

OMAP35xx Applications Processor Multimedia Card/Secure Digital/ Secure Digital I/O (MMC/SD/SDIO) Card Interface

Texas Instruments OMAP™ Family of Products

Technical Reference Manual

Literature Number: SPRUFD2
February 2008



1	Multimedia Card/Secure Digital/ Secure Digital I/O (MMC/SD/SDIO) Card Interface	9
1.1	MMC/SD/SDIO Overview	10
1.1.1	MMC/SD/SDIO Features	12
1.2	MMC/SD/SDIO Environment	14
1.2.1	MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card	14
1.2.2	MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card Through an External Transceiver Device	14
1.2.3	MMC/SD/SDIO Functional Interfaces	15
1.2.3.1	Basic MMC/SD/SDIOi Pins Without External Transceiver	15
1.2.3.2	Basic MMC/SD/SDIO2 Pins with External Transceiver	16
1.2.3.3	MMC/SD/SDIO Protocol and Data Format	16
1.2.3.3.1	Protocol	17
1.2.3.3.2	Data Format	19
1.3	MMC/SD/SDIO Integration	22
1.3.1	Clocking, Reset, and PowerManagement Scheme	22
1.3.1.1	Clocks	22
1.3.1.1.1	Module Clocks	22
1.3.1.1.2	Power Management	23
1.3.1.2	Resets	25
1.3.1.2.1	Hardware Reset	25
1.3.1.2.2	Software Reset	25
1.3.1.3	Power Domain	26
1.3.2	Hardware Requests	26
1.3.2.1	DMA Requests	26
1.3.2.1.1	DMA Receive Mode	26
1.3.2.1.2	DMA Transmit Mode	27
1.3.2.2	Interrupt Requests	28
1.3.2.2.1	Interrupt-Driven Operation	29
1.3.2.2.2	Polling	29
1.4	MMC/SD/SDIO Functional Description	30
1.4.1	Description	30
1.4.2	Mode Selection	33
1.4.3	Buffer Management	33
1.4.3.1	Data Buffer	33
1.4.3.1.1	Data Buffer Status	35
1.4.4	Transfer Process	36
1.4.4.1	Different Types of Commands	36
1.4.4.2	Different Types of Responses	36
1.4.5	Transfer or Command Status and Errors Reporting	36
1.4.6	Transfer Stop	37
1.4.7	MMC CE-ATA Command Completion Disable Management	38
1.5	MMC/SD/SDIO Basic Programming Model	39
1.5.1	MMC/SD/SDIO Host Controller Initialization Flow	39
1.5.1.1	Enable Interface and Functional clock for MMC Controller	39
1.5.1.2	MMCHS Soft Reset Flow	40

1.5.1.3	Set MMCHS Default Capabilities	40
1.5.1.4	Wakeup Configuration.....	41
1.5.1.5	MMC Host and Bus Configuration	42
1.5.2	Basic Operations for MMC/SD/SDIO Host Controller	43
1.5.2.1	Card Detection, Identification, and Selection	43
1.5.2.2	Read/Write Transfer Flow in DMA Mode with Interrupt.....	45
1.5.2.3	Read/Write Transfer Flow in DMA Mode with Polling	46
1.5.2.4	Read/Write Transfer Flow without DMA with Polling	47
1.5.2.5	Read/Write Transfer Flow in CE-ATA Mode	48
1.5.2.6	Suspend-Resume Flow	48
1.5.2.6.1	Suspend Flow	49
1.5.2.6.2	Resume Flow	50
1.5.2.7	Basic Operations - Steps Detailed.....	50
1.5.2.7.1	Command Transfer Flow	51
1.5.2.7.2	MMCHS Clock Frequency Change	53
1.5.3	MMC/SD/SDIO1 Bus Voltage Selection.....	53
1.6	MMC/SD/SDIO Use Cases and Tips	54
1.6.1	MMCHS Controller Usage	54
1.6.1.1	Overview	54
1.6.1.2	Environment	55
1.6.1.2.1	Command and Data Transfer Formats	56
1.6.1.3	Programming Flow.....	58
1.6.1.3.1	Initial Configuration	58
1.6.1.3.2	MMC Card Identification.....	60
1.6.1.3.3	MMC Bus Setting Change After Card Identification	62
1.6.1.3.4	Reading the CSD Register of a MMC Card	63
1.6.1.3.5	MMC Write Transfer	65
1.6.1.3.6	MMC Read Transfer	67
1.7	MMC/SD/SDIO Registers	69
1.7.1	MMC/SD/SDIO Register Mapping Summary	69
1.7.2	Register Descriptions	71
1.7.2.1	MMCHS_SYSCONFIG	71
1.7.2.2	MMCHS_SYSSTATUS.....	72
1.7.2.3	MMCHS_CSRE	73
1.7.2.4	MMCHS_SYSTEST.....	73
1.7.2.5	MMCHS_CON.....	74
1.7.2.6	MMCHS_PWCNT.....	77
1.7.2.7	MMCHS_BLK.....	78
1.7.2.8	MMCHS_ARG	79
1.7.2.9	MMCHS_CMD.....	79
1.7.2.10	MMCHS_RSP10	82
1.7.2.11	MMCHS_RSP32	82
1.7.2.12	MMCHS_RSP54	83
1.7.2.13	MMCHS_RSP76	84
1.7.2.14	MMCHS_DATA.....	84
1.7.2.15	MMCHS_PSTATE	85
1.7.2.16	MMCHS_HCTL.....	88
1.7.2.17	MMCHS_SYSCTL	90

1.7.2.18	MMCHS_STAT	92
1.7.2.19	MMCHS_IE	96
1.7.2.20	MMCHS_ISE	98
1.7.2.21	MMCHS_AC12.....	100
1.7.2.22	MMCHS_CAPA.....	101
1.7.2.23	MMCHS_CUR_CAPA.....	103
1.7.2.24	MMCHS_REV.....	103

List of Figures

1-1	MMC/SD/SDIO1 and 3 Overview	11
1-2	MMC/SD/SDIO2 Overview	12
1-3	MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card Without External Transceiver	14
1-4	MMC/SD/SDIO2 Connected to an MMC, an SD, or an SDIO Card with External Transceiver	15
1-5	MMC/SD/SDIOi Interface Signals	15
1-6	MMC/SD/SDIO2 Interface Signals	16
1-7	Sequential Read Operation (MMC Cards Only)	17
1-8	Sequential Write Operation (MMC Cards Only)	17
1-9	Multiple Block Read Operation	18
1-10	Multiple Block Write Operation with Card Busy Signal	18
1-11	Command Token Format	19
1-12	Response Token Format (R1, R3, R4, R5, R6)	19
1-13	Response Token Format (R2)	19
1-14	Data Token Format for 1-Bit Transfers	20
1-15	Data Token Format for 4-Bit Transfers	20
1-16	Data Token Format for 8-Bit Transfers	21
1-17	MMC/SD/SDIO1 Integration	22
1-18	DMA Receive Mode	27
1-19	DMA Transmit Mode	28
1-20	MMC/SD/SDIO Diagram	31
1-21	Buffer Management for a Write	34
1-22	Buffer Management for a Read	35
1-23	MMC/SD/SDIO Controller Meta Initialization Steps	39
1-24	MMC/SD/SDIO Controller Software Reset Flow	40
1-25	MMC/SD/SDIO Controller Wakeup Configuration	41
1-26	MMC/SD/SDIO Controller Bus Configuration	42
1-27	MMC/SD/SDIO Controller Card Identification and Selection - part 1	43
1-28	MMC/SD/SDIO Controller Card Identification and Selection - part 2	44
1-29	MMC/SD/SDIO Controller Read/Write Transfer Flow in DMA mode with interrupt	45
1-30	MMC/SD/SDIO Controller Read/Write Transfer Flow in DMA mode with Polling	46
1-31	MMC/SD/SDIO Controller Read/Write Transfer Flow without DMA with Polling	47
1-32	MMC/SD/SDIO Controller Read/Write in CE-ATA mode	48
1-33	MMC/SD/SDIO Controller Suspend Flow	49
1-34	MMC/SD/SDIO Controller Resume Flow	50
1-35	MMC/SD/SDIO Controller Command Transfer Flow with polling	51
1-36	MMC/SD/SDIO Controller Command Transfer Flow with interrupts	52
1-37	MMC/SD/SDIO Controller Clock Frequency Change Flow	53
1-38	Overview	54
1-39	Environment	56
1-40	Command Transfer	57
1-41	Data Read Transfer	57
1-42	Data Write Transfer	57

List of Tables

1-1	MMC/SD/SDIOi I/O Description	15
1-2	MMC/SD/SDIO2 I/O Description	16
1-3	Relation Between Configuration and Name of Response Type.....	20
1-4	SmartIdle Mode and Wakeup Capabilities	24
1-5	MMC, SD, SDIO responses in the MMCHS_RSPxx registers	36
1-6	CC and TC Values Upon Error Detected	36
1-7	MMC/SD/SDIOi Controller Transfer Stop Command Summary.....	38
1-8	Register Print for the MMCHS1 controller's clocks Initialization	58
1-9	MMCHS Controller Voltage Capabilities Initialization	58
1-10	MMC Controller Default Initialization Values.....	59
1-11	MMCHS Controller INIT Procedure Start	59
1-12	MMCHS Controller Pre-Card Identification Configuration.....	60
1-13	Sending CMD0	60
1-14	Sending CMD5	60
1-15	Sending CMD8	61
1-16	Sending CMD55.....	61
1-17	Sending CMD1	61
1-18	Sending CMD2	62
1-19	Sending CMD3	62
1-20	MMC Bus Setting Change Table	63
1-21	Sending CMD9	63
1-22	MMCHS_SYSCTL Value	64
1-23	Sending CMD7	64
1-24	Setting Data Bus Width with CMD6.....	64
1-25	MMCHS_CON Value	65
1-26	Enabling High Speed with CMD6	65
1-27	MMCHS_SYSCTL Value	65
1-28	Setting Block Length	66
1-29	Setting Number of Blocks	66
1-30	CMD25 Issuing	66
1-31	CMD18 Issuing	67
1-32	Instance Summary	69
1-33	MMC/SD/SDIO1 Register Offset Addresses.....	69
1-34	MMC/SD/SDIO2 Register Offset Addresses.....	70
1-35	MMC/SD/SDIO3 Register Offset Addresses.....	70
1-36	MMCHS_SYSCONFIG	71
1-37	Register Call Summary for Register MMCHS_SYSCONFIG.....	72
1-38	MMCHS_SYSSTATUS	72
1-39	Register Call Summary for Register MMCHS_SYSSTATUS.....	73
1-40	MMCHS_CSRE.....	73
1-41	Register Call Summary for Register MMCHS_CSRE	73
1-42	MMCHS_SYSTEST	74
1-43	Register Call Summary for Register MMCHS_SYSTEST	74
1-44	MMCHS_CON	74
1-45	Register Call Summary for Register MMCHS_CON.....	77
1-46	MMCHS_PWCNT	77
1-47	Register Call Summary for Register MMCHS_PWCNT.....	77
1-48	MMCHS_BLK	78
1-49	Register Call Summary for Register MMCHS_BLK.....	78

1-50	MMCHS_ARG	79
1-51	Register Call Summary for Register MMCHS_ARG	79
1-52	MMCHS_CMD	80
1-53	Register Call Summary for Register MMCHS_CMD	81
1-54	MMCHS_RSP10.....	82
1-55	Register Call Summary for Register MMCHS_RSP10	82
1-56	MMCHS_RSP32.....	83
1-57	Register Call Summary for Register MMCHS_RSP32	83
1-58	MMCHS_RSP54.....	83
1-59	Register Call Summary for Register MMCHS_RSP54	83
1-60	MMCHS_RSP76.....	84
1-61	Register Call Summary for Register MMCHS_RSP76	84
1-62	MMCHS_DATA	85
1-63	Register Call Summary for Register MMCHS_DATA	85
1-64	MMCHS_PSTATE.....	86
1-65	Register Call Summary for Register MMCHS_PSTATE	87
1-66	MMCHS_HCTL	88
1-67	Register Call Summary for Register MMCHS_HCTL.....	90
1-68	MMCHS_SYSCTL.....	90
1-69	Register Call Summary for Register MMCHS_SYSCTL	92
1-70	MMCHS_STAT	92
1-71	Register Call Summary for Register MMCHS_STAT.....	96
1-72	MMCHS_IE.....	96
1-73	Register Call Summary for Register MMCHS_IE	98
1-74	MMCHS_ISE.....	98
1-75	Register Call Summary for Register MMCHS_ISE	99
1-76	MMCHS_AC12.....	100
1-77	Register Call Summary for Register MMCHS_AC12	101
1-78	MMCHS_CAPA	101
1-79	Register Call Summary for Register MMCHS_CAPA.....	102
1-80	MMCHS_CUR_CAPA.....	103
1-81	Register Call Summary for Register MMCHS_CUR_CAPA	103
1-82	MMCHS_REV	104
1-83	Register Call Summary for Register MMCHS_REV.....	104

Multimedia Card/Secure Digital/ Secure Digital I/O (MMC/SD/SDIO) Card Interface

This chapter describes the features and functions of the multimedia card/secure digital/secure digital I/O (MMC/SD/SDIO) card interface for the OMAP35xx Applications Processor.

Topic	Page
1.1 MMC/SD/SDIO Overview.....	10
1.2 MMC/SD/SDIO Environment.....	14
1.3 MMC/SD/SDIO Integration	22
1.4 MMC/SD/SDIO Functional Description	30
1.5 MMC/SD/SDIO Basic Programming Model	39
1.6 MMC/SD/SDIO Use Cases and Tips	54
1.7 MMC/SD/SDIO Registers	69

1.1 MMC/SD/SDIO Overview

The multimedia card high-speed/secure data/secure digital I/O (MMC/SD/SDIO) host controller provides an interface between a local host (LH) such as a microprocessor unit (MPU) or digital signal processor (DSP) and either MMC, SD memory cards, or SDIO cards and handles MMC/SD/SDIO transactions with minimal LH intervention.

The application interface manages transaction semantics. The MMC/SD/SDIO host controller deals with MMC/SD/SDIO protocol at transmission level, data packing, adding cyclic redundancy checks (CRC), start/end bit, and checking for syntactical correctness.

The application interface can send every MMC/SD/SDIO 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 of end of operation.

The application interface can read card responses or flag registers. It can also mask individual interrupt sources. All these operations can be performed by reading and writing control registers. The MMC/SD/SDIO host controller also supports two DMA channels.

Note: Some features may not be available or supported in your particular device. For more information, see Chapter 1, the *OMAP35xx Family* section, and your device-specific data manual.

There are three MMC/SD/SDIO host controllers inside the device: [Figure 1-1](#) gives an overview of the MMC/SD/SDIO controller instances 1 and 3, and [Figure 1-2](#) gives an overview of the MMC/SD/SDIO2 controller instance.

Figure 1-1. MMC/SD/SDIO1 and 3 Overview

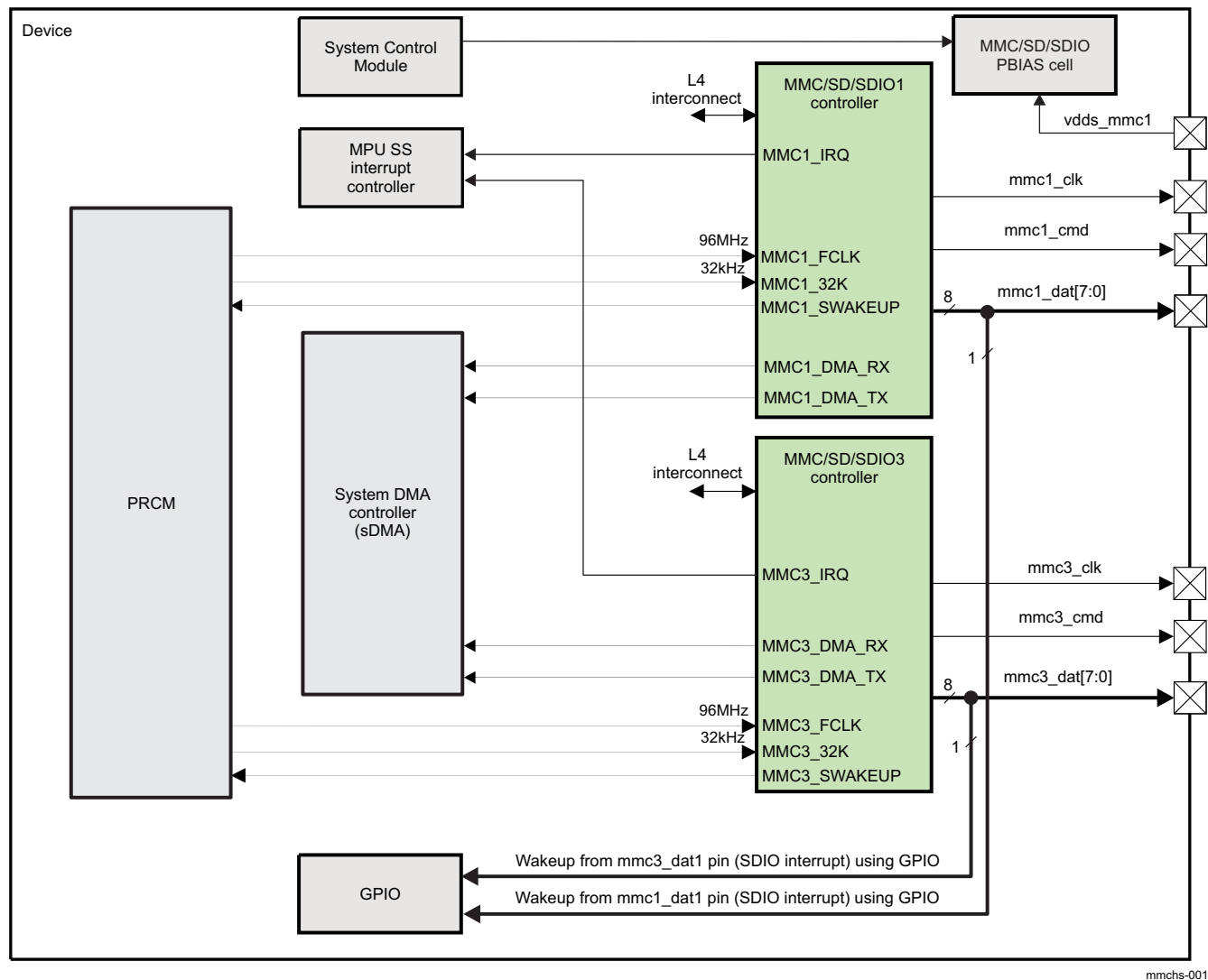
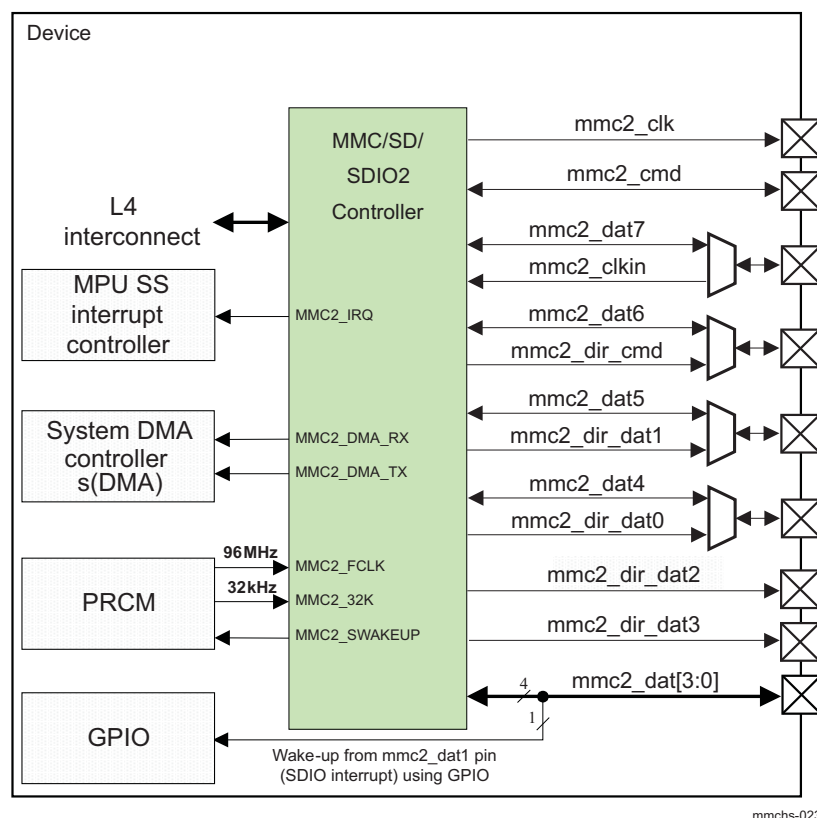


Figure 1-2. MMC/SD/SDIO2 Overview


1.1.1 MMC/SD/SDIO Features

The main features of the MMC/SD/SDIO host controller are:

- Full compliance with MMC command/response sets as defined in the *Multimedia Card System Specification*, v4.2. Including high-capacity (size >2GB) cards HC MMC.
- Full compliance with SD command/response sets as defined in the *SD Memory Card Specifications*, v2.0. Including high-capacity (size >2GB) cards HC SD.
- Full compliance with SDIO command/response sets and interrupt/read-wait mode as defined in the *SDIO Card Specification, Part E1*, v1.10
- Compliance with sets as defined in the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v1.00
- Full compliance with MMC bus testing procedure as defined in the *Multimedia Card System Specification*, v4.2
- Full compliance with CE-ATA command/response sets as defined in the *CE-ATA Standard Specification*
- Full compliance with ATA for MMCA specification
- Flexible architecture allowing support for new command structure
- Support:
 - 1-bit or 4-bit transfer mode specifications for SD and SDIO cards
 - 1-bit, 4-bit, or 8-bit transfer mode specifications for MMC cards
- Built-in 1024-byte buffer for read or write
- 32-bit-wide access bus to maximize bus throughput
- Single interrupt line for multiple interrupt source events
- Two slave DMA channels (1 for TX, 1 for RX)
- Designed for low power

- Programmable clock generation
- Support SDIO Read Wait and Suspend/Resume functions
- Support Stop at block gap
- Support command completion signal (CCS) and command completion signal disable (CCSD) management as specified in the *CE-ATA Standard Specification*

The known limitations are as follows:

- No built-in hardware support for error correction codes (ECC). See the *Multimedia Card System Specification*, v4.2, and the *SD Memory Card Specifications*, v2.0, for details about ECC.
- The maximum block size defined in the *SD Memory Card Specifications*, v2.0, that the host driver can read and write to the buffer in the host controller is 2048 bytes. MMC supports a maximum block size of 1024 bytes. Up to 512 byte transfers, the buffer in MMC is considered as a double buffering with ping-pong management; half of the buffer can be written while the other part is read. For 512 to 1024 byte transfers, the entire buffer is dedicated to the transfer (read only or write only).

Note: For more information about timing limitations, refer to your device-specific data manual.

The differences between the MMC/SD/SDIO host controllers and a Standard SD host controller are defined by the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v1.00, as follows:

- The MMC/SD/SDIO host controllers support MMC cards.
- The MMC/SD/SDIO host controller is defined as a DMA slave device. Standard SD host controller is defined as DMA master controller that can start and stop a DMA transfer. MMC/SD/SDIO host controllers support DMA transfers through slave DMA requests.
- The clock divider in MMC/SD/SDIO host controller supports a wider range of frequency than specified in the *SD Memory Card Specifications*, v2.0. The MMC/SD/SDIO host controller supports odd and even clock ratio.
- The MMC/SD/SDIO host controller supports configurable busy timeout.

1.2 MMC/SD/SDIO Environment

One MMC/SD/SDIO host controller can support one MMC memory card, one SD memory card, or one SDIO card.

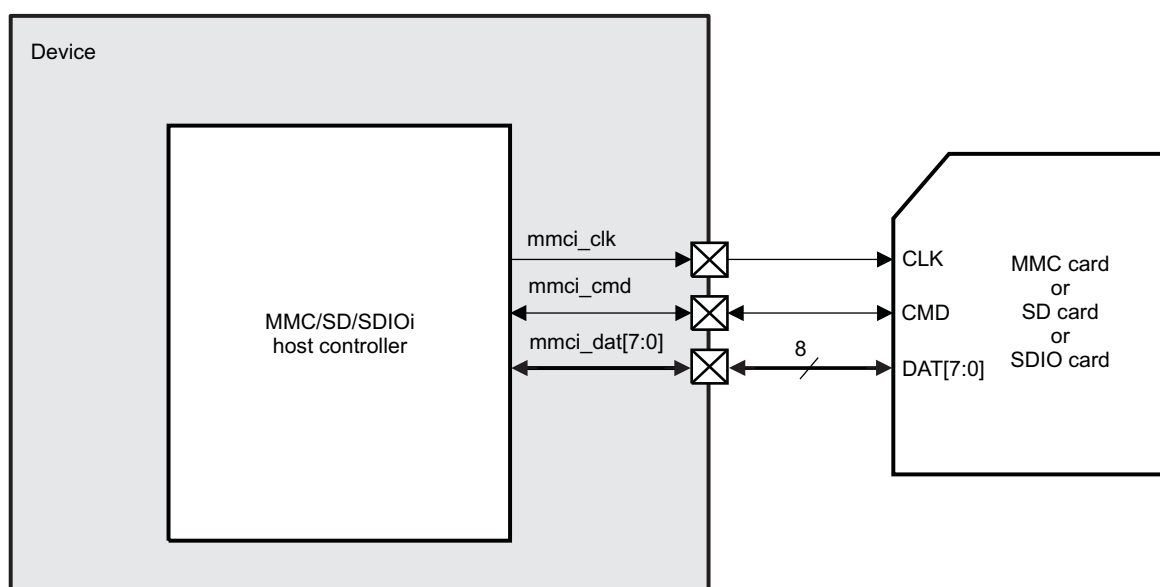
Other combinations (for example, two SD cards, one MMC card, and one SD card) are not supported through a single controller.

- The first controller (MMC/SD/SDIO1) integrates an internal transceiver that allows a direct connection to the MMC/SD/SDIO card (1.8 and 3V), without external transceiver.
- The second controller (MMC/SD/SDIO2) allows connecting MMC/SD/SDIO cards (only 1.8V cards) or an external device that uses the MMC/SD/SDIO interface (WLAN device for example). The second instance also supports an external transceiver and provides direction signals for data and command.
- The third controller (MMC/SD/SDIO3) allows connecting MMC/SD/SDIO cards (only 1.8V cards) or an external device that uses the MMC/SD/SDIO interface (Wireless USB card for example). This interface is used without external transceiver.

1.2.1 MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card

Figure 1-3 shows MMC/SD/SDIOi host controller, instance 1, 2 or 3, connected to an MMC, an SD, or an SDIO card and its related external connections.

Figure 1-3. MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card Without External Transceiver

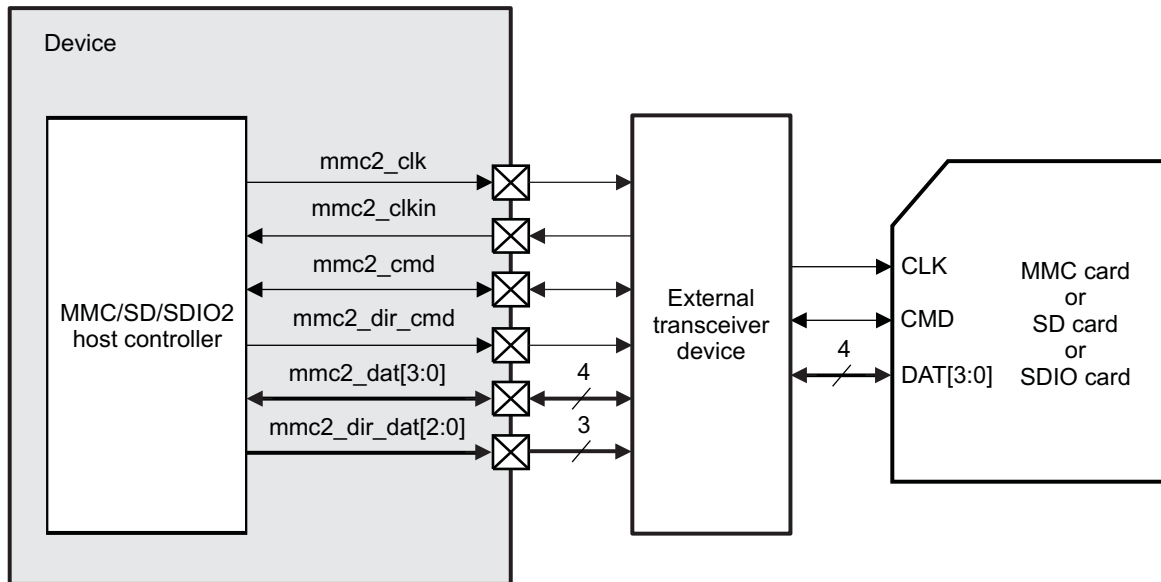


mmchs-002

1.2.2 MMC/SD/SDIO Connected to an MMC, an SD, or an SDIO Card Through an External Transceiver Device

This connection is supported only by the MMC/SD/SDIO2 host controller. Figure 1-4 shows the MMC/SD/SDIO2 host controller connected to an MMC, an SD, or an SDIO card through an external transceiver device.

Figure 1-4. MMC/SD/SDIO2 Connected to an MMC, an SD, or an SDIO Card with External Transceiver



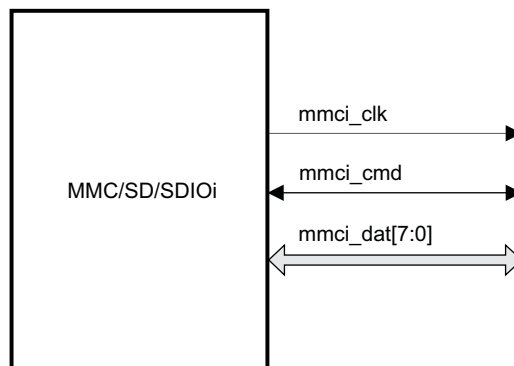
mmchs-003

1.2.3 MMC/SD/SDIO Functional Interfaces

1.2.3.1 Basic MMC/SD/SDIOi Pins Without External Transceiver

Figure 1-5 shows the MMC/SD/SDIOi host controller interface signals (instance 1, 2 or 3).

Figure 1-5. MMC/SD/SDIOi Interface Signals



mmchs-004

Table 1-1 describes the MMC/SD/SDIOi inputs/outputs.

Table 1-1. MMC/SD/SDIOi I/O Description

Signal Name	I/O	Description	Reset Value
<code>mmci_clk</code>	O	External clock for MMC/SD/SDIO card	0
<code>mmci_cmd</code>	I/O	Command signal	0
<code>mmci_dat[7:0]</code>	I/O	Data signals	0

1.2.3.2 Basic MMC/SD/SDIO2 Pins with External Transceiver

Figure 1-6 shows the MMC/SD/SDIO2 host controller interface signals.

Figure 1-6. MMC/SD/SDIO2 Interface Signals

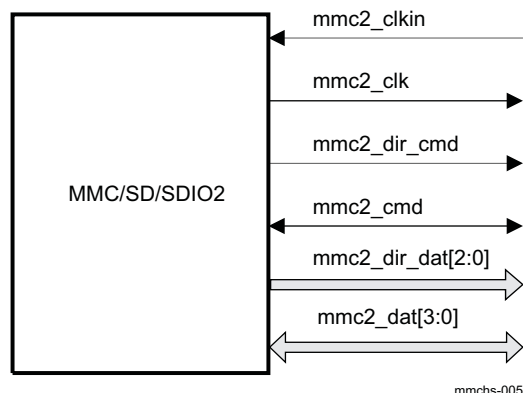


Table 1-2 describes the MMC/SD/SDIO2 inputs/outputs.

Table 1-2. MMC/SD/SDIO2 I/O Description

Signal Name	I/O	Description	Reset Value
mmc2_clk	O	External clock for MMC/SD/SDIO card	0
mmc2_clkin	I	Input clock from MMC/SD/SDIO card	0
mmc2_cmd	I/O	Command signal	0
mmc2_dir_cmd	O	Direction control for mmc2_cmd signal case an external transceiver is used (high when transmit, low when receive)	0
mmc2_dat[3:0]	I/O	Data signals	0
mmc2_dir_dat0	O	Direction control for mmc2_dat0 signal when an external transceiver is used (high when transmit, low when receive)	0
mmc2_dir_dat1	O	Direction control for mmc2_dat1 and mmc2_dat3 signal when an external transceiver is used (high when transmit, low when receive)	0
mmc2_dir_dat2	O	Direction control for mmc2_dat2 signal when an external transceiver is used (high when transmit, low when receive)	0
mmc2_dir_dat3	O	Direction control for mmc2_dat[7:4] signal when an external transceiver is used (high when transmit, low when receive). Unusable on the device because mmci_dat[7:4] are muxed with other direction control signals. See the <i>System Control Module</i> chapter for further details on the pin multiplexing.	0

1.2.3.3 MMC/SD/SDIO Protocol and Data Format

The bus protocol between the MMC/SD/SDIOi host controller and the card is message-based. Each message is represented by one of the following parts:

Command: a command starts an operation. The command is transferred serially from the MMC/SD/SDIO host controller to the card on the mmci_cmd line.

Response: a response is an answer to a command. The response is sent from the card to the MMC/SD/SDIO host controller. It is transferred serially on the mmci_cmd line.

Data: data are transferred from the MMC/SD/SDIOi host controller to the card or from a card to the MMC/SD/SDIO host controller using the DATA lines.

Busy: the mmci_dat0 signal is maintained low by the card as far as it is programming the data received.

CRC status: CRC result is sent by the card through the mmci_dat0 line when executing a write transfer. In the case of transmission error, occurring on any of the active data lines, the card sends a negative CRC status on mmci_dat0. In the case of successful transmission, over all active data lines, the card sends a positive CRC status on mmci_dat0 and starts the data programming procedure.

1.2.3.3.1 Protocol

There are two types of data transfer:

- Sequential operation
- Block-oriented operation

There are specific commands for each type of operation (sequential or block-oriented).

See the *Multimedia Card System Specification*, v4.2, the *SD Memory Card Specifications*, v2.0, and the *SDIO Card Specification, Part E1*, August 2004, for details about commands and programming sequences supported by the MMC, SD, and SDIO cards.

