SD Memory Card Specifications

Part 1
PHYSICAL LAYER SPECIFICATION
Version 1.0

March 2000

SD Group

Matsushita Electric Industrial Co., Ltd. (MEI)

SanDisk Corporation

Toshiba Corporation

Revision History

Date	Version	Changes compared to previous issue
March 22th, 2000	1.0	Base version

Conditions for publication:

- Publisher and Copyright Holder**:** SD Group (MEI, SanDisk, Toshiba)
- Confidentiality**:**

This document shall be treated as confidential under the Non Disclosure Agreement which has been signed by the obtainer. Reproduction in whole or in part is prohibited without prior written permission of SD Group.

- Exemption**:**

Non will be liable for any damages from use of this document.

1 General description

SD Memory Card (Secure Digital Memory Card) is a Flash-Based memory card that is specifically designed to meet the security, capacity, performance and environment requirements inherent in newly emerging audio and video consumer electronic devices. The SD Memory Card will include a copyright protection mechanism that complies with the security of the SDMI standard and will be faster and capable for higher Memory capacity. The SD Memory Card security system uses mutual authentication and a "new cipher algorithm" to protect from illegal usage of the card content. A none secured access to the user‘s own content is also available. The physical form factor, pin assignment and data transfer protocol are forward compatible with the MultiMediaCard with some additions.

The SD Memory Card communication is based on an advanced 9-pin interface (Clock, Command, 4xData and 3xPower lines) designed to operate in a low voltage range. The communication protocol is defined as a part of this specification. The SD Memory Card host interface supports regular MultiMediaCard operation as well. In other words, MultiMediaCard forward compatibility was kept. Actually the main difference between SD Memory Card and MultiMediaCard is the initialization process.

The SD Memory Card Specifications were divided to several documents. The SD Memory Card documentation structure is given in Figure 1.

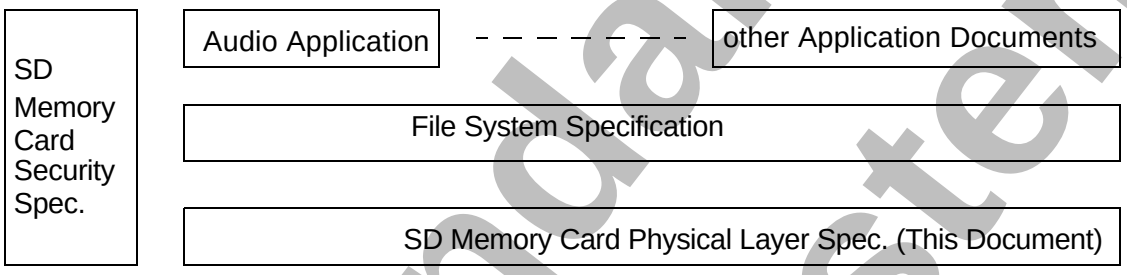


Figure 1: SD Memory Card Documentation Structure

- **SD Memory Card Audio Application Specification:**

This specification along with other application specifications describe the specification of certain application (in this case - Audio Application) and the requirements to implement it.

- **SD Memory Card File System Specification:**

Describes the specification of the file format structure of the data saved in the SD Memory Card (in protected and un-protected areas).

- **SD Memory Card Security Specification:**

Describes the copyright protection mechanism and the application specific commands that support it.

- **SD Memory Card Physical Layer Specification (this document):**

Describes the physical interface and the command protocol used by the SD Memory Card.

The purpose of the SD Memory Card Physical Layer specification is the definition of the SD Memory Card, its environment and handling.

The document is split up into several portions. Chapter 3 gives a general overview of the system

concepts. The common SD Memory Card characteristics are described in Chapter 4. As this description defines an overall set of card properties, we recommend to use the product documentation in parallel. The card registers are described in Chapter 5.

Chapter 6 defines the electrical parameters of the SD Memory Card's hardware interface.

Chapter 8 describes the physical and mechanical properties of the SD Memory Cards and the minimal recommendations to the card slots or cartridges.

As used in this document, "shall" or "will" denotes a mandatory provision of the standard. "Should" denotes a provision that is recommended but not mandatory. "May" denotes a feature whose presence does not preclude compliance, that may or may not be present at the option of the implementor.

2 System features

- **Targeted for portable and stationary applications**
- **Voltage range:**

  SD Memory Card -
  Basic communication (CMD0, CMD15, CMD55, ACMD41): 2.0 - 3.6V
  Other commands and memory access: 2.7 - 3.6V

  SDLV Memory Card (low voltage) - Operatiing voltage range: 1.6 - 3.6V
- **Designed for read-only and read/write cards.**
- **Variable clock rate 0 - 25 MHZ**
- **Up to 10MByte/sec Read/Write rate (using 4 parallel data lines).**
- **Maximum data rate with up to 10 cards**
- **Correction of memory field errors**
- **Card removal during read operation will never harm the content**
- **Forward compatibility to MultiMediaCard**
- **Copyrights Protection Mechanism - Complies with highest security of SDMI standard.**
- **Password Protection of cards (option)**
- **Write Protect feature using mechanical switch**
- **Built-in write protection features (permanent and temporary)**
- **Card Detection (Insertion/Removal)**
- **Application specific commands**
- **Comfortable erase mechanism**
- **Protocol attributes of the communication channel:**

SD Memory Card Communication Channel
Six-wire communication channel (clock, command, 4 data lines)
Error-protected data transfer
Single or Multiple block oriented data transfer

- **Thin (1.4mm) SD Memory Card (Preliminary)**

3 SD Memory Card System Concept

The SD Memory Card provides application designers with a low cost mass storage device, implemented as a removable card, that supports high security level for copyright protection and a compact, easy-to-implement interface.

SD Memory Cards can be grouped into several card classes which differ in the functions they provide (given by the subset of SD Memory Card system commands):

- Read/Write (RW) cards (Flash, One Time Programmable - OTP, Multiple Time Programmable - MTP). These cards are typically sold as blank (empty) media and are used for mass data storage, end user recording of video, audio or digital images.
- Read Only Memory (ROM) cards. These cards are manufactured with a fixed data content. They are typically used as a distribution media for software, audio, video etc.

In terms of operating supply voltage, two types of SD Memory Cards are defined:

- SD Memory Cards which supports initialization/identification process with a range of 2.0-3.6v and operating voltage within this range as defined in the CSD register.
- SDLV Memory Cards - Low Voltage SD Memory Cards, that can be operate in voltage range of 1.6-3.6V. The SDLV Memory Cards will be labeled differently then SD Memory Cards.

SD Memory Card system includes the SD Memory Card (or several cards) the bus and their Host / Application. The Host and Application specification is beyond the scope of this document. The following sections provides an overview of the card, bus topology and communication protocols of the SD Memory Card system. The copyright protection (security) system description is given in “SD Memory Card Security Specification” document.

3.1 Bus Topology

The SD Memory Card system defines two alternative communication protocols: SD and SPI. Applications can choose either one of modes. Mode selection is transparent to the host. The card automatically detects the mode of the reset command and will expect all further communication to be in the same communication mode. Therefore, applications which uses only one communication mode do not have to be aware of the other.

3.1.1 SD bus

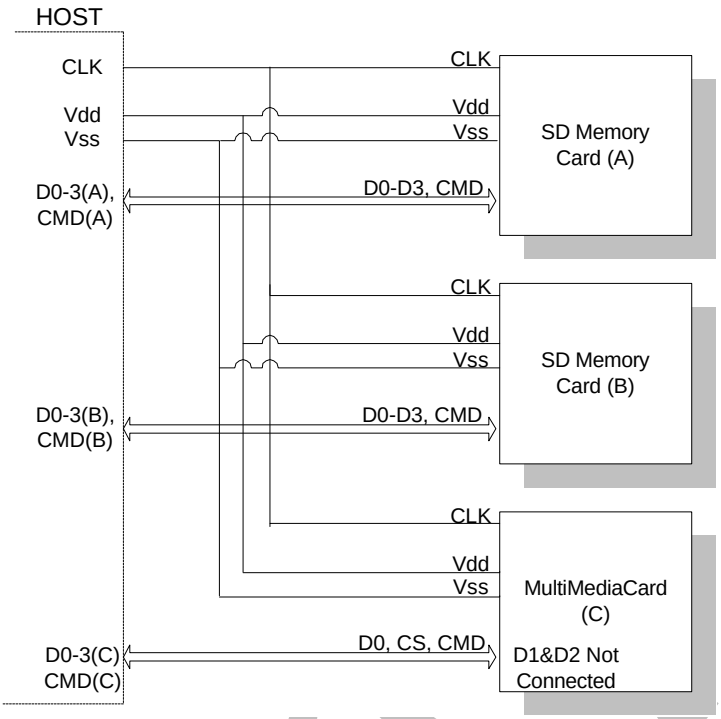


Figure 2: SD Memory Card system bus Topology

The SD bus includes the following signals:

CLK:	Host to card clock signal
CMD:	Bidirectional Command/Response signal
DAT0 - DAT3:	4 Bidirectional data signals.
VDD, VSS1, VSS2:	Power and ground signals.

The SD Memory Card bus has a single master (application), multiple slaves (cards), synchronous star topology (refer to Figure 2). Clock, power and ground signals are common to all cards. Command (CMD) and data (DAT0 - DAT3) signals are dedicated to each card providing continues point to point connection to all the cards.

During initialization process commands are sent to each card individually, allowing the application to detect the cards and assign logical addresses to the physical slots. Data is always sent (received) to (from) each card individually. However, in order to simply the handling of the card stack, after the initialization process, all commands may be sent concurrently to all cards. Addressing information is provided in the command packet.

SD bus allows dynamic configuration of the number of data lines. After power up, by default, the SD Memory Card will use only DAT0 for data transfer. After initialization the host can change the bus width (number of active data lines). This feature allows easy trade off between HW cost and system performance.

3.1.2 SPI bus

The SPI compatible communication mode of the SD Memory Card is designed to communicate with a SPI channel, commonly found in various microcontrollers in the market. The interface is selected during the first reset command after power up and cannot be changed as long as the part is powered on.

The SPI standard defines the physical link only, and not the complete data transfer protocol. The SD Memory Card SPI implementation uses the same command set of the SD mode. 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, relatively to the SD mode which enables the wide bus option.

The SD Memory Card SPI interface is compatible with SPI hosts available on the market. As any other SPI device the SD Memory Card 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 are 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.

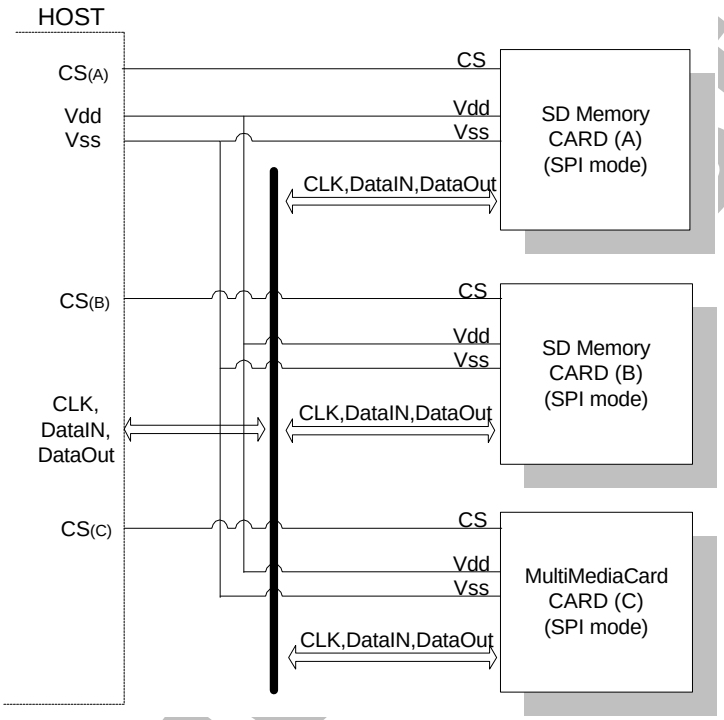


Figure 3: SD Memory Card system (SPI mode) bus topology

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 3).

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 SPI interface uses the 7 out of the SD 9 signals (DAT1 and DAT 2 are not used, DAT3 is the CS signal) of the SD bus.

3.2 Bus Protocol

3.2.1 SD bus

Communication over the SD bus is based on command and data bit streams which are initiated by a start bit and terminated by a stop bit.

- **Command**: a command is a token which starts an operation. A command is sent from the host either to a single card (addressed command) or to all connected cards (broadcast command). A command is transferred serially on the CMD line.
- **Response**: a response is a token which is sent from an addressed card, or (synchronously) from all connected cards, to the host as an answer to a previously received command. A response is transferred serially on the CMD line.
- **Data**: data can be transferred from the card to the host or vice versa. Data is transferred via the data lines.

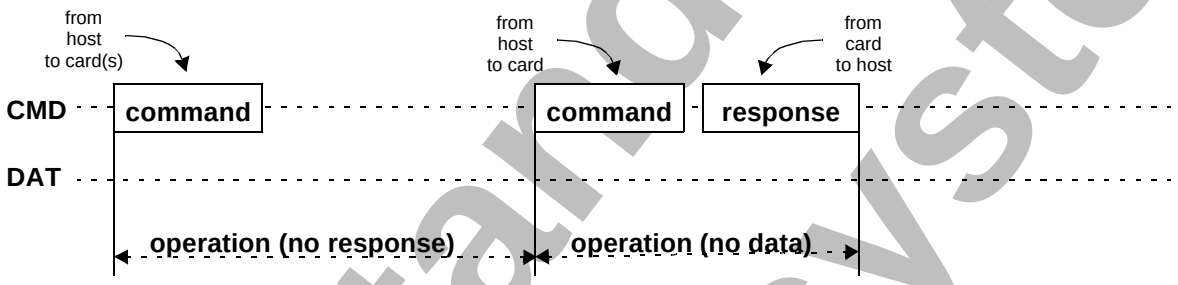


Figure 4: “no response” and “no data” operations

Card addressing is implemented using a session address, assigned to the card during the initialization phase. The structure of commands, responses and data blocks is described in Chapter 4. The basic transaction on the SD bus is the command/response transaction (refer to Figure 4). This type of bus transactions transfer their information directly within the command or response structure. In addition, some operations have a data token.

Data transfers to/from the SD Memory Card are done in blocks. Data blocks always succeeded by CRC bits. Single and multiple block operations are defined. Note that the Multiple Block operation mode is better for faster write operation. A multiple block transmission is terminated when a stop command follows on the CMD line. Data transfer can be configured by the host to use single or multiple data lines (as long as the card supports this feature).

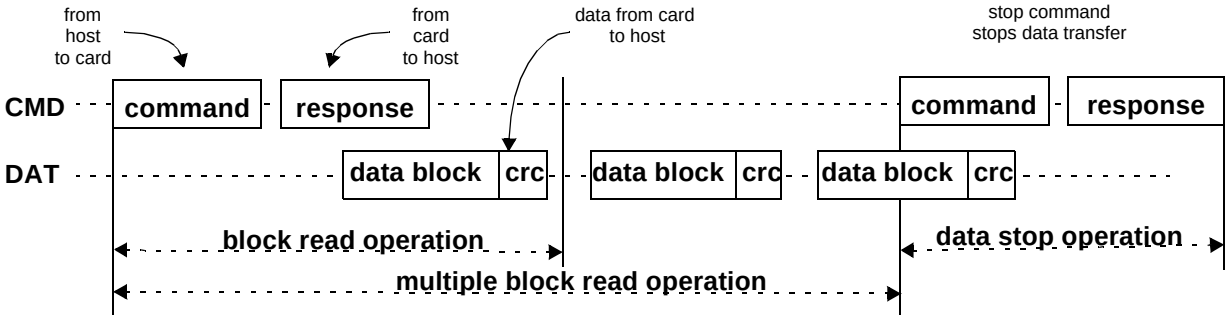


Figure 5: (Multiple) Block read operation

The block write operation uses a simple busy signaling of the write operation duration on the DAT0 data line (see Figure 6) regardless of the number of data lines used for transferring the data.

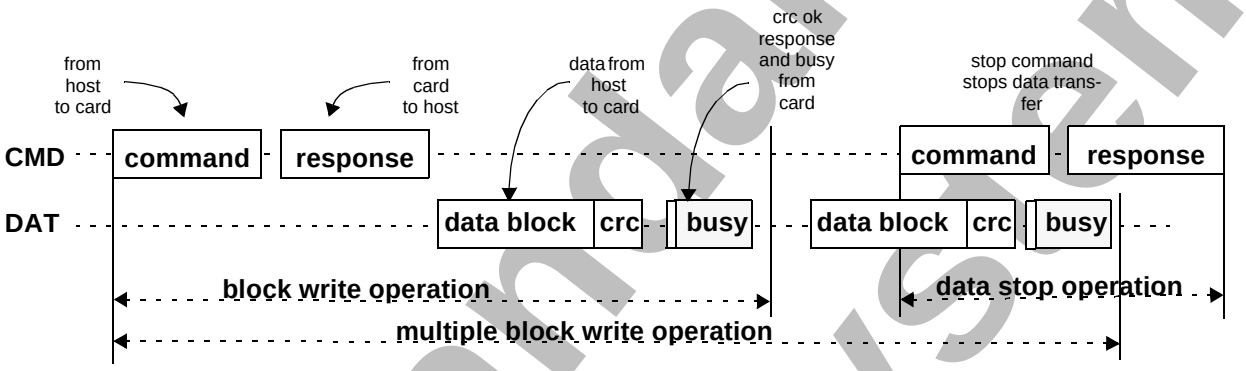


Figure 6: (Multiple) Block write operation

Command tokens have the following coding scheme:

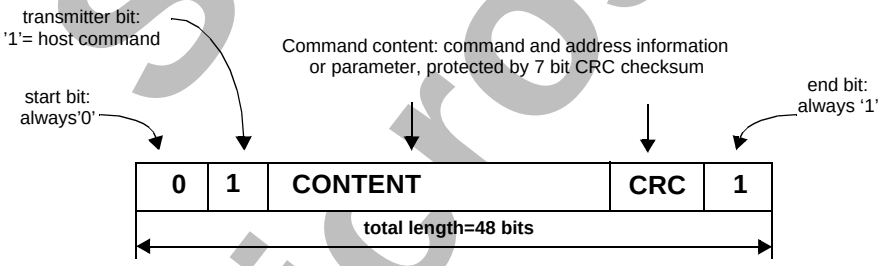


Figure 7: Command token format

Each command token is preceded by a start bit ('0') and succeeded by an end bit ('1'). The total length is 48 bits. Each token is protected by CRC bits so that transmission errors can be detected and the operation may be repeated.

Response tokens have four coding schemes depending on their content. The token length is either 48 or 136 bits. The detailed commands and response definition is given in Chapter 4.7. The CRC protection algorithm for block data is a 16 bit CCITT polynomial. All used CRC types are described in Chapter 4.5.

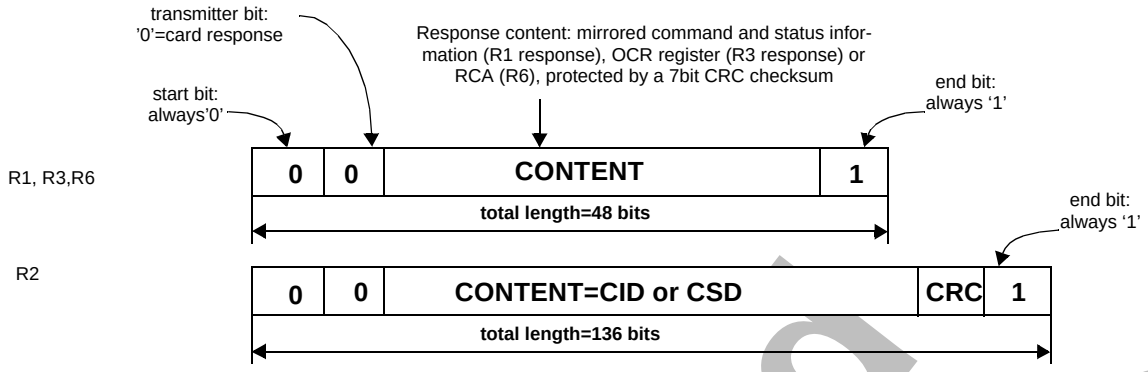


Figure 8: Response token format

In the CMD line the MSB bit is transmitted first the LSB bit is the last.

when the wide bus option is used, the data is transferred 4 bits at a time (refer to Figure 9). Start and end bits, as well as the CRC bits, are transmitted for every one of the DAT lines. CRC bits are calculated and checked for every DAT line individually. The CRC status response and Busy indication will be sent by the card to the host on DAT0 only (DAT1-DAT3 during that period are don't care).

