OMAP5910 Dual-Core Processor
MultiMedia Card/Secure Data Memory Card (MMC/SD)
Reference Guide

Literature Number: SPRU680
October 2003

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265

Preface

Read This First

About This Manual

This document describes the options to interface the OMAP5910 multimedia processor with MultiMedia Cards, SD Memory Cards, and serial flash cards.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.

Related Documentation From Texas Instruments

The following documents describe the OMAP5910 device and related peripherals. Copies of these documents are available on the Internet at www.ti.com. *Tip:* Enter the literature number in the search box provided at www.ti.com.

OMAP5910 Dual-Core Processor MPU Subsystem Reference Guide (literature number SPRU671)

OMAP5910 Dual-Core Processor DSP Subsystem Reference Guide (literature number SPRU672)

OMAP5910 Dual-Core Processor Memory Interface Traffic Controller Reference Guide (literature number SPRU673)

OMAP5910 Dual-Core Processor System DMA Controller Reference Guide (literature number SPRU674)

OMAP5910 Dual-Core Processor LCD Controller Reference Guide (literature number SPRU675)

OMAP5910 Dual-Core Processor Universal Asynchronous Receiver/Transmitter (UART) Devices Reference Guide (literature number SPRU676)

OMAP5910 Dual-Core Processor Universal Serial Bus (USB) and Frame Adjustment Counter (FAC) Reference Guide (literature number SPRU677)

OMAP5910 Dual-Core Processor Clock Generation and System Reset Management Reference Guide (literature number SPRU678)

OMAP5910 Dual-Core Processor General-Purpose Input/Output (GPIO) Reference Guide (literature number SPRU679)

OMAP5910 Dual-Core Processor MMC/SD Reference Guide (literature number SPRU680)

OMAP5910 Dual-Core Processor Inter-Integrated Circuit (I2C) Controller Reference Guide (literature number SPRU681)

OMAP5910 Dual-Core Processor Timer Reference Guide (literature number SPRU682)

OMAP5910 Dual-Core Processor Inter-Processor Communication Reference Guide (literature number SPRU683)

OMAP5910 Dual-Core Processor Camera Interface Reference Guide (literature number SPRU684)

OMAP5905 Dual-Core Processor Multichannel Serial Interface (MCSI) Reference Guide (literature number SPRU685)

OMAP5910 Dual-Core Processor Micro-Wire Interface Reference Guide (literature number SPRU686)

OMAP5910 Dual-Core Processor Real-Time Clock (RTC) Reference Guide (literature number SPRU687)

OMAP5910 Dual-Core Processor HDQ/1-Wire Interface Reference Guide (literature number SPRU688)

OMAP5910 Dual-Core Processor PWL, PWT, and LED Peripheral Reference Guide (literature number SPRU689)

OMAP5910 Dual-Core Processor Multichannel Buffered Serial Port (McBSP) Reference Guide (literature number SPRU708)

Trademarks

OMAP and the OMAP symbol are trademarks of Texas Instruments.

Contents

1 MMC/SD Host Controller 11
1.1 MMC/SD Host Controller Features 13
1.2 MMC/SD Host-Controller Signal Pads 13
1.3 MMC/SD Host-Controller Clocks and Reset 15
1.4 MMC/SD Host Controller DMA Request 15
1.5 MMC/SD Host-Controller Interrupt 16
1.6 MMC/SD Internal Pullups 16
1.7 MMC/SD Registers 17
1.7.1 MMC-Command Register (MMC_CMD) 18
Data Direction (DDir) 19
Stream Command or Broadcast Host Response (SHR) 19
Command Type (Type) 20
Command With Busy Response (Busy) 20
Command Response (Response) 20
Send Initialization Stream (Init) 21
Card Open-Drain Mode (OD) 21
Command Index (Cmd_index) 21
1.7.2 MMC-Argument-Low/High Registers (MMC_ARGL/MMC_ARGH) 22
1.7.3 MMC-System-Configuration Register (MMC_CON) 22
Bus Width During Data Phase (DW) 22
Mode Select (Mode) 23
Power Up-Control (Power_Up) 23
Clock Divider (Clk_div) 24
1.7.4 MMC-System-Status Register (MMC_STAT) 26
Card-Status Error (Card_Err) 26
Card IRQ (Card_IRQ) 27
OCR Busy (OCR_busy) 28
Buffer Almost-Empty (A_Empty) 28
Buffer Almost-Full (A_Full) 29
Command-CRC Error (Cmd_CRC) 29
Command-Time-Out Error (Cmd_timeout) 29
Data-CRC Error (Dat_CRC) 30
Data-Time-Out Error (Dat_timeout) 30
Card-Exit-Busy State (EOF_Busy) 31
Block-Received/Sent (Block_RS) 31

Card-Enter-Busy State (Card_Busy) 32
End-of-Command (End_of_Cmd) 32
1.7.5 MMC-System-Interrupt Register (MMC_IE) 33
1.7.6 MMC-Command-Time-out Register (MMC_CTO) 34
Command-Time-out Value (CTO) 34
1.7.7 MMC-Data-Time-Out Register (MMC_DTO) 34
Data-Time-Out Value (DTO) 35
1.7.8 MMC-Data-Access Register (MMC_DATA) 35
Transmit/Receive-FIFO-Data Value (DATA) 36
1.7.9 MMC-Block-Length Register (MMC_BLEN) 36
Block Length (BLEN) 37
1.7.10 MMC-Number-of-Blocks Register (MMC_NBLK) 38
Number of Blocks (NBLK) 38
1.7.11 MMC-Buffer-Configuration Register (MMC_BUF) 39
Receive-DMA-Channel Enable (RX_DMA_En) 39
Buffer Almost-Full Level (AF_Level) 39
Transmit-DMA-Channel Enable (TX_DMA_En) 40
Buffer Almost-Empty Level (AE_Level) 40
1.7.12 MMC-SPI-Configuration Register (MMC_SPI) 40
Start-SPI Transfer (Start) 41
Write/Not Read (WnR) 41
Chip-Select-Hold-Time Control (TCSH) 41
Chip-Select-Setup-Time Control (TCSS) 42
Chip-Select Control (CS) 43
Chip-Select Mode (CSM) 43
Chip-Select Disable (CSD) 44
Clock Phase (PHA) 44
Clock Polarity (POL) 45
1.7.13 MMC-SDIO-Mode-Configuration Register (MMC_SDIO) 46
Card-Status-Error-on-Bit-3-of-Response-R1 Enable (CER1_3_En) 46
Data-Time-Out-Prescaler Enable (DTO_PS_En) 46
1.7.14 MMC-System-Test Register (MMC_SYST) 46
Ready/Busy Data (RDY_dat) 47
DAT[3:0] Direction (DAT_dir) 47
DAT[3:0] Data (DATn_dat) 48
CMD-Direction (CMD_dir) 48
CMD−Data (CMD_dat) 48
MMC_CLK−Data (MMC_CK_dat) 48
SPI_CLK−Data (SPI_CK_dat) 48
CS[3:1]−Data (CSn_dat) 49
1.7.15 MMC-Module-Version Register (MMC_REV) 49
Module_Version-Number (REV) 49
1.7.16 MMC/SD-Command-Response Registers 0−7 (MMC_RSP[7:0]) 49
1.8 Command Flow 51

1.9 DMA Operation 56
1.9.1 MMC DMA-Receive Mode 56
1.9.2 MMC DMA-Transmit Mode 57
1.10 MPU (IRQ/Polling) Mode 58
1.10.1 MMC MPU (IRQ/Polling) Receive Mode 58

Figures

1 MMC/SD Host Controller Environment 12
2 Clock Control 25
3 SPI Mode C/S Timings Controls (POL = 0) 42
4 SPI Mode C/S Timings Controls (POL = 1) 43
5 SPI- Master Configuration Bits 45
6 Command Flow 51
7 Initialization Phase 52
8 Detail of Basic Operation 52
9 Command Transfer 53
10 Data Transfer 54
11 Data Transfer in MMC/SD Mode Example 55

Tables

1 MMC/SD Signal Pads 14
2 MMC_CMD Pullups 16
3 MMC_DAT Pullups 16
4 MMC/SD Registers 17
5 MMC-Command Register (MMC_CMD) 18
6 MMC-Argument-Low Register (MMC_ARGL) Field Descriptions 22
7 MMC-Argument-High Register (MMC_ARGH) Field Descriptions 22
8 MMC-System-Configuration Register (MMC_CON) 22
9 MMC_CLK/SPI_CLK High-/Low-Time Computation 25
10 MMC-System-Status Register (MMC_STAT) Field Descriptions 26
11 Response Types 27
12 MMC-System-Interrupt Register (MMC_IE) Field Descriptions 33
13 MMC-Command-Time-out Register (MMC_CTO) Field Descriptions 34
14 MMC-Data-Time-Out Register (MMC_DTO) Field Descriptions 34
15 Data-Time-Out Conditions 35
16 MMC-Data-Access Register (MMC_DATA) Field Descriptions 35
17 MMC-Block-Length Register (MMC_BLEN) Field Descriptions 36
18 MMC-Number-of-Blocks Register (MMC_NBLK) Field Descriptions 38
19 MMC-Buffer-Configuration Register (MMC_BUF) Field Descriptions 39
20 MMC-SPI-Configuration Register (MMC_SPI) Field Descriptions 40
21 Chip-Select Control (SPI Mode) 44
22 MMC-SDIO-Mode-Configuration Register (MMC_SDIO) Field Descriptions 46
23 MMC-System-Test Register (MMC_SYST) Field Descriptions 47
24 MMC-Module-Version Register (MMC_REV) Field Descriptions 49
25 MMC/SD-Command-Response Register 0 (MMC_RSP0) Field Descriptions 49
26 MMC/SD-Command-Response Register 1 (MMC_RSP1) Field Descriptions 49
27 MMC/SD-Command-Response Register 2 (MMC_RSP2) Field Descriptions 50
28 MMC/SD-Command-Response Register 3 (MMC_RSP3) Field Descriptions 50
29 MMC/SD-Command-Response Register 4 (MMC_RSP4) Field Descriptions 50
30 MMC/SD-Command-Response Register 5 (MMC_RSP5) Field Descriptions 50
31 MMC/SD-Command-Response Register 6 (MMC_RSP6) Field Descriptions 50
32 MMC/SD-Command-Response Register 7 (MMC_RSP7) Field Descriptions 50

MMC/SD

This document describes the options to interface the OMAP5910 multimedia processor with MultiMedia Cards, SD Memory Cards, and serial flash cards.

1 MMC/SD Host Controller

The MMC/SD host controller provides an interface between the MPU and either the MMC or SD memory card plus up to three serial flash cards and handles the MMC/SD or SPI transactions with minimal MPU intervention.

