SAMSUNG MultiMediaCard

Product Datasheet

Version 0.3
September 2005

Document Title

SAMSUNG MultiMediaCard

Revision History

Revision No	History	Date	Remark
0.0	- Initial issue - Changed model number (page 5,6)	Dec. 6. 2004 Feb. 22. 2005	Draft 6 Draft 7
0.1	- MMCmobile and MMCplus definitions were introduced. A minimum performance definition was introduced. (page 56,57) - Added new item to Spec_Vers register (page 25) - Added new item to CSD_Structure register (page 25,35) - Added new item to Extended CSD revision register (page 35) - Added Min_Performance register to EXT_CSD regiser (page 32,33)	Mar. 29. 2005	Preliminary
0.2	- Added 1GB Standard-Size High Speed MultiMediaCard - Added 512MB Dual Voltage Reduced-Size High Speed MultiMediaCard - Contens of Memory Array Partioning were clarified (page 8) - Several typos were corrected throughout the book	July. 21. 2005	
0.3	- Added 2GB Standard-Size High Speed MultiMediaCard and 1GB Dual Voltage Reduced-Size High Speed MultiMediaCard	Sep. 22. 2005	

TABLE OF CONTENTS

1. Ordering Information

M C X X X X X X X X X X - X X X X X

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18

1. Module: M

2. Card: C

3~4. Flash Density

64 : 64M	28 : 128M
56 : 256M	12 : 512M
5D : 512M DDP	1G : 1G
1D : 1G DDP	2G : 2G
2D : 2G DDP	4D : 4G DDP
4Q : 4G QDP	8Q : 8G QDP

5. Feature
H : High Speeed MultiMediaCard

6~8. Card Density

008 : 8M Byte	016 : 16M Byte
032 : 32M Byte	048 : 48M Byte
064 : 64M Byte	096 : 96M Byte
128 : 128M Byte	192 : 192M Byte
256 : 256M Byte	384 : 384M Byte
512 : 512M Byte	01G : 1G Byte
02G : 2G Byte	04G : 4G Byte
08G : 8G Byte	

9. Card Type
N : Standard-Size MultiMediaCard
H : Reduced-Size MultiMediaCard
D : Dual Voltage Reduced-Size MultiMediaCard

10. Component Generation

M : 1st Generation	A : 2nd Generation
B : 3rd Generation	C : 4th Generation
D : 5th Generation	

11. Flash Package

C : CHIP	Y : TSOP1
V : WSOP	B : TBGA

12. PCB Revision

A : None	B : 1st Rev.
C : 2nd Rev.	

13. " - "

14. Packing Type
0 : Bulk Type I
1 : Bulk Type II (By White Case)
2 : Bulk Type I (No Label)
3 : Bulk Type II (No Label)
4 : Bulk Type I (only Back Label)
5 : Bulk Type II (only Back Label)

15. Controller
S : S3F49SAX

16. Controller Firmware Revision
A : None
B : 1st Rev.
C : 2nd Rev.
D : 3rd Rev.
E : 4th Rev.

17 ~ 18. Customer Grade
" Customer List Reference "

2. Product Line-up

Model Number	Capacities	Remarks
MC12H064NBCA-2SA00	64MB	Standard-Size High Speed MultiMediaCard
MC1GH128NACA-2SA00	128MB	
MC2GH256NMCA-2SA00	256MB	
MC2GH512NMCA-2SA00	512MB	
MC4GH01GNMCA-2SA00	1GB	
MC4GH02GNMCA-2SA00	2GB	

Model Number	Capacities	Remarks
MC12H064DBCA-2SA00	64MB	Dual Voltage Reduced-Size High Speed MultiMediaCard
MC1GH128DACA-2SA00	128MB	
MC1GH256DACA-2SA00	256MB	
MC2GH512DMCA-2SA00	512MB	
MC2GH01GDMCA-2SA00	1GB	

3. Introduction

3.1 General Description

The SAMSUNG MultiMediaCard is a universal low cost data storage and communication media. It is designed to cover a wied area of applications as smart phones, cameras, organizers, PDAs, digital recorders, MP3 players, pagers, electronic toys, etc. Targeted features are high mobility and high performance at a low cost price. It might also be expressed in terms of low power consumption and high data throughput at the memory card interface.

The MultiMedaiCard communication is based on an advanced 13-pin bus. The communication protocol is defined as a part of this standard and referred to as MultiMediaCard mode. For compatibility to existing controllers the cards may offer, in addition to the MultiMediaCard mode, an alternate communication protocol which is based on the SPI standard.

The SAMSUNG MultiMediaCard is very small, removable flash storage devices, designed specifically for storage applications that put a premium on small form factor, low power and low cost. Flash is the ideal storage medium for portable, battery-powered devices. It features low power consumption and is non-volatile, requiring no power to maintain the stored data. It also has a wide operating range for temperature, shock and vibration.

3.2 System Features

- MultiMediaCard System Specification Ver. 4.1 compatible
- Full backward compatibilty with previous MultiMediaCard system (1bit data bus, multi-card systems)
- Supports SPI Mode (Single and multiple block read and write operations)
- Maximum data rate with up to 52MB/sec interface speed (using 8 parallel data lines)
- Voltage Range :

	High Voltage MultiMediaCard	Dual Voltage MultiMediaCard
Communication	2.7 - 3.6	1.65 - 1.95, 2.7 - 3.6[a]
Memory Access	2.7 - 3.6	1.65 - 1.95, 2.7 - 3.6

- Card supported clock frequencies 0~20MHz, 0~26MHz, 0~52MHz
- Card support for three different data bus width modes: 1bit(default), 4bit and 8 bit
- Two form factors: Normal size(24mm x 32mm x 1.4mm) and Reduced size (24mm x 18mm x 1.4mm)
- Correction of memory field errors
- Simple erase mechanism
- Password Protection of cards

3.3 Flash Independent Technology

The 512 byte sector size of the SAMSUNG MultiMediaCard is the same as that in an IDE magnetic disk drive. To write or read a sector (or multiple sectors), the host computer software simply issues a Read or Write command to the MultiMedia Card. This command contains the address and the number of sectors to write/read. The host software then waits for the command to complete. The host software does not get involved in the details of how the flash memory is erased, programmed or read. This is extremely important as flash devices are expected to get more and more complex in the future. Because the MultiMediaCard uses an intelligent on-board controller, the host system software will not require changing as new flash memory evolves. In other words, systems that support the MultiMeidaCard today will be able to access future MultiMediaCards built with new flash technology without having to update or change host software.

3.4 Defect and Error Management

The SAMSUNG MultiMediaCards contain a sophisticated defect and error management system. This system is analogous to the systems found in magnetic disk drives and in many cases offers enhancements. For instance, disk drives do not typically perform a read after write to confirm the data is written correctly because of the performance penalty that would be incurred. MultiMediaCards do a read after write under margin conditions to verify that the data is written correctly (except in the case fo a Write without Erase Command). In the rare case that a bit is found to be defective. MultiMediaCards will even replace the entire sector with a spare sector. This is completely transparent to the host and does not consume any user data space.
The MultiMediaCards soft error rate specification is much better than the magnetic disk drive specification. In the extremely rare case a read error does occur, MultiMediaCards have innovative algorithms to recover the data. This is similar to using retries on a disk drive but is much more sophisticated. The last line of defense is to employ powerful ECC to correct the data. If ECC is used to recover data, defective bits are replaced with spare bits to ensure they do not cause any future problems.
These defect and error management systems coupled with the solid-state construction give MultiMediaCards unparalleled reliability.

3.5 Endurance

MultiMediaCards have an endurance specification for each sector of 100,000 writes (reading a logical sector is unlimited). This is far beyond what is needed in nearly all applications of MultiMediaCards. Even very heavy use fo the MultiMediaCard in celluar phone, personal communicators, pagers and voice recorders will use only a fraction of the total endurance over the typical device's five year lifetime. For instance, it would take over 100 years to wear out an area on the MultiMediaCard on which a files of any size (from 512bytes to capacity) was rewritten 3 times per hour, 8 hours a day, 365 days per year. With typical applications the endurance limit is not of any practical concern to the vast majority of users.

3.6 Automatic Sleep Mode

An important feature of the MultiMediaCard is automatic entrance and exit from sleep mode. Upon completion of an operation, the MultiMediaCard will enter the sleep mode to conserve power if no further commands are received within 5 msec. The host does not have to take any action for this to occur. In most systems, the MultiMediaCard is in sleep mode except when the host is acccessing it, thus conserving power. When the host is ready to access the MultiMediaCard and it is in sleep moed, any command issued to the MultiMediaCard will cause it to exit sleep and respond. The host does not have to issue a reset first. It may do this if desired, but it is not needed. By not issuing the reset, performance is improved through the reduction of overhead.

3.7 Hot Insertion

Support for the hot insertion will be required on the host but will be supported through the connector. Connector manufactuers will provide connectors that have power pins long enough to be powered before contact is made with the other pins. Please see connector data sheets for more details. This approach is similar to that used in PCMCIA to allow for hot insertion. This applies to both MultiMediaCard and SPI modes.

3.8 MultiMediaCard Mode

3.8.1 MultiMediaCard Standard Compliance

The SAMSUNG MultiMediaCard is fully compliant with MultiMediaCard standard specification V4.1

3.8.2 Negotiating Operation Conditions

The MultiMediaCard supports the operation condition verification sequence defined in the MultiMediaCard standard specifications. The MultiMediaCard host should define an operating voltage range that is not supported by the MultiMediaCard. It will put itself in an inactive state and ignore any bus communication. The only way to get the card out of the inactive state is by powering it down and up again. In addtion the host can explicitly send the card to the inactive state by using the GO_INACTIVE_STATE command.

3.8.3 Card Status

MultiMediaCard status is stored in a 32 bit status register which is sent as the data field in the card respond to host commands. Status register provides information about the card's current state and completion codes for the last host command. The card status can be explicityly read(polled) with the SEND_STATUS command.

3.8.4 Memory Array Partitioning

The basic unit of data transfer to/from the MultiMediaCard is one byte. All data transfer operations which require a block size always define block lenghs as integer multiples of bytes. Some special funcions need other partition granularity.

For block oriented commands, the following definition is used:

- Block: is the unit which is related to the block oriented read and write commands. Its size is the number of bytes which will be transferred when one block command is sent by host. The size of a block is either programmable or fixed. The information about allowed block sizes and the programmability is stored in the CSD.

For R/W cards, special erase and write protect commands are defined:

- The granularity of the erasable units is the Erase Group: The smallest number of consecutive write blocks which can be addressed for erase. The size of the Erase Group is card specific and stored in the CSD.
- The granularity of the Write Protected units is the WP-Group: The minimal unit which may be individually write protected. Its size is defined in units of erase groups. The size of a WP-Group is card specific and stored in the CSD.

3.8.5 Read and Write Operations

The MultiMediaCard supports two read/write modes.

Single Block Mode

In this mode the host reads or write one data block in a pre-specified length block transmission is protected with 16bit CRC which is generated by the sending unit and checked by the receiveing unit. Misalignment is not allowed. Every data block must be contained in a single memroy sector. The block lenght for write opertaion must be identical to the sector size and the start address aligned to a sector boundary.

Multiple Block Mode

This mode is similar to the single block mode, but the host can read/write multiple data blocks (all have the same length) which will be stored or retrived from contiguous memory addresses starting at the address specified in the command. The operation is terminated with a stop transmission command. Misalignment and block length restrictions apply to multiple blocks as well and are identical to the single block read/write operations. Multiple block read with pre-defined block is supported.

3.8.6 Data Protection in the Flash Card

Every sector is protected with an Error Correction Code (ECC). The ECC is generated (in the memory card) when the sectors are written and validated when the data is read. IF defects are found, the data is corrected prior to transmission to the host.
The MultiMediaCard can be considered error free and no additional data protection is needed. However, if an applciation uses additional, exteranal, ECC protection, the data oragnization is defined in the user writeable section of the CSD register.

3.8.7 Erase

The smallest erasable unit in the MultiMediaCard is a erase group. In order to speed up the erase procedure, multiple erase groups can be erased in the same time. The erase operation is divided into two stages.

Tagging - Selecting the Sectors for Erasing
To facilitate selection, a first command with the starting address is followed by a second command with the final address, and all erase groups within this range will be selected for erase.

Erasing - Starting the Erase Process
Tagging can address erase groups. An arbitrary selection of erase groups may be erased at one time. Tagging and erasing must follow a strict command sequence (refer to the MultiMediaCard standard sepcification for datails).

3.8.8 Write Protection

Two-card level write-protection options are available: permanent and temporary. Both can be set using the PROGRAM_CSD command (refer to CSD Programming, Section 6.2.8). The permanent write protect bit, once set, cannot be clearded. This feature is implemented in the MultiMediaCard /RS-MultiMediaCard controller firmware and not with a physical OTP cell.

3.8.9 Copy Bit

The content of an MultiMediaCard can be marked as an original or a copy using the copy bit in the CSD register. Once the Copy bit is set (maked as a copy) it cannot be cleared.
The Copy bit of the MultiMediaCard is programmed (during test and formatting on the manufacturing floor) as a copy. The MultiMediaCard can be purchased with the copy bit set (copy) or cleared, indicating the card is a master.
The One Time Programmable (OTP) characteristic of the Copy bit is implemented in the MultiMediaCard controller firmware and not with a physical OTP cell.

3.8.10 The CSD Register

All the configuration information of the MultiMediaCard is stored in the CSD register. The MSB bytes of the register contain manufacturer data and two least significant bytes contains the host controlled data - the card Copy and write protection and the user ECC register.
The host can read the CSD register and alter the host controlled data bytes using the SEND_CSD and PROGRAM_CSD commands.

3.9 SPI Mode

The SPI mode is a secondary (optional) communication protocol offered for MultiMediaCard. This mode is a subset of the MultiMediaCard protocol, designed to communicate with an SPI channel, commonly found in Motorola's(and lately a few other vendors') microcontrollers.

3.9.1 Negotiating Operation Conditions

The operating condition negotiation function of the MultiMediaCard bus is not supported in SPI mode. The host must work within the valid voltage range (2.7 to 3.6 volts) of the card.

3.9.2 Card Acquisition and Identification

The card acquisition and identification function of the MultiMediaCard bus is not supported in SPI mode. The host must know the number of cards currently connected on the bus. Specific card selection is done via the CS signal.

3.9.3 Card status

In SPI mode only 16bits (containing the errors relevant to SPI mode) can be read out of the MultiMediaCard status register.

3.9.4 Memroy Array Partitioning

Memory partioning in SPI mode is equivalent to MultiMediaCard mode. All read and write commands are byte addressable.

3.9.5 Read and Write Operations

In SPI mode, only single block read/write mode is supported.

3.9.6 Data Transfer Rate

In SPI mode, only single block mode is supported. The typical access time (latency) for each data block, in read operation, is 1.5ms. The write typical access time (latency) for each data block, in read operation, is 1.5ms. The write block operation is done in handshake mode. The card will keep DataOut line low as long as the write operation is in progress and there are no write buffers available.

3.9.7 Data protection in the MultiMediaCard

Same as in the MultiMediaCard mode.

3.9.8 Erase

Same as in the MultiMediaCard mode.

4. Product Specifications

4.1 Reliability and Durability

Temperature	operation: -25°C / 85°C storage: -40°C (168h) / 85°C (500h) junction temperature: max. 95°C
Moisture and corrosion	operation: 25°C / 95% rel. humidity stress: 40°C / 93% rel. hum./500h salt water spray: 3% NaCl/35C; 24h acc. MIL STD 883 Method 1009
ESD protection	Contact Pads: +/- 4kV, Human body model according to ANSI EOS/ESD-S5.1-1998 Non Contact Pads area: +/-8kV (coupling plane discharge) +/-15kV (air discharge) Human body model according to IEC61000-4-2
Durability	10.000 mating cycles; test t.b.d.
Bending	t.b.d.
Torque	t.b.d.
Drop test	1.5m free fall
UV light exposure	UV: 200nm, 15Ws/cm² according to ISO 7816-1
Visual inspection **Shape and form**	no warpage; no mold skin; complete form; no cavities surface smoothness sigma-0.1 mm/cm^2 within contour; no cracks; no pollution (fat, oil dust, etc.)

Table 4-1 : Environmental Specification Summary

4.2 Mechanical Design and Format

Card Package Dimensions	Normal Size: 24mm x 32mm; (min. 23.9mm x 31.9mm; max.24.1mm x 32.1mm) other dimensions Figure 4-1 Testing according to MIL STD 883, Meth 2016
	Reduced Size: 24mm x 18mm; (min. 23.9mm x 17.9mm; max.24.1mm x 18.1mm) other dimensions Figure 4-3 Testing according to MIL STD 883, Meth 2016
Thickness	1.4mm +- 0.1mm
Restrictions on usage of package material	Some area of the external surface of the card edge may not contain conductive materials (refer to Figure 4-2)
Label or printable area	Whole card, except contact area
Surface	Plain (except contact area)
Edges	Smooth edges, see Figure 4-1
Inverse insertion	Protection on right corner (top view), see Figure 4-1
Position of ESC contacts	Along middle of shorter edge; -0.625mm Offset

Table 4-2 : Physical Specification Summary

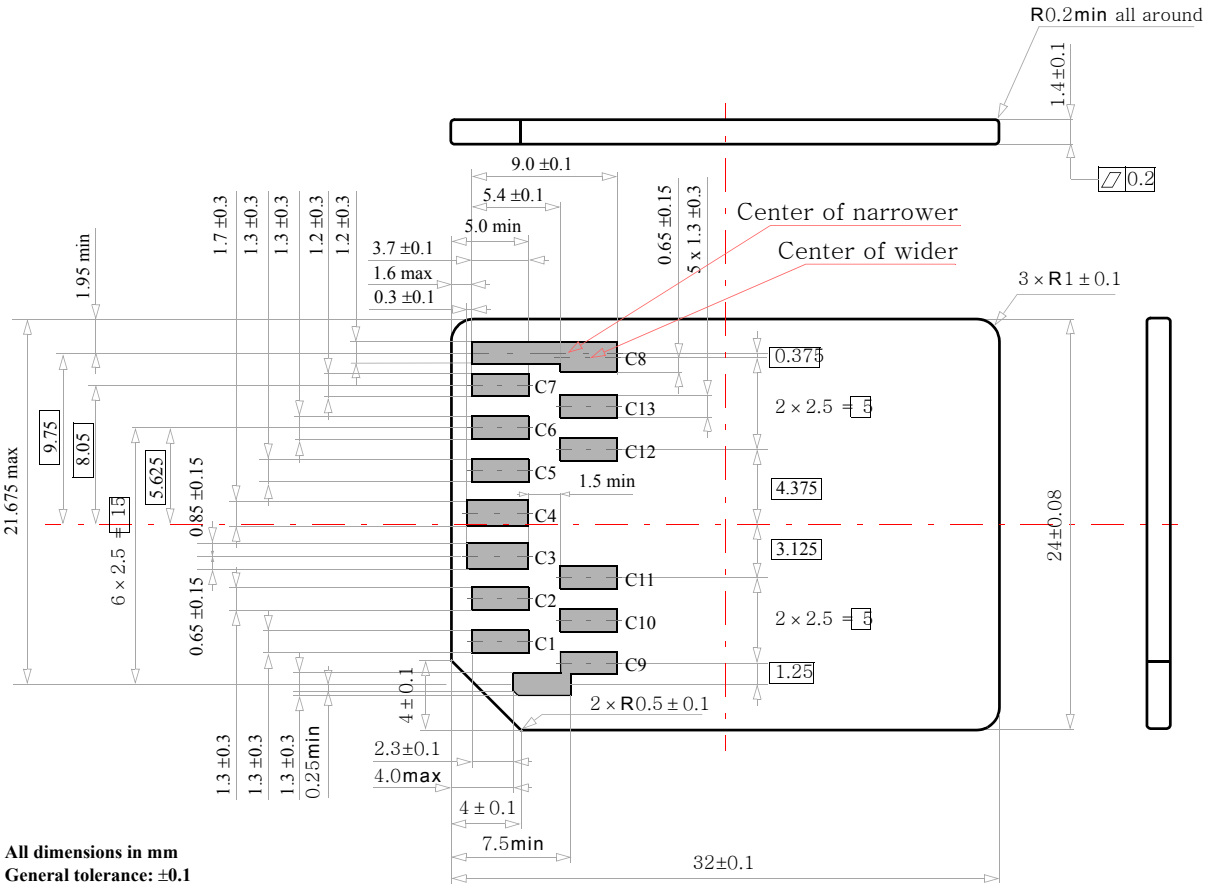
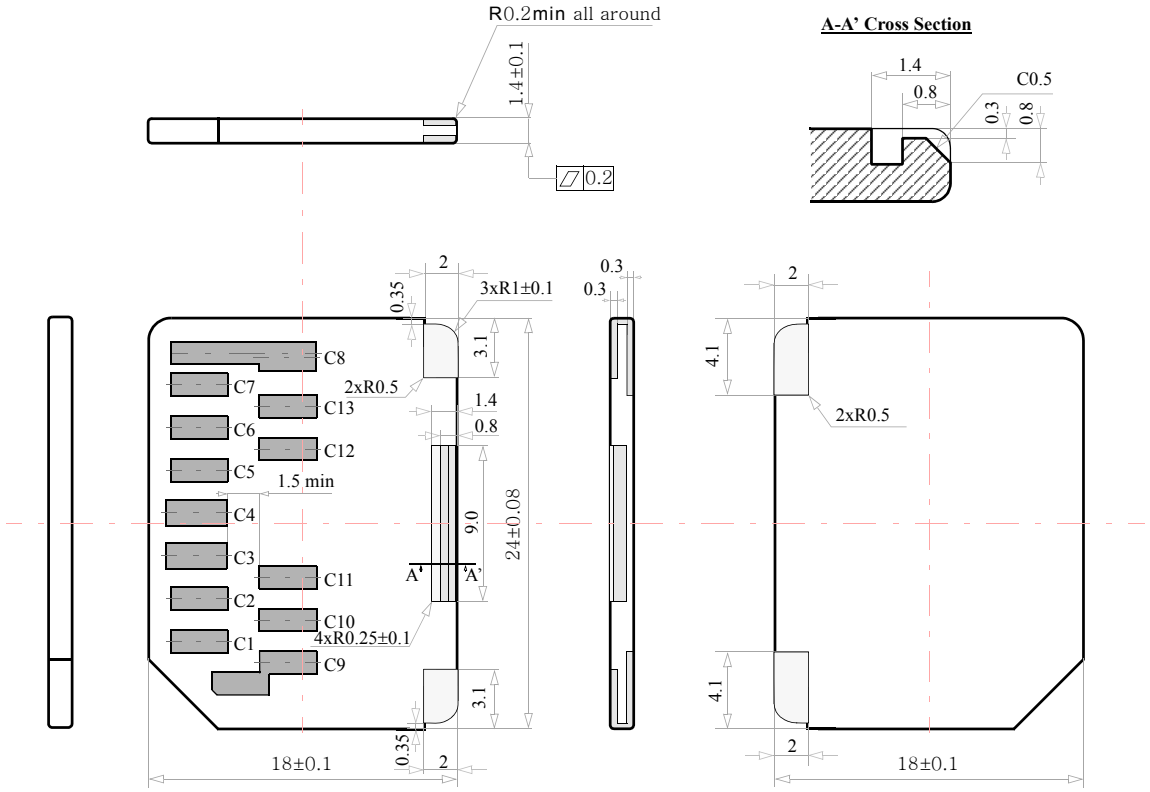


Figure 4-1 : Dimensions Of A Normal Size MultiMediaCard (Bottom View, DIN)



All dimensions in mm
General tolerance: ±0.1

Figure 4-3 : Dimensions Of A Reduced Size MultiMediaCard (DIN)

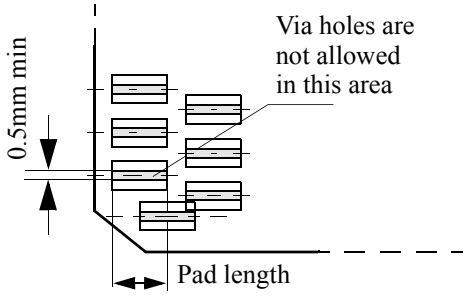


Figure 4-4 : Location Of Pads' Via Holes

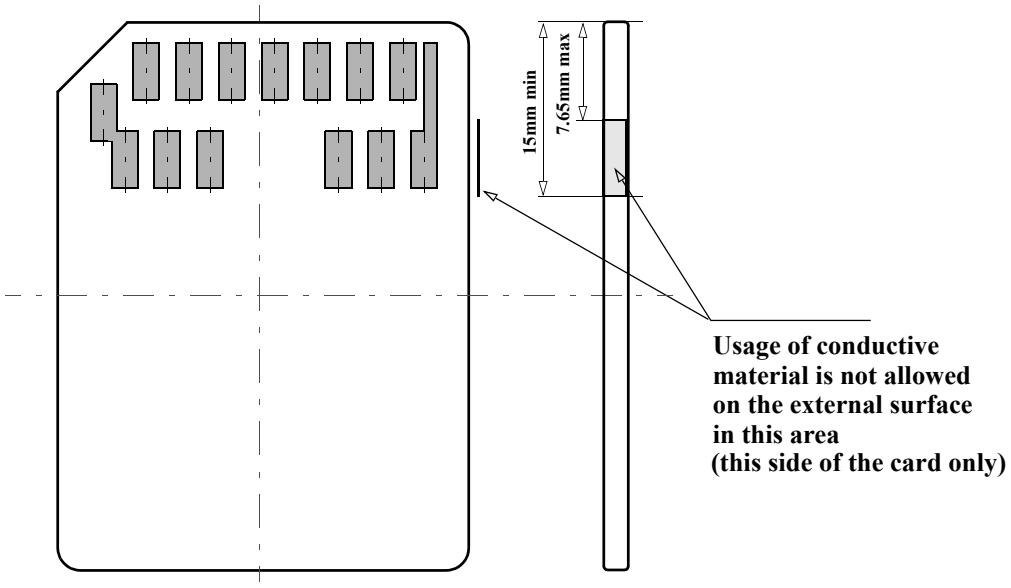


Figure 4-2 : Conductive Material Usage Restrictions

5. Interface Description

5.1 Physical Description

The MultiMediaCard has thirteen exposed contacts on one side. The host is connected to the car using a dedicated thirteen-pin connector.

5.1.1 Pin Assignments

Pin No.	Name	Type[a]	Description
MultiMediaCard Mode			
1	DAT3	I/O/PP	Data
2	CMD	I/O/PP/OD	Command/Response
3	V_{SS1}	S	Supply voltage ground
4	V_{DD}	S	Supply voltage
5	CLK	I	Clock
6	V_{SS2}	S	Supply voltage ground
7	DAT0[b]	I/O/PP	Data
8	DAT1	I/O/PP	Data
9	DAT2	I/O/PP	Data
10	DAT4	I/O/PP	Data
11	DAT5	I/O/PP	Data
12	DAT6	I/O/PP	Data
13	DAT7	I/O/PP	Data

a.S: power supply; I: input; O: output; PP: push-pull; OD: open-drain; NC: Not connected (or logical high)
b.The DAT0-DAT7 lines for read-only cards are output only

SPI Mode			
1	CS	I	Chip Select (neg true)
2	DI	I/PP	Data In
3	VSS	S	Supply voltage ground
4	VDD	S	Supply voltage
5	SCLK	I	Clock
6	VSS2	S	Supply voltage ground
7	DO	O/PP	Data Out
8	Not used		
9	Not used		
10	Not used		
11	Not used		
12	Not used		
13	Not used		

Table 5-1 : MultiMediaCard Interface Pin Configuration

5.2 MultiMediaCard Bus Topology

The MultiMediaCard bus has ten communication lines and three supply lines:

- CMD: Command is a bidirectional signal. The host and card drivers are operating in two modes, open drain and push pull.
- DAT0-7: Data lines are bidirectional signals. Host and card drivers are operating in push pull mode
- CLK: Clock is a host to card signal. CLK operates in push pull mode
- V_{DD}: V_{DD} is the power supply line for all cards.
- V_{SS1}, V_{SS2} are two ground lines.

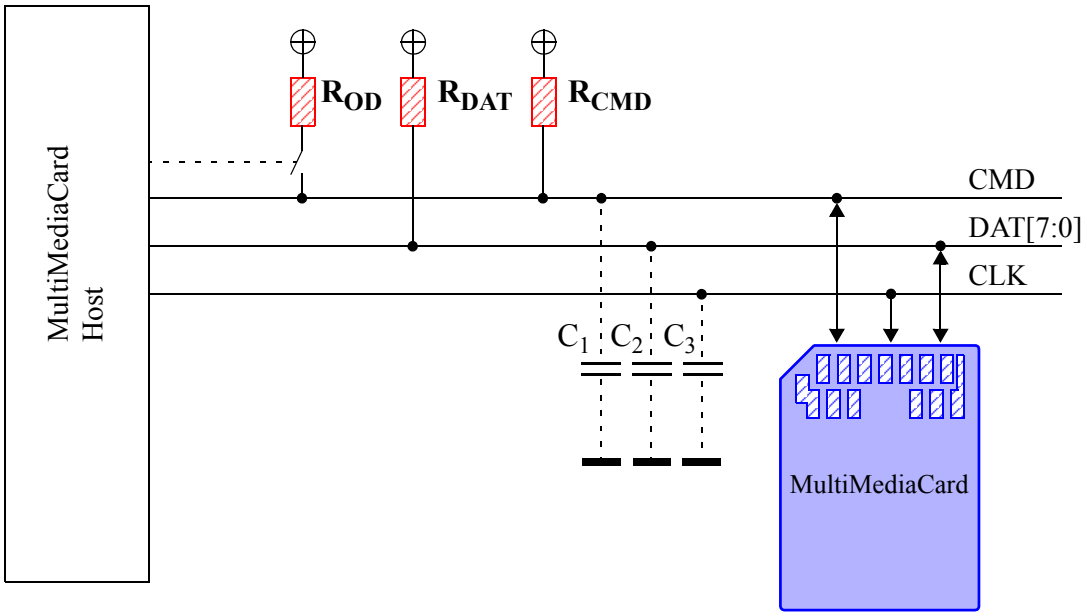


Figure 5-1 : Bus Circuitry Diagram

The R_{OD} is switched on and off by the host synchronously to the open-drain and push-pull mode transitions. The host does not have to have open drain drivers, but must recognize this mode to switch on the R_{OD}. R_{DAT} and R_{CMD} are pull-up resistors protecting the CMD and the DAT lines against bus floating when no card is inserted or when all card drivers are in a high-impedance mode.
A constant current source can replace the R_{OD} by achieving a better performance (constant slopes for the signal rising and falling edges). If the host does not allow the switchable R_{OD} implementation, a fixed R_{CMD} can be used (the minimum value is defined in the Chapter). Consequently the maximum operating frequency in the open drain mode has to be reduced if the used R_{CMD} value is higher than the minimal one given in Table 5-5.