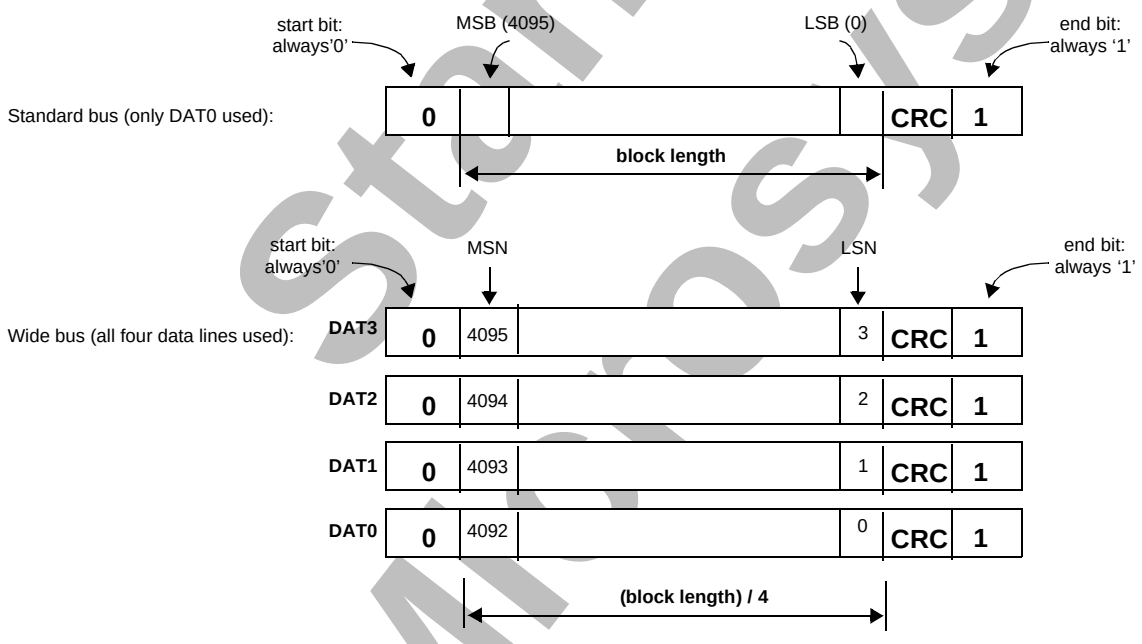


Figure 9: Data packet format

3.2.2 SPI Bus

While the SD 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 SD protocol, the SPI messages consist of command, response and data-block tokens All communication between host and cards 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 SD mode in the following three aspects:

- The selected card always responds to the command.
- Two new (8 & 16 bit) response structure is 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 SD mode.

In addition to the command response, every data block sent to the card during write operations will be responded with a special data response token.

- **Data Read**

Single and multiple block read commands are supported in SPI mode. However, in order to comply with the SPI industry standard, only two (unidirectional) signal are used (refer to Chapter 10). Upon reception of a valid read command the card will respond with a response token followed by a data token of the length defined in a previous SET_BLOCKLEN (CMD16) command. A multiple block read operation is terminated, similar to the SD protocol, with the STOP_TRANSMISSION command.

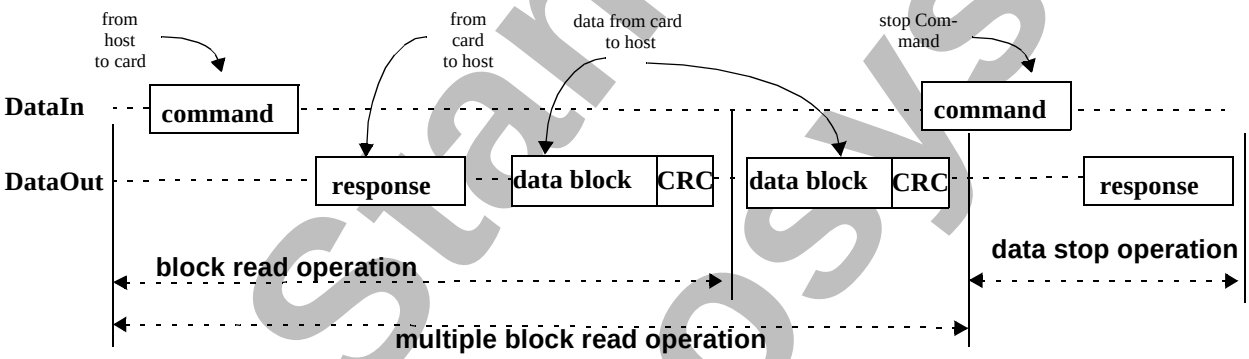


Figure 10: Read operation

A valid data block is suffixed with a 16 bit CRC generated by the standard CCITT polynomial $x^{16}+x^{12}+x^5+1$.

In case of a data retrieval error, the card will not transmit any data. Instead, a special data error token will be sent to the host. Figure 11 shows a data read operation which terminated with an error

token rather than a data block.

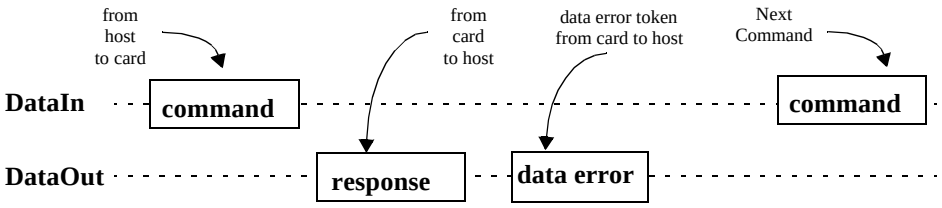


Figure 11: Read operation - data error

- **Data Write**

Single and multiple block write operations are supported in SPI mode. Upon reception of a valid write command, 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 identical to the read operation (see Figure 12).

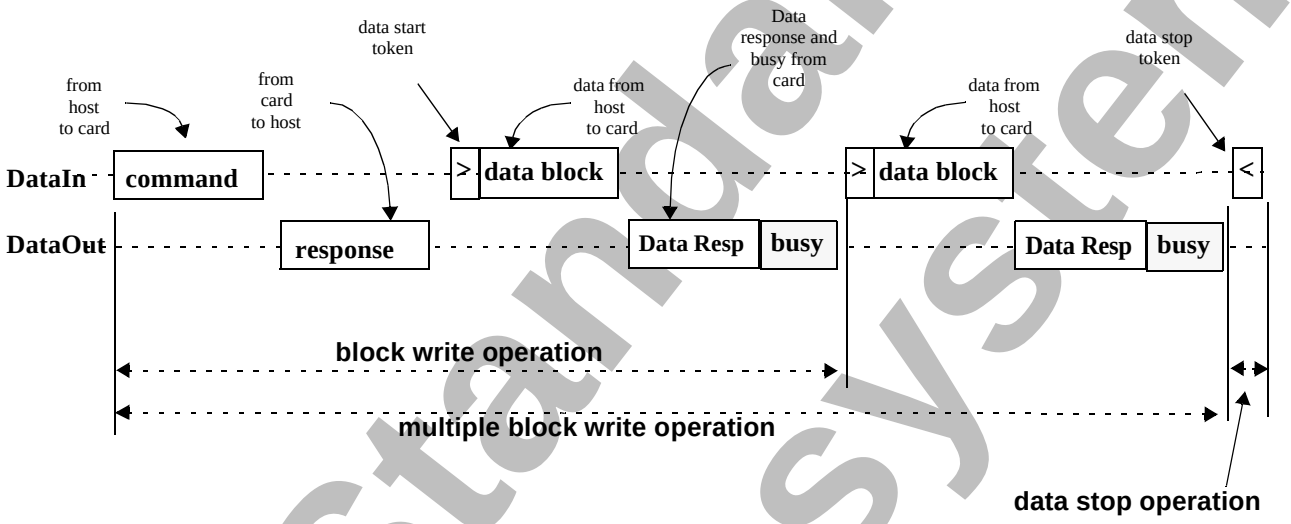


Figure 12: Write operation

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).

3.3 SD Memory Card - Pins and Registers

The SD Memory Card has the form factor 24mm x 32mm x 2.1mm.

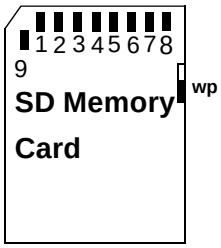


Figure 13: SD Memory Card shape and interface (top view)

Figure 13 describes the general idea of the shape and interface contacts of SD Memory Card. The detailed physical dimensions and mechanical description is given in chapter 9.

The following table defines the card contacts:

Pin #	SD Mode			SPI Mode		
	Name	Type[1]	Description	Name	Type	Description
1	CD/DAT3[2]	I/O/PP[3]	Card Detect / Data Line [Bit 3]	CS	I	Chip Select (neg true)
2	CMD	PP	Command/Response	DI	I	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 Line [Bit 0]	DO	O/PP	Data Out
8	DAT1	I/O/PP	Data Line [Bit 1]	RSV		
9	DAT2	I/O/PP	Data Line [Bit 2]	RSV		

Table 1: SD memory Card Pad Assignment

1) S: power supply; I: input; O: output using push-pull drivers; PP: I/O using push-pull drivers;
2) The extended DAT lines (DAT1-DAT3) are input on power up. They start to operate as DAT lines after SET_BUS_WIDTH command.
3) After power up this line is input with 50KOhm pull-up (can be used for card detection or SPI mode selection). The pull-up should be disconnected by the user, during regular data transfer, with SET_CLR_CARD_DETECT (ACMD42) command

Each card has a set of information registers (see also Chapter 5 in the SD Memory Card Physical

Layer Specification):

Name	Width	Description
CID	128	Card identification number; card individual number for identification. **Mandatory**.
RCA[1]	16	Relative card address; local system address of a card, dynamically suggested by the card and approved by the host during initialization. **Mandatory**.
DSR	16	Driver Stage Register; to configure the card's output drivers. **Optional.**
CSD	128	Card Specific Data; information about the card operation conditions. **Mandatory**
SCR	64	SD Configuration Register; information about the SD Memory Card's Special Features capabilities. **Mandatory**
OCR	32	Operation condition register. **Mandatory.**

Table 2: SD Memory Card registers

1) RCA register is not used (available) in SPI mode.

The host may reset the cards by switching the power supply off and on again. Each card shall have its own power-on detection circuitry which puts the card into a defined state after the power-on. No explicit reset signal is necessary. The cards can also be reset by sending the GO_IDLE (CMD0) command.

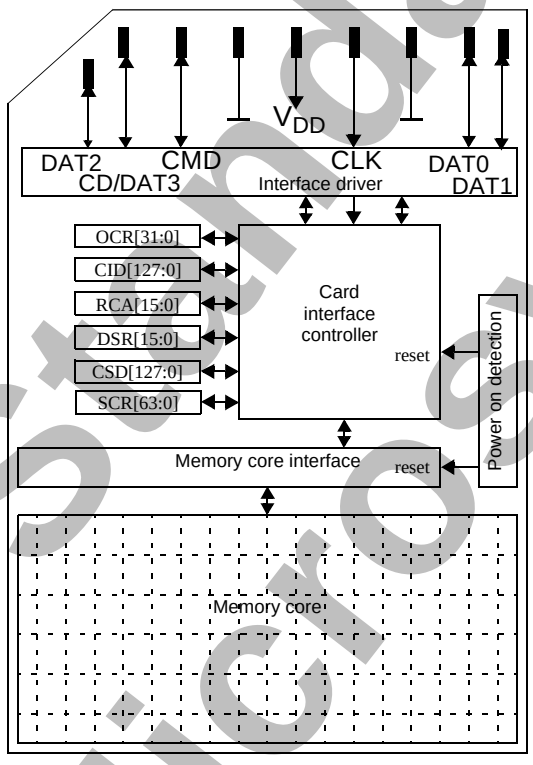


Figure 14: SD Memory Card architecture

3.4 Compatibility to MultiMediaCard

The SD Memory Card protocol is designed to be a super-set of the MultiMediaCard protocol[1]. The main additions are the wide bus option and the content protection support (refer to Table 3 for details). It is very easy to design host systems, capable of supporting both types of cards. The intent is to enable application designers to make use of the exiting install base of MultiMediaCard, unless the application cannot do without either the fast data transfer rate (wide bus), or content security.

	SD Memory Card	MultiMediaCard	Comments
Bus width	1bit or 4 bits	1 bit only	
System bus organization (multiple cards connection)	Star Topology	Bus Topology	
Initialization commands	CMD0 ACMD41 CMD2 CMD3	CMD0 CMD1 CMD2 CMD3	In MultiMediaCard CMD1 and CMD2 are sent concurrently from all cards using the OD drivers. In SD memory card each card is reset and identified independently and the RCA (CMD3) is assigned by the card.
Operation Commands	SEND_NUM_WR_BLOCK SET_WR_BLK_COUNT		two new command for improved write performance (ACMD23, ACMD22) are supported in SD.
Maximum Clock rate	25MHz	20MHz	
Copyright protection	supported	not supported	
Write protect switch	supported	not supported	When MultiMediaCard is inserted into an SD Memory Card slot, it always perceived as non-write-protected card (window closed)
Feature of Pin #1	Has a card internal pull-up resistor	Defined as "not connected"	In SD Memory Card pin #1 may be used for card detection.
CSD Strucrure	Different from MMC (mainly Sector Size/ Groups is different)		
CID Structure	Different from MMC		In SD the Manufacturing date field is bigger. Product ID field is smaller.
SPI R/W Multiple Block	Supported	Not supported	
Stream R/W mode	not supported	supported (optional)	SD Memory Card supports only single and multiple block read/ write operations.
I/O Mode	not supported	supported (optional)	I/O (Interrupt) mode is not supported in SD Memory Card.

Table 3: Differences between SD Memory Card and MultiMediaCard Feature set

1. As defined in the MultiMediaCard system specification V2.11. Published by the MMCA technical commitee.

The only difference (as opposed to addition) between the SD Memory Card and the MultiMediaCard is the bus topology and initialization protocol. While the MultiMediaCard stack is connected on the same bus and being identified using synchronous transmission of Open-drain outputs, each SD Memory Card has an independent point-to-point connection to the host, and the cards are identified serially, one at a time (refer to Table 4 for a command set comparison between SD Memory Card and MultiMediaCard. Detailed description of the SD Memory card protocol commands can be found in Chapter 4).

The initializing procedure in the SD protocol is defined to successfully identify either a MultiMediaCard or a SD Memory Card, which ever is currently connected on the bus. After card detection, the host executes the initializing procedure and ends up with an identified card of a known type.

Once the card is initialized the application can determine the card capabilities by querying the various configuration registers, and decide whether or not to use it.

The physical dimension of the SD Memory Card is thicker than MultiMediaCard (2.1mm vs. 1.4mm; refer to Chapter 9) but it is defined in such a way that a MultiMediaCard can be inserted into SD Memory Card socket.

Three different card detect mechanisms are defined for the SD Memory Card (e.g. mechanical insertion which can be sensed using the WP switch, Electrical insertion which can be sensed using the pull-up resistor on DAT3 and periodical attempts to initialize the card). Since some of these methods may be not relevant (or behave differently) for the MultiMediaCard, it is recommended not to depend on the preemptive card detects methods only. The host should implement a polling mechanism or allow the operator to request card identification.

Class	CMD	SD Memory Card	MultiMedia Card	Comment
Class 0	CMD0	CMD0 (Mandatory)	CMD0	Same command.
	CMD1	Reserved	CMD1	In SD Memory Card ACMD41 is used instead of CMD1
	CMD2	CMD2 (Mandatory)	CMD2	Similar command except buffer type used to transmit to response of the card. (SD Memory Card: push-pull, MultiMediaCard: open-drain)
	CMD3	CMD3 (Mandatory)	CMD3	In both protocols this command is used to assign a logical address to the card. While In MultiMediaCard the host assigned the address, in SD memory Card it is the responsibility of the card.
	CMD4-10	CMD4-10 (Mandatory)	CMD4-10	Same commands.
Class 1	CMD11	Reserved	CMD11	SD Memory Card doesn't support stream access.
Class 0	CMD12-15	CMD12-15 (Mandatory)	CMD12-15	Same commands.

Table 4: Commands comparison table.

Class	CMD	SD Memory Card	MultiMedia Card	Comment
Class 2	CMD16-19	CMD16-19 (Mandatory)	CMD16-19	Same commands.
Class 3	CMD20	Reserved	CMD20	SD Memory Card dose not support stream access.
	CMD21-23	Reserved	Reserved	All reserved.
Class 4	CMD24-27	CMD24-27 (Mandatory)	CMD24-27	Same commands.
Class 6	CMD28-31	CMD28-31 (Optional)	CMD28-31	Same commands.
Class 5	CMD32-33	CMD32-33 (Mandatory)	CMD32-33	Same commands.
	CMD34-37	Reserved	CMD34-37	SD Memory Card dose not support TAG and Erase Group commands
	CMD38	CMD38 (Mandatory)	CMD38	Same Command.
Class 9	CMD39-41	Reserved	CMD39-41	SD Memory Card dose not support I/O mode.
Class 7	CMD42-54	CMD42-54 (Optional)	CMD42-54	Same commands.
Class 8	CMD55-56	CMD55-56 (Mandatory)	CMD55-56	Same commands.
	CMD60-63	CMD60-63 (reserved for manu-facturer)	CMD60-63 (reserved for manufac-turer)	

Table 4: Commands comparison table.

4 SD Memory Card Functional Description

4.1 General

All communication between host and cards 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. 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 15 for the card identification mode and in Figure 16 for the data transfer mode. The commands are listed in the command tables (Table 9 - Table 16). The dependencies between current state, received command and following state are listed in Table 17. In the following sections, the different card operation modes will be described first. Thereafter, the restrictions for controlling the clock signal are defined. All SD Memory Card commands together with the corresponding responses, state transitions, error conditions and timings are presented in the succeeding sections.

Two operation modes are defined for the SD Memory Card system (host and cards):

- **Card identification mode**
  The host will be in card identification mode after reset and while it is looking for new cards on the bus. Cards will be in this mode after reset until the SEND_RCA command (CMD3) is received (in case of MultiMediaCard - SET_RCA command).

- **Data transfer mode**
  Cards will enter data transfer mode once their RCA is first published. The host will enter data transfer mode after identifying all the cards on the bus.

The following table shows the dependencies between operation modes and card states. Each state in the SD Memory Card state diagram (see Figure 15 and Figure 16) is associated with one operation mode:

Card state	Operation mode
Inactive State	inactive
Idle State	card identification mode
Ready State	
Identification State	
Stand-by State	data transfer mode
Transfer State	
Sending-data State	
Receive-data State	
Programming State	
Disconnect State	

Table 5: Overview of Card States vs. Operation modes

4.2 Card Identification Mode

While in card identification mode the host resets all the cards that are in card identification mode, validates operation voltage range, identifies cards and asks them to publish Relative Card Address (RCA). This operation is done to each card separately on its own CMD line. All data communication in the Card Identification Mode uses the command line (CMD) only.

4.2.1 Card Reset

The command GO_IDLE_STATE (CMD0) is the software reset command and sets each card into *Idle State* regardless of the current card state. Cards in *Inactive State* are not affected by this command.

After power-on by the host, all cards are in *Idle State*, including the cards that have been in *Inactive State* before.

After power-on or CMD0, all cards' CMD lines are in input mode, waiting for start bit of the next command. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

4.2.2 Operating Voltage Range Validation

All cards shall be able to establish communication with the host using any operating voltage in the maximal allowed voltage range specified in this standard (see Chapter 6.6). However, the supported minimum and maximum values for V_{DD} are defined in the Operation Conditions Register (OCR) and may not cover the whole range. Cards that store the CID and CSD data in the payload memory would be able to communicate these information only under data transfer V_{DD} conditions. That means if host and card have non compatible V_{DD} ranges, the card will not be able to complete the identification cycle, nor to send CSD data.

Therefore, a special command SD_SEND_OP_COND (ACMD41) is designed to provide SD Memory Card 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 5.1). Cards which can not perform data transfer in the specified range must discard themselves from further bus operations and go into *Inactive State*. The levels in the OCR register shall be defined accordingly (see Chapter 5.1). Note that ACMD41 is application specific command, therefore APP_CMD (CMD55) shall always precede ACMD41. The RCA to be used for CMD55 in *idle_state* shall be the card's default RCA = 0x0000.