The following combination of external devices is supported:

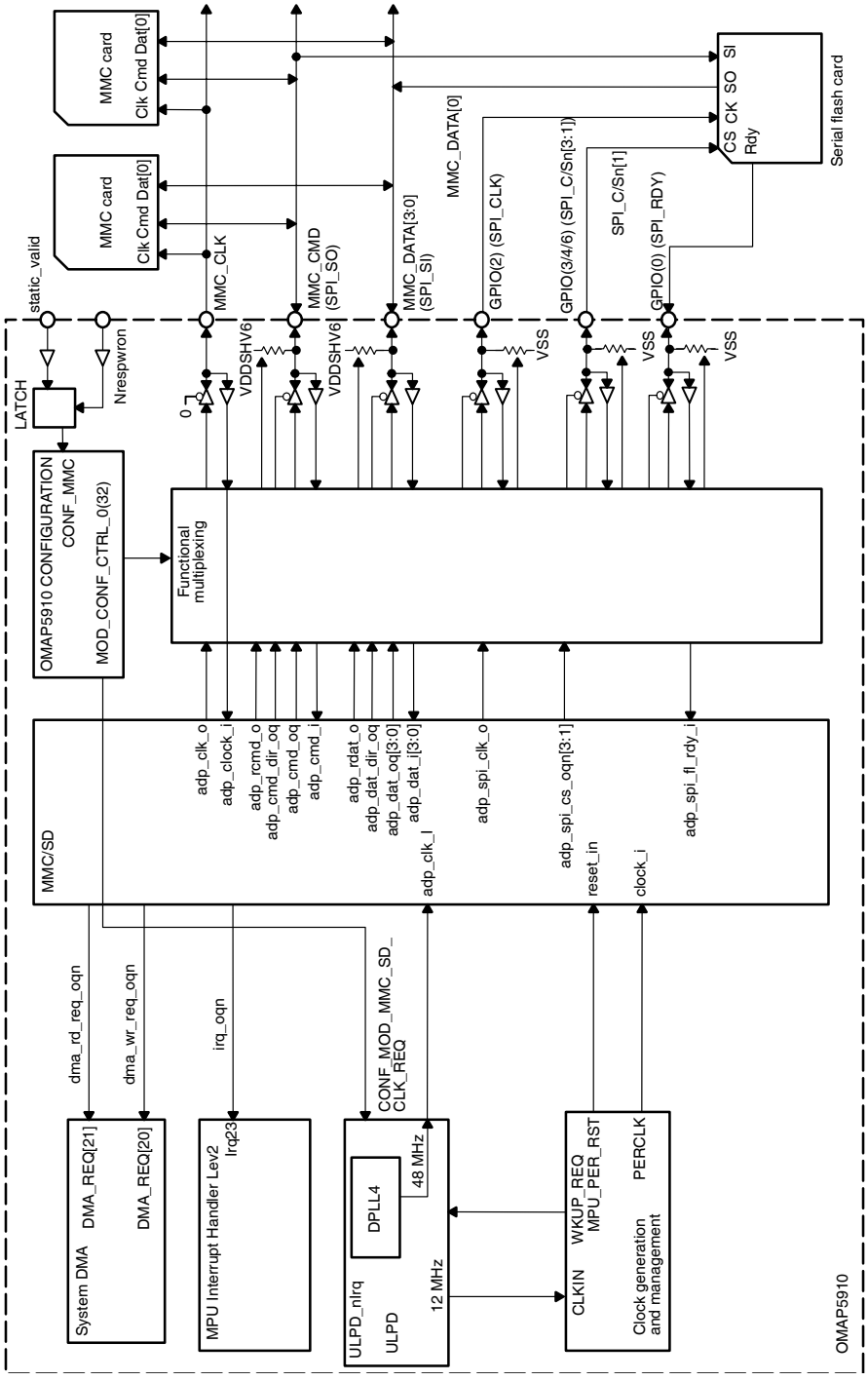
- One or more MMC memory cards sharing the same bus plus up to three devices with 8-bit SPI protocol interface (flash serial memory).
- One single-SD memory card plus up to three devices with 8-bit SPI protocol interface.

Other combinations , e.g., two SD cards, one MMC card + one SD card, are not supported.

The application interface is responsible for managing transaction semantics. The MMC/SD host controller handles the MMC/SD protocol at the transmission level, packing data, adding cyclic redundancy checking (CRC), start/end bit, and checking for syntactical correctness. The SD mode wide-bus width is also supported.

The application interface can send every MMC/SD command and either poll for the status of the adapter or wait for an interrupt request, which is sent back in case of exceptions or to warn for the end of operations. The application interface can read card responses or flag registers. It can also individually mask interrupt sources. All these operations can be performed by reading and writing the control registers.

Figure 1. MMC/SD Host Controller Environment



1.1 MMC/SD Host Controller Features

The main features of the controller are:

- Full compliance with the MMC command/response sets as defined in the MMC standard specification, *see MultiMediaCard System Specification Version 3.1 – June, 2001. MMCA Technical Committee* [1]
- Full compliance with the SD command/response sets as defined in the SD Physical Layer specification, *SD Memory Card Specifications – Part 1 Physical Layer Specification, Version 1.0 – March 2000 + Supplementary Notes Part 1 June 2000, SD Group* [2]
- Flexible architecture allowing support for the new command structure
- Separate SPI interface with 3 C/S. Provides supports for up to three serial devices such as serial flash
- Built-in 64-byte FIFO for a buffered read or write
- 16-bit-wide access bus to maximize the bus throughput
- Designed for low power usage
- Wide-interrupt capability
- Programmable clock generation
- Two DMA channels

Known limitations:

- No built-in hardware support for the correction codes (ECC)
- No built-in support for card detection
- No full compliance to the SDIO specification.

1.2 MMC/SD Host-Controller Signal Pads

The signal pads, see Table 1, describe the physical interface between the OMAP5910—the transceiver—and the target MMC/SD memory card(s) or serial flash memories.

The transceiver provides dc-level adaptation functions between the OMAP5910 device and the target devices.

The state of the OMAP5910 static-valid input during power on, determines the functional multiplexing on the OMAP5910 MMC/SD pads.

The OMAP5910 static-valid pad must be held to 1 during power on so that the MMC/SD host-controller signals are usable from the power-on reset on the OMAP5910 MMC/SD pads, see Table 1.

This functional multiplexing, which is configured in static mode, does not concern the SPI signals which are multiplexed on the GPIO pads. For these pads, the functional multiplexing is done classically by programming an OMAP5910 configuration register.

Table 1. MMC/SD Signal Pads

Pad Name	Type	Pullup/ Pulldown	Reset Value	Description
MMC.CLK	Out	-	0	The MMC/SD-card CLK signal. Only active during an active command to a MMC/SD card using the MMC or SPI protocols.
MMC.CMD_SPI. DO/SPI_SO	In-Out	Pullup (*3)	Input	The MMC/SD-card CMD signal in the MMC/SD mode. The SPI serial-out signal in the SPI mode (output—goes to the serial Input of the target device(s)).
MMC.DAT[0] /SPI_SI	In-Out	Pullup (*3)	Input	The MMC-card DAT or SD-card DAT[0] signal in the MMC/SD mode. The SPI serial-in signal in the SPI mode (input—comes from the serial Output of the target device(s)).
MMC.DAT[3-1] (*1)	In-Out	Pullup (*3)	Input	The SD-card DAT[3-1] signals in the MMC/SD mode. Reserved signals for the MMC card or in the SPI mode.
GPIO2 (SPI.CLK) (*2)	In- Out (Out)	Pulldown (disabled)	Input (0)	By default, the pad used by the GPIO2. The SPI_CLK output signal can be multiplexed. The SPI_CLK is only active during the SPI transfers to the serial flash or the other SPI devices (except the MMC/SD cards).
GPIO3 (SPI.C/Sn[3]) (*2)	In-Out (Out)	Pulldown (disabled)	Input (1)	By default, the pad used by the GPIO3. The SPI CSn(3) output signal can be multiplexed. The SPI CSn(3) is active low and is only active in the SPI mode during the SPI transfers. Reserved in the MMC/SD mode.
GPIO4 (SPI.C/Sn[2]) (*2)	In-Out (Out)	Pulldown (disabled)	Input (1)	By default, the pad used by the GPIO4. The SPI CSn(2) output signal can be multiplexed. The SPI CSn(2) is active low and is only active in the SPI mode during the SPI transfers. Reserved in the MMC/SD mode.

Notes: 1) Optional signals. Only needed for the SD cards.

2) Optional signals. Only needed for the devices with the SPI interfaces (serial flash, etc.).

3) This pullup is enabled/disabled dynamically by the MMC/SD host controller.

Table 1. MMC/SD Signal Pads

Pad Name	Type	Pullup/ Pulldown	Reset Value	Description
GPIO6 (SPI.C/Sn[1]) (*2)	In-Out (Out)	Pulldown (disabled)	Input (1)	By default, the pad used by the GPIO6. The SPI CSn(1) output signal can be multiplexed. The SPI CSn(1) is active low, only active in SPI mode during SPI transfers. Reserved in the MMC/SD mode.
GPIO0 (SPI.RDY) (*2)	In-Out (In)	Pulldown (disabled)	Input (Input)	By default, the pad used by the GPIO0. The SPI ready/busy input can be multiplexed. When the SPI_RDY is low, it denotes a busy condition. Only active in the SPI mode during the SPI transfers. Reserved signal in the MMC/SD mode

Notes: 1) Optional signals. Only needed for the SD cards.

2) Optional signals. Only needed for the devices with the SPI interfaces (serial flash, etc.).

3) This pullup is enabled/disabled dynamically by the MMC/SD host controller.

1.3 MMC/SD Host-Controller Clocks and Reset

The MMC/SD host controller has two clocks:

- An interface clock (clock_i) used between the MPU TIPB and the MMC/SD host controller and connected to the MPU peripheral programmable clock (PERCLK), is determined by dividing CK_GEN1 (the output of DPLL1) by the value associated with the PERDIV field of the ARM_CKCTL register (0xFFFECE00).

  This clock is a free-running clock when the system is awake.

- A 48-MHz functional clock (ADP_CLK_I), which is generated by the ULPD DPLL.

This clock is requested by setting to 1 the CONF_MOD_MMC_SD_CLK_REQ bit[23] of the MOD_CONF_CTRL_0 register.

The MPU-TIPB reset (MPU_PER_RST) resets the MMC/SD host controller.

1.4 MMC/SD Host Controller DMA Request

The MMC/SD host controller can use:

- Receive DMA channel (DMA_RD_REQ_OQN), which is connected to the SYSTEM DMA request [21].
- Transmit DMA channel (DMA_WR_REQ_OQN), which is connected to the SYSTEM DMA request [20].

See the *System DMA Controller Reference Guide*, SPRU674, for more details.

1.5 MMC/SD Host-Controller Interrupt

The MMC/SD controller can generate one interrupt (IRQ_OQN) which is connected to the MPU level 2 interrupt handler, line 23 (level-sensitive).

1.6 MMC/SD Internal Pullups

There are internal pullups on the following pins:

- ❏ MMC.CMD I/O pin
- ❏ MMC.DAT[3:0] I/O pins

The MMC cards work in open-drain mode on the MMC.CMD line during the identification phase, and more generally for broadcast MMC commands. Consequently, a pullup on the MMC.CMD line is needed.

When the MMC.CMD and MMC.DAT[3:0] lines work in push/pull mode, it is important to prevent bus-floating conditions. Consequently, pullups are needed.

These pullups are directly controlled by the MMC/SD host controller (adp_rcmd_o and adp_rdat_o) and are only active when required, which saves power.

Table 2 and Table 3 show the activation conditions for the MMC.CMD and MMC.DAT pullups.

Table 2. MMC_CMD Pullups

MMC_SD Host Controller Status	MMC CARD Status	MMC_CMD Pullup (Open Drain Mode)	MMC_CMD Pullup (Push/Pull Mode)
Input	Input	Active	Active
Input	Output	Active	Disabled
Output	Input	Disabled	Disabled

Table 3. MMC_DAT Pullups

MMC_SD Host Controller Status	MMC CARD Status	MMC_DAT Pullup (Both Modes)
Input	Input	Active
Input	Output	Disabled
Output	Input	Disabled

In the flash-SPI mode, when no data is on the MMC.CMD and MMC.DAT lines (input and output of the flash), the pullups are disabled.

1.7 MMC/SD Registers

Table 4 lists the MMC/SD controller registers. Table 5 through Table 32 describe the register bits.