5.2.1 Hot Insertion and Removal

To guarantee the proper sequence of card pin connection during hot insertion, the use of either a special hot-insertion capable card connector or an auto-detect loop on the host side (or some similar mechanism) is mandatory (see Chapter 4) No card shall be damaged by inserting or removing a card into the MultiMediaCard bus even when the power (V_{DD}) is up. Data transfer operations are protected by CRC codes, therefore any bit changes induced by card insertion and removal can be detected by the MultiMediaCard bus master.
The inserted card must be properly reset also when CLK carries a clock frequency f_{PP}. Each card shall have power protection to prevent card (and host) damage. Data transfer failures induced by removal/insertion are detected by the bus master. They must be corrected by the application, which may repeat the issued command.

5.2.2 Power Protection

Cards shall be inserted/removed into/from the bus without damage. If one of the supply pins (V_{DD} or V_{SS}) is not connected properly, then the current is drawn through a data line to supply the card. Every card's output also shall be able to withstand shortcuts to either supply. If hot insertion feature is implemented in the host, than the host has to withstand a shortcut between V_{DD} and V_{SS} without damage.

5.3 SPI Bus Topology

The SPI mode consists of a secondary, optional communication protocol which is offered by Flash-based MultiMediaCards. This mode is a subset of the MultiMediaCard protocol, designed to communicate with a SPI channel. The MultiMediaCard SPI interface is compatible with SPI hosts available on the market. As in any other SPI device, the MultiMediaCard SPI channel consists of the following four signals:

CS: Host to card Chip Select signal
CLK: Host to card clock signal
DataIn: Host to card data signal
DataOut: Card to host data signal

Another SPI common characteristic is byte transfers, which is implemented in the card as well. All data tokens are multiples of bytes (8 bit) and always byte aligned to the CS signal.The SPI standard defines the physical link only, and not the complete data transfer protocol.

The card identification and addressing methods are replaced by a hardware Chip Select (CS) signal. There are no broadcast commands. For every command, a card (slave) is selected by asserting (active low) the CS signal (see Figure).

The CS signal must be continuously active for the duration of the SPI transaction (command, response and data). The only exception occurs during card programming, when the host can de-assert the CS signal without affecting the programming process.

The bidirectional CMD and DAT lines are replaced by unidirectional *dataIn* and *dataOut* signals. This eliminates the ability to excute commands while data is being read or written which prevents swquential multi read/write operations. Only single block read/write is supported by the SPI channel.

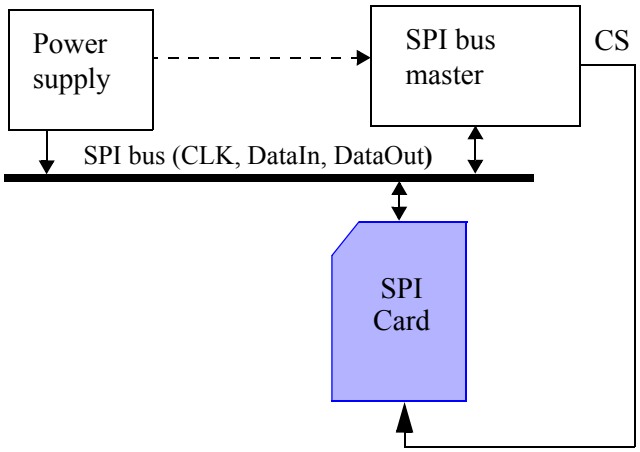


Figure 5-2 : MultiMediaCard Bus System

5.3.1 Power Protection

Power protection is the same as it is in MultiMediaCard mode.

5.4 Electrical Interface

The following sections provide valuable information about the electrical interface.

5.4.1 Power Up

The power-up of the MultiMediaCard bus is handled locally in each card and the bus master. See Figure 5-3

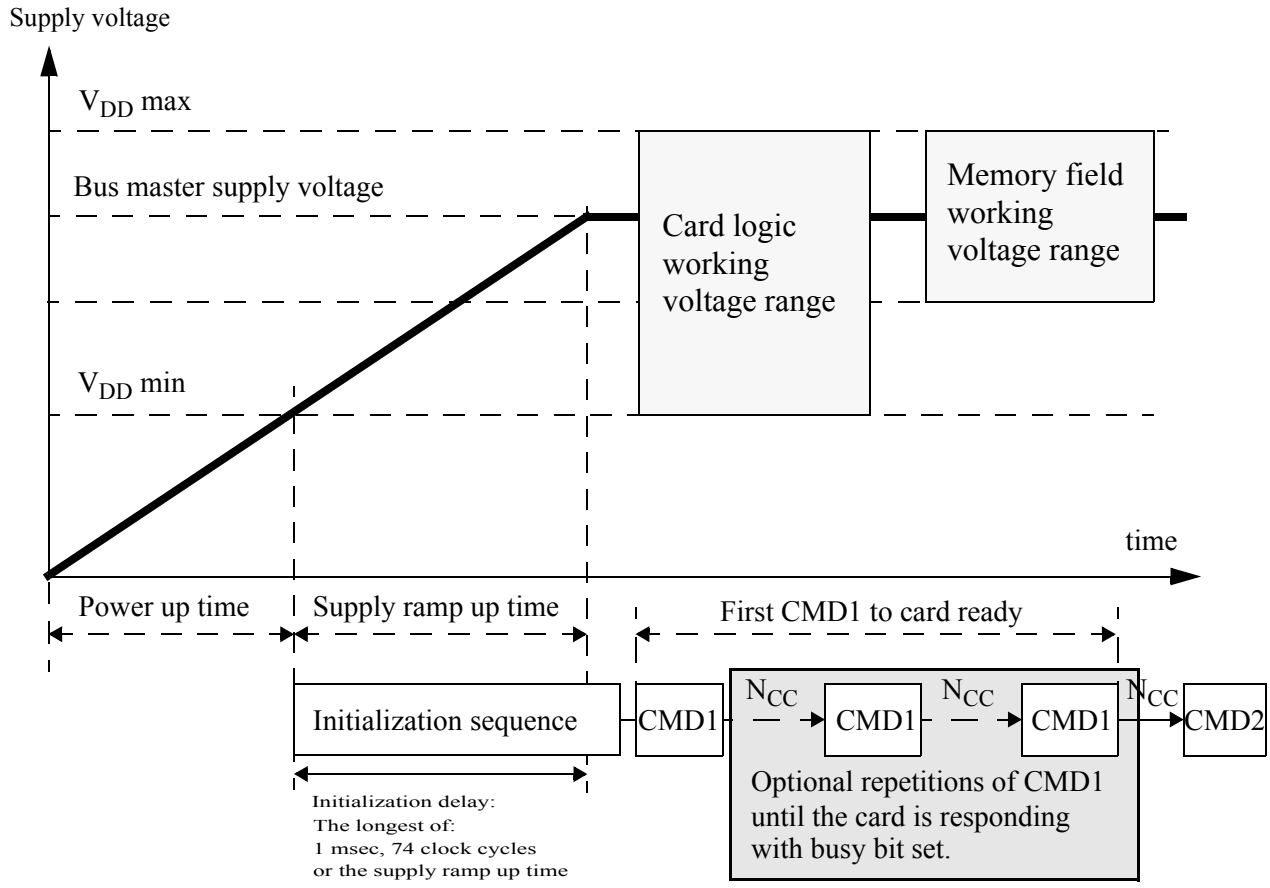


Figure 5-3 : Power-up Diagram

- After power up (including hot insertion, i.e. inserting a card when the bus is operating) the card enters the *idle state*. During this state the card ignores all bus transactions until CMD1 is received.
- The maximum initial load (after power up or hot insertion) that the MultiMediaCard can present on the VDD line shall be a maximum of 10 uF in parallel with a minimum of 330 ohms. At no time during operation shall the card capacitance on the VDD line exceed 10 uF
- CMD1 is a special synchronization command used to negotiate the operation voltage range and to poll the card until it is out of its power-up sequence. Besides the operation voltage profile of the card, the response to CMD1 contains a busy flag, indicating that the card is still working on its power-up procedure and is not ready for identification. This bit informs the host that the card is not ready. The host has to wait until this bit is cleared. The card shall complete its initialization within 1 second from the first CMD1 with a valid OCR range.
- Getting the card out of *idle state* is up to the responsibility of the bus master. Since the power up time and the supply ramp up time depend on application parameters as the bus length and the power supply unit, the host must ensure that the power is built up to the operating level (the same level which will be specified in CMD1) before CMD1 is transmitted.
- After power up the host starts the clock and sends the initializing sequence on the CMD line. This sequence is a contiguous stream of logical '1's. The sequence length is the longest of: 1msec, 74 clocks or the supply-ramp-up-time; The additional 10 clocks (over the 64 clocks after what the card should be ready for communication) is provided to eliminate power-up synchronization problems.

Every bus master has to implement CMD1. The CMD1 implementation is mandatory for all MultiMediaCards.

5.4.2 Bus Operating Conditions

General

Parameter	Symbol	Min	Max.	Unit	Remark
Peak voltage on all lines		-0.5	3.6	V	
All Inputs					
Input Leakage Current		-10	10	μA	
All Outputs					
Output Leakage Current		-10	10	μA	

Table 5-2 : Bus Opeating Conditions

Power Supply Voltage - High Voltage MultiMediaCard

Parameter	Symbol	Min	Max.	Unit	Remark
Supply voltage	V_{DD}	2.7	3.6	V	
Supply voltage differentials (V_{SS1}, V_{SS2})		-0.5	0.5	V	

Table 5-3 : Power Supply Voltage

Power Supply Voltage - Dual voltage MultiMediaCard

Parameter	Symbol	Min	Max.	Unit	Remark
Supply voltage (low voltage range)	V_{DDL}	1.65	1.95	V	1.95V - 2.7V is not supported
Supply voltage (high voltage range)	V_{DDH}	2.7	3.6	V	
Supply voltage differentials (V_{SS1}, V_{SS2})		-0.5	0.5	V	

Table 5-4 : Power Supply Voltage

The current consumption of the card for the different card configurations is defined in the power class fields in the EXT_CSD register.
The current consumption of any card during the power-up procedure, while the host has not sent yet a valid OCR range, must not exceed 10 mA

5.4.3 Bus Signal Line Load

The total capacitance C_L of each line of the MultiMediaCard bus is the sum of the bus master capacitance C_{HOST}, the bus capacitance C_{BUS} itself and the capacitance C_{CARD} of the card connected to this line:
$C_L = C_{HOST} + C_{BUS} + C_{CARD}$
Requiring the sum of the host and bus capacitances not to exceed 20 pF:

Parameter	Symbol	Min	Max.	Unit	Remark
Pull-up resistance for CMD	R_{CMD}	4.7	100	KOhm	to prevent bus floating
Pull-up resistance for DAT0-7	R_{DAT}	50	100	KOhm	to prevent bus floating
Bus signal line capacitance	C_L		30	pF	Single card
Single card capacitance	C_{CARD}		7	pF	
Maximum signal line inductance			16	nH	f_{PP} <= 52 MHz

Table 5-5 : Host and Bus Capacities

5.4.4 Bus Signal Levels

As the bus can be supplied with a variable supply voltage, all signal levels are related to the supply voltage.

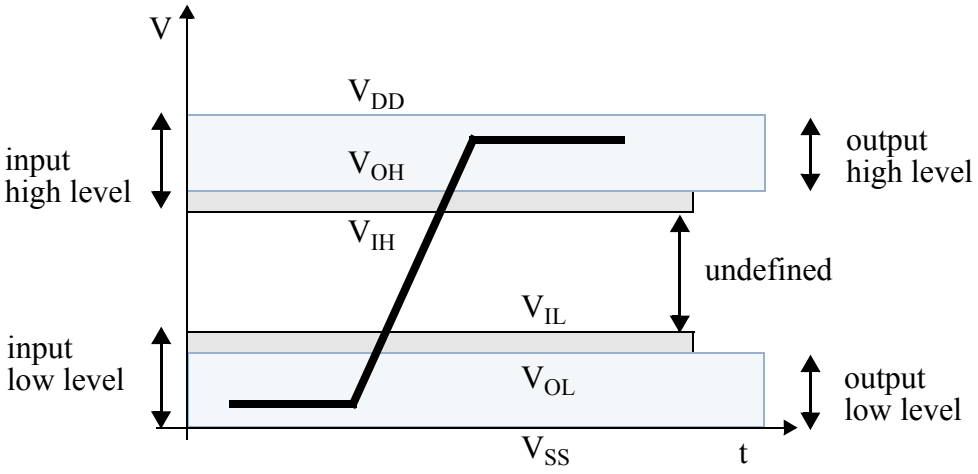


Figure 5-4 : Bus Signal Levels

5.4.5 Open-Drain Mode Bus Signal Level

Parameter	Symbol	Min	Max.	Unit	Conditions
Output HIGH voltage	V_{OH}	V_{DD}-0.2		V	I_{OH} = -100 µA
Output LOW voltage	V_{OL}		0.3	V	I_{OL} = 2 mA

Table 5-6 : Open Drain Mode Bus Signal Levels

The input levels are identical with the push-pull mode bus signal levels.

5.4.6 Push-Pull Mode Bus Signal Level - High Voltage MultiMediaCard

To meet the requirements of the JEDEC specification JESD8-1A, the card input and output voltages shall be within the following specified ranges for any V_{DD} of the allowed voltage range:

Parameter	Symbol	Min	Max.	Unit	Conditions
Output HIGH voltage	V_{OH}	0.75*V_{DD}		V	I_{OH}=-100 µA @V_{DD} min
Output LOW voltage	V_{OL}		0.125*V_{DD}	V	I_{OL}=100 µA @V_{DD} min
Input HIGH voltage	V_{IH}	0.625*V_{DD}	V_{DD} + 0.3	V	
Input LOW voltage	V_{IL}	VSS-0.3	0.25*V_{DD}	V	

Table 5-7 : Push-Pull Mode Bus Signal Level - High Voltage Mululti MediaCard

5.4.7 Push-Pull Mode Bus Signal Level - Dual Voltage MultiMediaCard

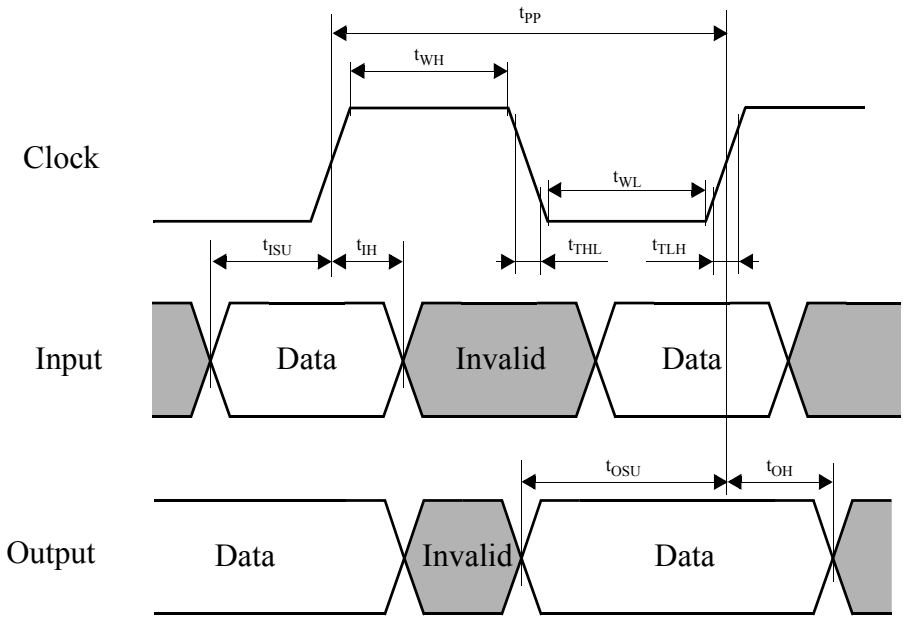
The definition of the I/O signal levels for the Dual voltage MultiMediaCard changes as a function of V_{DD}

2.7V - 3.6V: Identical to the High Voltage MultiMediaCard (refer to Chapter above)
1.95 - 2.7V: Undefined. The card is not operating at this voltage range
1.65 - 1.95V: Compatible with EIA/JEDEC Standard “EIA/JESD8-7 Wide Range” as defined in the following table

Parameter	Symbol	Min	Max.	Unit	Conditions
Output HIGH voltage	V_{OH}	V_{DD} - 0.2V		V	I_{OH}=-100 μA @V_{DD} min
Output LOW voltage	V_{OL}		0.2V	V	I_{OL}=100 μA @V_{DD} min
Input HIGH voltage	V_{IH}	0.7 * V_{DD}	V_{DD} + 0.3	V	
Input LOW voltage	V_{IL}	VSS-0.3	0.3 * V_{DD}	V	

Table 5-8 : Push-Pull Mode Bus Signal Level - Dual Voltage MultiMediCard

5.4.8 Bus & Card Interface Timing



Data must always be sampled on the rising edge of the clock.

Figure 5-5 : Timing Diagram: Data Input/Output

Parameter	Symbol	Min	Max.	Unit	Remark
Clock CLK[a]					
Clock frequency Data Transfer Mode (PP)[b]	f_{PP}	0	26/52	MHz	C_L <= 30 pF Tolerance: +100KHz
Clock frequency Identification Mode (OD)	f_{OD}	0	400	kHz	Tolerance: +20KHz
Clock low time	t_{WL}	6.5		ns	C_L <= 30 pF
Clock rise time[c]	t_{TLH}		3	ns	C_L <= 30 pF
Clock fall time	t_{THL}		3	ns	C_L <= 30 pF
Inputs CMD, DAT (referenced to CLK)					
Input set-up time	t_{ISU}	3		ns	C_L <= 30 pF
Input hold time	t_{IH}	3		ns	C_L <= 30 pF
Outputs CMD, DAT (referenced to CLK)					
Output set-up time	t_{OSU}	5		ns	C_L <= 30 pF
Output hold time	t_{OH}	5		ns	C_L <= 30 pF
Signal rise time[d]	t_{rise}		3	ns	C_L <= 30 pF
Signal fall time	t_{fall}		3	ns	C_L <= 30 pF

a.All timing values are measured relative to 50% of voltage level
b.A MultiMediaCard shall support the full frequency range from 0-26Mhz, or 0-52MHz
c.Rise and fall times are measured from 10%-90% of voltage level
d.Rise and fall times are measured from 10%-90% of voltage level

Table 5-9 : High Speed Card Interface Timings

Parameter	Symbol	Min	Max.	Unit	Remark
Clock CLK[a]					
Clock frequency Data Transfer Mode (PP)	f_{PP}	0	20	MHz	C_L <= 30 pF
Clock frequency Identification Mode (OD)	f_{OD}	0	400	kHz	
Clock low time	t_{WL}	10		ns	C_L <= 30 pF
Clock rise time[b]	t_{TLH}		10	ns	C_L <= 30 pF
Clock fall time	t_{THL}		10	ns	C_L <= 30 pF
Inputs CMD, DAT (referenced to CLK)					
Input set-up time	t_{ISU}	3		ns	C_L <= 30 pF
Input hold time	t_{IH}	3		ns	C_L <= 30 pF
Outputs CMD, DAT (referenced to CLK)					
Output set-up time	t_{OSU}	13.1		ns	C_L <= 30 pF
Output hold time	t_{OH}	9.7		ns	C_L <= 30 pF

a.All timing values are measured relative to 50% of voltage level
b.Clock rise and fall times are measured from VIL to VIH of voltage level

Table 5-10 : Backwards Compatible Card Interface Timing

5.5 MultiMediaCard Registers

Within the card interface six registers are defined: OCR, CID, CSD, EXT_CSD, RCA and DSR. These can be accessed only by corresponding commands (see Chapter 6.6). The OCR, CID and CSD registers carry the card/content specific information, while the RCA and DSR registers are configuration registers storing actual configuration parameters. The EXT_CSD register carries both, card specific information and actual configuration parameters.

5.5.1 OCR Register

The 32-bit operation conditions register stores the V_{DD} voltage profile of the card. In addition, this register includes a status information bit. This status bit is set if the card power up procedure has been finished. The OCR register shall be implemented by all cards.

OCR bit	VDD voltage window	High Voltage MultimediaCard	Dual voltage MultiMediaCard
[6:0]	Reserved	000 0000b	00 00000b
[7]	1.65 - 1.95	0b	1b
[14:8]	2.0-2.6	000 0000b	000 0000b
[23:15]	2.7-3.6	1 1111 1111b	1 1111 1111b
[30:24]	Reserved	000 0000b	000 0000b
[31]	card power up status bit (busy)[a]		

a.This bit is set to LOW if the card has not finished the power up routine

The supported voltage range is coded as shown in Table , for High Voltage and Dual voltage MultiMediaCards. As long as the card is busy, the corresponding bit (31) is set to LOW, the ‘wired-and’ operation, described in Chapter 6.1.2 yields LOW, if at least one card is still busy.

5.5.2 CID Register

The Card IDentification (CID) register is 128 bits wide. It contains the card identification information used during the card identification phase (MultiMediaCard protocol). Every individual flash or I/O card shall have an unique identification number. Every type of MultiMediaCard ROM cards (defined by content) shall have an unique identification number.
The structure of the CID register is defined in the following paragraphs:

Name	Field	Width	CID-slice	CID Value
Manufacturer ID	MID	8	[127:120]	0x15
OEM/Application ID	OID	16	[119:104]	---
Product name	PNM	48	[103:56]	---
Product revision	PRV	8	[55:48]	---
Product serial number	PSN	32	[47:16]	---
Manufacturing date	MDT	8	[15:8]	---
CRC7 checksum	CRC	7	[7:1]	---
not used, always ’1’	-	1	[0:0]	---

Table 5-12 : The CID fields

- **MID**

An 8 bit binary number that identifies the card manufacturer. The MID number is controlled, defined and allocated to a MultiMediaCard manufacturer by the MMCA. This procedure is established to ensure uniqueness of the CID register.

- **OID**

A 16 bit binary number that identifies the card OEM and/or the card contents (when used as a distribution media either on ROM or FLASH cards). The OID number is controlled, defined and allocated to a MultiMediaCard manufacturer by the MMCA. This procedure is established to ensure uniqueness of the CID register.

- **PNM**

The product name is a string, 6 ASCII characters long.

- **PRV**

The product revision is composed of two Binary Coded Decimal (BCD) digits, four bits each, representing an “n.m” revision number. The “n” is the most significant nibble and “m” is the least significant nibble.
As an example, the PRV binary value field for product revision “6.2” will be: 0110 0010

- **PSN**

A 32 bits unsigned binary integer.

- **MDT**

The manufacturing date is composed of two hexadecimal digits, four bits each, representing a two digits date code m/y;
The “m” field, most significant nibble, is the month code. 1 = January.
The “y” field, least significant nibble, is the year code. 0 = 1997.
As an example, the binary value of the MDT field for production date “April 2000” will be: 0100 0011

- **CRC**

CRC7 checksum (7 bits). This is the checksum of the CID contents computed according to Chapter 6.4.

5.5.3 CSD Register

The Card-Specific Data register provides information on how to access the card contents. The CSD defines the data format, error correction type, maximum data access time, data transfer speed, whether the DSR register can be used etc. The programmable part of the register (entries marked by W or E, see below) can be changed by CMD27. The type of the entries in the table below is coded as follows:

R = readable, W = writable once, E = erasable (multiple writable).

Name	Field	Width	Cell Type	CSD-slice	CSD Value
CSD structure	CSD_STRUCTURE	2	R	[127:126]	v.1.2
System specification version	SPEC_VERS	4	R	[125:122]	v.4.1
Reserved	-	2	R	[121:120]	0
Data read access-time 1	TAAC	8	R	[119:112]	1.5ms
Data read access-time 2 in CLK cycles (NSAC*100)	NSAC	8	R	[111:104]	100
Max. bus clock frequency	TRAN_SPEED	8	R	[103:96]	20MHz
Card command classes	CCC	12	R	[95:84]	Not support class:1,3,8,9,10,11
Max. read data block length	READ_BL_LEN	4	R	[83:80]	512Bytes
Partial blocks for read allowed	READ_BL_PARTIAL	1	R	[79:79]	No
Write block misalignment	WRITE_BLK_MISALIGN	1	R	[78:78]	No
Read block misalignment	READ_BLK_MISALIGN	1	R	[77:77]	No
DSR implemented	DSR_IMP	1	R	[76:76]	No
Reserved	-	2	R	[75:74]	0
Device size	C_SIZE	12	R	[73:62]	---
Max. read current @ V_{DD} min	VDD_R_CURR_MIN	3	R	[61:59]	60mA
Max. read current @ V_{DD} max	VDD_R_CURR_MAX	3	R	[58:56]	80mA
Max. write current @ V_{DD} min	VDD_W_CURR_MIN	3	R	[55:53]	60mA
Max. write current @ V_{DD} max	VDD_W_CURR_MAX	3	R	[52:50]	80mA
Device size multiplier	C_SIZE_MULT	3	R	[49:47]	---
Erase group size	ERASE_GRP_SIZE	5	R	[46:42]	---
Erase group size multiplier	ERASE_GRP_MULT	5	R	[41:37]	---
Write protect group size	WP_GRP_SIZE	5	R	[36:32]	---
Write protect group enable	WP_GRP_ENABLE	1	R	[31:31]	1
Manufacturer default ECC	DEFAULT_ECC	2	R	[30:29]	None
Write speed factor	R2W_FACTOR	3	R	[28:26]	---
Max. write data block length	WRITE_BL_LEN	4	R	[25:22]	512
Partial blocks for write allowed	WRITE_BL_PARTIAL	1	R	[21:21]	No
Reserved	-	4	R	[20:17]	0
Content protection application	CONTENT_PROT_APP	1	R	[16:16]	---
File format group	FILE_FORMAT_GRP	1	R/W	[15:15]	0
Copy flag (OTP)	COPY	1	R/W	[14:14]	---
Permanent write protection	PERM_WRITE_PROTECT	1	R/W	[13:13]	No
Temporary write protection	TMP_WRITE_PROTECT	1	R/W/E	[12:12]	No
File format	FILE_FORMAT	2	R/W	[11:10]	0
ECC code	ECC	2	R/W/E	[9:8]	None
CRC	CRC	7	R/W/E	[7:1]	
Not used, always ’1’	-	1	-	[0:0]	1

The following sections describe the CSD fields and the relevant data types. If not explicitly defined otherwise, all bit strings are interpreted as binary coded numbers starting with the left bit first.

- **CSD_STRUCTURE**

Describes the version of the CSD structure.

CSD_STRUCTURE	CSD structure version	Valid for System Specification Version
0	CSD version No. 1.0	Version 1.0 - 1.2
1	CSD version No. 1.1	Version 1.4 - 2.2
2	CSD version No. 1.2	Version 3.1 - 3.2 - 3.31 - 4.0 - 4.1
3	Version is coded in the CSD_STRUCTURE byte in the EXT_CSD register	

- **SPEC_VERS**

Defines the MultiMediaCard System Specification version supported by the card.

SPEC_VERS	System Specification Version Number
0	Version 1.0-1.2
1	Version 1.4
2	Version 2.0 - 2.2
3	Version 3.1 - 3.2 -3.31
4	Version 4.0 - 4.1
5 - 15	Reserved

- **TAAC**

Defines the asynchronous part of the data access time.

TAAC bit position	code
2:0	Time unit 0=1ns, 1=10ns, 2=100ns, 3=1µs, 4=10µs, 5=100µs, 6=1ms, 7=10ms
6:3	Multiplier factor 0=reserved, 1=1.0, 2=1.2, 3=1.3, 4=1.5, 5=2.0, 6=2.5, 7=3.0, 8=3.5, 9=4.0, A=4.5, B=5.0, C=5.5, D=6.0, E=7.0, F=8.0
7	Reserved

- **NSAC**

Defines the typical case for the clock dependent factor of the data access time. The unit for NSAC is 100 clock cycles. Therefore, the maximal value for the clock dependent part of the data access time is 25.5k clock cycles.
The total access time N_{AC} as expressed in Table 23 is calculated based on TAAC and NSAC. It has to be computed by the host for the actual clock rate. The read access time should be interpreted as a typical delay for the first data bit of a data block or stream.