The MultiMediaCard will not respond to ACMD41 (actually it will not respond to APP_CMD - CMD55, that preceding it). MultiMediaCard shall be initialized as per the MultiMediaCard spec, using SEND_OP_COND command (CMD1 of MultiMediaCard). The host should ignore an ILLEGAL_COMMAND status in the MultiMediaCard response to CMD3, as it is a residue of ACMD41 which is invalid in MultiMediaCard (CMD0, 1, 2 do not clear the status register). Actually, ACMD41 and CMD1 will be used by the host to distinguish between MultiMediaCard and SD Memory Cards in a system.

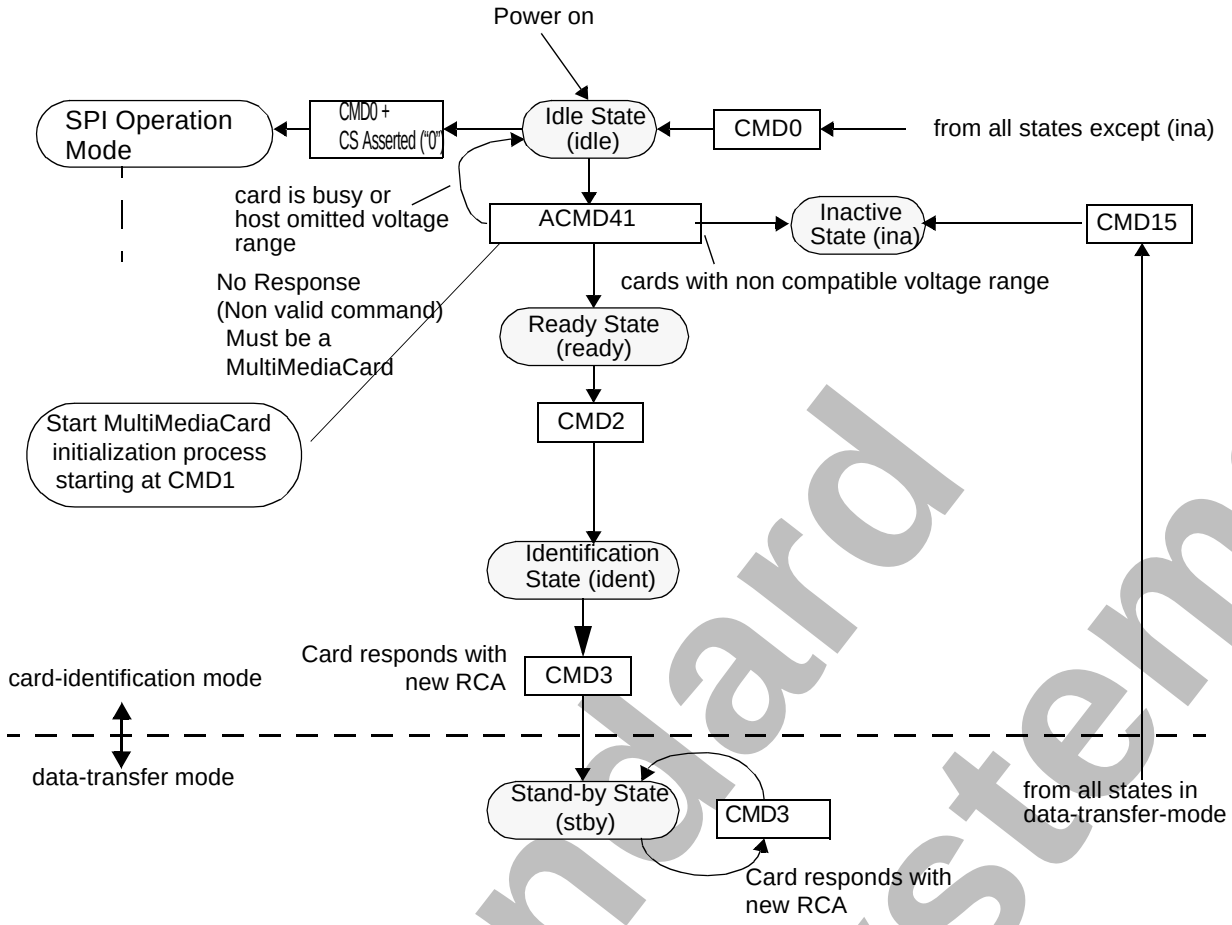


Figure 15: SD Memory Card state diagram (card identification mode)

By omitting the voltage range in the command, the host can query each card and determine the common voltage range before sending out-of-range cards into the *Inactive State*. This query should be used if the host is able to select a common voltage range or if a notification to the application of non usable cards in the stack is desired. Afterwards, the host must choose a voltage for operation and reissue ACMD41 with this condition, sending incompatible cards into the *Inactive State*.

The busy bit in the ACMD41 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 ACMD41 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 stack (sending CMD0 to all cards) and restart the initialization procedure. However, for accessing also the cards being already in *Inactive State*, a hard reset must be done by switching the power supply off and 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).

4.2.3 Card Identification Process

The host starts the card identification process with the identification clock rate f_{OD} (see Chapter 6.8). In SD Memory Card the CMD line output drives are push-pull drivers.

After the bus is activated the host will request the cards to send their valid operation conditions (ACMD41 preceding with APP_CMD - CMD55 with RCA=0x0000). The response to ACMD41 is the operation condition register of the card. The same command shall be send to all of the new cards in the system. Incompatible cards are sent into *Inactive State*. The host then issues the command ALL_SEND_CID (CMD2), to each card to get its unique card identification (CID) number. Card that is unidentified (i.e. which is in *Ready State*) sends its CID number as the response (on the CMD line). After the CID was sent by the card it goes into *Identification State*. Thereafter, the host issues CMD3 (SEND_RELATIVE_ADDR) asks the card to publish a new 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.* At this point, if the host wants that the card will have another RCA number, it may ask the card to publish a new number by sending another SEND_RELATIVE_ADDR command to the card. The last published RCA is the actual RCA number of the card.

The host repeats the identification process, i.e. the cycles with CMD2 and CMD3 for each card in the system.

After all the SD Memory Cards were initialized the host shall initialize the MultiMediaCards that are in the system (if any), using the CMD2 and CMD3 as given in the MultiMediaCard spec. Note that in the SD system all the cards are connected separately so each MultiMediaCard shall be initialized individually.

4.3 Data Transfer Mode

Until the contents of all CSD registers is known by the host, the f_{PP} clock rate must remain at f_{OD} because some cards may have operating frequency restrictions (see Chapter 6.8). 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.

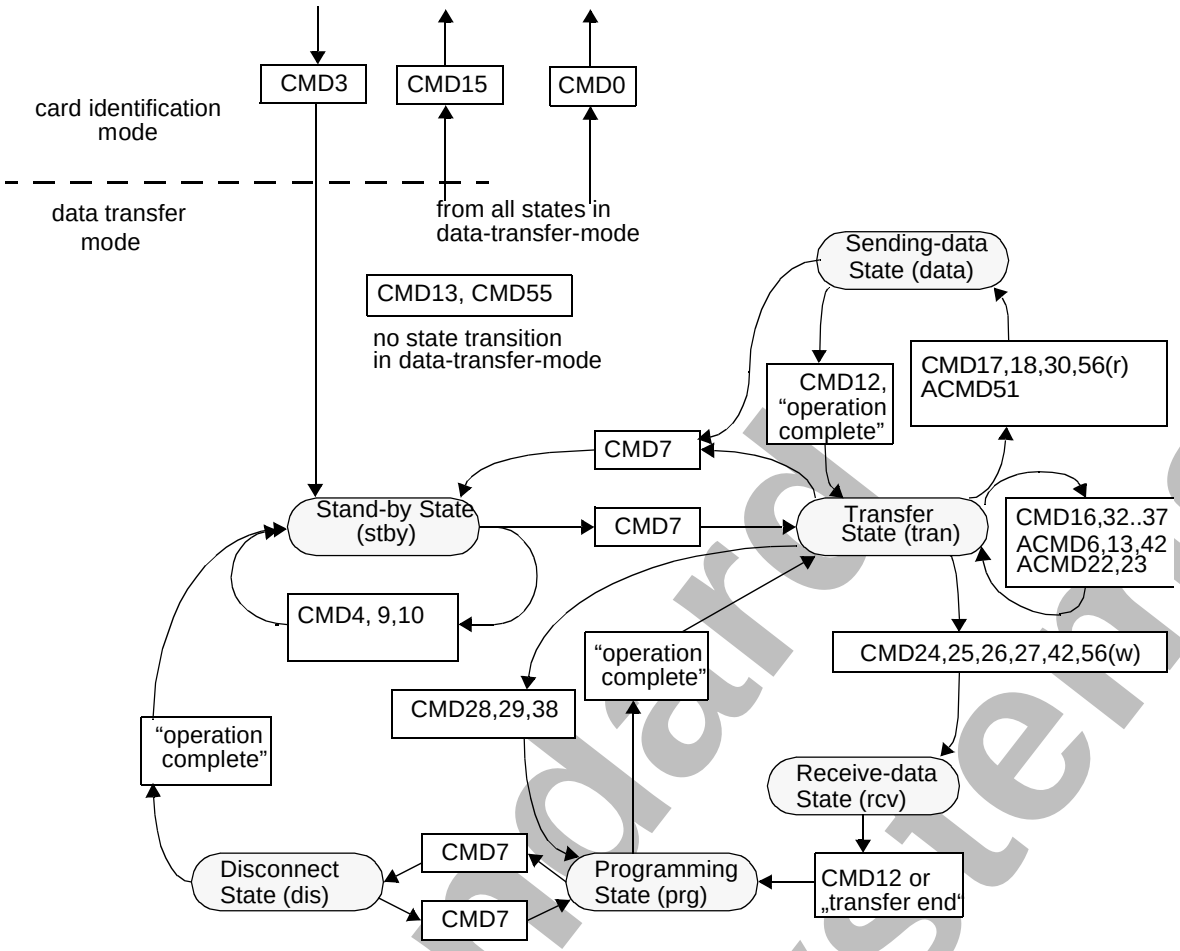


Figure 16: SD Memory Card state diagram (data transfer mode)

The broadcast command SET_DSR (CMD4) configures the driver stages of all identified cards. It programs their DSR registers corresponding to the application bus layout (length) and the number of cards on the bus and the data transfer frequency. The clock rate is also switched from f_{OD} to f_{PP} at that point. SET_DSR command is an option for the card and the host.

CMD7 is used to select one card and put it into the *Transfer State*. Only one card can be in the *Transfer State* at a given time. If a previously selected card is in the *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", all cards are put back to *Stand-by State*. This may be used before identifying new cards without resetting other already registered cards. Cards which already have an RCA do not respond to identification commands (CMD41, CMD2, CMD3, see Chapter 4.2.3) in this state.

Important Note: The card de-selection is done if certain card gets CMD7 with un-matched RCA. That happens automatically if selection is done to another card and the CMD lines are common. So, in SD Memory Card system it will be the responsibility of the host either to work with common CMD line (after initialization is done) - in that case the card de-selection will be done automatically (as in MultiMediaCard system) or if the CMD lines are separate then the host shall be aware to the neces-

sity to de-select cards.

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 16):

- 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), send write protect (CMD30), send scr (ACMD51) and general command in read mode (CMD56).
- 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), write CSD (CMD27), lock/unlock command (CMD42) and general command in write mode (CMD56).
- 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 programmed.
- The card may provide buffering for 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 SD Memory Card state diagram Figure 16), the DAT0 line will be kept low (BUSY).
- 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*. Actually if the CMD and DAT0 lines of the cards are kept separated and the host keep the busy DAT0 line disconnected from the other DAT0 lines (of the other cards) the host may access the other cards while the card is in busy.
- Parameter set commands are *not* allowed while card is programming.
  Parameter set commands are: set block length (CMD16), erase block start (CMD32) and erase block end (CMD33).
- Read commands are *not* allowed while card is programming.
- Moving another card from *Stand-by* to *Transfer State* (using CMD7) will not terminate erase and programming operations. The card will switch to the *Disconnect State* and will release the DAT line.
- A card can be reselected while in the *Disconnect State*, using CMD7. In this case the card will move to the *Programming 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 the host's responsibility to prevent this.

4.3.1 Wide Bus Selection/Deselection

Wide Bus (4 bit bus width) operation mode may be selected/deselected using ACMD6. The default bus width after power up or GO_IDLE (CMD0) is 1 bit bus width. ACMD6 command is valid in '*tran state*' only. That means that the bus width may be changed only after a card was selected (CMD7).

4.3.2 Data Read

The DAT bus line level is high by the pull-up when no data is transmitted. A transmitted data block consists of start bits (1 or 4 bits LOW), 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 end bits (1 or 4 bits HIGH) (see Figure 25-Figure 27). The data transmission is synchronous to the clock signal. The payload for block oriented data transfer is protected by 1 or 4 bits CRC check sum (see Chapter 3.2).

The Read operation from SD Memory Card may be interrupted by turning the power off. The SD Memory Card ensures that data is not destroyed during all the conditions except write or erase operations issued by the host even in the event of sudden shut down or removal.

- **Block Read**

Block read is block oriented data transfer. The basic unit of data transfer is a block whose maximum size is defined in the CSD (READ_BL_LEN). Smaller blocks whose starting and ending address are entirely contained within one physical block (as defined by READ_BL_LEN) may also be transmitted. It is a mandatory requirement that SD Memory Card shall have a capability to transfer blocks of 512 Bytes.

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. Blocks will be continuously transferred until a STOP_TRANSMISSION command (CMD12) is issued. The stop command has an execution delay due to the serial command transmission. The data transfer stops after the end bit of the stop 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 at the beginning of the first misaligned block, set the ADDRESS_ERROR error bit in the status register, abort transmission and wait in the *Data State* for a stop command.

4.3.3 Data Write

The data transfer format is similar to the data read format. For block oriented write data transfer, the CRC check bits are added to each data block. The card performs 1 or 4 bits CRC parity check (see Chapter 7.2) 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 - 27,42,56(w)) one or more blocks of data are transferred from the host to the card with 1 or 4 bits 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 and its 512bytes derivatives (for example: If write block length=1024bytes then write blocks of 1024 and 512bytes are supported). If WRITE_BL_PARTIAL is allowed (=1) then smaller blocks, up to resolution of one byte, can be used as well. If the CRC fails, the card shall indicate the failure on the DAT 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.

Multiple block write command shall be used rather than continuous single write command to make faster write operation.

If the host uses partial blocks whose accumulated length is not block aligned and block misalign-

ment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card shall detect the block misalignment error and abort programming before the beginning of the first misaligned block. The card shall set the ADDRESS_ERROR error bit in the status register, and while ignoring all further data transfer, wait in the *Receive-data-State* for a stop command.

The write operation shall also be aborted if the host tries to write over a write protected area. In this case, however, the card shall set the WP_VIOLATION bit.

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 (to select a different card) which will displace the card into the *Disconnect State* and release the DAT line without interrupting the write operation. When reselecting the card, it will reactivate busy indication by pulling DAT to low if programming is still in progress and the write buffer is unavailable. Actually, the host may perform simultaneous write operation to several cards with interleaving process. The interleaving process can be done by accessing each card separately while other cards are in busy. This process can be done by proper CMD and DAT0-3 line manipulations (disconnection of busy cards).

- **Pre-erase setting prior to a multiple block write operation**

Setting a number of write blocks to be pre_erased (ACMD23) will make a following Multiple Block Write operation faster compared to the same operation without preceding ACMD23. The host will use this command to define how many number of write blocks are going to be send in the next write operation. If the host will terminate the write operation (Using stop transmission) before all the data blocks sent to the card the content of the remaining write blocks is undefined(can be either erased or still have the old data). If the host will send more number of write blocks than defined in ACMD23 the card will erase block one by one(as new data is received). This number will be reset to the default(=1) value after Multiple Blocks Write operation.

It is recommended using this command preceding CMD25, some of the cards will be faster for Multiple Write Blocks operation. Note that The host must send ACMD23 just before WRITE command if the host wants to use the pre-erase feature. If not, pre-erase-count might be cleared automatically when another commands (ex: Security Aplication Commands) are executed.

- **Send Number of Written Blocks**

Systems that use PipeLine mechanism for data buffers management are, in some cases, unable to determine which block was the last to be well written to the flash if an error occurs in the middle of a Multiple Blocks Write operation. The card will respond to ACMD22 with the number of well written blocks.

4.3.4 Erase

It is desirable to erase many write blocks simultaneously in order to enhance the data throughput.

Identification of these write blocks is accomplished with the ERASE_WR_BLK_START(CMD32), ERASE_WR_BLK_END(CMD33) commands.

The host must adhere to the following command sequence: ERASE_WR_BLK_START, ERASE_WR_BLK_END and ERASE (CMD38).

If an erase (CMD38) or address setting (CMD32, 33) command is received out of sequence, the card shall set the ERASE_SEQ_ERROR bit in the status register and reset the whole sequence.

If an out of sequence command (except SEND_STATUS) is received, the card shall set the ERASE_RESET status bit in the status register, reset the erase sequence and execute the last command.

If the erase range includes write protected sectors, they shall be left intact and only the non protected sectors shall be erased. The WP_ERASE_SKIP status bit in the status register shall be set.

The address field in the address setting commands is a write block address in byte units. The card will ignore all LSB's below the WRITE_BLK_LEN (see CSD) size.

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 or perform card disconnection, as described in the Block Write section, above.

The data at the card after an erase operation is either '0' or '1', depends on the card vendor.

The SCR register bit DATA_STAT_AFTER_ERASE (bit 55) defines whether it is '0' or '1'.

4.3.5 Write Protect Management

Three write protect methods are supported in the SD Memory Card as follows:

- Mechanical write protect switch (Host responsibility only)
- Card internal write protect (Card's responsibility)
- Password protection card lock operation.

- **Mechanical Write Protect Switch**

A mechanical sliding tablet on the side of the card (refer to the mechanical description Chapter 8) will be used by the user to indicate that a given card is write protected or not. If the sliding tablet is positioned in such a way that the window is open it means that the card is write protected. If the window is close the card is not write protected.

A proper, matched, switch on the socket side will indicate to the host that the card is write protected or not. It is the responsibility of the host to protect the card. The position of the write protect switch is un-known to the internal circuitry of the card.

The same mechanical switch that is used for write protect may be used by the host for Card insertion/removal detection. Refer to Application Note "Card Detection Implementations" (Chapter 9) for further information.

- **Card's Internal Write Protection (Optional)**

Card data may be protected against either erase or write. The entire card may be permanently write protected by the manufacturer or content provider by setting the permanent or temporary write protect bits in the CSD. For cards which support write protection of groups of sectors by setting the WP_GRP_ENABLE bit in the CSD, portions of the data may be protected (in units of WP_GRP_SIZE sectors as specified in the CSD), and the write protection may be changed by the application. 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.

The Password Card Lock protection is described in the following section.

4.3.6 Card Lock/Unlock Operation (Optional)

The password protection feature enables the host to lock a card while 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.

Locked cards respond to (and execute) all commands in the "basic" command class (class 0), ACMD41 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') 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	PWDS_LEN							
2	Password data							
...								
PWDS_LEN + 1								

Table 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
- **PWDS_LEN**: Defines the following password/s length (in bytes). In case of Password change, this field include the total password lengths of old and new passwords.
- **PWD:** In case of set new password contains the new password. In case of password change it contains the old password followed by new password. All other commands it contains the current password.

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 a 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 (PWDS_LEN) and the password itself. In case that a password *replacement* is done, then the length value (PWDS_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 password. Note that card shall handle internally the calculation of the new password length by subscribing the old password length from PWDS_LEN field.
  - 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 PWDS_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 a 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 (PWDS_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 content 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 a 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 (PWDS_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 a 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 (PWDS_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 a 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. An attempt to force erase on an unlocked card will fail and LOCK_UNLOCK_FAILED error bit will be set in the status register.

4.3.7 Copyright Protection

Detailed description of the Copyrights Protection mechanism and the related security SD Memory Card commands can be found in "SD Memory Card Security Specification" document. All the SD Memory Card security related commands shall be operated in data transfer mode of operation.

As defined in the SDMI spec the data content that is saved in the card is saved already encrypted and it pass transparently to/from the card. NO operation is done on the data and there is no restriction to read the data at any time. Associated to every data packet (song, for example) that is saved in the un-protected memory there is a special data that shall be saved in a protected memory area. For any access (any Read or Write or Erase Command) from/to the data in the protected area, an authentication procedure shall be done between the card and the connected device, either the LCM

(PC for example) or the PD (Portable Device - SD Player for example). After the authentication process was passed OK, the card is ready to accept or give data from/to the connected device. While the card is in the secured mode of operation (after the authentication succeeded) the argument and the associated data that is sent to the card or read from the card are encrypted. At the end of the Read/Write/Erase operation the card gets out automatically of its secured mode.