Table 4. MMC/SD Registers

Register	Description	Access	Address
MMC_CMD	MMC command	R/W	FFFB:7800
MMC_ARGL	MMC argument low	R/W	FFFB:7804
MMC_ARGH	MMC argument high	R/W	FFFB:7808
MMC_CON	MMC system configuration	R/W	FFFB:780C
MMC_STAT	MMC status	R/W	FFFB:7810
MMC_IE	MMC system interrupt enable	R/W	FFFB:7814
MMC_CTO	MMC command time-out	R/W	FFFB:7818
MMC_DTO	MMC data time-out	R/W	FFFB:781C
MMC_DATA	MMC TX/RX FIFO data	R/W	FFFB:7820
MMC_BLEN	MMC block length	R/W	FFFB:7824
MMC_NBLK	MMC number of blocks	R/W	FFFB:7828
MMC_BUF	MMC buffer configuration	R/W	FFFB:782C
MMC_SPI	MMC serial port interface	R/W	FFFB:7830
MMC_SDIO	MMC SDIO mode configuration	R/W	FFFB:7834
MMC_SYST	MMC system test	R/W	FFFB:7838
MMC_REV	MMC module version	R	FFFB:783C
MMC_RSP0	MMC command response 0	R	FFFB:7840
MMC_RSP1	MMC command response 1	R	FFFB:7844
MMC_RSP2	MMC command response 2	R	FFFB:7848
MMC_RSP3	MMC command response 3	R	FFFB:784C
MMC_RSP4	MMC command response 4	R	FFFB:7850
MMC_RSP5	MMC command response 5	R	FFFB:7854
MMC_RSP6	MMC command response 6	R	FFFB:7858
MMC_RSP7	MMC command response 7	R	FFFB:785C
Reserved			FFFB:7860- FFFB:787C

1.7.1 MMC-Command Register (MMC_CMD)

Table 5. MMC-Command Register (MMC_CMD)

Bit	Name	Description
15	DDir	Data direction [read/write]
14	SHR	Stream command or a broadcast host response
13–12	Type	Command types [bc,bcr,ac,adtc]
11	Busy	Command with busy response [R1b]
10–8	Response	Command responses [no response, R1/R1b, R2, R3, R4, R5,R6]
7	Init	Send the initialization stream
6	OD	The card is in open-drain mode
5–0	Cmd_Index	Command index [63:0]

A write to the MMC-command register (MMC_CMD) sends a command to the card. If the MPU accesses this register byte-wise, the command is sent to the card only after a write access to the least significant bits (LSB) [7:0]. Hence, the MSB must always be written first in a byte-accessed situation.

A read has no effect except to return the last command that was previously sent.

Note:

A write into this register with Type = adtc resets the FIFO pointers and pre-fetch register. Writes with other type values (bc, bcr, ac) do not affect the FIFO contents. Hence, the data must be written inside the FIFO after sending a single or multiple-block write command.

A write into this register also clears the MMC_RSP[07] registers.

Data Direction (DDir)

This bit [15] specifies if the data transfer is a read or a write. This bit is only valid if the command type is adtc.

This bit has the same polarity as the RD/WR argument bit[0] for a GEN_CMD command (CMD56).

- ❏ 0: Data write
- ❏ 1: Data read

The value after reset is low.

Stream Command or Broadcast Host Response (SHR)

Only applies to the MMC card. The SD card does not support the stream operation or a host-generated response.

This bit [14] must be set to 1 in two cases:

- ❏ Associated with the adtc type, if the command is a stream-data transfer (read or write). A stream read is a class 1 command (CMD11: READ_DAT_UNTIL_STOP). A stream write is a class 3 command (CMD20: WRITE_DAT_UNTIL_STOP).
- ❏ Associated with the bc type, the host generates a 48-bit response instead of a command. It can be used to terminate the interrupt mode by generating a CMD40 response by the core (see Section 4.3, *Interrupt Mode,* in the MMC [1] specification).

This bit is only valid if the command type is adtc or bc.

- ❏ 0: Normal mode
- ❏ 1: Stream mode (type = adtc), host response (type = bc)

The value after reset is low.

Command Type (Type)

The encoded bits [13-12] that define the type of the command that is passed by the core to the MMC/SD memory card (see Section, 4.7.1, *Command Types*, in the MMC [1] specification or the SD [2] specifications).

- ❏ 00: bc (broadcast—no response)
- ❏ 01: bcr (broadcast with response)
- ❏ 10: ac (addressed—no data transfer)
- ❏ 11: adtc (addressed with data transfer)

Note:

Also resets the FIFO.

The values after reset are low (two bits).

Command With Busy Response (Busy)

This bit [11] must be set to 1 if the response to the command sent is of type R1b (R1 + busy).

- ❏ 0: Response without busy (R1, R2, R3, R4, R5, R6)
- ❏ 1: Response with busy (R1b)

The value after reset is low.

Command Response (Response)

The encoded bits [10-8] that define the response for the command passed by the core to the MMC/SD memory card (see Section, 4.9, *Responses*, in the MMC [1] specification or the SD 2] specification).

- ❏ 000: No response
- ❏ 001: R1/R1b (normal response command)
- ❏ 010: R2 (CID, CSD registers)
- ❏ 011: R3 (OCR register)
- ❏ 100: R4 (Fast I/O—MMC card only)
- ❏ 101: R5 (Interrupt request—MMC card only)
- ❏ 110: R6 (Published RCA response—SD card only)
- ❏ 111: Reserved

The values after reset are low (three bits).

Send Initialization Stream (Init)

When this bit [7] is set, an initialization sequence is sent prior to the command. This option can simplify acquisition of new cards. An initialization sequence consists of setting the CMD line to 1 during 80 CLK cycles (see Section 6.3, *Power-Up Description*, in the MMC [1] specification, or Section 6.4 in the SD [2] specification).

- 0: No initialization sequence (normal procedure)
- 1: Initialization sequence sent prior to command

The value after reset is low.

Card Open-Drain Mode (OD)

This bit [6] must be set to 1 if the MMC-card bus is operating in open-drain mode during the response phase to the command sent. Typically, during the card identification mode, the card is either in an idle, ready or identification state. This bit must be set for MMC card commands 1, 2, 3, and 40.

For an SD card, this bit must always be kept low, because SD cards do not have open-drain capability.

- 0: Push/pull
- 1: Open drain

The value after reset is low.

Command Index (Cmd_index)

The binary-encoded value [bits 5-0] of the decimal equivalent, 0 to 63, specifying the command number sent to the card.

- 000000: CMD0
- 000001: CMD1
- …
- 111111: CMD63

The values after reset are low (all 6 bits).

1.7.2 MMC-Argument-Low/High Registers (MMC_ARGL/MMC_ARGH)

The MMC-argument-low and -high registers specify the 32-bit argument value that is passed with the command. These registers must be initialized prior to sending the command itself to the card (write action into the MMC_CMD register). The only exception is for a command index specifying stuff bits in arguments, which makes a write unnecessary.

Table 6. MMC-Argument-Low Register (MMC_ARGL) Field Descriptions

Bits	Field	Description
15–0	ARG_low	Command argument bits [15:0]

The values after reset are low (all 16 bits).

Table 7. MMC-Argument-High Register (MMC_ARGH) Field Descriptions

Bits	Field	Description
15–0	ARG_high	Command argument bits [31:16]

The values after reset are low (all 16 bits).

1.7.3 MMC-System-Configuration Register (MMC_CON)

Table 8. MMC-System-Configuration Register (MMC_CON)

Bits	Field	Description
15	DW	Data-bus width
14	Reserved	
13–12	Mode	Operating mode select (MMC/SD, SPI, SYSTEST, or MMC SPI protocol).
11	Power-up	Power-up control
10–8	Reserved	
7–0	Clk_div	Clock divider [No clock, 1:255]

Bus Width During Data Phase (DW)

SD card only.

This bit [15] must be set following a valid SET_BUS_WIDTH command (ACMD6) with the value written in bit [1] of the argument. Prior to this command, the SD card configuration register (SCR) must be verified for the bus width supported by the SD card.

- 0: 1-bit data width (DAT[0] used)
- 1: 4-bit data width (DAT[3:0] used—SD card only).

The value after reset is low.

This bit must always be set to 0 for MMC cards or during the SPI transfer. Not setting this bit correctly can result in an unpredictable behavior.

Mode Select (Mode)

These bits [13-12] select between the MMC/SD mode, SPI mode 1, SYSTEST mode and SPI mode 2.

In the MMC/SD mode, transfers to the MMC/SD card follow the MMC protocol. The MMC clock is enabled and the SPI clock is disabled.

In the SPI mode1, transfers to up to three SPI controlled devices (serial flash, etc.) are supported. In this mode, the SPI clock is enabled and the MMC clock is disabled.

In the SYSTEST mode, the signal pins are configured as general-purpose input/output and the 64-byte FIFO is configured as a stack memory accessible only by the MPU. The pins retain their default type (input, output or in/out).

In the SPI mode 2, transfers to the MMC/SD card follow the SPI protocol. The MMC clock is enabled and the SPI clock is disabled. The MMC protocol must be implemented in software when using this mode since the MMC interface acts as a generic SPI port and does not utilize the MMC-specific features available in the MMC/SD mode.

- 00: MMC/SD mode (MMC/SD cards using MMC protocol)
- 01: SPI mode 1 (for serial flash or others SPI slave devices)
- 10: SYSTEST mode
- 11: SPI mode 2 (MMC/SD cards using SPI protocol)

The values after reset are low (2 bits).

Power Up-Control (Power_Up)

This bit [11] must be set to 1 prior to any valid transaction to either the MMC/SD or SPI memory cards.

- When 1, the card is considered powered up and the controller core is enabled.
- When 0, the card is considered powered down (system dependant) and the controller core logic in the pseudo-reset state. That is, the MMC_STAT

register flags are reset, the FIFO pointers are reset, any access to the DATA register has no effect, a write into the MMC_CMD register is ignored, and the setting of the MMC_SPI start to 1 is ignored.

- 0: Powered-down/pseudo-reset state
- 1: Powered-up/normal operation mode

The value after reset is low.

Clock Divider (Clk_div)

These bits [7-0] define the ratio between a reference-clock frequency (48 MHz) and the output-clock frequency on the CLK pin of either the memory card (MMC or SD) or other 8-bit mode SPI controlled devices.

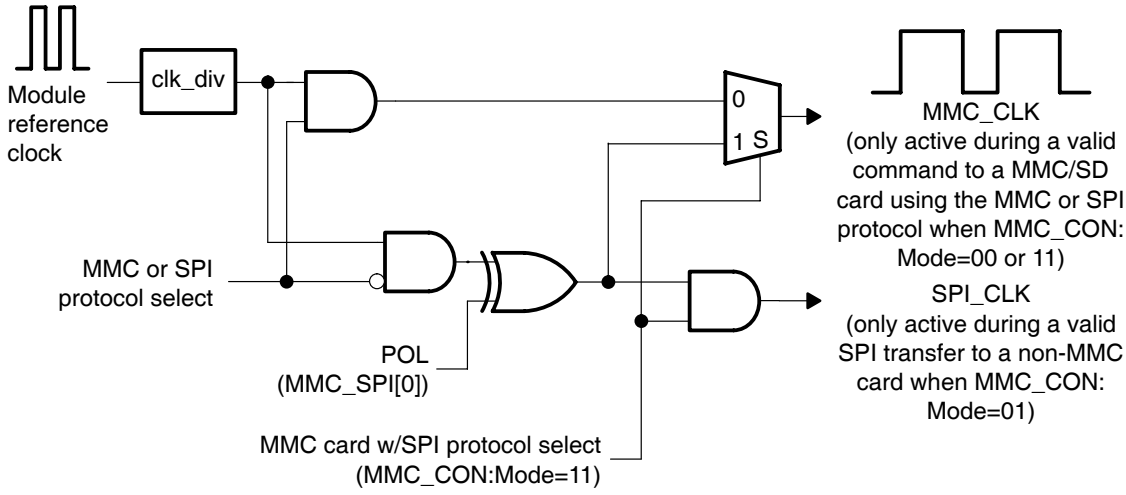
The division factor is exactly the binary-encoded-decimal value for values between 1 and 255.

A value of 0 disables the clock.

- 0x00: Clock disabled
- 0x01: Ref clk/1
- ....
- 0xFF: Ref clk/255

The values after reset are low (all 8 bits).

Figure 2. Clock Control



Notes: 1) During the identification phase, the maximum frequency on the MMC CLK line is 400 kHz (reference: *Bus Timing Specifications* in Chapter 6 of the MMC [1] specification or the SD [2] specification. That is, a value of 120 must be set into the frequency ratio register because the reference-clock frequency is 48 MHz.

Notes: 2) During the data transfer phase the maximum frequency is 16 MHz for MMC cards, 24 MHz for SD cards, and 12 MHz for the SPI serial flash cards.