- **TRAN_SPEED**

The following table defines the clock frequency when not in high speed mode. For cards supporting version 4.0, and higher, of the specification, the value shall be 20MHz (0x2A):

TRAN_SPEED bit	Code
2:0	Frequency unit 0=100KHz, 1=1MHz, 2=10MHz, 3=100MHz, 4...7=reserved
6:3	Multiplier factor 0=reserved, 1=1.0, 2=1.2, 3=1.3, 4=1.5, 5=2.0, 6=2.6, 7=3.0, 8=3.5, 9=4.0, A=4.5, B=5.2, C=5.5, D=6.0, E=7.0, F=8.0
7	reserved

- **CCC**

The MultiMediaCard command set is divided into subsets (command classes). The card command class register CCC defines which command classes are supported by this card. A value of '1' in a CCC bit means that the corresponding command class is supported. For command class definition refer to Table 6-8.

CCC bit	Supported card command class
0	class 0
1	class 1
.....	
11	class 11

- **READ_BL_LEN**

The data block length is computed as $2^{READ\_BL\_LEN}$. The block length might therefore be in the range 1, 2,4...2048 bytes (see Chapter 6.10 for details):

READ_BL_LEN	Block length
0	2^0 = 1 Byte
1	2^1 = 2 Bytes
.....	
11	2^{11} = 2048 Bytes
12-15	Reserved

- **READ_BL_PARTIAL**

Defines whether partial block sizes can be used in block read commands.
READ_BL_PARTIAL=0 means that only the READ_BL_LEN block size can be used for block oriented data transfers.
READ_BL_PARTIAL=1 means that smaller blocks can be used as well. The minimum block size will be equal to minimum addressable unit (one byte)

- **WRITE_BLK_MISALIGN**

Defines if the data block to be written by one command can be spread over more than one physical block of the memory device. The size of the memory block is defined in WRITE_BL_LEN.
WRITE_BLK_MISALIGN=0 signals that crossing physical block boundaries is invalid.
WRITE_BLK_MISALIGN=1 signals that crossing physical block boundaries is allowed.

- **READ_BLK_MISALIGN**

Defines if the data block to be read by one command can be spread over more than one physical block of the memory device. The size of the memory block is defined in READ_BL_LEN.
READ_BLK_MISALIGN=0 signals that crossing physical block boundaries is invalid.
READ_BLK_MISALIGN=1 signals that crossing physical block boundaries is allowed.

- **DSR_IMP**

Defines if the configurable driver stage is integrated on the card. If set, a driver stage register (DSR) must be implemented also (see Chapter 5.5.6).

DSR_IMP	DSR type
0	DSR is not implemented
1	DSR implemented

- **C_SIZE**

This parameter is used to compute the card capacity. The memory capacity of the card is computed from the entries C_SIZE, C_SIZE_MULT and READ_BL_LEN as follows:
memory capacity = BLOCKNR * BLOCK_LEN
where

BLOCKNR = (C_SIZE+1) * MULT

MULT = $2^{C\_SIZE\_MULT+2}$ (C_SIZE_MULT < 8)

BLOCK_LEN = $2^{READ\_BL\_LEN}$, (READ_BL_LEN < 12)

Therefore, the maximal capacity which can be coded is 4096*512*2048 = 4 GBytes. Example: A 4 MByte card with BLOCK_LEN = 512 can be coded by C_SIZE_MULT = 0 and C_SIZE = 2047.

- **VDD_R_CURR_MIN, VDD_W_CURR_MIN**

The maximum values for read and write currents at the minimal power supply V_{DD} are coded as follows:

VDD_R_CURR_MIN **VDD_W_CURR_MIN**	Code for current consumption @ V_{DD}
2:0	0=0.5mA; 1=1mA; 2=5mA; 3=10mA; 4=25mA; 5=35mA; 6=60mA; 7=100mA

The values in these fields are valid when the card is not in high speed mode. When the card is in high speed mode, the current consumption is chosen by the host, from the power classes defined in the PWR_ff_vvv registers, in the EXT_CSD register.

- **VDD_R_CURR_MAX, VDD_W_CURR_MAX**

The maximum values for read and write currents at the maximal power supply V_{DD} are coded as follows:

VDD_R_CURR_MAX VDD_W_CURR_MAX	Code for current consumption @ V_{DD}
2:0	0=1mA; 1=5mA; 2=10mA; 3=25mA; 4=35mA; 5=45mA; 6=80mA; 7=200mA

The values in these fields are valid when the card is not in high speed mode. When the card is in high speed mode, the current consumption is chosen by the host, from the power classes defined in the PWR_ff_vvv registers, in the EXT_CSD register.

- **C_SIZE_MULT**

This parameter is used for coding a factor MULT for computing the total device size (see 'C_SIZE'). The factor MULT is defined as $2^{C\_SIZE\_MULT+2}$.

C_SIZE_MULT	MULT
0	$2^2 = 4$
1	$2^3 = 8$
2	$2^4 = 16$
3	$2^5 = 32$
4	$2^6 = 64$
5	$2^7 = 128$
6	$2^8 = 256$
7	$2^9 = 512$

- **ERASE_GRP_SIZE**

The contents of this register is a 5 bit binary coded value, used to calculate the size of the erasable unit of the card. The size of the erase unit (also referred to as erase group) is determined by the ERASE_GRP_SIZE and the ERASE_GRP_MULT entries of the CSD, using the following equation:
size of erasable unit = (ERASE_GRP_SIZE + 1) * (ERASE_GRP_MULT + 1)
This size is given as minimum number of write blocks that can be erased in a single erase command.

- **ERASE_GRP_MULT**

A 5 bit binary coded value used for calculating the size of the erasable unit of the card. See ERASE_GRP_SIZE section for detailed description.

- **WP_GRP_SIZE**

The size of a write protected group. The contents of this register is a 5 bit binary coded value, defining the number of erase groups which can be write protected. The actual size is computed by increasing this number by one. A value of zero means 1 erase group, 31 means 32 erase groups.

- **WP_GRP_ENABLE**

A value of ‘0’ means no group write protection possible.

- **DEFAULT_ECC**

Set by the card manufacturer. It defines the ECC code which is recommended for use. The field definition is the same as for the ECC field described later.

- **R2W_FACTOR**

Defines the typical block program time as a multiple of the read access time. The following table defines the field format.

R2W_FACTOR	Multiples of read access time
0	1
1	2 (write half as fast as read)
2	4
3	8
4	16
5	32
6	64
7	128

- **WRITE_BL_LEN**

Block length for write operations. See READ_BL_LEN for field coding.

- **WRITE_BL_PARTIAL**

Defines whether partial block sizes can be used in block write commands.
WRITE_BL_PARTIAL=’0’ means that only the WRITE_BL_LEN block size can be used for block oriented data write.
WRITE_BL_PARTIAL=’1’ means that smaller blocks can be used as well. The minimum block size is one byte.

- **FILE_FORMAT_GRP**

Indicates the selected group of file formats. This field is read-only for ROM. The usage of this field is shown in Table 5-25 (see FILE_FORMAT).

- **COPY**

Defines if the contents is original (= ‘0’) or has been copied (=’1’). The COPY bit for OTP and MTP devices, sold to end consumers, is set to ‘1’ which identifies the card contents as a copy. The COPY bit is an one time programmable bit.

- **PERM_WRITE_PROTECT**

Permanently protects the whole card content against overwriting or erasing (all write and erase commands for this card are permanently disabled). The default value is ‘0’, i.e. not permanently write protected.

- **TMP_WRITE_PROTECT**

Temporarily protects the whole card content from being overwritten or erased (all write and erase commands for this card are temporarily disabled). This bit can be set and reset. The default value is '0', i.e. not write protected.

- **CONTENT_PROT_APP**

This field in the CSD indicates whether the content protection application is supported. MultiMediaCards which implement the content protection application will have this bit set to '1';

- **FILE_FORMAT**

Indicates the file format on the card. This field is read-only for ROM. The following formats are defined:

FILE_FORMAT_GRP	FILE_FORMAT	Type
0	0	Hard disk-like file system with partition table
0	1	DOS FAT (floppy-like) with boot sector only (no partition table)
0	2	Universal File Format
0	3	Others / Unknown
1	0, 1, 2, 3	Reserved

A more detailed description is given in "File Formats Specifications For MultiMediaCards".

- **ECC**

Defines the ECC code that was used for storing data on the card. This field is used by the host (or application) to decode the user data. The following table defines the field format.:

ECC	ECC type	Maximum number of correctable bits per block
0	None (default)	none
1	BCH (542,512)	3
2-3	reserved	-

- **CRC**

The CRC field carries the check sum for the CSD contents. It is computed according to Chapter 6.4. The checksum has to be recalculated by the host for any CSD modification. The default corresponds to the initial CSD contents.

The following table lists the correspondence between the CSD entries and the command classes. A '+' entry indicates that the CSD field affects the commands of the related command class.

	Command classes									
CSD Field	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
CSD_STRUCTURE	+	+	+	+	+	+	+	+	+	+
SPEC_VERS	+	+	+	+	+	+	+	+	+	+
TAAC		+	+	+	+	+	+	+	+	
NSAC		+	+	+	+	+	+	+	+	
TRAN_SPEED		+	+	+	+					
CCC	+	+	+	+	+	+	+	+	+	+
READ_BL_LEN			+							
READ_BL_PARTIAL			+							
WRITE_BLK_MISALIGN					+					
READ_BLK_MISALIGN			+							
DSR_IMP	+	+	+	+	+	+	+	+	+	+
C_SIZE_MANT		+	+	+	+	+	+	+	+	
C_SIZE_EXP		+	+	+	+	+	+	+	+	
VDD_R_CURR_MIN		+	+							
VDD_R_CURR_MAX		+	+							
VDD_W_CURR_MIN				+	+	+	+	+	+	
VDD_W_CURR_MAX				+	+	+	+	+	+	
ERASE_GRP_SIZE						+	+	+	+	
WP_GRP_SIZE							+	+	+	
WP_GRP_ENABLE							+	+	+	
DEFAULT_ECC		+	+	+	+	+	+	+	+	
R2W_FACTOR				+	+	+	+	+	+	
WRITE_BL_LEN				+	+	+	+	+	+	
WRITE_BL_PARTIAL				+	+	+	+	+	+	
FILE_FORMAT_GRP										
COPY	+	+	+	+	+	+	+	+	+	+
PERM_WRITE_PROTECT	+	+	+	+	+	+	+	+	+	+
TMP_WRITE_PROTECT	+	+	+	+	+	+	+	+	+	+
FILE_FORMAT										
ECC		+	+	+	+	+	+	+	+	
CRC	+	+	+	+	+	+	+	+	+	+

5.5.4 Extended CSD Register

The Extended CSD register defines the card properties and selected modes. It is 512 bytes long.
The most significant 320 bytes are the Properties segment, which defines the card capabilities and cannot be modified by the host. The lower 192 bytes are the Modes segment, which defines the configuration the card is working in. These modes can be changed by the host by means of the SWITCH command

Name	Field	Size (Bytes)	Cell Type	CSD-slice
Properties Segment				
Reserved[1]		7		[511:505]
Supported Command Sets	S_CMD_SET	1	R	[504]
Reserved[1]		293		[503:211]
Minimum Write Performance for 8bit @52MHz	MIN_PERF_W_8_52	1	R	[210]
Minimum Read Performance for 8bit @52MHz	MIN_PERF_R_8_52	1	R	[209]
Minimum Write Performance for 8bit @26MHz / 4bit @52MHz	MIN_PERF_W_8_26_4_52	1	R	[208]
Minimum Read Performance for 8bit @26MHz / 4bit @52MHz	MIN_PERF_R_8_26_4_52	1	R	[207]
Minimum Write Performance for 4bit @26MHz	MIN_PERF_W_4_26	1	R	[206]
Minimum Read Performance for 4bit @26MHz	MIN_PERF_R_4_26	1	R	[205]
Reserved[1]		1		[204]
Power Class for 26MHz @ 3.6V	PWR_CL_26_360	1	R	[203]
Power Class for 52MHz @ 3.6V	PWR_CL_52_360	1	R	[202]
Power Class for 26MHz @ 1.95V	PWR_CL_26_195	1	R	[201]
Power Class for 52MHz @ 1.95V	PWR_CL_52_195	1	R	[200]
Reserved[1]		3		[199:197]
Card Type	CARD_TYPE	1	R	[196]
Reserved[1]		1		[195]
CSD Structure Version	CSD_STRUCTURE	1	R	[194]
Reserved[1]		1		[193]
Extended CSD Revision	EXT_CSD_REV	1	R	[192]
Modes Segment				
Command Set	CMD_SET	1	R/W	[191]
Reserved[1]		1		[190]
Command Set Revision	CMD_SET_REV	1	RO	[189]
Reserved[1]		1		[188]
Power Class	POWER_CLASS	1	R/W	[187]
Reserved[1]		1		[186]
High Speed Interface Timing	HS_TIMING	1	R/W	[185]
Reserved[1]		1		[184]
Bus Width Mode	BUS_WIDTH	1	WO	[183]
Reserved[a]		183		[182:0]

a.Reserved bits should read as ‘0’

Table 5-27 : Extended CSD

- **S_CMD_SET**

This field defines which command sets are supported by the card.

Bit	Command Set
7-3	Reserved
2	Content Protection SecureMMC
1	SecureMMC
0	Standard MMC

- **MIN_PERF_a_b_ff**

These fields defines the overall minimum performance value for the read and write access with different bus width and max clock frequency modes. The value in the register is coded as follows. Other than defined values are illegal.

Value	Performance
0x00	For Cards not reaching the 2.4MB/s minimum value
0x08	Class A: 2.4MB/s and is the lowest allowed value for MMCplus and MMCmobile(16x150kB/s)
0x0A	Class B: 3.0MB/s and is the next allowed value (20x150kB/s)
0x0F	Class C: 4.5MB/s and is the next allowed value (30x150kB/s)
0x14	Class D: 6.0MB/s and is the next allowed value (40x150kB/s)
0x1E	Class E: 9.0MB/s and is the next allowed value (60x150kB/s) This is also the highest class which any MMCplus or MMCmobile card is needed to support in low bus category operation mode (26MHz with 4bit data bus). A MMCplus or MMCmobile card supporting any higher class than this have to support this class also (in low category bus operation mode).
0x28	Class F: Equals 12.0MB/s and is the next allowed value (80x150kB/s)
0x32	Class G: Equals 15.0MB/s and is the next allowed value (100x150kB/s)
0x3C	Class H: Equals 18.0MB/s and is the next allowed value (120x150kB/s)
0x46	Class J: Equals 21.0MB/s and is the next allowed value (140x150kB/s) This is also the highest class which any MMCplus or MMCmobile card is needed to support in mid bus category operation mode (26MHz with 8bit data bus or 52MHz with 4bit data bus). A MMCplus or MMCmobile card supporting any higher class than this have to support this Class (in mid category bus operation mode) and Class E also (in low category bus operation mode)
0x50	Class K: Equals 24.0MB/s and is the next allowed value (160x150kB/s)
0x64	Class M: Equals 30.0MB/s and is the next allowed value (200x150kB/s)
0x78	Class O: Equals 36.0MB/s and is the next allowed value (240x150kB/s)
0x8C	Class R: Equals 42.0MB/s and is the next allowed value (280x150kB/s)
0xA0	Class T: Equals 48.0MB/s and is the last defined value (320x150kB/s)

Table 5-28 : R/W Access Performance Value

- **PWR_CL_ff_vvv**

These fields define the supported power classes by the card. By default, the card has to operate at maximum frequency using 1 bit bus configuration, within the default max current consumption, as stated in the table below. If 4 bit/8 bits bus configurations, require increased current consumption, it has to be stated in these registers.
By reading these registers the host can determine the power consumption of the card in different bus modes. Bits [7:4] code the current consumption for the 8 bit bus configuration. Bits [3:0] code the current consumption for the 4 bit bus configuration
The PWR_52_vvv registers are not defined for 26MHz MultiMediaCards.

Voltage	Value	Max RMS Current	Max Peak Current	Remarks
3.6V	0	100 mA	200 mA	Default current consumption for high voltage cards
	1	120 mA	220 mA	
	2	150 mA	250 mA	
	3	180 mA	280 mA	
	4	200 mA	300 mA	
	5	220 mA	320 mA	
	6	250 mA	350 mA	
	7	300 mA	400 mA	
	8	350 mA	450 mA	
	9	400 mA	500 mA	
	10	450 mA	550 mA	
	11-15			Reserved for future use
1.95V	0	65 mA	130 mA	Default current consumption for Dual voltage cards
	1	70 mA	140 mA	
	2	80 mA	160 mA	
	3	90 mA	180 mA	
	4	100 mA	200 mA	
	5	120 mA	220 mA	
	6	140 mA	240 mA	
	7	160 mA	260 mA	
	8	180 mA	280 mA	
	9	200 mA	300 mA	
	10	250 mA	350 mA	
	6-15			Reserved for future use

The measurement for max RMS current is done as average RMS current consumption over a period of 100ms.
Max peak current is defined as absolute max value not to be exceeded at all.
The conditions under which the power classes are defined are:

- Maximum bus frequency
- Maximum operating voltage
- Worst case functional operation
- Worst case environmental parameters (temperature,...)

These registers define the maximum power consumption for any protocol operation in data transfer mode, Ready state and Identification state.

- **CARD_TYPE**

This field defines the type of the card. The only currently valid values for this field are 0x01 and 0x03.

Bit	Card Type
7:2	Reserved
1	High Speed MultiMediaCard @ 52MHz
0	High Speed MultiMediaCard @ 26MHz

- **CSD_STRUCTURE**

This field is a continuation of the CSD_STRUCTURE field in the CSD register

CSD_STRUCTURE	CSD structure version	Valid for System Specification Version
0	CSD version No. 1.0	Version 1.0 - 1.2
1	CSD version No. 1.1	Version 1.4 - 2.2
2	CSD version No. 1.2	Version 3.1 - 3.2 - 3.31 - 4.0 - 4.1
3	Reserved for future use	
4-255	Reserved for future use	

- **EXT_CSD_REV**

Defines the fixed parameters. related to the EXT_CSD, according to its revision

EXT_CSD_REV	Extended CSD Revision
255-2	Reserved
1	Revision 1.1
0	Revision 1.0

- **CMD_SET**

Contains the binary code of the command set that is currently active in the card. It is set to ‘0’ (Standard MMC) after power up and can be changed by a SWITCH command.

- **CMD_SET_REV**

Contains a binary number reflecting the revision of the currently active command set. For Standard MMC. command set it is:

Code	MMC Revisions
255-1	Reserved
0	v4.0

This field, though in the Modes segment of the EXT_CSD, is read only.

- **POWER_CLASS**

This field contains the 4 bit value of the selected power class for the card. The power classes are defined in Table . The host should be responsible of properly writing this field with the maximum power class it allows the card to use. The card uses this information to, internally, manage the power budget and deliver an optimized performance.
This field is 0 after power-on or software reset.

Bits	Description
[7:4]	Reserved
[3:0]	Card power class code (See Table 5-29)

- **HS_TIMING**

This field is 0 after power-on, or software reset, thus selecting the backwards compatibility interface timing for the card. If the host writes 1 to this field, the card changes its timing to high speed interface timing (refer to Chapter 5.4.8)

- **BUS_WIDTH**

It is set to '0' (1 bit data bus) after power up and can be changed by a SWITCH command.

Value	Bus Mode
255-3	Reserved
2	8 bit data bus
1	4 bit data bus
0	1 bit data bus

5.5.5 RCA Register

The writable 16-bit relative card address register carries the card address assigned by the host during the card identification. This address is used for the addressed host-card communication after the card identification procedure. The default value of the RCA register is 0x0001. The value 0x0000 is reserved to set all cards into the *Stand-by State* with CMD7.

5.5.6 DSR Register

The 16-bit driver stage register can be optionally used to improve the bus performance for extended operating conditions (depending on parameters like bus length, transfer rate or number of cards). The CSD register carries the information about the DSR register usage. The default value of the DSR register is 0x404.

6. MultiMediaCard Protocol Description

All communication between host and card is controlled by the host (master). The host sends commands of two types: broadcast and addressed (point-to-point) commands.

- Broadcast commands : Broadcast commands are intended for all cards in a MultiMediaCard system[1]. Some of these commands require a response.
- Addressed (point-to-point) commands : The addressed commands are sent to the addressed card and cause a response from this card.

A general overview of the command flow is shown in Figure 6-1 for the card identification mode and in Figure 6-2 for the data transfer mode. The commands are listed in the command tables (Table 6-9 - Table 6-17). The dependencies between current state, received command and following state are listed in Table 6-18. In the following sections, the different card operation modes are described first. Thereafter, the restrictions for controlling the clock signal are defined. All MultiMediaCard commands together with the corresponding responses, state transitions, error conditions and timings are presented in the succeeding sections.

Three operation modes are defined for the MultiMediaCard system (hosts and cards):

- Card identification mode
  The host will be in card identification mode after reset, while it is looking for a card on the bus. The card will be in this mode after reset, until the SET_RCA command (CMD3) is received.
- Interrupt mode
  Host and card enter and exit interrupt mode simultaneously. In interrupt mode there is no data transfer. The only message allowed is an interrupt service request from the card or the host.
- Data transfer mode
  The card will enter data transfer mode once an RCA is assigned to it. The host will enter data transfer mode after identifying the card on the bus.

The following table shows the dependencies between bus modes, operation modes and card states. Each state in the MultiMediaCard state diagram (see Figure 6-1 and Figure 6-2) is associated with one bus mode and one operation mode:

Card state	Operation mode	Bus mode
Inactive State	Inactive	Open-drain
Idle State	Card identification mode	
Ready State		
Identification State		
Stand-by State	Data transfer mode	Push-pull
Transfer State		
Bus-Test State		
Sending-data State		
Receive-data State		
Programming State		
Disconnect State		
Wait-IRQ State	Interrupt mode	Open-drain

1.Broadcast commands are kept for backwards compatibility to previous MultiMediaCard systems, where more than one card was allowed on the bus.

6.1 Card Identification Mode

While in card identification mode the host resets the card, validates operation voltage range, identifies the card and assigns a Relative Card Address (RCA) to the card on the bus. All data communication in the Card Identification Mode uses the command line (CMD) only.

6.1.1 Card Reset

After power-on by the host, the cards (even if it has been in *Inactive State)* is in MultiMediaCard mode (as opposed to SPI mode) and in *Idle State.*
Command GO_IDLE_STATE (CMD0) is the software reset command and puts the card into *Idle State.* It is also used to switch the card into SPI mode (refer to Chapter 7 for details).
After power-on, or CMD0, the cards' output bus drivers are in high-impedance state and the card is initialized with a default relative card address („0x0001“) and with a default driver stage register setting (lowest speed, highest driving current capability). The host clocks the bus at the identification clock rate f_{OD} (see Chapter 5.4.8).
CMD0 is valid in all states, with the exception of *Inactive* State. While in *Inactive* state the card does not accept CMD0, unless it is used to switch the card into SPI mode.

6.1.2 Operating Voltage Range Validation

Each type of MultiMediaCard (either High voltage or Dual Voltage) shall be able to establish communication with the host, as well as perform the actual card function (e.g. accessing memory), using any operating voltage within the voltage range specified in this standard, for the given card type (See Chapter 5.4.2).
The SEND_OP_COND (CMD1) command is designed to provide MultiMediaCard hosts with a mechanism to identify and reject cards which do not match the V_{DD} range desired by the host. This is accomplished by the host sending the required V_{DD} voltage window as the operand of this command (See Chapter 5.5.1). If the card can not perform data transfer in the specified range it must discard itself from further bus operations and go into *Inactive State*. Otherwise, the card shall respond sending back its V_{DD} range. For this, the levels in the OCR register shall be defined accordingly (see Chapter 5.5.1).

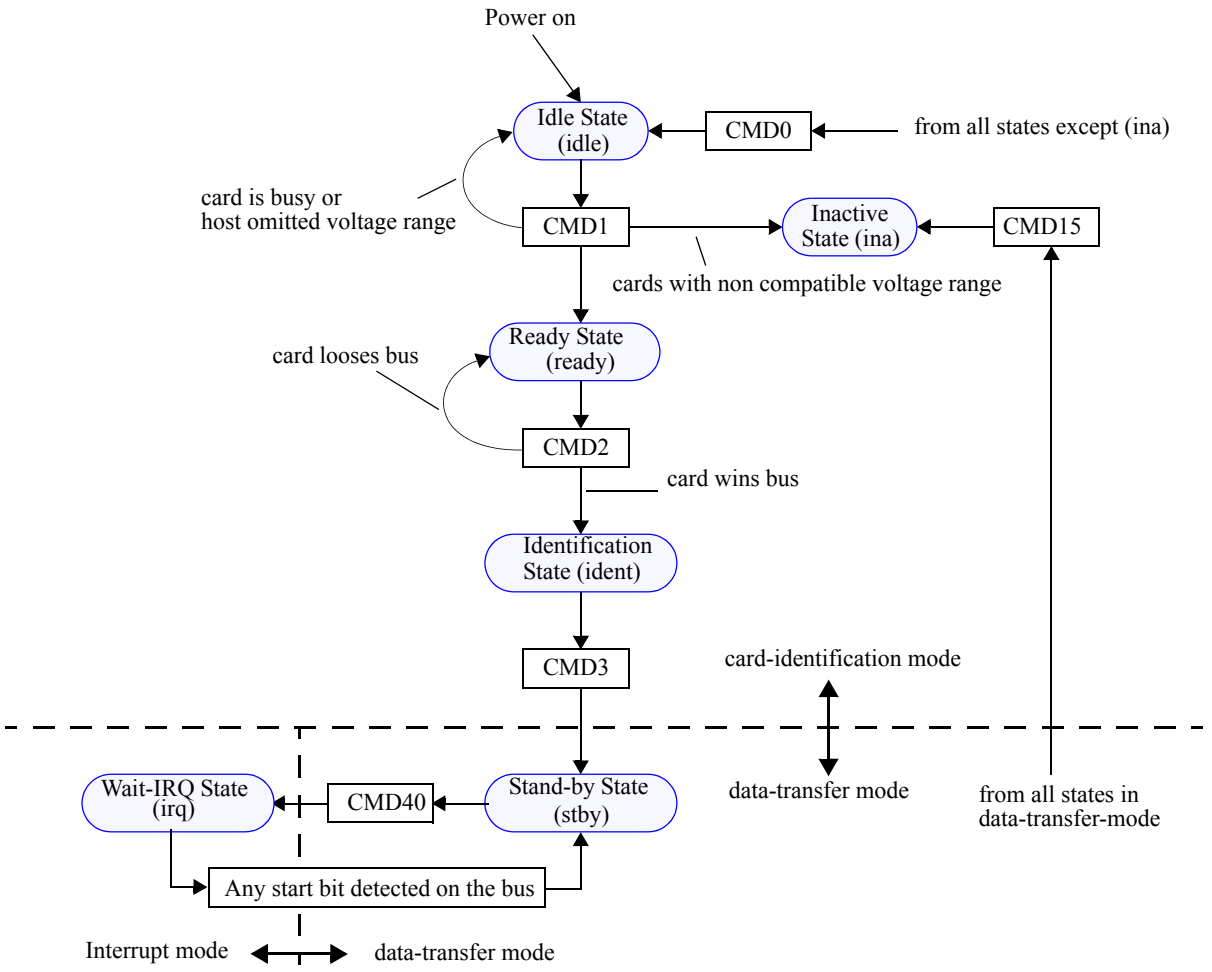


Figure 6-1 : MultiMediaCard State Diagram (Card Identification Mode)

By omitting the voltage range in the command (by setting the argument of CMD1 to 0), the host can query the card and determine the voltage type of the card. This bus query should be used if the host is able to select a common voltage range, or if a notification to the application of a non usable card in the bus is desired. Afterwards, the host must choose a voltage for operation, and reissue CMD1 with this condition, sending incompatible cards into the *Inactive State*.
The busy bit in the CMD1 response can be used by a card to tell the host that it is still working on its power-up/reset procedure (e.g. downloading the register information from memory field) and is not ready yet for communication. In this case the host must repeat CMD1 until the busy bit is cleared.
During the initialization procedure, the host is not allowed to change the operating voltage range. Such changes shall be ignored by the card. If there is a real change in the operating conditions, the host must reset the card (using CMD0) and restart the initialization procedure. However, for accessing cards already in *Inactive State*, a hard reset must be done by switching the power supply off and back on.
The command GO_INACTIVE_STATE (CMD15) can be used to send an addressed card into the *Inactive State*. This command is used when the host explicitly wants to deactivate a card (e.g. host is changing V_{DD} into a range which is known to be not supported by this card).

The command CMD1 shall be implemented by all cards defined by this standard.
If the host intends to operate the Dual Voltage MultiMediaCards in the 1.65V to 1.95V range, it is recommended that the host first validate the operating voltage in the 2.7V to 3.6V range, then power the card down fully, and finally power the card back up to the 1.65V to 1.95V range for operation. Using the 2.7V to 3.6V range initially, which is common to High and Dual voltage MultiMediaCards, will allow reliable screening of host & card voltage incompatibilities. High voltage cards may not function properly if VDD < 2.0V is used to establish communication. Dual voltage cards may fail if 1.95 to 2.7V is used.