4.3.8 Application specific commands:

The SD Memory Card is defined to be protocol forward compatible to the MultiMediaCard Standard.

The SD Memory Card system is designed to provide a standard interface for a variety applications types. In order to keep future compatibility to the MultiMediaCard standard together with new SD Memory Card specific commands the SD Memory Card use the Application Specific commands feature to implement its proprietary commands. Following is a description of the APP_CMD and GEN_CMD as were defined in the MultiMediaCard spec.

- **Application Specific Command – APP_CMD (CMD55)**

This command, when received by the card, will cause the card to interpret the following command as an application specific command, ACMD. The ACMD has the same structure as of regular MultiMediaCard standard commands and it may have the same CMD number. The card will recognize it as ACMD by the fact that it appears after APP_CMD.

The only effect of the APP_CMD is that if the command index of the, immediately, following command has an ACMD overloading it, the non standard version will be used. If, as an example, a card has a definition for ACMD13 but not for ACMD7 then, if received immediately after APP_CMD command, Command 13 will be interpreted as the non standard ACMD13 but, command 7 as the standard CMD7. In order to use one of the manufacturer specific ACMD's the host will:

- Send APP_CMD. The response will have the APP_CMD bit (new status bit) set signaling to the host that ACMD is now expected.
- Send the required ACMD. The response will have the APP_CMD bit set, indicating that the accepted command was interpreted as ACMD. If a non-ACMD is sent then it will be respected by the card as normal SD Memory Card command and the APP_CMD bit in the Card Status stays clear.

If a non valid command is sent (neither ACMD nor CMD) then it will be handled as a standard SD Memory Card illegal command error.

From the SD Memory Card protocol point of view the ACMD numbers will be defined by the manufacturers with some restrictions. The following ACMD numbers are reserved for the SD Memory Card proprietary applications and may not be used by any SD Memory Card manufacturer:

ACMD6, ACMD13, ACMD17-25, ACMD38-49, ACMD51.

- **General Command - GEN_CMD (CMD56)**

The bus transaction of the GEN_CMD is the same as the single block read or write commands (CMD24 or CMD17). The difference is that the argument denotes the direction of the data transfer (rather than the address) and the data block is not a memory payload data but has a vendor specific format and meaning. The card shall be selected ('*tran_state*') before sending CMD56. The data block size is the BLOCK_LEN that was defined with CMD16. The response to CMD56 will be R1.

Note that there are no reserved data pattern (of the associated data block) for SD Memory Card specific applications.

4.4 Clock Control

The SD Memory Card bus clock signal can be used by the host to turn the cards 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. For example, in a case that a host with 512Bytes of data buffer would like to transfer data to a card with 1KByte write blocks. So, in order to preserve a continuous data transfer, from the cards point of view, the clock to the card shall be stopped after the first 512Bytes. Then the host will fill it internal buffer with another 512Bytes. After the second half of the write block is ready in the host, it will continue the data transfer to the card by re-starting the clock supply. In such a way the card does not recognize any interruptions in the data transfer.

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 cards, 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 SD Memory Card 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 DAT line down, forever.

4.5 Cyclic redundancy codes (CRC)

The CRC is intended for protecting SD Memory Card commands, responses and data transfer against transmission errors on the SD Memory Card 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 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: $G(x) = x^7 + x^3 + 1$.

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

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

The first bit is the most left bit of the corresponding bitstring (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).

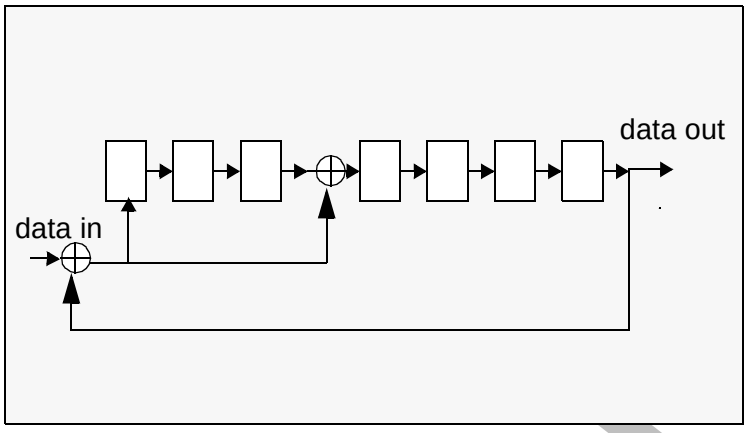


Figure 17: CRC7 generator/checker

- **CRC16**

In case of one DAT line usage (as in MultiMediaCard) than 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 $G(x) = x^{16} + x^{12} + x^5 + 1$

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

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

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 method shall be used in single DAT line mode and in wide bus mode.

In wide bus mode, the CRC16 is done on each line separately.

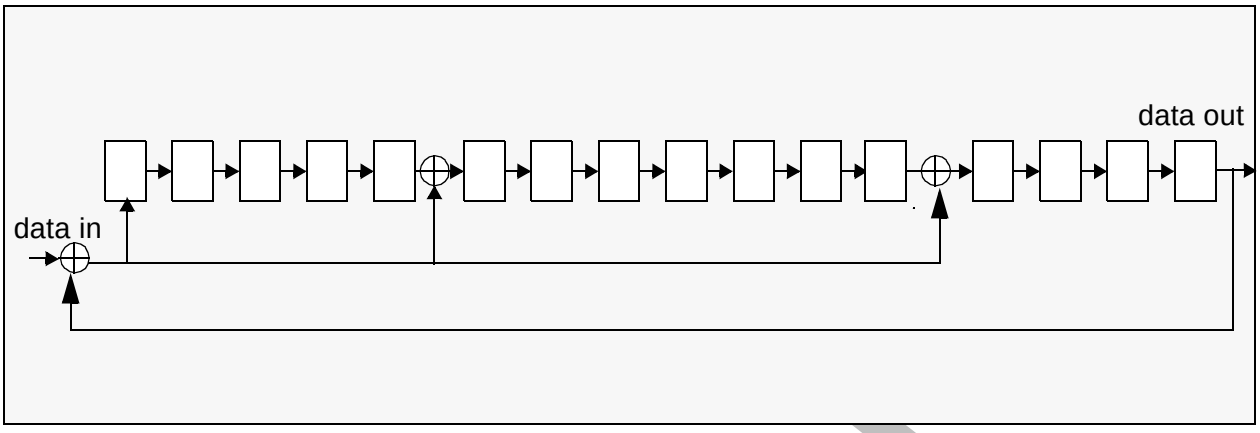


Figure 18: CRC16 generator/checker

4.6 Error Conditions

4.6.1 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, a card shall not change its state, shall not response 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 15 to Figure 16). Table 17 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. CMD5).

4.6.2 Read, Write and Erase Time-out Conditions

The times after which a time-out condition for read/write/erase operations occurs are (card independent) **either 100 times longer** than the typical access/program times for these operations given below or **or 100ms (the lower of them)**. A card shall complete the command within this time period, or give up and return an error message. If the host does not get any response with the given 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 4.12). 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 and should be used by the host to calculate throughput and the maximal frequency for stream read.

- **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 (WRITE_BL) to be erased multiplied by the block write delay.

4.7 Commands

4.7.1 Command Types

There are four kinds of commands defined to control the SD Memory Card:

- broadcast commands (bc), no response - The broadcast feature is only if all the CMD lines are connected together in the host. If they are separated then each card will accept it separately on his turn.
- broadcast commands with response (bcr)
  response from all cards simultaneously - Since there is no Open Drain mode in SD Memory Card this type of commands shall be used only if all the CMD lines are separated - the command will be accepted and responded by every card separately.
- addressed (point-to-point) commands (ac)
  no data transfer on DAT
- addressed (point-to-point) data transfer commands (adtc)
  data transfer on DAT

All commands and responses are sent over the CMD line of the SD Memory Card. The command transmission always starts with the left bit of the bitstring corresponding to the command codeword.

4.7.2 Command Format

All commands have a fixed code length of 48 bits, needing a transmission time of 2.4 µs @ 20 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 7: Command Format

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 7.2 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 9-Table 16.

4.7.3 Command Classes (Redefined for SD Memory Card)

The command set of the SD Memory Card system is divided into several classes (See Table 8). Each class supports a set of card functionalities.

Class 0, 2, 4, 5 and 8 are mandatory and shall be supported by all SD Memory Cards. The other classes are 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)	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9-11**
Supported commands	**class description**	basic	reserved	block read	reserved	block write	erase	write protection	lock card	application specific	reserved
CMD0	Mandatory	+									
CMD2	Mandatory	+									
CMD3	Mandatory	+									
CMD4	Mandatory	+									
CMD7	Mandatory	+									
CMD9	Mandatory	+									
CMD10	Mandatory	+									
CMD12	Mandatory	+									
CMD13	Mandatory	+									
CMD15	Mandatory	+									
CMD16	Mandatory			+		+					
CMD17	Mandatory			+							
CMD18	Mandatory			+							
CMD24	Mandatory					+					
CMD25	Mandatory					+					
CMD27	Mandatory					+					
CMD28	Optional							+			
CMD29	Optional							+			
CMD30	Optional							+			

Table 8: Card Command Classes (CCCs)

	Card Command Class (CCC)	0	1	2	3	4	5	6	7	8	9-11
Supported commands	**class description**	basic	reserved	block read	reserved	block write	erase	write protection	lock card	application specific	reserved
CMD32	Mandatory						+				
CMD33	Mandatory						+				
CMD38	Mandatory						+				
CMD42	Optional								+		
CMD55	Mandatory									+	
CMD56	Mandatory									+	
ACMD6	Mandatory									+	
ACMD13	Mandatory									+	
ACMD22	Mandatory									+	
ACMD23	Mandatory									+	
ACMD41	Mandatory									+	
ACMD42	Mandatory									+	
ACMD51	Mandatory									+	

Table 8: Card Command Classes (CCCs)

4.7.4 Detailed Command Description

The following tables define in detail all SD Memory Card bus commands. The responses R1-R3, R6 are defined in Chapter 4.9. The registers CID, CSD and DSR are described in Chapter 5.

CMD INDEX	type	argument	resp	abbreviation	command description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	resets all cards to idle state
CMD1	reserved				
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	asks any card to send the CID numbers on the CMD line (any card that is connected to the host will respond)
CMD3	bcr	[31:0] stuff bits	R6	SEND_RELATIVE_ ADDR	ask the card to publish a new relative address (RCA)
CMD4	bc	[31:16] DSR [15:0] stuff bits	-	SET_DSR	programs the DSR of all cards
CMD5	reserved				
CMD6	reserved				

Table 9: Basic commands (class 0)

CMD INDEX	type	argument	resp	abbreviation	command description
CMD7	ac	[31:16] RCA [15:0] stuff bits	R1b (only from the selected card)	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 all.
CMD8	reserved				
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	reserved				
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	reserved				
CMD15	ac	[31:16] RCA [15:0] stuff bits	-	GO_INACTIVE_ STATE	sets the card to inactive state in order to protect the card stack against communication breakdowns.

Table 9: Basic commands (class 0)

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. Supported only if Partial block RD/WR operation are allowed in CSD.
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_ BLOCK	reads a block of the size selected by the SET_BLOCKLEN command.[1]
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_ BLOCK	continuously transfers data blocks from card to host until interrupted by a STOP_TRANSMISSION command.
CMD19 ... CMD23	reserved				

Table 10: Block oriented read commands (class 2)

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

CMD INDEX	type	argument	resp	abbreviation	command description
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	writes a block of the size selected by the SET_BLOCKLEN command.[1]
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	continuously writes blocks of data until a STOP_TRANSMISSION follows.
CMD26	Reserved For Manufacturer				
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	programming of the programmable bits of the CSD.

Table 11: Block oriented write commands (class 4)

1)The data transferred must not cross a physical block boundary unless WRITE_BLK_MISALIGN is set in the CSD. In case that write partial blocks is not supported then the block length=default block length (given in 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. [1]
CMD31	reserved				

Table 12: Block oriented write protection commands (class 6)

1)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 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	ac	[31:0] data address	R1	ERASE_WR_BLK_START	sets the address of the first write-block to be erased.
CMD33	ac	[31:0] data address	R1	ERASE_WR_BLK_END	sets the address of the last write block of the continuous range to be erased.
CMD34 ... CMD37	reserved				

Table 13: Erase commands (class 5)

CMD INDEX	type	argument	resp	abbreviation	command description
CMD38	ac	[31:0] stuff bits	R1b	ERASE	erases all previously selected write blocks.
CMD39	reserved				
CMD40					Non Valid in SD Memory Card - Reserved for MultiMediaCard I/O mode
CMD41	reserved				

Table 13: Erase commands (class 5)

CMD INDEX	type	argument	resp	abbreviation	command description
CMD42	adtc	[31:0] stuff bits.	R1	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 14: Lock card (class 7)

CMD INDEX	type	argument	resp	abbreviation	command description
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application specific command rather than a standard command
CMD56	adtc	[31:1] stuff bits. [0]: RD/ WR[1]	R1	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 specific commands. The size of the data block shall be set by the SET_BLOCK_LEN command.
CMD57 ... CMD59	reserved				
CMD60 -63	reserved for manufacturer				

Table 15: Application specific commands (class 8)

1) RD/WR: “1” the host gets a block of data from the card.
“0” the host sends block of data to the card.

All the application specific commands (given in Table 15) are supported if Class 8 is allowed (mandatory in SD Memory Card).

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

The following table describes all the application specific commands supported/reserved by the SD Memory Card. All the following ACMDs shall be preceded with APP_CMD command (CMD55).

ACMD INDEX	type	argument	resp	abbreviation	command description
ACMD6	ac	[31:2] stuff bits [1:0]bus width	R1	SET_BUS_WIDTH	Defines the data bus width ('00'=1bit or '10'=4 bits bus) to be used for data transfer. The allowed data bus widths are given in SCR register.
ACMD13	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD Memory Card status. The status fields are given in Table 24.
ACMD17	reserved				
ACMD18	--	--	--	--	Reserved for SD security applications[1]
ACMD19 to ACMD21	reserved				
ACMD22	adtc	[31:0] stuff bits	R1	SEND_NUM_WR_ BLOCKS	Send the number of the written (without errors) write blocks. Responds with 32bit+CRC data block.
ACMD23	ac	[31:23] stuff bits [22:0]Number of blocks	R1	SET_WR_BLK_ ERASE_COUNT	Set the number of write blocks to be pre-erased before writing (to be used for faster Multiple Block WR command). "1"=default (one wr block)[(2)].
ACMD24	reserved				
ACMD25	--	--	--	--	Reserved for SD security applications[1]
ACMD26	--	--	--	--	Reserved for SD security applications[1]
ACMD38	--	--	--	--	Reserved for SD security applications[1]
ACMD39 to ACMD40	reserved				
ACMD41	bcr	[31:0]OCR without busy	R3	SD_APP_OP_COND	Asks the accessed card to send its operating condition register (OCR) content in the response on the CMD line.
ACMD42	ac	[31:1] stuff bits [0]set_cd	R1	SET_CLR_CARD_ DETECT	Connect[1]/Disconnect[0] the 50KOhm pull-up resistor on CD/DAT3 (pin 1) of the card. The pull-up may be used for card detection.
ACMD43 ACMD49	--	--	--	--	Reserved for SD security applications[1]
ACMD51	adtc	[31:0] staff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

(1) Refer to "SD Memory Card Security Specification" for detailed explanation about the SD Security Features
(2) Command STOP_TRAN (CMD12) shall be used to stop the transmission in Write Multiple Block whether the pre-erase (ACMD23) feature is used or not.

Table 16: Application Specific Commands used/reserved by SD Memory Card

4.8 Card State Transition Table

Table 17 defines the card state transitions in dependency of the received command.

	current state									
	idle	**ready**	**ident**	**stby**	**tran**	**data**	**rcv**	**prg**	**dis**	**ina**
command	**changes to**									
class independent										
CRC error	-	-	-	-	-	-	-	-	-	-
command not supported	-	-	-	-	-	-	-	-	-	-
class 0										
CMD0	idle	idle	idle	idle	idle	idle	idle	idle	idle	-
CMD2	-	ident	-	-	-	-	-	-	-	-
CMD3	-	-	stby	stby	-	-	-	-	-	-
CMD4	-	-	-	stby	-	-	-	-	-	-
CMD7, card is addressed	-	-	-	tran	-	-	-	-	prg	-
CMD7, card is not addressed	-	-	-	stby	stby	stby	-	dis	-	-
CMD9	-	-	-	stby	-	-	-	-	-	-
CMD10	-	-	-	stby	-	-	-	-	-	-
CMD12	-	-	-	-	-	tran	prg	-	-	-
CMD13	-	-	-	stby	tran	data	rcv	prg	dis	-
CMD15	-	-	-	ina	ina	ina	ina	ina	ina	-
class 2										
CMD16	-	-	-	-	tran	-	-	-	-	-
CMD17	-	-	-	-	data	-	-	-	-	-
CMD18	-	-	-	-	data	-	-	-	-	-
class 4										
CMD16	see class 2									
CMD24	-	-	-	-	rcv	-	-	-	-	-
CMD25	-	-	-	-	rcv	-	-	-	-	-
CMD27	-	-	-	-	rcv	-	-	-	-	-
class 6										
CMD28	-	-	-	-	prg	-	-	-	-	-
CMD29	-	-	-	-	prg	-	-	-	-	-
CMD30	-	-	-	-	data	-	-	-	-	-
class 5										

Table 17: Card state transition table

	current state									
	idle	**ready**	**ident**	**stby**	**tran**	**data**	**rcv**	**prg**	**dis**	**ina**
CMD32	-	-	-	-	tran	-	-	-	-	-
CMD33	-	-	-	-	tran	-	-	-	-	-
CMD38	-	-	-	-	prg	-	-	-	-	-
class 7										
CMD42	-	-	-	-	rcv	-	-	-	-	-
class 8										
CMD55	idle	-	-	stby	tran	data	rcv	prg	dis	-
CMD56; RD/WR = 0	-	-	-	-	rcv	-	-	-	-	-
CMD56; RD/WR = 1	-	-	-	-	data	-	-	-	-	-
ACMD6	-	-	-	-	tran	-	-	-	-	-
ACMD13	-	-	-	-	tran	-	-	-	-	-
ACMD22	-	-	-	-	tran	-	-	-	-	-
ACMD23	-	-	-	-	tran	-	-	-	-	-
ACMD18,25,26,38, 43,44,45,46,47,48,49	Refer to “SD Memory Card Security Specification” for explanation about the SD Security Features									
ACMD41, card V_{DD} range compatible	ready	-	-	-	-	-	-	-	-	-
ACMD41, card is busy	idle	-	-	-	-	-	-	-	-	-
ACMD41, card V_{DD} range not compatible	ina	-	-	-	-	-	-	-	-	-
ACMD42	-	-	-	-	tran	-	-	-	-	-
ACMD51	-	-	-	-	data	-	-	-	-	-
class 9- 11										
CMD41; CMD43...CMD54, CMD57-CMD59	reserved									
CMD60...CMD63	reserved for manufacturer									

Table 17: Card state transition table

The state transitions of the SD Memory Card application specific commands are given under Class 8, above.

4.9 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 four 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. Note that in case that data transfer to the card is involved then a busy signal may appear on the data line after the transmission of each block of data. The host shell check for busy after data block transmission.

The card status is described in Chapter 4.10.

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 18: Response R1

- **R1b** is identical to R1 with an optional busy signal transmitted on the data line. The card may become busy after receiving these commands based on its state prior to the command reception. The Host shell check for busy at the response. Refer to Chapter 4.12.3 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	reserved	CID or CSD register incl. internal CRC7	end bit

Table 19: Response R2

- **R3** (OCR register): code length 48 bits. The contents of the OCR register is sent as a response to ACMD41.

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	reserved	OCR register	reserved	end bit

Table 20: Response R3

- **R6** (Published RCA response): code length 48 bit. The bits 45:40 indicate the index of the

Bit position	47	46	[45:40]	[39:8] Argument field		[7:1]	0
Width (bits)	1	1	6	16	16	7	1
Value	'0'	'0'	x	x	x	x	'1'
Description	start bit	transmission bit	command index ('000011')	New published RCA [31:16] of the card	[15:0] *card status* bits: 23,22,19,12:0 (see Table 22)	CRC7	end bit

Table 21: Response R6

command to be responded to - in that case it will be '000011' (together with bit 5 in the status bits it means = CMD3). The 16 MSB bits of the argument field are used for the Published RCA number.