Figure 1-7 and Figure 1-8 show how sequential operations are defined. Sequential operation is only for 1-bit transfer and initiates a continuous data stream. The transfer terminates when a stop command follows on the mmci_cmd line.

CAUTION

Stream commands are supported only by MMC cards.

Figure 1-7. Sequential Read Operation (MMC Cards Only)

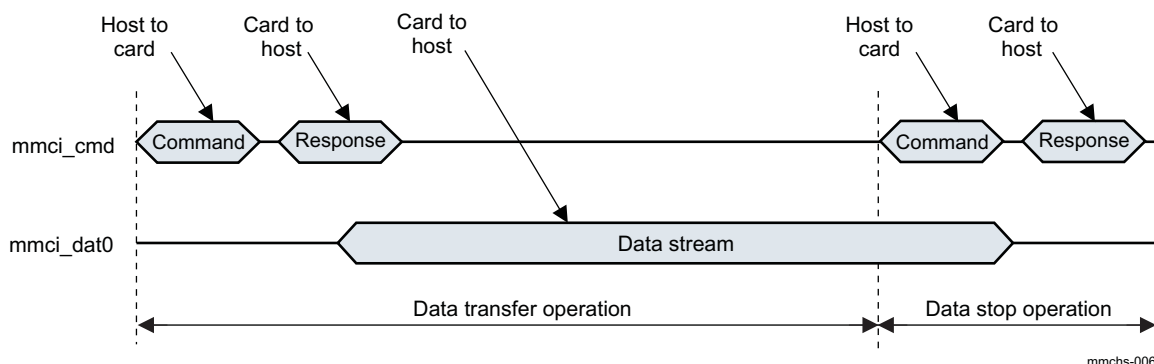


Figure 1-8. Sequential Write Operation (MMC Cards Only)

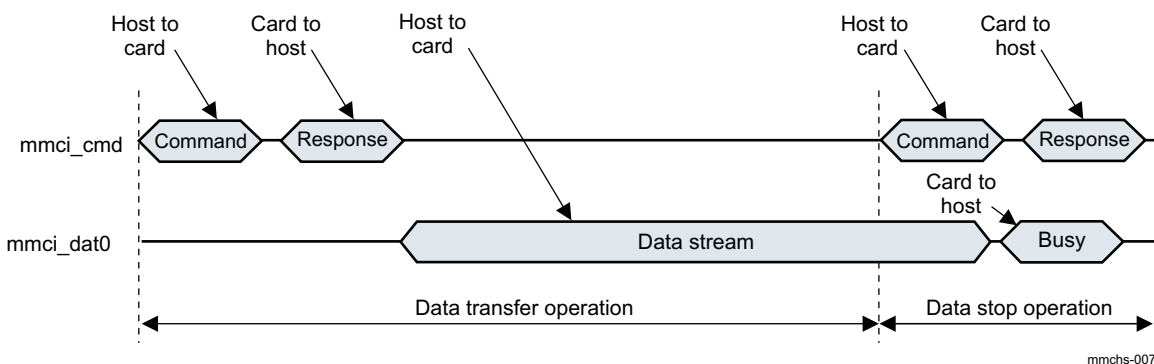


Figure 1-9 and Figure 1-10 show how multiple block-oriented operations are defined. A multiple block-oriented operation sends a data block plus CRC bits. The transfer terminates when a stop command follows on the mmci_cmd line. These operations are available for all kinds of cards.

Figure 1-9. Multiple Block Read Operation

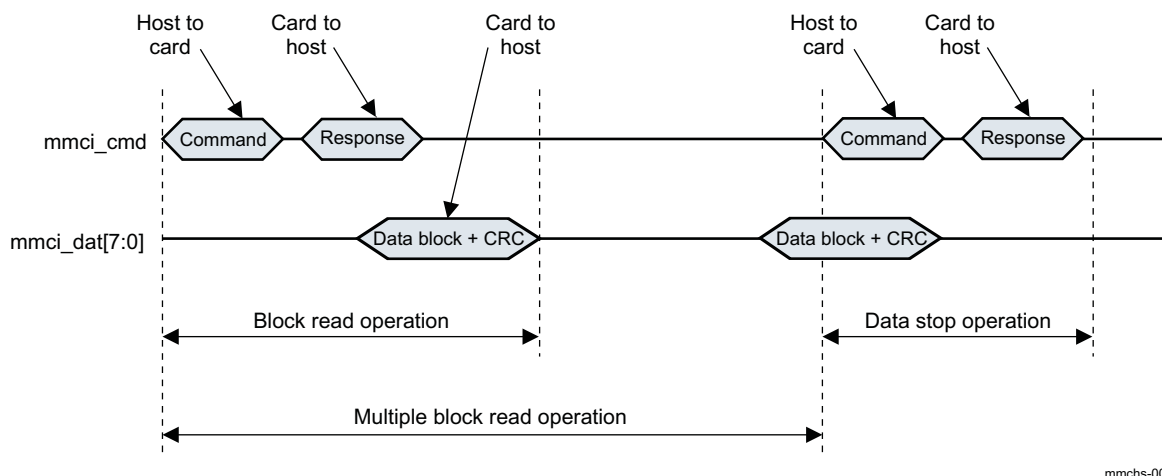
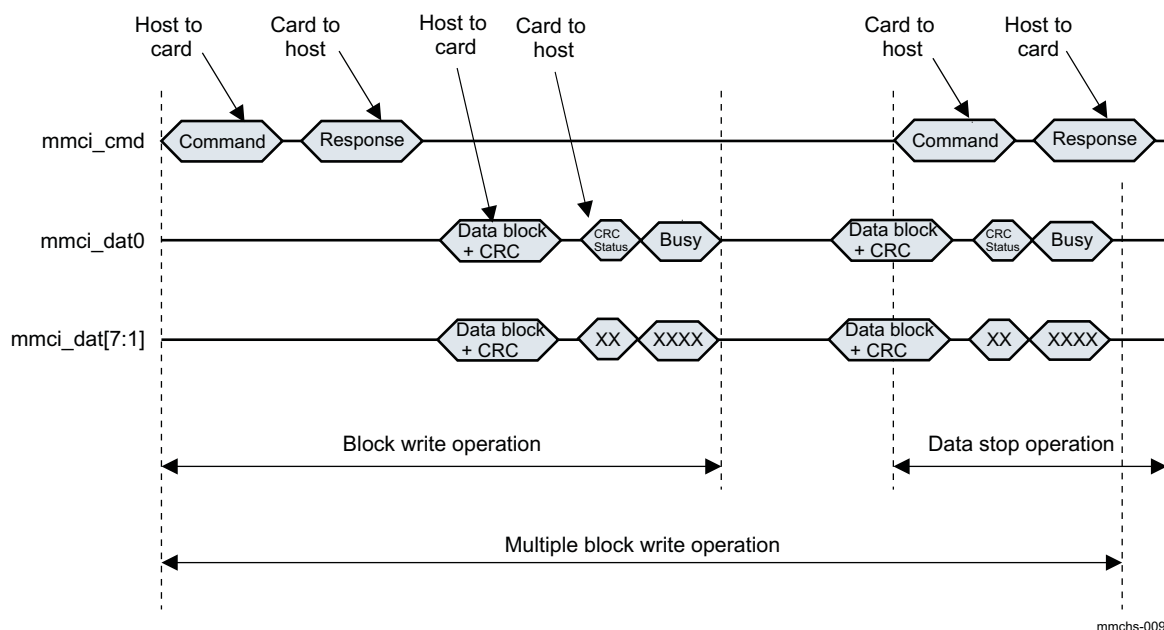


Figure 1-10. Multiple Block Write Operation with Card Busy Signal



Notes:

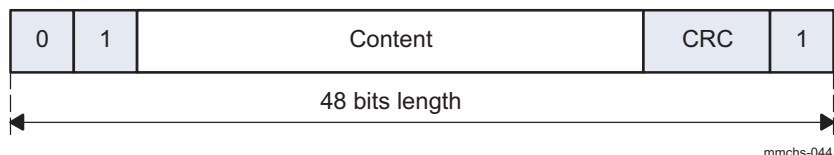
1. The card busy signal is not always generated by the card; the previous examples show a particular case.
2. It is software responsibility to do a software reset (set MMCI_MMCHS_SYSCTL[26] SRD bit to 0x1) after data timeout to ensure mmci_clk is stopped
3. For multiblock transfer, and especially for MMC cards, you can abort a transfer without using a stop command. Use a CMD23 before data transfer to define the number of blocks that will be transferred, then the transfer stops automatically after the last block (if the MMC card supports this feature).

1.2.3.3.2 Data Format

Coding Scheme for Command Token

Command tokens always start with 0 and end with 1. The second bit is a transmitter bit: 1 for a host command. The content is the command index (coded by 6 bits) and an argument (for example, an address), coded by 32 bits. The content is protected by 7-bit CRC checksum (see [Figure 1-11](#)).

Figure 1-11. Command Token Format



Coding Scheme for Response Token

Response tokens always start with 0 and end with a 1. The second bit is a transmitter bit: 0 for a card response. The content is different for each type of response (R1, R2, R3, R4, and R5 R6 [for SDIO]) and the content is protected by 7-bit CRC checksum (see [Figure 1-12](#) and [Figure 1-13](#)). Depending on the type of commands sent to the card, the `MMCHS_CMD` register must be configured differently to avoid false CRC or index errors to be flagged on command response (see [Table 1-3](#)). For more details about response types, see the *Multimedia Card System Specification*, v4.2, the *SD Memory Card Specifications*, v2.0, or the *SDIO Card Specification, Part E1*, v1.10.

Figure 1-12. Response Token Format (R1, R3, R4, R5, R6)

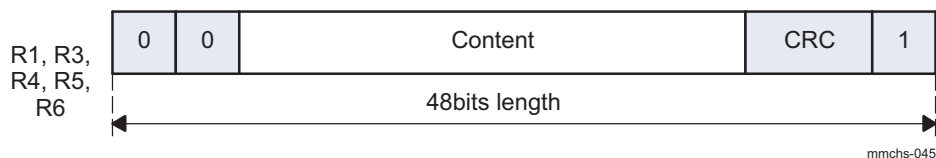


Figure 1-13. Response Token Format (R2)

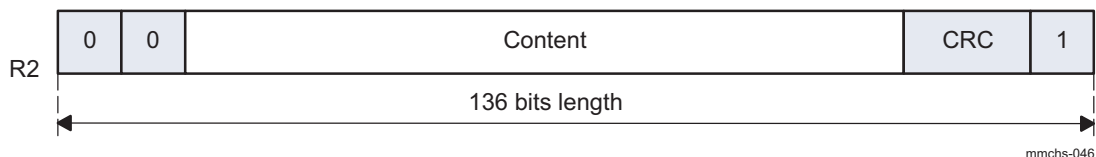


Table 1-3. Relation Between Configuration and Name of Response Type⁽¹⁾

Response Type MMCI.MMCHS_CMD[17:16] RSP_TYPE	Index Check Enable MMCI.MMCHS_CMD[20] CICE	CRC Check Enable MMCI.MMCHS_CMD[19] CCCE	Name of Response Type
00	0	0	No Response
01	0	1	R2
10	0	0	R3 (R4 for SD cards)
10	1	1	R1, R6, R5
11	1	1	R1b, R5b

⁽¹⁾ The MMC/SD/SDIO host controller assumes that both clocks may be switched off, whatever the value set in the MMCI.MMCHS_SYSCONFIG[9:8] CLOCKACTIVITY bit.

Coding Scheme for Data Token

Data tokens always start with 0 and end with 1 (see Figure 1-14, Figure 1-15, and Figure 1-16).

Figure 1-14. Data Token Format for 1-Bit Transfers

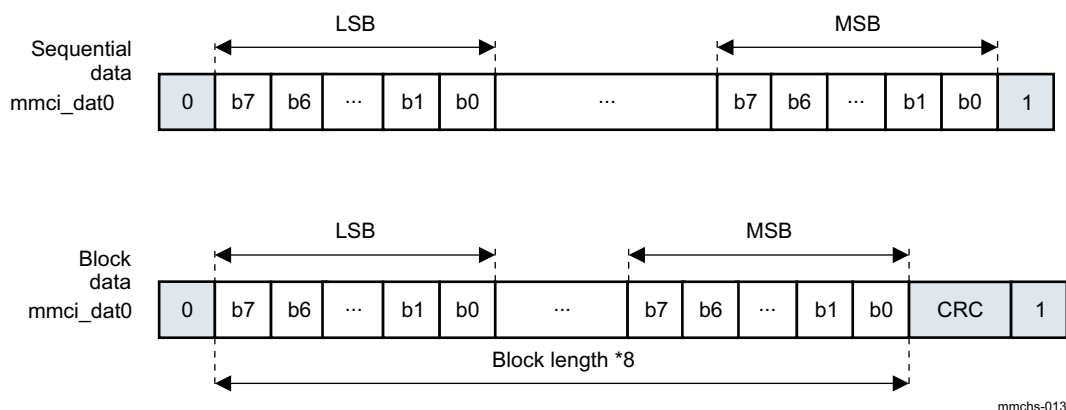


Figure 1-15. Data Token Format for 4-Bit Transfers

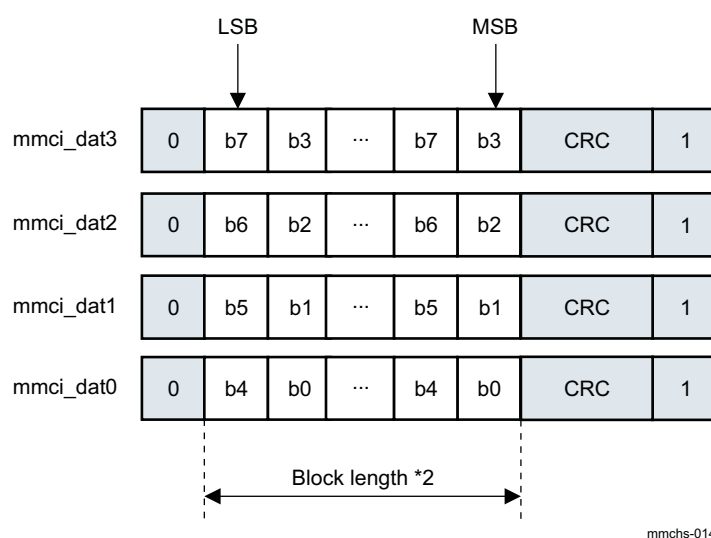
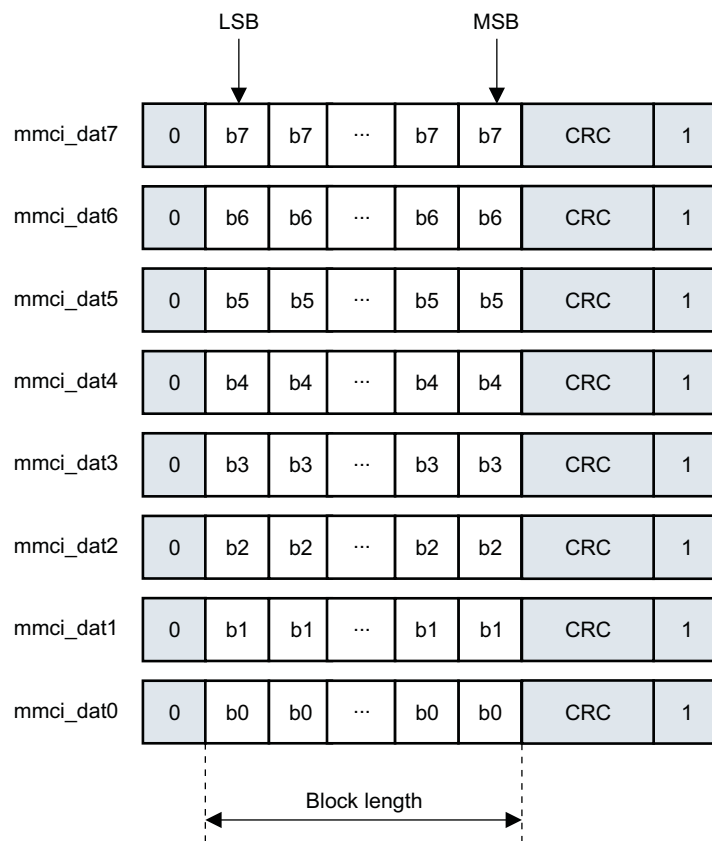


Figure 1-16. Data Token Format for 8-Bit Transfers

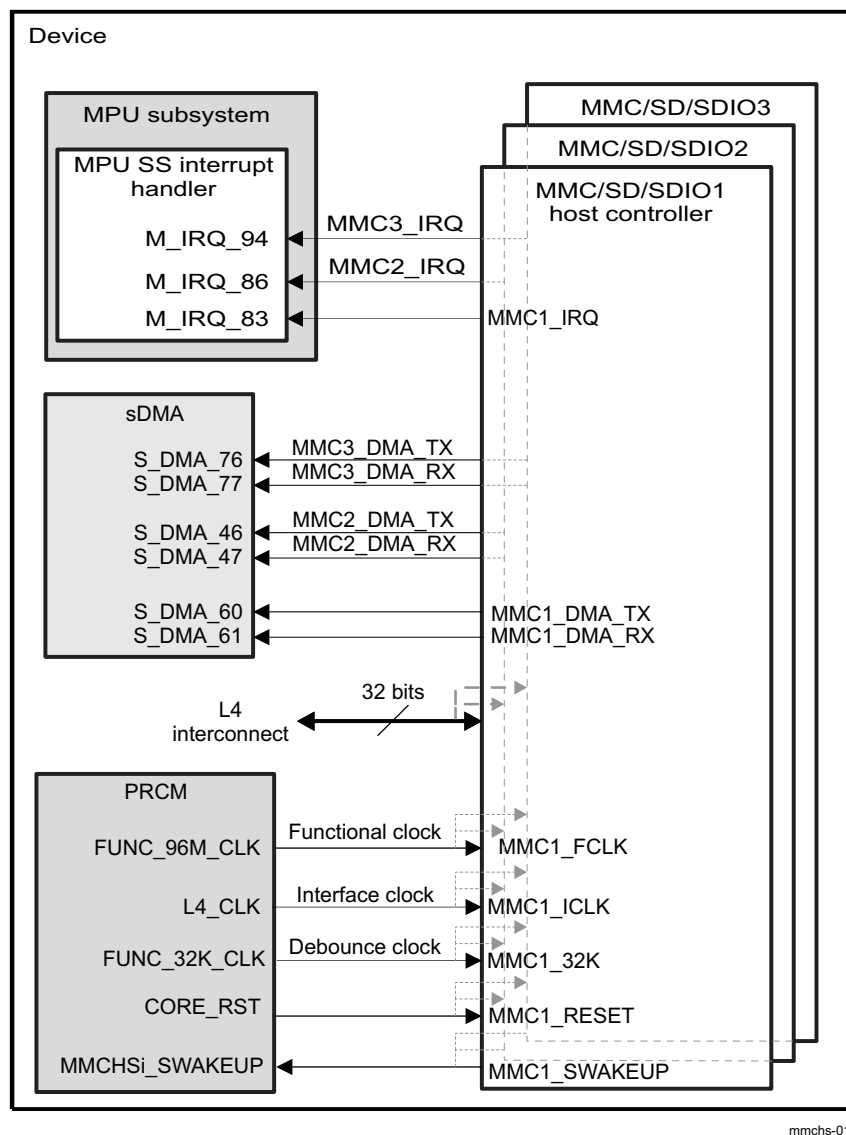


mmchs-015

1.3 MMC/SD/SDIO Integration

Figure 1-17 shows the internal connections between the three instances of the MMC/SD/SDIO host controller and the other modules

Figure 1-17. MMC/SD/SDIO1 Integration



1.3.1 Clocking, Reset, and PowerManagement Scheme

1.3.1.1 Clocks

1.3.1.1.1 Module Clocks

The MMC/SD/SDIO receives three clocks:

- A fixed functional clock of 96 MHz (the MMCi_FCLK) independent of the device external clock frequency
- An interface clock (the MMCi_ICLK) for interfacing with L4 interconnects (register accesses)
- A debounce clock, the MMCi_32K for reset process only

All these clocks are generated and controlled by the power, reset, and clock manager (PRCM) module (see the *Power Reset and Clock Management* chapter for more information).

1.3.1.1.2 Power Management

The MMC/SD/SDIO host controller can enter into different modes and save power:

- Normal mode
- Idle mode

The two modes are mutually exclusive (the module can be in normal mode or in idle mode). The MMC/SD/SDIO host controller is compliant with the PRCM module handshake protocol.

Normal Mode

The autogating of interface and functional clocks occurs when the following conditions are met:

- The MMCI.MMCHS_SYSCONFIG[0] AUTOIDLE bit is set to 1 (I = 1 for MMC/SD/SDIO1, 2 for MMC/SD/SDIO2 and 3 for MMC/SD/SDIO3 instances).
- There is no transaction on the MMC interface.

The autogating of interface and functional clocks stops when the following conditions are met:

- A register access occurs through the L4 interconnect.
- A wake-up event occurs (an interrupt from a SDIO card).
- A transaction on the MMC/SD/SIO interface starts.

Then the MMC/SD/SDIO host controller enters in low-power state (MMCI_ICLK clock autogated) even if MMCI.MMCHS_SYSCONFIG[0] AUTOIDLE is set to 0.

The functional clock is internally switched off and only interconnect read and write accesses are allowed.

Idle Mode

The MMCI_ICLK and MMCI_FCLK clocks provided to MMC/SD/SDIO are switched off upon a PRCM module request. They are switched back upon module request.

The MMC/SD/SDIO host controller complies with the PRCM module handshaking protocol:

- Idle request from the system power manager
- Idle acknowledgment from the MMC/SD/SDIO host controller
- Wake-up request from the MMC/SD/SDIO host controller

The idle acknowledgment varies according to the MMCI.MMCHS_SYSCONFIG[4:3] SIDLEMODE bit field:

- 0x0: Force-idle mode. The MMC/SD/SDIO host controller acknowledges the system power manager request unconditionally.
- 0x1: No-idle mode. The MMC/SD/SDIO host controller ignores the system power manager request and behaves normally as if the request was not asserted.
- 0x2: Smart-idle mode. The MMC/SD/SDIO host controller acknowledges the system power manager request according to its internal state.

During the smart-idle mode period, the MMC/SD/SDIO host controller acknowledges that the MMCI_ICLK and MMCI_FCLK clocks may be switched off whatever the value set in the MMCI.MMCHS_SYSCONFIG[9:8] CLOCKACTIVITY field.

Transition from Normal Mode to Smart-Idle Mode

Smart-idle mode is enabled when the MMCI.MMCHS_SYSCONFIG[4:3] SIDLEMODE bit field is set to 0x2.

The MMC/SD/SDIOi host controller goes into idle mode when the PRCM issues an idle request, according to its internal activity.

During normal to idle mode transition, any access to the registers of the MMC/SD/SDIOi host controller generates an error as long as the MMCI_ICLK clock is alive. The PRCM.CM_IDLEST1_CORE[25] ST_MMC2 and (respectively the PRCM.CM_IDLEST1_CORE[24] ST_MMC1 bit) is set to 0x0 when the MMC/SD/SDIO2 module (respectively the MMC/SD/SDIO1 module) can be accessed.

The MMC/SD/SDIO host controller acknowledges the idle request from the PRCM after ensuring the following:

- The current multi/single-block transfer is completed.
- Any interrupt or DMA request is asserted.
- There is no card interrupt on mmci_dat[1] signal.

As long as the MMC/SD/SDIOi controller do not acknowledge the idle request, if an event occurs, the MMC/SD/SDIOi host controller can still generate an interrupt or a DMA request. In this case, the module ignores the idle request from the PRCM.

As soon as the MMC/SD/SDIOi controller acknowledges the idle request from the PRCM, the module does not assert any new interrupt or DMA request.

Wakeup Event in Smart-Idle Mode

The wake-up feature is enabled when the following enable wake-up bits are set:

- MMCI.MMCHS_SYSCONFIG[2] ENAWAKEUP bit is set to 0x1
- MMCI.MMCHS_HCTL[24] IWE bit is set to 0x1
- MMCI.MMCHS_IE[8] CIRQ_ENABLE bit is set to 0x1

The wakeup is generated only in smart-idle mode only, when module is in idle mode.

Table 1-4 lists the supported cases in smart-idle mode.

Table 1-4. SmartIdle Mode and Wakeup Capabilities

Mode	MMCI_ICLK clock	MMCI_FCLK clock	Wake-up Event
Card interrupt	May be switched off ⁽¹⁾	May be switched off ⁽¹⁾	The module sends an asynchronous wake-up request on detection of a card interrupt on mmci_dat[1] signal

⁽¹⁾ The MMC/SD/SDIOi host controller assumes that both clocks may be switched off, whatever the value set in the MMCI.MMCHS_SYSCONFIG[9:8] CLOCKACTIVITY bit.

Transition from Smart-Idle Mode to Normal Mode

The MMC/SD/SDIO host controller detects the end of the idle period when the PRCM deasserts the idle request.

For the wake-up event, there is a corresponding interrupt status in the MMCI.MMCHS_STAT register. The MMC/SD/SDIOi host controller operates the conversion between wake-up and interrupt (or DMA request) upon exit from smart-idle mode if the associated enable bit is set in the MMCI.MMCHS_ISE register.

Interrupts and wake-up events have independent enable/disable controls, accessible through the MMCI.MMCHS_HCTL and MMCI.MMCHS_ISE registers. The overall consistency must be ensured by software.

The interrupt status register MMCI.MMCHS_STAT is updated with the event that caused the wake-up in the CIRQ bit when the MMCI.MMCHS_IE[8] CIRQ_ENABLE associated bit is enabled.

Then, the wake-up event at the origin of the transition from smart-idle mode to normal mode is converted into its corresponding interrupt or DMA request. (The MMCI.MMCHS_STAT register is updated and the status of the interrupt signal changes.)

When the idle request from the PRCM is deasserted, the module switches back to normal mode. The module is fully operational.

Force-Idle Mode

Force-idle mode is enabled when the MMCI.MMCHS_SYSCONFIG[4:3] SIDLEMODE bit field is set to 0x0.

Force-idle mode is an idle mode where the MMC/SD/SDIOi host controller responds unconditionally to the idle request from the PRCM. Moreover, in this mode, the MMC/SD/SDIOi host controller unconditionally deasserts interrupts and DMA request lines asserted.

The transition from normal mode to force-idle mode does not affect the bits of the MMCi.MMCHS_STAT register.

In force-idle mode, the interrupt and DMA request lines are deasserted. MMCi_ICLK and MMCi_FCLK can be switched off.

CAUTION

In Force-ilde mode, an idle request from the PRCM during a command or a data transfer can lead to an unexpected and unpredictable result. When the module is idle, any access to the module generates an error as long as the MMCi_ICLK clock is alive.

The module exits the force-idle mode when the PRCM deasserts the idle request. Then the module switches back to normal mode. The module is fully operational. Interrupt and DMA request lines are optionally asserted one clock cycle later.

1.3.1.2 Resets

1.3.1.2.1 Hardware Reset

The module is reinitialized by the hardware when the active-low reset signal (CORE_RST), synchronous to the MMCi_ICLK clock is asserted on the input pin MMCi_RESET (see the *Power, Reset, and Clock Management* chapter for more information).

A global status bit RESETDONE is provided in the status register MMCi.MMCHS_SYSSTATUS[0]. This bit is set to 1 when all the different clock domain resets (interface domain, functional domain, and 32K domain) have been released.

The MMCi.MMCHS_SYSSTATUS[0] RESETDONE bit can be monitored by the software to check if the module is ready-to-use after a hardware reset.

Note: Functional clock MMCi_FCLK, interface clock MMCi_ICLK, and debounce clock MMCi_32K must be provided to the module to allow the RESETDONE status bit to be set.

This hardware reset signal has a global reset action on the module. All configuration registers and all statemachines are reset in all clock domains.

1.3.1.2.2 Software Reset

The module is reinitialized by software through the MMCi.MMCHS_SYSCONFIG[1] SOFTRESET bit. This bit has the same action on the module logic as the hardware MMCi_RESET signal except for:

- Debounce logic
- MMCi.MMCHS_PSTATE, MMCi.MMCHS_CAPA, and MMCi.MMCHS_CUR_CAPA registers (see corresponding register descriptions)

The SOFTRESET bit is active high. The bit is automatically reinitialized to 0 by the hardware. The MMCi.MMCHS_SYSCTL[24] SRA bit has the same action as the SOFTRESET bit on the design.

The MMCi.MMCHS_SYSSTATUS[0] RESETDONE bit can be monitored by the software to check if the module is ready-to-use after a software reset.

Moreover, two partial software reset bits are provided:

- MMCi.MMCHS_SYSCTL[26] SRD bit
- MMCi.MMCHS_SYSCTL[25] SRC bit

These two reset bits are useful to reinitialize data or command processes respectively in case of line conflict. When set to 1, a reset process is automatically released when the reset completes:

- The MMCI.MMCHS_SYSCTL[26] SRD bit resets all finite state-machines and status management that handle data transfers on both the interface and functional side.
- The MMCI.MMCHS_SYSCTL[25] SRC bit resets all finite state-machines and status management that handle command transfers on both the interface and functional side.

1.3.1.3 Power Domain

MMC/SD/SDIOi power is supplied by the CORE power domain (see the *Power Reset and Clock Management* chapter for more information).

When the MMC/SD/SDIOi power domain is off, the only way to wake up the power domain and different MMC/SD/SDIOi clocks is to monitor mmci_dat[1] input pin state via a different GPIO line for each MMC/SD/SDIO interface (see the *General-Purpose Interface* chapter for more information).

1.3.2 Hardware Requests

1.3.2.1 DMA Requests

The MMC/SD/SDIOi host controller can be interfaced with a DMA controller. At system level, the advantage is to discharge the LH of the data transfers. The module does not support wide DMA access (above 1024 bytes) for SD cards as specified in the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v1.00.

The DMA request is issued if the three following conditions are met:

- The MMCI.MMCHS_CMD[0] DE bit is set to 1 to trigger the initial DMA request (the write must be done when running the data transfer command).
- A command was emitted on the mmci_cmd line.
- There is enough space in the buffer of the MMC/SD/SDIOi host controller to write an entire block (BLEN writes).

DMA request lines are connected on the system DMA (sDMA) inputs:

- S_DMA_60 (MMC1_DMA_TX)
- S_DMA_61 (MMC1_DMA_RX)
- S_DMA_46 (MMC2_DMA_TX)
- S_DMA_47 (MMC2_DMA_RX)
- S_DMA_76 (MMC3_DMA_TX)
- S_DMA_77 (MMC3_DMA_RX)

1.3.2.1.1 DMA Receive Mode

In a DMA block read operation (single or multiple), the request signal MMCI_DMA_RX is asserted to its active level when a complete block is written in the buffer. The block size transfer is specified in the MMCI.MMCHS_BLK[10:0] BLEN field.

The MMCI_DMA_RX signal is deasserted to its inactive level when the sDMA has read one single word from the buffer.

Only one request is sent per block; the DMA controller can make a 1-shot read access or several DMA bursts, in which case the DMA controller must manage the number of burst accesses, according to block size BLEN field.

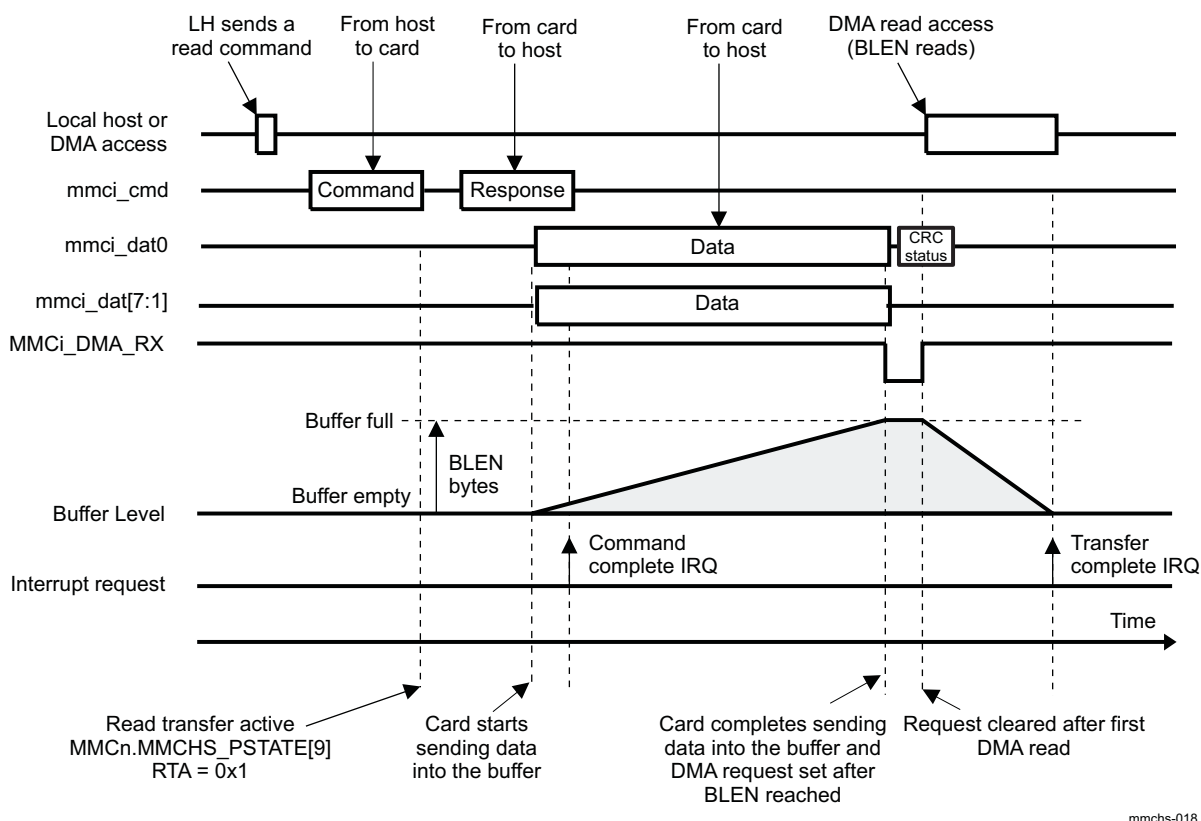
New DMA requests are internally masked if the sDMA has not read exactly BLEN bytes and a new complete block is not ready. As DMA accesses are in 32-bit, then the number of sDMA read is $\text{Integer}(\text{BLEN}/4)+1$.