6.1.3 Card Identification Process

The following explanation refers to a card working in a multi-card environment, as defined in versions of this standard previous to v4.0, and it is maintained for backwards compatibility to those systems.
The host starts the card identification process in open-drain mode with the identification clock rate f_{OD} (see Chapter 5.4.8). The open drain driver stages on the CMD line allow parallel card operation during card identification.
After the bus is activated, the host will request the cards to send its valid operation conditions (CMD1). The response to CMD1 is the 'wired and' operation on the condition restrictions of all cards in the system. Incompatible cards are sent into *Inactive State*. The host then issues the broadcast command ALL_SEND_CID (CMD2), asking all cards for its unique card identification (CID) number. All unidentified cards (i.e. those which are in *Ready State*) simultaneously start sending their CID numbers serially, while bit-wise monitoring their outgoing bitstream. Those cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods, stop sending their CID immediately and must wait for the next identification cycle (remaining in the *Ready State*). Since CID numbers are unique for each card, there should be only one card which successfully sends its full CID-number to the host. This card then goes into *Identification State*. Thereafter, the host issues CMD3 (SET_RELATIVE_ADDR) to assign to this card a relative card address (RCA), which is shorter than CID and which will be used to address the card in the future data transfer mode (typically with a higher clock rate than f_{OD}). Once the RCA is received the card state changes to the *Stand-by State,* and the card does not react to further identification cycles. Furthermore, the card switches its output drivers from open-drain to push-pull.
The host repeats the identification process, i.e. the cycles with CMD2 and CMD3 as long as it receives a response (CID) to its identification command (CMD2). If no more card responds to this command, all cards have been identified. The time-out condition to recognize completion of the identification process is the absence of a start bit for more than N_{ID} clock cycles after sending CMD2 (see timing values in Chapter 6.11).

6.2 Data Transfer Mode

When the card is in *Stand-by State,* communication over the CMD and DAT lines will be performed in push-pull mode. Until the contents of the CSD register is known by the host, the f_{PP} clock rate must remain at f_{OD} (see Chapter 5.4.8). The host issues SEND_CSD (CMD9) to obtain the Card Specific Data (CSD register), e.g. block length, card storage capacity, maximum clock rate, etc.

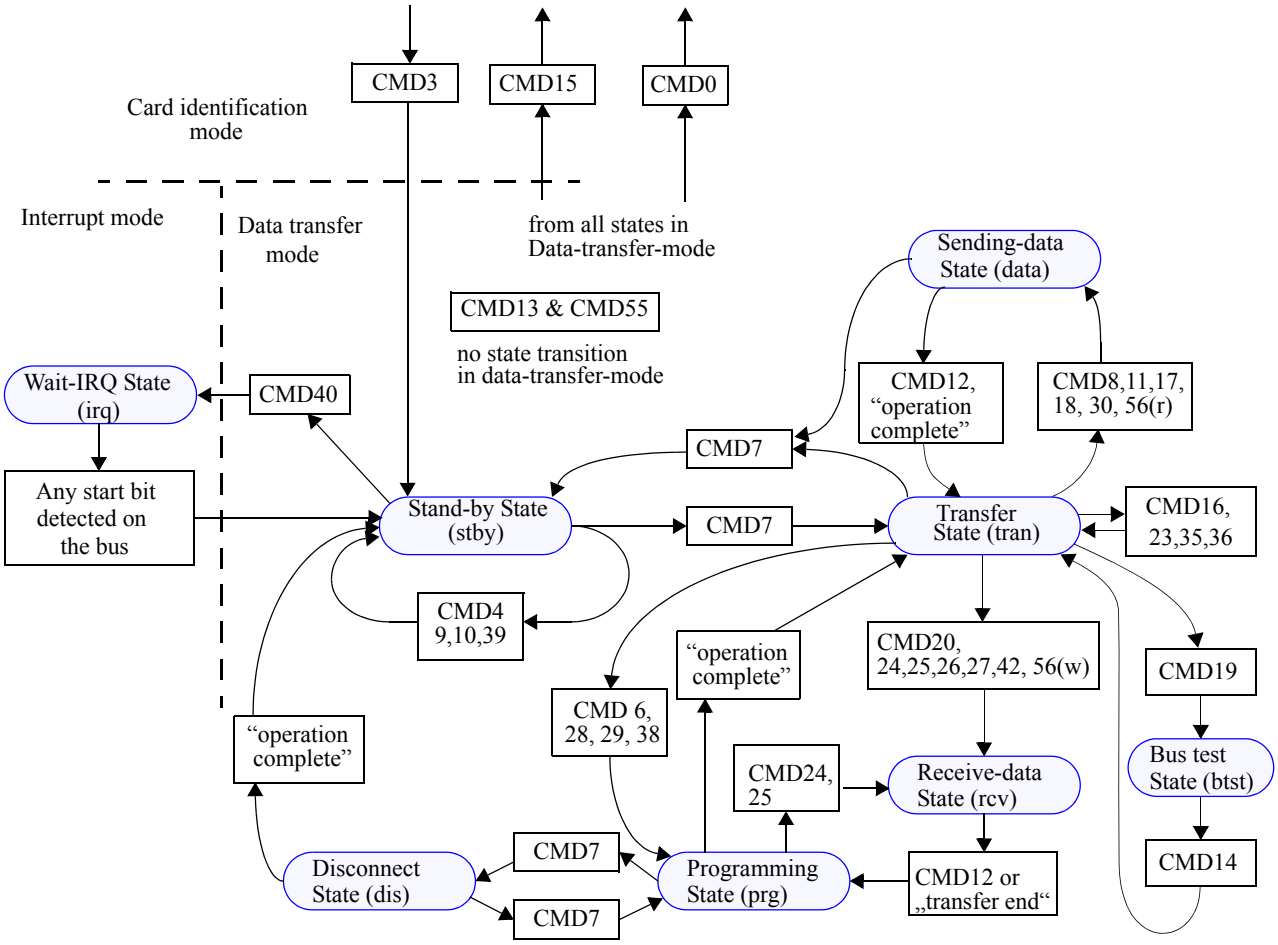


Figure 6-2 : MultiMediaCard State Diagram (Data Transfer Mode)

The broadcast command SET_DSR (CMD4) configures the driver stages of the card. It programs its DSR register corresponding to the application bus layout (length) and the data transfer frequency. The clock rate is also switched from f_{OD} to f_{PP} at that point.

CMD7 is used to select the card and put it into the *Transfer State*. If the card was previously selected and was in *Transfer State* its connection with the host is released and it will move back to the *Stand-by State*. When CMD7 is issued with the reserved relative card address "0x0000", the card is put back to *Stand-by State*. After the card is assigned an RCA it will not respond to identification commands (CMD1, CMD2, CMD3, see Chapter 6.1.3).

All data communication in the Data Transfer Mode is point-to point between the host and the selected card (using addressed commands). All addressed commands get acknowledged by a response on the CMD line.
The relationship between the various data transfer modes is summarized below (see Figure 6-2):

- All data read commands can be aborted any time by the stop command (CMD12). The data transfer will terminate and the card will return to the *Transfer State.* The read commands are: block read (CMD17), multiple block read (CMD18) and send write protect (CMD30).
- All data write commands can be aborted any time by the stop command (CMD12). The write commands must be stopped prior to deselecting the card by CMD7. The write commands are: block write (CMD24 and CMD25), write CID (CMD26), and write CSD (CMD27).
- As soon as the data transfer is completed, the card will exit the data write state and move either to the *Programming State* (transfer is successful) or *Transfer State* (transfer failed).
- If a block write operation is stopped and the block length and CRC of the last block are valid, the data will be pro grammed.
- The card may provide buffering for stream and block write. This means that the next block can be sent to the card while the previous is being programmed.
- If all write buffers are full, and as long as the card is in *Programming State* (see MultiMediaCard state diagram Figure -2), the DAT0 line will be kept low.
- There is no buffering option for write CSD, write CID, write protection and erase. This means that while the card is busy servicing any one of these commands, no other data transfer commands will be accepted. DAT0 line will be kept low as long as the card is busy and in the *Programming State*.
- Parameter set commands are *not* allowed while card is programming.
  Parameter set commands are: set block length (CMD16), and erase group selection (CMD35-36).
- Read commands are *not* allowed while card is programming.
- Moving another card from *Stand-by* to *Transfer State* (using CMD7) will not terminate a programming operation. The card will switch to the *Disconnect State* and will release the DAT0 line.
- A card can be reselected while in the *Disconnect State*, using CMD7. In this case the card will move to the *Program ming State* and reactivate the busy indication.
- Resetting a card (using CMD0 or CMD15) will terminate any pending or active programming operation. This may destroy the data contents on the card. It is up to the host's responsibility to prevent this.
- Prior to executing the bus testing procedure (CMD19, CMD14), it is recommended to set up the clock frequency used for data transfer. This way the bus test gives a true result, which might not be the case if the bus testing procedure is performed with lower clock frequency than the data transfer frequency.

In the following format definitions, all upper case flags and parameters are defined in the CSD (Chapter 5.5.3), and the other status flags in the Card Status (Chapter 6.9).

6.2.1 Command Sets And Extended Settings

The card operates in a given command set, by default, after a power cycle or reset by CMD0, it is the MultiMediaCard standard command set, using a single data line, DAT0. The host can change the active command set by issuing the SWITCH command (CMD6) with the 'Command Set' access mode selected.
The supported command sets, as well as the currently selected command set, are defined in the EXT_CSD register. The EXT_CSD register is divided in two segments, a Properties segment and a Modes segment. The Properties segment contains information about the card capabilities. The Modes segment reflects the current selected modes of the card.
The host reads the EXT_CSD register by issuing the SEND_EXT_CSD command. The card sends the EXT_CSD register as a block of data, 512 bytes long. Any reserved, or write only field, reads as '0'.
The host can write the Modes segment of the EXT_CSD register by issuing a SWITCH command and setting one of the access modes. All three modes access and modify one of the EXT_CSD bytes, the byte pointed by the Index field[1]

Access Bits	Access Name	Operation
00	Command Set	The command set is changed according to the Cmd Set field of the argument
01	Set Bits	The bits in the pointed byte are set, according to the '1' bits in the Value field.
10	Clear Bits	The bits in the pointed byte are cleared, according to the '1' bits in the Value field.
11	Write Byte	The Value field is written into the pointed byte.

Table 6-2 : EXT_CSD Access Modes

The SWITCH command can be used either to write the EXT_CSD register or to change the command set. If the SWITCH command is used to change the command set, the Index and Value field are ignored, and the EXT_CSD is not written. If the SWITCH command is used to write the EXT_CSD register, the Cmd Set field is ignored, and the command set remains unchanged.
The SWITCH command response is of type R1b, therefore, the host should read the card status, using SEND_STATUS command, after the busy signal is de-asserted, to check the result of the SWITCH operation.

6.2.2 High Speed Mode Selection

After the host verifies that the card complies with version 4.0, or higher, of this standard, it has to enable the high speed mode timing in the card, before changing the clock frequency to a frequency higher than 20MHz.
After power-on, or software reset, the interface timing of the card is set as specified in Table 5-7, Chapter 5. For the host to change to a higher clock frequency, it has to enable the high speed interface timing. The host uses the SWITCH command to write 0x01 to the HS_TIMING byte, in the Modes segment of the EXT_CSD register.
The valid values for this register are defined in 'HS_TIMING', in page 37. If the host tries to write an invalid value, the HS_TIMING byte is not changed, the high speed interface timing is not enabled, and the SWITCH_ERROR bit is set.

6.2.3 Power Class Selection

After the host verifies that the card complies with version 4.0, or higher, of this standard, it may change the power class of the card.
After power-on, or software reset, the card power class is class 0, which is the default, minimum current consumption class for the card type, either High Voltage or Dual voltage card. The PWR_CL_ff_vvv bytes, in the EXT_CSD register, reflect the power consumption levels of the card, for a 4 bits bus, an 8 bit bus, at the supported clock frequencies (26MHZ or 52MHz).
The host reads this information, using the SEND_EXT_CSD command, and determines if it will allow the card to use a higher power class. If a power class change is needed, the host uses the SWITCH command to write the POWER_CLASS byte, in the Modes segment of the EXT_CSD register.
The valid values for this register are defined in 'PWR_CL_ff_vvv', in page 84 If the host tries to write an invalid value, the POWER_CLASS byte is not changed and the SWITCH_ERROR bit is set.

1.The Index field can contain any value from 0-255, but only values 0-191 are valid values. If the Index value is in the 192-255 range the card does not perform any modification and the SWITCH_ERROR status bit is set.

6.2.4 Bus Testing Procedure

By issuing commands CMD19 and CMD14 the host can detect the functional pins on the bus. In a first step, the host sends CMD19 to the card, followed by a specific data pattern on each selected data lines. The data pattern to be sent per data line is defined in the table below. As a second step, the host sends CMD14 to request the card to send back the reversed data pattern. With the data pattern sent by the host and with the reversed pattern sent back by the card, the functional pins on the bus can be detected.

Start Bit	Data Pattern	End bit
0	1 0 x x x x ... x x	1

The card ignores all but the two first bits of the data pattern. Therefore, the card buffer size is not limiting the maximum length of the data pattern. The minimum length of the data pattern is two bytes, of which the first two bits of each data line are sent back, by the card, reversed. The data pattern sent by the host may optionally include a CRC16 checksum, which is ignored by the card.
The card detects the start bit on DAT0 and synchronizes accordingly the reading of all its data inputs.
The host ignores all but the two first bits of the reverse data pattern. The length of the reverse data pattern is eight bytes and is always sent using all the card's DAT lines (See Table through Table). The reverse data pattern sent by the card may optionally include a CRC16 checksum, which is ignored by the host.
The card has pull ups in all data inputs. In cases where the card is connected to only 1bit or only 4bit HS-MMC system, the input value of the upper bits (e.g. DAT1-DAT7 or DAT4-DAT7) are detected as logical "1" by the card. .

Data line	Data pattern sent by the host	Reversed pattern sent by the card	Notes
DAT0	0,10xxxxxxxxx,[CRC16],1	0,01000000,[CRC16],1	Start bit defines beginning of pattern
DAT1		0,00000000,[CRC16],1	No data pattern sent
DAT2		0,00000000,[CRC16],1	No data pattern sent
DAT3		0,00000000,[CRC16],1	No data pattern sent
DAT4		0,00000000,[CRC16],1	No data pattern sent
DAT5		0,00000000,[CRC16],1	No data pattern sent
DAT6		0,00000000,[CRC16],1	No data pattern sent
DAT7		0,00000000,[CRC16],1	No data pattern sent

Table 6-3 : 1-bit Bus Testing Pattern

Data line	Data pattern sent by the host	Reversed pattern sent by the card	Notes
DAT0	0,10xxxxxxxxx,[CRC16],1	0,01000000,[CRC16],1	Start bit defines beginning of pattern
DAT1	0,01xxxxxxxxx,[CRC16],1	0,10000000,[CRC16],1	
DAT2	0,10xxxxxxxxx,[CRC16],1	0,01000000,[CRC16],1	
DAT3	0,01xxxxxxxxx,[CRC16],1	0,10000000,[CRC16],1	
DAT4		0,00000000,[CRC16],1	No data pattern sent
DAT5		0,00000000,[CRC16],1	No data pattern sent
DAT6		0,00000000,[CRC16],1	No data pattern sent
DAT7		0,00000000,[CRC16],1	No data pattern sent

Table 6-4 : 4-bit Bus Testing Pattern

Data line	Data pattern sent by the host	Reversed pattern sent by the card	Notes
DAT0	0,10xxxxxxxxx,[CRC16],1	0,01000000,[CRC16],1	Start bit defines beginning of pattern
DAT1	0,01xxxxxxxxx,[CRC16],1	0,10000000,[CRC16],1	
DAT2	0,10xxxxxxxxx,[CRC16],1	0,01000000,[CRC16],1	
DAT3	0,01xxxxxxxxx,[CRC16],1	0,10000000,[CRC16],1	
DAT4	0,10xxxxxxxxx,[CRC16],1	0,01000000,[CRC16],1	
DAT5	0,01xxxxxxxxx,[CRC16],1	0,10000000,[CRC16],1	
DAT6	0,10xxxxxxxxx,[CRC16],1	0,01000000,[CRC16],1	
DAT7	0,01xxxxxxxxx,[CRC16],1	0,10000000,[CRC16],1	

Table 6-5 : 8-bit Bus Testing Pattern

6.2.5 Bus Width Selection

After the host has verified the functional pins on the bus it should change the bus width configuration accordingly, using the SWITCH command.
The bus width configuration is changed by writing to the BUS_WIDTH byte in the Modes Segment of the EXT_CSD register (using the SWITCH command to do so). After power-on, or software reset, the contents of the BUS_WIDTH byte is 0x00.
The valid values for this register are defined in 'BUS_WIDTH', in page 86. If the host tries to write an invalid value, the BUS_WIDTH byte is not changed and the SWITCH_ERROR bit is set. This register is write only.

6.2.6 Data Read

The DAT0-DAT7 bus line levels are high when no data is transmitted. A transmitted data block consists of a start bit (LOW), on each DAT line, followed by a continuous data stream. The data stream contains the payload data (and error correction bits if an off-card ECC is used). The data stream ends with an end bit (HIGH), on each DAT line (see Figure 6-12 - Figure 6-14). The data transmission is synchronous to the clock signal.
The payload for block oriented data transfer is protected by a CRC check sum, on each DAT line (see Chapter 6.4).

- **Block Read**

Block read is similar to stream read, except the basic unit of data transfer is a block whose maximum size is defined in the CSD (READ_BL_LEN). If READ_BL_PARTIAL is set, smaller blocks whose starting and ending address are entirely contained within one physical block (as defined by READ_BL_LEN) may also be transmitted. Unlike stream read, a CRC is appended to the end of each block ensuring data transfer integrity. CMD17 (READ_SINGLE_BLOCK) initiates a block read and after completing the transfer, the card returns to the *Transfer State*.
CMD18 (READ_MULTIPLE_BLOCK) starts a transfer of several consecutive blocks. Two types of multiple block read transactions are defined (the host can use either one at any time):

- Open-ended Multiple block read
  The number of blocks for the read multiple block operation is not defined. The card will continuously transfer data blocks until a stop transmission command is received.

- Multiple block read with pre-defined block count
  The card will transfer the requested number of data blocks, terminate the transaction and return to *transfer* state. Stop command is not required at the end of this type of multiple block read, unless terminated with an error. In order to start a multiple block read with pre-defined block count the host must use the SET_BLOCK_COUNT command (CMD23) immediately preceding the READ_MULTIPLE_BLOCK (CMD18) command. Otherwise the card will start an open -ended multiple block read which can be stopped using the STOP_TRANSMISION command.

The host can abort reading at any time, within a multiple block operation, regardless of the its type. Transaction abort is done by sending the stop transmission command.
If either one of the following conditions occur, the card will reject the command, remain in *Tran* state and respond with the respective error bit set.

- The host provides an out of range address as an argument to either CMD17 or CMD18. ADDRESS_OUT_OF_RANGE is set.
- The currently defined block length is illegal for a read operation. BLOCK_LEN_ERROR is set.
- The address/block-length combination positions the first data block misaligned to the card physical blocks. ADDRESS_MISALIGN is set.

If the card detects an error (e.g. out of range, address misalignment, internal error, etc.) during a multiple block read operation (both types) it will stop data transmission and remain in the *Data State*. The host must then abort the operation by sending the stop transmission command. The read error is reported in the response to the stop transmission command.
If the host sends a stop transmission command after the card transmits the last block of a multiple block operation with a pre-defined number of blocks, it is regarded as an illegal command, since the card is no longer in *data* state.
If the host uses partial blocks whose accumulated length is not block aligned, and block misalignment is not allowed, the card shall detect a block misalignment error condition during the transmission of the first misaligned block and the content of the further transferred bits is undefined. As the host sends CMD12 the card will respond with the ADDRESS_MISALIGN bit set and return to *Tran* state.
If the host sets the argument of the SET_BLOCK_COUNT command (CMD23) to all 0s, then the command is accepted; however, a subsequent read will follow the open-ended multiple block read protocol (STOP_TRANSMISSION command - CMD12 - is required).

6.2.7 Data Write

The data transfer format of write operation is similar to the data read. For block oriented write data transfer, the CRC check bits are added to each data block. The card performs a CRC parity check (see Chapter 6.4) for each received data block prior to the write operation. By this mechanism, writing of erroneously transferred data can be prevented.

- **Block Write**

During block write (CMD24 - 27) one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block by the host. A card supporting block write shall always be able to accept a block of data defined by WRITE_BL_LEN. If the CRC fails, the card shall indicate the failure on the DAT0 line (see below); the transferred data will be discarded and not written, and all further transmitted blocks (in multiple block write mode) will be ignored.
CMD25 (WRITE_MULTIPLE_BLOCK) starts a transfer of several consecutive blocks. Two types of multiple block write transactions, identical to the multiple block read, are defined (the host can use either one at any time):

- Open-ended Multiple block write
  The number of blocks for the write multiple block operation is not defined. The card will continuously accept and pro gram data blocks until a stop transmission command is received.
- Multiple block write with pre-defined block count
  The card will accept the requested number of data blocks, terminate the transaction and return to *transfer* state. Stop command is not required at the end of this type of multiple block write, unless terminated with an error. In order to start a multiple block write with pre-defined block count the host must use the SET_BLOCK_COUNT command (CMD23) immediately preceding the WRITE_MULTIPLE_BLOCK (CMD25) command. Otherwise the card will start an open -ended multiple block write which can be stopped using the STOP_TRANSMISION command.

The host can abort writing at any time, within a multiple block operation, regardless of the its type. Transaction abort is done by sending the stop transmission command. If a multiple block write with pre-defined block count is aborted, the data in the remaining blocks is not defined.
If either one of the following conditions occur, the card will reject the command, remain in *Tran* state and respond with the respective error bit set.

- The host provides an out of range address as an argument to either CMD24 or CMD25. ADDRESS_OUT_OF_RANGE is set.
- The currently defined block length is illegal for a write operation. BLOCK_LEN_ERROR is set.
- The address/block-length combination positions the first data block misaligned to the card physical blocks. ADDRESS_MISALIGN is set.

If the card detects an error (e.g. write protect violation, out of range, address misalignment, internal error, etc.) during a multiple block write operation (both types) it will ignore any further incoming data blocks and remain in the *Receive State.* The host must then abort the operation by sending the stop transmission command. The write error is reported in the response to the stop transmission command.
If the host sends a stop transmission command after the card received the last data block of a multiple block write with a pre-defined number of blocks, it is regarded as an illegal command, since the card is no longer in *data* state.
If the host uses partial blocks whose accumulated length is not block aligned, and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card shall detect the block misalignment error during the reception of the first misaligned block, abort the write operation, and ignore all further incoming data. As the host sends CMD12, the card will respond with the ADDRESS_MISALIGN bit set and return to *Tran* state.
If the host sets the argument of the SET_BLOCK_COUNT command (CMD23) to all 0s, then the command is accepted; however, a subsequent write will follow the open-ended multiple block write protocol (STOP_TRANSMISSION command - CMD12 - is required).

Programming of the CID and CSD registers does not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card will report an error and not change any register contents.
Some cards may require long and unpredictable times to write a block of data. After receiving a block of data and completing the CRC check, the card will begin writing and hold the DAT0 line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host may poll the status of the card with a SEND_STATUS command (CMD13) at any time, and the card will respond with its status. The status bit READY_FOR_DATA indicates whether the card can accept new data or whether the write process is still in progress). The host may deselect the card by issuing CMD7 which will displace the card into the *Disconnect State* and release the DAT0 line without interrupting the write operation. When reselecting the card, it will reactivate busy indication by pulling DAT0 to low if programming is still in progress and the write buffer is unavailable.

6.2.8 CSD Programming

Programming of the CSD register does not require a previous block length setting. After sending CMD27 and receiving an R1 response, the start bit (=0)is sent, the modified CSD register (=16 bytes), CRC16 (=2bytes), and end bit (=1). The host can change only the least significant 16bits [15:0] of the CSD. The rest of the CSD register content must match the MultiMediaCard CSD Register. If the card detects a content inconsistency between the old and new CSD register, it will not reprogram the CSD in order to ensure validity of the CRC field in the CSD register.

Bits [7:1] are the CRC7 of bits [127:8] of the CSD register, which should be recalculated once the register changes. After calculating CRC7, the CRC16 should also be calculated for all of the CSD register [127:0].

6.2.9 Erase

MultiMediaCards, in addition to the implicit erase executed by the card as part of the write operation, provides a host explicit erase function. The erasable unit of the MultiMediaCard is the “Erase Group”; Erase group is measured in write blocks which are the basic writable units of the card. The size of the Erase Group is a card specific parameter and defined in the CSD. The content of an explicitly erased memory range shall be 0.
The host can erase a contiguous range of Erase Groups. Starting the erase process is a three steps sequence. First the host defines the start address of the range using the ERASE_GROUP_START (CMD35) command, next it defines the last address of the range using the ERASE_GROUP_END (CMD36) command and finally it starts the erase process by issuing the ERASE (CMD38) command. The address field in the erase commands is an Erase Group address in byte units. The card will ignore all LSB’s below the Erase Group size, effectively rounding the address down to the Erase Group boundary.

If an erase command (either CMD35, CMD36, CMD38) is received out of the defined erase sequence, the card shall set the ERASE_SEQ_ERROR bit in the status register and reset the whole sequence.
If the host provides an out of range address as an argument to CMD35 or CMD36, the card will reject the command, respond with the ADDRESS_OUT_OF_RANGE bit set and reset the whole erase sequence.
If an ‘non erase’ command (neither of CMD35, CMD36, CMD38 or CMD13) is received, the card shall respond with the ERASE_RESET bit set, reset the erase sequence and execute the last command. Commands not addressed to the selected card do not abort the erase sequence.
If the erase range includes write protected blocks, they shall be left intact and only the non protected blocks shall be erased. The WP_ERASE_SKIP status bit in the status register shall be set.
As described above for block write, the card will indicate that an erase is in progress by holding DAT0 low. The actual erase time may be quite long, and the host may issue CMD7 to deselect the card.

6.2.10 Write Protect Management

In order to allow the host to protect data against erase or write, the MultiMediaCard shall support two levels of write protect commands:

- The entire card may be write protected by setting the permanent or temporary write protect bits in the CSD.
- Specific segments of the cards may be write protected. The segment size is defined in units of WP_GRP_SIZE erase groups as specified in the CSD. The SET_WRITE_PROT command sets the write protection of the addressed write-pro tect group, and the CLR_WRITE_PROT command clears the write protection of the addressed write-protect group.

The SEND_WRITE_PROT command is similar to a single block read command. The card shall send a data block containing 32 write protection bits (representing 32 write protect groups starting at the specified address) followed by 16 CRC bits. The address field in the write protect commands is a group address in byte units. The card will ignore all LSB’s below the group size.
If the host provides an out of range address as an argument to CMD28, CMD29 or CMD30, the card will reject the command, respond with the ADDRESS_OUT_OF_RANGE bit set and remain in the *Tran* state.

6.2.11 Card Lock/Unlock Operation

The password protection feature enables the host to lock the card by providing a password, which later will be used for unlocking the card. The password and its size is kept in an 128 bit PWD and 8 bit PWD_LEN registers, respectively. These registers are non-volatile so that a power cycle will not erase them.
A locked card responds to (and executes) all commands in the “basic” command class (class 0) and “lock card” command class. Thus the host is allowed to reset, initialize, select, query for status, etc., but not to access data on the card. If the password was previously set (the value of PWD_LEN is not ‘0’) the card will be locked automatically after power on.
Similar to the existing CSD and CID register write commands the lock/unlock command is available in “transfer state” only. This means that it does not include an address argument and the card has to be selected before using it.
The card lock/unlock command has the structure and bus transaction type of a regular single block write command. The transferred data block includes all the required information of the command (password setting mode, PWD itself, card lock/unlock etc.). The following table describes the structure of the command data block.

Byte #	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Reserved				ERASE	LOCK_UNLOCK	CLR_PWD	SET_PWD
1	PWD_LEN							
2	Password data							
...								
PWD_LEN + 1								

Table 6-6 : Lock Card Data Structure

- **ERASE:** ‘1’ Defines Forced Erase Operation (all other bits shall be ‘0’) and only the cmd byte is sent.
- **LOCK/UNLOCK**: ‘1’ = Locks the card. ‘0’ = Unlock the card (note that it is valid to set this bit together with SET_PWD but it is not allowed to set it together with CLR_PWD).
- **CLR_PWD**: ‘1’ = Clears PWD.
- **SET_PWD**: ‘1’ = Set new password to PWD
- **PWD_LEN**: Defines the following password length (in bytes). Valid password length are 1 to 16 bytes.
- **PWD:** The password (new or currently used depending on the command).

The data block size shall be defined by the host before it sends the card lock/unlock command. This will allow different password sizes.

The following paragraphs define the various lock/unlock command sequences:

- **Setting the Password**
  - Select the card (CMD7), if not previously selected already
  - Define the block length (CMD16), given by the 8bit card lock/unlock mode, the 8 bits password size (in bytes), and the number of bytes of the new password. In case that a password *replacement* is done, then the block size shall consider that both passwords, the old and the new one, are sent with the command.
  - Send Card Lock/Unlock command with the appropriate data block size on the data line including 16 bit CRC. The data block shall indicate the mode (SET_PWD), the length (PWD_LEN) and the password itself. In case that a password *replacement* is done, then the length value (PWD_LEN) shall include both passwords, the old and the new one, and the PWD field shall include the old password (currently used) followed by the new pass word.
  - In case that a password replacement is attempted with PWD_LEN set to the length of the old pass-word only, the LOCK_UNLOCK_FAILED error bit is set in the status register and the old password is not changed.
  - In case that the sent old password is not correct (not equal in size and content) then LOCK_UNLOCK_FAILED error bit will be set in the status register and the old password does not change. In case that PWD matches the sent old password then the given new password and its size will be saved in the PWD and PWD_LEN fields, respectively.