4.10 SD Memory Card Status

SD Memory Card supports two card status field as follows:

- '*Card Status*': compatible to the MultiMediaCard protocol.

- '*SD_Status*': Extended status field of 512bits that supports special features of the SD Memory Card and future Application Specific features.

4.10.1 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 (which may be stored in a local status register) to the host. If not specified otherwise, the status entries are always related to the previous issued command. The semantics of this register is according to the CSD entry SPEC_VERS (see Chapter 5.3), indicating the version of the response formats (possibly used for later extensions).

Table 22 defines the different entries of the status. The type and clear condition fields in the table are abbreviated as follows:

- **Type:**
  - E: Error bit.

- S: Status bit.
- R: Detected and set for the actual command response.
- X: Detected and set during command execution. The host must poll the card by issuing the status command in order to read these bits.

- **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	Value	Description	Clear Condition
31	OUT_OF_RANGE	E R	'0'= no error '1'= error	The command's argument was out of the allowed range for this card.	C
30	ADDRESS_ERROR	E R X	'0'= no error '1'= error	A misaligned address which did not match the block length was used in the command.	C
29	BLOCK_LEN_ERROR	E R	'0'= no error '1'= error	The transferred block length is not allowed for this card, or the number of transferred bytes does not match the block length.	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 write-blocks for erase occurred.	C
26	WP_VIOLATION	E R X	'0'= not protected '1'= protected	Attempt to program a write protected block.	C
25	CARD_IS_LOCKED	S X	'0' = card unlocked '1' = card locked	When set, signals that the card is locked by the host	A
24	LOCK_UNLOCK_FAILED	E R X	'0' = no error '1' = error	Set when a sequence or password error has been detected in lock/unlock card command or if there was an attempt to access a locked card	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 X	'0'= no error '1'= error	Internal card controller error	C
19	ERROR	E R X	'0'= no error '1'= error	A general or an unknown error occurred during the operation.	C

Table 22: Card status

Bits	Identifier	Type	Value	Description	Clear Condition
18	UNDERRUN	E X	'0'= no error '1'= error	The card could not sustain data transfer in stream read mode	C
17	OVERRUN	E X	'0'= no error '1'= error	The card could not sustain data programming in stream write mode	C
16	CID/ CSD_OVERWRITE	E R 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	S X	'0'= not protected '1'= protected	Only partial address space was erased due to existing write protected blocks.	C
14	CARD_ECC_DISABLED	S X	'0'= enabled '1'= disabled	The command has been executed without using the internal ECC.	A
13	ERASE_RESET	S R	'0'= cleared '1'= set	An erase sequence was cleared before executing because an out of erase sequence command was received	C
12:9	CURRENT_STATE	S X	0 = idle 1 = ready 2 = ident 3 = stby 4 = tran 5 = data 6 = rcv 7 = prg 8 = dis 9-15 = 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 to the next command. The four bits are interpreted as a binary coded number between 0 and 15.	B
8	READY_FOR_DATA	S X	'0'= not ready '1'= ready	corresponds to buffer empty signalling on the bus	A
7:6					
5	APP_CMD	S R	'0' = Disabled '1' = Enabled	The card will expect ACMD, or indication that the command has been interpreted as ACMD	C
4	reserved				
3	AKE_SEQ_ERROR (SD Memory Card app. spec.)	E R	'0' = no error '1' = error	Error in the sequence of authentication process	C

Table 22: Card status

Bits	Identifier	Type	Value	Description	Clear Condition
2	reserved for application specific commands				
1, 0	reserved for manufacturer test mode				

Table 22: Card status

The following table defines for each command responded by a R1 response the affected bits in the status field. An 'x' means the error/status bit may be set in the response to the respective command.

CMD#	Response Format 1 Status bit #																					
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12:9	8	5
3(1)									x	x			x							x		
7					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
12	x	x				x	x		x	x	x	x	x	x	x			x		x		
13	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x		x	x	
16			x		x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
17	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
18	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
24	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
25	x	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
26					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
27					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
28	x				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
29	x				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
30	x				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
32	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
33	x			x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
38				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
42					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		
55					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
56					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
ACMD6	x				x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
ACMD13					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
ACMD22					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
ACMD23					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
ACMD42					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x
ACMD51					x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x		x

Table 23: Card status field / command - cross reference

(1) The response to CMD3 is R6 that includes only bits 23, 22, 19 and 12:9 out of the Card Status

4.10.2 *SD Status*

The SD Status contains status bits that are related to the SD Memory Card proprietary features and may be used for future application specific usage. The size of the SD Status is one data block of 512bit. The content of this register is transmitted to the Host over the DAT bus along with 16 bit CRC. The SD Status is sent to the host over the DAT bus if ACMD13 is sent (CMD55 followed with CMD13). ACMD13 can be sent to a card only in '*tran_state'* (card is selected). SD Status structure is described in bellow.

The same abbreviation for 'type' and 'clear condition' were used as for the Card Status above.

Bits	Identifier	Type	Value	Description	Clear Condition
511: 510	DAT_BUS_WIDTH	S R	'00'= 1 (default) '01'= reserved '10'= 4 bit width '11'= reserved	Shows the currently defined data bus width that was defined by SET_BUS_WIDTH command	A
509	SECURED_MODE	S R	'0'= Not in the mode '1'= In Secured Mode	Card is in Secured Mode of operation (refer to "SD Security Specification").	A
508: 496	reserved				
495: 480	SD_CARD_TYPE	SR	All '0'= SD Memory Cards (as defined in Physical Spec Ver.1	Each bit will define different SD Type [Various SD Types to be defined in the future]	A
479: 448	SIZE_OF_PROTECTED_AREA	SR	Size of protected area (in units of MULT*BLOCK_LEN refer to CSD register Table 5.3)	Shows the size of protected area. The actual area = (SIZE_OF_PROTECTED_AREA) * MULT * BLOCK_LEN.	A
447: 312	reserved				
311: 0	reserved for manufacturer				

Table 24: SD Card Status

4.11 Memory Array Partitioning

The basic unit of data transfer to/from the SD Memory Card 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 devices which have erasable memory cells, special erase commands are defined. The granularity of the erasable units is in general not the same as for the block oriented commands:

- Sector: is the unit which is related to the erase commands. Its size is the number of blocks which will be erased in one portion. The size of a sector is fixed for each device. The information about the sector size (in blocks) is stored in the CSD.

For devices which include a write protection:

- WP-Group: is the minimal unit which may have individual write protection. Its size is the number of groups which will be write protected by one bit. The size of a WP-group is fixed for each device. The information about the size is stored in the CSD.

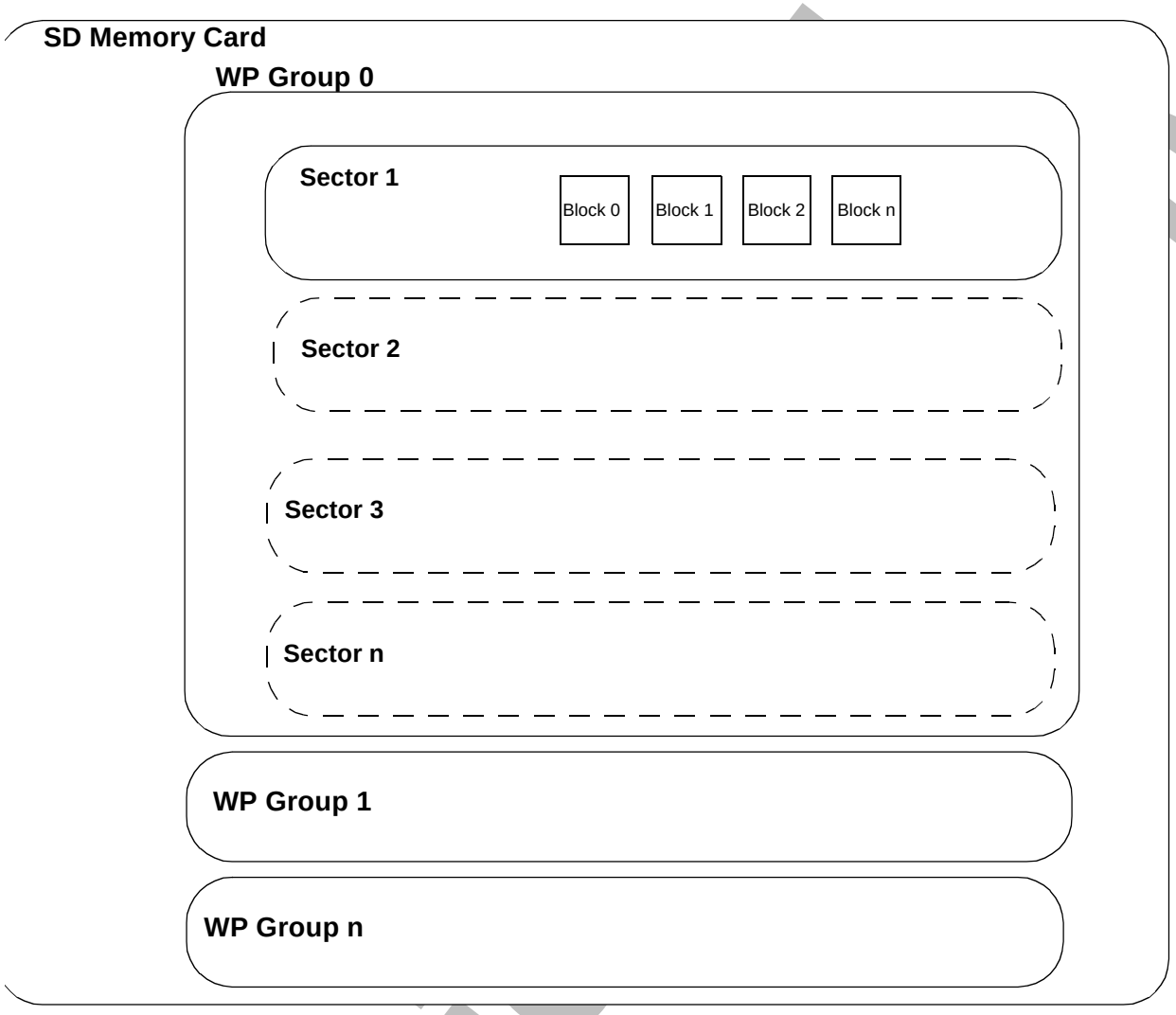


Figure 19: Write Protection hierarchy

Each WP-group may have an additional write protection bit. The write protection bits are programmable via special commands (see Chapter 4.7.4).

Both functions are optional and only useful for writable/erasable devices. The write protection may

also be useful for multi type cards (e.g. a ROM - Flash combination). The information about the availability is stored in the CSD.

4.12 Timings

All timing diagrams use the following schematics and abbreviations:

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')
D	Data bits
X	Don't Care data bits (from card)
*	Repetition
CRC	Cyclic redundancy check bits (7 bits)
	Card active
	Host active

Table 25: 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 26.

4.12.1 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 timing for CMD2 and ACMD41 is given bellow. The command is followed by a period of two Z bits (allowing time for direction switching on the bus) and then by P bits pushed up by the responding card. The card response to the host command starts after N_{ID} clock cycles.

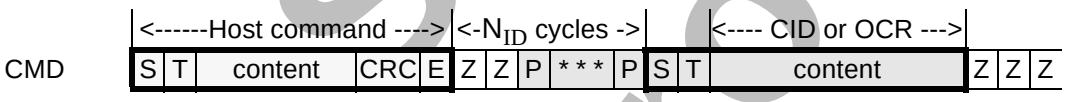


Figure 20: Identification timing (card identification mode)

- **Assign a card relative address**

The SEND_RELATIVE_ADDR (CMD 3) for SD Memory Card timing is given bellow. Note that CMD3 command's content, functionality and timing are different for MultiMediaCard. The minimum delay between the host command and card response is N_{CR} clock cycles.

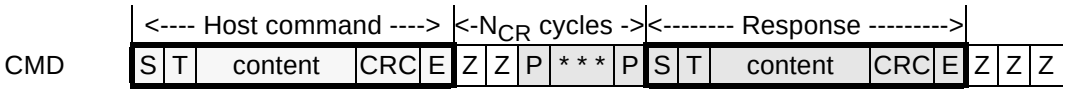


Figure 21: SEND_RELATIVE_ADDR timing

- **Data transfer mode.**

After the card published it own RCA it will switch to data transfer mode. The command is followed by a period of two Z bits (allowing time for direction switching on the bus) and then by P bits pushed up by the responding card. This timing diagram is relevant for all responded host commands except and ACMD41 and CMD2:

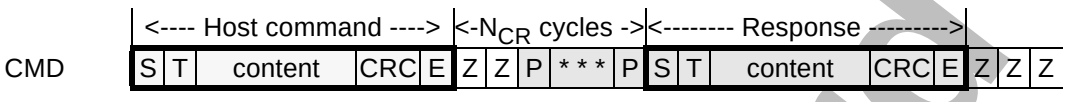


Figure 22: Command response timing (data transfer mode)

- **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.

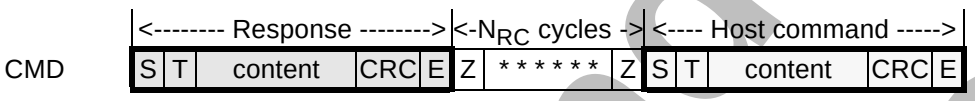


Figure 23: Timing response end to next CMD start (data transfer mode)

- **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.

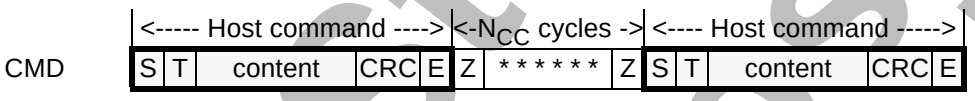


Figure 24: Timing of command sequences (all modes)

4.12.2 Data Read

*** Note: DAT line represents data bus (either 1 or 4 bits).**

- **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 25. 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.

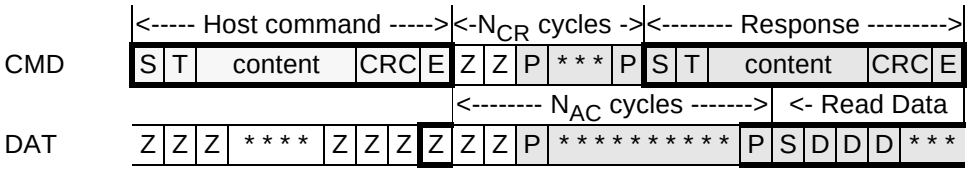


Figure 25: Timing of single block read

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 26 describes the timing of the data blocks and Figure 27 the response to a stop command. The data transmission stops two clock cycles after the end bit of the stop command.

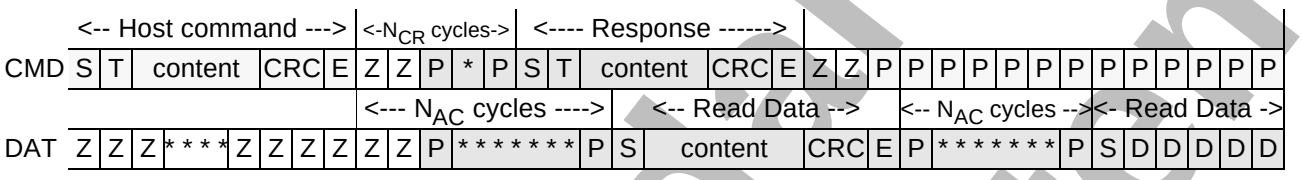


Figure 26: Timing of multiple block read command

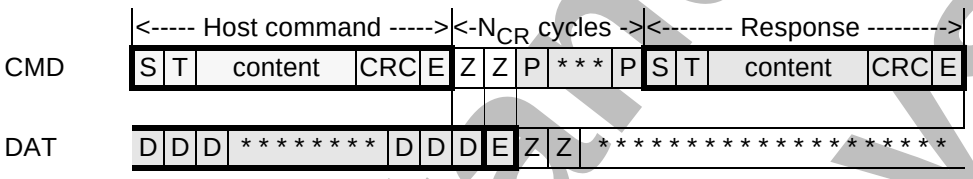


Figure 27: Timing of stop command (CMD12, data transfer mode)

4.12.3 Data Write

- **Single Block Write**

The host selects one card for data write operation by CMD7.

The host sets the valid block length for block oriented data transfer by CMD16.

The basic bus timing for a write operation is given in Figure 28. 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 the DAT0 line. In the case of transmission error the card sends a negative CRC status ('101'). In the case of non erroneous transmission the card sends a positive CRC status ('010') and starts the data programming procedure.

When a flash programming error occurs the card will ignore all further data blocks. In this case no CRC response will be sent to the host and, therefore, there will not be CRC start bit on the bus and the three CRC status bits will read (‘111‘).

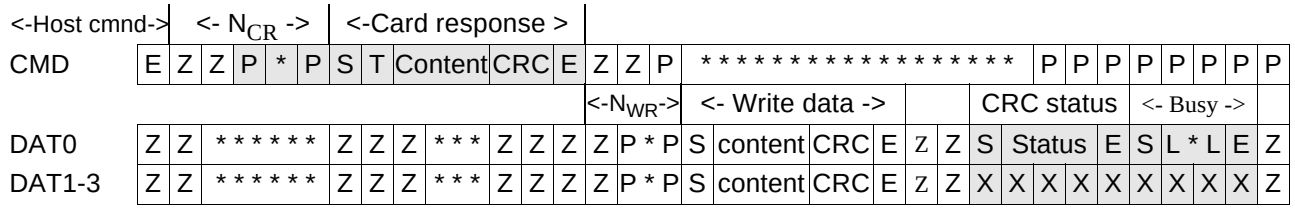


Figure 28: Timing of the block write command

Note that the CRC response output is always two clocks after the end of data.

If the card does not have a free data receive buffer, the card indicates this condition by pulling down the data line to LOW. The card stops pulling down the DAT0 line 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.

As in the case of single block write, 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 the DAT0 line. In the case of transmission error the card sends a negative CRC status (‘101’). In the case of non erroneous transmission the card sends a positive CRC status (‘010’) and starts the data programming procedure. When a flash programming error occurs the card will ignore all further data blocks. In this case no CRC response will be sent to the host and, therefore, there will not be CRC start bit on the bus and the three CRC status bits will read (‘111‘);

The data flow is terminated by a stop transmission command (CMD12). Figure 29 describes the timing of the data blocks with and without card busy signal.

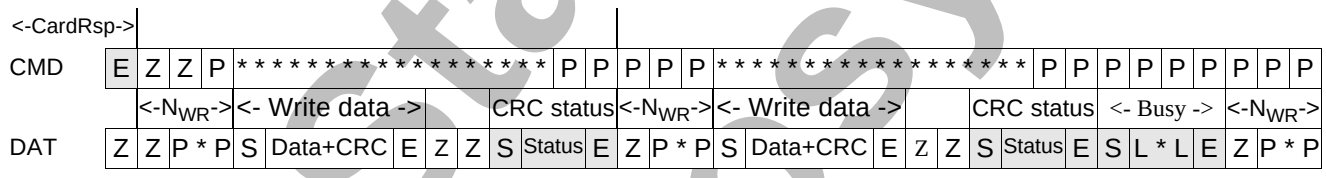


Figure 29: Timing of the multiple block write command

The stop transmission command works similar as in the read mode. Figure 30 to Figure 33 describe

the timing of the stop command in different card states.

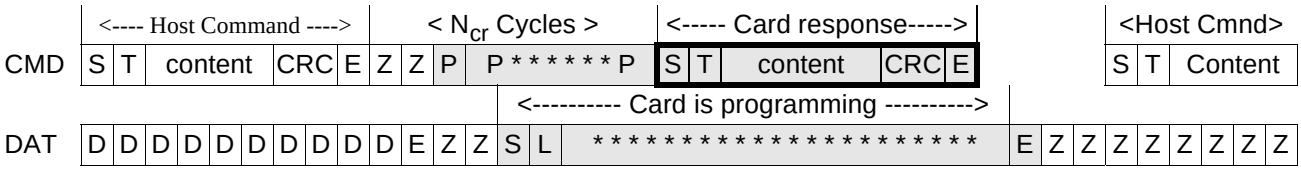
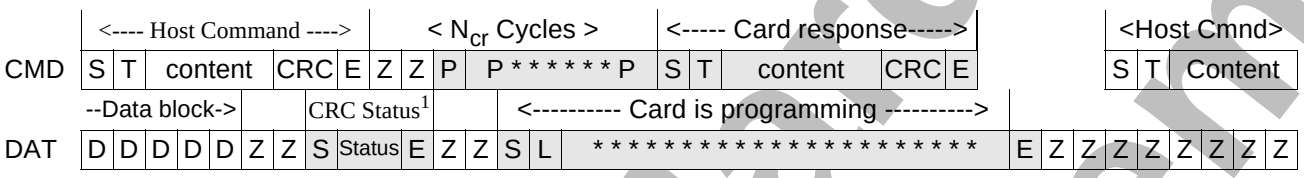


Figure 30: Stop transmission during data transfer from the host

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 token sent back to the host. Figure 31 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 line, with one more data bit, end bit and two Z clock for switching the bus direction. The received data block, in this case is considered incomplete and will not be programmed.