Notes: 3) The duty cycles of the generated MMC_CLK and SPI_CLK clock signals depend on the Clk_div value and on the polarity setting (MMC_SPI:POL) for the SPI_CLK signal only. The low- and high-time approximate values can be computed using the set-in rules.

Table 9. MMC_CLK/SPI_CLK High-/Low-Time Computation

Clk_Div	MMC_CLK/SPI_CLK High-Time	MMC_CLK/SPI_CLK Low-Time
1	ref_clk_high_time	ref_clk_low_time
Even ≥ 2	ref_clk_per (Clk_div/2)	ref_clk_per (Clk_div/2)
Odd ≥ 3 (POL=PHA)	ref_clk_per (TRUNC[Clk_div/2] + 1)	ref_clk_per (TRUNC[Clk_div/2])
Odd ≥ 3 (POL≠PHA)	ref_clk_per (TRUNC[Clk_div/2])	ref_clk_per (TRUNC[Clk_div/2] + 1)

ref_clk_per is the reference-clock period (in ns) to the module (end-system dependant).

TRUNC is the truncate-to-an-integer number function (round down).

Example 1: Module reference clock = 48 MHz (20.83 ns), target is MMC card.

- ❏ clk_div = 3 (MMC card is 20 MHz max).
- ❏ MMC_CLK period = 62.5 ns (> 50 ns OK)
- ❏ Ideal MMC_CLK high time = 41.66 ns (>>10 ns)
- ❏ Ideal MMC_CLK low time = 20.83 ns (>>10 ns)

1.7.4 MMC-System-Status Register (MMC_STAT)

Table 10. MMC-System-Status Register (MMC_STAT) Field Descriptions

Bits	Field	Description
15	Reserved	
14	Card_Err	Card-status error exists in response
13	Card_IRQ	Card IRQ was received (following CMD40)
12	OCR_busy	OCR is busy (following CMD1 or ACMD41)
11	A_Empty	Buffer almost-empty
10	A_Full	Buffer almost-full
9	Reserved	
8	Cmd_CRC	Command-CRC error
7	Cmd_timeout	Command-response time-out (no response)
6	Data_CRC	Data-CRC error
5	Data_timeout	Data-response time-out (no response)
4	EOF_Busy	Card exit busy state
3	Block_RS	Block received/sent
2	Card_Busy	Card enter busy state
1	Reserved	
0	End_of_Cmd	End-of-command phase

Common to all bits:

- ❏ The MPU can only clear a set-bit location by writing a 1 into the bit location. A write of 0 has no effect.
- ❏ When a bit location is set to 1 by the core, an interrupt is signaled to the MPU if the interrupt was enabled.

Card-Status Error (Card_Err)

MMC/SD mode only.

The core automatically sets this bit [14] when there is at least one error in a response of type R1, R1b or R6. Only the bits referenced as a type E (error) can set a card status error (see Table 11).

Table 11. Response Types

Response Type	Card Status Bits With Error	Response Register Significant Bits	Comments
R1 (MMC, SD)	31-26, 24-16,	MMC_RSP7[15:10,8-0]	These 15 bits can all generates errors.
	3* (opt)	MMC_RSP6[3]	This bit can also generate an error if enabled (bit 3 if MMC_SDIO[13]=1) per the SD application specification
R6 (SD)	15:13, 3	MMC_RSP6[15:13,3]	Corresponds to 23, 22, 19, 3 card status errors

The error handler must parse the response registers to understand the source of the error.

Other responses (types R2/R3/R4/R5) do not trigger a card-status error.

In the SPI or SYSTEST modes, this bit has no meaning and always reads as 0.

- ❏ 0: No action or no error
- ❏ 1: Error occurred

The value after reset is low.

Card IRQ (Card_IRQ)

MMC mode only.

The core automatically sets this bit [13] when a card is in the interrupt mode and exits the Wait_IRQ state (irq) by asserting a 0 on the CMD line (cards are in the open-drain mode). Only the Class 9 MMC cards can be put into the interrupt mode when in the stand-by state using a GO_IRQ_STATE (CMD40) command (see Section 4.3, *Interrupt Mode Description,* of the MMC [1] specification. SD cards do not support the interrupt mode.

In the SPI or SYSTEST modes, this bit has no meaning and always reads as 0.

- ❏ 0: No action or idle
- ❏ 1: Card exits the IRQ state

The value after reset is low.

OCR Busy (OCR_busy)

MMC/SD mode only.

The core automatically sets this bit [12] after a SEND_OP_COND (CMD1) or a SD_APP_OP_COND (ACMD41) command when one or more cards have not yet completed power up. When this bit is set, the CMD1/ACMD41 command must be repeated until the card stops responding with a busy condition. A low value on bit 31 of the OCR register indicates a busy condition. (See Section 6.3, *Power-Up Description*, of the MMC [1] specification or Section 6.4 of the SD [2] specification.).

In the SPI or SYSTEST modes, this bit has no meaning and always reads as 0.

- ❏ 0: No action or card is powered-up
- ❏ 1: OCR is busy

The value after reset is low.

Buffer Almost-Empty (A_Empty)

The core automatically sets this bit [11] during a write operation to the card when the level is below the threshold value set in the MMC_BUFF:AE_Level register bits. It indicates that the memory card has emptied the buffer to the specified level and that the MPU is able to write more data into the buffer.

If the DMA transmit mode is enabled, this bit is never set; instead a DMA request to the main DMA controller of the system is generated.

The A_Empty status bit and DMA-TX request are generated under the same conditions. This bit is set initially when a new block-write command is sent to the card. Also, once set, the core internally masks a new set condition till the MPU has performed [AE_Level+ 1] write access(es) to the FIFO.

The AE_Level is the decimal equivalent set to a binary value (0–31).

- ❏ 0: No action or the buffer is equal to or above the almost empty level.
- ❏ 1: Buffer is almost empty

The value after reset is low.

Buffer Almost-Full (A_Full)

The core automatically sets this bit [10] during a read operation to the card when the level is above the threshold value set in the MMC_BUFF:AF_Level register bits. This bit indicates that the memory card has filled out the buffer to the specified level and that the MPU needs to empty the buffer by reading it.

If the DMA-receive mode is enabled, this bit is never set; instead a DMA request to the main DMA controller of the system is generated.

The A_Full status bit and DMA-RX request are generated under the same conditions. Once set, the core internally masks a new set condition till the MPU has performed [AF_Level +1] read access(es) from the FIFO.

The AF_Level is the decimal equivalent set to a binary value (0 – 31).

- ❏ 0: No action or the buffer is below or equal to the almost full level.
- ❏ 1: Buffer is almost full

The value after reset is low.

Command-CRC Error (Cmd_CRC)

MMC/SD mode only.

The core automatically sets this bit [8] if there is a CRC7 error in the command response (bits 7:1 of all the response types except type R3). A CMD1 (MMC) or ACDM41 (SD) cannot trigger a CRC 7 error.

In the SPI or SYSTEST modes, this bit has no meaning and always reads as 0

- ❏ 0: No action or no CRC7 error
- ❏ 1: CRC7 error

The value after reset is low.

Command-Time-Out Error (Cmd_timeout)

MMC/SD mode only.

The core automatically sets this bit [7] if the card does not respond within the specified number of command-time-out clock cycles (CTO) that is set in the MMC_CTO register (see the N_{CR} timing requirements) to any command requiring a response.

If this bit is set after a command time-out, clearing this bit automatically stops the MMC clock and forces the controller state to idle.

In the SPI or SYSTEST modes, this bit has no meaning and always reads as 0.

- ❏ 0: No action or no command time-out
- ❏ 1: Command time-out

The value after reset is low.

Data-CRC Error (Dat_CRC)

MMC/SD mode only.

The core automatically sets this bit [6] if there is a CRC16 error in the data-phase response following a block-read command (single or multiple) or if there is a 3-bit CRC status token error 101 to signal for a data transmission error during a block-write command (single or multiple). For a multiple block transfer, the CRC is checked for every block.

In the SPI or SYSTEST modes, this bit has no meaning and always reads as 0.

- ❏ 0: No action or no CRC error
- ❏ 1: CRC16 error (read), 3-bit CRC token error (write)

The value after reset is low.

Data-Time-Out Error (Dat_timeout)

The core automatically sets this bit [5] if the card does not respond within the specified number of data-time-out clock cycles (DTO) that is set in the MMC_DTO register.

In the SPI mode, this bit also is set if the RDY/BUSY signal remains asserted in the busy condition for DTO consecutive clock cycles.

If this bit is set after a data time-out, a clear of this bit automatically stops the MMC or SPI clock and forces the controller state to idle.

In the SYSTEST mode, this bit has no meaning and always reads as 0.

- ❏ 0: No action or no data time-out.
- ❏ 1: Data time-out

The value after reset is low.

Card-Exit-Busy State (EOF_Busy)

MMC/SD mode only.

The core automatically sets this bit [4] when the addressed card releases the DAT line from its busy state (low level = busy). This bit can only get set during a programming phase (write operation) to a MMC or SD memory card.

In the SPI or SYSTEST modes, this bit has no meaning and always reads as 0.

- ❏ 0: No action
- ❏ 1: Data line is released/exit busy state

The value after reset is low.

Block-Received/Sent (Block_RS)

The core automatically sets this bit [3] at the end of a block transfer (read or write).

In the MMC or SD mode, this bit is set when the block transfer completes with no error. If a CRC error occurs, this bit is not set; instead a data CRC error is set to 1. For either multiple-block or stream transfers, this bit is set only once after the last successful block transfer (when the MMC_NBLK:NBLK decrements down to 0) or until interrupted by a stop command.

In the SPI mode, this bit is set when either the read or write command completes *(MMC_BLEN:BLEN decrements down-to 0).*

There is a distinction to be made between the DMA and non-DMA receive operation.

In the non-DMA-RX mode, this bit is set after the very last byte has been received in the FIFO. At this stage, the FIFO is not empty and must be read by the MPU till it becomes empty before sending a new command.

In the DMA-RX mode, this bit is set after both the last byte has been received and the FIFO is empty.

In the SYSTEST mode, this bit has no meaning and always reads as 0.

- ❏ 0: No action
- ❏ 1: Block is received/sent

The value after reset is low.

Card-Enter-Busy State (Card_Busy)

MMC/SD mode only.

The core automatically sets this bit [2] when the addressed card asserts the DAT line to a low level during a programming phase (write operation) to a MMC or SD memory card. For the MMC card only, the user can optionally use this interrupt to deselect the card (which continues to program) and select another card.

In the SPI or SYSTEST modes, this bit has no meaning and always reads as 0.

- ❏ 0: No action
- ❏ 1: Data line is asserted low/card busy

The value after reset is low.

End-of-Command (End_of_Cmd)

MMC/SD mode only.

The core automatically sets this bit [0] at the end of a successful command/response sequence or at the end of a command without a response. This bit is not set in case of a card-status error.

In the SPI or SYSTEST modes, this bit has no meaning and always reads as 0.

- ❏ 0: No action
- ❏ 1: End of command/response sequence

The value after reset is low.

When a CMD12 command is transferred after a multiple-block read, the End_of_Cmd bit [0] is not set. Alternatively, the Card_Busy bit [2] is set. After this, the EOF_Busy bit [4] cannot be set. Avoid this condition with software by using the following sequence:

1) Mask the busy interrupt before the CMD12 command is sent, because it is possible that the unexpected busy interrupt to the MPU is generated.

2) Clear the Card_Busy [2] after the response for CMD12 is returned, because it is possible that this flag is set.

3) Enable the busy interrupt if it is to be used (write 0xFFFF to the MMC_IE register).

