POINTER

Fig. 12-7 VMAC transmit buffer descriptor written by VMAC

The field descriptions for these registers are shown in the following table:

Table 12-3 VMAC-tx buffer descriptor for VMAC

Bit	Attr	Reset Value	Description
63:32	R	0x0	POINTER 32 bit physical address to the start of the buffer data. Does not have to be word aligned. The DMA controller will DMA bytes until it becomes word aligned, then it will transfer words. Unchanged by VMAC.
31	RW	0x0	OWN Cleared by VMAC. When this bit is cleared, the TXINT bit is set if END is also set and the transmission has complete.
30	N/A	0x0	Reserved
29	RW	0x0	UFLO Underflow error, packet data corrupted and dropped because data was not available in time. Larger FIFOs, more AHB bus bandwidth or lower system latency required.

Bit	Attr	Reset Value	Description
28	RW	0x0	LATECOL Late collision error, Packet dropped due to late collision
27:24	RW	0x0	RTRY Retry count, Number of times the packet was retried. Packet transmission is attempted up to 16 times. If the packet is transmitted successfully on the first try, the value is zero.
23	RW	0x0	DRP More than 16 retransmissions were attempted and the packet was dropped
22	RW	0x0	DEFR Transmission was deferred due to traffic on the wire
21	RW	0x0	CRLS Carrier sense was lost during transmission
20:19	N/A	0x0	Reserved
18	R	0x0	ADCR ADDCRC 1'b1: VMAC will compute and add the 4 byte CRC to the end of the packet. 1'b0: don't add the CRC (FCS) to the end of the packet
17	R	0x0	LAST This bit is one when it is the last buffer in a packet.
16	R	0x0	FIRST This bit is one when it is the first buffer in a packet
15:11	N/A	0x0	Reserved
10:0	R	0x0	TXLEN length of data in this buffer to be transmitted.

12.5.3 Receive Buffer Descriptor

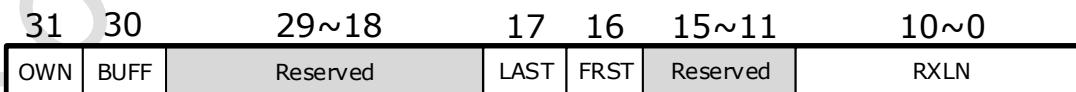
The receive buffer descriptor is very similar to the Transmit Buffer Descriptor and again consists of two 32-bit Words, the INFO and PTR Words. The INFO word is broken up into a number of smaller fields described on the following page.

Note that the processor fills the INFO field with one set of data, and then releases the buffer to the VMAC by setting the OWN bit. Once the VMAC has completed processing this buffer, it will fill the INFO word with different data and clear the OWN bit.

The processor must completely set up the Buffer Descriptor before setting the OWN bit. The data buffer can be left uninitialized as VMAC will fill it with data once a packet has been received. Once the OWN bit is set, the processor must NOT alter the Buffer Descriptor. Once the OWN bit is set the VMAC may begin operating on the buffer immediately. The VMAC will clear the OWN bit once the buffer has been filled with data or an error condition has occurred.

1. Receive Buffer Descriptor Written by VMAC

Address: 0x00



Address: 0x04

30~0

POINTER

Fig. 12-8 VMAC receive buffer descriptor written by VMAC

The field descriptions for this register are shown below.

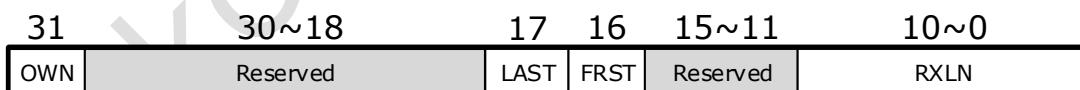
Table 12-4 VMAC-rx buffer descriptor for VMAC

Bit	Attr	Reset Value	Description
63:32	R	0x0	POINTER 32 bit physical address to the start of the buffer data. Does

Bit	Attr	Reset Value	Description
			not have to be Word aligned. The DMA controller will DMA bytes until it becomes word aligned, then it will transfer words.
31	RW	0x0	OWN 1'b0: buffer owned by the CPU 1'b1: buffer owned by the VMAC This bit is cleared by VMAC. When this bit is cleared and LAST is set to 1, the RXINT bit is set.
30	RW	0x0	BUFF Buffer error, an error occurred during packet chaining, with data in one or more of the buffers in the chain corrupted. Software should discard the entire packet and return the buffers back to the VMAC.
29:18	N/A	0x0	Reserved
17	RW	0x0	LAST This bit must be set to one when it is the last buffer in a packet.
16	RW	0x0	FIRST This bit must be set to one when it is the first buffer in a packet. Note that in certain error conditions, you may come across a FIRST bit being set without a corresponding END. In this case the previous packet must be dropped.
15:11	N/A	0x0	Reserved
10:0	RW	0x0	RXLEN Length of the data in this buffer in bytes. Might be zero length. Note that if a packet of exactly the same length as the RXLEN is received, there is a chance that the VMAC will release the buffer without the LAST bit set in anticipation of using buffer chaining. The next buffer will be released with the LAST bit set but will be zero length. This is especially true when the CPU is clocking much faster than the VMAC .To prevent chaining, the CPU should set RXLEN to a slightly larger value than the largest expected packet length.

2. Receive Buffer Descriptor Written by CPU

Address: 0x00



Address: 0x04

30~0

POINTER

Fig. 12-9 VMAC receive buffer descriptor written by CPU

The field descriptions for this register are shown below.

Table 12-5 VMAC-rx buffer descriptor for CPU

Bit	Attr	Reset Value	Description
63:32	R	0x0	POINTER 32 bit physical address to the start of the buffer data. Does not have to be word aligned. The DMA controller will DMA bytes until it becomes word aligned, then it will transfer words.
31	RW	0x0	OWN 1'b0: Buffer owned by the CPU

Bit	Attr	Reset Value	Description
			1'b1: Buffer owned by the VMAC
30:18	N/A	0x0	Reserved
17	RW	0x0	LAST This bit is one when it is the last buffer in a packet. CPU clears it to zero.
16	RW	0x0	FIRST This bit is one when it is the first buffer in a packet, CPU clears it to zero
15:11	N/A	0x0	Reserved
10:0	RW	0x0	RXLEN Maximum length of data in this buffer in bytes. Can be zero but strongly discouraged. Minimum of 64 bytes recommended. 1536 recommended unless using chaining.

12.5.4 Buffer Chain

The length of each buffer is defined in the RXLEN or TXLEN fields within the INFO word of each Buffer Descriptor. The length of a buffer can be between zero and 2047 bytes though buffer of less than 64 bytes and strongly discouraged. Larger FIFOs are highly recommended when using buffer chaining.