1) The card CRC status response was interrupted by the host.

Figure 31: 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.

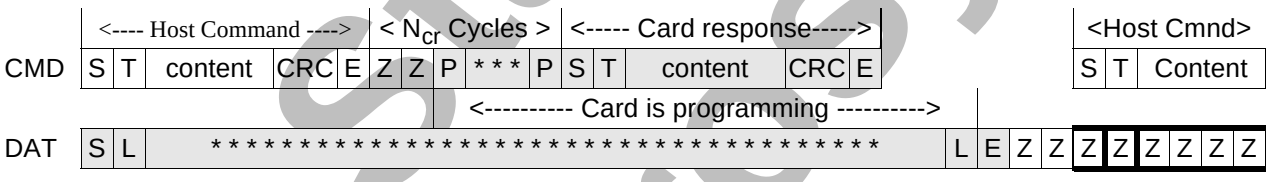


Figure 32: Stop transmission received after last data block. Card is busy programming.

:

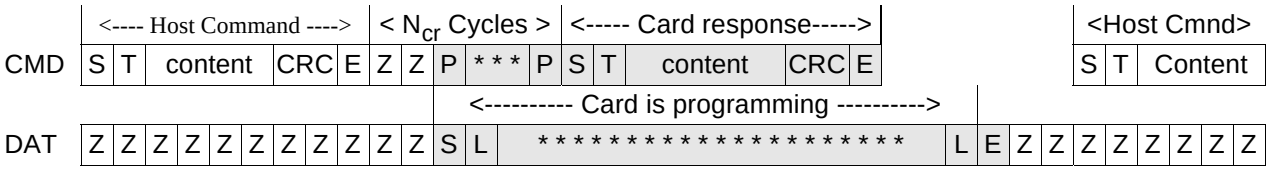


Figure 33: Stop transmission received after last data block. Card becomes busy.

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

The host must first tag the start (CMD32) and end (CMD33) addresses of the range to be erased. The erase command (CMD38), once issued, will erase all the selected write blocks. Similarly, set and clear write protect commands start a programming operation as well. The card will signal “busy” (by pulling the DAT line low) for the duration of the erase or programming operation. The bus transaction timings are the same as given for stop tran command in Figure 33.

- **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. The timing diagram for this command / response / busy transaction is the same as given for stop tran command in Figure 33.

4.12.4 Timing Values

Table 26 defines all timing values.

	Min	Max	Unit
N_{CR}	2	64	clock cycles
N_{ID}	5	5	clock cycles
N_{AC}	2	TAAC + NSAC	clock cycles
N_{RC}	8	-	clock cycles
N_{CC}	8	-	clock cycles
N_{WR}	2	-	clock cycles

Table 26: Timing values

5 Card Registers

Within the card interface six registers are defined: OCR, CID, CSD, RCA, DSR and SCR. These can be accessed only by corresponding commands (see Chapter 4.7). The OCR, CID, CSD and SCR registers carry the card/content specific information, while the RCA and DSR registers are configuration registers storing actual configuration parameters.

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 the cards which do not support the full operating voltage range of the SD Memory Card bus, or if the card power up extends the definition in the timing diagram (Figure 37).

OCR bit position	VDD voltage window
0-3	reserved
4	1.6-1.7
5	1.7-1.8
6	1.8-1.9
7	1.9-2.0
8	2.0-2.1
9	2.1-2.2
10	2.2-2.3
11	2.3-2.4
12	2.4-2.5
13	2.5-2.6
14	2.6-2.7
15	2.7-2.8
16	2.8-2.9
17	2.9-3.0
18	3.0-3.1
19	3.1-3.2
20	3.2-3.3
21	3.3-3.4
22	3.4-3.5
23	3.5-3.6
24-30	reserved
31	card power up status bit (busy)[1]

Table 27: OCR register definition

1)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 27. A voltage range is not supported if the corresponding bit value is set to LOW. As long as the card is busy, the corresponding bit (31) is set

to LOW.

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. Every individual flash card shall have a unique identification number. The structure of the CID register is defined in the following paragraphs:

Name	Field	Width	CID-slice
Manufacturer ID	MID	8	[127:120]
OEM/Application ID	OID	16	[119:104]
Product name	PNM	40	[103:64]
Product revision	PRV	8	[63:56]
Product serial number	PSN	32	[55:24]
reserved	--	4	[23:20]
Manufacturing date	MDT	12	[19:8]
CRC7 checksum	CRC	7	[7:1]
not used, always '1'	-	1	[0:0]

Table 28: The CID fields

- **MID**

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

- **PNM**

The product name is a string, 5 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**

The Serial Number is 32 bits of unsigned binary integer.

- **MDT**

The manufacturing date composed of two hexadecimal digits, one is 8 bit representing the year(y) and the other is four bits representing the month(m).

The “m” field [11:8] is the month code. 1 = January.

The “y” field [19:12] is the year code. 0 = 2000.

As an example, the binary value of the Date field for production date “April 2001” will be:

00000001 0100.

- **CRC**

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

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(1) = writable once, W = multiple writable.

Name	Field	Width	Cell Type	CSD-slice
CSD structure	CSD_STRUCTURE	2	R	[127:126]
reserved	-	6	R	[125:120]
data read access-time-1	TAAC	8	R	[119:112]
data read access-time-2 in CLK cycles (NSAC*100)	NSAC	8	R	[111:104]
max. data transfer rate	TRAN_SPEED	8	R	[103:96]
card command classes	CCC	12	R	[95:84]
max. read data block length	READ_BL_LEN	4	R	[83:80]
partial blocks for read allowed	READ_BL_PARTIAL	1	R	[79:79]
write block misalignment	WRITE_BLK_MISALIGN	1	R	[78:78]
read block misalignment	READ_BLK_MISALIGN	1	R	[77:77]
DSR implemented	DSR_IMP	1	R	[76:76]
reserved	-	2	R	[75:74]
device size	C_SIZE	12	R	[73:62]
max. read current @V_{DD} min	VDD_R_CURR_MIN	3	R	[61:59]
max. read current @V_{DD} max	VDD_R_CURR_MAX	3	R	[58:56]
max. write current @V_{DD} min	VDD_W_CURR_MIN	3	R	[55:53]

Table 29: The CSD fields compatible to CSD Structure Ver 1/MultiMediaCard Specification Ver 2.11

Name	Field	Width	Cell Type	CSD-slice
max. write current @V_{DD} max	VDD_W_CURR_MAX	3	R	[52:50]
device size multiplier	C_SIZE_MULT	3	R	[49:47]
erase single block enable	ERASE_BLK_EN	1	R	[46:46]
erase sector size	SECTOR_SIZE	7	R	[45:39]
write protect group size	WP_GRP_SIZE	7	R	[38:32]
write protect group enable	WP_GRP_ENABLE	1	R	[31:31]
reserved for MultiMediaCard compatibility		2	R	[30:29]
write speed factor	R2W_FACTOR	3	R	[28:26]
max. write data block length	WRITE_BL_LEN	4	R	[25:22]
partial blocks for write allowed	WRITE_BL_PARTIAL	1	R	[21:21]
reserved	-	5	R	[20:16]
File format group	FILE_FORMAT_GRP	1	R/W(1)	[15:15]
copy flag (OTP)	COPY	1	R/W(1)	[14:14]
permanent write protection	PERM_WRITE_PROTECT	1	R/W(1)	[13:13]
temporary write protection	TMP_WRITE_PROTECT	1	R/W	[12:12]
File format	FILE_FORMAT	2	R/W(1)	[11:10]
reserved		2	R/W	[9:8]
CRC	CRC	7	R/W	[7:1]
not used, always'1'	-	1	-	[0:0]

Table 29: The CSD fields compatible to CSD Structure Ver 1/MultiMediaCard Specification Ver 2.11

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**

Version number of the related CSD structure.

CSD_STRUCTURE	CSD structure version	Valid for SD Memory Card Physical Specification Version
0	CSD version No. 1.0	Version 1.0
1-3	reserved	

Table 30: CSD register structure

- **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	time value 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

Table 31: TAAC access time definition

- **NSAC**

Defines the worst 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 the Table is the sum of 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 maximum data transfer rate per one data line - TRAN_SPEED:

TRAN_SPEED bit	code
2:0	transfer rate unit 0=100kbit/s, 1=1Mbit/s, 2=10Mbit/s, 3=100Mbit/s, 4... 7=reserved
6:3	time value 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

Table 32: Maximum data transfer rate definition

- **CCC**

The SD Memory Card 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 8.

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

Table 33: Supported card command classes

- **READ_BL_LEN**

The maximum read data block length is computed as $2^{READ\_BL\_LEN}$. The maximum block length might therefore be in the range 512...2048 bytes (see Chapter 4.11 for details). Note that in SD Memory Card the WRITE_BL_LEN is always equal to READ_BL_LEN

READ_BL_LEN	Block length	Remark
0-8	reserved	
9	2^9 = 512 Bytes	
.....		
11	2^{11} = 2048 Bytes	
12-15	reserved	

Table 34: Data block length

- **READ_BL_PARTIAL (allways = 1 in SD Memory Card)**

Partial Block Read is always allowed in SD Memory Card. It means that smaller blocks can be used as well. The minimum block size will be 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).

DSR_IMP	DSR type
0	no DSR implemented
1	DSR implemented

Table 35: DSR implementation code table

- **C_SIZE**

This parameter is used to compute the user's data card capacity (not include the security protected area). 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 32 MByte card with BLOCK_LEN = 512 can be coded by C_SIZE_MULT = 3 and C_SIZE = 2000.

- **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

Table 36: $V_{DD,min}$ current consumption

- **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

Table 37: $V_{DD,max}$ current consumption

- **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	Remark
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	$2^9 = 512$	

Table 38: Multiply factor for the device size

- **ERASE_BLK_EN**

Defines whether erase of one write block (see WRITE_BL_LEN) is allowed (beside the SECTOR_SIZE given bellow).

If ERASE_BLK_EN = '0' host can erase unit of SECTOR_SIZE.

if ERASE_BLK_EN = '1' host can erase either unit of SECTOR_SIZE or unit of WRITE_BL_LEN.

- **SECTOR_SIZE**

The size of an erasable sector. The contents of this register is a 7 bit binary coded value, defining the number of write blocks (see WRITE_BL_LEN). The actual size is computed by increasing this number by one. A value of zero means 1 write block, 127 means 128 write blocks.

- **WP_GRP_SIZE**

The size of a write protected group. The contents of this register is a 7 bit binary coded value, defining the number of erase sectors (see SECTOR_SIZE). The actual size is computed by increasing this number by one. A value of zero means 1 erase sector, 127 means 128 erase sectors.

- **WP_GRP_ENABLE**

A value of '0' means no group write protection possible.

- **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)

Table 39: R2W_FACTOR

R2W_FACTOR	Multiples of read access time
2	4
3	8
4	16
5	32
6,7	reserved

Table 39: R2W_FACTOR

- **WRITE_BL_LEN**

The maximum write data block length is computed as $2^{WRITE\_BL\_LEN}$. The maximum block length might therefore be in the range from 512 up to 2048 bytes. Write Block Length of 512 bytes is always supported.

Note that in SD Memory Card the WRITE_BL_LEN is always equal to READ_BL_LEN.

WRITE_BL_LEN	Block length	Remark
0-8	reserved	
9	2^9 = 512 Bytes	
.....		
11	2^{11} = 2048 Bytes	
12-15	reserved	

Table 40: Data block length

- **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 and its partial derivatives, in resolution of units of 512 bytes, 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 41 (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.

- **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

Table 41: File formats

A more detailed description is given in SD Memory Card File System specification.

- **CRC**

The CRC field carries the check sum for the CSD contents. It is computed according to Chapter 7.2. 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**	**2**	**4**	**5**	**6**	**7**	**8**	**9**
CSD_STRUCTURE	+	+	+	+	+	+	+	+
TAAC		+	+	+	+	+	+	
NSAC		+	+	+	+	+	+	

Table 42: Cross reference of CSD fields vs. command classes

	Command classes							
CSD Field	**0**	**2**	**4**	**5**	**6**	**7**	**8**	**9**
TRAN_SPEED		+	+					
CCC	+	+	+	+	+	+	+	+
READ_BL_LEN		+						
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_BLK_EN				+	+	+	+	
SECTOR_SIZE				+	+	+	+	
WP_GRP_SIZE					+	+	+	
WP_GRP_ENABLE					+	+	+	
R2W_FACTOR			+	+	+	+	+	
WRITE_BL_LEN			+	+	+	+	+	
WRITE_BL_PARTIAL			+	+	+	+	+	
FILE_FORMAT_GRP								
COPY	+	+	+	+	+	+	+	+
PERM_WRITE_PROTECT	+	+	+	+	+	+	+	+
TMP_WRITE_PROTECT	+	+	+	+	+	+	+	+
FILE_FORMAT								
CRC	+	+	+	+	+	+	+	+

Table 42: Cross reference of CSD fields vs. command classes

5.4 RCA Register

The writable 16-bit relative card address register carries the card address that is published by the card 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 0x0000. The value 0x0000 is reserved to set all cards into the *Stand-by State* with CMD7.

5.5 DSR Register (Optional)

The 16-bit driver stage register is described in detail in Chapter 6.5. It 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.

5.6 SCR Register

In addition to the CSD register there is another configuration register that named - SD CARD Configuration Register (SCR). SCR provides information on SD Memory Card's special features that were configured into the given card. The size of SCR register is 64 bit. This register shall be set in the factory by the SD Memory Card manufacturer.

The following table describes the SCR register content.

Description	Field	Width	Cell Type	SCR Slice
SCR Structure	SCR_STRUCTURE	4	R	[63:60]
SD Memory Card - Spec. Version	SD_SPEC	4	R	[59:56]
data_status_after erases	DATA_STAT_AFTER_ERASE	1	R	[55:55]
SD Security Support	SD_SECURITY	3	R	[54:52]
DAT Bus width supported	SD_BUS_WIDTHS	4	R	[51:48]
reserved	-	16	R	[47:32]
reserved for manufacturer usage	-	32	R	[31:0]

Table 43: The SCR Fields

- **SCR_STRUCTURE**

Version number of the related SCR structure in the SD Memory Card Physical Layer Specification.

CSD_STRUCTURE	CSD structure version	Valid for SD Physical Layer Specification Version
0	SCR version No. 1.0	Version 1.0
1-15	reserved	

Table 44: SCR register structure version

- **SD_SPEC**

Describes the SD Memory Card Physical Layer Specification version supported by this card.

SPEC_VERS	Physical Layer Specification Version Number
0	Version 1.0
1-15	reserved

Table 45: SD Memory Card Physical Layer Specification Version

- **DATA_STAT_AFTER_ERASE**

Defines the data status after erase, whether it is '0' or '1' (the status is card vendor dependent).

- **SD_SECURITY**

Describes the security algorithm supported by the card.

SD_SECURITY	Supported algorithm
0	no security
1	security protocol 1.0
2	security protocol 2.0
3 .. 7	reserved

Table 46: SD Supported security algorithm

- **SD_BUS_WIDTHS**

Describes all the DAT bus widths that are supported by this card.

SD_BUS_WIDTHS	Supported Bus Widths
Bit 0	1 bit (DAT0)
Bit 1	reserved
Bit 2	4 bit (DAT0-3)
Bit 3 [MSB]	reserved

Table 47: SD Memory Card Supported Bus Widths

In case that certain SD Memory Card supports data bus widths of either 1 or 4 bit then Bits 0 and 2 will be set to '1' (SD_BUS_WIDTH="0101").

6 SD Memory Card Hardware Interface

The SD Memory Card has six communication lines and three supply lines:

- CMD: Command is a bidirectional signal. The host and card drivers are operating in push pull mode.
- DAT0-3: 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.

In addition to those lines that are connected to the internal card circuitry there are two contacts of the Write Protect/Card Detect switch that are part of the socket. Those contacts are not mandatory but if they are exist they should be connected as given in the following figure.

When DAT3 is used for card detection, R_{DAT} for DAT3 should be unconnected and an another resistor should be connected to the ground.

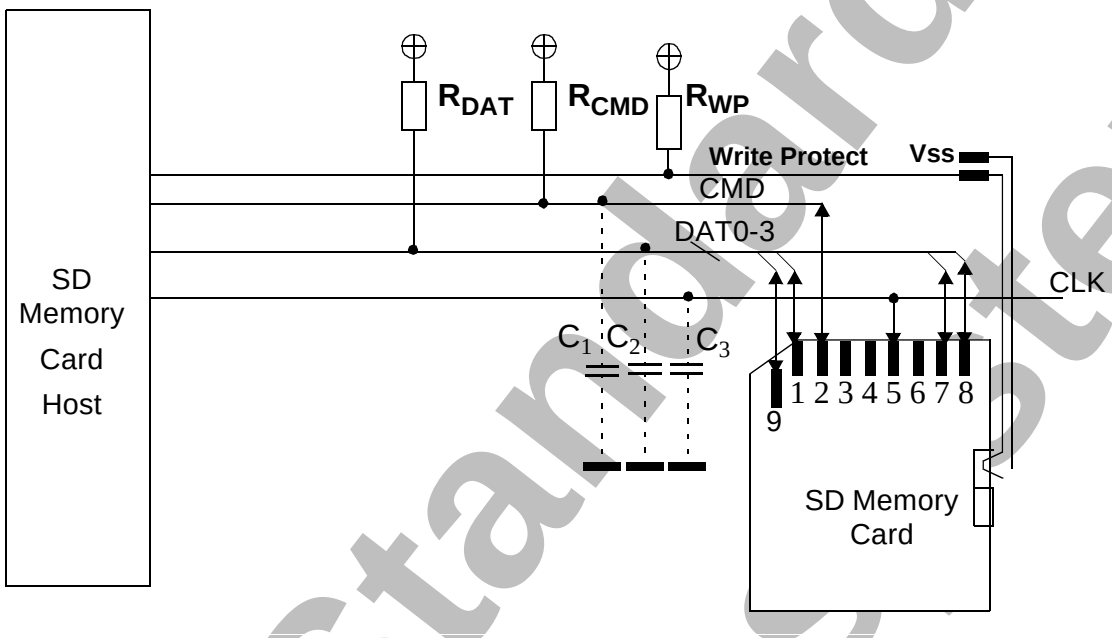


Figure 34: Bus circuitry diagram

R_{DAT} and R_{CMD} are pull-up resistors protecting the CMD and the DAT line against bus floating when no card is inserted or when all card drivers are in an high-impedance mode.

R_{WP} is used for the Write Protect/Card Detection switch.

Refer to Chapter 6.6 for components values and conditions.

6.1 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 8).

No card shall be damaged by inserting or removing a card into the SD Memory Card 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 SD Memory Card 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.

6.2 Card Detection (Insertion/Removal)

SD Memory Card system shall implement detection of card insertion or removal. One method is by sensing pin 1 of the card, and detecting the pull-up resistance on it. Detailed description of this and several other card detection options is given in Application Note “Card Detection Implementations”. In any case since it is not guaranty that all the MultiMediaCards in the market support the proposed card detection methods in SD Memory Card then the SD Memory Card host shall not ignore non-detected cards (empty sockets). If the host gets command from the user to operate a card it shall try to initialize a card in the socket even if no card insertion was previously reported - just in case that MultiMediaCards that do not support the card detection method is inside the socket.

6.3 Power protection (Insertion/Removal)

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.