Note that the password length register (PWD_LEN) indicates if a password is currently set. When it equals '0' there is no password set. If the value of PWD_LEN is not equal to zero the card will lock itself after power up. It is possible to lock the card immediately in the current power session by setting the LOCK/UNLOCK bit (while setting the password) or sending additional command for card lock.

- **Reset the Password:**
  - Select the card (CMD7), if not previously selected already
  - Define the block length (CMD16), given by the 8 bit card lock/unlock mode, the 8 bit password size (in bytes), and the number of bytes of the currently used password.
  - Send the card lock/unlock command with the appropriate data block size on the data line including 16 bit CRC. The data block shall indicate the mode CLR_PWD, the length (PWD_LEN) and the password (PWD) itself (LOCK/ UNLOCK bit is don't care). If the PWD and PWD_LEN content match the sent password and its size, then the con tent of the PWD register is cleared and PWD_LEN is set to 0. If the password is not correct then the LOCK_UNLOCK_FAILED error bit will be set in the status register.

- **Locking a card:**
  - Select the card (CMD7), if not previously selected already
  - Define the block length (CMD16), given by the 8 bit card lock/unlock mode, the 8 bit password size (in bytes), and the number of bytes of the currently used password.
  - Send the card lock/unlock command with the appropriate data block size on the data line including 16 bit CRC. The data block shall indicate the mode LOCK, the length (PWD_LEN) and the password (PWD) itself.

If the PWD content equals to the sent password then the card will be locked and the card-locked status bit will be set in the status register. If the password is not correct then LOCK_UNLOCK_FAILED error bit will be set in the status register.
Note that it is possible to set the password and to lock the card in the same sequence. In such case the host shall perform all the required steps for setting the password (as described above) including the bit LOCK set while the new password command is sent.
If the password was previously set (PWD_LEN is not '0'), then the card will be locked automatically after power on reset. An attempt to lock a locked card or to lock a card that does not have a password will fail and the LOCK_UNLOCK_FAILED error bit will be set in the status register.

- **Unlocking the card:**
  - Select the card (CMD7), if not previously selected already.
  - Define the block length (CMD16), given by the 8 bit card lock/unlock mode, the 8 bit password size (in bytes), and the number of bytes of the currently used password.
  - Send the card lock/unlock command with the appropriate data block size on the data line including 16 bit CRC. The data block shall indicate the mode UNLOCK, the length (PWD_LEN) and the password (PWD) itself.

If the PWD content equals to the sent password then the card will be unlocked and the card-locked status bit will be cleared in the status register. If the password is not correct then the LOCK_UNLOCK_FAILED error bit will be set in the status register.
Note that the unlocking is done only for the current power session. As long as the PWD is not cleared the card will be locked automatically on the next power up. The only way to unlock the card is by clearing the password.
An attempt to unlock an unlocked card will fail and LOCK_UNLOCK_FAILED error bit will be set in the status register.

- **Forcing Erase:**

In case that the user forgot the password (the PWD content) it is possible to erase all the card data content along with the PWD content. This operation is called *Forced Erase*.

- Select the card (CMD7), if not previously selected already.
- Define the block length (CMD16) to 1 byte (8bit card lock/unlock command). Send the card lock/unlock command with the appropriate data block of one byte on the data line including 16 bit CRC. The data block shall indicate the mode ERASE (the ERASE bit shall be the only bit set).

If the ERASE bit is not the only bit in the data field then the LOCK_UNLOCK_FAILED error bit will be set in the status register and the erase request is rejected.
If the command was accepted then ALL THE CARD CONTENT WILL BE ERASED including the PWD and PWD_LEN register content and the locked card will get unlocked. In addition, if the card is temporary write protected it will be unprotected (write enabled), the temporary-write-protect bit in the CSD and all Write-Protect-Groups will be cleared.
An attempt to force erase on an unlocked card will fail and LOCK_UNLOCK_FAILED error bit will be set in the status register.
If a force erase command is issued on a permanently-write-protect media the command will fail (card stays locked) and the LOCK_UNLOCK_FAILED error bit will be set in the status register.
The Force Erase time-out is specified in Chapter 6.5.2

6.3 Clock Control

The MultiMediaCard bus clock signal can be used by the host to put the card into energy saving mode, or to control the data flow (to avoid under-run or over-run conditions) on the bus. The host is allowed to lower the clock frequency or shut it down.
There are a few restrictions the host must follow:

- The bus frequency can be changed at any time (under the restrictions of maximum data transfer frequency, defined by the card, and the identification frequency defined by the specification document).
- It is an obvious requirement that the clock must be running for the card to output data or response tokens. After the last MultiMediaCard bus transaction, the host is required, to provide **8 (eight)** clock cycles for the card to complete the operation before shutting down the clock. Following is a list of the various bus transactions:
- A command with no response. 8 clocks after the host command end bit.
- A command with response. 8 clocks after the card response end bit.
- A read data transaction. 8 clocks after the end bit of the last data block.
- A write data transaction. 8 clocks after the CRC status token.
- The host is allowed to shut down the clock of a “busy” card. The card will complete the programming operation regardless of the host clock. However, the host must provide a clock edge for the card to turn off its busy signal. Without a clock edge the card (unless previously disconnected by a deselect command -CMD7) will force the DAT0 line down, forever.

6.4 Cyclic Redundancy Codes (CRC)

The CRC is intended for protecting MultiMediaCard commands, responses and data transfer against transmission errors on the MultiMediaCard bus. One CRC is generated for every command and checked for every response on the CMD line. For data blocks one CRC per transferred block, per data line, is generated. The CRC is generated and checked as described in the following.

- **CRC7**

The CRC7 check is used for all commands, for all responses except type R3, and for the CSD and CID registers. The CRC7 is a 7-bit value and is computed as follows:

Generator polynomial $G(x) = x^7 + x^3 + 1$

$$M(x) = (\text{first bit}) \times x^n + (\text{second bit}) \times x^{n-1} + \ldots + (\text{last bit}) \times x^0$$

$$CRC[6 \ldots 0] = \text{Remainder}[(M(x) \cdot x^7) / G(x)]$$

All CRC registers are initialized to zero. The first bit is the most left bit of the corresponding bit string (of the command, response, CID or CSD). The degree *n* of the polynomial is the number of CRC protected bits decreased by one. The number of bits to be protected is 40 for commands and responses (n = 39), and 120 for the CSD and CID (n = 119).

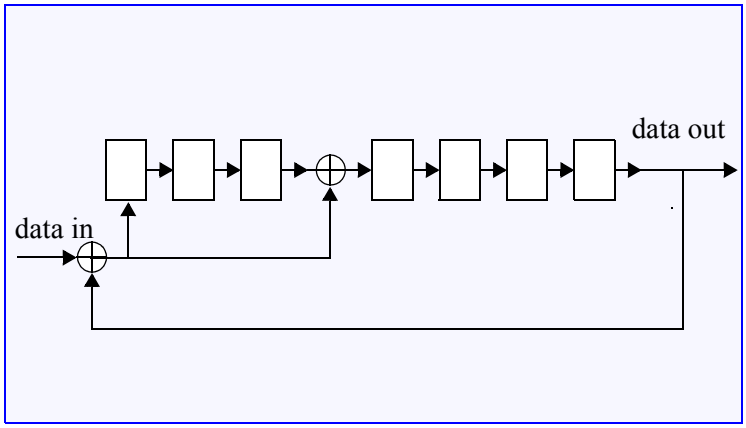


Figure 6-3 : CRC7 Generator/Checker

- **CRC16**

The CRC16 is used for payload protection in block transfer mode. The CRC check sum is a 16-bit value and is computed as follows:

Generator polynomial $G(x) = x^{16} + x^{12} + x^5 + 1$

$$M(x) = (\text{first bit}) \times x^n + (\text{second bit}) \times x^{n-1} + \ldots + (\text{last bit}) \times x^0$$

$$CRC[15\ldots0] = \text{Remainder}[(M(x) \cdot x^{16}) / G(x)]$$

All CRC registers are initialized to zero. The first bit is the first data bit of the corresponding block. The degree *n* of the polynomial denotes the number of bits of the data block decreased by one (e.g. n = 4095 for a block length of 512 bytes). The generator polynomial $G(x)$ is a standard CCITT polynomial. The code has a minimal distance d=4 and is used for a payload length of up to 2048 Bytes (n <= 16383).
The same CRC16 calculation is used for all bus configurations. In 4 bit and 8 bit bus configurations, the CRC16 is calculated for each line separately. Sending the CRC is synchronized so the CRC code is transferred at the same time in all lines.

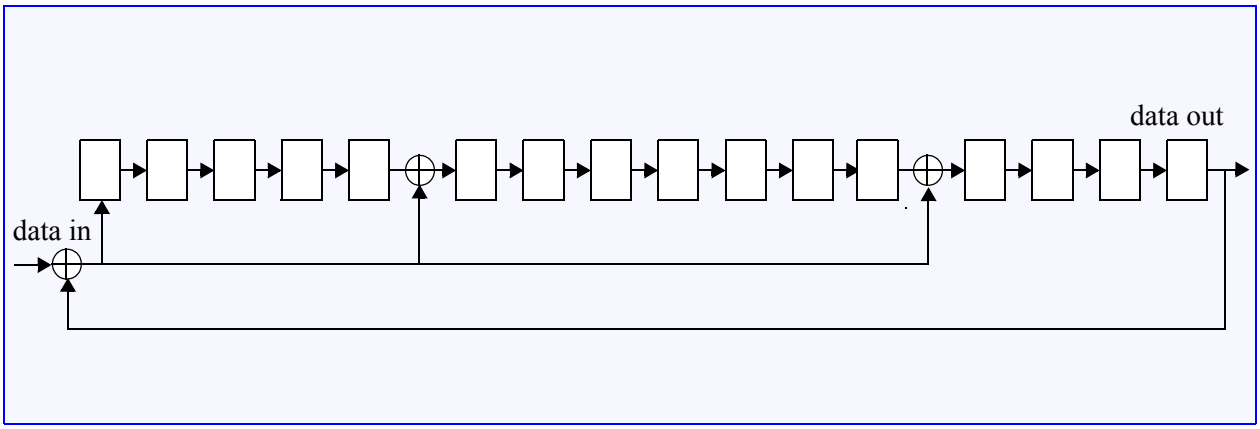


Figure 6-4 : CRC16 Generator/Checker

6.5 Error Conditions

6.5.1 CRC and Illegal Command

All commands are protected by CRC (cyclic redundancy check) bits. If the addressed card's CRC check fails, the card does not respond, and the command is not executed; the card does not change its state, and COM_CRC_ERROR bit is set in the status register.
Similarly, if an illegal command has been received, the card shall not change its state, shall not respond and shall set the ILLEGAL_COMMAND error bit in the status register. Only the non-erroneous state branches are shown in the state diagrams (see Figure 6-1 to Figure 6-2). Table 6-18 contains a complete state transition description.
There are different kinds of illegal commands:

- Commands which belong to classes not supported by the card (e.g. write commands in read only cards).
- Commands not allowed in the current state (e.g. CMD2 in Transfer State).
- Commands which are not defined (e.g. CMD44).

6.5.2 Read, Write, Erase And Force Erase Time-out Conditions

The times after which a time-out condition for read/write/erase operations occurs are (card independent) **10 times longer** than the typical access/program times for these operations given below. A card shall complete the command within this time period, or give up and return an error message. If the host does not get a response within the defined time-out it should assume the card is not going to respond anymore and try to recover (e.g. reset the card, power cycle, reject, etc.).
The typical access and program times are defined as follows:

- **Read**

The read access time is defined as the sum of the two times given by the CSD parameters TAAC and NSAC (see Chapter 6.11). These card parameters define the typical delay between the end bit of the read command and the start bit of the data block.

- **Write**

The R2W_FACTOR field in the CSD is used to calculate the typical block program time obtained by multiplying the read access time by this factor. It applies to all write/erase commands (e.g. SET(CLEAR)_WRITE_PROTECT, PROGRAM_CSD(CID) and the block write commands).

- **Erase**

The duration of an erase command will be (order of magnitude) the number of write blocks to be erased multiplied by the block write delay.

- **Force Erase**

The duration of the Force Erase command using CMD42 is specified to be a fixed time-out of 3 minutes.

6.6 Minimum Performance

A MMCplus and MMCmobile card has to fullfill the requirements set for the read and write access performance.

6.6.1 Speed Class Definition

The speed class definition is for indication of the minimum performance of a card. The classes are defined based on the 150kB/s base value. The minimum performance of the card can then be marked by defined multiples of the base value e.g. 2.4MB/s. Only following speed classes are defined (note that MMCplus and MMCmobile cards are always including 8bit data bus and the categories below states the configuration with which the card is operated):

Low bus category classes (26MHz clock with 4bit data bus operation)

- 2.4 MB/s Class A
- 3.0 MB/s Class B
- 4.5 MB/s Class C
- 6.0 MB/s Class D
- 9.0 MB/s Class E

Mid bus category classes (26MHz clock with 8bit data bus or 52MHz clock with 4bit data bus operation):

- 12.0 MB/s Class F
- 15.0 MB/s Class G
- 18.0 MB/s Class H
- 21.0 MB/s Class J

High bus category classes (52MHz clock with 8bit data bus operation):

- 24.0 MB/s Class K
- 30.0 MB/s Class M
- 36.0 MB/s Class O
- 42.0 MB/s Class R
- 48.0 MB/s Class T

The performance values for both write and read accesses are stored into the EXT_CSD register for electrical reading (see chapter 5.5.4 on page 32). Only the defined values and classes are allowed to be used.

6.6.2 Absolute Minimum

Absolute minimum read and write access performance which all MMCplus and MMCmobile cards has to fullfill is 2.4MB/s. This is the Class A.

6.6.3 Measurement of the Performance

The procedure for the measurement of the performance of the card is defined in detail in the Compliance Documentation. Initial state of the memory in prior to the test is: filled with random data. The test is performed by writing/reading a 64kB chunk of data to/from random logical addresses (aligned to physical block boundaries)of the card. A predefined multiple block write/read is used with block count of 128 (64kB as 512B blocks are used). The performance is calculated as average out of several 64kB accesses.

Same test is performed with all applicable clock frequency and bus width options as follows:

- 52MHz, 8bit bus (if 52MHz clock frequency is supported by the card)
- 52MHz, 4bit bus (if 52MHz clock frequency is supported by the card)
- 26MHz, 8bit bus
- 26MHz, 4bit bus

In case the minimum performance of the card exceeds the physical limit of one of the above mentioned options the card has to also fulfill accordingly the performance criteria as defined in **MIN_PERF_a_b_ff** in chapter 5.5.4 on page 32

6.7 Commands

6.7.1 Command Types

There are four kinds of commands defined to control the MultiMediaCard:

* broadcast commands (bc), no response
* broadcast commands with response (bcr)
* addressed (point-to-point) commands (ac), no data transfer on DAT lines
* addressed (point-to-point) data transfer commands (adtc), data transfer on DAT lines
* All commands and responses are sent over the CMD line of the MultiMediaCard bus. The command transmission always starts with the left bit of the bitstring corresponding to the command codeword.

6.7.2 Command Format

All commands have a fixed code length of 48 bits, needing a transmission time of 0.92 microSec @ 52 MHz

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'1'	x	x	x	'1'
Description	start bit	transmission bit	command index	argument	CRC7	end bit

Table 6-7 : Format (0.92us @52MHz)

A command always starts with a start bit (always '0'), followed by the bit indicating the direction of transmission (host = '1'). The next 6 bits indicate the index of the command, this value being interpreted as a binary coded number (between 0 and 63). Some commands need an argument (e.g. an address), which is coded by 32 bits. A value denoted by 'x' in the table above indicates this variable is dependent on the command. All commands are protected by a CRC (see Chapter 6.4 for the definition of CRC7). Every command codeword is terminated by the end bit (always '1'). All commands and their arguments are listed in Table 6-9 -Table 6-17.

6.7.3 Command Classes

The command set of the MultiMediaCard system is divided into several classes (See Table 6-8). Each class supports a subset of card functions.
Class 0 is mandatory and shall be supported by all cards. The other classes are either mandatory only for specific card types or optional. The supported Card Command Classes (CCC) are coded as a parameter in the card specific data (CSD) register of each card, providing the host with information on how to access the card.

Card Command Class (CCC)	Class Description	Supported commands																								
		0	1	2	3	4	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	23	24	25	26	27
class 0	basic	+	+	+	+	+	+	+	+	+	+		+	+	+	+				+						
class 1	stream read											+														
class 2	block read																+	+	+			+				
class 3	stream write																				+					
class 4	block write																+					+	+	+	+	+
class 5	erase																									
class 6	write protection																									
class 7	lock card																+									
class 8	application specific																									
class 9	I/O mode																									
class 10-11	reserved																									

Card Command Class (CCC)	Class Description	Supported commands									
		28	29	30	35	36	38	39	40	42	55	56
class 0	basic											
class 1	stream read											
class 2	block read											
class 3	stream write											
class 4	block write											
class 5	erase				+	+	+					
class 6	write protection	+	+	+								
class 7	lock card									+		
class 8	application specific										+	+
class 9	I/O mode							+	+			
class 10-11	reserved											

Table 6-8 : Card Command Classes (CCCs)

6.7.4 Detailed Command Description

The following tables define in detail all MultiMediaCard bus commands. The responses R1-R5 are defined in Chapter 6.8. The registers CID, CSD, EXT_CSD and DSR are described in Chapter 5.5.

CMD INDEX	Type	Argument	Resp	Abbreviation	Command Description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets the card to idle state
CMD1	bcr	[31:0] OCR without busy	R3	SEND_OP_COND	Asks the card, in idle state, to send its Operating Conditions Register contents in the response on the CMD line.
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks the card to send its CID number on the CMD line
CMD3	ac	[31:16] RCA [15:0] stuff bits	R1	SET_RELATIVE_ADDR	Assigns relative address to the card
CMD4	Not Supported				
CMD5	Reserved				
CMD6	ac	[31:26] Set to 0 [25:24] Access [23:16] Index [15:8] Value [7:3] Set to 0 [2:0] Cmd Set	R1b	SWITCH	Switches the mode of operation of the selected card or modifies the EXT_CSD registers. (See chapter 6.2.1)
CMD7	ac	[31:16] RCA [15:0] stuff bits	R1b[a]	SELECT/ DESELECT_CARD	Command toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases the card is selected by its own relative address and gets deselected by any other address; address 0 deselects the card.
CMD8	adtc	[31:0] stuff bits	R1	SEND_EXT_CSD	The card sends its EXT_CSD register as a block of data.
CMD9	ac	[31:16] RCA [15:0] stuff bits	R2	SEND_CSD	Addressed card sends its card-specific data (CSD) on the CMD line.
CMD10	ac	[31:16] RCA [15:0] stuff bits	R2	SEND_CID	Addressed card sends its card identification (CID) on CMD the line.
CMD11	Not Supported				
CMD12	ac	[31:0] stuff bits	R1b	STOP_TRANSMISSION	Forces the card to stop transmission
CMD13	ac	[31:16] RCA [15:0] stuff bits	R1	SEND_STATUS	Addressed card sends its status register.
CMD14	adtc	[31:0] stuff bits	R1	BUSTEST_R	A host reads the reversed bus testing data pattern from a card.
CMD15	ac	[31:16] RCA [15:0] stuff bits	-	GO_INACTIVE_STATE	Sets the card to inactive state
CMD19	adtc	[31:0] stuff bits	R1	BUSTEST_W	A host sends the bus test data pattern to a card.

Table 6-9 : Basic Commands And Read Stream Commands (Class 0 And Class 1)

a. Only from the selected card

CMD INDEX	Type	Argument	Resp	Abbreviation	Command Description
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	Sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD.
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command.[a]
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously transfers data blocks from card to host until interrupted by a stop command, or the requested number of data blocks is transmitted

Table 6-10 : Block Oriented Read Commands (Class 2)

a.The transferred data must not cross a physical block boundary, unless READ_BLK_MISALIGN is set in the CSD register

CMD INDEX	Type	Argument	Resp	Abbreviation	Command Description
CMD20	Not supported				
CMD21 ... CMD22	Reserved				

Table 6-11 : Stream Write Commands (Class 3)

CMD INDEX	Type	Argument	Resp	Abbreviation	Command Description
CMD23	ac	[31:16] set to 0 [15:0] number of blocks	R1	SET_BLOCK_COUNT	Defines the number of blocks which are going to be transferred in the immediately succeeding multiple block read or write command. If the argument is all 0s, the subsequent read/ write operation will be open-ended.
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.[a]
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a STOP_TRANSMISSION follows or the requested number of block received.
CMD26	Not applicable				
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD.

Table 6-12 : Block Oriented Write Commands (Class 4)

a.The transferred data must not cross a physical block boundary unless WRITE_BLK_MISALIGN is set in the CSD

CMD INDEX	Type	Argument	Resp	Abbreviation	Command Description
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	ac	[31:0] data address	R1b	CLR_WRITE_PROT	If the card provides write protection features, this command clears the write protection bit of the addressed group.
CMD30	adtc	[31:0] write protect data address	R1	SEND_WRITE_PROT	If the card provides write protection features, this command asks the card to send the status of the write protection bits. [a]
CMD31	Reserved				

Table 6-13 : Block Oriented Write Protection Commands (Class 6)

a. 32 write protection bits (representing 32 write protect groups starting at the specified address) followed by 16 CRC bits are transferred in a payload format via the data lines. The last (least significant) bit of the protection bits corresponds to the first addressed group. If the addresses of the last groups are outside the valid range, then the corresponding write protection bits shall be set to zero.

CMD INDEX	Type	Argument	Resp	Abbreviation	Command Description
CMD32 ... CMD34	Reserved. These command indexes cannot be used in order to maintain backwards compatibility with older versions of the MultiMediaCards				
CMD35	ac	[31:0] data address	R1	ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase
CMD36	ac	[31:0] data address	R1	ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase
CMD37	Reserved. This command index cannot be used in order to maintain backwards compatibility with older versions of the MultiMediaCards				
CMD38	ac	[31:0] stuff bits	R1b	ERASE	Erases all previously selected write blocks

Table 6-14 : Erase Commands (Class 5)

CMD INDEX	Type	Argument	Resp	Abbreviation	Command Description
CMD39 CMD40	MMCA Optional Command, currently not supported.				
CMD41	Reserved				

Table 6-15 : I/O Mode Commands (Class 9)

CMD INDEX	Type	Argument	Resp	Abbreviation	Command Description
CMD42	adtc	[31:0] stuff bits.	R1b	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the card. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43... CMD54	Reserved				

Table 6-16 : Lock Card (Class 7)

CMD INDEX	Type	Argument	Resp	Abbreviation	Command Description
CMD55 CMD56	MMCA Optional Command, currently not supported.				
CMD57 ... CMD59	Reserved				
CMD60 ... CMD63	Reserved for manufacturer				

Table 6-17 : Application Specific Commands (Class 8)

All future reserved commands shall have a codeword length of 48 bits, as well as their responses (if there are any).

6.8 Card State Transition

Table defines the card state transitions in dependency of the received command.

	Current State											
	idle	**ready**	**ident**	**stby**	**tran**	**data**	**btst**	**rcv**	**prg**	**dis**	**ina**	**irq**
Command	Changes to											
Class Independent												
CRC error	-	-	-	-	-	-	-	-	-	-	-	stby
command not supported	-	-	-	-	-	-	-	-	-	-	-	stby
Class 0												
CMD0	idle	idle	idle	idle	idle	idle	idle	idle	idle	idle	-	stby
CMD1, card V_{DD} range compatible	ready	-	-	-	-	-	-	-	-	-	-	stby
CMD1, card is busy	idle	-	-	-	-	-	-	-	-	-	-	stby
CMD1, card V_{DD} range not compatible	ina	-	-	-	-	-	-	-	-	-	-	stby
CMD2, card wins bus	-	ident	-	-	-	-	-	-	-	-	-	stby
CMD2, card loses bus	-	ready	-	-	-	-	-	-	-	-	-	stby
CMD3	-	-	stby	-	-	-	-	-	-	-	-	stby
CMD4	-	-	-	stby	-	-	-	-	-	-	-	stby
CMD6	-	-	-	-	prg	-	-	-	-	-	-	stby
CMD7, card is addressed	-	-	-	tran	-	-	-	-	-	prg	-	stby
CMD7, card is not addressed	-	-	-	-	stby	stby	-	-	dis	-	-	stby
CMD8	-	-	-	-	data	-	-	-	-	-	-	stby
CMD9	-	-	-	stby	-	-	-	-	-	-	-	stby
CMD10	-	-	-	stby	-	-	-	-	-	-	-	stby
CMD12	-	-	-	-	-	tran	-	prg	-	-	-	stby
CMD13	-	-	-	stby	tran	data	btst	rcv	prg	dis	-	stby
CMD14	-	-	-	-	-	-	tran	-	-	-	-	stby
CMD15	-	-	-	ina	ina	ina	ina	ina	ina	ina	-	stby
CMD19	-	-	-	-	btst	-	-	-	-	-	-	stby
Class 1												
CMD11	-	-	-	-	data	-	-	-	-	-	-	stby
Class 2												
CMD16	-	-	-	-	tran	-	-	-	-	-	-	stby
CMD17	-	-	-	-	data	-	-	-	-	-	-	stby
CMD18	-	-	-	-	data	-	-	-	-	-	-	stby
CMD23	-	-	-	-	tran	-	-	-	-	-	-	stby

Table 6-18 : Card State Transition Table

	Current State											
	idle	**ready**	**ident**	**stby**	**tran**	**data**	**btst**	**rcv**	**prg**	**dis**	**ina**	**irp**
Class 3												
CMD20	-	-	-	-	rcv	-	-	-	-	-	-	stby
Class 4												
CMD16	see class 2											
CMD23	see class 2											
CMD24	-	-	-	-	rcv	-	-	-	rcv	-	-	stby
CMD25	-	-	-	-	rcv	-	-	-	rcv	-	-	stby
CMD26	-	-	-	-	rcv	-	-	-	-	-	-	stby
CMD27	-	-	-	-	rcv	-	-	-	-	-	-	stby
Class 6												
CMD28	-	-	-	-	prg	-	-	-	-	-	-	stby
CMD29	-	-	-	-	prg	-	-	-	-	-	-	stby
CMD30	-	-	-	-	data	-	-	-	-	-	-	stby
Class 5												
CMD35	-	-	-	-	tran	-	-	-	-	-	-	stby
CMD36	-	-	-	-	tran	-	-	-	-	-	-	stby
CMD38	-	-	-	-	prg	-	-	-	-	-	-	stby
Class 7												
CMD16	see class 2											
CMD42	-	-	-	-	rcv	-	-	-	-	-	-	stby
Class 8												
CMD55	-	-	-	stby	tran	data	btst	rcv	prg	dis	-	irq
CMD56; RD/WR = 0	-	-	-	-	rcv	-	-	-	-	-	-	stby
CMD56; RD/WR = 1	-	-	-	-	data	-	-	-	-	-	-	stby
Class 9												
CMD39 CMD40	MMCA Optional Command, currently not supported											
Class 10 - 11												
CMD41; CMD43...CMD54, CMD57-CMD59	Reserved											
CMD60...CMD63	Reserved for Manufacturer											

Table 6-18 : Card State Transition Table

6.9 Responses

All responses are sent via the command line CMD. The response transmission always starts with the left bit of the bitstring corresponding to the response codeword. The code length depends on the response type.
A response always starts with a start bit (always '0'), followed by the bit indicating the direction of transmission (card = '0'). A value denoted by 'x' in the tables below indicates a variable entry. All responses except for the type R3 (see below) are protected by a CRC (see Chapter 7.2 for the definition of CRC7). Every command codeword is terminated by the end bit (always '1').

There are five types of responses. Their formats are defined as follows:

• **R1** (normal response command): code length 48 bit. The bits 45:40 indicate the index of the command to be responded to, this value being interpreted as a binary coded number (between 0 and 63). The status of the card is coded in 32 bits. The card status is described in Chapter

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'0'	x	x	x	'1'
Description	start bit	transmission bit	command index	card status	CRC7	end bit

Table 6-19 : Response R1

• **R1b** is identical to R1 with an optional busy signal transmitted on the data line DAT0. The card may become busy after receiving these commands based on its state prior to the command reception. Refer to Section for detailed description and timing diagrams.
• **R2** (CID, CSD register): code length 136 bits. The contents of the CID register are sent as a response to the commands CMD2 and CMD10. The contents of the CSD register are sent as a response to CMD9. Only the bits [127...1] of the CID and CSD are transferred, the reserved bit [0] of these registers is replaced by the end bit of the response.

Bit position	135	134	[133:128]	[127:1]	0
Width (bits)	1	1	6	127	1
Value	'0'	'0'	'111111'	x	'1'
Description	start bit	transmission bit	check bits	CID or CSD register incl. internal CRC7	end bit

Table 6-20 : Response R2

• **R3** (OCR register): code length 48 bits. The contents of the OCR register is sent as a response to CMD1. The **level coding** is as follows: restricted voltage windows=LOW, card busy=LOW.

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'0'	'111111'	x	'1111111'	'1'
Description	start bit	transmission bit	check bits	OCR register	check bits	end bit

Table 6-21 : Response R3

• *R4 and R5 : responses are not supported.*

6.10 Card Status

The response format R1 contains a 32-bit field named *card status*. This field is intended to transmit the card's status information.
Three different attributes are associated with each one of the card status bits:

• Bit type.
Two types of card status bits are defined:
(a) **Error bit**. Signals an error condition detected by the card. These bits are cleared as soon as the response (reporting the error) is sent out.
(b) **Status bit**. These bits serve as information fields only, and do not alter the execution of the command being responded to. These bits are set and cleared in accordance with the card status.
The "Type" field of Table 6-22 defines the type of each bit in the card status register. The symbol "E" is used to denote an Error bit while the symbol "S" is used to denote a Status bit.