Buffer chaining is also useful for receive packets by using more buffers of smaller sizes. Chaining allows the use of smaller buffers which in turn creates more buffers in the same amount of memory. More receive buffers means there a lower chance of dropping important short packets and relying on upper level software to transmit the dropped packets. Note that by chaining buffers you can still assemble the packet data to be contiguous for large packets.

12.5.5 Automatic Descriptor Polling

The TX and RX BDT rings will be polled at the frequency programmed into the POLLRATE register. The DMA controller polls the BDT rings to see if the next descriptors in the chain are owned by the VMAC. There are also several instances where automatic polling of the descriptor chains will occur.

Chapter 13 Audio Codec

13.1 Overview

The Audio Codec is a low power, high resolution, stereo CODEC solution that employs Sigma-Delta noise-shaping technique. The DAC with 24bit resolution and power amplifier are integrated.

It supports the following features:

- 24bit DAC
- Support 16Ω to 32Ω headphone output and speaker output
- Support Mono, Stereo channel
- Digital interpolation and decimation filter integrated
- Sampling rate of 8/12/16/24/32/44.1/48/96KHz
- Optional fractional PLL available that support 6MHz to 20MHz clock input to any clock

13.2 Block Diagram

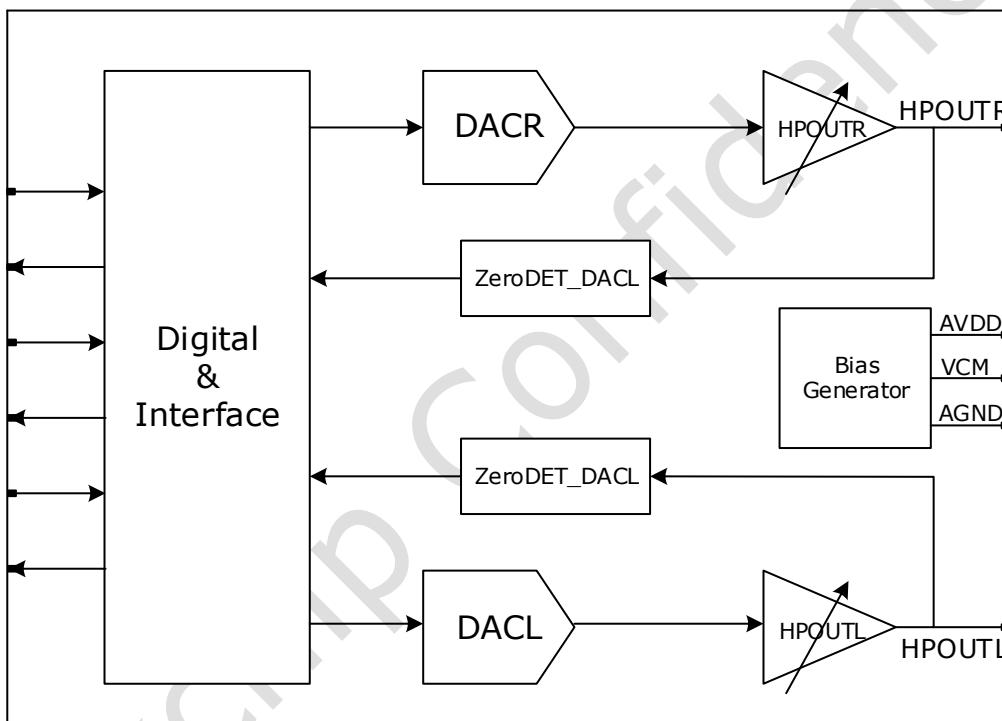


Fig. 13-1 Audio Codec Block Diagram

13.3 Function Description

13.3.1 Digital Interface

The Codec has the I2S PCM interface of audio data stream in for DAC, which can be configured in master or slave mode. Different audio data formats are available for different operating modes, which are demonstrated in below table.

Table 13-1 Supported Data Formats in Different Modes

Data Formats	DAC	
	Master	Slave
Left Justified	✓	✓
Right Justified	✓	✓
I ² S	✓	✓
DSP/PCM mode A	✓	✓
DSP/PCM mode B	✓	✓

I2S_PCM interface supports five audio data formats: Left Justified mode, Right Justified mode,

I2S mode, DSP/PCM mode A and mode B. They are valid when the device operates as a master or slave.

For Left Justified mode, the data format is illustrated in Fig. 24-2. The MSB is valid at the first rising edge of sck after ws transition is done. The other valid bits up to the LSB are transmitted sequentially. Due to varied word length, different sck frequency and sample rate, some unused sck cycles may appear before every ws transition, which means the data in this period is invalid.

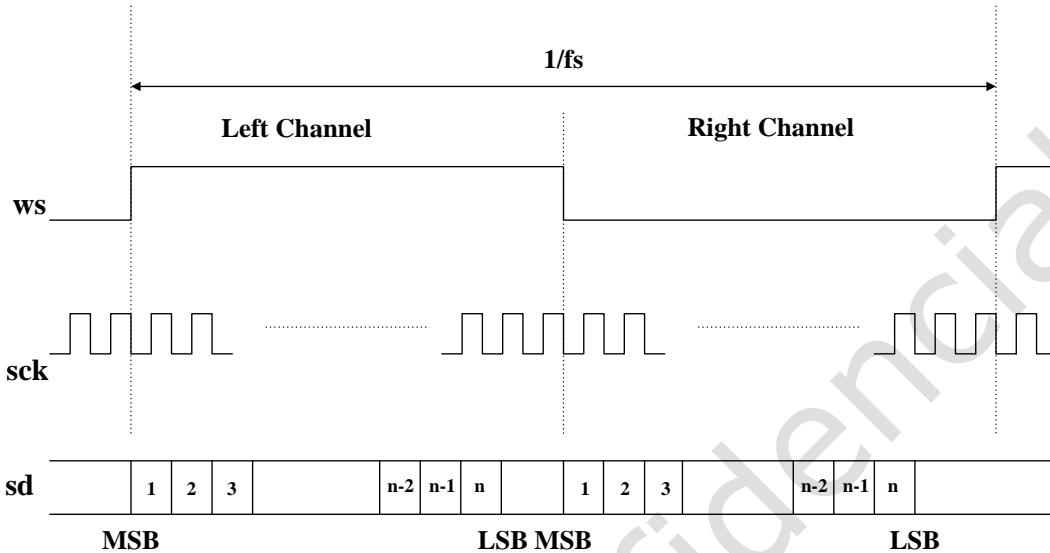


Fig. 13-2 Left Justified Mode (assuming n-bit word length)

For Right Justified mode, the data format is shown in Fig. 24-3. The LSB becomes valid at the last rising edge of sck before ws transition is done. As the MSB is transmitted first, the other valid bits up to the LSB are followed in order. Due to varied word length, different sck frequency and sample rate, some unused sck cycles may exist after every ws transition, which means the data in this period is invalid.