1.7.5 MMC-System-Interrupt Register (MMC_IE)

Table 12. MMC-System-Interrupt Register (MMC_IE) Field Descriptions

Bits	Field	Description
15	Reserved	
14	Card_Err_IE	Card status error interrupt enable
13	Card_IRQ_IE	Card IRQ interrupt enable
12	OCR_busy_IE	OCR busy interrupt enable
11	A_Empty_IE	Buffer almost-empty interrupt enable
10	A_Full_IE	Buffer almost-full Interrupt enable
9	Reserved	
8	Cmd_CRC_IE	Command CRC-error interrupt enable
7	Cmd_timeout_IE	Command response time-out Interrupt enable
6	Data_CRC_IE	Data CRC-error interrupt enable
5	Data_timeout_IE	Data response time-out interrupt enable
4	EOF_Busy_IE	Card exit busy state interrupt enable
3	Block_RS_IE	Block received/sent interrupt enable
2	Card_Busy_IE	Card enter busy state interrupt enable
1	Reserved	
0	End_of_Cmd_IE	End-of-command interrupt enable

Common to all bits:

- When a bit location is set to 1 by the MPU, an interrupt is signaled to the MPU if the corresponding bit location in MMC_STAT register is asserted to 1 by the core.
- If set to 0, the interrupt is masked and not signaled to the MPU.
  - 0: Interrupt disabled
  - 1: Interrupt enabled
- The values after reset are low (all bits).

1.7.6 MMC-Command-Time-out Register (MMC_CTO)

The 16-bit MMC-command-time-out register (MMC_CTO) specifies the maximum number of clock cycles before a command time-out condition occurs.

Table 13. MMC-Command-Time-out Register (MMC_CTO) Field Descriptions

Bits	Field	Description
15–8	Reserved	
7–0	CTO	MMC-command time-out value.

Command-Time-out Value (CTO)

MMC/SD mode only.

The MPU sets this field (bits 7:0) based on the N_{CR} clock cycles. The MMC and SD card specifies the N_{CR} to be between 2 and 64 clock cycles.

If the card does not respond within the specified number of cycles, the command time-out gets set to 1 in the MMC_STAT[7] register bit.

For the MMC-card interrupt-mode support, this time-out is disabled when the command passes with an R5 response (CMD40).

- 0x00: Command time-out is disabled
- 0x01: One clock cycle
- 0xFD: 253 clocks cycles ($2^8 - 3$)

  The 0xFF and 0xFE cannot be used.

The values after reset are low (all 8 bits).

1.7.7 MMC-Data-Time-Out Register (MMC_DTO)

This 16-bit register specifies the maximum number of clock cycles before a data-time-out condition occurs.

Table 14. MMC-Data-Time-Out Register (MMC_DTO) Field Descriptions

Bits	Field	Description
15–0	DTO	Data read time-out

Data-Time-Out Value (DTO)

In the MMC/SD mode, the MPU sets this field (bits 15-0) based on the N_{AC} clock cycles. The N_{AC} is computed from the parameters TAAC and NSAC and the operating clock frequency.

The TAAC and NSAC are CSD card parameters and can be obtained by reading the response register after a successful execution of a SEND_CSD command (CMD9).

If the card does not respond within the specified number of cycles, the data-time-out value gets set to 1 in the MMC_STAT[5] register bit.

The effective number of clock cycles for the time-out value is to be multiplied by 1024 if the MMC_SDIO:DTO_PS_En=1 and by 1 if DTO_PS_En=0.

In the SPI mode, a data-time-out condition is also generated if the RDY/BUSY signal is asserted low (BUSY) for DTO consecutive clocks cycles (see Table 15).

Table 15. Data-Time-Out Conditions

DTO	DTO_PS_En=0	DTO_PS_En=1	
0x0000	No time-out	No time-out	
0x0001	1	1024	MMC clock cycles
0x0002	2	2048	
…	…	…	
0xFFFF	65535 (2^{16}-1)	67107840 (2^{26}-2^{10})	

The values after reset are low (all 16 bits).

The MMC-data-access register (MMC_DATA) is the entry point for the MPU to read data from, or write data into, the FIFO buffer. The FIFO size is 32x16bits (64 bytes). Bytes within a word are stored and read in the little endian format.

If the MPU accesses this register byte-wise, the MSB (bits [15:8]) must be always written/read first. A byte access to the LSB without a prior write into the MSB results in having the MSB filled with 0x00.

1.7.8 MMC-Data-Access Register (MMC_DATA)

Table 16. MMC-Data-Access Register (MMC_DATA) Field Descriptions

Bits	Field	Description
15–0	DATA	Transmit/receive FIFO data

Transmit/Receive-FIFO-Data Value (DATA)

In the MMC/SD mode, this register field (bits 15-0) contains either the data packet associated with a block transfer (read or write), the CID contents for a PROGRAM_CID (CMD26) command, or the CSD contents for a PROGRAM_CSD (CMD27) command.

Since the block length is passed as an argument, it is legal for the MPU to perform only 16-bit accesses (read or write) to the buffer, even if the block length is not an even number. In case of an odd number of bytes to read, the upper byte of the last access always reads as 0x00. Conversely, for an odd number of bytes to write, the upper byte must be filled with 0x00 for the last data value.

In the SPI mode, the register contains both the command (op-code and address for a serial flash) and the data.

In the SYSTEST mode, the FIFO behaves as a stack accessible only by the MPU (push and pop operations). In this mode, the set FIFO threshold values are active, as are the associated interrupts and DMA, if enabled. This special mode can be used for system test purposes.

The values after reset are low (all 16 bits).

A read access when the buffer is empty returns the previous read data value. A write access when the buffer is full is ignored. In both events, the FIFO pointers are not updated and a remote access error (hardware error) is generated (access qualifier). No remote error is generated if the MPU performs a 16-bit access if the buffer contains a single byte.

1.7.9 MMC-Block-Length Register (MMC_BLEN)

This register configures the core for the number of bytes to read or write. It must be initialized at least once prior to starting an MMC, SD, or SPI block-data transfer (read or write).

Table 17. MMC-Block-Length Register (MMC_BLEN) Field Descriptions

Bits	Field	Description
15–11	Reserved	
10–0	BLEN	Block-length value

Block Length (BLEN)

General operation: A write into this register (bits 10-0) initializes an 11-bit counter that decrements by 1 after each byte is transferred. A read into this register returns the number of bytes remaining to be transferred. When the counter reaches 0 and after the last byte transfer completes, BLEN is automatically reloaded to its programmed value by the core.

In the MMC/SD mode, this 11-bit value specifies the data-block length. This value must be set respectively with max $2^{READ\_BL\_LEN}$-1 for a block read or max $2^{WRITE\_BL\_LEN}$-1 for a block write.

READ_BL_LEN and WRITE_BL_LEN are CSD register settings of the card returned in a response, R2, following a SEND_CSD command (CMD9).

In the SPI modes and for a read transaction, the BLEN must be initialized with the exact byte-count to read negative 1, excluding the op-code and address arguments.

The op-code and address arguments that are passed to the SPI device must be written into the FIFO buffer prior to starting the SPI transfer. The BLEN starts to decrement as soon as the buffer contents have been shifted out to the SPI device. The buffer then starts to be filled with the data received from the SPI device.

In the SPI modes and for a write transaction, the BLEN must be initialized with the exact byte-count to write negative 1, including the number of bytes needed to pass the op-code and address arguments.

It is recommended to have the op-code and addresses that are passed to the SPI module written into the FIFO buffer prior to starting the SPI transfer. This allows a DMA write operation to access only the data portion. BLEN starts to decrement for every byte shifted out to the SPI device.

- 0x000: 1 byte
- 0x7FF: 2048 bytes

The values after reset are low (all 11 bits).

1.7.10 MMC-Number-of-Blocks Register (MMC_NBLK)

This register configures the number of blocks for a multiple-block data transfer (read or write) operation for the MMC/SD cards. This register is not used for the SPI transfers.

Table 18. MMC-Number-of-Blocks Register (MMC_NBLK) Field Descriptions

Bits	Field	Description
15–11	Reserved	
10–0	NBLK	Number-of-blocks value

Number of Blocks (NBLK)

MMC/SD mode only.

In the MMC/SD mode, this 11-bit value (bits 10-0) specifies the number of blocks for a multiple-block data transfer (read or write). Each block is of size MMC_BLEN:BLEN (block length value). This value must be set with the number of blocks – 1.

This register must be programmed prior to any multiple-block data transfer. A write into this register initializes an 11-bit counter that decrements by one after each block transfer. A read into this register returns the number of blocks remaining to be transferred to the card.

When the counter reaches 0, the transfer stops after the last transfer completes.

For stream or multiple-block transfers, a Block_RS interrupt is generated only once after the last successful transfer when NBLK reaches 0.

In stream mode, the minimum allowable number of blocks is two.

Note:

This value must be 0x000 for a single-block transfer. In stream mode, the minimum allowable number of blocks is two blocks. If the transfer is interrupted by a STOP_TRANSMISSION command (CMD12) before the counter reached 0, this register must be reprogrammed prior to starting any new single or multiple-block data transfers.

- 0x000: 1 block
- 0x7FF: 2048 blocks

The values after reset are low (all 11 bits).

1.7.11 MMC-Buffer-Configuration Register (MMC_BUF)

This register configures the buffer-threshold level of the thirty-two 16-bit-word FIFO and enables the DMA transfers.

Table 19. MMC-Buffer-Configuration Register (MMC_BUF) Field Descriptions

Bits	Field	Description
15	RX_DMA_En	Receive DMA-channel enable
14–13	Reserved	
12–8	AF_Level	Buffer almost-full level
7	TX_DMA_En	Transmit DMA-channel enable
6–5	Reserved	
4–0	AE_Level	Buffer almost-empty level

Receive-DMA-Channel Enable (RX_DMA_En)

When this bit [15] is set to 1, the receive DMA channel is enabled and the A_Full status bit is forced to 0 by the core irrespectively of the AF_level setting (see Table 19).

- ❏ 0: Receive DMA-channel disabled
- ❏ 1: Receive DMA-channel enabled

The value after reset is low.

Buffer Almost-Full Level (AF_Level)

This register (bits 12-8) holds the programmable almost-full level value used to determine the almost-full buffer condition. If an interrupt or a DMA read request is to be issued during a read operation when the data buffer holds n words of 16 bits, then the AF_Level must be set with n-1.

- ❏ 0x00 1 16-bit word (2 bytes)
- ❏ 0x1E: 31 16-bit word (62 bytes)
- ❏ 0x1F: 32 16-bit word (64 bytes)

The values after reset are 0x1F.

Transmit-DMA-Channel Enable (TX_DMA_En)

When this bit [7] is set to 1, the transmit-DMA channel is enabled and the A_Empty status bit is forced to 0 by the core irrespectively of the AE_level setting (see Table 19).

More information regarding DMA operation can be found in the *System DMA Controller Reference Guide*, SPRU674.

- ❏ 0: Transmit DMA-channel disabled
- ❏ 1: Transmit DMA-channel enabled

The value after reset is low.

Buffer Almost-Empty Level (AE_Level)

This register (bits 4-0) holds the programmable almost-empty level value used to determine the almost-empty buffer condition. If an interrupt or a DMA write request is to be issued during a write operation when the data buffer holds n words of 16 bits, then the AE_Level must be set with n.

- ❏ 0x00: 0 16-bit word (0 bytes)
- ❏ 0x1E: 30 16-bit word (60 bytes)
- ❏ 0x1F: 31 16-bit word (62 bytes)

The value after reset is low.

1.7.12 MMC-SPI-Configuration Register (MMC_SPI)

This register is used to configure the SPI interface and start an SPI transfer if the SPI mode has been enabled.

Table 20. MMC-SPI-Configuration Register (MMC_SPI) Field Descriptions

Bits	Field	Description
15	Start	Start SPI transfer
14	WnR	Write/not read
13–12	Reserved	
11–10	TCSH	Chip-select hold-time control
9–8	TCSS	Chip-select setup-time control
7–6	Reserved	
5–4	CS	Chip-select control

Table 20. MMC-SPI-Configuration Register (MMC_SPI) Field Descriptions (Continued)

Bits	Field	Description
3	CSM	Chip-select mode
2	CSD	Chip-select disable
1	PHA	Phase control
0	POL	Polarity control

Start-SPI Transfer (Start)

This set-only bit [15] always reads as 0. A write to 0 has no effect.

When set to 1 by the MPU, a SPI transfer is automatically started.

Note:

The user must take care to initialize the MMC_BLEN:BLEN before starting an SPI transfer.

The SPI transfer automatically stops when the size programmed in the MMC_BLEN:BLEN decrements down to 0 (in read and in write).