• Detection mode of Error bits.
Exceptions are detected by the card either during the command interpretation and validation phase (Response Mode) or during command execution phase (Execution Mode). Response mode exceptions are reported in the response to a STOP_TRANSMISSION command used to terminate the operation or in the response to a GET_STATUS command issued after the operation is completed.
The "Det Mode" field of Table 6-22 defines the detection mode of each bit in the card status register. The symbol "R" is used to denote a Response Mode detection while the symbol "X" is used to denote an Execution Mode detection.
When an error bit is detected in "R" mode the card will report the error in the response to the command that raised the exception. The command will not be executed and the associated state transition will not take place. When an error is detected in "X" mode the execution is terminated. The error will be reported in the response to the next command.
The ADDRESS_OUT_OF_RANGE and ADDRESS_MISALIGN exceptions may be detected both in Response and Execution modes. The conditions for each one of the modes are explicitly defined in the Table 6-22.

• Clear Condition:

A - According to the card current state
B - Always related to the previous command. Reception of a valid command will clear it (with a delay of one command)
C - Clear by read.

Bits	Identifier	Type	Det-Mode	Value	Descript	Clear Cond
31	ADDRESS_OUT_OF_RANGE	E	R	'0'= no error '1'= error	The command's address argument was out of the allowed range for this card.	C
			X		A multiple block or stream read/write operation is (although started in a valid address) attempting to read or write beyond the card capacity	
30	ADDRESS_MISALIGN	E	R	'0'= no error '1'= error	The command' s address argument (in accordance with the currently set block length) positions the first data block misaligned to the card physical blocks.	C
			X		A multiple block read/write operation (although started with a valid address/block-length combination) is attempting to read or write a data block which does not align with the physical blocks of the card.	
29	BLOCK_LEN_ERROR	E	R	'0'= no error '1'= error	Either the argument of a SET_BLOCKLEN command exceeds the maximum value allowed for the card, or the previously defined block length is illegal for the current command (e.g. the host issues a write command, the current block length is smaller than the card's maximum and write partial blocks is not allowed)	C
28	ERASE_SEQ_ERROR	E	R	'0'= no error '1'= error	An error in the sequence of erase commands occurred.	C
27	ERASE_PARAM	E	X	'0'= no error '1'= error	An invalid selection of erase groups for erase occurred.	C
26	WP_VIOLATION	E	X	'0'= no error '1'= error	Attempt to program a write protected block.	C
25	CARD_IS_LOCKED	S	R	'0' = card unlocked '1' = card locked	When set, signals that the card is locked by the host	A
24	LOCK_UNLOCK_FAILED	E	X	'0' = no error '1' = error	Set when a sequence or password error has been detected in lock/unlock card command	C
23	COM_CRC_ERROR	E	R	'0'= no error '1'= error	The CRC check of the previous command failed.	B
22	ILLEGAL_COMMAND	E	R	'0'= no error '1'= error	Command not legal for the card state	B
21	CARD_ECC_FAILED	E	X	'0'= success '1'= failure	Card internal ECC was applied but failed to correct the data.	C
20	CC_ERROR	E	R	'0'= no error '1'= error	(Undefined by the standard) A card error occurred, which is not related to the host command.	C

Table 6-22 : Card Status

Bits	Identifier	Type	Det-Mode	Value	Description	Clear Cond
19	ERROR	E	X	'0'= no error '1'= error	(Undefined by the standard) A generic card error related to the (and detected during) execution of the last host command (e.g. read or write failures).	C
18 - 17	Not applicable. This bit is always set to 0.					
16	CID/ CSD_OVERWRITE	E	X	'0'= no error '1'= error	Can be either one of the following errors: - The CID register has been already written and can not be overwritten - The read only section of the CSD does not match the card content. - An attempt to reverse the copy (set as original) or permanent WP (unprotected) bits was made.	C
15	WP_ERASE_SKIP	E	X	'0'= not protected '1'= protected	Only partial address space was erased due to existing write protected blocks.	C
14	Reserved, must be set to 0					
13	ERASE_RESET	E	R	'0'= cleared '1'= set	An erase sequence was cleared before executing because an out of erase sequence command was received (commands other than CMD35, CMD36, CMD38 or CMD13	C
12:9	CURRENT_STATE	S	R	0 = Idle 1 = Ready 2 = Ident 3 = Stby 4 = Tran 5 = Data 6 = Rcv 7 = Prg 8 = Dis 9 = Btst 10-15 = reserved	The state of the card when receiving the command. If the command execution causes a state change, it will be visible to the host in the response on the next command. The four bits are interpreted as a binary number between 0 and 15.	B
8	READY_FOR_DATA	S	R	'0'= not ready '1'= ready	Corresponds to buffer empty signalling on the bus	A
7	SWITCH_ERROR	E	X	'0'= no error '1'= switch error	If set, the card did not switch to the expected mode as requested by the SWITCH command	C
6	Reserved					
5	Not applicable. This bit is always set to 0.					
4	Reserved					
3:2	Reserved for Application Specific commands					
1:0	Reserved for Manufacturer Test Mode					

Table 6-22 : Card Status

6.11 Memory Array Partitioning

The basic unit of data transfer to/from the MultiMediaCard is one byte. All data transfer operations which require a block size always define block lengths as integer multiples of bytes. Some special functions need other partition granularity. For block oriented commands, the following definition is used:

• **Block**: is the unit which is related to the block oriented read and write commands. Its size is the number of bytes which will be transferred when one block command is sent by the host. The size of a block is either programmable or fixed. The information about allowed block sizes and the programmability is stored in the CSD.

For R/W cards, special erase and write protect commands are defined:

The granularity of the erasable units is the **Erase Group:** The smallest number of consecutive write blocks which can be addressed for erase. The size of the Erase Group is card specific and stored in the CSD.

The granularity of the Write Protected units is the **WP-Group:** The minimal unit which may be individually write protected. Its size is defined in units of erase groups. The size of a WP-group is card specific and stored in the CSD.

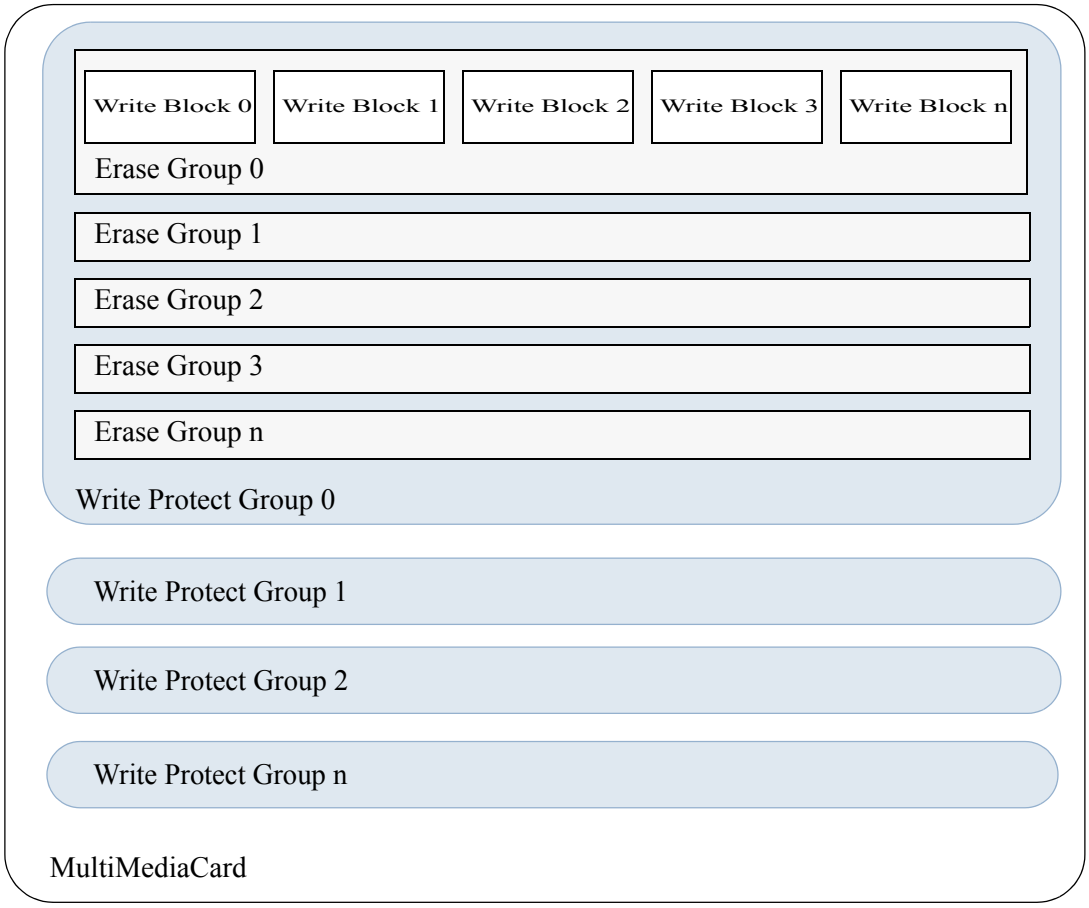


Figure 6-5 : Memory Array Partitioning

6.12 Timing Diagrams

All timing diagrams use the following schematics and abbreviations:

Symbol	Definition
S	Start bit (= '0')
T	Transmitter bit (Host = '1', Card = '0')
P	One-cycle pull-up (= '1')
E	End bit (='1')
Z	High impedance state (-> = '1')
X	Driven value, '1' or '0'
D	Data bits
*	Repetition
CRC	Cyclic redundancy check bits (7 bits)
	Card active
	Host active

Table 6-24 : Timing Diagram Symbols

The difference between the P-bit and Z-bit is that a P-bit is actively driven to HIGH by the card respectively host output driver, while Z-bit is driven to (respectively kept) HIGH by the pull-up resistors R_{CMD} respectively R_{DAT}. Actively-driven P-bits are less sensitive to noise.
All timing values are defined in Table 6-25.

6.12.1 Command and Response

Both host command and card response are clocked out with the rising edge of the host clock.

- **Card identification and card operation conditions timing**

The card identification (CMD2) and card operation conditions (CMD1) timing are processed in the open-drain mode. The card response to the host command starts after exactly N_{ID} clock cycles.

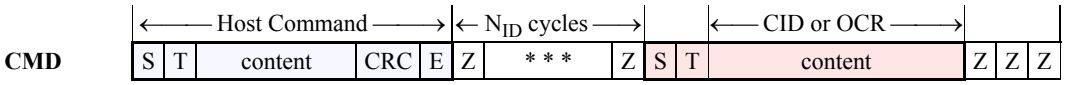


Figure 6-7 : Identification Timing (Card Identification Mode)

- **Assign a card relative address**

The SET_RCA (CMD 3) is also processed in the open-drain mode. The minimum delay between the host command and card response is N_{CR} clock cycles.

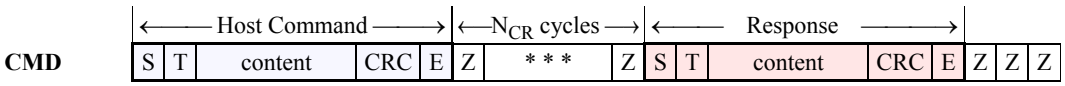


Figure 6-8 : SET_RCA Timing (Card Identification Mode)

- **Data transfer mode.**

After a card receives its RCA it will switch to data transfer mode. In this mode the CMD line is driven with push-pull drivers. The command is followed by a period of two Z bits (allowing time for direction switching on the bus) and than by P bits pushed up by the responding card. This timing diagram is relevant for all responded host commands except CMD1,2,3:

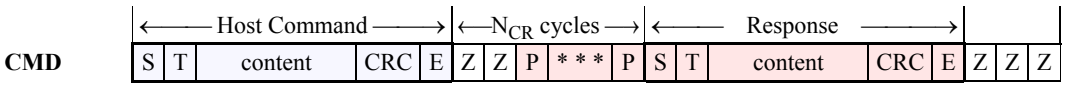


Figure 6-9 : Command Response Timing (Data Transfer Mode)

- **R1b Responses**

Some commands, like CMD6, may assert the BUSY signal after responding with R1. If the busy signal is asserted, it is done two clock cycles after the end bit of the command. the DAT0 line is driven low, DAT1-DAT7 lines are driven by the card though their value is not relevant.

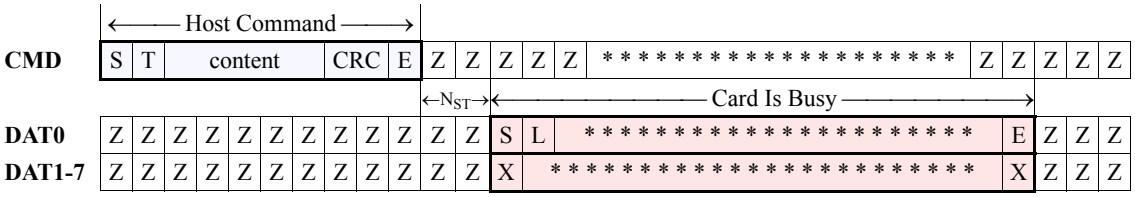


Figure 6-10 : R1b Response Timing

- **Last Card Response - Next Host Command Timing**

After receiving the last card response, the host can start the next command transmission after at least N_{RC} clock cycles. This timing is relevant for any host command.

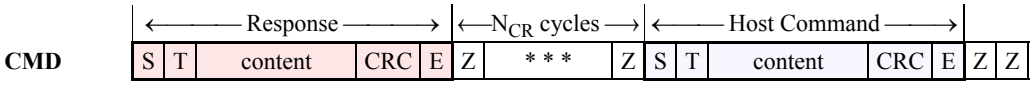


Figure 6-11 : Timing Response End To Next Command Start (Data Transfer Mode)

- **Last Host Command - Next Host Command Timing**

After the last command has been sent, the host can continue sending the next command after at least N_{CC} clock periods. If the ALL_SEND_CID command is not responded by the card after N_{ID} + 1 clock periods, the host can conclude there is no card present in the bus.

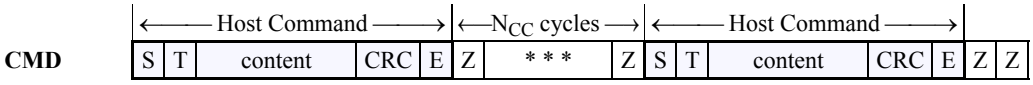


Figure 6-12 : Timing Of Command Sequences (All Modes)

6.13 Data Read

- **Single Block Read**

The host selects one card for data read operation by CMD7, and sets the valid block length for block oriented data transfer by CMD16. The basic bus timing for a read operation is given in Figure 6-12. The sequence starts with a single block read command (CMD17) which specifies the start address in the argument field. The response is sent on the CMD line as usual.

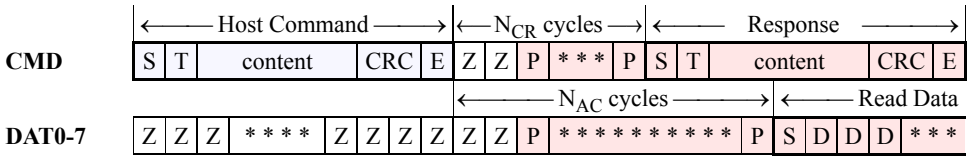


Figure 6-12 : Single Block Read Timing

Data transmission from the card starts after the access time delay N_{AC} beginning from the end bit of the read command. After the last data bit, the CRC check bits are suffixed to allow the host to check for transmission errors.

- **Multiple Block Read**

In multiple block read mode, the card sends a continuous flow of data blocks following the initial host read command. The data flow is terminated by a stop transmission command (CMD12). Figure 6-13 describes the timing of the data blocks and Figure 6-14 the response to a stop command. The data transmission stops two clock cycles after the end bit of the stop command.

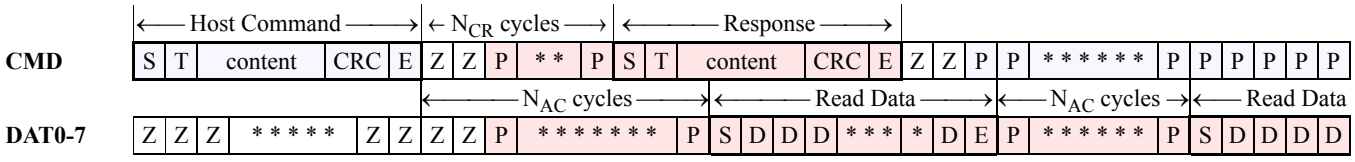


Figure 6-13 : Multiple Block Read Timing

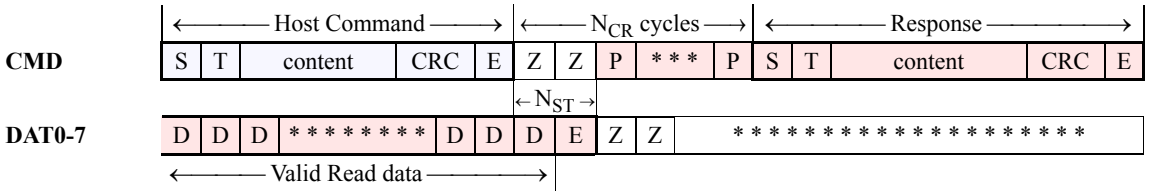


Figure 6-14 : Stop Command Timing (CMD12, Data Transfer Mode)

6.14 Data Write

- **Single Block Write**

The host selects the card for data write operation by CMD7.
The host sets the valid block length for block oriented data transfer (a stream write mode is also available) by CMD16.
The basic bus timing for a write operation is given in Figure 6-15. The sequence starts with a single block write command (CMD24) which determines (in the argument field) the start address. It is responded by the card on the CMD line as usual. The data transfer from the host starts N_{WR} clock cycles after the card response was received.
The data is suffixed with CRC check bits to allow the card to check it for transmission errors. The card sends back the CRC check result as a CRC status token on DAT0. In the case of transmission error, occurring on any of the active data lines, the card sends a negative CRC status (‘101’) on DAT0. In the case of successful transmission, over all active data lines, the card sends a positive CRC status (‘010’) on DAT0 and starts the data programming procedure

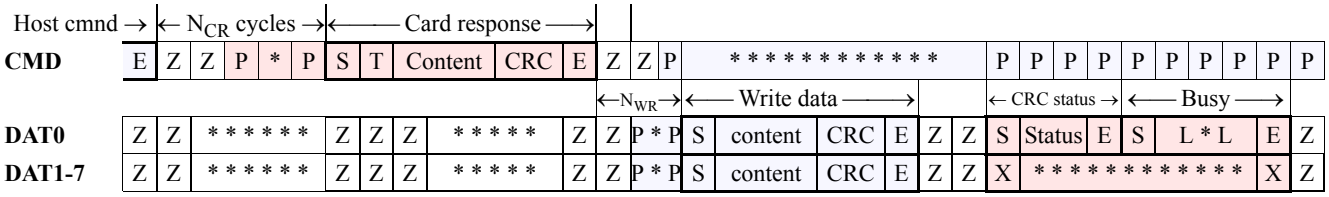


Figure 6-15 : Block Write Command Timing

If the card does not have a free data receive buffer, the card indicates this condition by pulling down the data line DAT0 to LOW. The card stops pulling down DAT0 as soon as at least one receive buffer for the defined data transfer block length becomes free. This signalling does not give any information about the data write status which must be polled by the host.

- **Multiple Block Write**

In multiple block write mode, the card expects continuous flow of data blocks following the initial host write command. The data flow is terminated by a stop transmission command (CMD12). Figure 6-16 describes the timing of the data blocks with and without card busy signal.

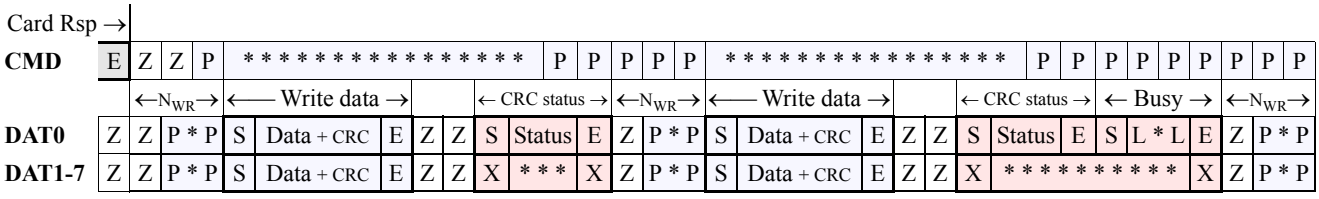


Figure 6-16 : Multiple Block Write Timing

The stop transmission command works similar as in the read mode. Figure 6-17 to Figure 6-20 describe the timing of the stop command in different card states.

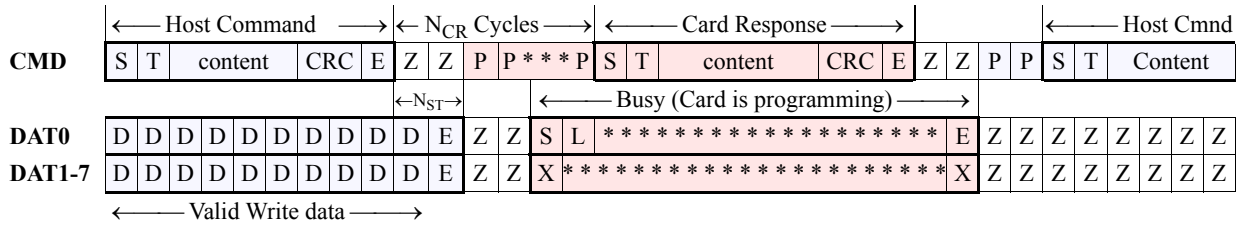
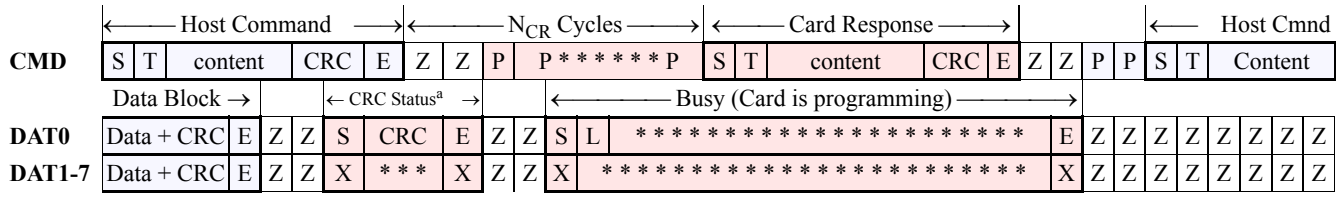


Figure 6-17 : Stop Transmission During Data Transfer From The Host

The card will treat a data block as successfully received and ready for programming only if the CRC data of the block was validated and the CRC status tokens sent back to the host. Figure 6-18 is an example of an interrupted (by a host stop command) attempt to transmit the CRC status block. The sequence is identical to all other stop transmission examples. The end bit of the host command is followed, on the data lines, with one more data bit, an end bit and two Z clocks for switching the bus direction. The received data block, in this case is considered incomplete and will not be programmed.



a.The card CRC status response is interrupted by the host.

Figure 6-18 : Stop Transmission During CRC Status Transfer From The Card

All previous examples dealt with the scenario of the host stopping the data transmission during an active data transfer. The following two diagrams describe a scenario of receiving the stop transmission between data blocks. In the first example the card is busy programming the last block while in the second the card is idle. However, there are still unprogrammed data blocks in the input buffers. These blocks are being programmed as soon as the stop transmission command is received and the card activates the busy signal.

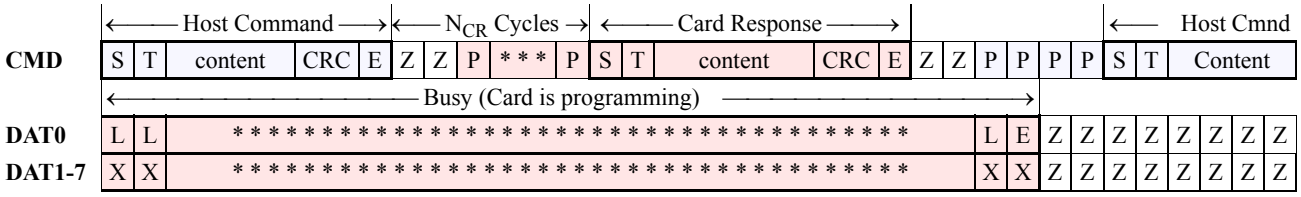


Figure 6-19 : Stop Transmission After Last Data Block. Card Is Busy Programming.

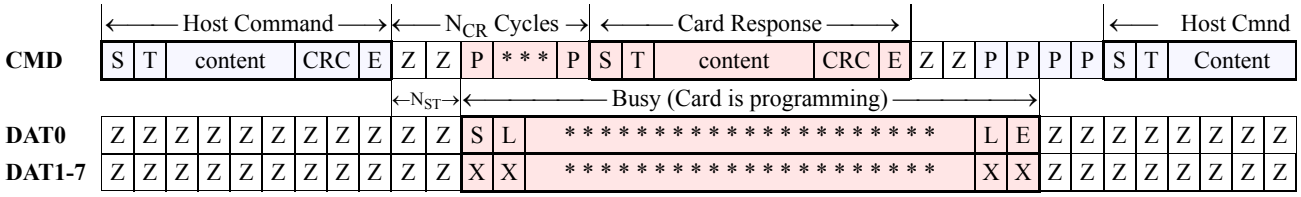


Figure 6-20 : Stop Transmission After Last Data Block. Card Becomes Busy.

- **Erase, Set and Clear Write Protect Timing**

The host must first select the erase groups to be erased using the erase start and end command (CMD35, CMD36). The erase command (CMD38), once issued, will erase all selected erase groups. Similarly, set and clear write protect commands start a programming operation as well. The card will signal “busy” (by pulling the DAT0 line low) for the duration of the erase or programming operation. The bus transaction timings are identical to the variation of the stop transmission described in Figure 6-20.

- **Reselecting a busy card**

When a busy card which is currently in the dis state is reselected it will reinstate its busy signaling on the data line DAT0. The timing diagram for this command / response / busy transaction is given in Figure 6-20.

6.15 Bus Test Procedure Timing

After reaching the Tran-state a host can initiate the Bus Testing procedure. If there is no response to the CMD19 sent by the host, the host should read the status from the card with CMD13. If there was no response to CMD19, the host may assume that this function is not supported by the card.

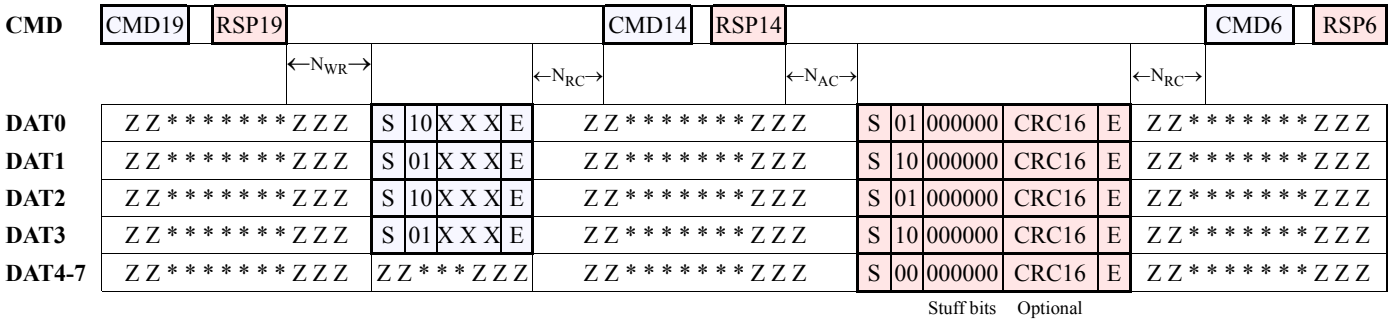


Figure 6-21: 4 bit System Bus Testing Procedure

6.16 Timing Values

Symbol	Min	Max	Unit
N_{CR}	2	64	clock cycles
N_{ID}	5	5	clock cycles
N_{AC}	2	10 * (TAAC *F_{OP} + 100 * NSAC)[a]	clock cycles
N_{RC}	8	-	clock cycles
N_{CC}	8	-	clock cycles
N_{WR}	2	-	clock cycles
N_{ST}	2	2	clock cycles

a. F_{OP} is the MMC clock frequency the host is using for the read operation.
Following is a calculation example:
CSD value for TAAC is 0x26; this is equal to 1.5mSec;
CSD value for NSAC is 0;
The host frequency F_{OP} is 10MHz

$$N_{AC} = 10 \times (1.5 \times 10^{-3} \times 10 \times 10^{6} + 0) = 150,000 \text{ clock cycles}$$

7. SPI Mode

Introduction

The SPI mode consists of a secondary, optional communication protocol which is offered by Flash-based MultiMediaCards. This mode is a subset of the MultiMediaCard protocol, designed to communicate with a SPI channel, commonly found in Motorola's (and lately a few other vendors') microcontrollers. The interface is selected during the first reset command after power up (CMD0) and cannot be changed once the part is powered on.
The SPI standard defines the physical link only, and not the complete data transfer protocol. The MultiMediaCard SPI implementation uses a subset of the MultiMediaCard protocol and command set. It is intended to be used by systems which typically require one card and have lower data transfer rates (compared to MultiMediaCard protocol based systems). From the application point of view, the advantage of the SPI mode is the capability of using an off-the-shelf host, hence reducing the design-in effort to minimum. The disadvantage is the loss of performance of the SPI mode versus MultiMediaCard mode (lower data transfer rate, hardware CS, etc.).