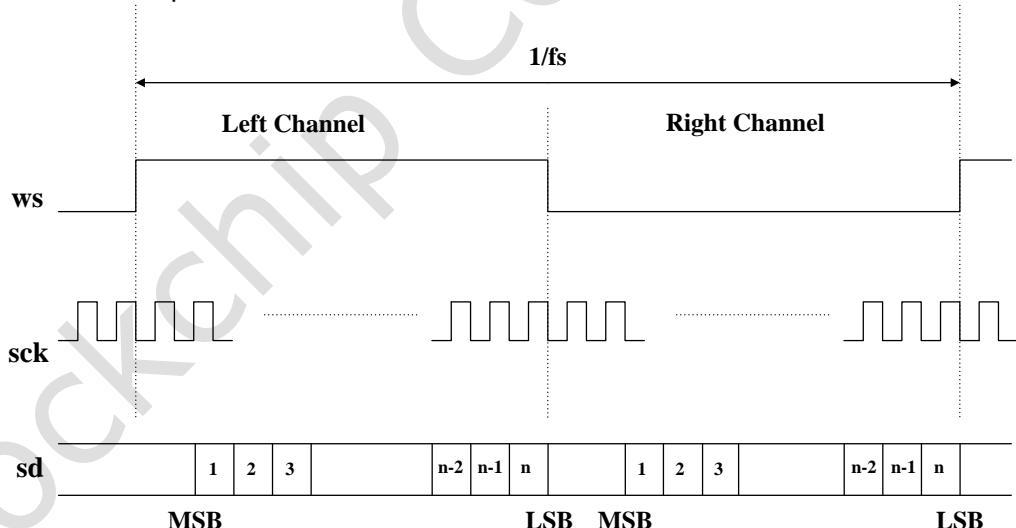


Fig. 13-3 Right Justified Mode (assuming n-bit word length)

For I2S mode, the data format is depicted in Fig. 24-4. The MSB becomes available at the second rising edge of sck when ws transition is done. The other valid bits up to the LSB are transmitted in order. Due to varied word length, different sck frequency and sample rate, some unused sck cycles may appear between the LSB of the current sample and the MSB of the next one, which means the data in this period can be ignored.

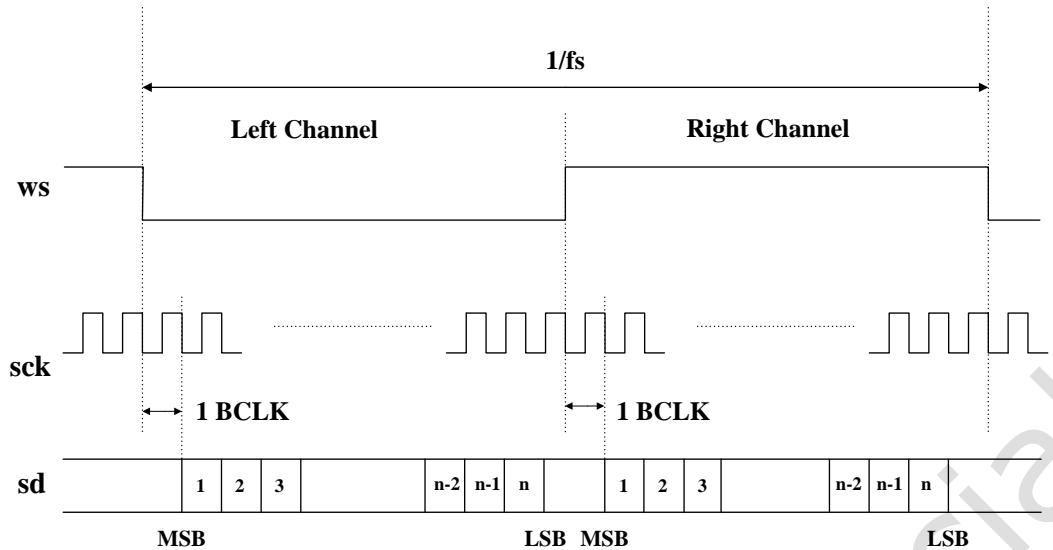


Fig. 13-4 I2S Mode (assuming n-bit word length)

For DSP/PCM mode, the left channel data is transmitted first, followed by right channel data. For DSP/PCM mode A/B, the MSB is available at the second and first rising edge of sck after the rising edge of ws respectively, as shown in Fig. 24-5 and Fig. 24-6. Based on word length, sck frequency and sample rate, there may be some invalid data between the LSB of the right channel data and the next sample.

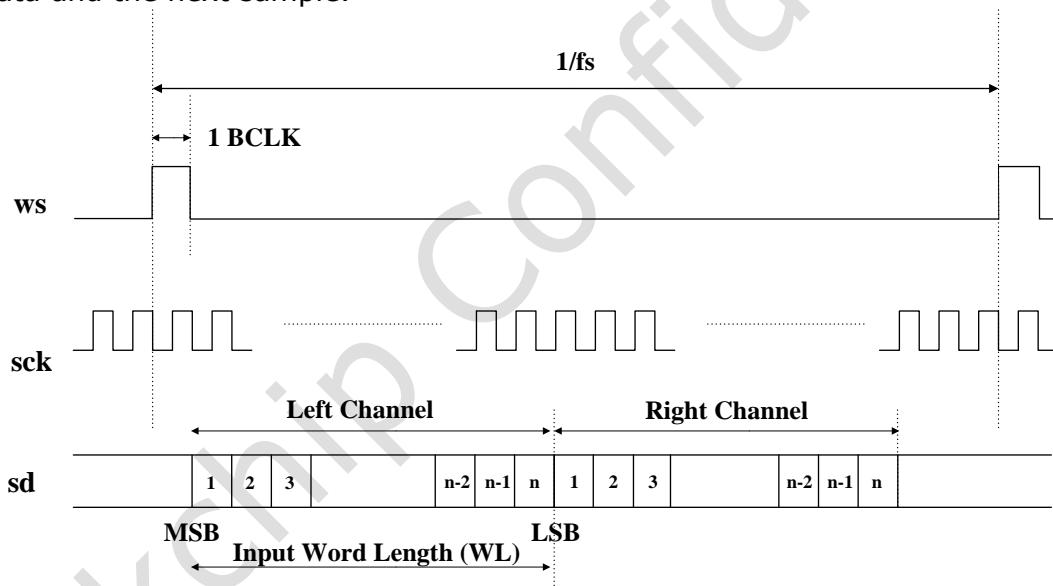


Fig. 13-5 DSP/PCM Mode A (assuming n-bit word length)