The receive buffer never overflows. In multiple block transfers for block size above 512 bytes, when the buffer gets full, the mmci_clk clock signal (provided to the card) is momentarily stopped until the sDMA or the MPU performs a read access, which reads a complete block in the buffer.

Summary (see [Figure 1-18](#)):

- DMA transfer size = BLEN buffer size (maximum 1024 32-bit words) in one shot or by burst
- One DMA request per block

Figure 1-18. DMA Receive Mode



1.3.2.1.2 DMA Transmit Mode

In a DMA block write operation (single or multiple), the request signal `MMCi_DMA_TX` is asserted to its active level when a complete block is to be written to the buffer. The block size transfer is specified in the `MMCi.MMCHS_BLK[10:0]` BLEN field.

The `MMCi_DMA_TX` signal is deasserted to its inactive level when the sDMA has written one single word to the buffer.

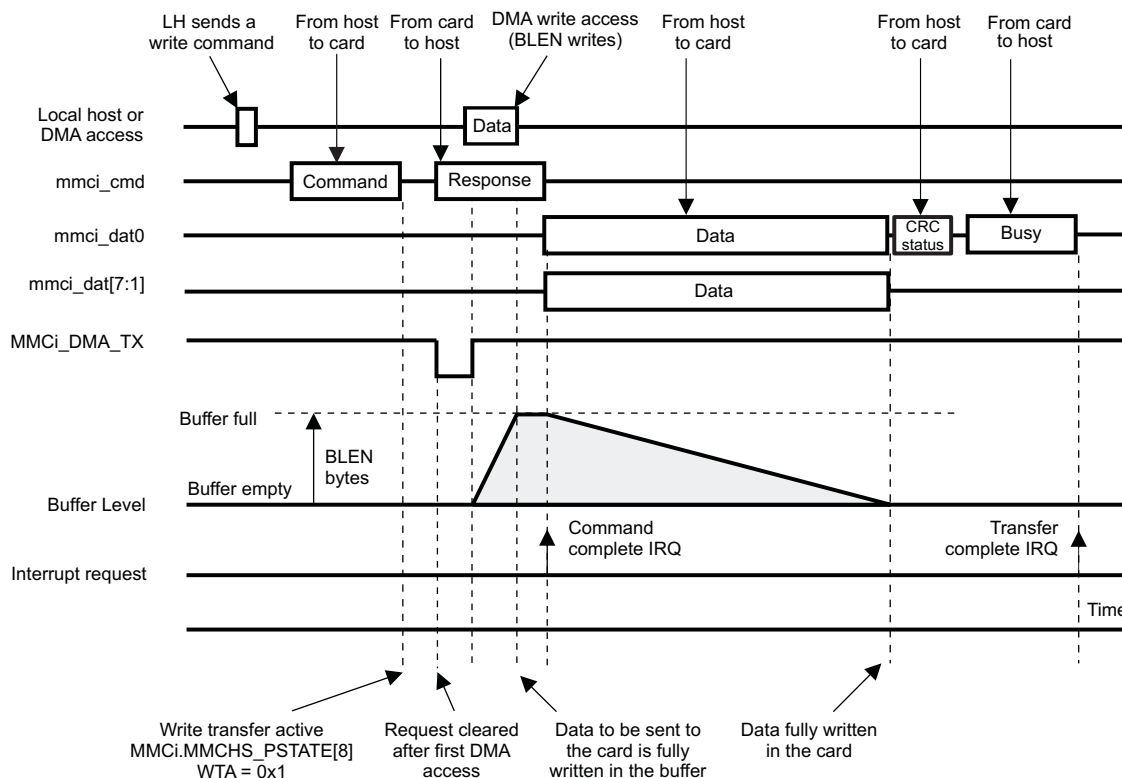
Only one request is sent per block; the DMA controller can make a 1-shot write access or multiple write DMA bursts, in which case the DMA controller must manage the number of burst accesses, according to block size BLEN field.

New DMA requests are internally masked if the sDMA has not written exactly BLEN bytes (as DMA accesses are in 32-bit, then the number of sDMA read is $\text{Integer}(\text{BLEN}/4)+1$) and if there is not enough memory space to write a complete block in the buffer.

Summary (see Figure 1-19):

- DMA transfer size = BLEN buffer size (maximum 1024 32-bit words) in one shot or by burst
- One DMA request per block

Figure 1-19. DMA Transmit Mode



mmchs-019

1.3.2.2 Interrupt Requests

Several internal module events can generate an interrupt. Each interrupt has a status bit, an interrupt enable bit, and a signal status enable:

- The status of each type of interrupt is automatically updated in the MMCI_MMCHS_STAT register; it indicates which service is required.
- The interrupt status enable bits of the MMCI_MMCHS_IE register enable/disable the automatic update of the MMCI_MMCHS_STAT register on an event-by-event basis.
- The interrupt signal enable bits of the MMCI_MMCHS_ISE register enable/disable the transmission of an interrupt request on the interrupt line MMCI_IRQ (from the MMC/SD/SDIO host controller to the MPU subsystem interrupt controller) on an event-by-event basis.

If an interrupt status is disabled in the MMCI_MMCHS_IE register, then the corresponding interrupt request is not transmitted, and the value of the corresponding interrupt signal enable in the MMCI_MMCHS_ISE register is ignored.

When an interrupt event occurs, the corresponding status bit is automatically set to 0x1 (the MMC/SD/SDIO host controller updates the status bit) in the MMCI_MMCHS_STAT register. If later a mask is applied on the interrupt in the MMCI_MMCHS_ISE register, the interrupt request is deactivated.

When the interrupt source has not been serviced, if the interrupt status is cleared in the MMCI_MMCHS_STAT register and the corresponding mask is removed from the MMCI_MMCHS_ISE register, the interrupt status is not asserted again in the MMCI_MMCHS_STAT register and the MMC/SD/SDIO host controller does not transmit an interrupt request.

CAUTION

If the buffer write ready interrupt (BWR) or the buffer read ready only interrupt (BRR) are not serviced and are cleared in the MMCi.MMCHS_STAT register, and the corresponding mask is removed, then the MMC/SD/SDIOi host controller will wait for the service of the interrupt without updating the status MMCi.MMCHS_STAT or transmitting an interrupt request.

The MMC/SD/SDIOi host controller supports interrupt-driven operation and polling.

There are one interrupt line for each instance of the MMC/SD/SDIOi host controller:

- MMC1_IRQ is connected to M_IRQ_83 for MMC/SD/SDIO1 instance.
- MMC2_IRQ is connected to M_IRQ_86 for MMC/SD/SDIO2 instance.
- MMC3_IRQ is connected to M_IRQ_94 for MMC/SD/SDIO3 instance.

1.3.2.2.1 Interrupt-Driven Operation

An interrupt enable bit must be set in the MMCi.MMCHS_IE register to enable the module internal source of interrupt.

When an interrupt event occurs, the single interrupt line is asserted and the LH must:

- Read the MMCi.MMCHS_STAT register to identify which event occurred.
- Write 1 into the corresponding bit of the MMCi.MMCHS_STAT register to clear the interrupt status and release the interrupt line (if a read is done after this write, this would return 0).

Note: In the MMCi.MMCHS_STAT register, Card Interrupt (CIRQ) and Error Interrupt (ERRI) bits cannot be cleared.
The MMCi.MMCHS_STAT[8] CIRQ status bit must be masked by disabling the MMCi.MMCHS_IE[8] CIRQ_ENABLE bit (set to 0x0), then the interrupt routine must clear SDIO interrupt source in SDIO card common control register (CCCR). See the *Interconnect* chapter for more information.
The MMCi.MMCHS_STAT[15] ERRI bit is automatically cleared when all status bits in MMCi.MMCHS_STAT[31:16] are cleared.

1.3.2.2.2 Polling

When the interrupt capability of an event is disabled in the MMCi.MMCHS_ISE register, the interrupt line is not asserted:

- Software can poll the status bit in the MMCi.MMCHS_STAT register to detect when the corresponding event occurs.
- Writing 1 into the corresponding bit of the MMCi.MMCHS_STAT register clears the interrupt status and does not affect the interrupt line state.

Note: Refer to the previous note concerning CIRQ and ERRI bits clearing.

1.4 MMC/SD/SDIO Functional Description

1.4.1 Description

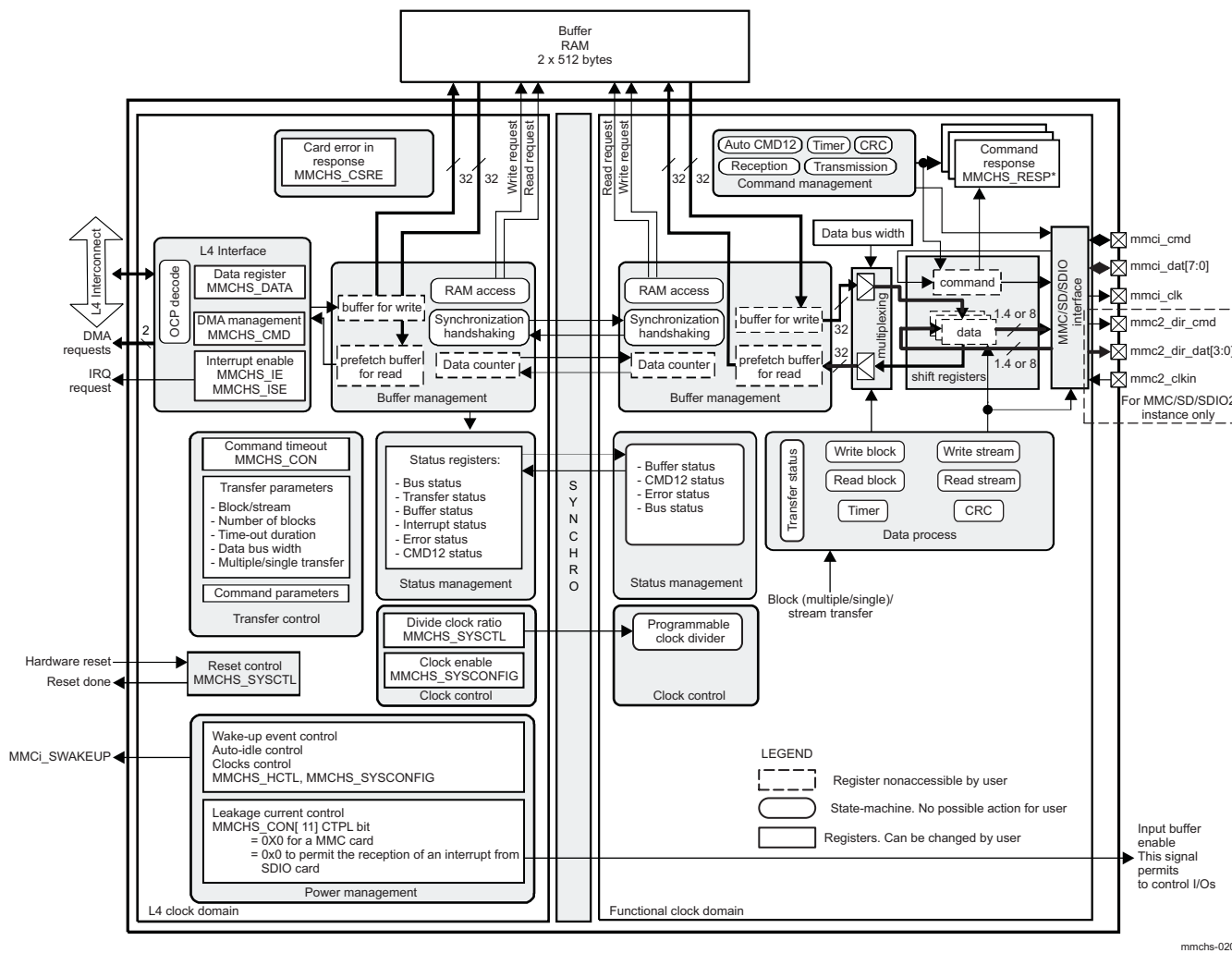
MMC/SD/SDIOi host controller is partitioned into two clock domains:

- The interface L4 clock domain
- The functional clock domain

Any two domains are considered asynchronous to each other, and exchanges between the two domains are synchronized through a synchronization stage and an asynchronous buffer. Data are transferred from one domain to other through the buffer (2*512 RAM).

[Figure 1-20](#) shows a block diagram of the MMC/SD/SDIO host controller.

Figure 1-20. MMC/SD/SDIO Diagram



mmchs-020

In the L4 clock domain, the user can:

- Control the transfer (configure parameters for the transfer)
- Control clock activities and reset
- Manage interrupts and hardware requests
- Consult different status:

MMC/SD/SDIO Functional Description

- Bus
- Buffer
- Interrupts
- Errors
- Transmit and receive data through the buffer

In the functional clock domain, the internal mechanisms perform the transfers of data and commands.

For more detail see the command response registers (MMCi.MMCHS_RSPn registers: [Table 1-54](#), [Table 1-56](#), [Table 1-58](#), and [Table 1-60](#)).

CAUTION

Read access to the command response registers is allowed only when the command process is completed.

1.4.2 Mode Selection

The MMC/SD/SDIO host controller can be use in two modes: MMC and SD/SDIO modes. It has been designed to be the most transparent with the type of card.

The type of the card connected is differentiated by the software initialization procedure. Software identifies the type of card connected during software initialization. For each given card type, there are corresponding commands. Some commands are not supported by all cards. See the *Multimedia Card System Specification*, v4.2, the *SD Memory Card Specifications*, v2.0, and the *SDIO Card Specification, Part E1*, v1.10, for more details.

The purpose of the module is to transfer commands and data, to whatever card is connected, respecting the protocol of the connected card.

Writes and reads to the card must respect the appropriate protocol of that card.

1.4.3 Buffer Management

1.4.3.1 Data Buffer

The MMC/SD/SDIOi host controller uses a data buffer divided into two 512-byte portions that are 32 bits wide by 128 words deep. This buffer transfers data from one data bus (Interconnect) to another data bus (SD SDIO or MMC card bus) and vice versa.

The buffer is the heart of the interface and ensures the transfer between the two interfaces (L4 and the card).

To enhance performance, the data buffer is completed by a prefetch register and a post-write buffer that are not accessible by the host controller.

The read access time of the prefetch register is faster than the one of the data buffer. The prefetch register allows data to be read from the data buffer at an increased speed by preloading data into the prefetch register.

The entry point of the data buffer for the two portions, the prefetch buffer and the post-write buffer, is the 32-bit register MMCI.MMCHS_DATA. A write access to the MMCI.MMCHS_DATA register followed by a read access from the MMCI.MMCHS_DATA register corresponds to a write access to the post-write buffer followed by a read access to the prefetch buffer. As a consequence, it is normal that the data of the write access to the MMCI.MMCHS_DATA register and the data of the read access to the MMCI.MMCHS_DATA register are different.

The number of 32-bit accesses to the MMCI.MMCHS_DATA register that are needed to read (or write) a data block with a size of MMCI.MMCHS_BLK[10:0] BLEN, and equals the rounded up result of BLEN divided by 4.

The maximum block size supported by the host controller is 1024 bytes. This value is hard-coded in the register MMCI.MMCHS_CAPA[17:16] MBL field and cannot be changed.

A read access to the MMCI.MMCHS_DATA register is allowed only when the buffer read enable status is set to 1 (MMCI.MMCHS_PSTATE[11] BRE); otherwise, a bad access (MMCI.MMCHS_STAT[29] BADA) is signaled.

A write access to the MMCI.MMCHS_DATA register is allowed only when the buffer write enable status is set to 1 (MMCI.MMCHS_PSTATE[10] BWE); otherwise, a bad access (MMCI.MMCHS_STAT[29] BADA) is signaled and the data is not written.

The data buffer has two modes of operation to store and read of the first and second portions of the data buffer:

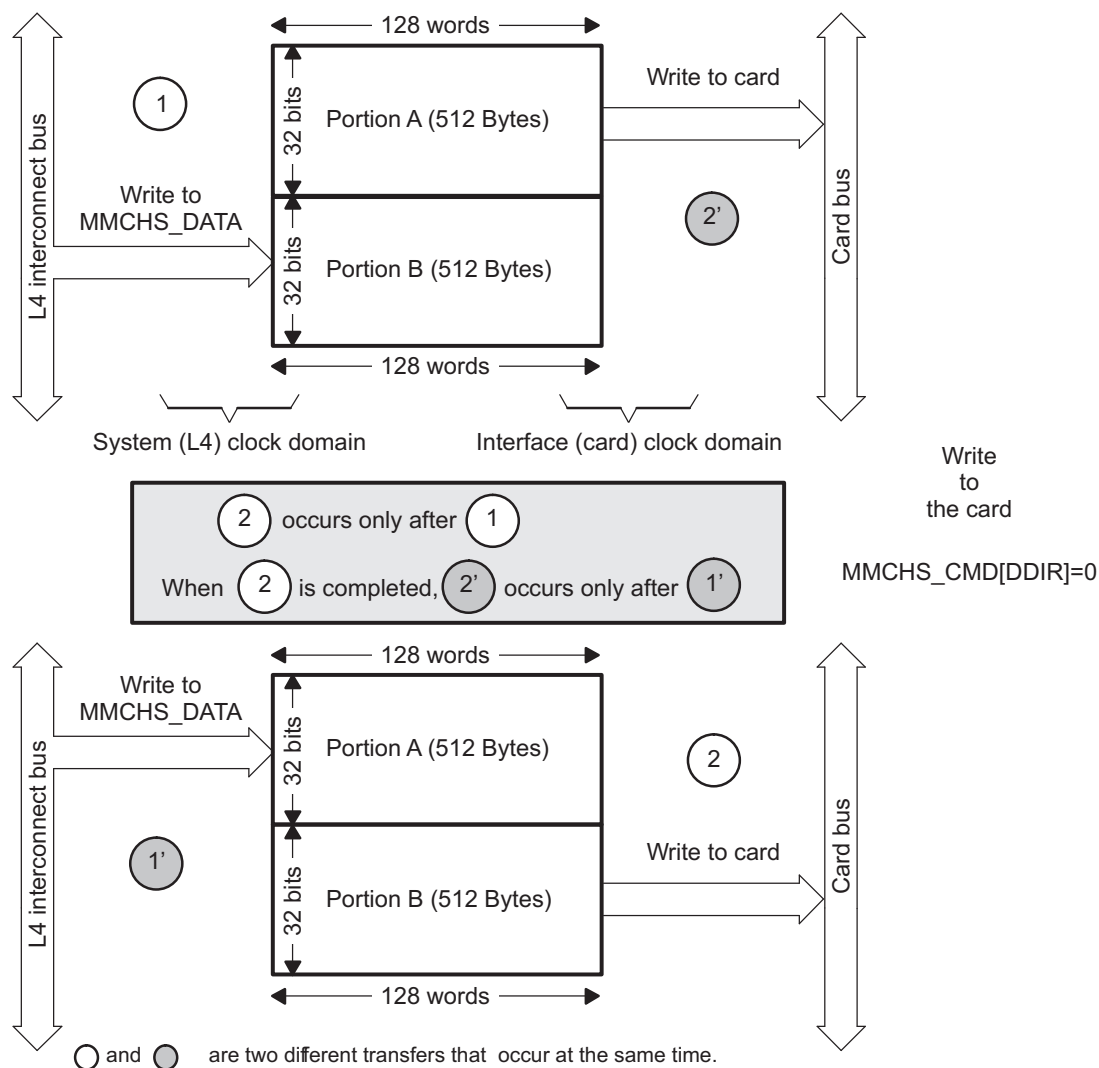
- When the size of the data block to transfer is less than or equal to 512 bytes, meaning the value written in BLEN is less than or equal to 0x200, two data transfers can occur from one data bus to the other data bus and vice versa at the same time. The MMC/SD/SDIOi host controller uses the two portions of the data buffer in a pingpong manner so that storing and reading of the first and second portions of the data buffer are automatically interchanged from time to time so that data may be read from one portion (for instance, through a DMA read access on the interconnect bus) while data (for instance, from the card) is being stored into the other portion and vice versa.

CAUTION

The MMCI.MMCHS_CMD[4] DDIR bit must be configured before a transfer to indicate the direction of the transfer.

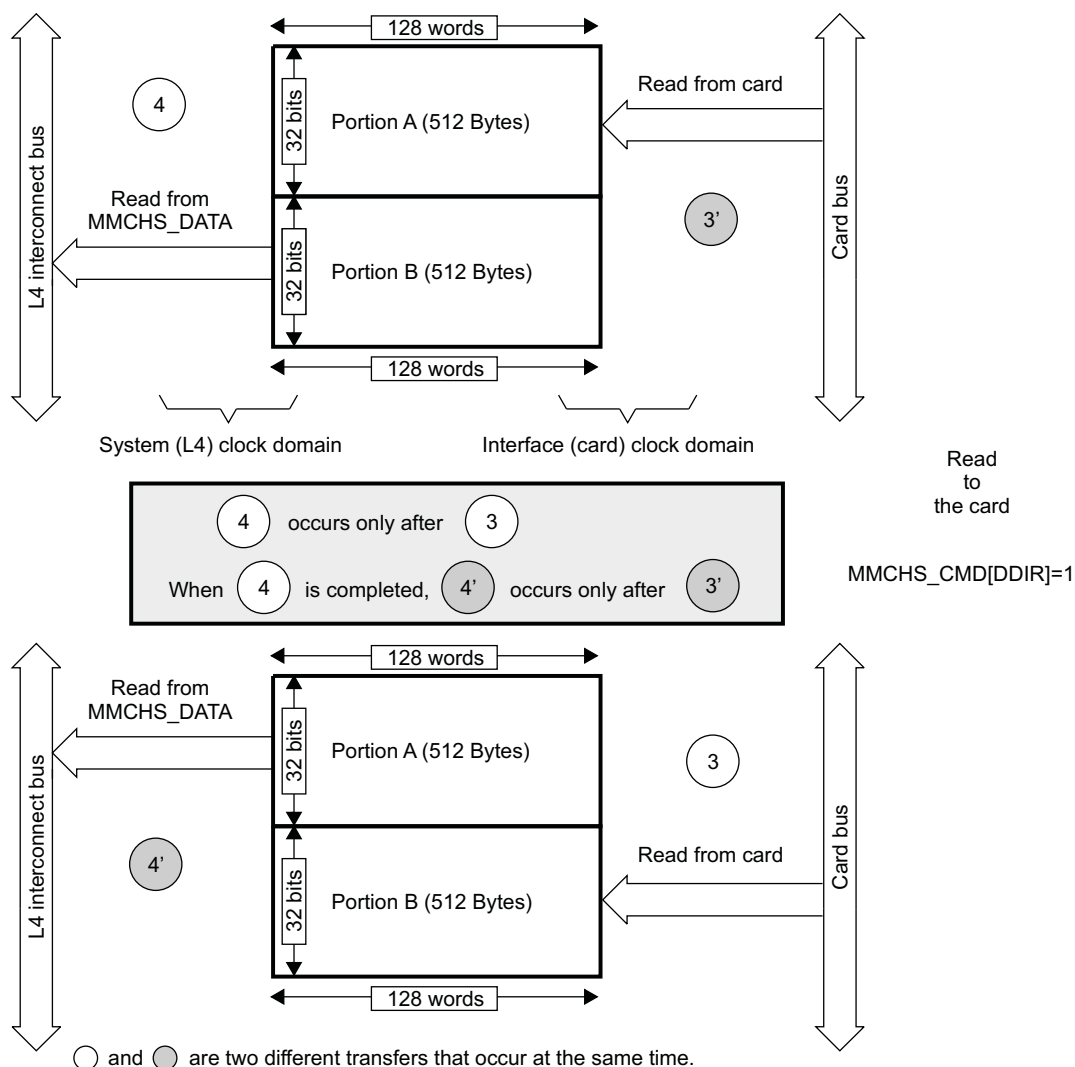
Figure 1-21 and Figure 1-22 show the buffer management for a write and for a read, respectively.

Figure 1-21. Buffer Management for a Write



mmchs-047

Figure 1-22. Buffer Management for a Read



mmchs-043

- When the size of the data block to transfer is larger than 512 bytes, meaning the value written in BLEN is 0x201 or larger, only one data transfer can occur from one data bus to the other data bus at a time. The MMC/SD/SDIOi host controller uses the entire data buffer as a single 1024-byte portion. In this mode, a bad access (MMCi.MMCHS_STAT[29] BADA) is signaled when two data transfers occur from one data bus to the other data bus and vice versa at the same time.

1.4.3.1.1 Data Buffer Status

The data buffer status is defined in the following interrupt status register and status register:

- Interrupt status registers (see Table 1-70):
 - MMCi.MMCHS_STAT[29] BADA: Bad access to data space
 - MMCi.MMCHS_STAT[5] BRR: Buffer read ready
 - MMCi.MMCHS_STAT[4] BWR: Buffer write ready
- Status registers (see Table 1-64):
 - MMCi.MMCHS_PSTATE[11] BRE: Buffer read enable
 - MMCi.MMCHS_PSTATE[10] BWE: Buffer write enable

1.4.4 Transfer Process

The process of a transfer is dependent on the type of command. It can be with or without a response, with or without data.

1.4.4.1 Different Types of Commands

Different types of commands are specific to MMC, SD, or SDIO cards. See the *Multimedia Card System Specification*, v4.2, the *SD Memory Card Specifications*, v2.0, the *SDIO Card Specification, Part E1*, v1.10, or the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v1.00, for more details.

1.4.4.2 Different Types of Responses

Different types of responses are specific to MMC, SD, or SDIO cards. See the *Multimedia Card System Specification*, v4.2, the *SD Memory Card Specifications*, v2.0, the *SDIO Card Specification, Part E1*, v1.10, or the *SD Card Specification, Part A2, SD Host Controller Standard Specification*, v1.00, for more details.

Table 1-5 shows how the MMC, SD, and SDIO responses are stored in the MMCHS_RSPxx registers.

Table 1-5. MMC, SD, SDIO responses in the MMCHS_RSPxx registers

Kind of Response	Response Field	Response Register
R1, R1b (normal response), R3, R4, R5, R5b, R6	RESP[39:8] ⁽¹⁾	MMCHS_RSP10[31:0]
R1b (Auto CMD12 response)	RESP[39:8] ⁽¹⁾	MMCHS_RSP76[31:0]
R2	RESP[127:0] ⁽¹⁾	MMCHS_RSP76[31:0] MMCHS_RSP54[31:0] MMCHS_RSP32[31:0] MMCHS_RSP10[31:0]

⁽¹⁾ RESP refers to the command response format described in the specifications mentioned above.

When the host controller modifies part of the MMCHS_RSPxx registers, it preserves the unmodified bits.

The host controller stores the Auto CMD12 response in the MMCHS_RSP76[31:0] register because the Host Controller may have a multiple block data DAT line transfer executing concurrently with a command. This allows the host controller to avoid overwriting the Auto CMD12 response with the command response stored in MMCHS_RSP10 register and vice versa.

1.4.5 Transfer or Command Status and Errors Reporting

Flags in the MMC/SD/SDIOi host controller show status of communication with the card:

- A timeout (of a command, a data, or a response)
- A CRC

Error conditions generate interrupts. See Table 1-6 and register description for more details.

Table 1-6. CC and TC Values Upon Error Detected

Error hold in the MMCi.MMCHS_STAT register	CC	TC	Comments
29 BADA			No dependency with CC nor TC BADA is related to MMCHS_DATA register accesses. Its assertion is not dependent of the ongoing transfer.
28 CERR	1		CC is set upon CERR.
22 DEB		1	TC is set upon DEB.

Table 1-6. CC and TC Values Upon Error Detected (continued)

Error hold in the MMCi.MMCHS_STAT register		CC	TC	Comments
21	DCRC		1	TC is set upon DCRC.
20	DTO			DTO and TC are mutually exclusive DCRC and DEB cannot occur with DTO.
19	CIE	1		CC is set upon CIE.
18	CEB	1		CC is set upon CEB.
17	CCRC	1		CC can be set upon CCRC - See CTO comment
16	CTO			CTO and CC are mutually exclusive. CIE, CEB and CERR cannot occur with CTO. CTO can occur at the same time as CCRC: it indicates a command abort due to a contention on CMD line. In this case no CC appears.

1.4.6 Transfer Stop

Whenever a transfer is initiated, the transmission may be willed to stop whereas it is still not finished. Several cases can be faced depending on the transfer type:

- Multiple blocks oriented transfers (for which transfer length is known)
- Continuous stream transfers (which have an infinite length)

Note: Since the MMC/SD/SDIOi controller manages transfers based on a block granularity, the buffer will accept a block only if there is enough space to completely store it. Consequently, if a block is pending in the buffer, no command will be sent to the card because the card clock will be shut off by the controller.

The MMC/SD/SDIOi controller includes two features which makes a transfer stop more convenient and easier to manage:

- Auto CMD12 (for MMC and SD only).
This feature is enabled by setting the MMCi.MMCHS_CMD[2] ACEN bit to 0x1 (this setting is relevant for a MMC/SD transfer with a known number of blocks to transfer). When the Auto CMD12 feature is enabled, the MMC/SD/SDIOi controller will automatically issue a CMD12 command when the expected number of blocks has been exchanged.
- Stop at block gap
This feature is enabled by setting the MMCi.MMCHS_HCTL[16] SBGR bit to 0x1. When enabled, this capability holds the transfer on until the end of a block boundary. If a stop transmission is needed, software can use this pause to send a CMD12 to the card.

Note: For MMC and SD cards, the stop at block gap feature is not supported in READ mode.

For SDIO cards, this setting can be supported in READ mode if the card has a read wait capability.

Table 1-7 shows the common way to stop a transfer, indicating command to send and features to enable.

Table 1-7. MMC/SD/SDIOi Controller Transfer Stop Command Summary

		WRITE transfer		READ transfer	
		SD / MMC	SDIO	SD / MMC	SDIO
Single block		Transfer ends automatically Wait TC	Transfer ends automatically Wait TC	Transfer ends automatically Wait TC	Transfer ends automatically Wait TC
Multi blocks (finite or infinite)	Before the programmed block boundary	Send CMD12 Wait TC	Send CMD52 Wait TC	Send CMD12 Wait TC	Send CMD52 Wait TC
	Stop at the end of the transfer (finite transfer only)	Auto CMD12 active Transfer ends automatically Wait TC	Set MMCi.MMCHS_HC TL[16] SBGR bit to 0x1. Send CMD52 Wait TC	Auto CMD12 active Transfer ends automatically Wait TC	If READ_WAIT supported Stop at block gap Wait TC If READ_WAIT not supported Send CMD52 Wait TC

Note: The MMC/SD/SDIOi controller will send the stop command to the card on a block boundary, regardless the moment the command was written to the controller registers.

1.4.7 MMC CE-ATA Command Completion Disable Management

The MMC/SD/SDIOi host controller supports CE-ATA features, in particular the detection of command completion token. When a command that requires a command completion signal ([MMCHS_CON\[12\]](#) CEATA and [MMCHS_CMD\[2\]](#) ACEN set to 1) is launched, host system is no longer allowed to emit a new command in parallel of data transfer unless it is a command completion disable.

The settings to emit a command completion disable token follow:

- [MMCHS_CON\[12\]](#) CEATA is set to 1.
- [MMCHS_CON\[2\]](#) HR set to 1.
- Clear the [MMCHS_ARG](#) register.
- Write into [MMCHS_CMD](#) register with value 0x00000000.

When a command completion disable token was emitted (that is, [MMCHS_STAT\[0\]](#) CC received), the host system is again allowed to emit another type of command (for example a transfer abort command CMD12 to abort transfer).

A critical case can be met when command completion signal disable (CCSD) is emitted during the last data block transfer, the sequence on command line could be sent very close to command completion signal (CCS) token sent by the card.

Three cases can be met:

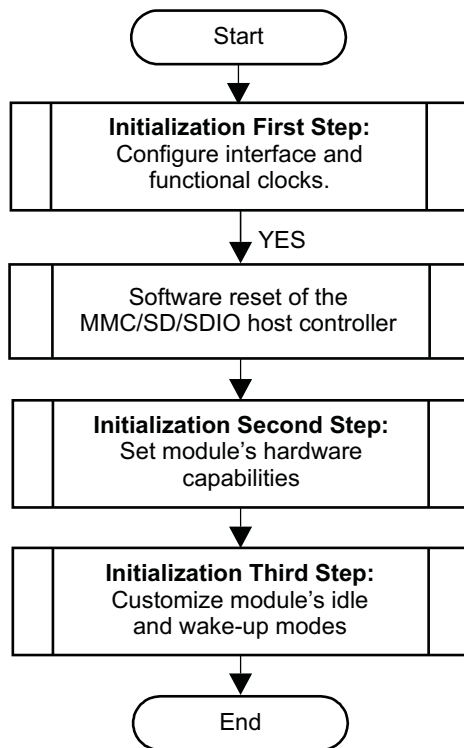
- CCS is receive just before CCSD is emitted:
An interrupt CIRQ is generated with CCS detection, CCSD is transmitted to card then an interrupt CC is generated when CCSD ends. In this case, card consider the CCSD sequence.
- CCS is not generated or generated during the CCSD transfer:
The CCS bit cannot be detected (conflict is not possible as they drive the same level on command line, then no CIRQ interrupt is generated; besides CC interrupt is generated when CCSD ends).
- CCS is generated without CCSD token required:
Only the interrupt CIRQ is generated when CCS is detected.

1.5 MMC/SD/SDIO Basic Programming Model

1.5.1 MMC/SD/SDIO Host Controller Initialization Flow

Figure 1-23 shows the general boot process.