- ❏ 0: No action
- ❏ 1: SPI transfer started

The value after reset is low.

Write/Not Read (WnR)

This bit [14] instructs the 11-bit block-length counter in the MMC_BLEN:BLEN to decrement either on a byte read when WnR = 0 or on a byte write when WnR = 1.

- ❏ 0: Decrement on byte received
- ❏ 1: Decrement on byte sent

The value after reset is low.

Chip-Select-Hold-Time Control (TCSH)

This field (bits 11-10) defines the number of interface clock cycles that the core waits after the last SPI_CLK clock edge before asserting the chip-select signals to their inactive-high level.

- 00: Minimum 0.5 clock cycle
- 01: Minimum 1.5 clock cycles
- 10: Minimum 2.5 clock cycles
- 11: Minimum 3.5 clock cycles

Values after reset are low (2 bits).

Chip-Select-Setup-Time Control (TCSS)

This field (bits 9-8) defines the number of interface clock cycles that the core waits after asserting the chip-select signals to their active-low level before asserting the first SPI_CLK clock edge.

- 00: Minimum 1 clock cycle
- 01: Minimum 2 clock cycles
- 10: Minimum 3 clock cycles
- 11: Minimum 4 clock cycles

The values after reset are low (2 bits).

Figure 3. SPI Mode C/S Timings Controls (POL = 0)

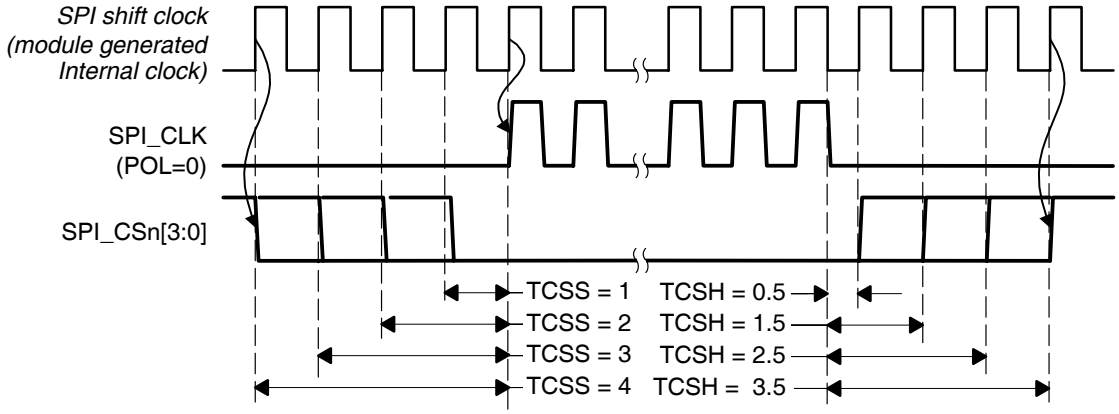
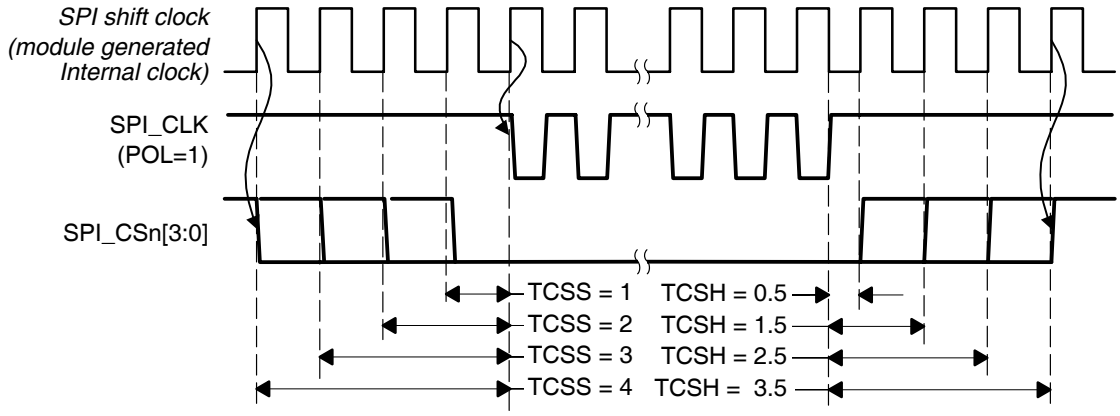


Figure 4. SPI Mode C/S Timings Controls (POL = 1)



Chip-Select Control (CS)

Encoded value (bits 5-4) that selects the device being targeted for the SPI transfer.

- ❏ 00: Reserved (no device is selected)
- ❏ 01: C/S 1
- ❏ 10: C/S 2
- ❏ 11:C/S 3

The value after reset is low (2 bits).

Chip-Select Mode (CSM)

When this bit [3] is set to 0 and enabled (CSD=0), the selected CS-signal pin goes active (low) only when the SPI transfer is started and brought back automatically to its inactive state (high), when the SPI transfer completes.

When set to 1, the automatic control of the CS signal is disabled. Instead, the selected CS-signal pin is manually controlled by the chip-select disable register bit (CSD). This mode provides support for the complex SPI transfer scheme that requires the CS to be kept active during the entire transfer (e.g., the MMC card write with busy condition).

- ❏ 0: Automatic mode
- ❏ 1: Manual mode (controlled by CSD)

The value after reset is low.

Chip-Select Disable (CSD)

When this bit [2] is set to 0, the selected CS signal is asserted to its active (low) state either automatically when the CSM = 0, or manually when the CSM = 1.

When set to 1, the selected CS signal is forced to its inactive (high) state. It can be used to send dummy clocks with the CS inactive to a MMC or SD card.

- 0: Selected CS is conditionally asserted (low).
- 1: Selected CS is deasserted (high).

The value after reset is low.

Table 21. Chip-Select Control (SPI Mode)

CSM	CSD	Selected CS	Comment
0	0	High-low-high	Automatic mode: CS asserted active (low) during the SPI transfer.
0	1	High	Automatic mode: CS forced inactive (high)
1	0	Low	Manual mode: CS asserted active (low)
1	1	High	Manual mode: CS asserted inactive (high)

Clock Phase (PHA)

The clock polarity and clock phase bits select the four different clocking schemes for the SPICLK pin.

The clock phase bit [1] selects a half-cycle delay for the clock.

When the clock phase = 0:

- MSB data is ready one half-cycle of the SPICLK before the SPI clock starts.
- Data is shifted-in in reception on the first-edge transition of the SPICLK.
- Data is shifted-out in transmission on the second-edge transition of the SPICLK.

When the clock phase = 1:

- Data is shifted-out in transmission on the first-edge transition of the SPICLK.

Data is shifted-in in reception on the second-edge transition of the SPICLK:

- 0: Phase 0
- 1: Phase 1

The value after reset is low.

Clock Polarity (POL)

The clock-polarity bit [0] selects the active edge of the clock, either rising or falling.

When 0, the idle value of the SPI clock signal is low and the rising edge is active.

When 1, the idle value of the SPI clock signal is high and the falling edge is active.

- ❏ 0: Rising edge active
- ❏ 1: Falling edge active

The value after reset is low.

Figure 5. SPI- Master Configuration Bits

SPI MASTER configuration			Shift in / Shift out
POL	PHA	SPI mode	
0	0	0	SPI_CLK
0	1	1	SPI_CLK
1	0	2	SPI_CLK
1	1	3	SPI_CLK

1.7.13 MMC-SDIO-Mode-Configuration Register (MMC_SDIO)

This register provides additional controls for the MMC/SD interface. It is also reserved for future SDIO operation (not supported in the present version).

Table 22. MMC-SDIO-Mode-Configuration Register (MMC_SDIO) Field Descriptions

Bits	Field	Description
15–14	Reserved	
13	CER1_3_En	Card-status error on bit 3 of response 1 enable
12–6	Reserved	
5	DTO_PS_En	Data-time-out prescaler enable
4–0	Reserved	

Card-Status-Error-on-Bit-3-of-Response-R1 Enable (CER1_3_En)

This bit [13] must be set to 1 for the SD cards only or for application-specific commands that generate an error.

If set to 1, a card-status error is generated if bit 3 of the status is 1 for a R1 or R1b response.

- ❏ 0: Error on bit 3 masked
- ❏ 1: Card-status errors on bit 3 of response 1 enabled (SD card or application specific only)

The value after reset is low.

Data-Time-Out-Prescaler Enable (DTO_PS_En)

When this bit [5] is set to 1 by the MPU, the data time-out set in the MMC_DTO register is x1024 the number of the MMC_CLK cycles.

- ❏ 0: x1 (Prescaler off)
- ❏ 1: x1024 (Prescaler on)

The value after reset is low.

1.7.14 MMC-System-Test Register (MMC_SYST)

The MMC-system-test register (MMC_SYST) is used to control the signals that connect to the I/O pins when the module is configured in the system-test (SYSTEST) mode (see Table 23).

Table 23. MMC-System-Test Register (MMC_SYST) Field Descriptions

Bits	Field	Description
15–14	Reserved	
13	RDY_dat	Ready/busy input signal data value
12	DAT-dir	DAT[3–0] signals direction
11	DAT3_dat	DAT3 input/output signal data value
10	DAT2_dat	DAT2 input/output signal data value
9	DAT1_dat	DAT1 input/output signal data value
8	DAT0_dat	DAT0/SI input/output signal data value
7	CMD-dir	CMD/SO signal direction
6	CMD_dat	CMD/SO input/output signal data value
5	MMC_CK_dat	MMC clock output signal data value
4	SPI_CK_dat	SPI clock output signal data value
3	CS3_dat	C/S3 output signal data value
2	CS2_dat	C/S2 output signal data value
1	CS1_dat	C/S1 output signal data value
0	Reserved	

Ready/Busy Data (RDY_dat)

This read-only bit [13] returns the value of the signal on the input pad (high or low).

- ❑ 0: Ready/busy low
- ❑ 1: Ready/busy high

The value after reset is high.

DAT[3:0] Direction (DAT_dir)

When set, this bit [12] places all the in/out DAT[3:0] pins in the output mode.

- ❑ 0: Input
- ❑ 1: Output

The value after reset is low.

DAT[3:0] Data (DATn_dat)

If the DAT_dir = 0 (input mode direction), these bits (11-8) return the value on the corresponding DAT pins (high or low). A write into these bits has no effect.

If the DAT_dir =1 (output mode direction), the DAT pins are driven high or low according to the value written into these register bits.

The values after reset are low (all 4 bits).

CMD-Direction (CMD_dir)

When set, this bit [7] places the in/out CMD pin in the output mode.

- ❏ 0: Input
- ❏ 1: Output

The value after reset is low.

CMD–Data (CMD_dat)

If the CMD_dir=0 (input mode direction), this bit [6] returns the value on the CMD pin (high or low). A write into this bit has no effect.

If the CMD_dir =1 (output mode direction), the CMD pin is driven high or low according to the value written into this register bit.

The value after reset is low.

MMC_CLK–Data (MMC_CK_dat)

The MMC_CK pin is driven high or low according to the value written into this register bit (5).

The value after reset is low.

SPI_CLK–Data (SPI_CK_dat)

The SPI_CK pin is driven high or low according to the value written into this register bit (4).

The value after reset is low.

CS[3:1]–Data (CSn_dat)

The CS[3:1] pins are driven high or low according to the values written into these register bits (3-1).

The values after reset are low (all 3 bits).

1.7.15 MMC-Module-Version Register (MMC_REV)

The read-only MMC-module-version register (MMC_REV) contains the revision number of the module. A write to this register has no effect.

Table 24. MMC-Module-Version Register (MMC_REV) Field Descriptions

Bits	Field	Description
15–8	–	Reserved
7–0	REV	Module-version number

Module_Version-Number (REV)

This 8-bit field (bits 7-0) indicates the revision number of the RTL for this module. This value is fixed by hardware.

The four LSBs indicate a minor revision.

The four MSBs indicate a major revision.

- 0x14: Version 1.4
- 0x20: Version 2.0

A reset has no effect on the value returned.