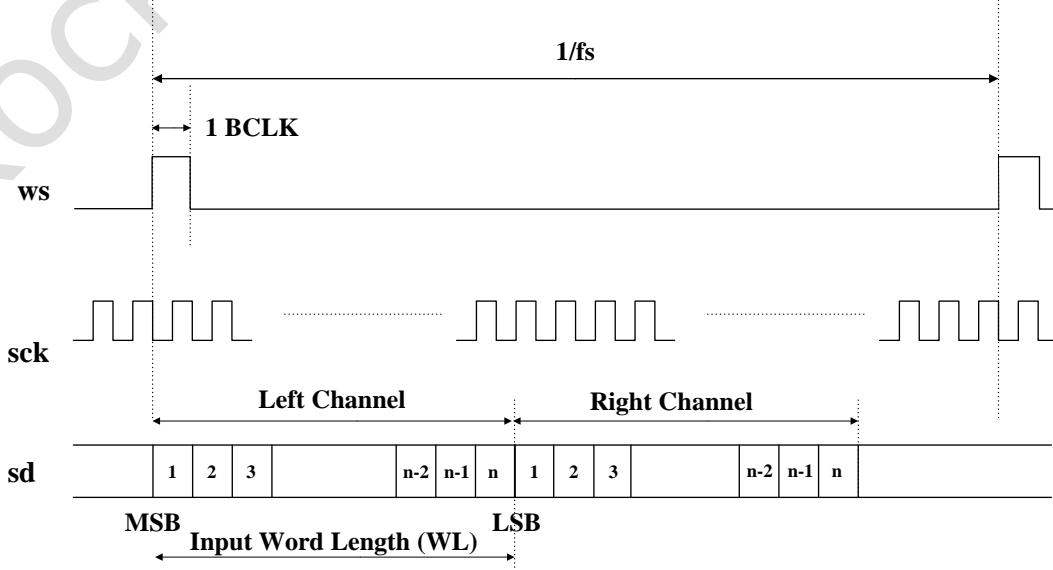


Fig. 13-6 DSP/PCM Mode B (assuming n-bit word length)

13.3.2 Analog Interface

The Codec DAC output can drive 16Ω or 32Ω headphone load either through DC-blocking capacitor.

In the configuration using DC-blocking capacitor, shown in following figure, the headphone ground is connected to the real ground. The capacitance and the load resistance determine the lower cut-off frequency. For instance, if 16Ω headphone and $100\mu F$ DC-blocking capacitor are used, the lower cut-off frequency is:

$$f = \frac{1}{2\pi RC} = \frac{1}{2\pi \times 16 \times 100 \times 10^{-6}} = 99.5Hz$$

The DC-blocking capacitor can be increased to lower the cut-off frequency for better bass response.

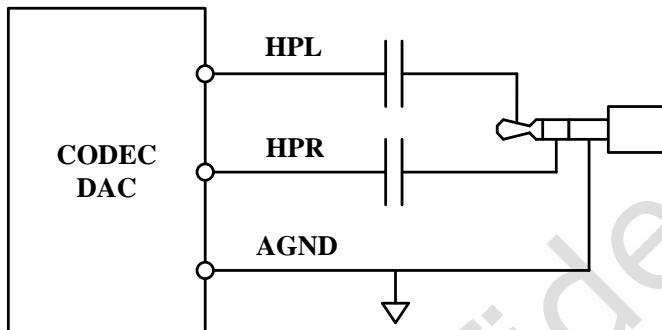


Fig. 13-7 DC-blocking capacitor

Microphone bias output is used to bias external microphones. The bias voltage can varies from $0.5*AVDD$ to $0.85* AVDD$ with a step of $0.05*AVDD$.

13.3.3 Interface Relationship

The relationship between I2S interface and the parallel audio data for DAC channels are shown in the following figure.

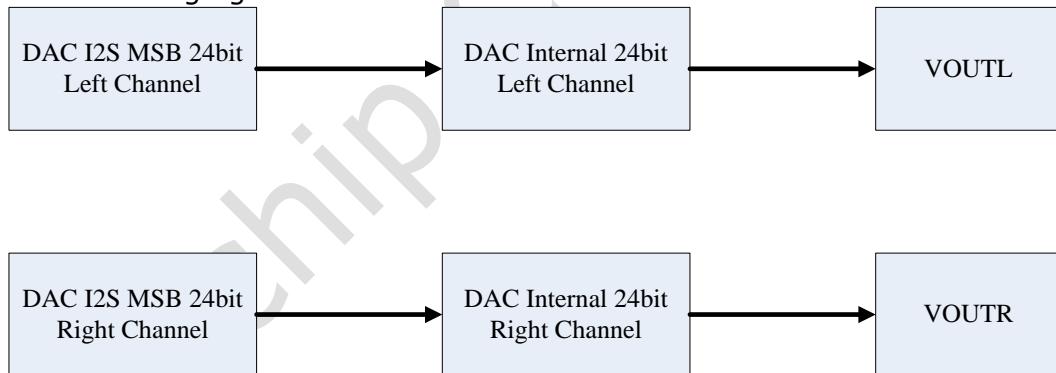


Fig. 13-8 DAC Channel Relationship

Following table lists the corresponding signals between I2S and ACODEC.

Table 13-2 Signals between I2S and ACODEC

I2S signal	ACODEC signal
i2s_8ch_sclk_out	Pin_sck_i
I2s_8ch_sclk_in	Pin_sck_o
I2s_8ch_tx_lrck_out	Pin_dac_ws_i
I2s_8ch_tx_lrck_in	Pin_dac_ws_o
I2s_8ch_sdo[0]	Pin_dac_sd_i

13.4 Register Description

13.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
Codec_REG0	0x0000	B	0x03	Audio Codec register 0

Name	Offset	Size	Reset Value	Description
Codec_REG1	0x000c	B	0x00	Audio Codec register 1
Codec_REG2	0x0010	B	0x50	Audio Codec register 2
Codec_REG3	0x0014	B	0xe0	Audio Codec register 3
Codec_REG4	0x0088	B	0x00	Audio Codec register 4
Codec_REG5	0x008c	B	0x00	Audio Codec register 5
Codec_REG6	0x0090	B	0x00	Audio Codec register 6
Codec_REG7	0x0094	B	0x00	Audio Codec register 7
Codec_REG8	0x0098	B	0xc0	Audio Codec register 8
Codec_REG9	0x009c	B	0x05	Audio Codec register 9
Codec_REG10	0x00a0	B	0x00	Audio Codec register 10

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

13.4.2 Detail Register Description

Codec_REG0

Address: Operational Base + offset (0x0000)