Figure 1-23. MMC/SD/SDIO Controller Meta Initialization Steps



mmchs-024

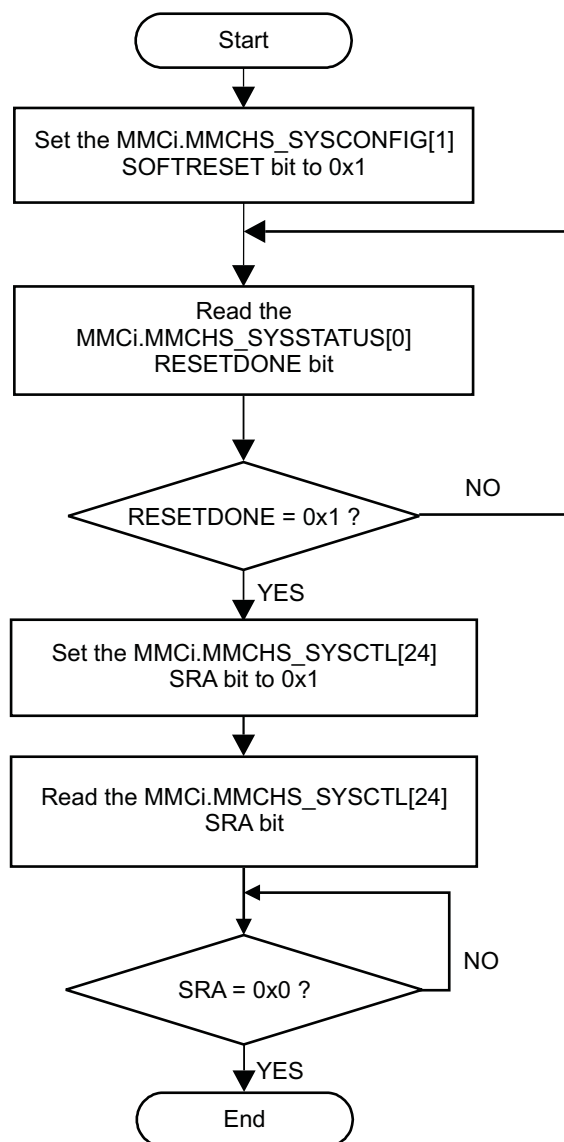
1.5.1.1 Enable Interface and Functional clock for MMC Controller

Prior to any MMCHS register access one must enable MMCHS interface clock and functional clock in PRCM module registers PRCM.CM_ICLKEN1_CORE and PRCM.CM_FCLKEN1_CORE. Please refer to the *Power, Reset, and Clock Management* chapter.

1.5.1.2 MMCHS Soft Reset Flow

Figure 1-24 shows the soft reset process of MMCHS controller.

Figure 1-24. MMC/SD/SDIO Controller Software Reset Flow



mmchs-025

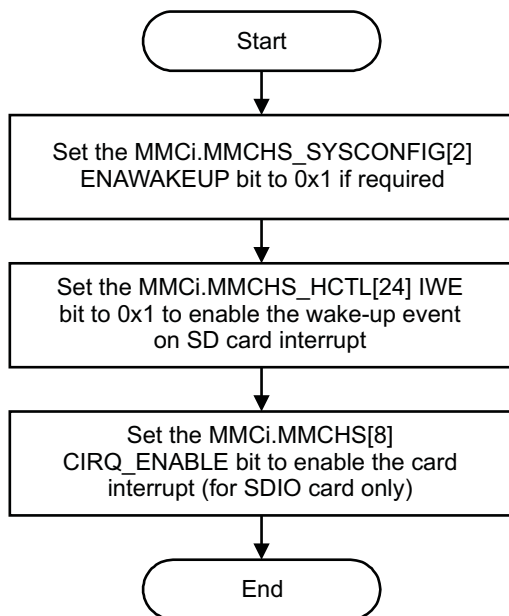
1.5.1.3 Set MMCHS Default Capabilities

Software must read capabilities (in boot ROM for instance) and is allowed to set (write) `MMCi.MMCHS_CAPA[26:24]` and `MMCi.MMCHS_CUR_CAPA[23:0]` registers before the MMC/SD/SDIO host driver is started.

1.5.1.4 Wakeup Configuration

Figure 1-25 details MMCHS controller wakeup configuration.

Figure 1-25. MMC/SD/SDIO Controller Wakeup Configuration

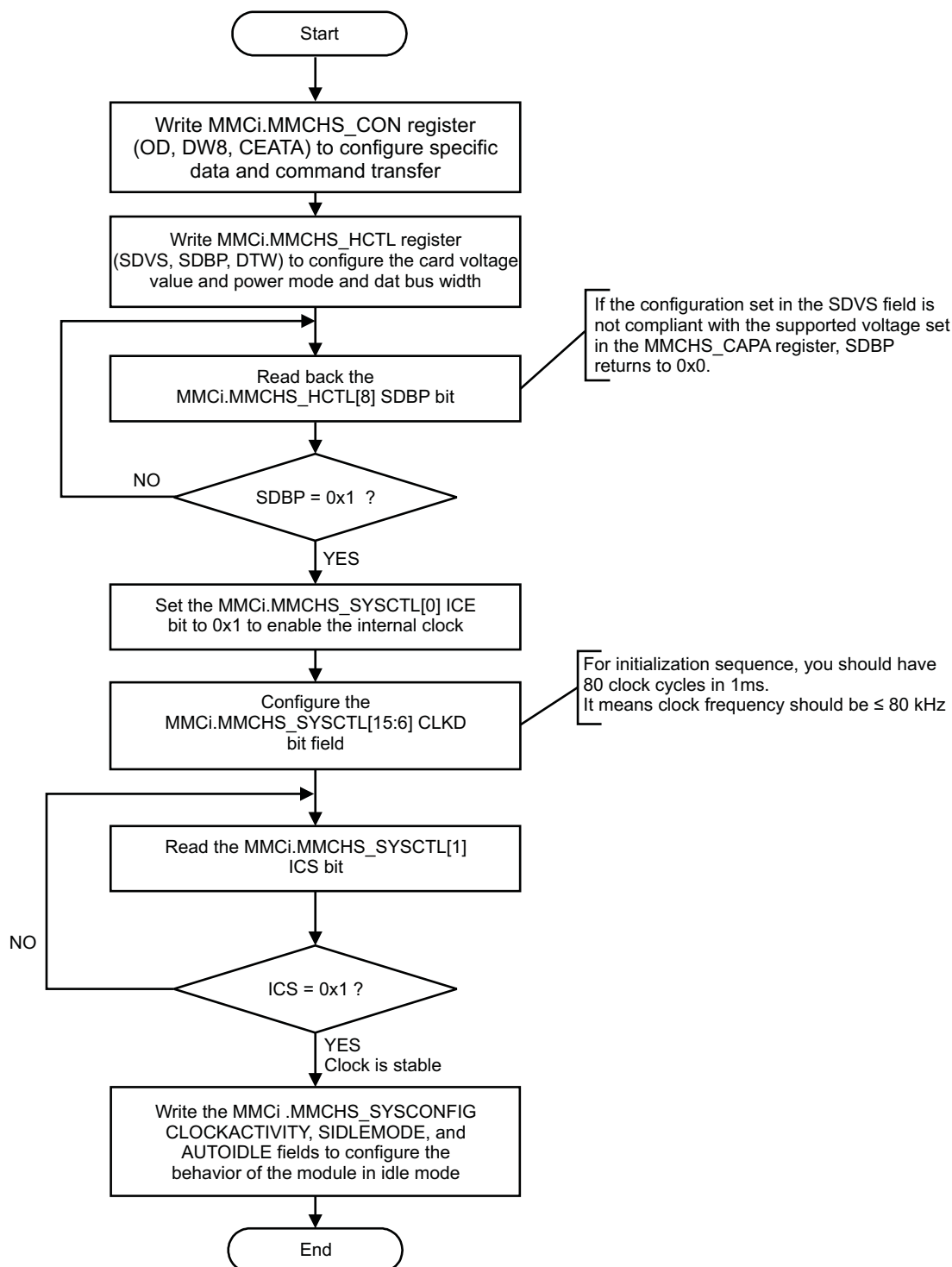


mmchs-027

1.5.1.5 MMC Host and Bus Configuration

Figure 1-26 details MMC bus configuration process.

Figure 1-26. MMC/SD/SDIO Controller Bus Configuration



mmchs-028

1.5.2 Basic Operations for MMC/SD/SDIO Host Controller

The MMC/SD/SDIO host controller performs data transfers: data to card (referred to as write transfers) and data from card (referred to as read transfers).

The host controller requires transfers to run on a block-by-block basis, rather than on a DMA burst size basis. A single DMA request (or block request interrupt) is signaled for each block. Pipelining is supported as long as the block size is less than one half of the memory buffer size.

1.5.2.1 Card Detection, Identification, and Selection

Figure 1-27 and Figure 1-28 show the card identification and selection process.

Figure 1-27. MMC/SD/SDIO Controller Card Identification and Selection - part 1

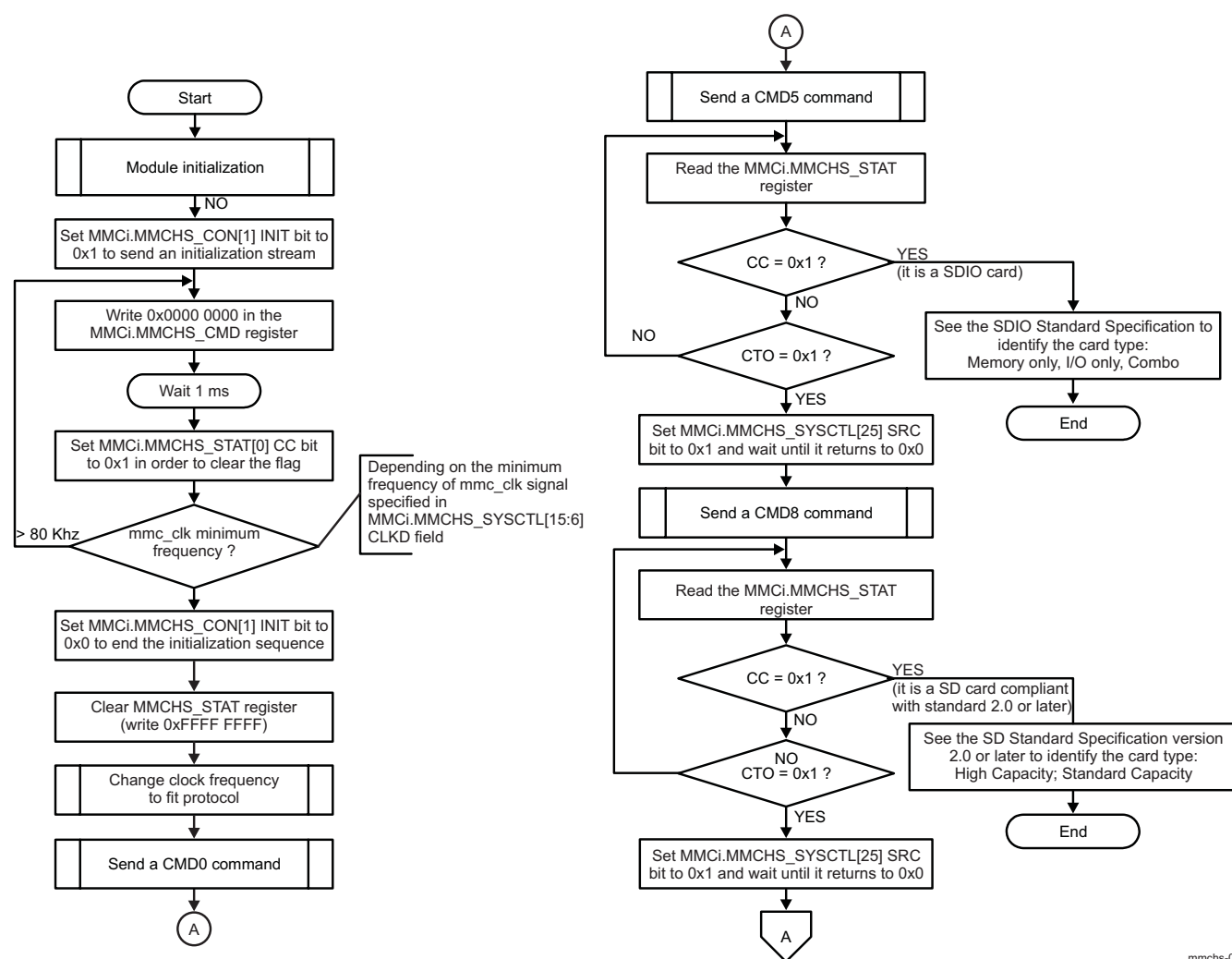
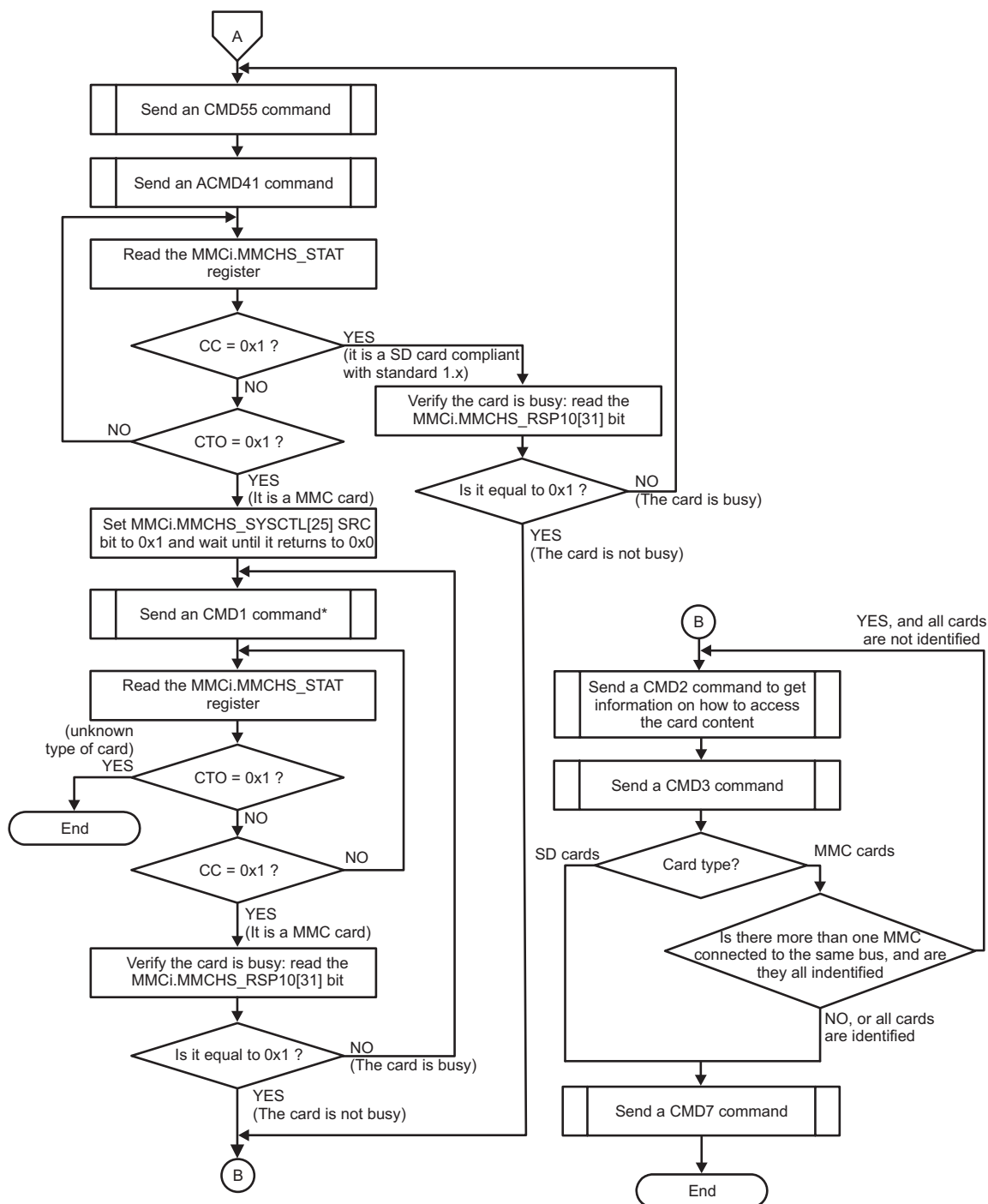


Figure 1-28. MMC/SD/SDIO Controller Card Identification and Selection - part 2



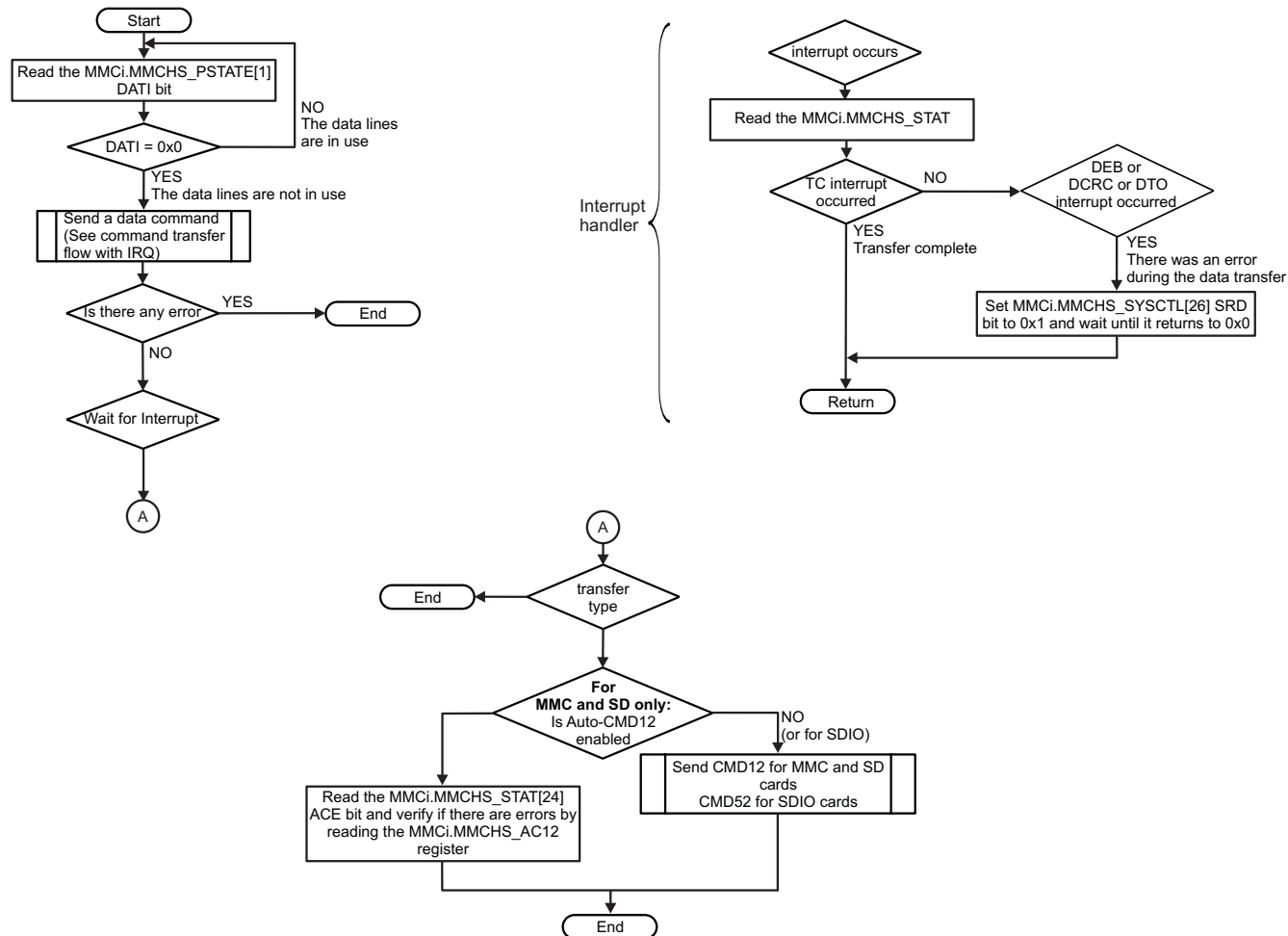
*With OCR 0.
In case of a CMD1 with OCR=0, a second CMD1 must be sent to the card with the "negotiated" voltage.

108-031

1.5.2.2 Read/Write Transfer Flow in DMA Mode with Interrupt

Figure 1-29 describes the read and write protocol in DMA mode with interrupt signaling. Refer to the DMA chapter for more information on the DMA settings.

Figure 1-29. MMC/SD/SDIO Controller Read/Write Transfer Flow in DMA mode with interrupt

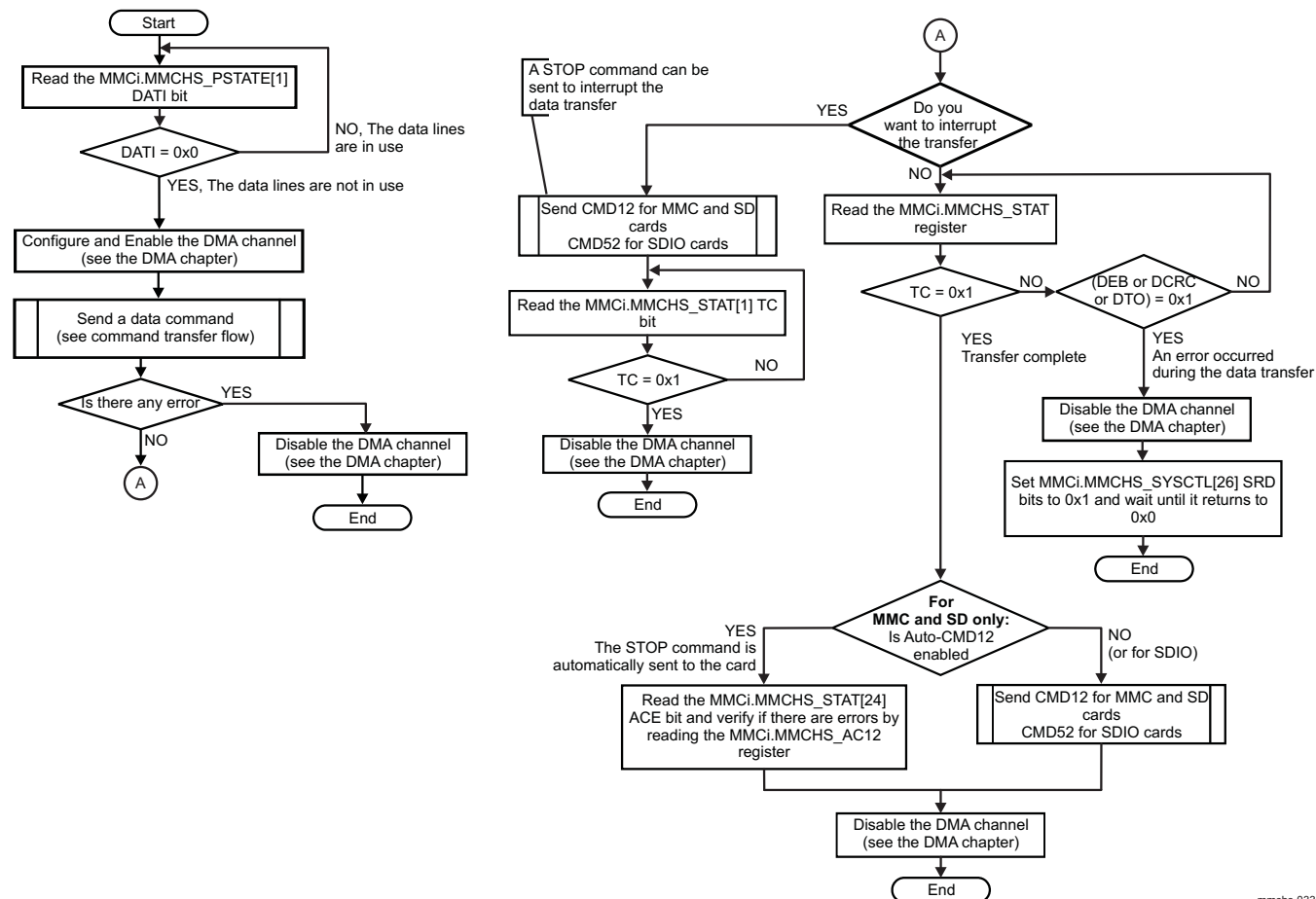


mmchs-032

1.5.2.3 Read/Write Transfer Flow in DMA Mode with Polling

Figure 1-30 describes the read and write protocol in DMA mode. Refer to the *DMA* chapter for more information on the DMA settings.

Figure 1-30. MMC/SD/SDIO Controller Read/Write Transfer Flow in DMA mode with Polling

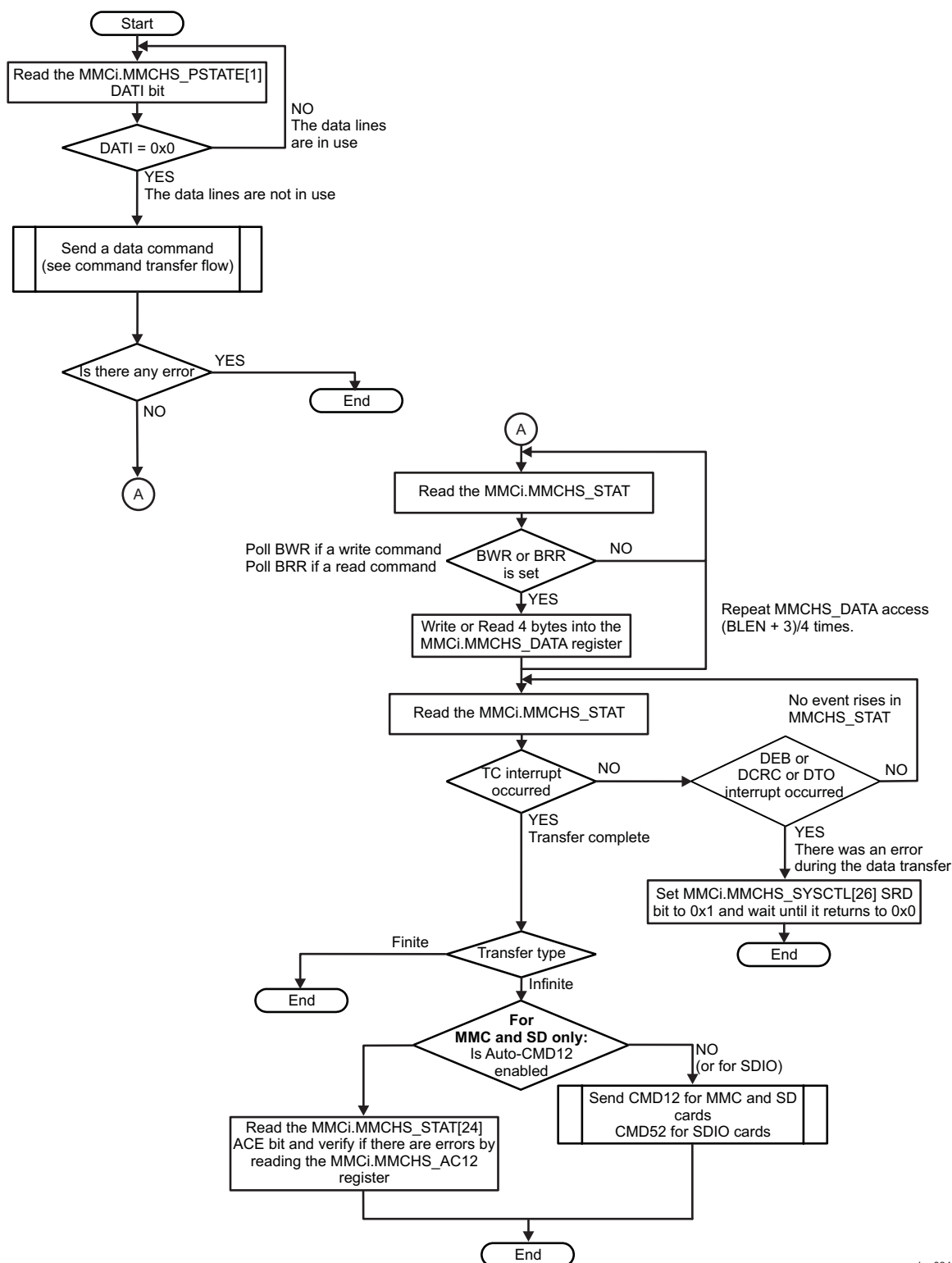


mmchs-033

1.5.2.4 Read/Write Transfer Flow without DMA with Polling

Figure 1-31 describes a read/write transfer without using the DMA and with polling.

Figure 1-31. MMC/SD/SDIO Controller Read/Write Transfer Flow without DMA with Polling

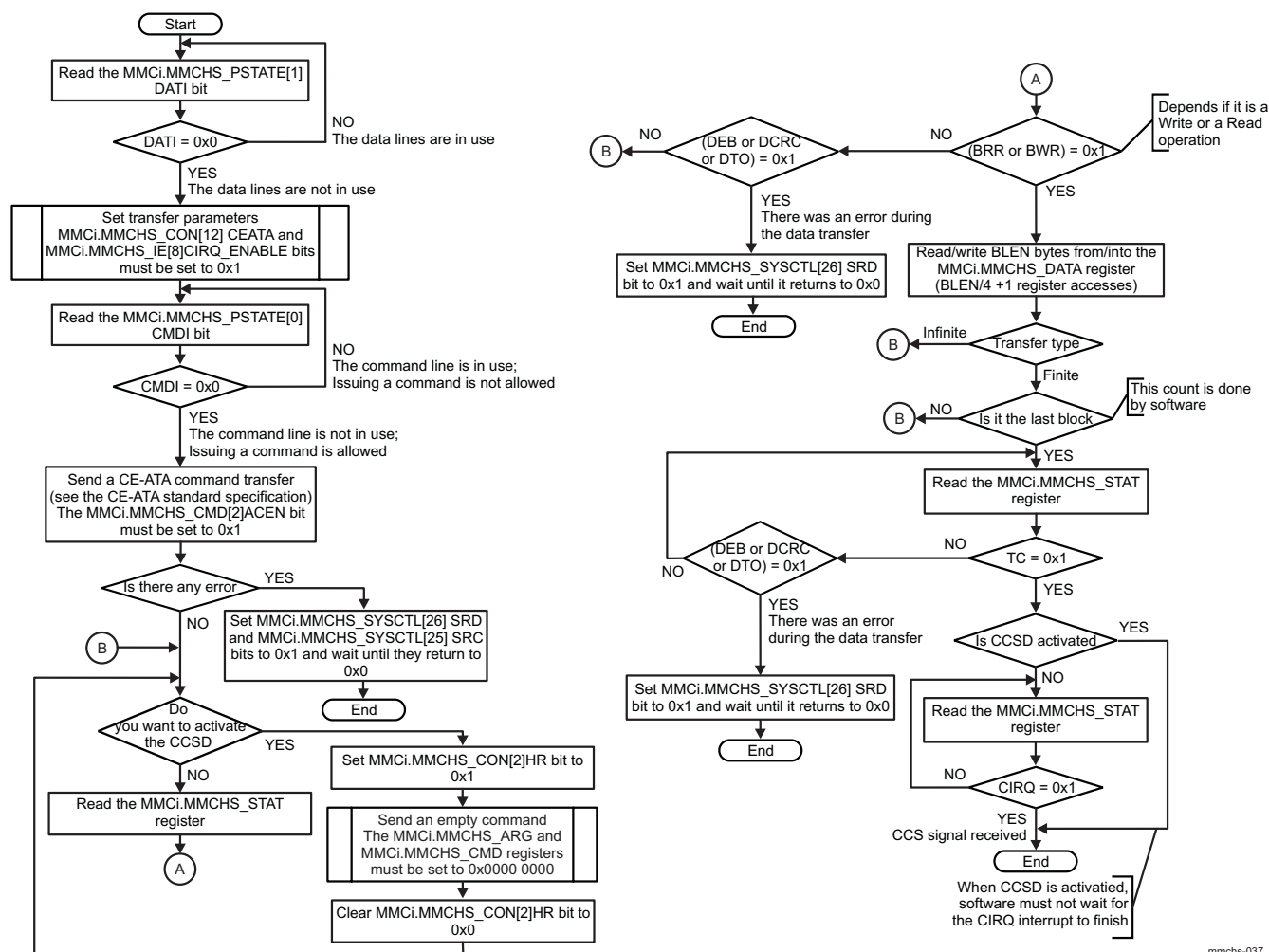


mmchs-034

1.5.2.5 Read/Write Transfer Flow in CE-ATA Mode

Figure 1-32 describes the read and write CE-ATA protocol when in Polling mode.

Figure 1-32. MMC/SD/SDIO Controller Read/Write in CE-ATA mode



mmchs-037

CAUTION

CE-ATA protocol is only supported by MMC cards.

In CE-ATA mode, issuing command during the transfer (except a CCSD command) is not allowed

In CE-ATA mode, infinite transfers are not allowed, only finite transfers are permitted.