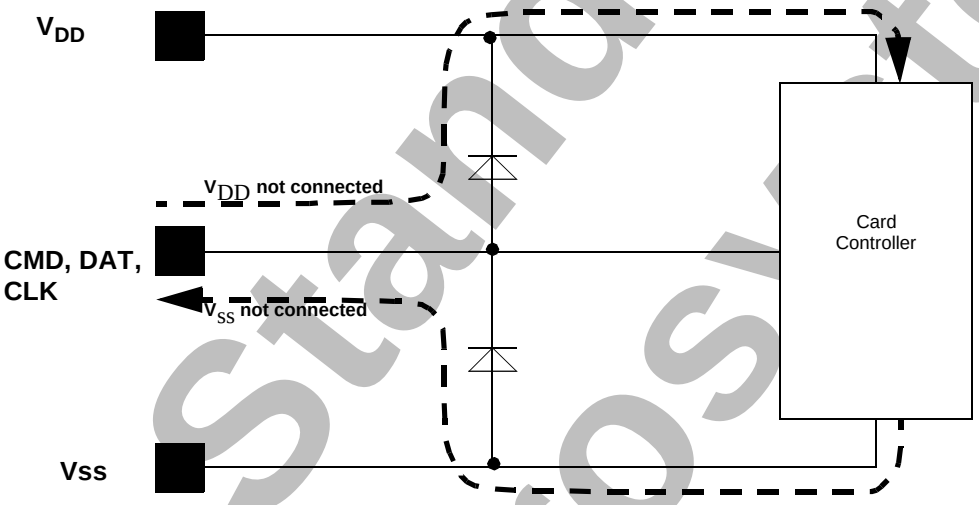


Figure 35: Improper power supply

Every card's output also shall be able to withstand shortcuts to either supply.

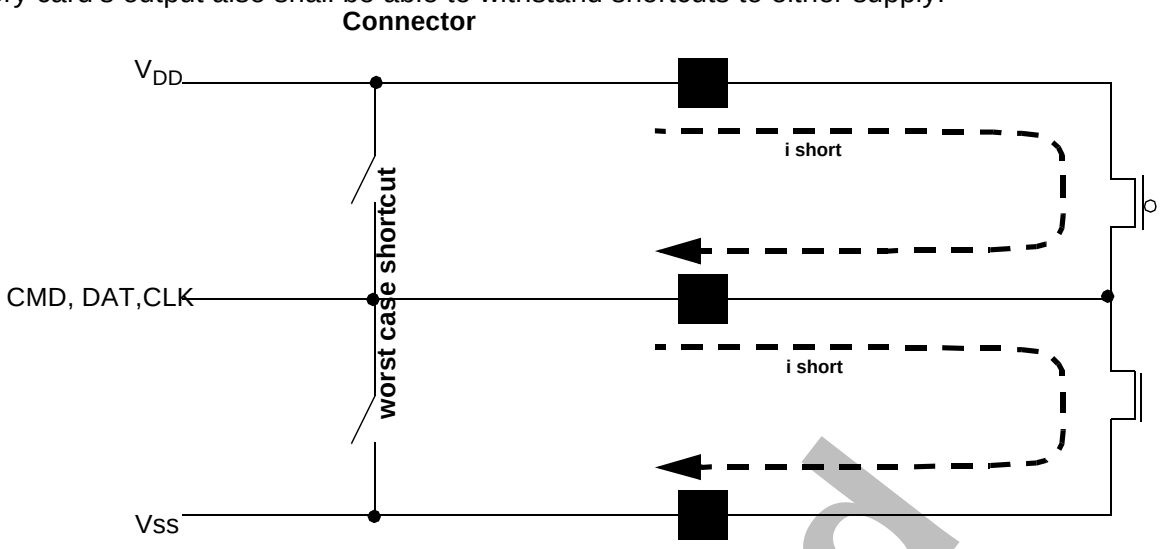


Figure 36: Short cut protection

If hot insertion feature is implemented in the host, than the host has to withstand an instant shortcut between V_{DD} and V_{SS} without damage.

6.4 Power up

The power up of the SD Memory Card bus is handled locally in each SD Memory Card and in the bus master.

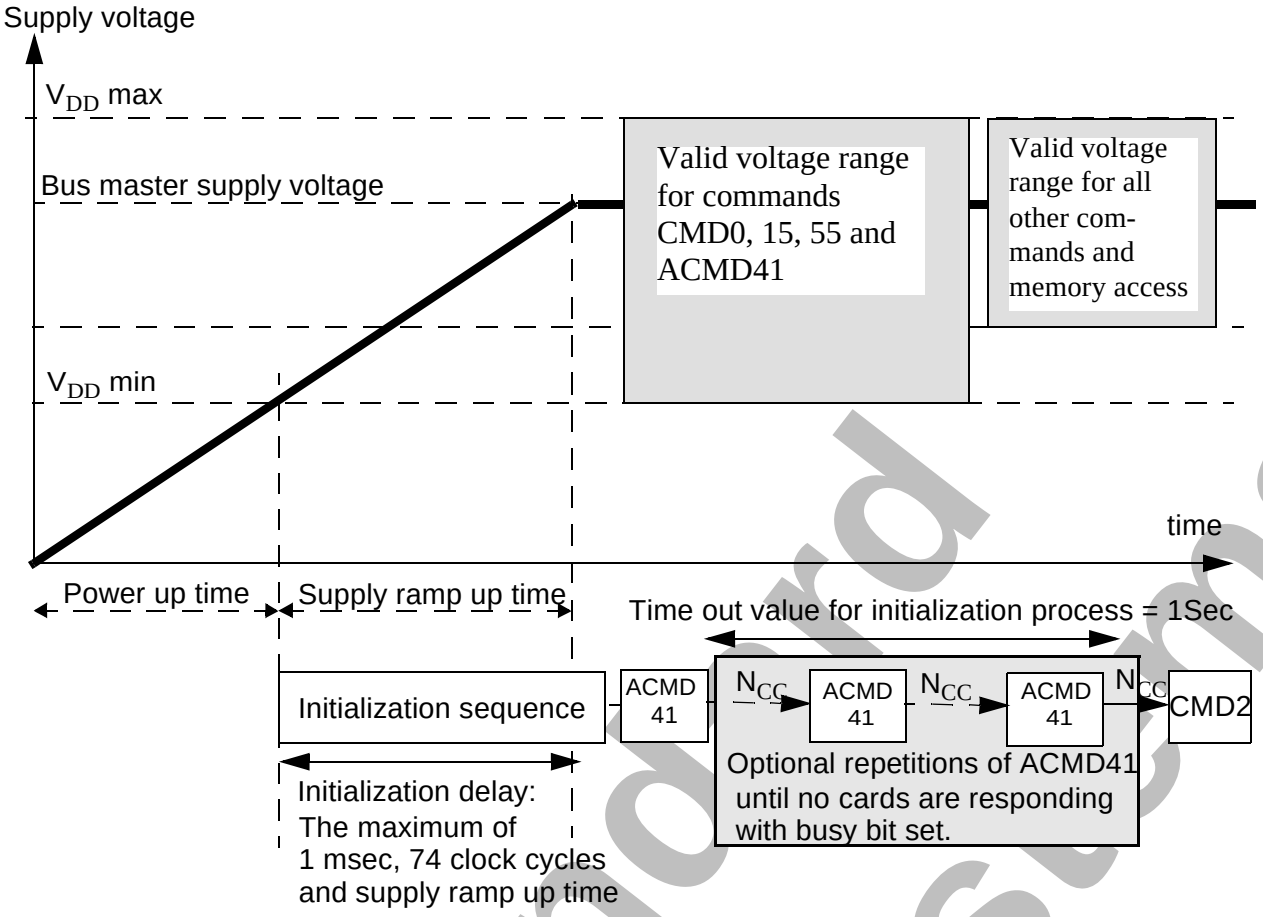


Figure 37: Power-up diagram

- After power up (including hot insertion, i.e. inserting a card when the bus is operating) the SD Memory Card enters the *idle state*. During this state the SD Memory Card ignores all bus transactions until ACMD41 is received (ACMD command type shall always precede with CMD55).
- ACMD41 is a special synchronization command used to negotiate the operation voltage range and to poll the cards until they are out of their power-up sequence. Besides the operation voltage profile of the cards, the response to ACMD41 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 (and continue to poll the cards, each one on his turn) until this bit is cleared. The maximum period of power up procedure of single card shall not exceed 1 second.
- Getting individual cards, as well as the whole SD Memory Card system, 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 such as the maximum number of SD Memory Card s, 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 ACMD41) before ACMD41 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 maximum 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 shall have the capability to implement ACMD41 and CMD1. CMD1 will be used to ask MultiMediaCards to send their Operation Conditions. In any case the ACMD41 or the CMD1 shall be send separately to each card accessing it through its own CMD line.

6.5 Programmable card output driver (Optional)

The bus capacitance of each line of the SD Memory Card bus is the sum of the bus master capacitance, the bus capacitance itself and the capacitance of each inserted card. The sum of host and bus capacitance are fixed for one application, but may vary between different applications.The card load may vary in one application with each of the inserted cards.

In the following, programmable card output drivers for the push pull mode are described as an optional method for ensuring the defined maximum clock rate independently of the topology and of the number of inserted cards.

Both data and command driver stages in the push-pull mode have programmable peak current driving capabilities and programmable rise and fall times. The driver stage register (DSR) consists of two 8-bit latches. The contents of the latches is calculated from the required transfer speed of the interface and the bus load.

The CMD and DAT bus drivers consist of a predriver stage and a complementary driver transistor (Figure 38). The predriver stage output rise and fall time is set with the DSR1 register and determines the speed of the driver stage. The complementary driver transistor size is set with the DSR2 register and determines the current driving capabilities of the driver stage and also influences the peak current consumption of the bus driver. The proper combination of both allows the optimum bus performance.

Table 48 defines the DSR register contents:

DSR1	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
$t_{switch\text{-}on\ max}$	reserved				5ns	20ns	100ns	500ns
$t_{switch\text{-}on\ min}$					2ns	10ns	50ns	200ns

DSR2	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
$i_{peak\ min}$	reserved				100mA	20mA	5mA	1mA
$i_{peak\ max}$					200mA	50mA	10mA	2mA
$t_{rise\ typ}$					5ns	20ns	100ns	500ns

Table 48: DSR register contents

The time in DSR1 specifies the switch-on time of the output driver transistors. At the external interface, it is measurable as a delay time between the clock and driver stage output signal (e.g. for testing).

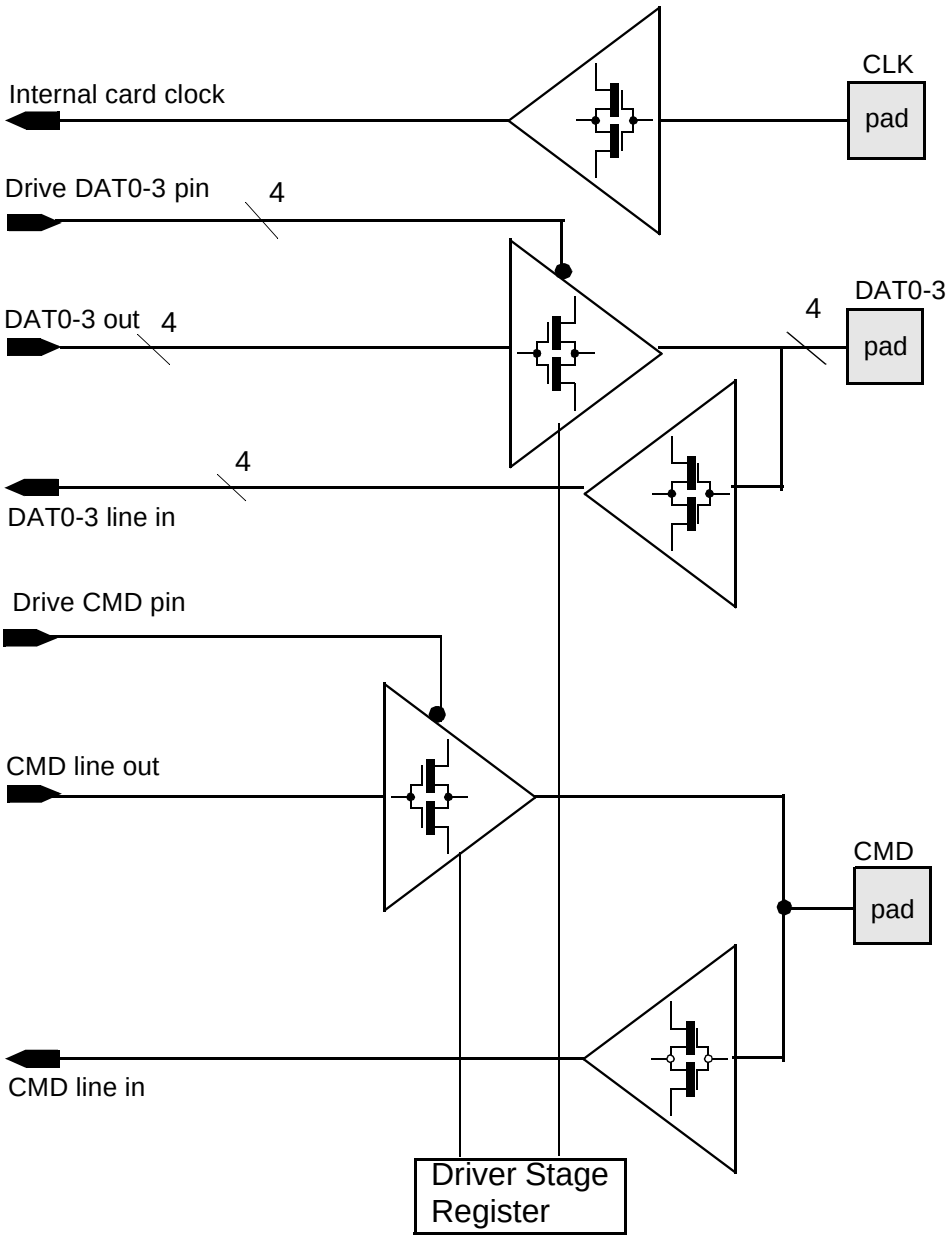


Figure 38: SD Memory Card bus driver

All data is valid for the specified operating range (voltage, temperature). Any combination of DSR1 and DSR2 bits may be programmed. DSR1 has to be programmed for the required clock frequency, where

$f_{clock} = (2\ t_{switch\text{-}on\ max})^{-1}$.

The DSR2 register must be programmed with the required driver size. Hints for the proper driver stage selection are part of future application notes (see Appendix).

6.6 Bus operating conditions

- **General**

Parameter	Symbol	Min	Max.	Unit	Remark
Peak voltage on all lines		-0.3	VDD+0.3	V	
All Inputs					
Input Leakage Current		-10	10	µA	
All Outputs					
Output Leakage Current		-10	10	µA	

Table 49: Bus Operating Cond. - General

- **Power supply voltage**

Parameter	Symbol	Min	Max.	Unit	Remark
Supply voltage	**V_{DD}**	**2.0**	**3.6**	**V**	**CMD0, 15, 55, ACMD41 commands**
Supply voltage specified in OCR register					**Except CMD0, 15, 55, ACMD41 commands**
Supply voltage differentials (V_{SS1}, V_{SS2})		**-0.3**	**0.3**	**V**	
Power up time			**250**	**mS**	**from 0v to V_{DD} Min.**

Table 50: Bus Operating Cond. - Power Supply Voltage

The current consumption of any card during the power-up procedure must not exceed 10 mA.

- **Bus signal line load**

The total capacitance C_L the CLK line of the SD Memory Card bus is the sum of the bus master capacitance C_{HOST}, the bus capacitance C_{BUS} itself and the capacitance C_{CARD} of each card connected to this line:

$$C_L = C_{HOST} + C_{BUS} + N * C_{CARD}$$

where N is the number of connected cards. Requiring the sum of the host and bus capacitances not to exceed 30 pF for up to 10 cards, and 40 pF for up to 30 cards, the following values must not be exceeded:

Parameter	Symbol	Min	Max.	Unit	Remark
Pull-up resistance	R_{CMD} R_{DAT}	10	100	kΩ	to prevent bus floating
Bus signal line capacitance	C_L		250	pF	$f_{PP} \le 5$ MHz, 21 cards

Parameter	Symbol	Min	Max.	Unit	Remark
Bus signal line capacitance	C_L		100	pF	$f_{PP} \leq 20$ MHz, 7 cards
Single card capacitance	C_{CARD}		10	pF	
Maximum signal line inductance			16	nH	$f_{PP} \leq 20$ MHz
Pull-up resistance inside card (pin1)	R_{DAT3}	10	90	kΩ	May be used for card detection

Table 51: Bus Operating Cond. - Signal Line's Load

Note that the total capacitance of CMD and DAT lines will be consist of C_{HOST}, C_{BUS} and one C_{CARD} only since they are connected separately to the SD Memory Card host.

6.7 Bus signal levels

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

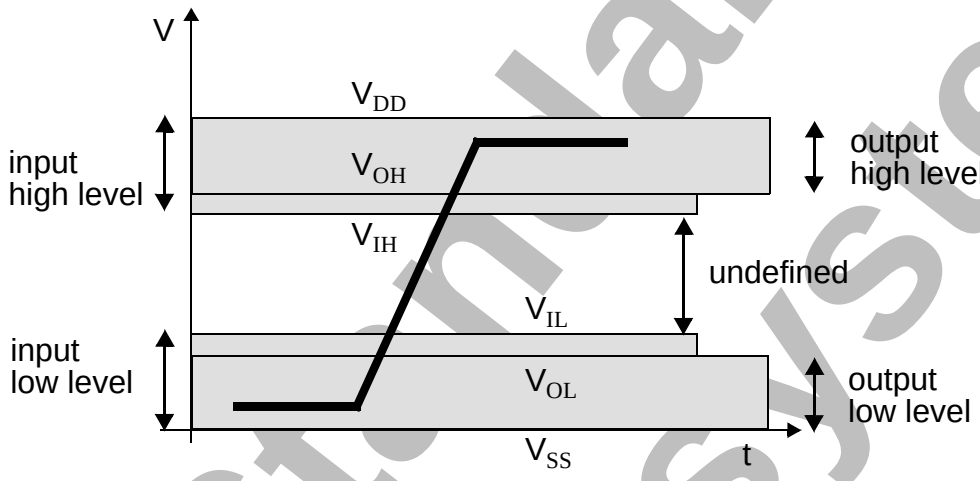


Figure 39: Bus signal levels

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 52: Bus Signal Cond. - I/O Signal Voltages

6.8 Bus timing

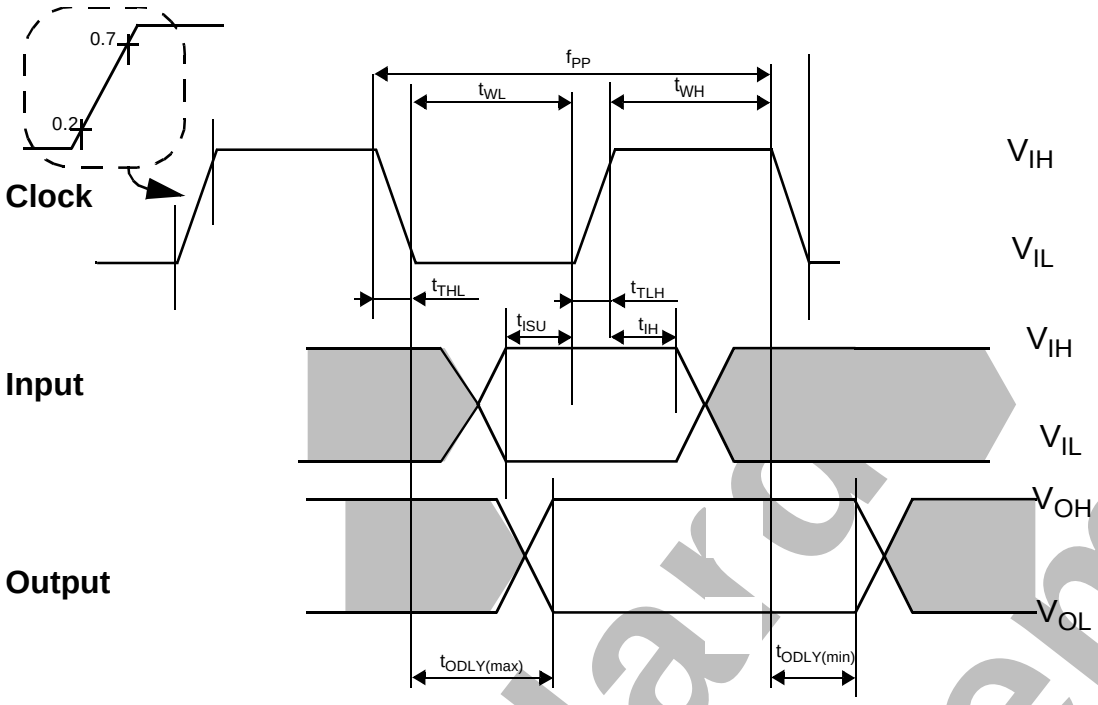


Figure 40: Timing diagram data input/output referenced to clock

Parameter	Symbol	Min	Max.	Unit	Remark
Clock CLK (All values are referred to min (V_{IH}) and max (V_{IL}),					
Clock frequency Data Transfer Mode	f_{PP}	0	25	MHz	$C_L \leq 100$ pF (7 cards)
Clock frequency Identification Mode (the low freq. is required for MultiMediaCard compatibility).	f_{OD}	0	400	kHz	$C_L \leq 250$ pF (21 cards)
Clock low time	t_{WL}	10		ns	$C_L \leq 100$ pF (7 cards)
Clock high time	t_{WH}	10		ns	$C_L \leq 100$ pF (7 cards)
Clock rise time	t_{TLH}		10	ns	$C_L \leq 100$ pF (7 cards)
Clock fall time	t_{THL}		10	ns	$C_L \leq 100$ pF (7 cards)
Clock low time	t_{WL}	50		ns	$C_L \leq 250$ pF (21 cards)

Parameter	Symbol	Min	Max.	Unit	Remark
Clock high time	t_{WH}	**50**		**ns**	$C_L \leq$ **250 pF (21 cards)**
Clock rise time	t_{TLH}		**50**	**ns**	$C_L \leq$ **250 pF (21 cards)**
Clock fall time	t_{THL}		**50**	**ns**	$C_L \leq$ **250 pF (21 cards)**
Inputs CMD, DAT (referenced to CLK)					
Input set-up time	t_{ISU}	**5**		**ns**	$C_L \leq$ **25 pF (1 cards)**
Input hold time	t_{IH}	**5**		**ns**	$C_L \leq$ **25 pF (1 cards)**
Outputs CMD, DAT (referenced to CLK)					
Output Delay time	t_{ODLY}	**0**	**14**	**ns**	$C_L \leq$ **25 pF (1 cards)**

Table 53: Bus Timing - Parameters Values

6.9 SDLV Memory Card - Low Voltage SD Memory Cards

Next generation of SD Memory Cards will support operation at lower supply voltage.