Audio Codec register 0

Bit	Attr	Reset Value	Description
7	RO	0x0	reserved
6	RW	0x0	PRB Power reset bypass 1'b0: Disable Bypass 1'b1: Enable Bypass
5:2	RO	0x0	reserved
1	RW	0x1	CDCR Codec digital core reset This reset only reset the codec data path. 1'b0: Reset 1'b1: Work
0	RW	0x1	CSR Codec system reset This signal will reset the registers which control all the digital and analog part. 1'b0: Reset 1'b1: Work

Codec_REG1

Address: Operational Base + offset (0x000c)

Audio Codec register 1

Bit	Attr	Reset Value	Description
7:6	RO	0x0	reserved
5	RW	0x0	PINDIR Choose the input or output direction of the pin which is related to the I2S mode. 1'b0: Input(Slave Mode) 1'b1: Output(Master Mode)

Bit	Attr	Reset Value	Description
4	RW	0x0	I2SMODE DAC I2S Mode Select 1'b0: Slave mode• 1'b1: Master mode
3:0	RO	0x0	reserved

Codec_REG2

Address: Operational Base + offset (0x0010)

Audio Codec register 2

Bit	Attr	Reset Value	Description
7	RW	0x0	LRCP DAC LRC Polarity 1'b0: Normal 1'b1: Reversal
6:5	RW	0x2	VWL DAC Valid Word Length in one 1/2 Frame 2'b11: 32 bits 2'b10: 24 bits 2'b01: 20 bits 2'b00: 16 bits
4:3	RW	0x2	DACM DAC mode 2'b11: PCM Mode 2'b10: I2S Mode 2'b01: Left Justified Mode 2'b00: Right Justified Mode Note. Same word length in 1/2frame and valid data is not supported in Right Justified Mode. For example, 32/24 or 24/20 is supported, but 32/32 or 24/24 is not supported. (1/2frame length/valid data length)
2	RW	0x0	LRS DAC Left-Right Swap 1'b0: Normal 1'b1: Swap
1:0	RO	0x0	reserved

Codec_REG3

Address: Operational Base + offset (0x0014)

Audio Codec register 3

Bit	Attr	Reset Value	Description
7:4	RO	0x0	reserved

Bit	Attr	Reset Value	Description
3:2	RW	0x3	FWL DAC 1/2Frame Word Length 2'b11: 32 bits 2'b10: 24 bits 2'b01: 20 bits 2'b00: 16 bits
1	RW	0x1	DACR DAC Reset 1'b0: Reset 1'b1: Work
0	RW	0x0	BCP DAC Bit Clock polarity 1'b0: Normal 1'b1: Reversal

Codec_REG4

Address: Operational Base + offset (0x0088)

Audio Codec register 4

Bit	Attr	Reset Value	Description
7:6	RO	0x0	reserved
5	RW	0x0	DACLIVEN The enable signal of reference Voltage buffer for left DAC PATH: 1'b0: Stop Working 1'b1: Work
4	RW	0x0	DACRRVEN The enable signal of reference Voltage buffer for right DAC PATH: 1'b0: Stop Working 1'b1: Work
3	RW	0x0	DACLCLKEN The enable signal of CLOCK module for DACL 1'b0: Set CLOCK to logic 1'b1 1'b1: Work
2	RW	0x0	DACRCLKEN The enable signal of CLOCK module for DACR 1'b0: Set CLOCK to logic 1'b1 1'b1: Work
1	RW	0x0	DACLEN The enable signal of DACL •module 1'b0: Stop work 1'b1: Work
0	RW	0x0	DACREN The enable signal of DACR •module 1'b0: Stop work 1'b1: Work

Codec_REG5

Address: Operational Base + offset (0x008c)

Audio Codec register 5

Bit	Attr	Reset Value	Description
7:4	RO	0x0	reserved
3	RW	0x0	HPOUTLINIT The initial signal of HPOUTL module 1'b0: Initialization 1'b1: Work
2	RW	0x0	HPOUTRINIT The initial signal of HPOUTR module 1'b0: Initialization 1'b1: Work
1	RW	0x0	HPOUTLEN The enable signal of HPOUTL module 1'b0: Stop working 1'b1: Work
0	RW	0x0	HPOUTREN The enable signal of HPOUTR module 1'b0: Stop working 1'b1: Work

Codec_REG6

Address: Operational Base + offset (0x0090)

Audio Codec register 6

Bit	Attr	Reset Value	Description
7:6	RO	0x0	reserved
5	RW	0x0	COUREN The enable signal of current source for CODEC DAC 1'b0: Stop Working 1'b1: Work
4	RW	0x0	PRECTRL The Control signal of the precharge for CODEC DAC. 1'b0: Precharge 1'b1: Discharge
3	RW	0x0	DACLRVEN The enable signal of high and low reference Voltage buffer for DACL module 1'b0: Stop Working 1'b1: Work
2	RW	0x0	DACRRVEN The enable signal of high and low reference Voltage buffer for DACR module 1'b0: Stop Working 1'b1: Work

Bit	Attr	Reset Value	Description
1	RW	0x0	VOUTLCROSSZEROEN The enable signal of zero-crossing detection module for VOUTL: 1'b0: Stop Working, output 0 electrical level 1'b1: Work
0	RW	0x0	VOUTRCROSSZEROEN The enable signal of zero-crossing detection module for VOUTR: 1'b0: Stop Working, output 0 electrical level 1'b1: Work

Codec_REG7

Address: Operational Base + offset (0x0094)

Audio Codec register 7

Bit	Attr	Reset Value	Description
7:5	RO	0x0	reserved
4:0	RW	0x00	HPOUTLGAIN The signal to select gain of HPOUTL module: 5'b00000: -39dB 5'b11010: 6dB 5'b11111: 0dB Step: 1.5dB

Codec_REG8

Address: Operational Base + offset (0x0098)

Audio Codec register 8

Bit	Attr	Reset Value	Description
7:5	RO	0x0	reserved
4:0	RW	0x00	HPOUTRGAIN The signal to select gain of HPOUTR module: 5'b00000: -39dB 5'b11010: 6dB 5'b11111: 0dB Step: 1.5dB

Codec_REG9

Address: Operational Base + offset (0x009c)

Audio Codec register 9

Bit	Attr	Reset Value	Description
7	RW	0x0	DACLINIT The initial signal of DACL module 1'b0: Initialization 1'b1: Work
6	RW	0x0	DACRINIT The initial signal of DACR module 1'b0: Initialization 1'b1: Work