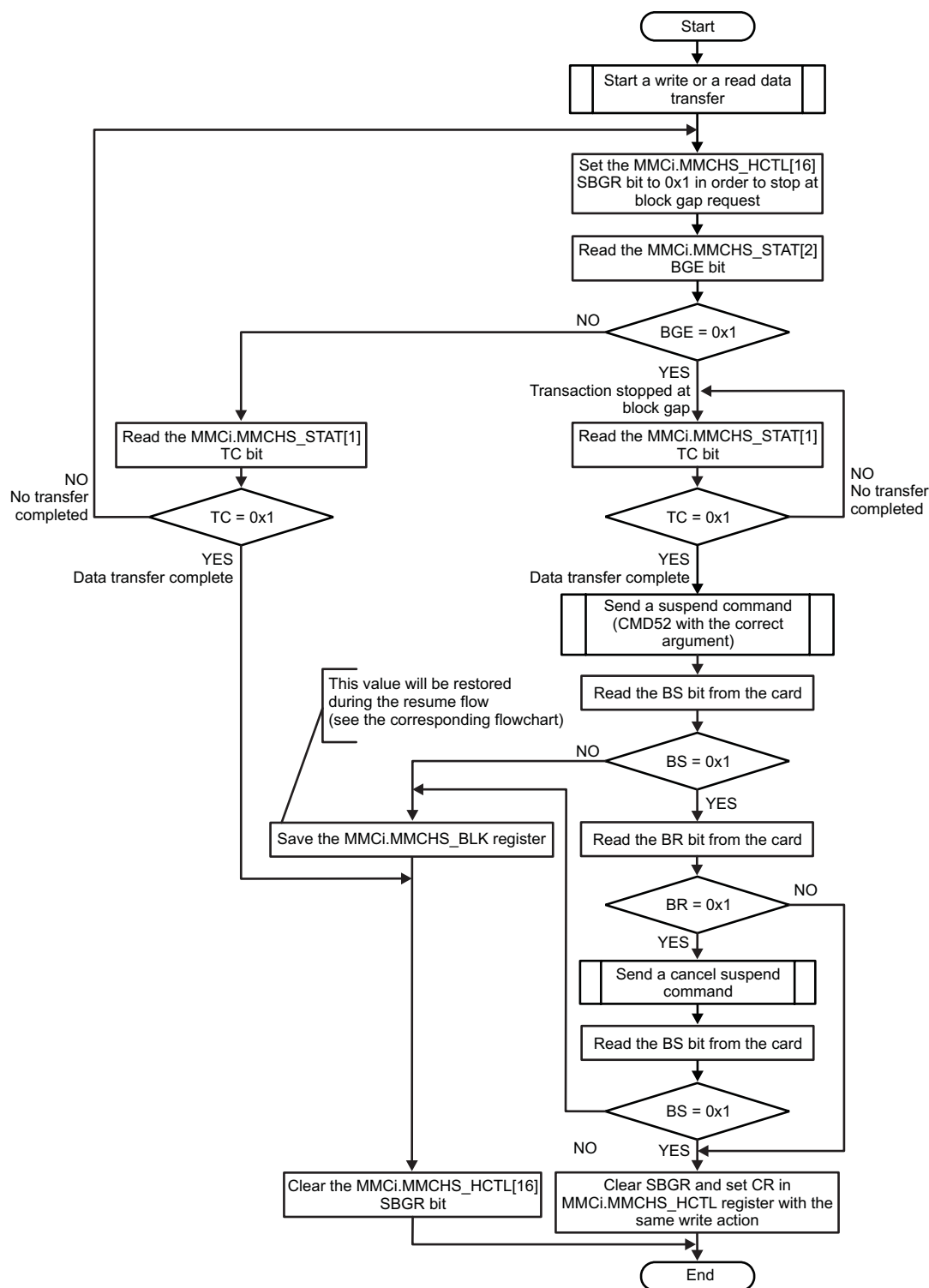
1.5.2.6 Suspend-Resume Flow

The suspend and resume feature is only supported by SDIO cards.

1.5.2.6.1 Suspend Flow

Figure 1-33 describes the suspend flow for SDIO cards.

Figure 1-33. MMC/SD/SDIO Controller Suspend Flow

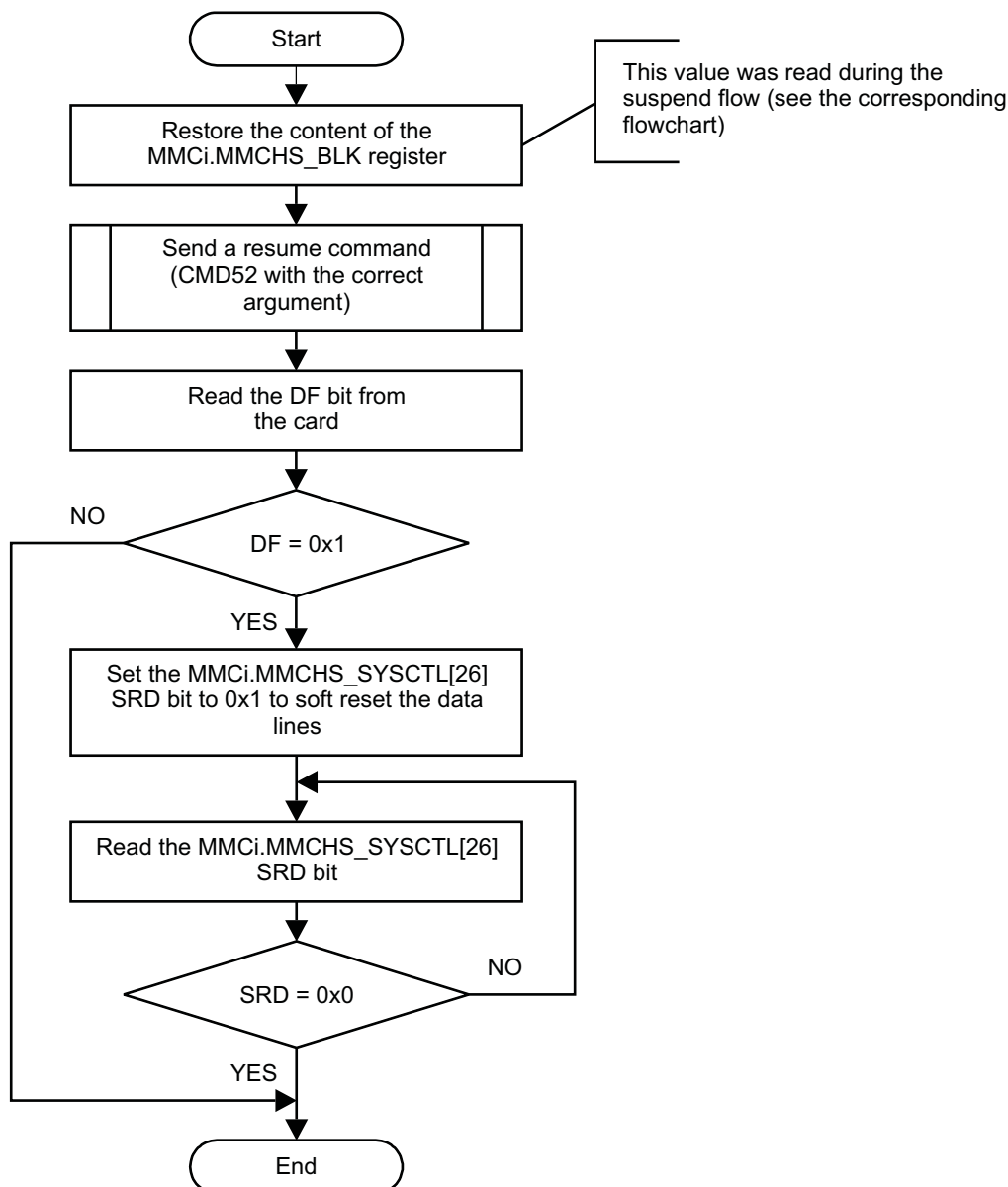


mmchs-038

1.5.2.6.2 Resume Flow

Figure 1-34 describes the resume flow for SDIO cards.

Figure 1-34. MMC/SD/SDIO Controller Resume Flow



mmchs-039

1.5.2.7 Basic Operations - Steps Detailed

1.5.2.7.1 Command Transfer Flow

Figure 1-35 describes how to send a command to the card using polling instead of interrupts for event signaling.

Figure 1-35. MMC/SD/SDIO Controller Command Transfer Flow with polling

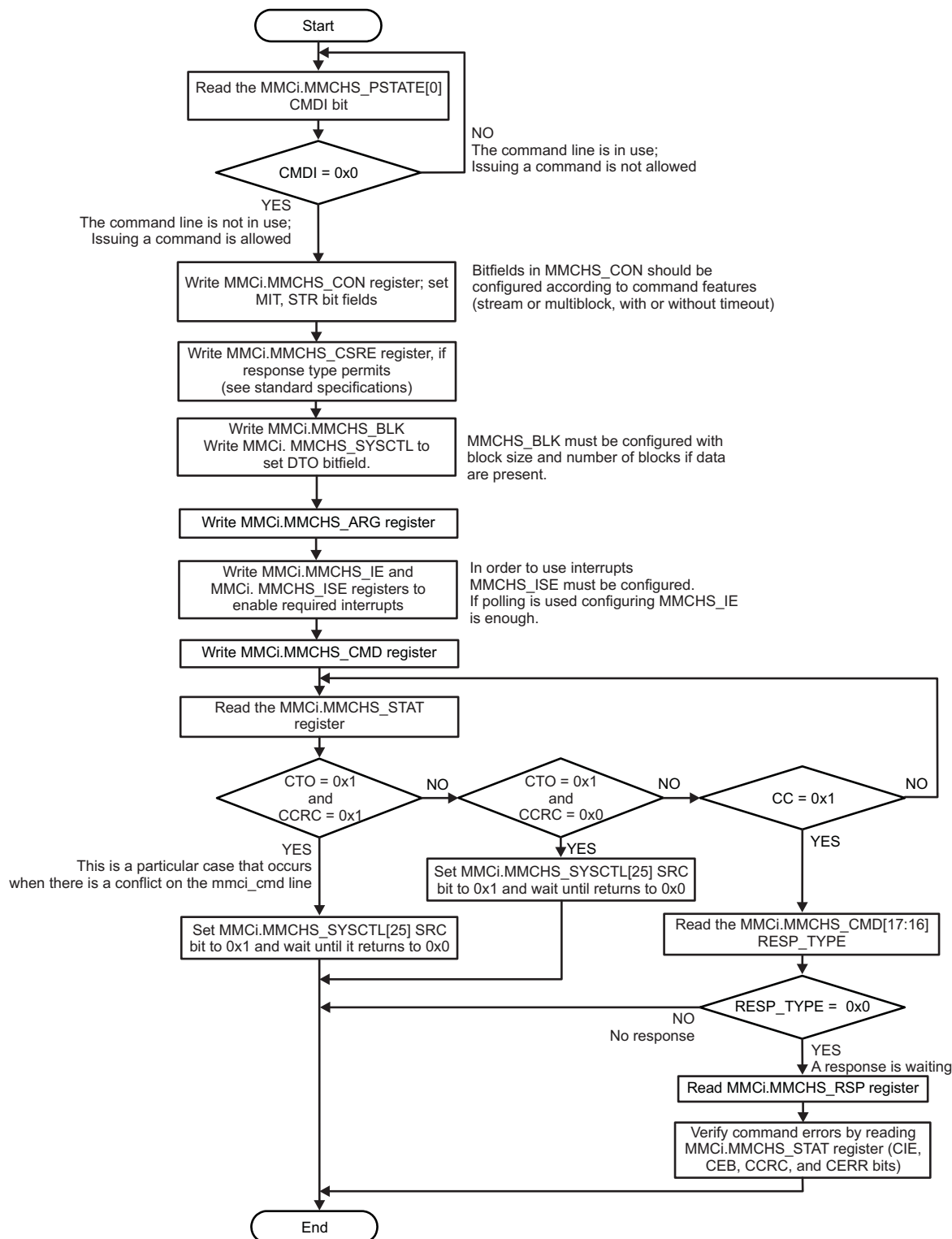
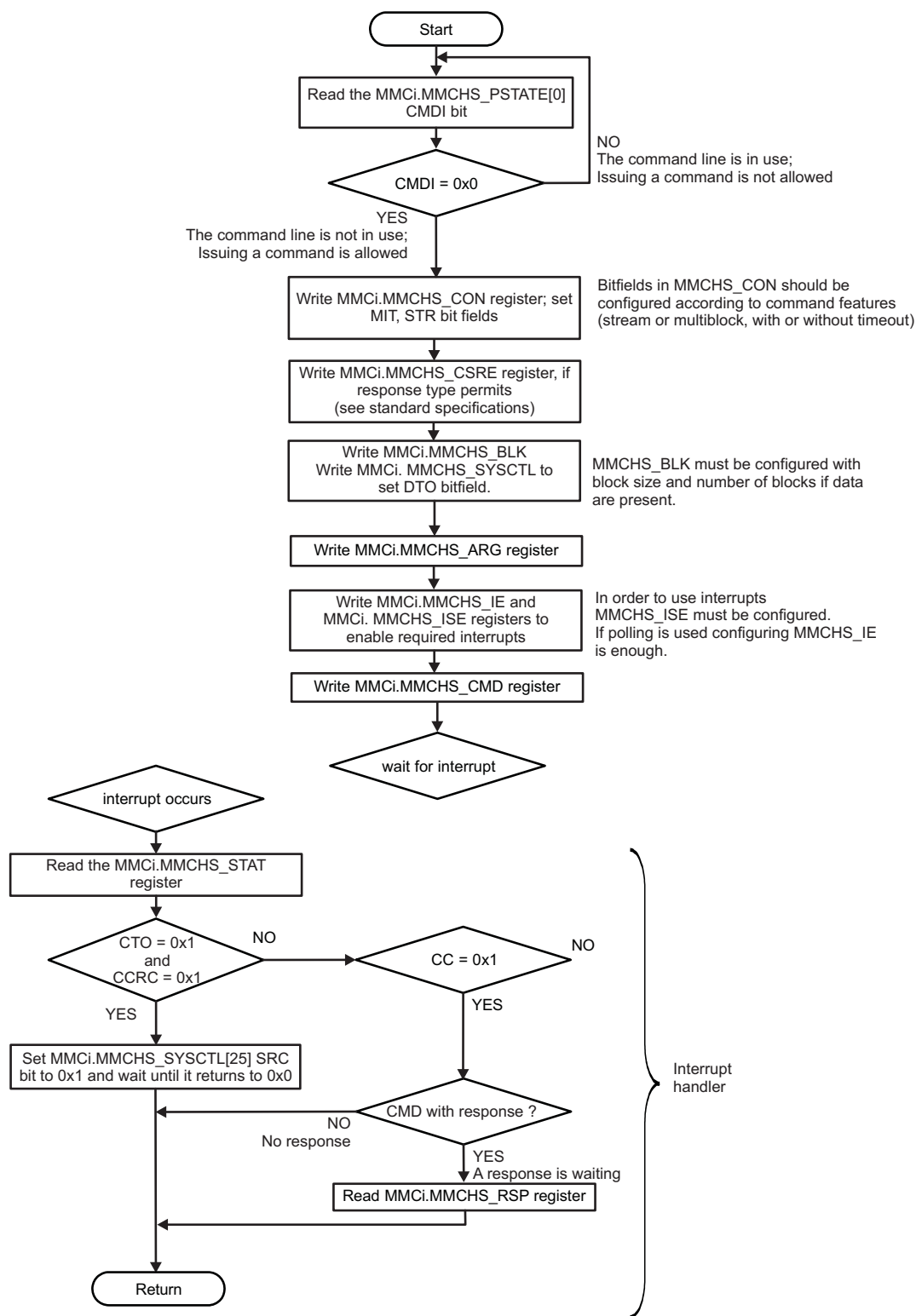


Figure 1-36 describes how to send a command to the card using interrupts for event signaling.

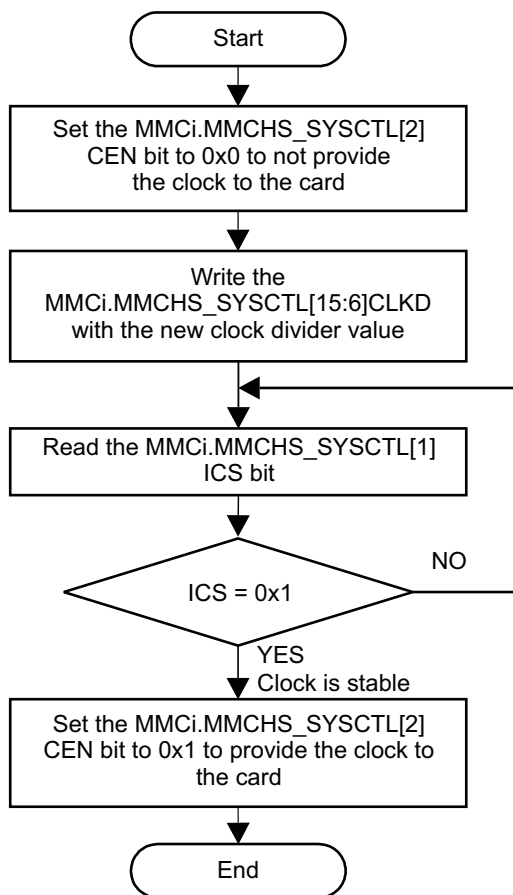
Figure 1-36. MMC/SD/SDIO Controller Command Transfer Flow with interrupts



1.5.2.7.2 MMCHS Clock Frequency Change

Figure 1-37 describes the different steps that allow to change the MMC/SD/SDIO output clock frequency.

Figure 1-37. MMC/SD/SDIO Controller Clock Frequency Change Flow



mmchs-042

1.5.3 MMC/SD/SDIO1 Bus Voltage Selection

The MMC/SD/SDIO1 controller can operate with two types of card voltages: 1.8 V and 3.0 V. For this reason, dual voltage pads are implemented on this interface. For technological concerns those pads must have an internal bias voltage reference to operate. The PBIAS_LITE module supplies this bias voltage, depending on the CONTROL.CONTROL_PBIAS_LITE register settings.

See the *Extended-Drain I/O Pin and PBIAS Cell* for more information about the PBIAS_LITE cell.

The *Extended-Drain I/Os and PBIAS Cell Basic Programming Guide* describes the steps involved in transitioning from 1.8 V to 3.0 V and from 3.0 V to 1.8 V, applicable to the MMC/SD/SDIO1 controller.

CAUTION

The BIAS voltage must be set using the procedure described in the *Extended-Drain I/Os and PBIAS Cell Basic Programming Guide*. Failure to follow this procedure can damage the MMCHS interface.

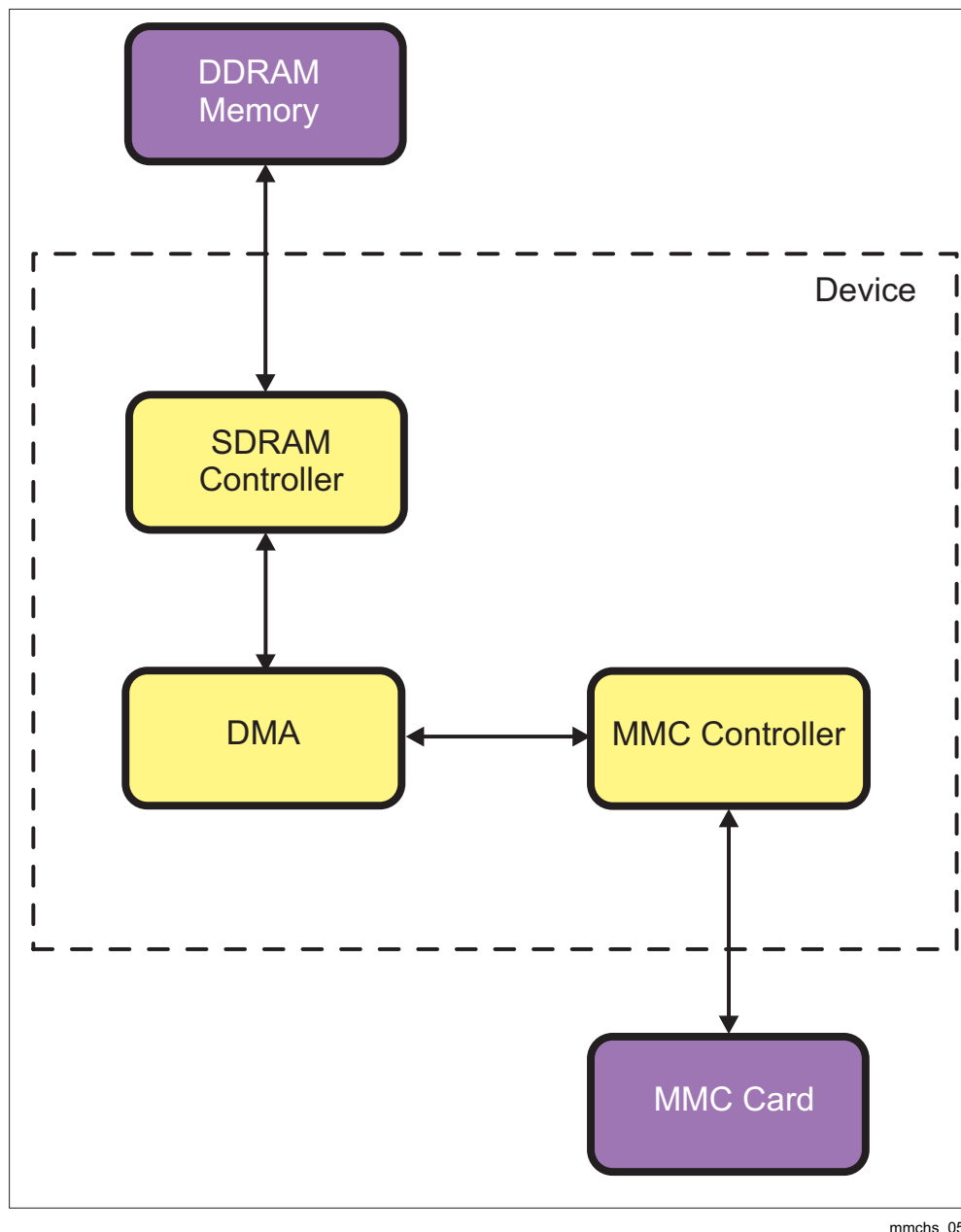
1.6 MMC/SD/SDIO Use Cases and Tips

1.6.1 MMCHS Controller Usage

1.6.1.1 Overview

The MMCHS controller is in charge of managing raw data storage in a MMC memory card. The MMCHS controller transfers data between DDRAM and MMC card. [Figure 1-38](#) gives an overview of MMCHS controller position in the use case.

Figure 1-38. Overview



mmchs_050

For the Camcorder use case, the MMCHS controller is configured to operate with the following features :

- High speed mode with a card clock frequency of 48 MHz.
- 8 data lines.

- 1.8 V and 3.0 V voltage capabilities.

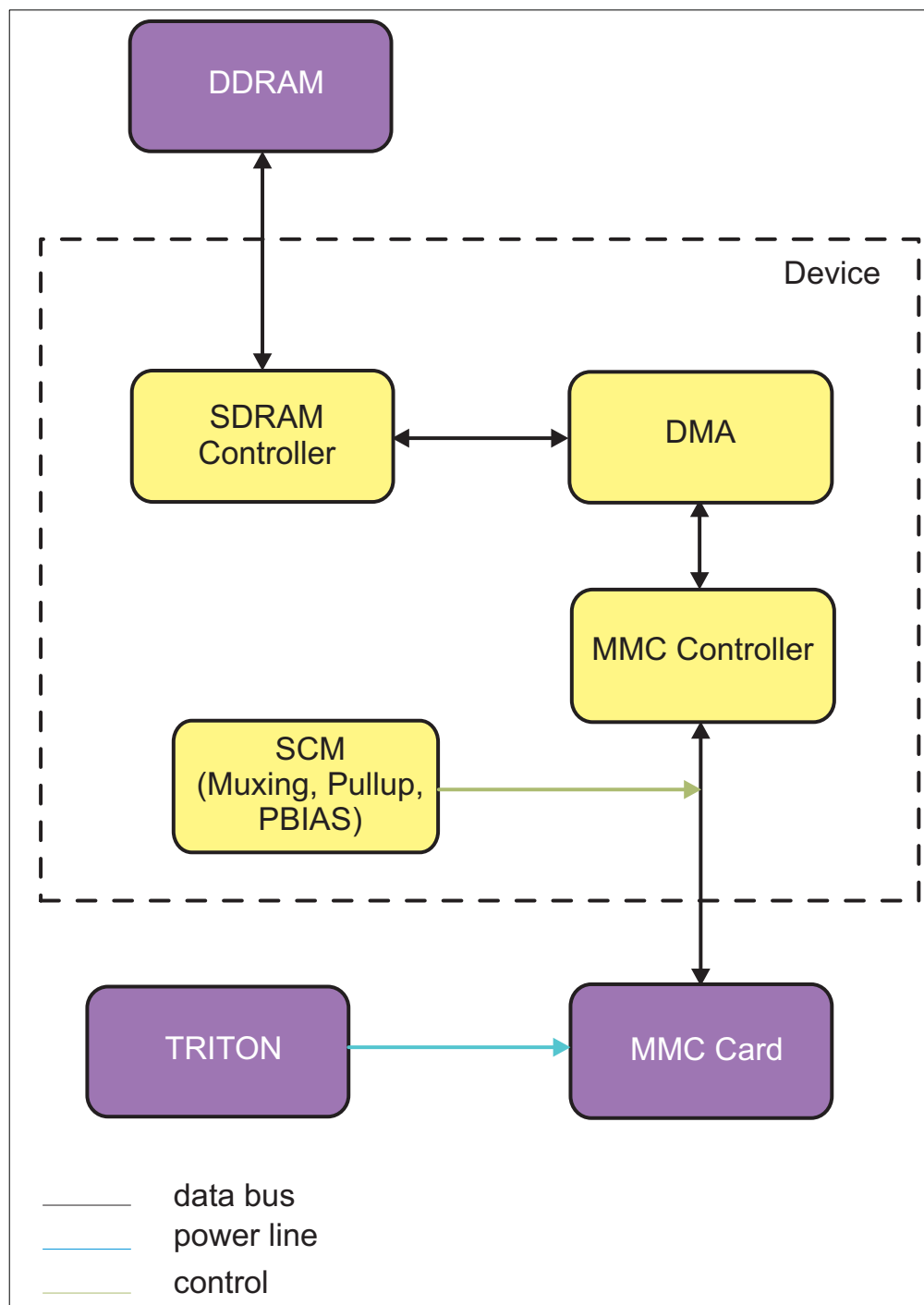
MMC card power supply is not provided by the MMCHS controller itself but rather by the companion device. please refer to I2C TRM output to learn how to set these voltage levels.

Only MMC1 controller is used in this configuration. Hence this document describes the output of the test carried out with MMC1 controller.

1.6.1.2 Environment

In order to operate in the correct manner, the MMCHS controller needs the System Control Module (SCM) to be configured with the right muxing mode and with the right pull up state. The MMC card needs to be powered with the right power level and this is done with the companion device. [Figure 1-39](#) gives the environment picture of the MMCHS controller.

Figure 1-39. Environment



mmchs_051

1.6.1.2.1 Command and Data Transfer Formats

When communicating with a MMC card, The MMCHS controller is always the master. The communication between the MMCHS controller and the MMC card always starts by sending a command. Both command and data transfers are started with a command.

The data transfer type used by the MMCHS controller is a finite multiple block transfer. [Figure 1-40](#) illustrates a command transfer without data. [Figure 1-41](#) and [Figure 1-42](#) illustrate a multiple block transfer between the MMCHS controller and the MMC card.

Figure 1-40. Command Transfer

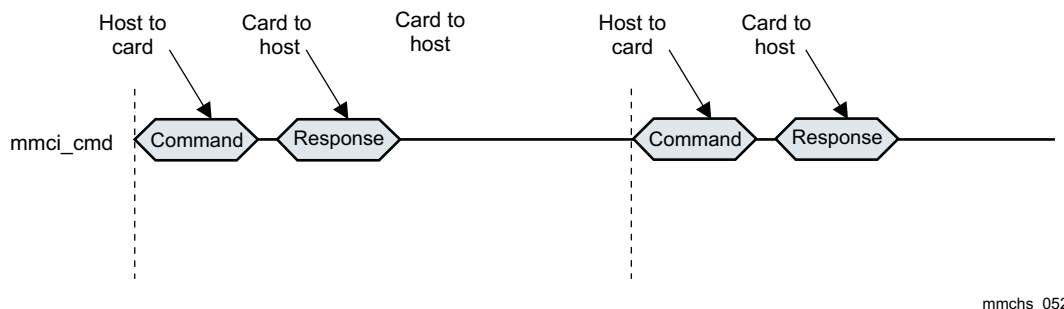


Figure 1-41. Data Read Transfer

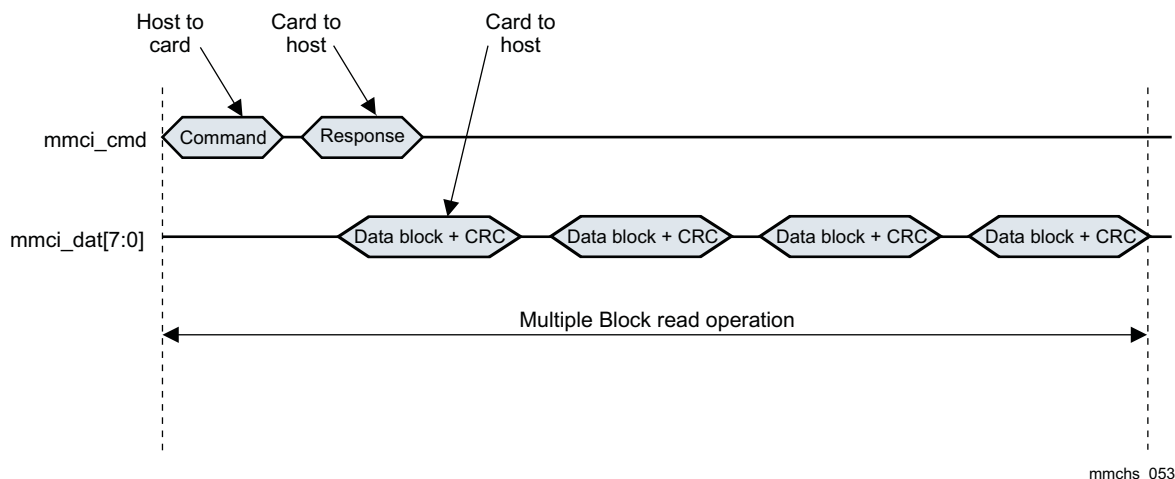
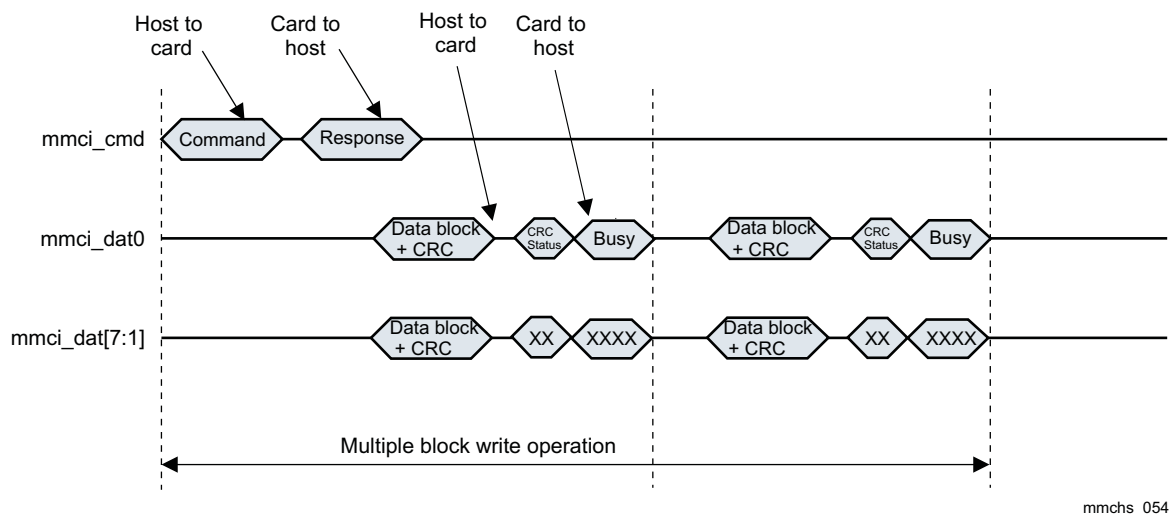


Figure 1-42. Data Write Transfer



1.6.1.3 Programming Flow

1.6.1.3.1 Initial Configuration

The initialization of the MMCHS controller is done through these steps :

1. MMCHS controller interface and functional clocks enabling.
2. MMCHS controller software reset.
3. MMCHS controller voltage capabilities initialization.
4. MMCHS controller default initialization.
5. MMCHS controller INIT procedure start.
6. MMCHS controller pre-card identification configuration.

For more information about different steps the MMCHS controller goes through during initial configuration refer to [Section 1.5](#), *MMC/SD/SDIO Basic Programming Model*.

1.6.1.3.1.1 MMCHS Controller Interface and Functional Clocks Enabling

To enable the interface and functional clocks of the MMCHS1 controller, the following steps must be done:

1. Enable the interface clock for the MMCHS1 controller (set the PRCM.CM_ICLKEN1_CORE[24] EN_MMCHS1).
2. Enable the functional clock for the MMCHS1 module (set the PRCM.CM_FCLKEN1_CORE[24] EN_MMCHS1).

[Table 1-8](#) shows all PRCM registers to be configured to enable interface and functional clocks for MMCHS1 controller.

Table 1-8. Register Print for the MMCHS1 controller's clocks Initialization

Register Name	Register Address	Value	Value Description
PRCM.CM_ICLKEN1_CORE	0x4800 4A10	0x01000000	MMCHS1 interface clock enabled
PRCM.CM_FCLKEN1_CORE	0x4800 4A00	0x01000000	MMCHS1 functional clock enabled

1.6.1.3.1.2 MMCHS Controller Software Reset

In order to software reset the MMCHS1 controller, the following steps must be done:

1. Write 0x2 in MMCHS1.MMCHS_SYSCONFIG register.
2. Wait until MMCHS1.MMCHS_SYSSTATUS[0] RESETDONE turns 1.

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_SYSCONFIG	0x4809 C010	0x00000002	Activate software reset
MMCHS1.MMCHS_SYSSTATUS	0x4809 C014	0x00000001	Reset is over.

1.6.1.3.1.3 MMCHS Controller Voltage Capabilities Initialization

MMCHS1 controller's voltage capabilities should be set in MMCHS1.MMCHS_CAPA. Refer to [Table 1-9](#).

Table 1-9. MMCHS Controller Voltage Capabilities Initialization

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CAPA	0x4809 C140	current_value 0x06000000	Activate VS18 and VS30 in MMCHS_CAPA register.

1.6.1.3.1.4 MMCHS Controller Default Initialization

Before sending any command, the MMCHS controller is configured with default values :

1. Default voltage support is set to 1.8 v in MMCHS1.MMCHS_HCTL[11:9] SDVS.
2. MMC bus is set to open drain in MMCHS1.MMCHS_CON[0] OD.
3. MMC data bus width is set to 1 in MMCHS1.MMCHS_HCTL[1] DTW.
4. MMC Card's power is off.
5. MMC Card's clock is on in MMCHS1.MMCHS_SYSCTL[0] ICE and MMCHS1.MMCHS_SYSCTL[2] CEN.
6. MMCHS controller bus power up in MMCHS1.MMCHS_HCTL[8] SDBP.
7. MMC card's clock frequency is set to 150 KHz in MMCHS1.MMCHS_SYSCTL[15:6] CLKD.

Table 1-10 shows the values that should be written in the right register pool.

Table 1-10. MMC Controller Default Initialization Values

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_HCTL	0x4809C128	0x00000b00	data bus width = 1, voltage = 1.8v, MMC bus power is on (not card's power)
MMCHS1.MMCHS_SYSCTL	0x4809C12C	0x0000a007	card's clock enable and card's clock frequency divider.
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	Set MMC bus mode to open drain.