1.7.16 MMC/SD-Command-Response Registers 0–7 (MMC_RSP[7:0])

Table 25 through Table 32 describe the 16-bit registers that hold the specified bits positions for a 128-bit response of type R2.

Table 25. MMC/SD-Command-Response Register 0 (MMC_RSP0) Field Descriptions

Bits	Field	Description
15 – 0	RESP0	CMD response (R2[15:0])

Table 26. MMC/SD-Command-Response Register 1 (MMC_RSP1) Field Descriptions

Bits	Field	Description
15–0	RESP0	CMD response (R2[31:16])

Table 27. MMC/SD-Command-Response Register 2 (MMC_RSP2) Field Descriptions

Bits	Field	Description
15–0	RESP0	CMD response (R2[47:32])

Table 28. MMC/SD-Command-Response Register 3 (MMC_RSP3) Field Descriptions

Bits	Field	Description
15–0	RESP0	CMD response (R2[63:48])

Table 29. MMC/SD-Command-Response Register 4 (MMC_RSP4) Field Descriptions

Bits	Field	Description
15–0	RESP0	CMD response (R2[79:64])

Table 30. MMC/SD-Command-Response Register 5 (MMC_RSP5) Field Descriptions

Bits	Field	Description
15–0	RESP0	CMD response (R2[95:80])

Table 31 and Table 32 describe the registers that also hold the specified bit positions for a 32-bit response of type R1/R1b/R3/R4/R5/R6.

Table 31. MMC/SD-Command-Response Register 6 (MMC_RSP6) Field Descriptions

Bits	Field	Description
15–0	RESP6	CMD response (R2[111:96], R1/R1b/R3/R4/R5/R6[23:8])

Table 32. MMC/SD-Command-Response Register 7 (MMC_RSP7) Field Descriptions

Bits	Field	Description
15–0	RESP6	CMD response (R2[111:96], R1/R1b/R3/R4/R5/R6[39:24])

1.8 Command Flow

To correctly drive the MMC/SD adapter for a command execution, the MPU must follow the process shown in Figure 6 and Figure 7.

Figure 6. Command Flow

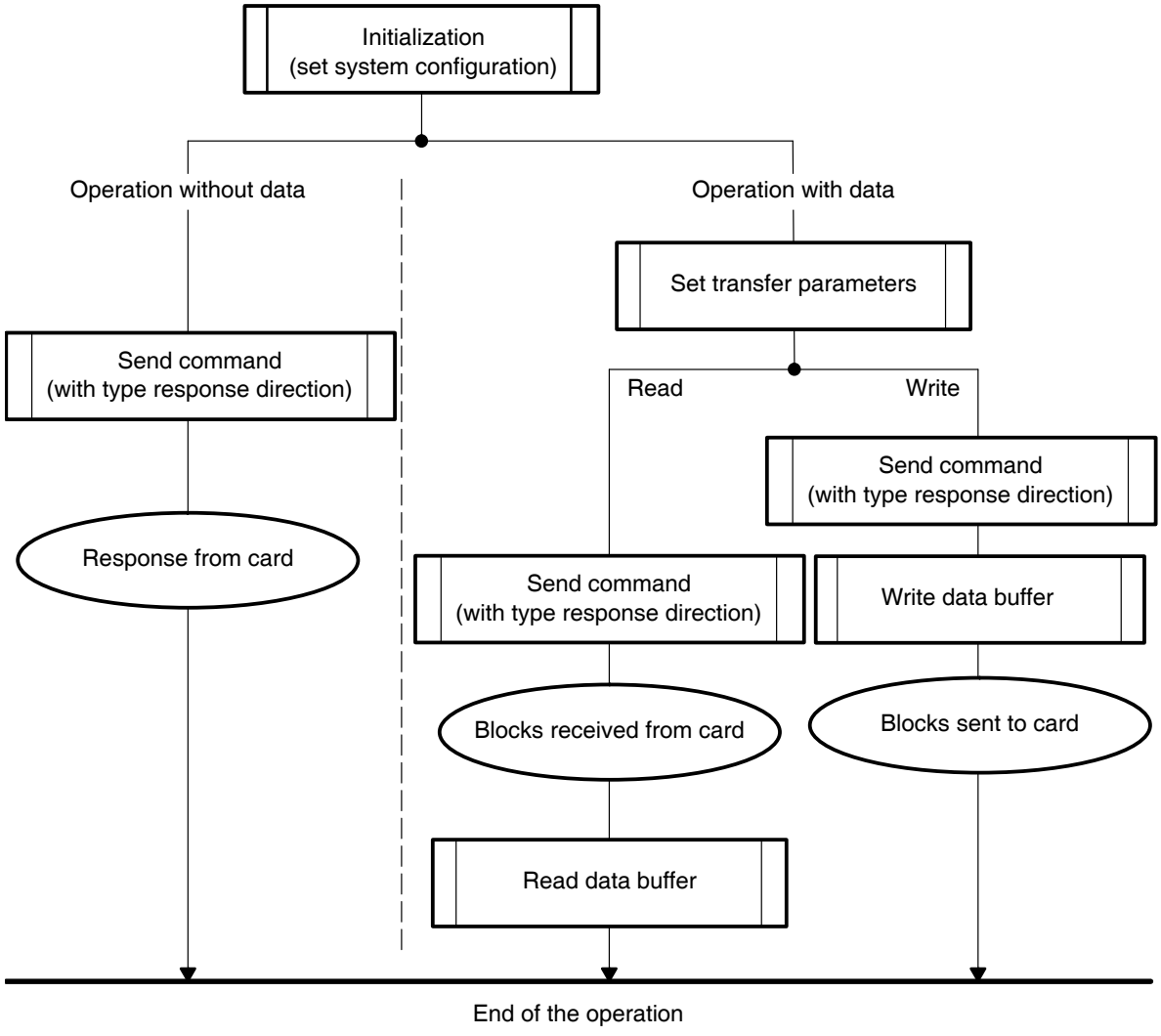
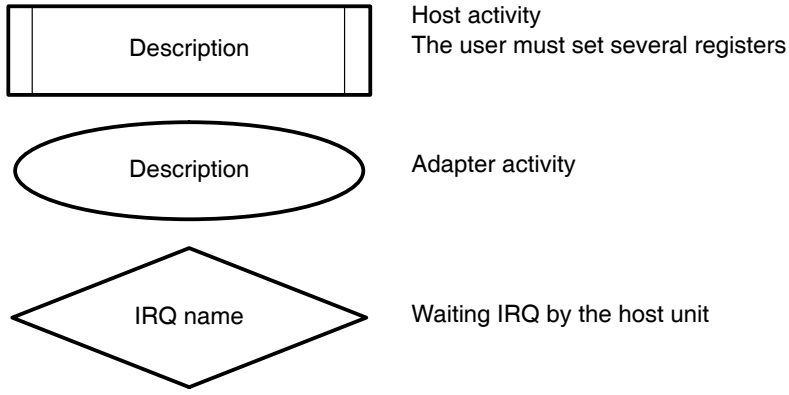


Figure 7. Initialization Phase



In all modes (MMC, SD, and SPI) an initialization phase is necessary at the beginning. After the MMCSD adapter, different actions are taken depending on whether the MPU sends a command with/without data, and depending on the type, the index of the response and the direction.

Figure 8. Detail of Basic Operation

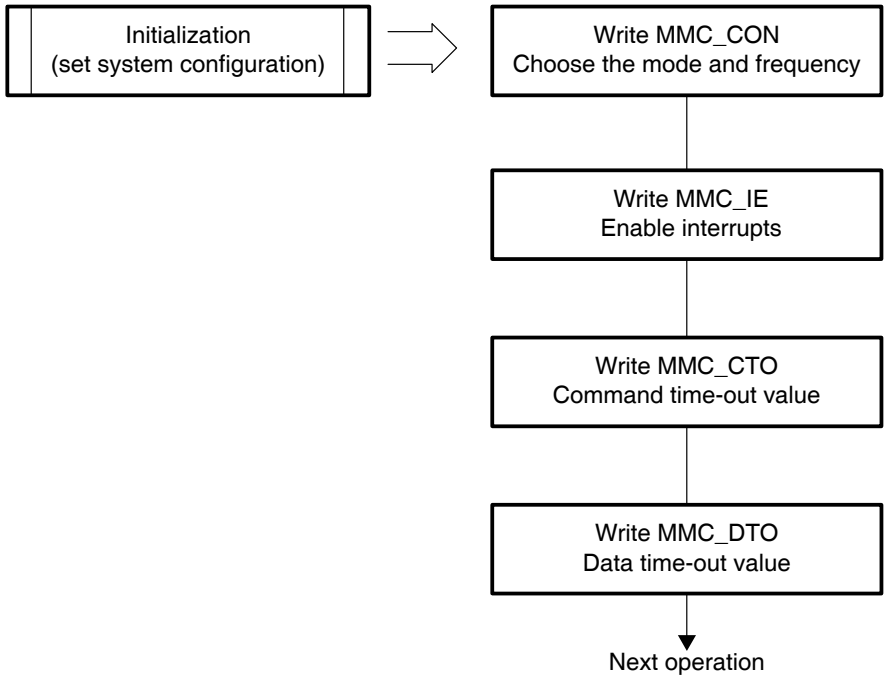
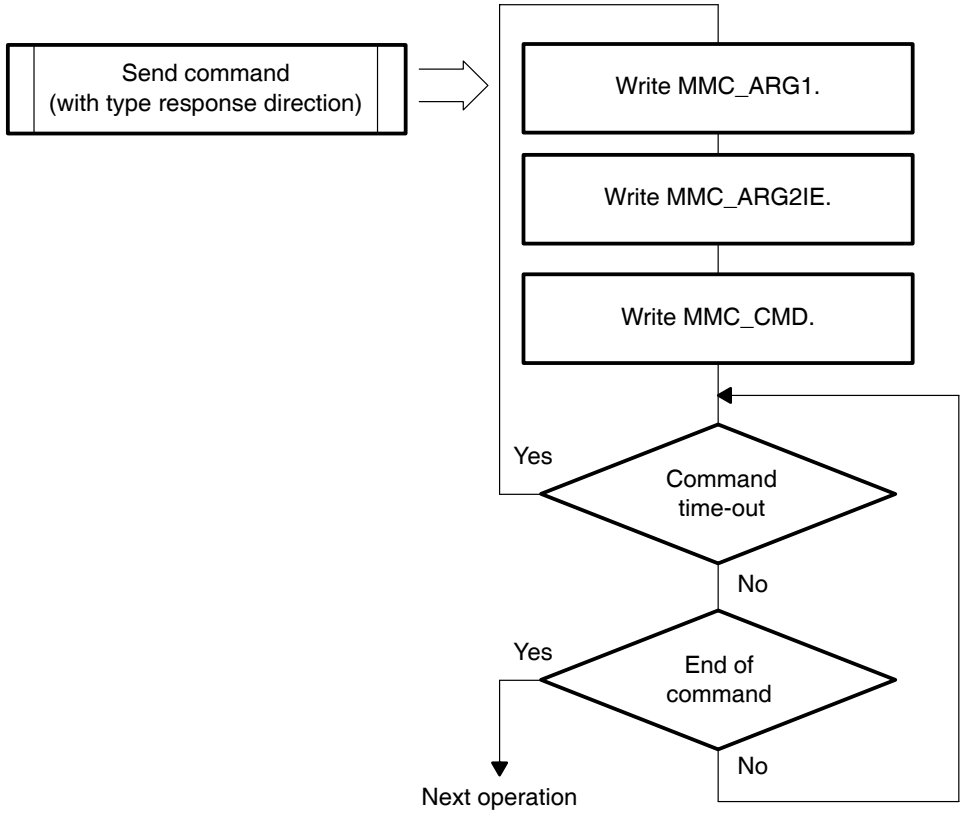
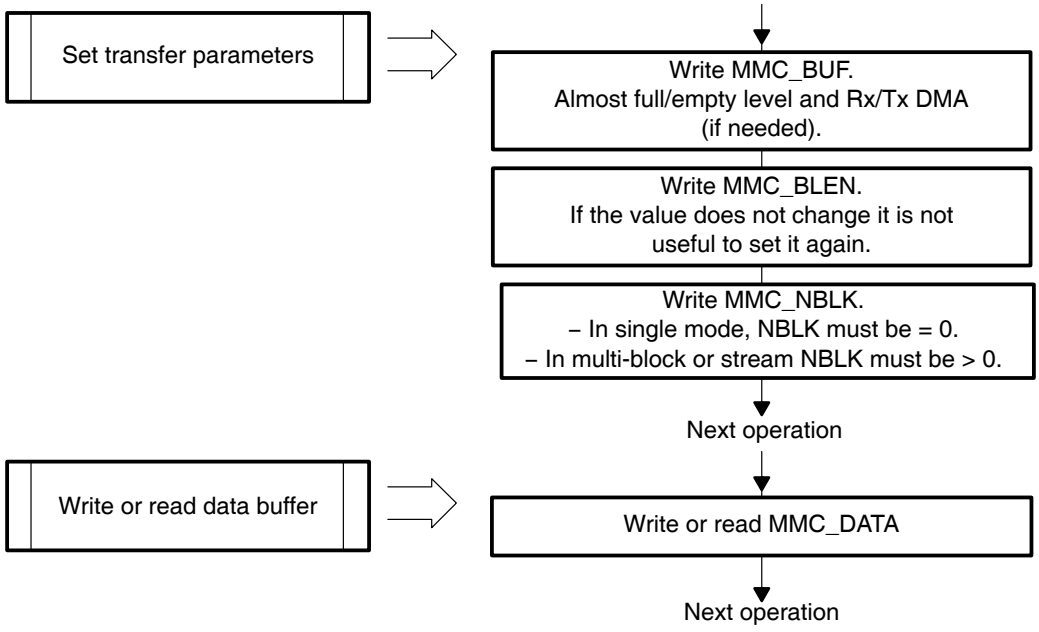