Bit	Attr	Reset Value	Description
5	RW	0x0	HPOUTLMUTE The mute signal of HPOUTL module 1'b0: MUTE 1'b1: Work
4	RW	0x0	HPOUTRMUTE The mute signal of HPOUTR module 1'b0: MUTE 1'b1: Work
3:2	RW	0x1	HPOUTLPOPCTRL The POP Sound Control signal of the HPOUTL module. 2'b01: Precharge(or Discharge) 2'b10: Work
1:0	RW	0x1	HPOUTRPOPCTRL The POP Sound Control signal of the HPOUTR module. 2'b01: Precharge(or Discharge) 2'b10: Work

Codec_REG10

Address: Operational Base + offset (0x00a0)

Audio Codec register 10

Bit	Attr	Reset Value	Description
7:6	RO	0x0	reserved
5:0	RW	0x00	CHARGESEL The signal to select current to Precharge/Discharge [5]: Select current 0.27*I0 1'b0: Select 1'b1: Don't select [4]: Select current 0.5*I0 1'b0: Select 1'b1: Don't select [3]: Select current I0 1'b0: Select 1'b1: Don't select [2]: Select current 1.3* I0 1'b0: Select 1'b1: Don't select [1]: Select current 2.6*I0 1'b0: Select 1'b1: Don't select [0]: Select current 4*I0 1'b0: Select 1'b1: Don't select I0 is a reference current, the current can stack

Chapter 14 SFC (Serial Flash Controller)

14.1 Overview

The serial flash controller (SFC) is used to control the data transfer between the chip system and the serial nor/nand flash device.

The SFC supports the following features:

- Support AHB slave interface to configure register and read/write serial flash
- Support AHB master interface to transfer data from/to SPI flash device
- Support AHB burst with incr4x32bits, or incr x32bits
- Support two independent clock domain: AHB clock and SPI clock
- Support x1,x2,x4 data bits mode
- Support up to 4 chip select
- Support interrupt output, interrupt maskable
- Support Spansion, MXIC, Gigadevice ... vendor's nor flash memory.

14.2 Block Diagram

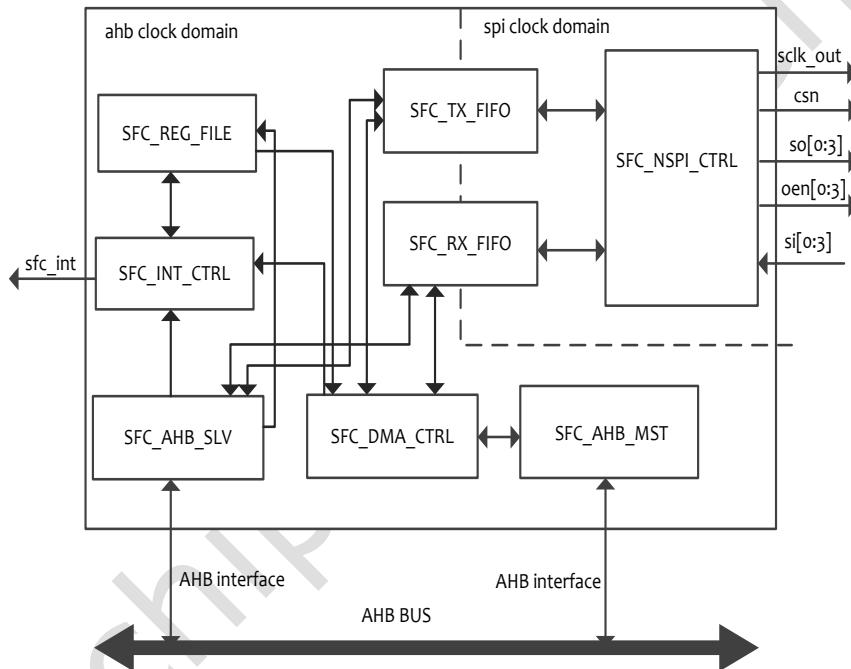


Fig. 14-1 SFC architecture

14.3 Function Description

14.3.1 SFC slave

The AHB slave is used to configure the register, and also write to/read from the serial nor/nand flash device.

The SFC_CTRL register is a global control register, when the controller is in busy state (SFC_SR), SFC_CTRL cannot be set. The field sclk_idle_level_cycles of this register is used to configure the idle level cycles of sclk before read the first bit of the read command.

Like the following picture shows: the red line of the sclk is the idle cycles, during these cycles, the chip pad is switched to output. When sclk_idle_level_cycles=0, it means there will be not such idle level.

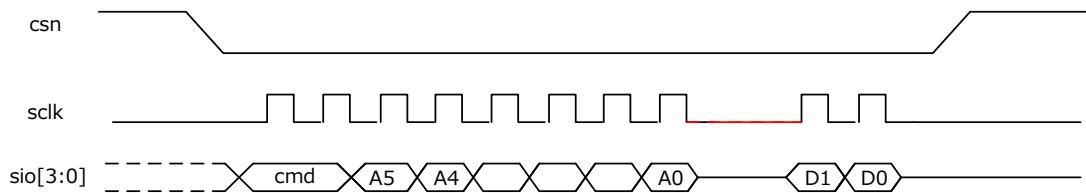


Fig. 14-2 idle cycles

When the field spi mode is set, the transfer waveform will like following, and switch to mode3.

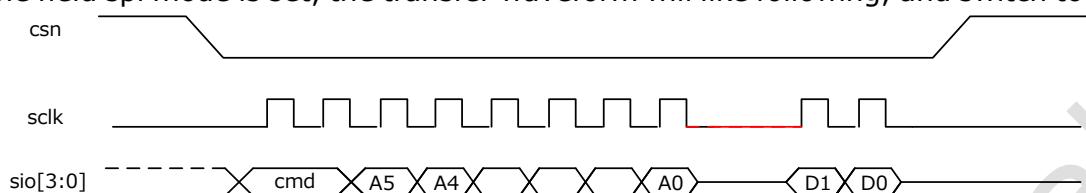


Fig. 14-3 SPI mode

14.4 Register Description

14.4.1 Registers Summary

Name	Offset	Size	Reset Value	Description
SFC_CTRL	0x0000	W	0x00000000	Control Register 0
SFC_IMR	0x0004	W	0x00000000	Interrupt Mask
SFC_ICLR	0x0008	W	0x00000000	Interrupt Clear
SFC_FTLLR	0x000c	W	0x00000000	FIFO Threshold Level
SFC_RCVR	0x0010	W	0x00000000	SFC Recover
SFC_AX	0x0014	W	0x00000000	SFC AX Value
SFC_ABIT	0x0018	W	0x00000000	Flash Address bits
SFC_ISR	0x001c	W	0x00000000	Interrupt Status
SFC_FSR	0x0020	W	0x00000001	FIFO Status
SFC_SR	0x0024	W	0x00000000	SFC Status
SFC_DMAADDR	0x0080	W	0x00000000	DMA Address
SFC_DMATR	0x0084	W	0x00000000	DMA Trigger
SFC_CMD	0x0100	W	0x00000000	SFC CMD
SFC_ADDR	0x0104	W	0x00000000	SFC Address
SFC_DATA	0x0108	W	0x00000000	SFC DATA