A small notice about MMCHS1.MMCHS_HCTL. Even if the value written in it is 0x00000b00, the value read from it is equal to 0x00000a00. This is due to the fact that the MMCHS controller is not in a card state mode which sets automatically MMCHS_HCTL[8] SDBP bit to 0.

1.6.1.3.1.5 MMCHS Controller INIT Procedure Start

Prior to issuing any command, the MMCHS controller has to execute a special INIT procedure. The MMCHS controller has to generate a clock during 1ms. During the INIT procedure, the MMCHS controller generates 80 clock periods. In order to keep the 1ms gap, the MMCHS controller should be configured to generate a clock whose frequency is smaller or equal to 80 KHz. If the MMCHS controller divider bitfield width doesn't allow to choose big values, the MMCHS controller driver should perform the INIT procedure twice or three times. Twice is generally enough.

The INIT procedure is executed by setting MMCHS1.MMCHS_CON[1] INIT bitfield to 1 and by sending a dummy command, writing 0x00000000 in MMCHS1.MMCHS_CMD register.

Table 1-11 shows the values that should be written in the right register pool.

Table 1-11. MMCHS Controller INIT Procedure Start

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	current_value 0x00000002	sets MMCHS1.MMCHS_CON[1] INIT to 1
MMCHS1.MMCHS_CMD	0x4809C10C	0x00000000	sends dummy command.

1.6.1.3.1.6 MMCHS Controller Pre-card Identification Configuration

Before card identification starts, the MMCHS controller's configuration should change. MMC card's clock should now be 400 KHz according to MMC system spec requirements. Table 1-12 shows the values that should be written in the right register pool.

Table 1-12. MMCHS Controller Pre-Card Identification Configuration

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_HCTL	0x4809C128	0x00000b00	data bus width = 1, voltage = 1.8v, MMC bus power is on (not card's power)
MMCHS1.MMCHS_SYSCTL	0x4809C12C	0x00003C07	card's clock enable and card's clock frequency divider.
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	Set MMC bus mode to open drain.

1.6.1.3.2 MMC Card Identification

MMC card identification is performed by issuing many MMC commands. Each command imposes certain configuration values in a pool of registers. The status of command transfer is read in MMCHS1.MMCHS_STAT register and command response if any is read from MMCHS1.MMCHS_RSP10, MMCHS1.MMCHS_RSP32, MMCHS1.MMCHS_RSP54 and MMCHS1.MMCHS_RSP76.

This TRM output describes a use case where interrupts are used to signal MMCHS controller status changes.

For more details about the card identification sequence, please refer to [Section 1.5 MMC/SD/SDIO Basic Programming Model](#).

1.6.1.3.2.1 Sending CMD0

This command resets the MMC card (see [Table 1-13](#)).

Table 1-13. Sending CMD0

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x00040001	Enables CC and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x00040001	Enables CC and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x00000000	Sends CMD0 whose opcode is 0.

1.6.1.3.2.2 Sending CMD5

This command asks a SDIO card to send its operating conditions (see [Table 1-14](#)). This command will fail if there is no SDIO card. In case of success the response will be in MMCHS1.MMCHS_RSP10 register.

Table 1-14. Sending CMD5

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x00050001	Enables CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x00050001	Enables CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x05020000	Sends CMD5 whose opcode is 5 and response type is 48 bits.

1.6.1.3.2.2.1 Sending CMD8

This command asks a SD card version 2.X to send its operating conditions (see [Table 1-15](#)). This command will fail if there is no SD card version 2.X. In case of success the response will be in MMCHS1.MMCHS_RSP10 register.

Table 1-15. Sending CMD8

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x81a0000	Sends CMD8 whose opcode is 8, response type is 48 bits with CICE and CCCE enabled.

1.6.1.3.2.3 Sending CMD55

This is a special command used to prevent the card that the following command is going to be an application one (see [Table 1-16](#)). This is used to prepare the issuing of ACMD41 (opcode = 41) that usually asks a SD card version 1.X to send its operating conditions. If no SD card version 1.X is connected to the MMCHS controller this command will fail. In case of success, the response will be received in MMCHS1.MMCHS_RSP10 register.

Table 1-16. Sending CMD55

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x371a0000	Sends CMD55 whose opcode is 55, response type is 48 bits with CICE and CCCE enabled.

1.6.1.3.2.4 Sending CMD1

This command asks a MMC card to send its operating conditions (see [Table 1-17](#)). The response is received in MMCHS1.MMCHS_RSP10 register.

Table 1-17. Sending CMD1

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x00050001	Enables CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x00050001	Enables CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x01020000	Sends CMD1 whose opcode is 1 and response type is 48 bits.

Once the card response is available in register MMCHS1.MMCHS_RSP10, the software is responsible to compare Card OCR and Host OCR, and then send a second CMD1 command with the cross-checked OCR. This way, the card is notified of the Operating Voltage to work with.

1.6.1.3.2.5 Sending CMD2

This command asks the MMC card to send its CID register's content (see [Table 1-18](#)). The response is 128 bit wide and is received in MMCHS1.MMCHS_RSP10, MMCHS1.MMCHS_RSP32, MMCHS1.MMCHS_RSP54 and MMCHS1.MMCHS_RSP76 registers.

Table 1-18. Sending CMD2

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x00070001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x00070001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x02090000	Sends CMD2 whose opcode is 2, response type is 136 bits with CCCE enabled.

1.6.1.3.2.6 Sending CMD3

This command sets MMC card address (see [Table 1-19](#)). Useful when MMCHS controller switches to addressed mode.

Table 1-19. Sending CMD3

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000001	MMC bus is still in open drain state for broadcast.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x031a0000	Sends CMD3 whose opcode is 3, response type is 48 bits with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00010000	MMCHS_ARG register carries MMC card's address. We choose to assign address 1 any other 16 bit wide value is valid.

1.6.1.3.3 MMC Bus Setting Change After Card Identification

After CMD3 command transfer is completed successfully, an auto-negotiation on voltage value an start. This is the frontier when the MMCHS controller should switch from identification mode to transfer mode. This impacts the controller in a way that it should change its bus state from open drain to push-pull. [Table 1-20](#) gives several registers and their values.

Table 1-20. MMC Bus Setting Change Table

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000000	Bus is now in push-pull mode.
MMCHS1.MMCHS_HCTL	0x4809C128	0x00000B00	Bus power is on, 1.8V is selected.
MMCHS1.MMCHS_SYSCTL	0x4809C12C	0x00003C07	MMCHS controller's internal clock is stable and enabled, MMC card's clock is on. Divider value is 240 which means that MMCHS controller is still supplying a 400 KHz clock.

1.6.1.3.4 Reading the CSD Register of a MMC Card

After settling on a voltage value, additional information must be read from the MMC card. This data is stored in MMC card CSD register. The card sends CSD register content after receiving CMD9 command. The CSD register holds important information on the card, MMC system specification version support, maximum clock speed support, memory capacity, minimum block length, read and write transfer latency timings.

1.6.1.3.4.1 Sending CMD9

This command asks the card to send its csd register's content (see [Table 1-21](#)). The 136 bit (128 bits are valid payload) response is received in MMCHS1.MMCHS_RSP10, MMCHS1.MMCHS_RSP32, MMCHS1.MMCHS_RSP54 and MMCHS1.MMCHS_RSP76 registers. CMD9 is an addressed command which means that card's address must be written in MMCHS1.MMCHS_ARG register before the command is issued.

Table 1-21. Sending CMD9

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000000	MMC bus is in push-pull mode.
MMCHS1.MMCHS_IE	0x4809C134	0x00070001	Enables CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x00070001	Enables CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x09090000	Sends CMD9 whose opcode is 9, response type is 136 bits with CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00010000	MMCHS_ARG register carries MMC card's address. We choose to assign address 1 any other 16 bit wide value is valid.

After receiving and parsing CMD9 response, MMC card clock speed must change to take advantage to card's maximum speed. This has an impact on MMC bus as we should perform a clock frequency change. At this stage the maximum clock frequency will be 20 MHz (please refer to MMC system specification from www.mmca.org).

Clock frequency change procedure is performed in several steps. Please refer to [Section 1.5](#), MMC/SD/SDIO Basic Programming Model, [Section 1.5.2.7.2](#), MMCHS Clock Frequency Change.

[Table 1-22](#) shows the value written in MMCHS1.MMCHS_SYSCTL register.

Table 1-22. MMCHS_SYSCTL Value

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_SYSCTL	0x4809C12C	0x00000147	MMCHS controller's internal clock is stable and enabled, MMC card's clock is on. Divider value is 5 which means that MMCHS controller is supplying a 19.2 MHz clock < 20 MHz.

Another important parameter read from CSD register is MMC system specification version. If this parameter points to a value greater than or equal to 4, the MMC card is capable of a speed up to 52 MHz and a bus width up to 8 (1, 4 or 8 are the possible options). In order to enable these two extra features, MMCHS controller must issue a CMD6 command with specific argument.

A CMD6 command is issued in the data transfer mode after MMC card is selected. MMC card selection consists of sending CMD7 command with MMC card's address in command argument.

Table 1-23 shows the set of register impacted by CMD7 issue action.

Table 1-23. Sending CMD7

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000000	MMC bus is in push-pull mode.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x071a0000	Sends CMD7 whose opcode is 7, response type is 48 bits with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00010000	MMCHS_ARG register carries MMC card's address. We choose to assign address 1 any other 16 bit wide value is valid.

After a CMD7 transfer is complete, the MMC card is ready to receive a CMD6 command. CMD6 command is used to write a byte in MMC card extended CSD register (ext_csd). It is an IO access function. There are two write actions, the first one enables a specific data bus width in the card. For our use case we used maximum data bus width 8. The second one enables high speed feature in the card.

1.6.1.3.4.1.1 Setting Data Bus Width to 8

Table 1-24 shows the set of registers impacted by issuing CMD6.

Table 1-24. Setting Data Bus Width with CMD6

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000000	MMC bus is in push-pull mode.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x061b0000	Sends CMD6 whose opcode is 6, response type is 48 bits with busy, with CICE and CCCE enabled.

Table 1-24. Setting Data Bus Width with CMD6 (continued)

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_ARG	0x4809C108	0x03b70200	(3 << 24) (byte_address << 16) (byte_value << 8). byte_address is the byte address in ext_csd register.

After issuing CMD6 completes successfully and MMC card leaves busy state, MMCHS controller should change its data bus width. This is done by changing MMCHS1.MMCHS_CON configuration value.

Table 1-25. MMCHS_CON Value

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMCHS controller's data bus width is set to 8.

1.6.1.3.4.1.2 Enable High Speed Feature

Table 1-26 shows the set of registers impacted by issuing CMD6 issuing.

Table 1-26. Enabling High Speed with CMD6

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMC bus is in push-pull mode. DW8 is enabled.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x061b0000	Sends CMD6 whose opcode is 6, response type is 48 bits with busy, with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x03b90100	(3 << 24) (byte_address << 16) (byte_value << 8). byte_address is the byte address in ext_csd register.

After issuing CMD6 completes successfully and MMC card leaves busy state, MMCHS controller should now change its output clock to bring it to 48 MHz. 52 MHz, max frequency value supported by MMC card version 4 and above, is not supported because 96 MHz, MMCHS controller functional clock, is not a multiple of 52 MHz. We fall off to 48 MHz.

Table 1-27. MMCHS_SYSCTL Value

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_SYSCTL	0x4809C12C	0x00000087	MMCHS controller's internal clock is stable and enabled, MMC card's clock is on. Divider value is 2 which means that MMCHS controller is supplying a 48 MHz clock.

1.6.1.3.5 MMC Write Transfer

Either data read or data write transfer uses DMA controller to perform memory (DDRAM) to/from MMCHS controller transfers. The DMA part is not described in this chapter; it is described in the *DMA* chapter.

Before any data transfer begins, the card must selected by issuing CMD7 command (Table 1-23).

A write transfer is a finite multiple block write transfer. In order to perform a write transfer, the following steps must be performed.

1.6.1.3.5.1 Send CMD16

Issuing CMD16 allows to set the block length. For our use case we decided to use a static block length of 512 bytes. The block length value is passed to the MMC card via MMCHS1.MMCHS_ARG register. The registers impacted by this operation are as follows:

Table 1-28. Setting Block Length

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMC bus is in push-pull mode. DW8 is enabled.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x101a0000	Sends CMD16 whose opcode is 16, response type is 48 bits, with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00000200	Block length is 512 = 0x200

1.6.1.3.5.2 Send CMD23

Issuing CMD23 allows to set the number of how many 512-byte blocks the MMC card should expect from the MMCHS controller. The number of blocks is passed to MMC card via MMCHS1.MMCHS_ARG register. The registers impacted by this operation are as follows:

Table 1-29. Setting Number of Blocks

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMC bus is in push-pull mode. DW8 is enabled.
MMCHS1.MMCHS_IE	0x4809C134	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x100f0001	Enables CERR, CIE, CCRC, CC, CTO and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x171a0000	Sends CMD23 whose opcode is 23, response type is 48 bits, with CICE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00000008	Number of 512-byte blocks in 4 KB buffer is 8.

1.6.1.3.5.3 Send CMD25

Issuing CMD25 starts the finite, multiple block write transfer. Before the transfer starts, DMA controller should be configured for this operation. For more details about DMA configuration please refer to the DMA chapter.

Table 1-30. CMD25 Issuing

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMC bus is in push-pull mode. DW8 is enabled.

Table 1-30. CMD25 Issuing (continued)

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_IE	0x4809C134	0x107f0013	Enables CERR, CIE, CCRC, CC, TC, BWR, CTO, DTO, DCRC, DEB and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x107f0013	Enables CERR, CIE, CCRC, CC, TC, BWR, CTO, DTO, DCRC, DEB and CEB interrupts to rise.
MMCHS1.MMCHS_CMD	0x4809C10C	0x193a0023	Sends CMD25 whose opcode is 25, response type is 48 bits, with CICE, DP, MSBS, BCE, DE and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00000000	Not used.
MMCHS1.MMCHS_BLK	0x4809C104	0x00080200	(number_blocks << 16) (block_length)

1.6.1.3.6 MMC Read Transfer

Either data read or data write transfer uses DMA controller to perform memory (DDRAM) to/from MMCHS controller transfers. The DMA part is not described in this document; it is described in the *DMA* chapter.

Before any data transfer begins, the card must selected by issuing CMD7 command ([Table 1-23](#)).

A read transfer is a finite multiple block write transfer. In order to perform a read transfer, the following steps must be performed.

1.6.1.3.6.1 Send CMD16

Issuing CMD16 allows to set the block length. For our use case we decided to use a static block length of 512 bytes. The block length value is passed to the MMC card via MMCHS1.MMCHS_ARG register. See [Table 1-28](#).

1.6.1.3.6.2 Send CMD23

Issuing CMD23 allows to set the number of how many 512-byte blocks the MMC card should expect from the MMCHS controller. The number of blocks is passed to MMC card via MMCHS1.MMCHS_ARG register. See [Table 1-29](#).

1.6.1.3.6.3 Send CMD18

Issuing CMD18 starts the finite, multiple block read transfer. Before the transfer starts, DMA controller should be configured for this operation. For more details about DMA configuration please refer to the *DMA* chapter.

Table 1-31. CMD18 Issuing

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CON	0x4809C02C	0x00000020	MMC bus is in push-pull mode. DW8 is enabled.
MMCHS1.MMCHS_IE	0x4809C134	0x107f0023	Enables CERR, CIE, CCRC, CC, TC, BRR, CTO, DTO, DCRC, DEB and CEB events to occur.
MMCHS1.MMCHS_ISE	0x4809C138	0x107f0023	Enables CERR, CIE, CCRC, CC, TC, BRR, CTO, DTO, DCRC, DEB and CEB interrupts to rise.

Table 1-31. CMD18 Issuing (continued)

Register Name	Register Address	Value	Value Description
MMCHS1.MMCHS_CMD	0x4809C10C	0x123a0033	Sends CMD18 whose opcode is 18, response type is 48 bits, with CICE, DP, MSBS, BCE, DE, DDIR and CCCE enabled.
MMCHS1.MMCHS_ARG	0x4809C108	0x00000000	Not used.
MMCHS1.MMCHS_BLK	0x4809C104	0x00080200	(number_blocks << 16) (block_length)

1.7 MMC/SD/SDIO Registers

Table 1-32 lists the instance summary.

Table 1-32. Instance Summary

Module Name	Base Address	Size
MMCHS1	0x4809 C000	512 bytes
MMCHS2	0x480B 4000	512 bytes
MMCHS3	0x480A D000	512 bytes

1.7.1 MMC/SD/SDIO Register Mapping Summary

Table 1-33 to Table 1-35 list the MMC/SD/SDIO1 to MMC/SD/SDIO3 registers. Table 1-36 through Table 1-82 describe the register bits.

CAUTION

The MMC/SD/SDIO/ registers are limited to 32-bit data accesses. 16-bit and 8-bit are not allowed and can corrupt register content.

Table 1-33. MMC/SD/SDIO1 Register Offset Addresses

Register	Type	Register Width (Bits)	Offset Address	Physical Address
MMCHS_SYSCONFIG	RW	32	0x10	0x4809 C010
MMCHS_SYSSTATUS	R	32	0x14	0x4809 C014
MMCHS_CSRE	RW	32	0x24	0x4809 C024
MMCHS_SYSTEST	RW	32	0x28	0x4809 C028
MMCHS_CON	RW	32	0x2C	0x4809 C02C
MMCHS_PWCNT	RW	32	0x30	0x4809 C030
MMCHS_BLK	RW	32	0x104	0x4809 C104
MMCHS_ARG	RW	32	0x108	0x4809 C108
MMCHS_CMD	RW	32	0x10C	0x4809 C10C
MMCHS_RSP10	R	32	0x110	0x4809 C110
MMCHS_RSP32	R	32	0x114	0x4809 C114
MMCHS_RSP54	R	32	0x118	0x4809 C118
MMCHS_RSP76	R	32	0x11C	0x4809 C11C
MMCHS_DATA	RW	32	0x120	0x4809 C120
MMCHS_PSTATE	R	32	0x124	0x4809 C124
MMCHS_HCTL	RW	32	0x128	0x4809 C128
MMCHS_SYSCTL	RW	32	0x12C	0x4809 C12C
MMCHS_STAT	RW	32	0x130	0x4809 C130
MMCHS_IE	RW	32	0x134	0x4809 C134
MMCHS_ISE	RW	32	0x138	0x4809 C138
MMCHS_AC12	R	32	0x13C	0x4809 C13C
MMCHS_CAPA	RW	32	0x140	0x4809 C140
MMCHS_CUR_CAPA	RW	32	0x148	0x4809 C148
MMCHS_REV	R	32	0x1FC	0x4809 C1FC

Table 1-34. MMC/SD/SDIO2 Register Offset Addresses

Register	Type	Register Width (Bits)	Offset Address	Physical Address
MMCHS_SYSCONFIG	RW	32	0x10	0x480B 4010
MMCHS_SYSSTATUS	R	32	0x14	0x480B 4014
MMCHS_CSRE	RW	32	0x24	0x480B 4024
MMCHS_SYSTEST	RW	32	0x28	0x480B 4028
MMCHS_CON	RW	32	0x2C	0x480B 402C
MMCHS_PWCNT	RW	32	0x30	0x480B 4030
MMCHS_BLK	RW	32	0x104	0x480B 4104
MMCHS_ARG	RW	32	0x108	0x480B 4108
MMCHS_CMD	RW	32	0x10C	0x480B 410C
MMCHS_RSP10	R	32	0x110	0x480B 4110
MMCHS_RSP32	R	32	0x114	0x480B 4114
MMCHS_RSP54	R	32	0x118	0x480B 4118
MMCHS_RSP76	R	32	0x11C	0x480B 411C
MMCHS_DATA	RW	32	0x120	0x480B 4120
MMCHS_PSTATE	R	32	0x124	0x480B 4124
MMCHS_HCTL	RW	32	0x128	0x480B 4128
MMCHS_SYSCTL	RW	32	0x12C	0x480B 412C
MMCHS_STAT	RW	32	0x130	0x480B 4130
MMCHS_IE	RW	32	0x134	0x480B 4134
MMCHS_ISE	RW	32	0x138	0x480B 4138
MMCHS_AC12	R	32	0x13C	0x480B 413C
MMCHS_CAPA	RW	32	0x140	0x480B 4140
MMCHS_CUR_CAPA	RW	32	0x148	0x480B 4148
MMCHS_REV	R	32	0x1FC	0x480B 41FC

Table 1-35. MMC/SD/SDIO3 Register Offset Addresses

Register	Type	Register Width (Bits)	Offset Address	Physical Address
MMCHS_SYSCONFIG	RW	32	0x10	0x480A D010
MMCHS_SYSSTATUS	R	32	0x14	0x480A D014
MMCHS_CSRE	RW	32	0x24	0x480A D024
MMCHS_SYSTEST	RW	32	0x28	0x480A D028
MMCHS_CON	RW	32	0x2C	0x480A D02C
MMCHS_PWCNT	RW	32	0x30	0x480A D030
MMCHS_BLK	RW	32	0x104	0x480A D104
MMCHS_ARG	RW	32	0x108	0x480A D108
MMCHS_CMD	RW	32	0x10C	0x480A D10C
MMCHS_RSP10	R	32	0x110	0x480A D110
MMCHS_RSP32	R	32	0x114	0x480A D114
MMCHS_RSP54	R	32	0x118	0x480A D118
MMCHS_RSP76	R	32	0x11C	0x480A D11C
MMCHS_DATA	RW	32	0x120	0x480A D120
MMCHS_PSTATE	R	32	0x124	0x480A D124
MMCHS_HCTL	RW	32	0x128	0x480A D128
MMCHS_SYSCTL	RW	32	0x12C	0x480A D12C
MMCHS_STAT	RW	32	0x130	0x480A D130

Table 1-35. MMC/SD/SDIO3 Register Offset Addresses (continued)

Register	Type	Register Width (Bits)	Offset Address	Physical Address
MMCHS_IE	RW	32	0x134	0x480A D134
MMCHS_ISE	RW	32	0x138	0x480A D138
MMCHS_AC12	R	32	0x13C	0x480A D13C
MMCHS_CAPA	RW	32	0x140	0x480A D140
MMCHS_CUR_CAPA	RW	32	0x148	0x480A D148
MMCHS_REV	R	32	0x1FC	0x480A D1FC

1.7.2 Register Descriptions

1.7.2.1 MMCHS_SYSCONFIG

Table 1-36. MMCHS_SYSCONFIG

Address Offset		0x010		
Physical Address		0x4809 C010	Instance	MMCHS1
		0x480A D010		MMCHS3
		0x480B 4010		MMCHS2
Description		System Configuration Register This register allows controlling various parameters of the Interconnect interface.		
Type		RW		
Bits	Field Name	Description	Type	Reset
31:10	Reserved	These bits are initialized to zero, and writes to them are ignored. Reads return 0	R	0x00000
9:8	CLOCKACTIVITY	Clocks activity during wake up mode period. Bit8: Interface clock Bit9: Functional clock 0x0: Interface and Functional clock may be switched off. 0x1: Interface clock is maintained. Functional clock may be switched-off. 0x2: Functional clock is maintained. Interface clock may be switched-off. 0x3: Interface and Functional clocks are maintained.	RW	0x0
7:5	Reserved	These bits are initialized to zero, and writes to them are ignored. Reads return 0	R	0
4:3	SIDLEMODE	Power management 0x0: If an idle request is detected, the MMC/SD/SDIO host controller acknowledges it unconditionally and goes in Inactive mode. Interrupt and DMA requests are unconditionally deasserted. 0x1: If an idle request is detected, the request is ignored and the module keeps on behaving normally. 0x2: If an idle request is detected, the module will switch to wake up mode based on its internal activity, and the wake up capability can be used if the wake up capability is enabled (bit MMCHS_MMCHS_SYSCONFIG[2] ENAWAKEUP bit is set to 1). 0x3: Reserved - do not use	RW	0x2
2	ENAWAKEUP	Wakeup feature control 0x0: Wakeup capability is disabled 0x1: Wakeup capability is enabled	RW	1
1	SOFTRESET	Software reset. The bit is automatically reset by the hardware. During reset, it always returns 0.	RW	0

Table 1-39. Register Call Summary for Register MMCHS_SYSSTATUS

MMC/SD/SDIO Integration	
• Resets: [0] [1] [2]	
Use Cases and Tips	
• Programming Flow: [3] [4]	
MMC/SD/SDIO Registers	
• MMC/SD/SDIO Registers Mapping Summary: [5] [6] [7]	

1.7.2.3 MMCHS_CSRE

Table 1-40. MMCHS CSRE

Address Offset	0x024		
Physical Address	0x4809 C024	Instance	MMCHS1
	0x480A D024		MMCHS3
	0x480B 4024		MMCHS2
Description	Card status response error This register enables the host controller to detect card status errors of response type R1, R1b for all cards and of R5, R5b and R6 response for cards types SD or SDIO. When a bit MMCI.MMCHS_CSRE[I] is set to 1, if the corresponding bit at the same position in the response MMCI.MMCHS_RSP10[I] is set to 1, the host controller indicates a card error (MMCI.MMCHS_STAT[28] CERR bit) interrupt status to avoid the host driver reading the response register (MMCI.MMCHS_RSP10). Note: No automatic card error detection for autoCMD12 is implemented; the host system has to check autoCMD12 response register (MMCI.MMCHS_RSP76) for possible card errors.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSRE																															

Bits	Field Name	Description	Type	Reset
31:0	CSRE	Card status response error	RW	0x00000000

Table 1-41. Register Call Summary for Register MMCHS CSRE

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[0\] \[1\] \[2\] \[3\] \[4\]](#)

1.7.2.4 MMCHS SYSTEST

Table 1-42. MMCHS_SYSTEST

Address Offset	0x028		
Physical Address	0x4809 C028	Instance	MMCHS1
	0x480A D028		MMCHS3
	0x480B 4028		MMCHS2
Description	System Test register This register is used to control the signals that connect to I/O pins when the module is configured in system test (SYSTEST) mode for boundary connectivity verification. Note: In SYSTEST mode, a write into MMCi.MMCHS_CMD register will not start a transfer. The buffer behaves as a stack accessible only by the local host (push and pop operations). In this mode, the Transfer Block Size (MMCi.MMCHS_BLK[10:0] BLEN bits) and the Blocks count for current transfer (MMCi.MMCHS_BLK[31:16] NBLK bits) are needed to generate a Buffer write ready interrupt (MMCi.MMCHS_STAT[4] BWR bit) or a Buffer read ready interrupt (MMCi.MMCHS_STAT[5] BRR bit) and DMA requests if enabled.		
Type	RW		

Table 1-43. Register Call Summary for Register MMCHS_SYSTEST

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[0\] \[1\] \[2\] \[3\] \[4\] \[5\] \[6\] \[7\] \[8\] \[9\] \[10\] \[11\] \[12\] \[13\] \[14\] \[15\] \[16\] \[17\] \[18\] \[19\] \[20\] \[21\] \[22\] \[23\] \[24\] \[25\] \[26\] \[27\] \[28\] \[29\] \[30\] \[31\] \[32\] \[33\] \[34\] \[35\] \[36\] \[37\] \[38\] \[39\] \[40\] \[41\] \[42\] \[43\] \[44\] \[45\] \[46\] \[47\] \[48\] \[49\] \[50\] \[51\] \[52\] \[53\] \[54\] \[55\] \[56\] \[57\] \[58\] \[59\] \[60\] \[61\] \[62\] \[63\] \[64\] \[65\] \[66\] \[67\] \[68\] \[69\] \[70\] \[71\] \[72\] \[73\] \[74\]](#)

1.7.2.5 MMCHS_CON

Table 1-44. MMCHS_CON

Address Offset	0x02C		
Physical Address	0x4809 C02C	Instance	MMCHS1
	0x480A D02C		MMCHS3
	0x480B 402C		MMCHS2
Description	Configuration register This register is used: - to select the functional mode for any card. - to send an initialization sequence to any card. - to enable the detection on the mmci_dat[1] signal of a card interrupt for SDIO cards only. And also to configure: - specific data and command transfers for MMC cards only. - the parameters related to the card detect and write protect input signals.		
Type	RW		

Bits	Field Name	Description	Type	Reset
31:17	Reserved	Reserved bit field. Do not write any value	R	0x00000
16	CLKEXTFREE	External clock free running. This register is used to maintain card clock out of transfer transaction to enable slave module (for example to generate a synchronous interrupt on mmci_dat[1]). The Clock will be maintain only if MMCi.MMCHS_SYSTCTL[2] CEN bit is set. 0x0: External card clock is cut off outside active transaction period. 0x1: External card clock is maintain even out of active transaction period only if MMCi.MMCHS_SYSTCTL[2] CEN bit is set.	RW	0
15	PADEN	Control Power for MMC Lines. This register is only useful when MMC PADs contain power saving mechanism to minimize its leakage power. It works as a GPIO that directly control the ACTIVE pin of PADs. Excepted for mmci_dat[1], the signal is also combine outside the module with the dedicated power control MMCi.MMCHS_CON[11] CTPL bit.	RW	0

Bits	Field Name	Description	Type	Reset
		0x0: ADPIDLE module pin is not forced, it is automatically generated by the MMC fsms.		
		0x1: ADPIDLE module pin is forced to active state.		
14	OBIE	Out-of-Band Interrupt Enable (MMC cards only). This bit enables the detection of Out-of-Band Interrupt on MMC_OBI input pin. The usage of the Out-of-Band signal (OBI) is optional and depends on the system integration. 0x0: Out-of-Band interrupt detection disabled. 0x1: Out-of-Band interrupt detection enabled.	RW	0
13	OBIP	Out-of-Band Interrupt Polarity (MMC cards only). This bit selects the active level of the out-of-band interrupt coming from MMC cards. The usage of the Out-of-Band signal (OBI) is optional and depends on the system integration. 0x0: active high level. 0x1: active low level.	RW	0
12	CEATA	CE-ATA control mode (MMC cards compliant with CE-ATA): By default, this bit is set to 0. It is use to indicate that next commands are considered as specific CE-ATA commands that potentially use 'command completion' features. 0x0: Standard MMC/SD/SDIO mode. 0x1: CE-ATA mode. Next commands are considered as CE-ATA commands.	RW	0
11	CTPL	Control Power for mmci_dat[1] line (MMC and SD cards): By default, this bit is set to 0 and the host controller automatically disables all the input buffers outside of a transaction to minimize the leakage current. SDIO cards: When this bit is set to 1, the host controller automatically disables all the input buffers except the buffer of mmci_dat[1] outside of a transaction in order to detect asynchronous card interrupt on mmci_dat[1] line and minimize the leakage current of the buffers. 0x0: Disable all the input buffers outside of a transaction. 0x1: Disable all the input buffers except the buffer of mmci_dat[1] outside of a transaction.	RW	0
10:9	DVAL	Debounce filter value (All cards) This register is used to define a debounce period to filter the card detect input signal (SDCD). The usage of the card detect input signal (SDCD) is optional and depends on the system integration and the type of the connector housing that accommodates the card. 0x0: 33 us debounce period. 0x1: 231 us debounce period. 0x2: 1 ms debounce period. 0x3: 8.4 ms debounce period.	RW	0x3
8	WPP	Write protect polarity (SD and SDIO cards only) This bit selects the active level of the write protect input signal (SDWP). The usage of the write protect input signal (SDWP) is optional and depends on the system integration and the type of the connector housing that accommodates the card. 0x0: Active high level. 0x1: Active low level.	RW	0
7	CDP	Card detect polarity (All cards) This bit selects the active level of the card detect input signal (SDCD). The usage of the card detect input signal (SDCD) is optional and depends on the system integration and the type of the connector housing that accommodates the card. 0x0: Active high level. 0x1: Active low level.	RW	0

MMC/SD/SDIO Registers

Bits	Field Name	Description	Type	Reset
6	MIT	MMC interrupt command (Only for MMC cards.) This bit must be set to 1, when the next write access to the command register (MMCI.MMCHS_CMD) is for writing a MMC interrupt command (CMD40) requiring the command timeout detection to be disabled for the command response. 0x0: Command timeout enabled 0x1: Command timeout disabled	RW	0
5	DW8	8-bit mode MMC select For SD/SDIO cards, this bit must be set to 0. For MMC card, this bit must be set following a valid SWITCH command (CMD6) with the correct value and extend CSD index written in the argument. Prior to this command, the MMC card configuration register (CSD and EXT_CSD) must be verified for compliancy with MMC standard specification. 0x0: 1-bit or 4-bit Data width (mmci_dat[0] or mmci_dat[3:0] used, MMC, SD cards) 0x1: 8-bit Data width (mmci_dat[7:0] used, MMC cards)	RW	0
4	MODE	Mode select (All cards) These bits select the functional mode. 0x0: Functional mode. Transfers to the MMC/SD/SDIO cards follow the card protocol. MMC clock is enabled. MMC/SD transfers are operated under the control of the MMCI.MMCHS_CMD register. 0x1: SYSTEST mode. The signal pins are configured as general-purpose input/output and the 1024-byte buffer is configured as a stack memory accessible only by the local host or system DMA. The pins retain their default type (input, output or in-out). SYSTEST mode is operated under the control of the SYSTEST register.	RW	0
3	STR	Stream command (Only for MMC cards). This bit must be set to 1 only for the stream data transfers (read or write) of the adtc commands. Stream read is a class 1 command (CMD11: READ_DAT_UNTIL_STOP). Stream write is a class 3 command (CMD20: WRITE_DAT_UNTIL_STOP). 0x0: Block oriented data transfer. 0x1: Stream oriented data transfer.	RW	0
2	HR	Broadcast host response (Only for MMC cards). This register is used to force the host to generate a 48-bit response for bc command type. It can be used to terminate the interrupt mode by generating a CMD40 response by the core. In order to have the host response to be generated in open drain mode, the register MMCHS_CON[OD] must be set to 1. When MMCI.MMCHS_CON[12] CEATA bit is set to 1 and MMCI.MMCHS_ARG set to 0x00000000, when writing 0x00000000 into MMCI.MMCHS_CMD register, the host controller performs a 'command completion signal disable' token (i.e. mmci_cmd line held to '0' during 47 cycles followed by a 1). 0x0: The host does not generate a 48-bit response instead of a command. 0x1: The host generates a 48-bit response instead of a command or a command completion signal disable token.	RW	0