7.1 SPI Interface Concept

The Serial Peripheral Interface (SPI) is a general purpose synchronous serial interface originally found on certain Motorola microcontrollers. A virtually identical interface can now be found on certain TI and SGS Thomson microcontrollers as well. The MultiMediaCard SPI interface is compatible with SPI hosts available on the market. As in any other SPI device, the MultiMediaCard SPI channel consists of the following four signals:

CS: Host to card Chip Select signal.
CLK:Host to card clock signal
DataIn:Host to card data signal.
DataOut: Card to host data signal.

Another SPI common characteristic is byte transfers, which is implemented in the card as well. All data tokens are multiples of bytes (8 bit) and always byte aligned to the CS signal.

7.2 SPI Bus Topology

The card identification and addressing methods are replaced by a hardware Chip Select (CS) signal. There are no broadcast commands. For every command, a card (slave) is selected by asserting (active low) the CS signal (see Figure).
The CS signal must be continuously active for the duration of the SPI transaction (command, response and data). The only exception occurs during card programming, when the host can de-assert the CS signal without affecting the programming process.
The bidirectional CMD and DAT lines are replaced by unidirectional *dataIn* and *dataOut* signals.
The MultiMediaCard pin assignment in SPI mode (compared to MultiMediaCard mode) is given in Table 7-1.

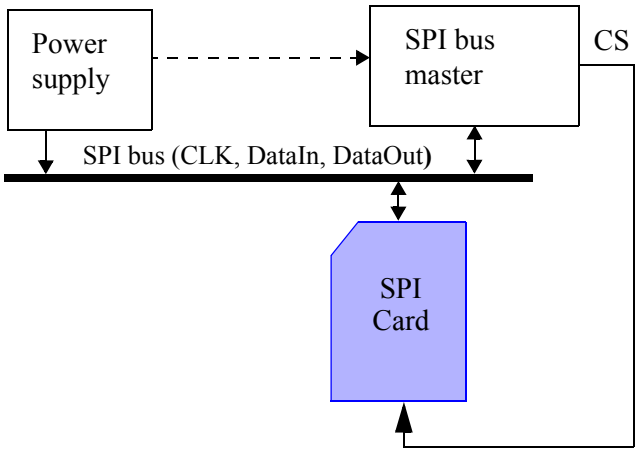


Figure 7-1 : MultiMediaCard Bus System

Pin #	MultiMediaCard Mode			SPI Mode		
	Name	Type[a]	Description	Name	Type	Description
1	DAT3	I/O/PP	Data	CS	I	Chip Select (neg true)
2	CMD	I/O/PP/OD	Command/Response	DI	I/PP	Data In
3	V_{SS1}	S	Supply voltage ground	VSS	S	Supply voltage ground
4	V_{DD}	S	Supply voltage	VDD	S	Supply voltage
5	CLK	I	Clock	SCLK	I	Clock
6	V_{SS2}	S	Supply voltage ground	VSS2	S	Supply voltage ground
7	DAT0	I/O/PP	Data	DO	O/PP	Data Out
8	DAT1	I/O/PP	Data	Not used		
9	DAT2	I/O/PP	Data	Not used		
10	DAT4	I/O/PP	Data	Not used		
11	DAT5	I/O/PP	Data	Not used		
12	DAT6	I/O/PP	Data	Not used		
13	DAT7	I/O/PP	Data	Not used		

Table 7-1 : SPI Interface Pin Configuration

a. S: power supply; I: input; O: output; PP: push-pull; OD: open-drain; NC: Not connected (or logical high)

7.3 Card Registers in SPI Mode

The register usage in SPI mode is summarized in Table 7-2. Most of them are inaccessible.

Name	Available in SPI mode	Width [Bytes]	Description
CID	Yes	16	Card identification data (serial number, manufacturer ID, etc.)
RCA	No		
DSR	No		
CSD	Yes	16	Card-specific data, information about the card operation conditions.
EXT_CSD	Yes	512	Extended Card-specific data, information about the card supported properties and configured modes
OCR	Yes	32	Operation condition register.

Table 7-2 : MultiMediaCard Registers In SPI Mode

7.4 SPI Bus Protocol

While the MultiMediaCard channel is based on command and data bit streams which are initiated by a start bit and terminated by a stop bit, the SPI channel is byte oriented. Every command or data block is built of 8-bit bytes and is byte aligned to the CS signal (i.e. the length is a multiple of 8 clock cycles).
Similar to the MultiMediaCard protocol, the SPI messages consist of command, response and data-block tokens. All communication between host and card is controlled by the host (master). The host starts every bus transaction by asserting the CS signal low.
The response behavior in the SPI mode differs from the MultiMediaCard mode in the following three aspects:

- The selected card always responds to the command.
- Additional (8, 16 & 40 bit) response structures are used
- When the card encounters a data retrieval problem, it will respond with an error response (which replaces the expected data block) rather than by a time-out, as in the MultiMediaCard mode.

Only single and multiple block read/write operations are supported in SPI mode (sequential mode is not supported). In addition to the command response, every data block sent to the card during write operations will be responded to with a special data response token. A data block may be as big as one card write block and as small as a single byte. Partial block read/write operations are enabled by card options specified in the CSD register.

7.5 Mode Selection

The MultiMediaCard wakes up in the MultiMediaCard mode. It will enter SPI mode if the CS signal is asserted (negative) during the reception of the reset command (CMD0). Selecting SPI mode is not restricted to *Idle* state (the state the card enters after power up) only. Every time the card receives CMD0, including while in *Inactive* state, CS signal is sampled. If the card recognizes that the MultiMediaCard mode is required (CS signal is high), it will not respond to the command and remain in the MultiMediaCard mode. If SPI mode is required (CS signal is low), the card will switch to SPI and respond with the SPI mode R1 response.
The only way to return to the MultiMediaCard mode is by a power cycle (turn the power off an on). In SPI mode, the MultiMediaCard protocol state machine is not observed. All the MultiMediaCard commands supported in SPI mode are always available.

7.6 Bus Transfer Protection

Every MultiMediaCard token transferred on the bus is protected by CRC bits. In SPI mode, the MultiMediaCard offers a non-protected mode which enables systems built with reliable data links to exclude the hardware or firmware required for implementing the CRC generation and verification functions. In the non-protected mode, the CRC bits of the command, response and data tokens are still required in the tokens. However, they are defined as 'don't care' for the transmitter and ignored by the receiver.
The SPI interface is initialized in the non-protected mode. However, the RESET command (CMD0), which is used to switch the card to SPI mode, is received by the card while in MultiMediaCard mode and, therefore, must have a valid CRC field.
Since CMD0 has no arguments, the content of all the fields, including the CRC field, are constants and need not be calculated in run time. A valid reset command is:

0x40, 0x0, 0x0, 0x0, 0x0, 0x95

The host can turn the CRC option on and off using the CRC_ON_OFF command (CMD59).

7.7 Data Read

The SPI mode supports single and multiple block read operations. The main difference between SPI and MultiMediaCard modes is that the data and the response are both transmitted to the host on the DataOut signal (refer to Figure 7-2 and Figure 7-3). Therefore the card response to the STOP_COMMAND may cut-short and replace the last data block.

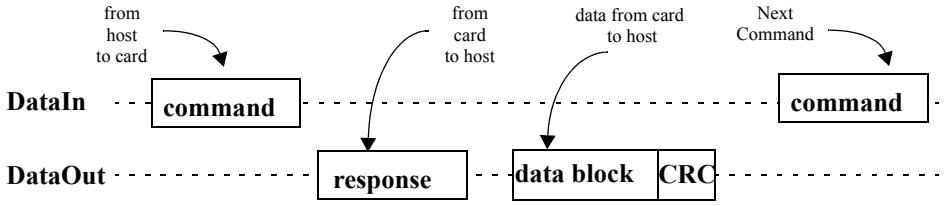


Figure 7-2 : SPI Single Block Read Operation

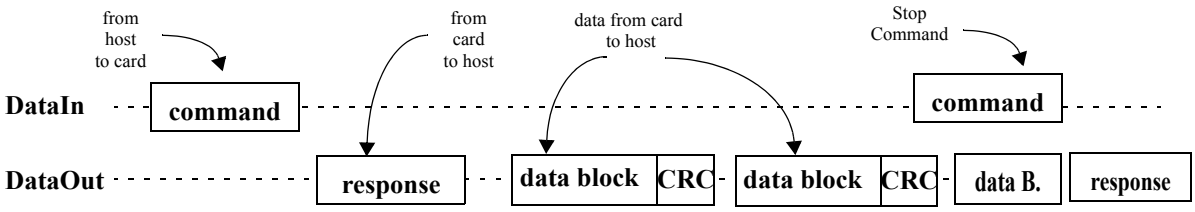


Figure 7-3 : SPI Multiple Block Read Operation

The basic unit of data transfer is a block whose maximum size is defined in the CSD (READ_BL_LEN).If READ_BL_PARTIAL is set, smaller blocks whose starting and ending address are entirely contained within one physical block (as defined by READ_BL_LEN) may also be transmitted. A CRC is appended to the end of each block ensuring data transfer integrity. CMD17 (READ_SINGLE_BLOCK) initiates a single block read. CMD18 (READ_MULTIPLE_BLOCK) starts a transfer of several consecutive blocks. Two types of multiple block read transactions are defined (the host can use either one at any time):

• Open-ended Multiple block read
The number of blocks for the read multiple block operation is not defined. The card will continuously transfer data blocks until a stop transmission command is received.

• Multiple block read with pre-defined block count
The card will transfer the requested number of data blocks and terminate the transaction. Stop command is not required at the end of this type of multiple block read, unless terminated with an error. In order to start a multiple block read with pre-defined block count the host must use the SET_BLOCK_COUNT command (CMD23) immediately preceding the READ_MULTIPLE_BLOCK (CMD18) command. Otherwise the card will start an open-ended multiple block read which can be stopped using the STOP_TRANSMISION command.
If the host provides an out of range address as an argument to either CMD17 or CMD18, or the currently defined block length is illegal for a read operation, the card will reject the command and respond with the ADDRESS_OUT_OF_RANGE or BLOCK_LEN_ERROR bit set, respectively.
If the host sets the argument of the SET_BLOCK_COUNT command (CMD23) to all 0s, then the command is accepted; however, a subsequent read will follow the open-ended multiple block read protocol (STOP_TRANSMISSION command - CMD12 - is required).

The host can abort reading at any time, within a multiple block operation, regardless of the its type. Transaction abort is done by sending the stop transmission command.
If the host provides an out of range address as an argument to either CMD17 or CMD18, or the currently defined block length is illegal for a read operation, the card will reject the command and respond with the ADDRESS_OUT_OF_RANGE or BLOCK_LEN_ERROR bit set, respectively.
If the host sets the argument of the SET_BLOCK _COUNT command (CMD23) to all 0s, then the command is accepted: however , a subsequent read will follow the open-ended multiple block read protocol (STOP_TRANSMISSION command - CMD12 - is required).
In case of a data retrieval error (e.g. out of range, address misalignment, internal error, etc.) detected during data transfer, the card will not transmit any data. Instead (as opposed to MultiMediaCard mode where the card times out), a special data error token will be sent to the host. Figure 7-4 shows a single block read operation which terminates with an error token rather than a data block.

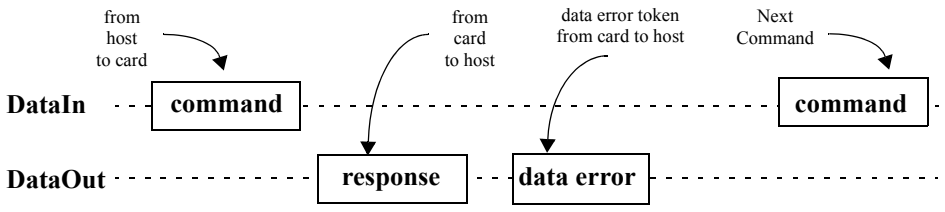


Figure 7-4 : SPI Read Operation - Data Error

Multiple block read operation can be terminated the same way, the error token replacing a data block anywhere in the sequence. The host must than abort the operation by sending the stop transmission command.
If the host sends a stop transmission command after the card transmitted the last block of a multiple block read with a pre-defined number of blocks, it will be responded to as an illegal command.

If the host uses partial blocks whose accumulated length is not block aligned, and block misalignment is not allowed, the card shall detect a block misalignment error condition during the transmission of the first misaligned block and the content of the further transferred bits is undefined. As the host sends CMD12, the card will respond with the ADDRESS_MISALIGN bit set.

7.8 Data Write

The SPI mode supports single block and Multiple block write commands. Upon reception of a valid write command (CMD24 or CMD25), the card will respond with a response token and will wait for a data block to be sent from the host. CRC suffix, block length and start address restrictions are (with the exception of the CSD parameter WRITE_BL_PARTIAL controlling the partial block write option) identical to the read operation (see Figure 7-5). If a CRC error is detected it will be reported in the data-response token and the data block will not be programmed.

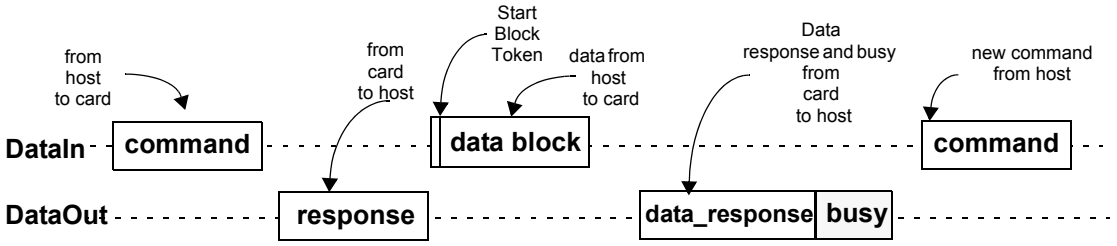


Figure 7-5 : SPI Single Block Write Operation

Every data block has a prefix of 'Start Block' token (one byte).
After a data block has been received, the card will respond with a data-response token. If the data block has been received without errors, it will be programmed. As long as the card is busy programming, a continuous stream of busy tokens will be sent to the host (effectively holding the DataOut line low).
In Multiple Block write operation the stop transmission will be done by sending 'Stop Tran' token instead of 'Start Block' token at the beginning of the next block.

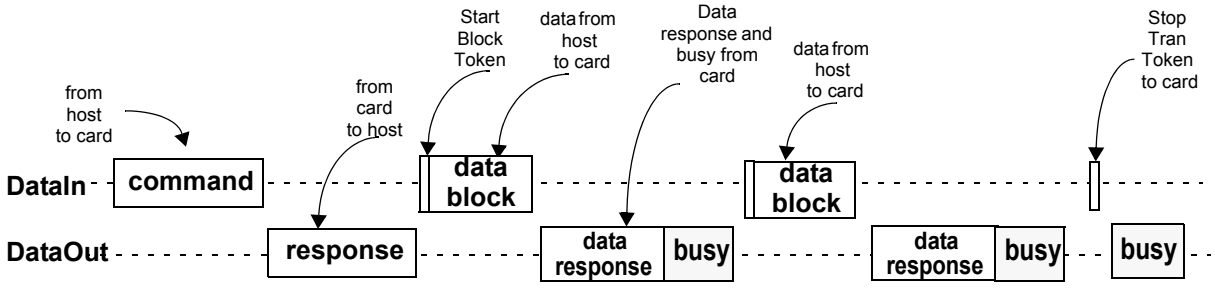


Figure 7-6 : SPI Multiple Block Write Operation

Two types of multiple block write transactions, identical to the multiple block read, are defined (the host can use either one at any time):

- Open-ended Multiple block write

The number of blocks for the write multiple block operation is not defined. The card will continuously accept and program data blocks until a 'Stop Tran' token is received.

- Multiple block write with pre-defined block count

The card will accept the requested number of data blocks and terminate the transaction. 'Stop tran' token is not required at the end of this type of multiple block write, unless terminated with an error. In order to start a multiple block write with pre-defined block count the host must use the SET_BLOCK_COUNT command (CMD23) immediately preceding the WRITE_MULTIPLE_BLOCK (CMD25) command. Otherwise the card will start an open-ended multiple block write which can be stopped using the 'Stop tran' token.

The host can abort writing at any time, within a multiple block operation, regardless of the its type. Transaction abort is done by sending the 'Stop tran' token. If a multiple block write with pre-defined block count is aborted, the data in the remaining blocks is not defined.
If the host provides an out of range address as an argument to either CMD17 or CMD18, or the currently defined block length is illegal for a read operation, the card will reject the command, remain in *Tran* state and respond with the ADDRESS_OUT_OF_RANGE or BLOCK_LEN_ERROR bit set, respectively.
If the host sets the argument of the SET_BLOCK_COUNT command (CMD23) to all 0s, then the command is accepted; however, a subsequent write will follow the open-ended multiple block write protocol (STOP_TRANSMISSION command - CMD12 - is required).
If the card detects a CRC error or a programming error (e.g. write protect violation, out of range, address misalignment, internal error, etc.) during a multiple block write operation (both types) it will report the failure in the data-response token and ignore any further incoming data blocks. The host must than abort the operation by sending the 'Stop Tran' token.
If the host uses partial blocks whose accumulated length is not block aligned, and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card shall detect the block misalignment error during the reception of the first misaligned block, abort the write operation, and ignore all further incoming data. The host must abort the operation by sending the 'Stop Tran' token, to which the card will respond with the ADDRESS_MISALIGN bit set.
Once the programming operation is completed (either successfully or with an error), the host must check the results of the programming (or the cause of the error if already reported in the data-response token) using the SEND_STATUS command (CMD13).
If the host sends a 'Stop Trans' token after the card received the last data block of a multiple block operation with pre-defined number of blocks, it will be interpreted as the beginning of an illegal command and responded accordingly.
While the card is busy, resetting the CS signal will not terminate the programming process. The card will release the DataOut line (tri-state) and continue with programming. If the card is reselected before the programming is finished, the DataOut line will be forced back to low and all commands will be rejected.
Resetting a card (using CMD0) will terminate any pending or active programming operations. This may destroy the data formats on the card. It is in the responsibility of the host to prevent it.

7.9 Erase & Write Protect Management

The erase and write protect management procedures in the SPI mode are identical to those of the MultiMediaCard mode. While the card is erasing or changing the write protection bits of the predefined erase groups list, it will be in a busy state and hold the DataOut line low. Figure 7-7 illustrates a 'no data' bus transaction with and without busy signalling.

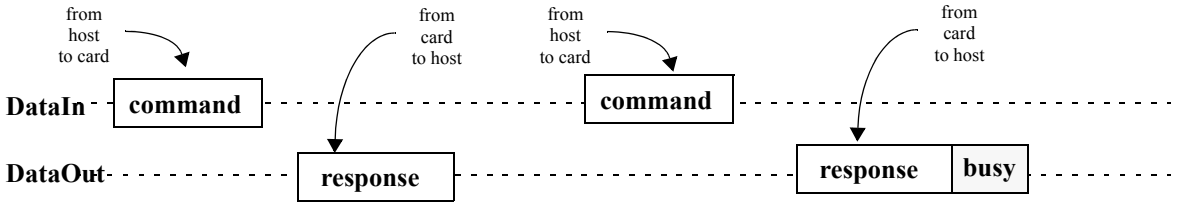


Figure 7-7 : SPI 'No data' Operations

7.10 Read CID/CSD Registers

Unlike the MultiMediaCard protocol (where the register contents is sent as a command response), reading the contents of the CSD and CID registers in SPI mode is a simple read-block transaction. The card will respond with a standard response token (see Figure) followed by a data block of 16 bytes suffixed with a 16 bit CRC.
The data time out for the CSD command cannot be set to the card TAAC since this value is stored in the CSD.
Refer to Section 7.23.2 for detailed timing. For consistency, read CID transaction is identical to read CSD.

7.11 Reset Sequence

The MultiMediaCard requires a defined reset sequence. After power on reset or CMD0 (software reset) the card enters an idle state. At this state the only legal host commands are CMD1 (SEND_OP_COND) and CMD58 (READ_OCR).
The host must poll the card (by repeatedly sending CMD1) until the 'in-idle-state' bit in the card response indicates (by being set to 0) that the card has completed its initialization processes and is ready for the next command.
In SPI mode, as opposed to MultiMediaCard mode, CMD1 has no operands and does not return the contents of the OCR register. Instead, the host may use CMD58 (available in SPI mode only) to read the OCR register. Furthermore, it is in the responsibility of the host to refrain from accessing a card that does not support its voltage range.
The usage of CMD58 is not restricted to the initializing phase only, but can be issued at any time. The host must poll the card (by repeatedly sending CMD1) until the 'in-idle-state' bit in the card response indicates (by being set to 0) that the card has completed its initialization processes and is ready for the next command.

7.12 Clock Control

The SPI bus clock signal can be used by the SPI host to put the card into energy saving mode or to control the data flow (to avoid under-run or over-run conditions) on the bus. The host is allowed to change the clock frequency or shut it down. There are a few restrictions the SPI host must follow:

- The bus frequency can be changed at any time (under the restrictions of maximum data transfer frequency, defined by the MultiMediaCards)
- It is an obvious requirement that the clock must be running for the MultiMediaCard to output data or response tokens. After the last SPI bus transaction, the host is required, to provide 8 (eight) clock cycles for the card to complete the operation before shutting down the clock. throughout this 8 clocks period the state of the CS signal is irrelevant. it can be asserted or de-asserted.

Following is a list of the various SPI bus transactions:
- A command / response sequence. 8 clocks after the card response end bit. The CS signal can be asserted or de-asserted during these 8 clocks.
- A read data transaction. 8 clocks after the end bit of the last data block.
- A write data transaction. 8 clocks after the CRC status token.
- The host is allowed to shut down the clock of a "busy" card. The MultiMediaCard will complete the programming operation regardless of the host clock. However, the host must provide a clock edge for the card to turn off its busy signal. Without a clock edge the MultiMediaCard (unless previously disconnected by de-asserting the CS signal) will force the dataOut line down, permanently.

7.13 Error Conditions

7.13.1 CRC and Illegal Command

All commands are (optionally) protected by CRC (cyclic redundancy check) bits. If the addressed MultiMediaCard's CRC check fails, the COM_CRC_ERROR bit will be set in the card's response. Similarly, if an illegal command has been received the ILLEGAL_COMMAND bit will be set in the card's response.
There are different kinds of illegal commands:

- Commands which belong to classes not supported by the MultiMediaCard (e.g. interrupt and I/O commands).
- Commands not allowed in SPI mode (e.g. CMD20 - write stream)
- Commands which are not defined (e.g. CMD47).

7.13.2 Read, Write, Erase And Force Erase Time-out Conditions

The time period after which a time-out condition for read/write/erase operations occurs are (card independent) 10 times longer than the typical access/program times for these operations given below. A card shall complete the command within this time period, or give up and return an error message. If the host does not get a response within the defined time-out it should assume the card is not going to respond any more and try to recover (e.g. reset the card, power cycle, reject, etc.).
The typical access and program times are defined as follows:

7.13.3 Read

The read access time is defined as the sum of the two times given by the CSD parameters TAAC and NSAC. These card parameters define the typical delay between the end bit of the read command and the start bit of the data block. This number is card dependent.

7.13.4 Write

The R2W_FACTOR field in the CSD is used to calculate the typical block program time obtained by multiplying the read access time by this factor. It applies to all write/erase commands (e.g. SET(CLEAR)_WRITE_PROTECT, PROGRAM_CSD(CID) and the block write commands).

7.13.5 Erase

The duration of an erase command will be (order of magnitude) the number of write blocks to be erased multiplied by the block write delay.

7.13.6 Force Erase

The Force Erase time-out is specified in Chapter 6.5.2

7.14 Read ahead in Multiple Block read operation

In Multiple Block read operations, in order to improve read performance, the card may fetch data from the memory array, ahead of the host. In this case, when the host is reading the last addresses of the memory, the card attempts to fetch data beyond the last physical memory address and generates an ADDRESS_OUT_OF_RANGE error.
Therefore, even if the host times the stop transmission command to stop the card immediately after the last byte of data was read, the card may already have generated the error, and it will show in the response to the stop transmission command. The host should ignore this error.

7.15 Memory Array Partitioning

Same as for MultiMediaCard mode.

7.16 Card Lock/unlock Operation

Usage of card lock and unlock commands in SPI mode is identical to MultiMediaCard mode. In both cases, the command response is of type R1b. After the busy signal clears, the host should obtain the result of the operation by issuing a GET_STATUS command. Please refer to Chapter 6.2.10 for details.

7.17 SPI Command Set

7.17.1 Command Format

All the MultiMediaCard commands are 6 bytes long. The command transmission always starts with the left bit of the bit-string corresponding to the command codeword. All commands are protected by a CRC. The commands and arguments are listed in Table 7-5.

Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bits)	1	1	6	32	7	1
Value	'0'	'1'	x	x	x	'1'
Description	start bit	transmission bit	command index	argument	CRC7	end bit

Table 7-3 : Command format in SPI Mode

7.17.2 Command Classes

As in MultiMediaCard mode, the SPI commands are divided into several classes (See Table 7-4). Each class supports a set of card functions. A MultiMediaCard will support the same set of optional command classes in both communication modes (there is only one command class table in the CSD register). The available comand classes, and the supported command for a specific class, however, are different in the MultiMediaCard and the SPI communication mode.

Card CMD Class (CCC)	Class Description	Supported commands																										
		0	1	6	8	9	10	12	13	16	17	18	23	24	25	27	28	29	30	35	36	38	42	55	56	58	59
class 0	Basic	+	+	+	+	+	+		+																	+	+
class 1	Not supported in SPI																										
class 2	Block read							+		+	+	+	+														
class 3	Not supported in SPI																										
class 4	Block write									+			+	+	+	+											
class 5	Erase																			+	+	+					
class 6	Write-protection																+	+	+								
class 7	Lock Card									+													+				
class 8	Application specific																							+	+		
class 9	Not supported in SPI																										
class 10-11	Reserved																										

7.17.3 Command Description

The following table provides a detailed description of the SPI mode commands. The responses are defined in Chapter 7.18. Table 7-5 lists all MultiMediaCard commands. A “yes” in the SPI mode column indicates that the command is supported in SPI mode. With these restrictions, the command class description in the CSD is still valid. If a command does not require an argument, the value of this field should be set to zero. The reserved commands are also reserved in MultiMediaCard mode.
The card can be switched to a new command space, using the SWITCH command, just as in MultiMediaCard mode; with the only limitation that in SPI mode the bus is always one bit wide.
The binary code of a command is defined by the mnemonic symbol. As an example, the content of the **command index** field is (binary) ‘000000’ for CMD0 and ‘100111’ for CMD39.

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD0	Yes	None	R1	GO_IDLE_STATE	Resets the MultiMediaCard
CMD1	Yes	None	R1	SEND_OP_COND	Activates the card’s initialization process
CMD2	No				
CMD3	No				
CMD4	No				
CMD5	reserved				

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD6	Yes	[31:26] Set to 0 [25:24] Access [23:16] Index [15:8] Value [7:3] Set to 0 [2:0] Cmd Set	R1b	SWITCH	Switches the mode of operation of the selected card and modifies the EXT_CSD registers. Access modes are: 00Command Set 01Set bits 10Clear bits 11Write Byte
CMD7	No				
CMD8	Yes	[31:0] stuff bits	R1	SEND_EXT_CSD	The card sends its EXT_CSD register as a block of data.
CMD9	Yes	None	R1	SEND_CSD	Asks the selected card to send its card-specific data (CSD)
CMD10	Yes	None	R1	SEND_CID	Asks the selected card to send its card identification (CID)
CMD11	No				
CMD12	Yes	None	R1	STOP_TRANSMISSION	Stop transmission on multiple block read
CMD13	Yes	None	R2	SEND_STATUS	Asks the selected card to send its status register
CMD14	This command is not applicable in SPI mode and the card should regard it as illegal command				
CMD15	No				
CMD16	Yes	[31:0] block length	R1	SET_BLOCKLEN	selects a block length (in bytes) for all following block commands (read and write)[1]
CMD17	Yes	[31:0] data address	R1	READ_ SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command[2]
CMD18	Yes	[31:0] data address	R1	READ_ MULTIPLE_BLOCK	Continuously transfers data blocks from card to host until interrupted by a stop command or the requested number of data blocks transmitted
CMD19	This command is not applicable in SPI mode and the card should regard it as illegal command				
CMD20	No				
CMD21 ... CMD22	reserved				
CMD23	Yes	[31:16] set to 0 [15:0] number of blocks	R1	SET_ BLOCK_COUNT	Defines the number of blocks which are going to be transferred in the immediately exceeding multiple block read or write command. If the argument is all 0s, then the subsequent read/write operation will be open-ended.