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

14.4.2 Detail Register Description

SFC_CTRL

Address: Operational Base + offset (0x0000)

Control Register 0

Bit	Attr	Reset Value	Description
31:14	RO	0x0	reserved
13:12	RW	0x0	DATB Data bits width Data bits width 2'b00: 1bit 2'b01: 2bits 2'b10: 4bits

Bit	Attr	Reset Value	Description
11:10	RW	0x0	ADRB Address bits width Address bits width 2'b00: 1bit 2'b01: 2bits 2'b10: 4bits
9:8	RW	0x0	CMDB Command bits width Command bits width 2'b00: 1bit 2'b01: 2bits 2'b10: 4bits 2'b11:reserved
7:4	RW	0x0	IDLE_CYCLE Sclk Idle Level Cycles 4'b0000: idle hold is disable 4'b0001: hold the sclk_out in idle for two cycles when switch to shift in
3:2	RO	0x0	reserved
1	RW	0x0	SHIFTPHASE Shift Phase Select 1'b0: shift in the data at posedge sclk_out 1'b1: shift in the data at negedge sclk_out
0	RW	0x0	SPIM SPI MODE Select SPI MODE Select 1'b0: mode 0 1'b1: mode 3

SFC_IMR

Address: Operational Base + offset (0x0004)

Interrupt Mask

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7	RW	0x0	DMAM DMA finish interrupt mask 1'b0: dma_intr interrupt is not masked 1'b1: dma_intr interrupt is masked
6	RW	0x0	NSPIM SPI Error interrupt mask 1'b0: nspi_intr interrupt is not masked 1'b1: nspi_intr interrupt is masked
5	RW	0x0	AHBM AHB Error interrupt mask 1'b0: ahb_intr interrupt is not masked 1'b1: ahb_intr interrupt is masked
4	RW	0x0	TRANSM Transfer finish Interrupt Mask 1'b0: transf_intr interrupt is not masked 1'b1: transf_intr interrupt is masked

Bit	Attr	Reset Value	Description
3	RW	0x0	TXEM Transmmit FIFO Empty Interrupt Mask 1'b0: txe_intr interrupt is not masked 1'b1: txe_intr interrupt is masked
2	RW	0x0	TXOM Transmmit FIFO Overflow Interrupt Mask 1'b0: txo_intr interrupt is not masked 1'b1: txo_intr interrupt is masked
1	RW	0x0	RXUM Receive FIFO Underflow Interrupt Mask 1'b0: rxu_intr interrupt is not masked 1'b1: rxu_intr interrupt is masked
0	RW	0x0	RXFM Receive FIFO Full Interrupt Mask 1'b0: rxf_intr interrupt is not masked 1'b1: rxf_intr interrupt is masked

SFC_ICLR

Address: Operational Base + offset (0x0008)

Interrupt Clear

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7	W1C	0x0	DMAC DMA finish Interrupt Clear Write and clear
6	W1C	0x0	NSPIC SPI Error Interrupt Clear Write and clear
5	W1C	0x0	AHBC AHB Error Interrupt Clear Write and clear
4	W1C	0x0	TRANSC Transfer Finish Interrupt Clear Write and clear
3	W1C	0x0	TXEC Transmit FIFO Empty Interrupt Clear Write and clear
2	W1C	0x0	TXOC Transmit FIFO Overflow Interrupt Clear Write and clear
1	W1C	0x0	RXUC Receive FIFO Underflow Interrupt Clear Write and clear
0	W1C	0x0	RXFC Receive FIFO Full Interrupt Clear Write and clear

SFC_FTLR

Address: Operational Base + offset (0x000c)

FIFO Threshold Level

Bit	Attr	Reset Value	Description
31:16	RO	0x0	reserved

Bit	Attr	Reset Value	Description
15:8	RW	0x00	RXFTLR Receive FIFO Threshold Level When the number of receive FIFO entries is bigger than or equal to this value, the receive FIFO full interrupt is triggered.
7:0	RW	0x00	TXFTLR Transmit FIFO Threshold Level When the number of transmit FIFO entries is less than or equal to this value, the transmit FIFO empty interrupt is triggered.

SFC_RCVR

Address: Operational Base + offset (0x0010)

SFC Recover

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RO	0x0	RCVR SFC Recover Write 1 to recover the SFC State Machine, FIFO state and other logic state.

SFC_AX

Address: Operational Base + offset (0x0014)

SFC AX Value

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7:0	RO	0x00	AX The AX Value when doing the continuous read (enhance mode).

SFC_ABIT

Address: Operational Base + offset (0x0018)

Flash Address bits

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	RW	0x0	ABIT Flash Address bits

SFC_ISR

Address: Operational Base + offset (0x001c)

Interrupt Status

Bit	Attr	Reset Value	Description
31:8	RO	0x0	reserved
7	RW	0x0	DMAS DMA Finish Interrupt Status 1'b0: not active 1'b1: active
6	RW	0x0	NSPIS SPI Error Interrupt Status 1'b0: not active 1'b1: active

Bit	Attr	Reset Value	Description
5	RW	0x0	AHBS AHB Error Interrupt Status 0: not active 1: active
4	RW	0x0	TRANSS Transfer finish Interrupt Status 1'b0: not active 1'b1: active
3	RW	0x0	TXES Transmit FIFO Empty Interrupt Status 1'b0: not active 1'b1: active
2	RW	0x0	TXOS Transmit FIFO Overflow Interrupt Status 1'b0: not active 1'b1: active
1	RW	0x0	RXUS Receive FIFO Underflow Interrupt Status 1'b0: not active 1'b1: active
0	RW	0x0	RXFS Receive FIFO Full Interrupt Status 1'b0: not active 1'b1: active

SFC_FSR

Address: Operational Base + offset (0x0020)

FIFO Status

Bit	Attr	Reset Value	Description
31:21	RO	0x0	reserved
20:16	RW	0x00	RXWLVL RX FIFO Water Level 5'b00000: fifo is empty 5'b00001: 1 entry is taken ...
15:13	RO	0x0	reserved
12:8	RO	0x00	TXWLVL TX FIFO Water Level 5'b00000: fifo is full 5'b00001: left 1 entry ...
7:4	RO	0x0	reserved
3	RO	0x0	RXFS Receive FIFO Full Status 1'b0: rx fifo is not full 1'b1: rx fifo is full
2	RO	0x0	RXES Receive FIFO Empty Status 1'b0: rx fifo is not empty 1'b1: rx fifo is empty
1	RO	0x0	TXFS Transmit FIFO Full Status 1'b0: tx fifo is not full 1'b1: tx fifo is full

Bit	Attr	Reset Value	Description
0	RO	0x1	TXES Transmit FIFO Empty Status 1'b0: tx fifo is not empty 1'b1: tx fifo is empty

SFC_SR

Address: Operational Base + offset (0x0024)

SFC Status

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	WO	0x0	SR SFC Status 1'b0: SFC is idle 1'b1: SFC is busy When busy, don't set the control register.