Bits	Field Name	Description	Type	Reset
1	INIT	Send initialization stream (All cards). When this bit is set to 1, and the card is idle, an initialization sequence is sent to the card. An initialization sequence consists of setting the mmci_cmd line to 1 during 80 clock cycles. The initialization sequence is mandatory - but it is not required to do it through this bit - this bit makes it easier. Clock divider (MMCi.MMCHS_SYSCCTL[15:6] CLKD bits) should be set to ensure that 80 clock periods are greater than 1ms. Note: in this mode, there is no command sent to the card and no response is expected. A command complete interrupt will be generated once the initialization sequence is completed. MMCi.MMCHS_STAT[0] CC bit can be polled. 0x0: The host does not send an initialization sequence. 0x1: The host sends an initialization sequence.	RW	0
0	OD	Card open drain mode (Only for MMC cards). This bit must be set to 1 for MMC card commands 1, 2, 3 and 40, and if the MMC card bus is operating in open-drain mode during the response phase to the command sent. Typically, during card identification mode when the card is either in idle, ready or ident state. It is also necessary to set this bit to 1, for a broadcast host response (see Broadcast host response register MMCi.MMCHS_CON[2] HR bit) 0x0: No Open Drain 0x1: Open Drain or Broadcast host response	RW	0

Table 1-45. Register Call Summary for Register MMCHS_CON

MMC/SD/SDIO Functional Description

- [MMC CE-ATA Command Completion Disable Management: \[0\] \[1\] \[2\]](#)

Use Cases and Tips

- [Programming Flow: \[3\] \[4\] \[5\] \[6\] \[7\] \[8\] \[9\] \[10\] \[11\] \[12\] \[13\] \[14\] \[15\] \[16\] \[17\] \[18\] \[19\] \[20\] \[21\] \[22\] \[23\] \[24\] \[25\] \[26\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[27\] \[28\] \[29\] \[30\] \[31\] \[32\] \[33\] \[34\] \[35\] \[36\] \[37\] \[38\] \[39\] \[40\] \[41\]](#)

1.7.2.6 MMCHS_PWCNT

Table 1-46. MMCHS_PWCNT

Address Offset	0x030	Instance	MMCHS1
Physical Address	0x4809 C030		MMCHS3
	0x480A D030		MMCHS2
	0x480B 4030		
Description	Power counter register This register is used to program a mmc counter to delay command transfers after activating the PAD power, this value depends on PAD characteristics and voltage.		
Type	RW		

Table 1-47. Register Call Summary for Register MMCHS_PWCNT

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[0\] \[1\] \[2\]](#)

1.7.2.7 MMCHS_BLK

Table 1-48. MMCHS_BLK

Address Offset	0x104																																																																																																		
Physical Address	0x4809 C104																Instance																MMCHS1																																																																		
	0x480A D104																																MMCHS3																																																																		
	0x480B 4104																																MMCHS2																																																																		
Description	Transfer Length Configuration register This register shall be used for any card.																																																																																																		
Type	RW																																																																																																		
<table><tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td colspan="16">NBLK</td><td colspan="6">Reserved</td><td colspan="12">BLEN</td></tr></table>																																		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	NBLK																Reserved						BLEN											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																				
NBLK																Reserved						BLEN																																																																													
Bits	Field Name		Description																										Type		Reset																																																																				
31:16	NBLK		Blocks count for current transfer. This register is enabled when Block count Enable (MMCi.MMCHS_CMD[1] BCE bit) is set to 1 and is valid only for multiple block transfers. Setting the block count to 0 results no data blocks being transferred. Note: The host controller decrements the block count after each block transfer and stops when the count reaches zero. This register can be accessed only if no transaction is executing (i.e., after a transaction has stopped). Read operations during transfers may return an invalid value and write operation will be ignored. In suspend context, the number of blocks yet to be transferred can be determined by reading this register. When restoring transfer context prior to issuing a Resume command, The local host shall restore the previously saved block count. <table><tr><td>0x0:</td><td>Stop count</td></tr><tr><td>0x1:</td><td>1 block</td></tr><tr><td>0x2:</td><td>2 blocks</td></tr><tr><td>0xFFFF:</td><td>65535 blocks</td></tr></table>																										0x0:	Stop count	0x1:	1 block	0x2:	2 blocks	0xFFFF:	65535 blocks	RW		0x0000																																																												
0x0:	Stop count																																																																																																		
0x1:	1 block																																																																																																		
0x2:	2 blocks																																																																																																		
0xFFFF:	65535 blocks																																																																																																		
15:11	Reserved		Reserved bit field. Do not write any value																										R		0x00																																																																				
10:0	BLEN		Transfer Block Size. This register specifies the block size for block data transfers. Read operations during transfers may return an invalid value, and write operations are ignored. When a CMD12 command is issued to stop the transfer, a read of the BLEN field after transfer completion (MMCi.MMCHS_STAT[1] TC bit set to 1) will not return the true byte number of data length while the stop occurs but the value written in this register before transfer is launched. <table><tr><td>0x0:</td><td>No data transfer</td></tr><tr><td>0x1:</td><td>1 byte block length</td></tr><tr><td>0x2:</td><td>2 bytes block length</td></tr><tr><td>0x3:</td><td>3 bytes block length</td></tr><tr><td>0x1FF:</td><td>511 bytes block length</td></tr><tr><td>0x200:</td><td>512 bytes block length</td></tr><tr><td>0x3FF:</td><td>1023 bytes block length</td></tr><tr><td>0x400:</td><td>1024 bytes block length</td></tr></table>																										0x0:	No data transfer	0x1:	1 byte block length	0x2:	2 bytes block length	0x3:	3 bytes block length	0x1FF:	511 bytes block length	0x200:	512 bytes block length	0x3FF:	1023 bytes block length	0x400:	1024 bytes block length	RW		0x000																																																				
0x0:	No data transfer																																																																																																		
0x1:	1 byte block length																																																																																																		
0x2:	2 bytes block length																																																																																																		
0x3:	3 bytes block length																																																																																																		
0x1FF:	511 bytes block length																																																																																																		
0x200:	512 bytes block length																																																																																																		
0x3FF:	1023 bytes block length																																																																																																		
0x400:	1024 bytes block length																																																																																																		

Table 1-49. Register Call Summary for Register MMCHS_BLK

- MMC/SD/SDIO Integration
- DMA Requests: [0] [1]

Table 1-49. Register Call Summary for Register MMCHS_BLK (continued)

MMC/SD/SDIO Functional Description	
• Data Buffer: [2]	
Use Cases and Tips	
• Programming Flow: [3] [4]	
MMC/SD/SDIO Registers	
• MMC/SD/SDIO Registers Mapping Summary: [5] [6] [7] [8] [9] [10] [11] [12] [13] [14]	

1.7.2.8 MMCHS ARG

Table 1-50. MMCHS ARG

Address Offset	0x108		
Physical Address	0x4809 C108	Instance	MMCHS1
	0x480A D108		MMCHS3
	0x480B 4108		MMCHS2
Description	Command argument Register This register contains command argument specified as bit 39-8 of Command-Format These registers must be initialized prior to sending the command itself to the card (write action into the register MMCi.MMCHS_CMD register). Only exception is for a command index specifying stuff bits in arguments, making a write unnecessary.		
Type	RW		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARG																															

Bits	Field Name	Description	Type	Reset
31:0	ARG	Command argument bits [31:0] ⁽¹⁾	RW	0x00000000

(1) For CMD52, ARG has to be programmed with IO_RW_DIRECT[39:8]. Refer to SDIO specification.

Table 1-51. Register Call Summary for Register MMCHS ARG

MMC/SD/SDIO Functional Description	
• MMC CE-ATA Command Completion Disable Management: [0]	
Use Cases and Tips	
• Programming Flow: [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [13] [14] [15] [16] [17]	
MMC/SD/SDIO Registers	
• MMC/SD/SDIO Registers Mapping Summary: [18] [19] [20] [21]	

1.7.2.9 MMCHS CMD

Table 1-52. MMCHS_CMD

Address Offset	0x10C		Instance	MMCHS1
Physical Address	0x4809 C10C			MMCHS3
	0x480A D10C			MMCHS2
	0x480B 410C			
Description	Command and transfer mode register MMCi.MMCHS_CMD[31:16] = the command register MMCi.MMCHS_CMD[15:0] = the transfer mode. This register configures the data and command transfers. A write into the most significant byte send the command. A write into MMCi.MMCHS_CMD[15:0] registers during data transfer has no effect. This register shall be used for any card.			
Type	RW			

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		INDX							CMD_TYPE	DP	CICE	CCCE	Reserved	RSP_TYPE	Reserved								MSBS		DDIR		Reserved		ACEN	BCE	DE

Bits	Field Name	Description	Type	Reset
31:30	Reserved	Reserved bit field. Do not write any value	R	0x0
29:24	INDX	Command index Binary encoded value from 0 to 63 specifying the command number send to card 0x0: CMD0 or ACMD0 0x1: CMD1 or ACMD1 0x3F: CMD63 or ACMD63	RW	0x00
23:22	CMD_TYPE	Command type. This register specifies three types of special command: Suspend, Resume and Abort. These bits shall be set to 0b00 for all other commands. 0x0: Others Commands 0x1: Upon CMD52 "Bus Suspend" operation 0x2: Upon CMD52 "Function Select" operation 0x3: Upon CMD12 or CMD52 "I/O Abort" command	RW	0x0
21	DP	Data present select. This register indicates that data is present and mmci_dat line shall be used. It must be set to 0 in the following conditions: - Command using only mmci_cmd line -Command with no data transfer but using busy signal on mmci_dat[0] -Resume command 0x0: Command with no data transfer 0x1: Command with data transfer	RW	0
20	CICE	Command Index check enable. This bit must be set to 1 to enable index check on command response to compare the index field in the response against the index of the command. If the index is not the same in the response as in the command, it is reported as a command index error (MMCi.MMCHS_STAT[19] CIE bit set to1) Note: The CICE bit cannot be configured for an Auto CMD12, then index check is automatically checked when this command is issued. 0x0: Index check disable 0x1: Index check enable	RW	0
19	CCCE	Command CRC check enable. This bit must be set to 1 to enable CRC7 check on command response to protect the response against transmission errors on the bus. If an error is detected, it is reported as a command CRC error (MMCi.MMCHS_STAT[17] CCRC bit set to 1). Note: The CCCE bit cannot be configured for an Auto CMD12, and then CRC check is automatically checked when this command is issued.	RW	0

Bits	Field Name	Description	Type	Reset
		0x0: CRC7 check disable 0x1: CRC7 check enable		
18	Reserved	Reserved bit field. Do not write any value.	R	0
17:16	RSP_TYPE	Response type. This bits defines the response type of the command. 0x0: No response 0x1: Response Length 136 bits 0x2: Response Length 48 bits 0x3: Response Length 48 bits with busy after response	RW	0x0
15:6	Reserved	Reserved bit field. Do not write any value.	R	0x000
5	MSBS	Multi/Single block select. This bit must be set to 1 for data transfer in case of multi block command. For any others command this bit shall be set to 0. 0x0: Single block. If this bit is 0, it is not necessary to set the register MMCi.MMCHS_BLK[31:16] NBLK bits. 0x1: Multi block. When Block Count is disabled (MMCi.MMCHS_CMD[1] BCE bit is set to 0) in Multiple block transfers (MMCi.MMCHS_CMD[5] MSBS bit is set to 1), the module can perform infinite transfer.	RW	0
4	DDIR	Data transfer Direction. Select This bit defines either data transfer will be a read or a write. 0x0: Data Write (host to card) 0x1: Data Read (card to host)	RW	0
3	Reserved	Reserved bit field. Do not write any value.	R	0
2	ACEN	Auto CMD12 Enable. (SD cards only). When this bit is set to 1, the host controller issues a CMD12 automatically after the transfer completion of the last block. The Host Driver shall not set this bit to issue commands that do not require CMD12 to stop data transfer. In particular, secure commands do not require CMD12. For CE-ATA commands (MMCi.MMCHS_CON[12] CEATA bit set to 1), auto CMD12 is useless; therefore when this bit is set the mechanism to detect command completion signal, named CCS, interrupt is activated. 0x0: Auto CMD12 disable 0x1: Auto CMD12 enable or CCS detection enabled.	RW	0
1	BCE	Block Count Enable (Multiple block transfers only). This bit is used to enable the block count register (MMCHS_BLK[31:16] NBLK bits). When Block Count is disabled (MMCHS_CMD[1] BCE bit is set to 0) in Multiple block transfers (MMCHS_CMD[5] MSBS bits is set to 1), the module can perform infinite transfer. 0x0: Block count disabled for infinite transfer. 0x1: Block count enabled for multiple block transfer with known number of blocks	RW	0
0	DE	DMA Enable. This bit is used to enable DMA mode for host data access. 0x0: DMA mode disable 0x1: DMA mode enable	RW	0

Table 1-53. Register Call Summary for Register MMCHS_CMD

MMC/SD/SDIO Environment

- [MMC/SD/SDIO Protocol and Data Format: \[0\] \[1\] \[2\] \[3\]](#)

MMC/SD/SDIO Integration

- [DMA Requests: \[4\]](#)

Table 1-53. Register Call Summary for Register MMCHS_CMD (continued)

MMC/SD/SDIO Functional Description
• Data Buffer: [5] • Transfer Stop: [6] • MMC CE-ATA Command Completion Disable Management: [7] [8]
Use Cases and Tips
• Programming Flow: [9] [10] [11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25]
MMC/SD/SDIO Registers
• MMC/SD/SDIO Registers Mapping Summary: [26] [27] [28] [29] [30] [31] [32] [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47] [48]

1.7.2.10 MMCHS_RSP10

Table 1-54. MMCHS_RSP10

Address Offset	0x110																															
Physical Address	0x4809 C110																Instance MMCHS1															
	0x480A D110																MMCHS3															
	0x480B 4110																MMCHS2															
Description	Command response[31:0] Register This 32-bit register holds bits positions [31:0] of command response type R1/R1b/R2/R3/R4/R5/R5b/R6																															
Type	R																															
313029282726252423222120191817161514131211109876543210																																
RSP1																RSP0																

Bits	Field Name	Description	Type	Reset
31:16	RSP1	R1/R1b (normal response) /R3/R4/R5/R5b/R6 : Command Response [39:24] R2: Command Response [31:16]	R	0x0000
15:0	RSP0	R1/R1b (normal response) /R3/R4/R5/R5b/R6 : Command Response [23:8] R2: Command Response [15:0]	R	0x0000

Table 1-55. Register Call Summary for Register MMCHS_RSP10

MMC/SD/SDIO Functional Description
• Different Types of Responses: [0] [1] [2]
Use Cases and Tips
• Programming Flow: [3] [4] [5] [6] [7] [8] [9]
MMC/SD/SDIO Registers
• MMC/SD/SDIO Registers Mapping Summary: [10] [11] [12] [13] [14]

1.7.2.11 MMCHS_RSP32

Table 1-56. MMCHS_RSP32

Address Offset	0x114																																		
Physical Address	0x4809 C114																Instance	MMCHS1																	
	0x480A D114																	MMCHS3																	
	0x480B 4114																	MMCHS2																	
Description	Command response[63:32] Register This 32-bit register holds bits positions [63:32] of command response type R2																																		
Type	R																																		
313029282726252423222120191817161514131211109876543210																																			
RSP3																	RSP2																		
Bits	Field Name		Description																												Type		Reset		
31:16		RSP3		R2: Command Response [63:48]																												R		0x0000	
15:0		RSP2		R2: Command Response [47:32]																												R		0x0000	

Table 1-57. Register Call Summary for Register MMCHS_RSP32

MMC/SD/SDIO Functional Description

- [Different Types of Responses: \[0\]](#)

Use Cases and Tips

- [Programming Flow: \[1\] \[2\] \[3\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[4\] \[5\] \[6\]](#)

1.7.2.12 MMCHS_RSP54

Table 1-58. MMCHS_RSP54

Address Offset	0x118																																																																																																
Physical Address	0x4809 C118																Instance	MMCHS1																																																																															
	0x480A D118																	MMCHS3																																																																															
	0x480B 4118																	MMCHS2																																																																															
Description	Command response[95:64] Register This 32-bit register holds bits positions [95:64] of command response type R2																																																																																																
Type	R																																																																																																
<table><tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td colspan="16">RSP5</td><td colspan="17">RSP4</td></tr></table>																																	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	RSP5																RSP4																
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																																		
RSP5																RSP4																																																																																	
Bits	Field Name		Description		Type		Reset																																																																																										
31:16	RSP5		R2: Command Response [95:80]		R		0x0000																																																																																										
15:0	RSP4		R2: Command Response [79:64]		R		0x0000																																																																																										

Table 1-59. Register Call Summary for Register MMCHS_RSP54

MMC/SD/SDIO Functional Description

- [Different Types of Responses: \[0\]](#)

Use Cases and Tips

- [Programming Flow: \[1\] \[2\] \[3\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[4\] \[5\] \[6\]](#)

1.7.2.13 MMCHS_RSP76

Table 1-60. MMCHS_RSP76

Address Offset	0x11C		
Physical Address	0x4809 C11C	Instance	MMCHS1
	0x480A D11C		MMCHS3
	0x480B 411C		MMCHS2
Description	Command response[127:96] Register This 32-bit register holds bits positions [127:96] of command response type R2		
Type	R		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSP7																RSP6															

Bits	Field Name	Description	Type	Reset
31:16	RSP7	R1b (Auto CMD12 response): Command Response [39:24] R2: Command Response [127:112]	R	0x0000
15:0	RSP6	R1b (Auto CMD12 response): Command Response [23:8] R2: Command Response [111:96]	R	0x0000

Table 1-61. Register Call Summary for Register MMCHS_RSP76

MMC/SD/SDIO Functional Description

- [Different Types of Responses: \[0\] \[1\] \[2\]](#)

Use Cases and Tips

- [Programming Flow: \[3\] \[4\] \[5\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[6\] \[7\] \[8\] \[9\] \[10\]](#)

1.7.2.14 MMCHS_DATA

Table 1-62. MMCHS_DATA

Address Offset	0x120																																																																		
Physical Address	0x4809 C120	Instance	MMCHS1																																																																
	0x480A D120		MMCHS3																																																																
	0x480B 4120		MMCHS2																																																																
Description	Data Register This register is the 32-bit entry point of the buffer for read or write data transfers. The buffer size is 32bits x256(1024 bytes). Bytes within a word are stored and read in little endian format. This buffer can be used as two 512 byte buffers to transfer data efficiently without reducing the throughput. Sequential and contiguous access is necessary to increment the pointer correctly. Random or skipped access is not allowed. In little endian, if the local host accesses this register byte-wise or 16bit-wise, the least significant byte (bits [7:0]) must always be written/read first. The update of the buffer address is done on the most significant byte write for full 32-bit DATA register or on the most significant byte of the last word of block transfer. Example 1: Byte or 16-bit access Mbyteen[3:0]=0001 (1-byte) => Mbyteen[3:0]=0010 (1-byte) => Mbyteen[3:0]=1100 (2-bytes) OK Mbyteen[3:0]=0001 (1-byte) => Mbyteen[3:0]=0010 (1-byte) => Mbyteen[3:0]=0100 (1-byte) OK Mbyteen[3:0]=0001 (1-byte) => Mbyteen[3:0]=0010 (1-byte) => Mbyteen[3:0]=1000 (1-byte) Bad																																																																		
Type	RW																																																																		
<table><tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td colspan="32">DATA</td></tr></table>				31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	DATA																															
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																				
DATA																																																																			
Bits	Field Name	Description	Type Reset																																																																
31:0	DATA	Data Register [31:0] In functional mode (MMCI.MMCHS_CON[4] MODE bit set to the default value 0): A read access to this register is allowed only when the buffer read enable status is set to 1 (MMCI.MMCHS_PSTATE[11] BRE bit), otherwise a bad access (MMCI.MMCHS_STAT[29] BADA bit) is signaled. A write access to this register is allowed only when the buffer write enable status is set to 1 (MMCI.MMCHS_PSTATE[10] BWE bit), otherwise a bad access (MMCI.MMCHS_STAT[29] BADA bit) is signaled and the data is not written.	RW 0x00000000																																																																

Table 1-63. Register Call Summary for Register MMCHS_DATA

MMC/SD/SDIO Functional Description

- [Data Buffer: \[0\] \[1\] \[2\] \[3\] \[4\] \[5\] \[6\] \[7\]](#)
- [Transfer or Command Status and Errors Reporting: \[8\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[9\] \[10\] \[11\] \[12\] \[13\] \[14\] \[15\]](#)

1.7.2.15 MMCHS_PSTATE

Table 1-64. MMCHS_PSTATE

Address Offset	0x124																																																																																												
Physical Address	0x4809 C124															Instance																																																																													
	0x480A D124															MMCHS1																																																																													
	0x480B 4124															MMCHS3																																																																													
																MMCHS2																																																																													
Description	Present state register. The Host can get status of the Host Controller from this 32-bit read only register.																																																																																												
Type	R																																																																																												
<table><tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td colspan="7">Reserved</td><td>CLEV</td><td colspan="4">DLEV</td><td>Reserved</td><td>Reserved</td><td>Reserved</td><td colspan="4">Reserved</td><td>BRE</td><td>BWE</td><td>RTA</td><td>WTA</td><td colspan="4">Reserved</td><td>DLA</td><td>DATI</td><td>CMDI</td></tr></table>																																31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reserved							CLEV	DLEV				Reserved	Reserved	Reserved	Reserved				BRE	BWE	RTA	WTA	Reserved				DLA	DATI	CMDI
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																														
Reserved							CLEV	DLEV				Reserved	Reserved	Reserved	Reserved				BRE	BWE	RTA	WTA	Reserved				DLA	DATI	CMDI																																																																
Bits	Field Name		Description														Type		Reset																																																																										
31:25	Reserved		Reserved bit field. Do not write any value.														R		0x00																																																																										
24	CLEV		mmci_cmd line signal level. This status is used to check the mmci_cmd line level to recover from errors, and for debugging. The value of this register after reset depends on the mmci_cmd line level at that time. Read 0x0: The mmci_cmd line level is 0. Read 0x1: The mmci_cmd line level is 1.														R		-																																																																										
23:20	DLEV		mmci_dat[3:0] line signal level mmci_dat[3] => bit 23 mmci_dat[2] => bit 22 mmci_dat[1] => bit 21 mmci_dat[0] => bit 20 This status is used to check mmci_dat line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from mmci_dat[0]. The value of these registers after reset depends on the mmci_dat lines level at that time.														R		0x-																																																																										
19	Reserved		Reserved bit field. Do not write any value														R		0																																																																										
18	Reserved		Reserved bit field. Do not write any value This bit is not affected by soft reset.														R		1																																																																										
17:16	Reserved		Reserved bit field. Do not write any value The value of these bits after soft reset is 0x0. These bits will be automatically set to 0x3 after debounce time. Debounce time is fixed to 256 x32kHz clock cycles.														R		00																																																																										
15:12	Reserved		Reserved bit field. Do not write any value														R		0x0																																																																										
11	BRE		Buffer read enable. This bit is used for non-DMA read transfers. It indicates that a complete block specified by MMCI.MMCHS_BLK[10:0] BLEN bits has been written in the buffer and is ready to be read. It is set to 0 when the entire block is read from the buffer. It is set to 1 when a block data is ready in the buffer and generates the Buffer read ready status of interrupt (MMCI.MMCHS_STAT[5] BRR bit). Read 0x0: Read BLEN bytes disable Read 0x1: Read BLEN bytes enable. Readable data exists in the buffer.														R		0																																																																										
10	BWE		Buffer Write enable. This status is used for non-DMA write transfers. It indicates if space is available for write data. Read 0x0: There is no room left in the buffer to write BLEN bytes of data. Read 0x1: There is enough space in the buffer to write BLEN bytes of data.														R		0																																																																										

Bits	Field Name	Description	Type	Reset
9	RTA	Read transfer active. This status is used for detecting completion of a read transfer. It is set to 1 after the end bit of read command or by activating a continue request (MMCi.MMCHS_HCTL[17] CR bit) following a stop at block gap request. This bit is set to 0 when all data have been read by the local host after last block or after a stop at block gap request. Read 0x0: No valid data on the mmci_dat lines. Read 0x1: Read data transfer on going.	R	0
8	WTA	Write transfer active. This status indicates a write transfer active. It is set to 1 after the end bit of write command or by activating a continue request (MMCi.MMCHS_HCTL[17] CR bit) following a stop at block gap request. This bit is set to 0 when CRC status has been received after last block or after a stop at block gap request. Read 0x0: No valid data on the mmci_dat lines. Read 0x1: Write data transfer on going.	R	0
7:3	Reserved	Reserved bit field. Do not write any value	R	0x00
2	DLA	mmci_dat line active. This status bit indicates whether one of the mmci_dat line is in use. In the case of read transactions (card to host): This bit is set to 1 after the end bit of read command or by activating continue request MMCi.MMCHS_HCTL[17] CR bit. This bit is set to 0 when the host controller received the end bit of the last data block or at the beginning of the read wait mode. In the case of write transactions (host to card): This bit is set to 1 after the end bit of write command or by activating continue request MMCi.MMCHS_HCTL[17] CR bit. This bit is set to 0 on the end of busy event for the last block; host controller must wait 8 clock cycles with line not busy to really consider not "busy state" or after the busy block as a result of a stop at gap request. Read 0x0: mmci_dat Line inactive Read 0x1: mmci_dat Line active	R	0
1	DATi	Command inhibit (mmci_dat). This status bit is generated if either mmci_dat line is active (MMCi.MMCHS_PSTATE[2] DLA bit) or Read transfer is active (MMCi.MMCHS_PSTATE[9] RTA bit) or when a command with busy is issued. This bit prevents the local host to issue a command. A change of this bit from 1 to 0 generates a transfer complete interrupt (MMCi.MMCHS_STAT[1] TC bit). Read 0x0: Issuing of command using the mmci_dat lines is allowed Read 0x1: Issuing of command using mmci_dat lines is not allowed	R	0
0	CMDi	Command inhibit(mmci_cmd). This status bit indicates that the mmci_cmd line is in use. This bit is set to 0 when the most significant byte is written into the command register. This bit is not set when Auto CMD12 is transmitted. This bit is set to 0 in either the following cases: - After the end bit of the command response, excepted if there is a command conflict error (MMCi.MMCHS_STAT[17] CCRC bit or MMCi.MMCHS_STAT[18] CEB bit set to 1) or a Auto CMD12 is not executed (MMCi.MMCHS_AC12[0] ACNE bit). - After the end bit of the command without response (MMCi.MMCHS_CMD[17:16] RSP_TYPE bits set to "00"). In case of a command data error is detected (MMCi.MMCHS_STAT[19] CTO bit set to 1), this register is not automatically cleared. Read 0x0: Issuing of command using mmci_cmd line is allowed Read 0x1: Issuing of command using mmci_cmd line is not allowed	R	0

Table 1-65. Register Call Summary for Register MMCHS_PSTATE

MMC/SD/SDIO Integration

- **Resets:** [0]

Table 1-65. Register Call Summary for Register MMCHS_PSTATE (continued)

MMC/SD/SDIO Functional Description

- [Data Buffer: \[1\] \[2\] \[3\] \[4\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[5\] \[6\] \[7\] \[8\] \[9\] \[10\] \[11\] \[12\] \[13\] \[14\] \[15\] \[16\] \[17\] \[18\] \[19\] \[20\] \[21\]](#)

1.7.2.16 MMCHS_HCTL

Table 1-66. MMCHS_HCTL

Address Offset	0x128			
Physical Address	0x4809 C128	Instance	MMCHS1	
	0x480A D128		MMCHS3	
	0x480B 4128		MMCHS2	
Description	Control register. This register defines the host controls to set power, wakeup and transfer parameters. MMCi.MMCHS_HCTL[31:24] = Wakeup control MMCi.MMCHS_HCTL[23:16] = Block gap control MMCi.MMCHS_HCTL[15:8] = Power control MMCi.MMCHS_HCTL[7:0] = Host control			
Type	RW			

Bits	Field Name	Description	Type	Reset
31:28	Reserved	Reserved bit field. Do not write any value.	R	0x00
27	OBWE	Wakeup event enable for 'Out-of-Band' Interrupt. This bit enables wakeup events for 'Out-of-Band' assertion. Wakeup is generated if the wakeup feature is enabled (MMCi.MMCHS_SYSCONFIG[2] ENAWAKEUP bit). The write to this register is ignored when MMCi.MMCHS_CON[14] OBIE bit is not set. 0x0: Disable wakeup on 'Out-of-Band' Interrupt. 0x1: Enable wakeup on 'Out-of-Band' Interrupt.	RW	0
26	REM	Wakeup event enable on SD card removal. This bit enables wakeup events for card removal assertion. Wakeup is generated if the wakeup feature is enabled (MMCi.MMCHS_SYSCONFIG[2] ENAWAKEUP bit). 0x0: Disable wakeup on card removal. 0x1: Enable wakeup on card removal.	RW	0
25	INS	Wakeup event enable on SD card insertion This bit enables wakeup events for card insertion assertion. Wakeup is generated if the wakeup feature is enabled (MMCi.MMCHS_SYSCONFIG[2] ENAWAKEUP bit). 0x0: Disable wakeup on card insertion. 0x1: Enable wakeup on card insertion.	RW	0
24	IWE	Wakeup event enable on SD card interrupt. This bit enables wakeup events for card interrupt assertion. Wakeup is generated if the wakeup feature is enabled (MMCi.MMCHS_SYSCONFIG[2] ENAWAKEUP bit) and enable status bit is set (MMCi.MMCHS_IE[8] CIRQ_ENABLE bit). 0x0: Disable wakeup on card interrupt 0x1: Enable wakeup on card interrupt	RW	0
23:20	Reserved	Reserved bit field. Do not write any value	R	0x0
19	IBG	Interrupt block at gap. This bit is valid only in 4-bit mode of SDIO card to enable interrupt detection in the interrupt cycle at block gap for a multiple block transfer. For MMC cards and for SD card this bit should be set to 0. 0x0: Disable interrupt detection at the block gap in 4-bit mode 0x1: Enable interrupt detection at the block gap in 4-bit mode	RW	0

Bits	Field Name	Description	Type	Reset
18	RWC	Read wait control. The read wait function is optional only for SDIO cards. If the card supports read wait, this bit must be enabled, then requesting a stop at block gap (MMCi.MMCHS_HCTL[16] SBGR bit) generates a read wait period after the current end of block. Be careful, if read wait is not supported it may cause a conflict on mmci_dat line. 0x0: Disable Read Wait Control. Suspend/Resume cannot be supported. 0x1: Enable Read Wait Control	RW	0
17	CR	Continue request. This bit is used to restart a transaction that was stopped by requesting a stop at block gap (MMCi.MMCHS_HCTL[16] SBGR bit). Set this bit to 1 restarts the transfer. The bit is automatically set to 0 by the host controller when transfer has restarted i.e. mmci_dat line is active (MMCi.MMCHS_PSTATE[2] DLA bit) or transferring data (MMCi.MMCHS_PSTATE[8] WTA bit). The Stop at block gap request must be disabled (MMCi.MMCHS_HCTL[16] SBGR bit =0) before setting this bit. 0x0: No affect 0x1: transfer restart	RW	0
16	SBGR	Stop at block gap request. This bit is used to stop executing a transaction at the next block gap. The transfer can restart with a continue request (MMCi.MMCHS_HCTL[17] CR bit) or during a suspend/resume sequence. In case of read transfer, the card must support read wait control. In case of write transfer, the host driver shall set this bit after all block data written. Until the transfer completion (MMCi.MMCHS_STAT[1] TC bit set to 1), the host driver shall leave this bit set to 1. If this bit is set, the local host shall not write to the data register (MMCi.MMCHS_DATA). 0x0: Transfer mode 0x1: Stop at block gap	RW	0
15:12	Reserved	Reserved bit field. Do not write any value.	R	0x0
11:9	SDVS	SD bus voltage select (All cards). The host driver should set these bits to select the voltage level for the card according to the voltage supported by the system (MMCi.MMCHS_CAPA[26] VS18 bit, MMCi.MMCHS_CAPA[25] VS30 bit, MMCi.MMCHS_CAPA[24] VS33 bit) before starting a transfer. 0x5: 1.8V (Typical) 0x6: 3.0V (Typical) 0x7: 3.3V (Typical) MMCHS2: This field must be set to 0x5. MMCHS3: This field must be set to 0x5.	RW	0x0
8	SDBP	SD bus power. Before setting this bit, the host driver shall select the SD bus voltage (MMCi.MMCHS_HCTL[11:9] SDVS bits). If the host controller detects the No card state, this bit is automatically set to 0. If the module is power off, a write in the command register (MMCi.MMCHS_CMD) will not start the transfer. A write to this bit has no effect if the selected SD bus voltage is not supported according to capability register (MMCi.MMCHS_CAPA[VS*]). 0x0: Power off 0x1: Power on	RW	0
7:2	Reserved	Reserved bit field. Do not write any value.	R	0x00

MMC/SD/SDIO Registers

Bits	Field Name	Description	Type	Reset
1	DTW	Data transfer width. For MMC card, this bit must be set following a valid SWITCH command (CMD6) with the correct value and extend CSD index written in the argument. Prior to this command, the MMC card configuration register (CSD and EXT_CSD) must be verified for compliance with <i>MMC standard specification 4.x</i> . This register has no effect when the MMC 8-bit mode is selected (MMCI.MMCHS_CON[5] DW8 bit set to 1). For SD/SDIO cards, this bit 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 supported bus width by the SD card. 0x0: 1-bit Data width (mmci_dat[0] used) 0x1: 4-bit Data width (mmci_dat[3:0] used)	RW	0
0	Reserved	Reserved bit field. Do not write any value.	R	0

Table 1-67. Register Call Summary for Register MMCHS_HCTL

MMC/SD/SDIO Integration
• Clocks: [0] [1]
MMC/SD/SDIO Functional Description
• Transfer Stop: [2] [3]
Use Cases and Tips
• Programming Flow: [4] [5] [6] [7] [8] [9] [10] [11]
MMC/SD/SDIO Registers
• MMC/SD/SDIO Registers Mapping Summary: [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] [29] [30]