Figure 9. Command Transfer



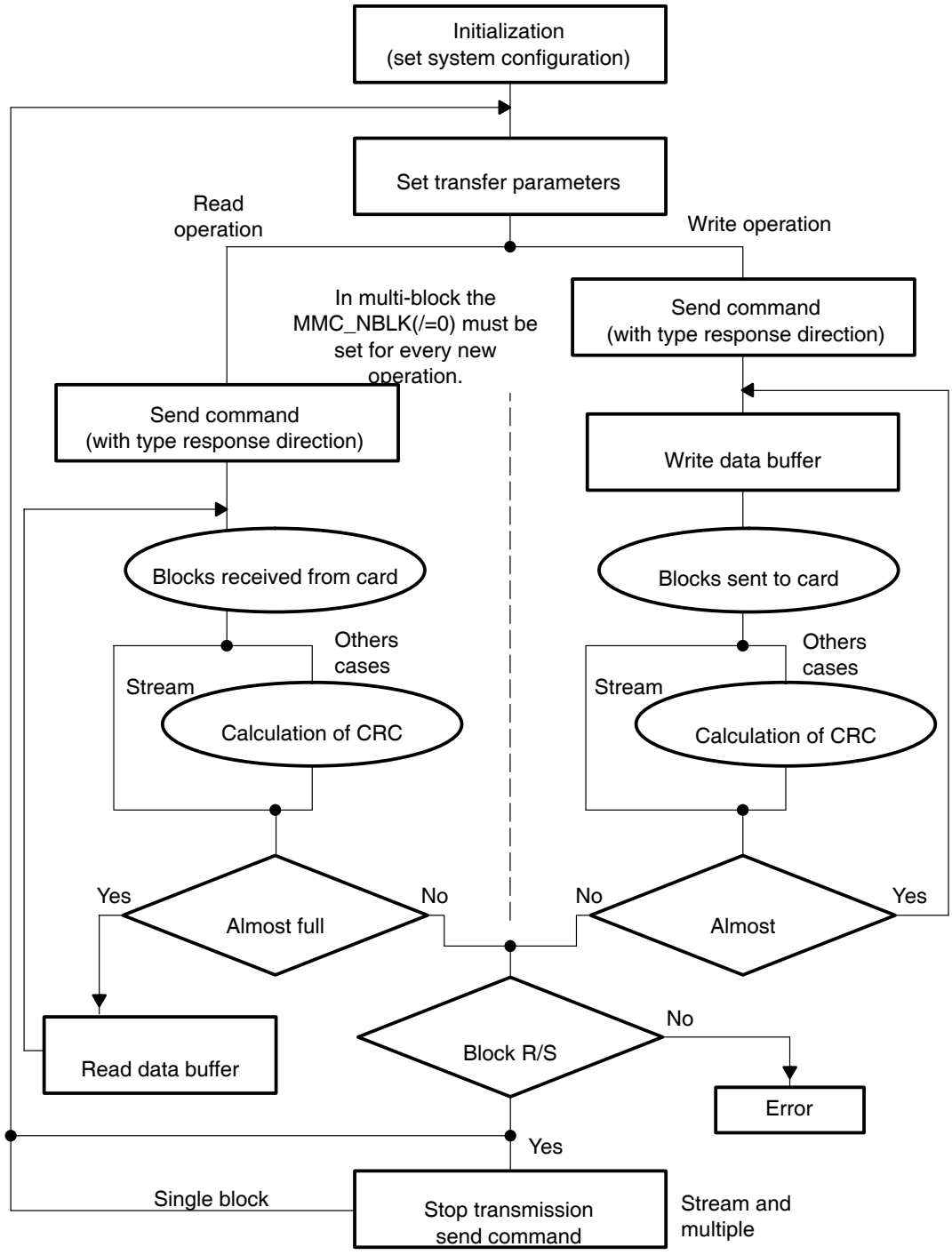
When a command is used that has no response, i.e., CMD0, CMD4, CMD15), the command-timeout condition can never occur since there is no response expected. In this case, the NO path from the command-timeout clock is taken, and the timeout interrupt is not valid.

Figure 10. Data Transfer



For the example shown in Figure 11, the mode selected is MMC/SD (MMC_CON[13:12] = 00).

Figure 11. Data Transfer in MMC/SD Mode Example



1.9 DMA Operation

1.9.1 MMC DMA-Receive Mode

In a DMA block-read operation (single or multiple):

- The DMA-RX request signal is asserted to its active level when the FIFO level becomes equal to or greater than the threshold set in the AF_level.
- The DMA-RX request is deasserted to its inactive level when the system DMA has read one word from the FIFO.

Because the request lasts one 16-bit word-read cycle, it is recommended that the threshold level (AF_level) be equal to the DMA burst size (n) minus 1. For instance, if the system DMA is programmed to support one word-read access, the AF_level must be set to 0.

The MMC/SD host controller does not generate a new DMA request until the system DMA has read the N words corresponding to the previous DMA request, even if the FIFO level is equal to or greater than the programmed threshold.

Since each DMA transfer has an equal size, it is necessary to have the total data size of the transfer be a multiple of the DMA read access size (max 32 words).

Summary:

DMA transfer size = n ≤ FIFO size
(max 32 16-bit words)

AF_level = n -1 (FIFO threshold level)

n = submultiple of total transfer size

Example: Multiple-block read of 10 blocks of 512 bytes each.

The DMA transfer size n can be set to 20 words (40 bytes) and AF_level= 0x13 (0x14-1). Then the read-transfer operation completes after the 128 system DMA read requests.

The receive FIFO does not overflow. If the FIFO gets full, the MMC_clk clock signal is momentarily stopped until the system DMA or the MPU performs a read access which starts emptying the FIFO.

When using the MMC DMA-receive mode, the MPU software must enforce that the MPU never accesses the MMC_DAT register at the same time the DMA is accessing the register. Failure to enforce this restriction may cause unexpected results.

1.9.2 MMC DMA-Transmit Mode

In a DMA block-write operation (single or multiple):

- The DMA-TX request signal is asserted to its active level when the FIFO level becomes less than the threshold set in the AE_level after the block-write command has been set (write action into MMC_CMD).
- The DMA-TX request is deasserted to its inactive level when the system DMA has written one single word into the FIFO.

Because the request lasts one 16-bit word-write cycle, it is recommended that the threshold level (AE_level) be equal to the DMA burst size (n) minus 1. For instance, if the system DMA is programmed to transfer one word-write access, the AE_level must be set to 0.

In the DMA mode, because a new DMA TX request can be generated after the first read from the FIFO by the core, it is possible for the FIFO to hold a maximum of two DMA transfers of n words minus one. Hence the maximum permitted DMA transfer size is half the FIFO size.

The MMC/SD host controller does not generate a new DMA request until the system DMA has written the n words corresponding to the previous DMA request, even if the FIFO level is equal to or greater than the programmed threshold.

Because each DMA transfer has an equal size, it is necessary to have the total data size of the transfer be a multiple of the DMA write access size (max 16 words).

Summary:

DMA transfer size = n ≤ FIFO size/2
(max 16 16-bit words)

AE_level = n -1 (FIFO threshold level)

n = submultiple of total transfer size

Example: Multiple-block write of 10 blocks of 512 bytes each. The DMA transfer size n can be set to 10 words (20 bytes) and the AE_level= 0x9. Then the write transfer operation completes after 256 system DMA write requests.

The transmit FIFO does not underflow. If the FIFO becomes empty, the MMC_clk clock signal is momentarily stopped till the system DMA or the MPU performs a write access which starts filling the FIFO.

1.10 MPU (IRQ/Polling) Mode

1.10.1 MMC MPU (IRQ/Polling) Receive Mode

During a MPU block-read operation (single or multiple) using the interrupt/polling mode, the A_Full status bit is set active (high level) when the FIFO level becomes equal to or greater than the threshold set in the AF_level. The MPU can only clear the A_Full set bit by writing a '1' into the A_Full status bit location.

The threshold level (AF_level) equals to the MPU transfer size (n) minus 1. If the threshold is set to 0 (AF_level=0x00), the MPU has to read one word by one word. If the threshold is set to 31 (AF_level=0x1F), the MPU has to read 32 words by 32 words.

Each new A_Full status bit assertion (high level) is internally masked until the MPU has not performed exactly n reads.

Unlike the DMA mode, it is not needed to have the total data size of the transfer be a multiple of the MPU read-access size (maximum 32 words). The last MPU read access can be down to the MPU transfer size when ending any total data size transfer, as follows:

- MPU transfer size = n =FIFO size (max 32 16-bit words)
- AF_level = n −1 (FIFO threshold level)

An example is a multiple-block read of 10 blocks of 512 bytes each.

The MPU transfer size n can be set to 20 words (40 bytes) and AF_level= 0x13 (0x14−1). Then the MPU read-transfer operation completes after 128 A_Full status bit assertions.

The MPU transfer size n can also be set to 15 words (30 bytes) and AF_level = 0x0E (0x0F−1). Then the MPU read-transfer operation completes after 170 system A_Full status bit assertion/deassertions, plus a last assertion for the last transfer of 10 words (20bytes).

The receive FIFO never overflows. If the FIFO gets full, the MMC_clk clock signal is momentarily stopped until the system DMA or the MPU performs a read access, which starts emptying the FIO.

Index

C

clock, MMC/SD, host controller 17
command flow, MMC/SD 53

D

DMA
  MMC
    receive mode 58
    transmit mode 59
  operation 58
  request, MMC/SD host controller 17
  transfer, threshold level 58

H

host, controller
  MMC/SD 13
  MMC/SD clocks 17
  MMC/SD DMA request 17
  MMC/SD features 15
  MMC/SD interrupt 18
  MMC/SD reset 17
  MMC/SD signal pads 15

I

interrupt, MMC/SD, host controller 18

M

MMC, DMA
  receive mode 58
  transmit mode 59
MMC/SD
  command flow 53
  host controller
    clocks 17
    description 13
    DMA request 17
    features 15
    interrupt 18
    reset 17
    signal pads 15
  internal pullups 18

O

operation, DMA 58

P

pullup, internal, MMC/SD 18

R

reset, MMC/SD, host controller 17

S

signal pads, MMC/SD, host controller 15

T

threshold level, DMA transfer 58