SFC_DMAADDR

Address: Operational Base + offset (0x0080)

DMA Address

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	DMAADDR DMA Address

SFC_DMATR

Address: Operational Base + offset (0x0084)

DMA Trigger

Bit	Attr	Reset Value	Description
31:1	RO	0x0	reserved
0	W1C	0x0	DMATR DMA Trigger Write 1 to start the DMA transfer.

SFC_CMD

Address: Operational Base + offset (0x0100)

SFC CMD

Bit	Attr	Reset Value	Description
31:30	RW	0x0	CS Flash chip select 2'b00: chip select 0 2'b01: chip select 1 2'b10: chip select 2 2'b11: chip select 3
29:16	RW	0x0000	TRB Transfer Bytes number Total Data Bytes number that will write to /read from the flash.

Bit	Attr	Reset Value	Description
15:14	RW	0x0	ADDRB Address bits number select 2'b00: 0bits 2'b01: 24bits 2'b10: 32bits 2'b11: From the ABIT register
13	RW	0x0	CONT Continuous read mode 1'b0: disable continuous read mode 1'b1: enable continuous read mode
12	RW	0x0	WR Flash Write or Read 1'b0:read 1'b1:write
11:8	WO	0x0	DUMM Dummy Bits Number Dummy Bits Number
7:0	WO	0x00	CMD Flash CMD Flash Command

SFC_ADDR

Address: Operational Base + offset (0x0104)

SFC Address

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	ADDR SFC Address Flash's address

SFC_DATA

Address: Operational Base + offset (0x0108)

SFC DATA

Bit	Attr	Reset Value	Description
31:0	RW	0x00000000	ADDR SFC DATA Flash's Data

14.5 Interface Description

Table 14-1 1SPI interface description

Module Pin	Direction	Pad Name	IOMUX Setting
sfc_clk	O	IO_NANDrdy_EMMCcmd_SFCclk _GPIO2a4	GRF_GPIO2A_IOMUX[9:8]=2'b1 1
sfc_csn0	O	IO_NANDwrn_SFCcsn0_GPIO2a2	GRF_GPIO2A_IOMUX[5:4]=2'b1 0
sfc_csn1	O	IO_NANDrdn_SFCcsn1_GPIO2a3	GRF_GPIO2A_IOMUX[7:6]=2'b1 0
sfc_sio0	I/O	IO_NANDd0_EMMCd0_SFCsio0_ GPIO1d0	GRF_GPIO1D_IOMUX[1:0]=2'b1 1
sfc_sio1	I/O	IO_NANDd1_EMMCd1_SFCsio1_ GPIO1d1	GRF_GPIO1D_IOMUX[3:2]=2'b1 1

Module Pin	Direction	Pad Name	IOMUX Setting
sfc_sio2	I/O	IO_NANDd2_EMMCd2_SFCsio2_GPIO1d2	GRF_GPIO1D_IOMUX[5:4]=2'b11
sfc_sio3	I/O	IO_NANDd3_EMMCd3_SFCsio3_GPIO1d3	GRF_GPIO1D_IOMUX[7:6]=2'b11

Notes: I=input, O=output, I/O=input/output, bidirectional.

14.6 Application Notes

14.6.1 AHB Slave write flash flow

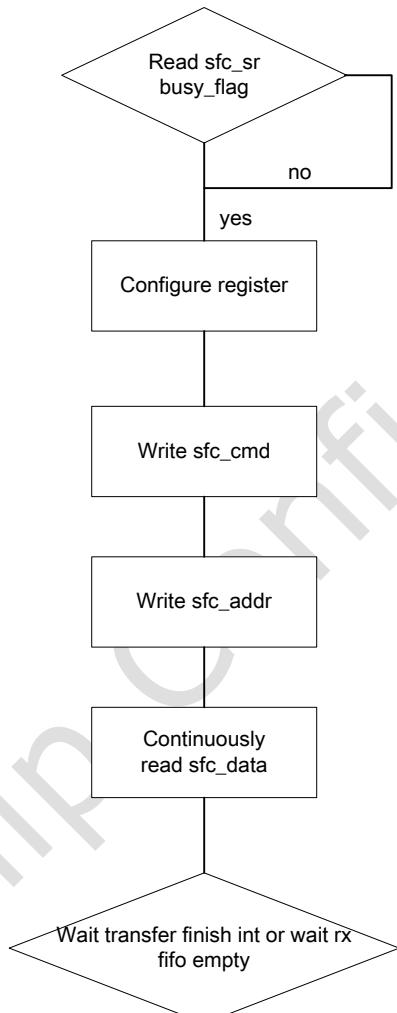


Fig. 14-4 slave mode read

14.6.2 AHB DMA transfer flow

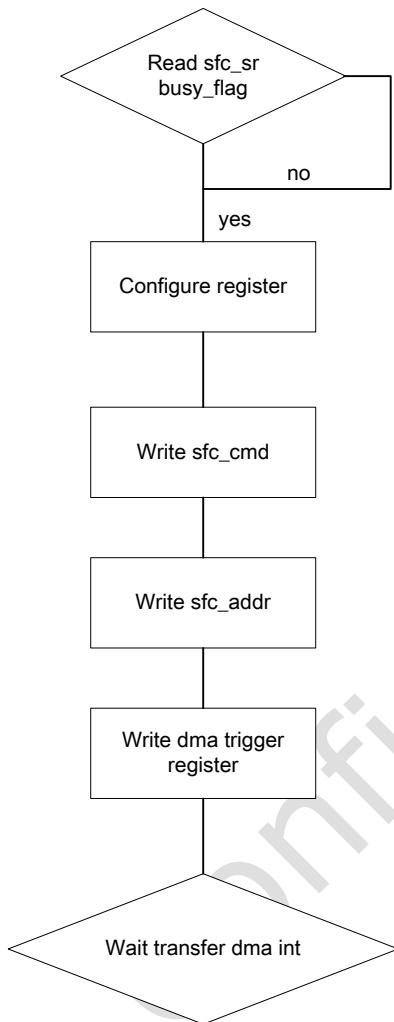


Fig. 14-5 master mode flow

14.6.3 AHB dma transfer flow

The SFC sclk need to be kept under 200MHZ. It's better to soft reset the SFC before data transfer.