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD24	Yes	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command. [3]
CMD25	Yes	[31:0] data address	R1	WRITE_ MULTIPLE_BLOCK	Continuously writes blocks of data until a “Stop Tran’ Token or the requested number of blocks received.
CMD26	No				
CMD27	Yes	None	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD
CMD28	Yes	[31:0] data address	R1b[4]	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	Yes	[31:0] data address	R1b	CLR_WRITE_PROT	If the card has write protection features, this command clears the write protection bit of the addressed group
CMD30	Yes	[31:0] write protect data address	R1	SEND_WRITE_PROT	If the card has write protection features, this command asks the card to send the status of the write protection bits [5]
CMD31	reserved				
CMD32 ... CMD34	Reserved. These command indexes cannot be used in order to maintain backwards compatibility with older versions of the MultiMediaCards				
CMD35	Yes	[31:0] data address	R1	ERASE_GROUP_ START	Sets the address of the first erase group within a range to be selected for erase
CMD36	Yes	[31:0] data address	R1	ERASE_GROUP_ END	Sets the address of the last erase group within a continuous range to be selected for erase
CMD37	Reserved. This command index cannot be used in order to maintain backwards compatibility with older versions of the MultiMediaCards				
CMD38	Yes	[31:0] stuff bits	R1b	ERASE	Erases all previously selected erase groups
CMD39	No				
CMD40	No				
CMD41	reserved				
CMD42	Yes	[31:0] stuff bits.	R1b	LOCK_UNLOCK	Used to Set/Reset the Password or lock/unlock the card. The structure of the data block is described in Chapter 4.4.10. The size of the Data Block is defined by the SET_BLOCK_LEN command.

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD43 ... CMD54	reserved				
CMD55	Yes	[31:0] stuff bits	R1	APP_CMD	Defines to the card that the next command is an application specific command rather than a standard command
CMD56	Yes	[31:1] stuff bits [0] RD/WR_[6]	R1b	GEN_CMD	Used either to transfer a data block to the card or to get a data block from the card for general purpose / application sepcific commands. The size of the data block is defined by the SET_BLOCK_LEN command.
CMD57	Reserved				
CMD58	Yes	None	R3	READ_OCR	Reads the OCR register of a card
CMD59	Yes	[31:1] stuff bits [0:0] CRC option	R1	CRC_ON_OFF	Turns the CRC option on or off. A ‘1’ in the CRC option bit will turn the option on, a ‘0’ will turn it off
CMD60 ... CMD63	No				

1) The default block length is as specified in the CSD.
2) The data transferred must not cross a physical block boundary unless READ_BLK_MISALIGN is set in the CSD.
3) The data transferred must not cross a physical block boundary unless WRITE_BLK_MISALIGN is set in the CSD.
4) R1b : R1 is response with an optional trailing busy signal.
5) 32 write protection bits (representing 32 write protect groups starting at the specified address) followed by 16 CRC bits are transferred in a payload format via the data line. The last (least significant) bit of the protection bits corresponds to the first addressed group. If the address of the last groups are outside the valid range, then the corresponding write protection bits are set to zero.
6) RD/WR_: "1" the host receives a data block from the card.
"0" the host sends a data block to the card.

7.18 Responses

There are several types of response tokens. As in the MultiMediaCard mode, all are transmitted MSB first:

7.18.1 Format R1

This response token is sent by the card after every command, with the exception of SEND_STATUS commands. It is one byte long, and the MSB is always set to zero. The other bits are error indications, an error being signaled by a ‘1’. The structure of the R1 format is given in Figure 7-8. The meaning of the flags is defined as follows:

- **In idle state**: The card is in idle state and running the initializing process.
- **Erase Reset:** An erase sequence was cleared before executing because ‘non erase’ command (neither of CMD35, CMD36, CMD38 or CMD13) was received.
- **Illegal Command**: An illegal command code was detected or the card did not switch to the requested mode.
- **Communication CRC Error**: The CRC check of the last command failed.
- **Erase Sequence Error**: An error occurred in the sequence of erase commands (CMD35, CMD36, CMD38).
- **Address Misaligned**: A misaligned block is detected during data transfer.
- **Address Out Of Range | Block Length Error**: The command’s argument was out of the allowed range for this card.

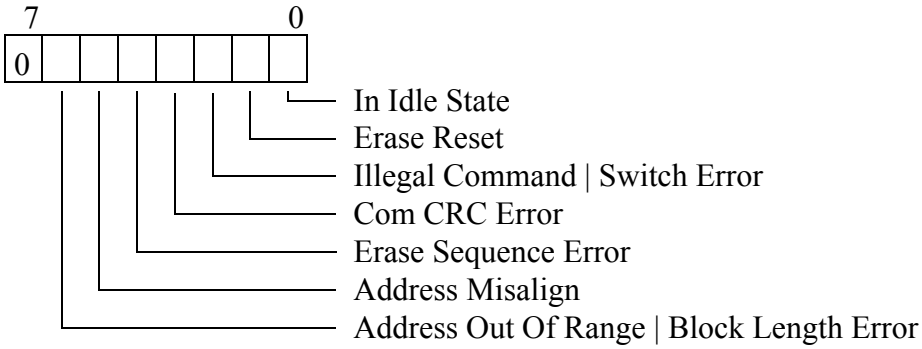


Figure 7-8 : R1 Response Format

7.18.2 Format R1b

This response token is identical to the R1 format with the addition of an immediately following busy signal.

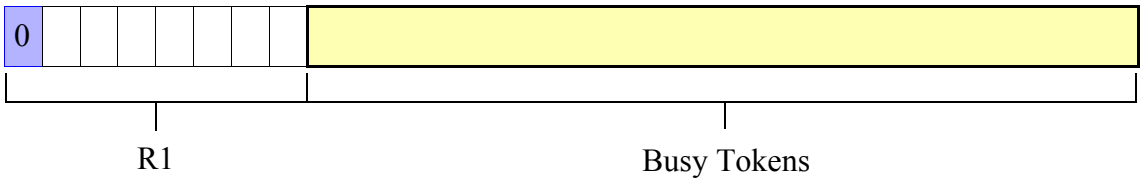


Figure 7-9 : R1b Response Format

7.18.3 Busy

The busy signal token can be any number of bytes. A zero value indicates card is busy. A non-zero value indicates the card is ready for the next command.

7.18.4 Format R2

This response token is two bytes long and sent as a response to the SEND_STATUS command. The format is given in Figure 7-10.

The first byte is identical to the response R1. The content of the second byte is described in the following:

- **CSD Overwrite:** This status bit is set if the host is trying to change the ROM section, or reverse the copy bit (set as original) or the permanent WP bit (un-protect) of the CSD register.
- **Erase Param**: An invalid selection of erase groups, for erase.
- **Write Protect Violation**: The command tried to write a write-protected block.
- **Card ECC Failed**: Card internal ECC was applied but failed to correct the data.
- **Card Error**: Generic (undifined by the standard) internal card error unrelated to the host activities.
- **Execution Error**: Generic (undefined by the standard) internal card error, occured while (and related to) execution of the last hot command
- **Write Protect Erase Skip | Lock/Unlock Command Failed**: This status bit has two functions. It is set when the host attempts to erase a write-protected block or if a sequence or password error occurred during a card lock/unlock operation.
- **Card Is Locked**: Set when the card is locked by the user. Reset when it is unlocked.

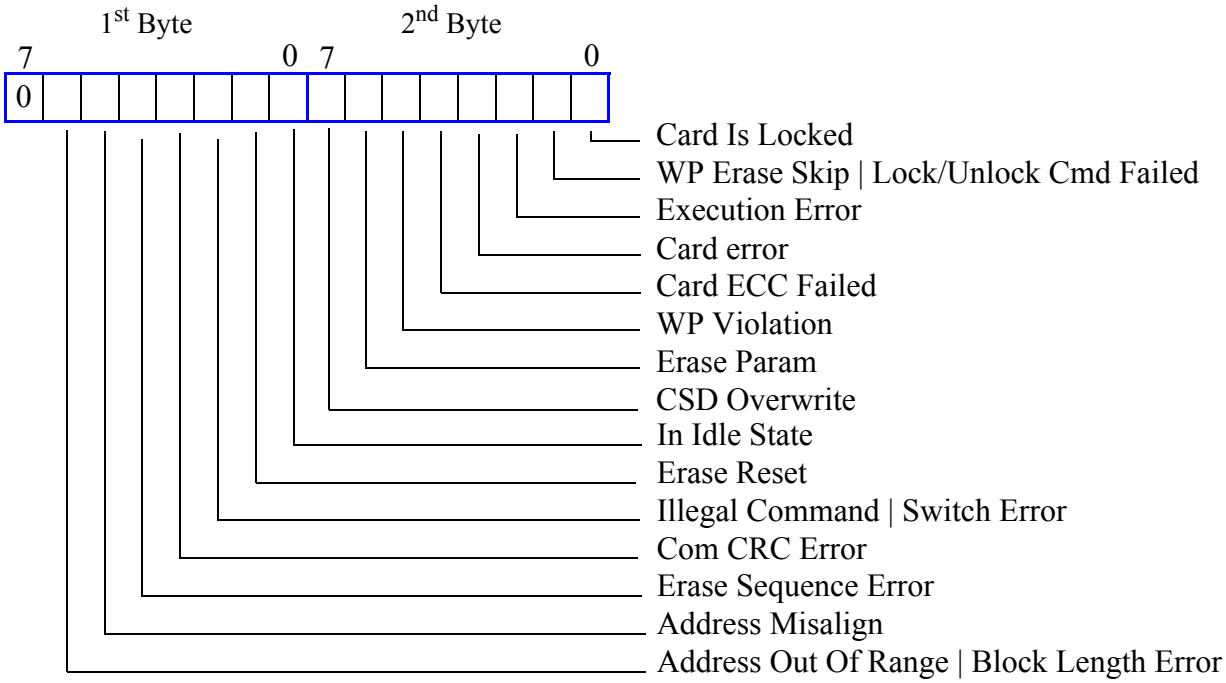


Figure 7-10 : R2 Response Format

7.18.5 Format R3

This response token is sent by the card when a READ_OCR command is received. The response length is 5 bytes (see Figure 7-11). The structure of the first (MSB) byte is identical to response type R1. The other four bytes contain the OCR register.

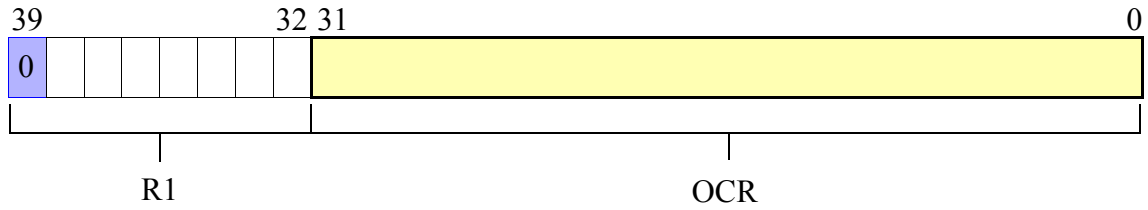


Figure 7-11 : R3 Response Format

7.18.6 Data Response

Every data block written to the card will be acknowledged by a data response token. It is one byte long and has the following format:

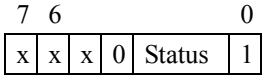


Figure 7-12 : Data Response Format

The meaning of the status bits is defined as follows:

‘010’ - Data accepted.
‘101’ - Data rejected due to a CRC error.
’110’ - Data Rejected due to a Write Error

In case of any error (CRC or Write Error) during Write Multiple Block operation, the host shall abort the operation using the “Stop Tran” Token. In case of Write Error (response ‘110’) the host should send CMD13 (SEND_STATUS) in order to get the cause of the write problem.

7.19 Data Tokens

Read and write commands have data transfers associated with them. Data is being transmitted or received via data tokens. All data bytes are transmitted MSB first.
Data tokens are 4 to (N + 3) bytes long (Where N is the data block length set using the SET_BLOCK_LENGTH Command) and have the following format:

- First byte:

Token Type	Transaction Type	7	Bit Position						0
Start Block	Single Block Read	1	1	1	1	1	1	1	0
Start Block	Multiple Block Read	1	1	1	1	1	1	1	0
Start Block	Single Block Write	1	1	1	1	1	1	1	0
Start Block	Multiple Block Write	1	1	1	1	1	1	0	0
Stop Tran	Multiple Block Write	1	1	1	1	1	1	0	1

Figure 7-13 : Start Data Block Token Format

- Bytes 2 - (N + 1): User data
- Last two bytes: 16 bit CRC.

7.20 Data Token Error

If a read operation fails and the card cannot provide the required data, it will send a data error token instead. This token is one byte long and has the following format:

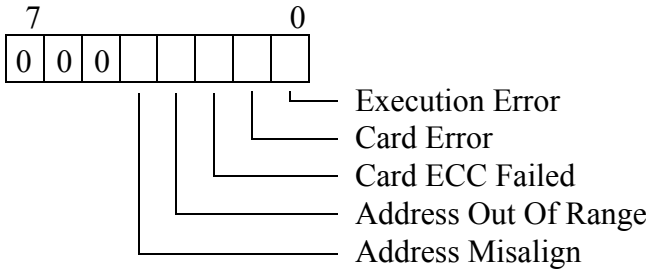


Figure 7-14 : Data Error Token

The 5 least significant bits (LSB) are the same error bits as in the response format R2.

7.21 Clearing Status Bits

As described in the previous paragraphs, in SPI mode, error and status bits are reported to the host in three different formats: response R1, response R2 and data error token (the same bits may exist in multiple response types—e.g Address Out Of Range. All Error bits defined in MultiMediaCard mode, with the exception of underrun and overrun, have the same meaning and usage in SPI mode. There are some differences in the Status bits due to the different protocol (e.g. current state is not defined in SPI mode). The detection mode and clear condition of Error and Status bits are identical to the MultiMediaCard mode, with one exception, Error bits are cleared when read by the host, regardless of the response format.

The following table describes the the various status bits:.

Idnetifier	Included in rep	Type	Det Mode	Value	Description	Clear Cond
Address Out Of Range	R1 R2 DataErr	E	R	'0'= no error '1'= error	The command's address argument was out of the allowed range for this card.	C
			X		A multiple block or stream read/write operation is attempting to read or write beyond the card capacity (Although it started in a valid address)	
Address Misalign	R1 R2 DataErr	E	R	'0'= no error '1'= error	The command' s address argument (in accordance with the currently set block length) positions the first data block misaligned to the card physical blocks.	C
			X		A multiple block read/write operation is attempting to read or write a data block, which is not aligned to the physical blocks of the card (Although it started with a valid address/block-length combination)	
Erase Sequence Error	R1 R2	E	R	'0'= no error '1'= error	An error in the sequence of erase commands occurred.	C
Erase Param	R2	E	X	'0'= no error '1'= error	An invalid selection of erase groups, for erase, occurred.	C

Idnetifier	Included in rep	Type	Det Mode	Value	Description	Clear Cond
Block Length Error	R1 R2	E	R	'0'= no error '1'= error	Either the argument of a SET_BLOCKLEN command exceeds the maximum allowed value for the card, or the previously defined block length is illegal for the current command (e.g. the host is issues a write command and the current block length is smaller than the card maximum value and write partial blocks is not allowed)	C
WP violation	R2	E	X	'0'= not protected '1'= protected	Attempt to program a write protected block.	C
Com CRC Error	R1 R2	E	R	'0'= no error '1'= error	The CRC check of the received command failed.	C
Illegal Command	R1 R2	E	R	'0'= no error '1'= error	The received command is not legal for the card state.	C
Switch Error	R1 R2	E	X	'0'= no error '1'= error	If set, the card did not switch to the expected mode as requested by the SWITCH command	C
Card ECC failed	R2 DataErr	E	X	'0'= success '1'= failure	Card internal ECC was applied but failed to correct the data.	C
Card Error	R2 DataErr	E	R	'0'= no error '1'= error	(Undefined by the standard) A card error occurred, which is not related to the host command.	C
Execution Error	R2 DataErr	E	X	'0'= no error '1'= error	(Undefined by the standard) A generic card error related to the (and detected during) execution of the last host command (e.g. read or write failures).	C
WP Erase Skip	R2	S	X	'0'= not protected '1'= protected	Only partial address space was erased due to existing write protected blocks.	C
Lock/Unlock Cmd Failed	R2	E	X	'0'= no error '1'= error	Sequence or password error during card lock/unlock operation.	C
Card Is Locked	R2	S		'0' = card is not locked '1' = card is locked	Card is locked by a user password	A
Erase Reset	R1 R2	E	R	'0'= cleared '1'= set	An erase sequence was cleared before executing because an out of erase sequence command was received.(other than CMD35, CMD36, CMD38 or CMD13)	C
In Idle State	R1 R2	S		0 = Card is ready 1 = Card is in idle state	The card enters the idle state after power up or reset command. It will exit this state and become ready upon completion of its initialization procedures.	A
CSD Overwrite	R2	E	X	'0'= no error '1'= error	The host is trying to change the ROM section, or is trying to reverse the copy bit (set as original) or permanent WP bit (un-protect) of the CSD register.	C

7.22 Card Registers

In SPI mode, only the OCR, CSD and CID registers are accessible. Their format is identical to the format in the MultiMediaCard mode. However, a few fields are irrelevant in SPI mode.

7.23 SPI Bus Timing Diagrams

All timing diagrams use the following schematics and abbreviations:

Symbol	Definition
H	Signal is high (logical '1'
L	Signal is low (logical '0')
X	Don't care (Undefined Value)
Z	High impedance state (-> = 1)
*	Repeater
Busy	Busy Token
Command	Command token
Response	Response token
Data block	Data token

All timing values are defined in Table7-9. The host must keep the clock running for at least N_{CR} clock cycles after receiving the card response. This restriction applies to both command and data response tokens.

7.23.1 Command / Response

- **Host Command to Card Response - Card is ready**

The following timing diagram describes the basic command response (no data) SPI transaction.

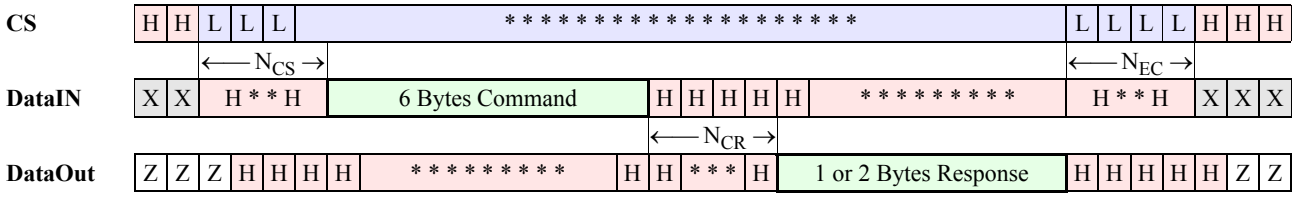


Figure 7−15 : SPI Command/Response Transaction, Card Is Ready

- **Host Command to Card Response - card is busy**

The following timing diagram describes the command response transaction for commands when the card response is of type R1b (e.g. SET_WRITE_PROT and ERASE). When the card is signaling busy, the host may deselect it (by raising the CS) at any time. The card will release the DataOut line one clock after the CS going high. To check if the card is still busy, it needs to be reselected by asserting (set to low) the CS signal. The card will resume busy signal (pulling DataOut low) one clock after the falling edge of CS.

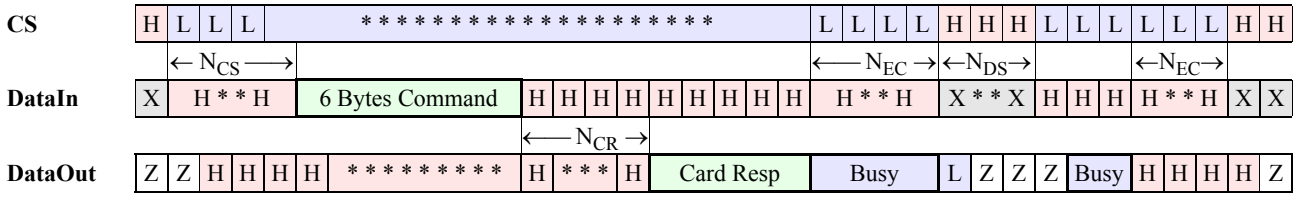


Figure 7-16 : SPI Command/Response Transaction, Card Is Busy

- **Card Response to Host Command**

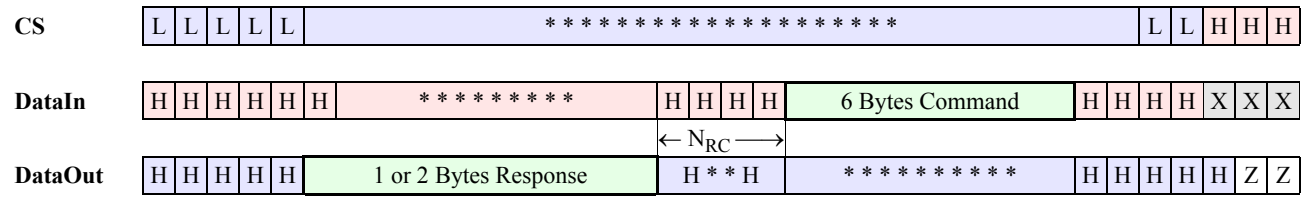


Figure 7-17 : SPI Card Response To The Next Host Command

7.23.2 Data read

- **Single Block Read**

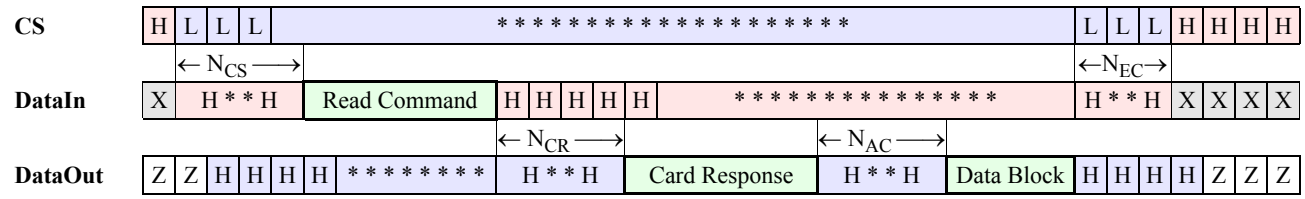


Figure 7-18 : SPI Single Block Read

- **Multiple Block Read - Stop Transmission is sent between blocks**

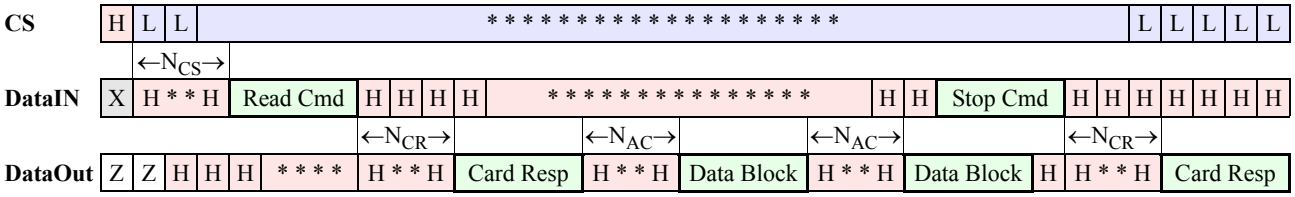


Figure 7-19 : SPI Multiple Block Read, Stop Transmission Does Not Overlap Data

The timing for de-asserting the CS signal after the last card response is identical to a standard command/response transaction as described in Figure 7-15;

- **Multiple Block Read - Stop Transmission is sent within a block**

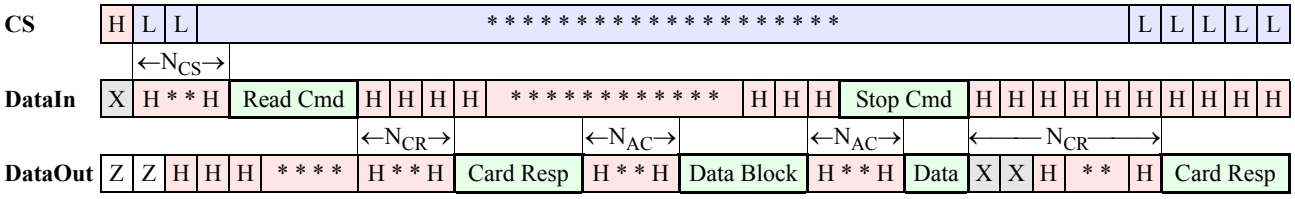


Figure 7-20 : SPI Multiple Block Read, Stop Transmission Overlaps Data

In an Open-ended (or host aborted) multiple block read transaction the stop transmission command may be sent asynchronously to the data transmitted out of the card and may overlap the data block. In this case the card will stop sending the data and transmit the response token as well. The delay between command and response is standard N_{CR} Clocks. The first byte, however, is not guaranteed to be all set to '1'. The card is allowed up to two clocks to stop data transmission.
The timing for de-asserting the CS signal after the last card response is identical to a standard command/ response transaction as described in Figure 7-15;

- **Reading the CSD and CID registers**

The following timing diagram describes the SEND_CSD and SEND_CID command bus transaction. The time-out values between the response and the data block is N_{CX} , and not N_{ac} , which is used for data read (since $N_{ac}$is still unknown at the time the CSD register is read). The SEND_CID transaction complies with the same timing diagram for consistency of the read register commands

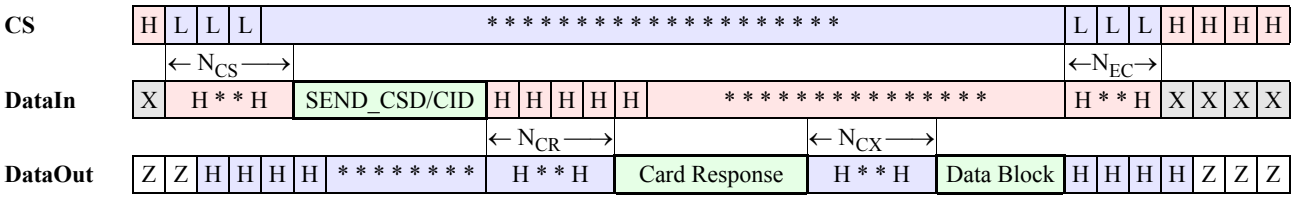


Figure 7-21 : SPI Read CSD and CID Registers

7.23.3 Data write

- **Single Block Write**

The host may deselect a card (by raising the CS) at any time during the card busy period (refer to the given timing diagram). The card will release the DataOut line one clock after the CS going high. To check if the card is still busy it needs to be reselected by asserting (set to low) the CS signal. The card will resume busy signal (pulling DataOut low) one clock cycle after the falling edge of CS.

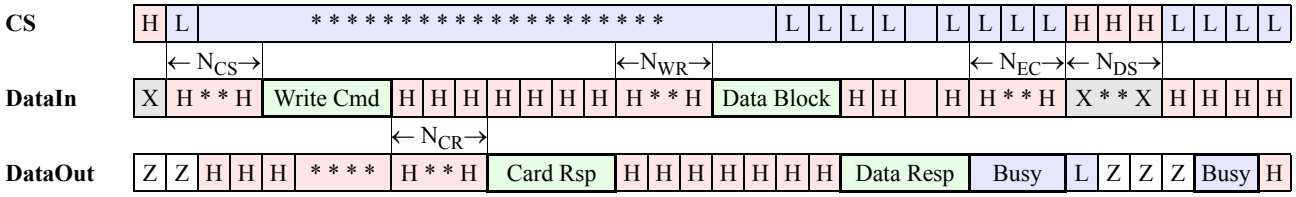


Figure 7-22 : SPI Single Block Write

- **Multiple Block Write**

The timing behavior of the multiple block write transaction starting from the command up to the first data block is identical to the single block write. Figure 7-23 describes the timing between the data blocks of a multiple block write transaction. Timing of the 'Stop Tran' token is identical to a standard data block. After the "Stop Tran" token is received ny the card, the data on the DataOut line is undefined for one byte (N_{BR}), after which a Busy token may appear. The host may deselect and reselect the card during every busy period between the data blocks. Timing for toggling the CS signal is identical to the Single block write transaction.

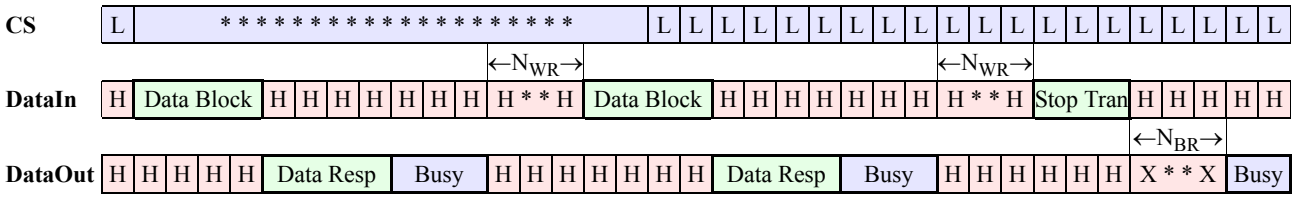


Figure 7-23 : SPI Multiple Block Write

7.24 Timing Values

Symbol	Min	Max	Unit
N_{CS}	0	-	8 clock cycles
N_{CR}	1	8	8 clock cycles
N_{CX}	0	8	8 clock cycles
N_{RC}	1	-	8 clock cycles
N_{AC}	1	(10/8) * (TAAC * F_{OP} + 100 * NSAC)[a]	8 clock cycles
N_{WR}	1	-	8 clock cycles
N_{EC}	0	-	8 clock cycles
N_{DS}	0	-	8 clock cycles
N_{BR}	1	1	8 clock cycles

a. F_{OP} is the MMC clock frequency the host is using for the read operation.

7.25 SPI Electrical Interface

Identical to MultiMediaCard mode, with the exception of the programmable card output drivers option, which is not supported in SPI mode.

7.26 SPI Bus Operation Conditions

Identical to MultiMediaCard mode.

7.27 SPI Bus Timing

Identical to MultiMediaCard mode. The timing of the CS signal is the same as any other card input.