1.7.2.17 MMCHS_SYSCTL

Table 1-68. MMCHS_SYSCTL

Address Offset	0x12C																																																																																													
Physical Address	0x4809 C12C															Instance	MMCHS1																																																																													
	0x480A D12C																MMCHS3																																																																													
	0x480B 412C																MMCHS2																																																																													
Description	SD system control register. This register defines the system controls to set software resets, clock frequency management and data timeout. MMCHS_SYSCTL [31:24] = Software resets MMCHS_SYSCTL [23:16] = Timeout control MMCHS_SYSCTL [15:0] = Clock control																																																																																													
Type	RW																																																																																													
<table><tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td colspan="5">Reserved</td><td>SRD</td><td>SRC</td><td>SRA</td><td colspan="4">Reserved</td><td colspan="4">DTO</td><td colspan="8">CLKD</td><td colspan="4">Reserved</td><td>CEN</td><td>ICS</td><td>ICE</td></tr></table>																																31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reserved					SRD	SRC	SRA	Reserved				DTO				CLKD								Reserved				CEN	ICS	ICE
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																															
Reserved					SRD	SRC	SRA	Reserved				DTO				CLKD								Reserved				CEN	ICS	ICE																																																																
Bits	Field Name	Description																			Type	Reset																																																																								
31:27	Reserved	Reserved bit field. Do not write any value.																			R	0x00																																																																								
26	SRD	Software reset for mmci_dat line. This bit is set to 1 for reset and released to 0 when completed .mmci_dat finite state machine in both clock domain are also reset. Here below are the registers cleared by the MMCHS_SYSCTL [26] SRD bit: MMCi.MMCHS_DATA MMCi.MMCHS_PSTATE : BRE, BWE, RTA, WTA, DLA and DATI MMCi.MMCHS_HCTL : SBGR and CR MMCi.MMCHS_STAT : BRR, BWR, BGE and TC Interconnect and MMC buffer data management is reinitialized.																			RW	0																																																																								

Bits	Field Name	Description	Type	Reset
		0x0: Reset completed 0x1: Software reset for mmci_dat line		
25	SRC	Software reset for mmci_cmd line. This bit is set to 1 for reset and released to 0 when completed. mmci_cmd finite state machine in both clock domain are also reset. Here below the registers cleared by the MMCI.MMCHS_SYSCCTL[25] SRC bit: MMCI.MMCHS_PSTATE: CMDI MMCI.MMCHS_STAT: CC Interconnect and MMC command status management is reinitialized. 0x0: Reset completed 0x1: Software reset for mmci_cmd line	RW	0
24	SRA	Software reset for all. This bit is set to 1 for reset , and released to 0 when completed. This reset affects the entire host controller except for the card detection circuit and capabilities registers. 0x0: Reset completed 0x1: Software reset for all the design	RW	0
23:20	Reserved	Reserved bit field. Do not write any value.	R	0x0
19:16	DTO	Data timeout counter value and busy timeout. This value determines the interval by which mmci_dat lines timeouts are detected. The host driver needs to set this bit field based on - the maximum read access time (NAC) (Refer to the SD Specification Part1 Physical Layer), - the data read access time values (TAAC and NSAC) in the card specific data register (CSD) of the card, - the timeout clock base frequency (MMCI.MMCHS_CAPA[5:0] TCF bits). If the card does not respond within the specified number of cycles, a data timeout error occurs (MMCI.MMCHS_STAT[20] DTO bit). The MMCI.MMCHS_SYSCCTL[19,16] DTO bitfield is also used to check busy duration, to generate busy timeout for commands with busy response or for busy programming during a write command. Timeout on CRC status is generated if no CRC token is present after a block write. 0x0: TCF x 2 ¹³ 0x1: TCF x 2 ¹⁴ 0xE: TCF x 2 ²⁷ 0xF: Reserved	RW	0x0
15:6	CLKD	Clock frequency select These bits define the ratio between a reference clock frequency (system dependant) and the output clock frequency on the mmci_clk pin of either the memory card (MMC, SD or SDIO). 0x0: Clock Ref bypass 0x1: Clock Ref bypass 0x2: Clock Ref / 2 0x3: Clock Ref / 3 0x3FF: Clock Ref / 1023	RW	0x000
5:3	Reserved	Reserved bit field. Do not write any value	R	0x0
2	CEN	Clock enable. This bit controls if the clock is provided to the card or not. 0x0: The clock is not provided to the card . Clock frequency can be changed . 0x1: The clock is provided to the card and can be automatically gated when MMCI.MMCHS_SYSCONFIG[0] AUTOIDLE bit is set to 1 (default value) . The host driver shall wait to set this bit to 1 until the Internal clock is stable (MMCI.MMCHS_SYSCCTL[1] ICS bit).	RW	0
1	ICS	Internal clock stable (status)This bit indicates either the internal clock is stable or not. Read 0x0: The internal clock is not stable.	R	0

MMC/SD/SDIO Registers

Bits	Field Name	Description	Type	Reset
		Read 0x1: The internal clock is stable after enabling the clock (MMCi.MMCHS_SYSCTL[0] ICE bit) or after changing the clock ratio (MMCi.MMCHS_SYSCTL[15:6] CLKD bits).		
0	ICE	Internal clock enable. This register controls the internal clock activity. In very low power state, the internal clock is stopped. Note: The activity of the debounce clock (used for wakeup events) and the interface clock (used for reads and writes to the module register map) are not affected by this register. 0x0: The internal clock is stopped (very low power state). 0x1: The internal clock oscillates and can be automatically gated when MMCi.MMCHS_SYSCONFIG[0] AUTOIDLE bit is set to 1 (default value) .	RW	0

Table 1-69. Register Call Summary for Register MMCHS_SYSCTL

MMC/SD/SDIO Environment

- [MMC/SD/SDIO Protocol and Data Format: \[0\]](#)

MMC/SD/SDIO Integration

- [Resets: \[1\] \[2\] \[3\] \[4\] \[5\]](#)

Use Cases and Tips

- [Programming Flow: \[6\] \[7\] \[8\] \[9\] \[10\] \[11\] \[12\] \[13\] \[14\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[15\] \[16\] \[17\] \[18\] \[19\] \[20\] \[21\] \[22\] \[23\] \[24\] \[25\] \[26\] \[27\] \[28\] \[29\] \[30\] \[31\]](#)

1.7.2.18 MMCHS_STAT

Table 1-70. MMCHS_STAT

Address Offset	0x130			
Physical Address	0x4809 C130	Instance	MMCHS1	
	0x480A D130		MMCHS3	
	0x480B 4130		MMCHS2	
Description	Interrupt status register The interrupt status regroups all the status of the module internal events that can generate an interrupt. MMCHS_STAT[31:16] = Error Interrupt Status MMCHS_STAT[15:0] = Normal Interrupt Status			
Type	RW			
Bits	Field Name	Description	Type	Reset
31:30	Reserved	Reserved bit field. Do not write any value	R	0x0
29	BADA	Bad access to data space. This bit is set automatically to indicate a bad access to buffer when not allowed: - During a read access to the data register (MMCi.MMCHS_DATA) while buffer reads are not allowed (MMCi.MMCHS_PSTATE[11] BRE bit =0) - During a write access to the data register (MMCi.MMCHS_DATA) while buffer writes are not allowed (MMCi.MMCHS_PSTATE[10] BWE bit=0) Read 0x0: No Interrupt. Write 0x0: Status bit unchanged Read 0x1: Bad Access Write 0x1: Status is cleared	RW	0

Bits	Field Name	Description	Type	Reset
28	CERR	Card error. This bit is set automatically when there is at least one error in a response of type R1, R1b, R6, R5 or R5b. Only bits referenced as type E (error) in status field in the response can set a card status error. An error bit in the response is flagged only if corresponding bit in card status response error MMCi.MMCHS_CSRE in set. There is no card error detection for autoCMD12 command. The host driver shall read MMCi.MMCHS_RSP76 register to detect error bits in the command response. Read 0x0: No Error Write 0x0: Status bit unchanged Read 0x1: Card error Write 0x1: Status is cleared	RW	0
27:25	Reserved	Reserved bit field. Do not write any value	R	0x0
24	ACE	Auto CMD12 error. This bit is set automatically when one of the bits in Auto CMD12 Error status register has changed from 0 to 1. Read 0x0: No Error Write 0x0: Status bit unchanged Read 0x1: AutoCMD12 error Write 0x1: Status is cleared	RW	0
23	Reserved	Reserved bit field. Do not write any value	R	0
22	DEB	Data End Bit error. This bit is set automatically when detecting a 0 at the end bit position of read data on mmci_dat line or at the end position of the CRC status in write mode. Read 0x0: No Error Write 0x0: Status bit unchanged Read 0x1: Data end bit error Write 0x1: Status is cleared	RW	0
21	DCRC	Data CRC Error. This bit is set automatically when there is a CRC16 error in the data phase response following a block read command or if there is a 3-bit CRC status different of a position "010" token during a block write command. Read 0x0: No Error. Write 0x0: Status bit unchanged Read 0x1: Data CRC error Write 0x1: Status is cleared	RW	0
20	DTO	Data timeout error. This bit is set automatically according to the following conditions: - Busy timeout for R1b, R5b response type. - Busy timeout after write CRC status. - Write CRC status timeout. - Read data timeout. Read 0x0: No error Write 0x0: Status bit unchanged Read 0x1: Time out Write 0x1: Status is cleared	RW	0
19	CIE	Command index error. This bit is set automatically when response index differs from corresponding command index previously emitted. It depends on the enable bit (MMCi.MMCHS_CMD[20] CICE). Read 0x0: No error Write 0x0: Status bit unchanged Read 0x1: Command index error Write 0x1: Status is cleared	RW	0

MMC/SD/SDIO Registers

Bits	Field Name	Description	Type	Reset
18	CEB	Command end bit error. This bit is set automatically when detecting a 0 at the end bit position of a command response. Read 0x0: No error Write 0x0: Status bit unchanged Read 0x1: Command end bit error Write 0x1: Status is cleared	RW	0
17	CCRC	Command CRC Error. This bit is set automatically when there is a CRC7 error in the command response depending on the enable bit (MMCi.MMCHS_CMD[19] CCCE). Read 0x0: No Error Write 0x0: Status bit unchanged Read 0x1: Command CRC error Write 0x1: Status is cleared	RW	0
16	CTO	Command Timeout Error. This bit is set automatically when no response is received within 64 clock cycles from the end bit of the command. For commands that reply within 5 clock cycles - the timeout is still detected at 64 clock cycles. Read 0x0: No error Write 0x0: Status bit unchanged Read 0x1: Time Out Write 0x1: Status is cleared	RW	0
15	ERRI	Error Interrupt. If any of the bits in the Error Interrupt Status register (MMCi.MMCHS_STAT[31:16]) are set, then this bit is set to 1. Therefore the host driver can efficiently test for an error by checking this bit first. Writes to this bit are ignored. Read 0x0: No Interrupt Read 0x1: Error interrupt event(s) occurred	R	0
14:10	Reserved	Reserved bit field. Do not write any value.	R	0x00
9	OBI	Out-Of-Band interrupt (This interrupt is only useful for MMC card). This bit is set automatically when MMCi.MMCHS_CON[14] OBIE bit is set and an Out-of-Band interrupt occurs on OBI pin. The interrupt detection depends on polarity controlled by MMCi.MMCHS_CON[13] OBIP bit. The Out-of-Band interrupt signal is a system specific feature for future use, this signal is not required for existing specification implementation. Read 0x0: No Out-Of-Band interrupt. Write 0x0: Status bit unchanged. Read 0x1: Interrupt Out-Of-Band occurs. Write 0x1: Status is cleared.	R	0
8	CIRQ	Card interrupt. This bit is only used for SD and SDIO cards. In 1-bit mode, interrupt source is asynchronous (can be a source of asynchronous wakeup). In 4-bit mode, interrupt source is sampled during the interrupt cycle. In CE-ATA mode, interrupt source is detected when the card drives mmc_i_cmd line to zero during one cycle after data transmission end. All modes above are fully exclusive. The controller interrupt must be clear by setting MMCi.MMCHS_IE[8] CIRQ_ENABLE to 0, then the host driver must start the interrupt service with card (clearing card interrupt status) to remove card interrupt source. Otherwise the Controller interrupt will be reasserted as soon as MMCi.MMCHS_IE[8] CIRQ_ENABLE is set to 1. Writes to this bit are ignored. Read 0x0: No card interrupt Read 0x1: Generate card interrupt	R	0
7:6	Reserved	Reserved bit field. Do not write any value.	RW	00

Bits	Field Name	Description	Type	Reset
5	BRR	Buffer read ready. This bit is set automatically during a read operation to the card (see class 2 - block oriented read commands) when one block specified by the MMCi.MMCHS_BLK[10:0] BLEN bitfield is completely written in the buffer. It indicates that the memory card has filled out the buffer and that the local host needs to empty the buffer by reading it. Note: If the DMA receive-mode is enabled, this bit is never set; instead a DMA receive request to the main DMA controller of the system is generated. Read 0x0: Not Ready to read buffer Write 0x0: Status bit unchanged Read 0x1: Ready to read buffer Write 0x1: Status is cleared	RW	0
4	BWR	Buffer write ready. This bit is set automatically during a write operation to the card (see class 4 - block oriented write command) when the host can write a complete block as specified by MMCi.MMCHS_BLK[10:0] BLEN. It indicates that the memory card has emptied one block from the buffer and that the local host is able to write one block of data into the buffer. Note: If the DMA transmit mode is enabled, this bit is never set; instead, a DMA transmit request to the main DMA controller of the system is generated. Read 0x0: Not Ready to write buffer Write 0x0: Status bit unchanged Read 0x1: Ready to write buffer Write 0x1: Status is cleared	RW	0
3	Reserved	Reserved bit field. Do not write any value	R	0
2	BGE	Block gap event. When a stop at block gap is requested (MMCi.MMCHS_HCTL[16] SBGR bit), this bit is automatically set when transaction is stopped at the block gap during a read or write operation. This event does not occur when the stop at block gap is requested on the last block. In read mode, a 1-to-0 transition of the mmci_dat line active status (MMCi.MMCHS_PSTATE[2] DLA bit) between data blocks generates a Block gap event interrupt. Read 0x0: No block gap event Write 0x0: Status bit unchanged Read 0x1: Transaction stopped at block gap Write 0x1: Status is cleared	RW	0
1	TC	Transfer completed. This bit is always set when a read/write transfer is completed or between two blocks when the transfer is stopped due to a stop at block gap request (MMCi.MMCHS_HCTL[16] SBGR bit). This bit is also set when exiting a command in a busy state (if the command has a busy notification capability). In Read mode: This bit is automatically set on completion of a read transfer (MMCi.MMCHS_PSTATE[9] RTA bit). In write mode: This bit is set automatically on completion of the mmci_dat line use (MMCi.MMCHS_PSTATE[2] DLA bit). Read 0x0: No transfer complete Write 0x0: Status bit unchanged Read 0x1: Data transfer complete Write 0x1: Status is cleared	RW	0
0	CC	Command complete. This bit is set when a 1-to-0 transition occurs in the register command inhibit (MMCi.MMCHS_PSTATE[0] CMDI bit) If the command is a type for which no response is expected, then the command complete interrupt is generated at the end of the command. A command timeout error (MMCi.MMCHS_STAT[16] CTO bit) has higher priority than command complete (MMCi.MMCHS_STAT[0] CC bit). If a response is expected but none is received, then a command timeout error is detected and signaled instead of the command complete interrupt.	RW	0

MMC/SD/SDIO Registers

Bits	Field Name	Description	Type	Reset
		Read 0x0: No Command complete		
		Write 0x0: Status bit unchanged		
		Read 0x1: Command complete		
		Write 0x1: Status is cleared		

Table 1-71. Register Call Summary for Register MMCHS_STAT

MMC/SD/SDIO Integration

- [Clocks: \[0\] \[1\] \[2\] \[3\]](#)
- [Interrupt Requests: \[4\] \[5\] \[6\] \[7\] \[8\] \[9\] \[10\] \[11\] \[12\] \[13\] \[14\] \[15\] \[16\] \[17\] \[18\]](#)

MMC/SD/SDIO Functional Description

- [Data Buffer: \[19\] \[20\] \[21\] \[22\] \[23\] \[24\]](#)
- [Transfer or Command Status and Errors Reporting: \[25\]](#)
- [MMC CE-ATA Command Completion Disable Management: \[26\]](#)

Use Cases and Tips

- [Programming Flow: \[27\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[28\] \[29\] \[30\] \[31\] \[32\] \[33\] \[34\] \[35\] \[36\] \[37\] \[38\] \[39\] \[40\] \[41\] \[42\] \[43\] \[44\] \[45\] \[46\] \[47\] \[48\] \[49\] \[50\] \[51\] \[52\] \[53\] \[54\] \[55\] \[56\] \[57\] \[58\]](#)

1.7.2.19 MMCHS_IE

Table 1-72. MMCHS_IE

Address Offset	0x134			
Physical Address	0x4809 C134	Instance	MMCHS1	
	0x480A D134		MMCHS3	
	0x480B 4134		MMCHS2	
Description	Interrupt SD enable register This register allows to enable/disable the module to set status bits, on an event-by-event basis. MMCHS_IE[31:16] = Error Interrupt Status Enable MMCHS_IE[15:0] = Normal Interrupt Status Enable			
Type	RW			
Bits	Field Name	Description	Type	Reset
31:30	Reserved	Reserved bit field. Do not write any value.	R	0
29	BADA_ENABLE	Bad access to data space Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
28	CERR_ENABLE	Card error interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
27:25	Reserved	Reserved bit field. Do not write any value	R	0x0
24	ACE_ENABLE	Auto CMD12 error Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
23	Reserved	Reserved bit field. Do not write any value	R	0
22	DEB_ENABLE	Data end bit error Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
21	DCRC_ENABLE	Data CRC error Interrupt Enable 0x0: Masked	RW	0

Bits	Field Name	Description	Type	Reset
		0x1: Enabled		
20	DTO_ENABLE	Data timeout error Interrupt Enable 0x0: The data timeout detection is deactivated. The host controller provides the clock to the card until the card sends the data or the transfer is aborted. 0x1: The data timeout detection is enabled.	RW	0
19	CIE_ENABLE	Command index error Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
18	CEB_ENABLE	Command end bit error Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
17	CCRC_ENABLE	Command CRC error Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
16	CTO_ENABLE	Command timeout error Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
15	NULL	Fixed to 0 The host driver shall control error interrupts using the Error Interrupt Signal Enable register. Writes to this bit are ignored.	R	0
14:10	Reserved	Reserved bit field. Do not write any value.	R	0x00
9	OBI_ENABLE	Out-of-Band interrupt Enable A write to this register when MMCi.MMCHS_CON[14] OBIE is set to '0' is ignored. 0x0: Masked 0x1: Enabled	RW	0
8	CIRQ_ENABLE	Card interrupt Enable A clear of this bit also clears the corresponding status bit. During 1-bit mode, if the interrupt routine does not remove the source of a card interrupt in the SDIO card, the status bit is reasserted when this bit is set to 1. This bit must be set to 1 when entering in smart idle mode to enable system to identity wakeup event and to allow controller to clear internal wakeup source. 0x0: Masked 0x1: Enabled	RW	0
7:6	Reserved	Reserved bit field. Do not write any value	RW	00
5	BRR_ENABLE	Buffer Read Ready Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
4	BWR_ENABLE	Buffer Write Ready Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
3	Reserved	Reserved bit field. Do not write any value.	R	0
2	BGE_ENABLE	Block Gap Event Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
1	TC_ENABLE	Transfer completed Interrupt Enable 0x0: Masked 0x1: Enabled	RW	0
0	CC_ENABLE	Command completed Interrupt Enable 0x0: Masked	RW	0

MMC/SD/SDIO Registers

Bits	Field Name	Description	Type	Reset
		0x1: Enabled		

Table 1-73. Register Call Summary for Register MMCHS_IE

MMC/SD/SDIO Integration

- [Clocks: \[0\] \[1\]](#)
- [Interrupt Requests: \[2\] \[3\] \[4\] \[5\]](#)

Use Cases and Tips

- [Programming Flow: \[6\] \[7\] \[8\] \[9\] \[10\] \[11\] \[12\] \[13\] \[14\] \[15\] \[16\] \[17\] \[18\] \[19\] \[20\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[21\] \[22\] \[23\] \[24\] \[25\] \[26\] \[27\] \[28\]](#)

1.7.2.20 MMCHS_I SE

Table 1-74. MMCHS_I SE

Address Offset	0x138			
Physical Address	0x4809 C138	Instance	MMCHS1	
	0x480A D138		MMCHS3	
	0x480B 4138		MMCHS2	
Description	Interrupt signal enable register This register allows to enable/disable the module internal sources of status, on an event-by-event basis. MMCHS_I SE[31:16] = Error Interrupt Signal Enable MMCHS_I SE[15:0] = Normal Interrupt Signal Enable			
Type	RW			
Bits	Field Name	Description	Type	Reset
31:30	Reserved	Reserved bit field. Do not write any value.	R	0
29	BADA_SIGEN	Bad access to data space signal status Enable 0x0: Masked 0x1: Enabled	RW	0
28	CERR_SIGEN	Card error interrupt signal status Enable 0x0: Masked 0x1: Enabled	RW	0
27:25	Reserved	Reserved bit field. Do not write any value	R	0x0
24	ACE_SIGEN	Auto CMD12 error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
23	Reserved	Reserved bit field. Do not write any value	R	0
22	DEB_SIGEN	Data end bit error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
21	DCRC_SIGEN	Data CRC error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
20	DTO_SIGEN	Data timeout error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
19	CIE_SIGEN	Command index error signal status Enable 0x0: Masked 0x1: Enabled	RW	0

Bits	Field Name	Description	Type	Reset
18	CEB_SIGEN	Command end bit error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
17	CCRC_SIGEN	Command CRC error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
16	CTO_SIGEN	Command timeout error signal status Enable 0x0: Masked 0x1: Enabled	RW	0
15	NULL	Fixed to 0 The host driver shall control error interrupts using the Error Interrupt Signal Enable register. Writes to this bit are ignored	R	0
14:10	Reserved	Reserved bit field. Do not write any value	R	0x00
9	OBI_SIGEN	Out-Of-Band Interrupt signal status Enable. A write to this register when MMCHS_CON[14] OBIE bit is set to '0' is ignored. 0x0: Masked 0x1: Enabled	RW	0
8	CIRQ_SIGEN	Card interrupt signal status Enable 0x0: Masked 0x1: Enabled	RW	0
7	Reserved	Reserved bit field. Do not write any value	RW	0
6	Reserved	Reserved bit field. Do not write any value	RW	0
5	BRR_SIGEN	Buffer Read Ready signal status Enable 0x0: Masked 0x1: Enabled	RW	0
4	BWR_SIGEN	Buffer Write Ready signal status Enable 0x0: Masked 0x1: Enabled	RW	0
3	Reserved	Reserved bit field. Do not write any value	R	0
2	BGE_SIGEN	Black Gap Event signal status Enable 0x0: Masked 0x1: Enabled	RW	0
1	TC_SIGEN	Transfer completed signal status Enable 0x0: Masked 0x1: Enabled	RW	0
0	CC_SIGEN	Command completed signal status Enable 0x0: Masked 0x1: Enabled	RW	0

Table 1-75. Register Call Summary for Register MMCHS_ISE

MMC/SD/SDIO Integration

- [Clocks: \[0\] \[1\]](#)
- [Interrupt Requests: \[2\] \[3\] \[4\] \[5\] \[6\]](#)

Use Cases and Tips

- [Programming Flow: \[7\] \[8\] \[9\] \[10\] \[11\] \[12\] \[13\] \[14\] \[15\] \[16\] \[17\] \[18\] \[19\] \[20\] \[21\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[22\] \[23\] \[24\] \[25\] \[26\] \[27\]](#)

1.7.2.21 MMCHS_AC12

Table 1-76. MMCHS AC12

Address Offset	0x13C																																																																																													
Physical Address	0x4809 C13C								Instance								MMCHS1																																																																													
	0x480A D13C																MMCHS3																																																																													
	0x480B 413C																MMCHS2																																																																													
Description	Auto CMD12 Error Status Register. The host driver may determine which of the errors cases related to Auto CMD12 has occurred by checking this MMCi.MMCHS_AC12 register when an Auto CMD12 Error interrupt occurs. This register is valid only when Auto CMD12 is enabled (MMCi.MMCHS_CMD[2] ACEN bit) and Auto CMD12Error (MMCi.MMCHS_STAT[24] ACE bit) is set to 1. Note: These bits are automatically reset when starting a new adtc command with data.																																																																																													
Type	R																																																																																													
<table><tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td colspan="24">Reserved</td><td>CNI</td><td>Reserved</td><td>ACIE</td><td>ACEB</td><td>ACCE</td><td>ACTO</td><td>ACNE</td></tr></table>																																31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Reserved																								CNI	Reserved	ACIE	ACEB	ACCE	ACTO	ACNE
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																																															
Reserved																								CNI	Reserved	ACIE	ACEB	ACCE	ACTO	ACNE																																																																
Bits	Field Name	Description																				Type				Reset																																																																				
31:8	Reserved	Reserved bit field. Do not write any value.																				R				0x0																																																																				
7	CNI	Command not issue by Auto CMD12 error. If this bit is set to 1, it means that pending command is not executed due to Auto CMD12 error: ACEB, ACCE, ACTO or ACNE. Read 0x0: Not error Read 0x1: Command not issued																				R				0																																																																				
6:5	Reserved	Reserved bit field. Do not write any value.																				R				0x0																																																																				
4	ACIE	Auto CMD12 index error. This bit is a set to 1 when response index differs from corresponding command auto CMD12 index previously emitted. This bit depends on the command index check enable (MMCi.MMCHS_CMD[20] CICE bit). Read 0x0: No error Read 0x1: Auto CMD12 Index Error																				R				0																																																																				
3	ACEB	Auto CMD12 end bit error. This bit is set to 1 when detecting a 0 at the end bit position of auto CMD12 command response. Read 0x0: No error Read 0x1: AutoCMD12 End bit Error																				R				0																																																																				
2	ACCE	Auto CMD12 CRC error. This bit is automatically set to 1 when a CRC7 error is detected in the auto CMD12 command response depending on the enable in the MMCi.MMCHS_CMD[19] CCCE bit. Read 0x0: No error Read 0x1: Auto CMD12 CRC Error																				R				0																																																																				
1	ACTO	Auto CMD12 timeout error. This bit is set to 1 if no response is received within 64 clock cycles from the end bit of the auto CMD12 command. Read 0x0: No error Read 0x1: Auto CMD12 Time Out																				R				0																																																																				
0	ACNE	Auto CMD12 not executed. This bit is set to 1 if multiple block data transfer command has started and if an error occurs in command before Auto CMD12 starts. Read 0x0: Auto CMD12 Executed Read 0x1: Auto CMD12 Not Executed																				R				0																																																																				

Table 1-77. Register Call Summary for Register MMCHS_AC12

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[0\] \[1\] \[2\] \[3\] \[4\]](#)

1.7.2.22 MMCHS_CAPA

Table 1-78. MMCHS_CAPA

Address Offset	0x140																														
Physical Address	0x4809 C140															Instance	MMCHS1														
	0x480A D140																MMCHS3														
	0x480B 4140																MMCHS2														
Description	Capabilities register. This register lists the capabilities of the MMC/SD/SDIO host controller.																														
Type	RW																														
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					VS18	VS30	VS33	SRS	DS	HSS	Reserved				MBL		Reserved	BCF					TCU	Reserved	TCF						
Bits	Field Name		Description																	Type		Reset									
31:27	Reserved		Reserved bit field. Do not write any value.																	R		0x00									
26	VS18		Voltage support 1.8V Initialization of this register (via a write access to this register) depends on the system capabilities. The host driver shall not modify this register after the initialization. This register is only reinitialized by a hard reset (via MMCi_RESET signal). Read 0x0: 1.8V Not Supported Write 0x0: 1.8V Not supported Read 0x1: 1.8V Supported Write 0x1: 1.8V Supported MMCHS1, 2 and 3: This bit must be set to 1.																	RW		0									
25	VS30		Voltage support 3.0V Initialization of this register (via a write access to this register) depends on the system capabilities. The host driver shall not modify this register after the initialization. This register is only reinitialized by a hard reset (via MMCi_RESET signal) Read 0x0: 3.0V Not Supported Write 0x0: 3.0V Not supported Read 0x1: 3.0V Supported Write 0x1: 3.0V Supported MMCHS1: This bit must be set to 1. MMCHS2 and 3: This bit must be left to 0.																	RW		0									
24	VS33		Voltage support 3.3V Initialization of this register (via a write access to this register) depends on the system capabilities. The host driver shall not modify this register after the initialization. This register is only reinitialized by a hard reset (via MMCi_RESET signal) Read 0x0: 3.3V Not Supported Write 0x0: 3.3V Not supported Read 0x1: 3.3V Supported Write 0x1: 3.3V Supported MMCHS1, 2 and 3: This bit must be left to 0.																	RW		0									
23	SRS		Suspend/Resume support (SDIO cards only)This bit indicates whether the host controller supports Suspend/Resume functionality.																	R		1									

MMC/SD/SDIO Registers

Bits	Field Name	Description	Type	Reset
		Read 0x0: The Host controller does not Suspend/Resume functionality. Read 0x1: The Host controller supports Suspend/Resume functionality.		
22	DS	DMA support. This bit indicates that the Host Controller is able to use DMA to transfer data between system memory and the Host Controller directly. Read 0x0: DMA Not Supported Read 0x1: DMA Supported	R	1
21	HSS	High speed support. This bit indicates that the host controller supports high speed operations and can supply an up-to-52 MHz clock to the card. Read 0x0: High Speed Not Supported Read 0x1: High Speed Supported	R	1
20:18	Reserved	Reserved bit field. Do not write any value.	R	0x0
17:16	MBL	Maximum block length. This value indicates the maximum block size that the host driver can read and write to the buffer in the host controller. The host controller supports 512 bytes and 1024 bytes block transfers. Read 0x0: 512 bytes Read 0x1: 1024 bytes Read 0x2: 2048 bytes	R	0x1
15:14	Reserved	Reserved bit field. Do not write any value	R	0x0
13:8	BCF	Base clock frequency for clock provided to the card. Read 0x0: The value indicating the base (maximum) frequency for the output clock provided to the card is system dependent and is not available in this register. Get the information via another method. See the <i>Power Reset and Clock Management</i> chapter for more information on the value of FUNC_96M_CLK clock signal.	R	0x00
7	TCU	Timeout clock unit. This bit shows the unit of base clock frequency used to detect Data Timeout Error (MMCi.MMCHS_STAT[20] DTO bit). Read 0x0: kHz Read 0x1: MHz	R	1
6	Reserved	Reserved. This bit is initialized to zero, and writes to it are ignored.	R	0
5:0	TCF	Timeout clock frequency. The timeout clock frequency is used to detect Data Timeout Error (MMCi.MMCHS_STAT[20] DTO bit). Read 0x0: The timeout clock frequency depends on the frequency of the clock provided to the card. The value of the timeout clock frequency is not available in this register. Note: You can have the timeout clock frequency by dividing FUNC_96M_CLK by the value of the MMCi.MMCHS_SYSCTL[15:6] CLKD bitfield.	R	0x00

Table 1-79. Register Call Summary for Register MMCHS_CAPA

MMC/SD/SDIO Integration

- [Resets: \[0\]](#)

MMC/SD/SDIO Functional Description

- [Data Buffer: \[1\]](#)

MMC/SD/SDIO Basic Programming Model

- [Set MMCHS Default Capabilities: \[2\]](#)

Use Cases and Tips

- [Programming Flow: \[3\] \[4\] \[5\]](#)

MMC/SD/SDIO Registers

- [MMC/SD/SDIO Registers Mapping Summary: \[6\] \[7\] \[8\] \[9\] \[10\] \[11\] \[12\] \[13\] \[14\]](#)

1.7.2.23 MMCHS_CUR_CAPA

Table 1-80. MMCHS_CUR_CAPA

Address Offset	0x148			
Physical Address	0x4809 C148	Instance	MMCHS1	
	0x480A D148		MMCHS3	
	0x480B 4148		MMCHS2	
Description	Maximum current capabilities Register. This register indicates the maximum current capability for each voltage. The value is meaningful if the voltage support is set in the capabilities register (MMCi.MMCHS_CAPA). Initialization of this register (via a write access to this register) depends on the system capabilities. The host driver shall not modify this register after the initialization. This register is only reinitialized by a hard reset (via MMCi_RESET signal)			
Type	RW			

Bits	Field Name	Description	Type	Reset
31:24	Reserved	Reserved. This bit is initialized to zero, and writes to it are ignored.	R	0x0
23:16	CUR_1V8	Maximum current for 1.8V	RW	0x0
		Read 0x0 The maximum current capability for this voltage is not available. Feature not implemented.		
15:8	CUR_3V0	Maximum current for 3.0V	RW	0x0
		Read 0x0 The maximum current capability for this voltage is not available. Feature not implemented.		
7:0	CUR_3V3	Maximum current for 3.3V	RW	0x0
		Read 0x0 The maximum current capability for this voltage is not available. Feature not implemented.		

Table 1-81. Register Call Summary for Register MMCHS_CUR_CAPA

MMC/SD/SDIO Integration
• Resets: [0]
MMC/SD/SDIO Basic Programming Model
• Set MMCHS Default Capabilities: [1]
MMC/SD/SDIO Registers
• MMC/SD/SDIO Registers Mapping Summary: [2] [3] [4]

1.7.2.24 MMCHS_REV

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 TI 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. Information of third parties may be subject to additional restrictions.

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.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

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

Products

Amplifiers	amplifier.ti.com
Data Converters	dataconverter.ti.com
DSP	dsp.ti.com
Clocks and Timers	www.ti.com/clocks
Interface	interface.ti.com
Logic	logic.ti.com
Power Mgmt	power.ti.com
Microcontrollers	microcontroller.ti.com
RFID	www.ti-rfid.com
RF/IF and ZigBee® Solutions	www.ti.com/lprf

Applications

Audio	www.ti.com/audio
Automotive	www.ti.com/automotive
Broadband	www.ti.com/broadband
Digital Control	www.ti.com/digitalcontrol
Medical	www.ti.com/medical
Military	www.ti.com/military
Optical Networking	www.ti.com/opticalnetwork
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
Copyright © 2008, Texas Instruments Incorporated