Parameter	Symbol	Min	Max.	Unit	Remark
SDLV Memory Card Supply voltage	V_{DD}	**1.6**	**3.6**	**V**	**for Identification and regular operation**
Supply voltage differentials (V_{SS1}, V_{SS2})		**-0.2**	**0.2**	**V**	
Power up time			**250**	**mS**	**from 0v to V_{DD} Min.**

Table 54: SDLV Memory Card - Power Supply Voltage

The SDLV Memory Cards (low voltage cards) will support identification and operation voltage between 1.6V up to 3.6v. That means that the host may operate between 1.6 and 3.6 volts. Though it is strongly recommended that the host will be developed in such a way that the initialization will be done in the range of 2.0-3.6 volt. That will allow to communicate with regular SD Memory Cards at least for the initialization process and then to put them in inactive state in case that the host would prefer to work in a lower voltage range of SDLV Memory Card rather than SD Memory Card.

Although not mechanically different, new and old cards will not necessarily be interchangeable in a given system. The system definition allows an attempt to use old (high voltage) cards with new (low

voltage) hosts. In case that the host supports initialization in the range of 2-3.6v the user will be informed that this is a wrong type of card.

In order to minimize customer dissatisfaction the SDLV Memory Cards will be labeled in such a way that the user will be able to distinguish between SDLV Memory Cards and regular SD Memory Cards (2.0-3.6v operation).

The marking method is to be defined **[T.B.D]**.

7 SPI Mode

7.1 Introduction

The SPI mode consists of a secondary communication protocol which is offered by Flash-based SD Memory Cards. This mode is a subset of the SD Memory Card 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 SD Memory Card SPI implementation uses a subset of the SD Memory Card protocol and command set. 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 SD mode (e.g. Single data line and hardware CS signal per card).

7.2 SPI Bus Protocol

While the SD Memory Card 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 SD Memory Card protocol, the SPI messages consist of command, response and data-block tokens. All communication between host and cards 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 SD mode in the following three aspects:

- The selected card always responds to the command.
- An additional (8 bit) response structure is 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 SD mode.

In addition to the command response, every data block sent to the card during write operations will be responded with a special data response token. A data block may be as big as one card write block (WRITE_BL_LEN) and as small as a single byte. Partial block read/write operations are enabled by card options specified in the CSD register.

7.2.1 Mode Selection

The SD Memory Card wakes up in the SD mode. It will enter SPI mode if the CS signal is asserted (negative) during the reception of the reset command (CMD0) and the card is in *idle_state*. If the card recognizes that the SD mode is required it will not respond to the command and remain in the SD mode. If SPI mode is required the card will switch to SPI and respond with the SPI mode R1 response.

The only way to return to the SD mode is by entering the power cycle. In SPI mode the SD Memory Card protocol state machine is not observed. All the SD Memory Card commands supported in SPI mode are always available.

7.2.2 Bus Transfer Protection

Every SD Memory Card token transferred on the bus is protected by CRC bits. In SPI mode, the SD Memory Card 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 SD 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).

7.2.3 Data Read

The SPI mode supports single block read and Multiple Block read operations (CMD17 or CMD18 in the SD Memory Card protocol). Upon reception of a valid read command the card will respond with a response token followed by a data token of the length defined in a previous SET_BLOCKLEN (CMD16) command (refer to Figure 41).

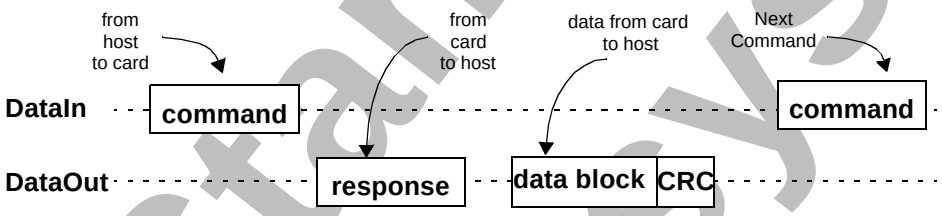


Figure 41: Single Block Read operation

A valid data block is suffixed with a 16 bit CRC generated by the standard CCITT polynomial $x^{16}+x^{12}+x^5+1$.

The maximum block length is given by READ_BL_LEN, defined in the CSD. If partial blocks are allowed (i.e. the CSD parameter READ_BL_PARTIAL equals 1), the block length can be any number between 1 and the maximum block size. Otherwise, the only valid block length for data read is given by READ_BL_LEN.

The start address can be any byte address in the valid address range of the card. Every block, however, must be contained in a single physical card sector.

In case of a data retrieval error, the card will not transmit any data. Instead, a special data error token will be sent to the host. Figure 42 shows a data read operation which terminated with an error

token rather than a data block.

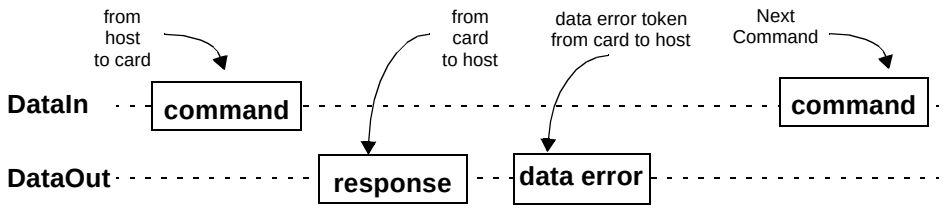


Figure 42: Read operation - data error

In case of Multiple block read operation every transferred block has its suffixed of 16 bit CRC.

Stop transmission command (CMD12) will actually stop the data transfer operation (the same as in SD Memory Card operation mode).

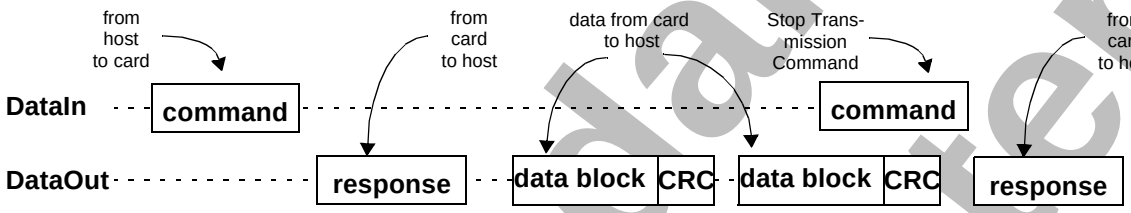


Figure 43: Multiple Block Read operation

7.2.4 Data Write

In SPI mode the SD Memory Card supports single block and Multiple block write commands. Upon reception of a valid write command (CMD24 or CMD25 in the SD Memory Card protocol), 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 44).

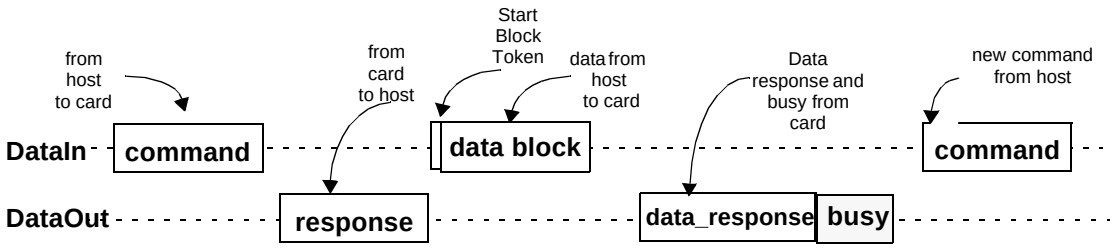


Figure 44: Single Block Write operation

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).

Once the programming operation is completed, the host must check the results of the programming using the SEND_STATUS command (CMD13). Some errors (e.g. address out of range, write protect violation etc.) are detected during programming only. The only validation check performed on the data block and communicated to the host via the data-response token is the CRC and general Write Error indication.

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. In case of Write Error indication (on the data response) the host shall use SEND_NUM_WR_BLOCKS (ACMD22) in order to get the number of well written write blocks. The data tokens description is given in Chapter 7.3.3.

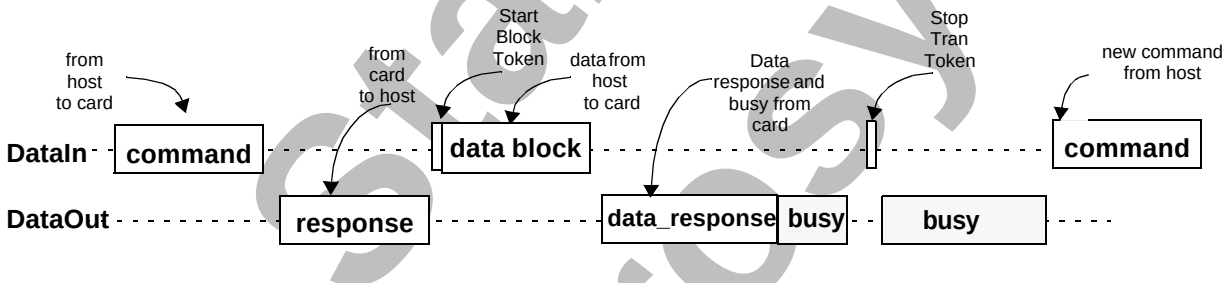


Figure 45: Multiple Block Write operation

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 operation. This may destroy the data formats on the card. It is in the responsibility of the host to prevent it.

7.2.5 Erase & Write Protect Management

The erase and write protect management procedures in the SPI mode are identical to those of the SD mode. While the card is erasing or changing the write protection bits of the predefined sector list, it will be in a busy state and hold the DataOut line low. Figure 46 illustrates a 'no data' bus transaction with and without busy signalling.

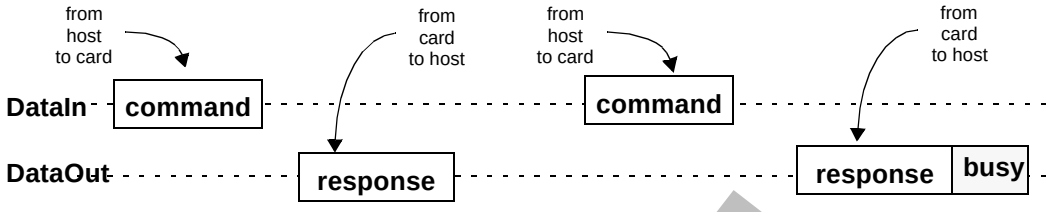


Figure 46: 'No data' operations

7.2.6 Read CID/CSD Registers

Unlike the SD Memory Card 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 42) 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 cards TAAC since this value is stored in the card's CSD. Therefore the standard response time-out value (N_{CR}) is used for read latency of the CSD register.

7.2.7 Reset Sequence

The SD Memory Card 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), ACMD41 (SD_SEND_OP_COND) and CMD58 (READ_OCR).

In SPI mode CMD1 and ACMD41 have the same behavior.

The host must poll the card (by repeatedly sending CMD1 or ACMD41) until the 'in-idle-state' bit in the card response indicates (by being set to 0) that the card completed its initialization processes and is ready for the next command.

In SPI mode, as opposed to SD mode, CMD1 (and ACMD41 as well) 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 cards that do not support its voltage range.

The usage of CMD58 is not restricted to the initializing phase only, but can be issued at any time.

7.2.8 Error Conditions

Unlike the SD Memory Card protocol, in the SPI mode the card will always respond to a command.

The response indicates acceptance or rejection of the command. A command may be rejected if it is not supported (illegal opcode), if the CRC check failed, if it contained an illegal operand, or if it was out of sequence during an erase sequence.

7.2.9 Memory Array Partitioning

Same as for SD mode.

7.2.10 Card Lock/unlock

Usage of card lock and unlock commands in SPI mode is identical to SD mode. In both cases the command is responded with a R1b response type. After the busy signal clears, the host should obtain the result of the operation by issuing a GET_STATUS command. Refer to Chapter 4.3.6 for details.

7.2.11 Application Specific commands

Identical to SD mode with the exception of the APP_CMD status bit (refer to Chapter 4.10.1) which is not available in SPI.

7.2.12 Copyright Protection commands

All the special Copyright Protection ACMDs and security functionality is the same as for SD mode.

7.3 SPI Mode Transaction Packets

7.3.1 Command Tokens

- **Command Format**

All the SD Memory Card commands are 6 bytes long. The command transmission always starts with the left bit of the bitstring corresponding to the command codeword. All commands are protected by a CRC (see Chapter 7.2). The commands and arguments are listed in Table 57.

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 55: Command Format

- **Command Classes**

As in SD mode, the SPI commands are divided into several classes (See Table 56). Each class supports a set of card functions. A SD Memory Card 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 command classes, and the supported command for a specific class, however, are different in the SD Memory Card and the SPI communication mode.

Note that except the classes that are not supported in SPI mode (class 1, 3 and 9), the mandatory required classes for the SD mode are the same for the SPI mode.

Card CMD Class (CCC)	Class Description	Supported commands																					
		0	1	9	10	12	13	16	17	18	24	25	27	28	29	30	32	33	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 (Optional)													+	+	+								
class 7	Lock Card (Optional)																			+				
class 8	Application specific																				+	+		
class 9	Not supported in SPI																							
class 10-11	Reserved																							

Table 56: Command classes in SPI mode

• Detailed Command Description

The following table provides a detailed description of the SPI bus commands. The responses are defined in Chapter 7.3.2. The Table 57 lists all SD Memory Card commands. A “yes” in the SPI mode colon 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 reserved in SD mode as well.

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 SD Memory Card
CMD1	Yes	None	R1	SEND_OP_ COND	Activates the card’s initialization process.
CMD2	No				
CMD3	No				
CMD4	No				
CMD5	reserved				
CMD6	reserved				
CMD7	No				

Table 57: Commands and arguments

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD8	reserved				
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	Non	R1b	STOP_ TRANSMISSION	forces the card to stop transmission in Multiple Block Read Operation
CMD13	Yes	None	R2	SEND_STATUS	asks the selected card to send its status register.
CMD14	reserved				
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_MULTIPL E_BLOCK	continuously transfers data blocks from card to host until interrupted by a STOP_TRANSMISSION command.
CMD19	reserved				
CMD20	No				
CMD21 ... CMD23	reserved				
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_MULTIPL E_BLOCK	continuously writes blocks of data until stop Tran' token is sent (instead'Start Block').
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.

Table 57: Commands and arguments

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
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	Yes	[31:0] data address	R1	ERASE_WR_BLK _START_ADDR	sets the address of the first write block to be erased.
CMD33	Yes	[31:0] data address	R1	ERASE_WR_BLK _END_ADDR	sets the address of the last write block of the continuous range to be erased.
CMD34 ... CMD37	reserved				
CMD38	Yes	[31:0] stuff bits	R1b	ERASE	erases all previously selected write blocks
CMD39	No				
CMD40	No				
CMD41	reserved				
CMD42	Yes	[31:0] stuff bits.	R1	LOCK_UNLOCK	Used to Set/Reset the Password or lock/unlock the card. A transferred data block includes all the command details - refer to Chapter 4.3.6. The size of the Data Block is defined with SET_BLOCK_LEN command.
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]	R1	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 specific commands. The size of the Data Block shall be defined with 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 -63	Reserved For Manufacturer				

Table 57: Commands and arguments

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 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 addresses of the last groups are outside the valid range, then the corresponding write protection bits shall be set to zero.
6) **RD/WR**_: "1" the Host shall get a block of data from the card.
"0" the host sends block of data to the card.

The following table describes all the application specific commands supported/reserved by the SD Memory Card. All the following commands shall be preceded with APP_CMD (CMD55).

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
ACMD6	No				
ACMD13	yes	[31:0] stuff bits	R2	SD_STATUS	Send the SD Memory Card status. The status fields are given in Table 24
ACMD17	reserved				
ACMD18	yes	--	--	--	Reserved for SD security applications[1]
ACMD19 to ACMD21	reserved				
ACMD22	yes	[31:0] stuff bits	R1	SEND_NUM_WR_ BLOCKS	Send the numbers of the well written (without errors) blocks. Responds with 32bit+CRC data block.
ACMD23	yes	[31:23] stuff bits [22:0]Number of blocks	R1	SET_WR_BLK_ ERASE_COUNT	Set the number of write blocks to be pre-erased before writing (to be used for faster Multiple Block WR command). "1"=default (one wr block)(2).
ACMD24	reserved				
ACMD25	yes	--	--	--	Reserved for SD security applications[1]
ACMD26	yes	--	--	--	Reserved for SD security applications[1]
ACMD38	yes	--	--	--	Reserved for SD security applications[1]
ACMD39 to ACMD40	reserved				

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
ACMD41	yes	None	R1	SEND_OP_ COND	Activates the card's initialization process.
ACMD42	yes	[31:1] stuff bits [0]set_cd	R1	SET_CLR_CARD_ DETECT	Connect[1]/Disconnect[0] the 50KOhm pull-up resistor on CD/DAT3 (pin 1) of the card. The pull-up may be used for card detection.
ACMD43 ... ACMD49	yes	--	--	--	Reserved for SD security applications[1]
ACMD51	yes	[31:0] staff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

(1) Refer to "SD Memory Card Security Specification" for detailed explanation about the SD Security Features
(2) Command STOP_TRAN (CMD12) shall be used to stop the transmission in Write Multiple Block whether the pre-erase (ACMD23) feature is used or not.

Table 58: Application Specific Commands used/reserved by SD Memory Card - SPI Mode

7.3.2 Responses

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

- **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 47. The meaning of the flags is defined as following:

- **In idle state**: The card is in idle state and running the initializing process.
- **Erase reset:** An erase sequence was cleared before executing because an out of erase sequence command was received.
- **Illegal command**: An illegal command code was detected.
- **Communication CRC error**: The CRC check of the last command failed.
- **Erase sequence error**: An error in the sequence of erase commands occurred.
- **Address error**:A misaligned address, which did not match the block length, was used in the command.
- **Parameter error**: The command's argument (e.g. address, block length) was out of the allowed range for this card.

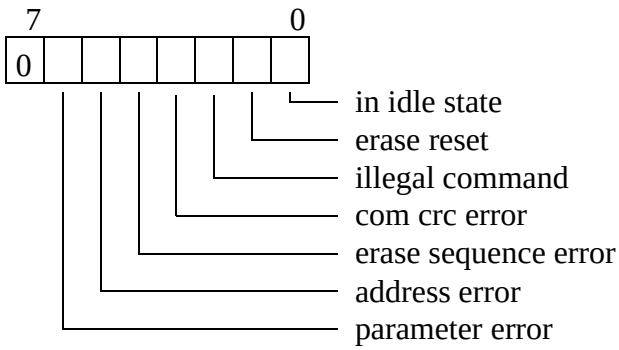


Figure 47: R1 Response Format

- **Format R1b**

This response token is identical to the R1 format with the optional addition of the busy signal. 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.

- **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 48.

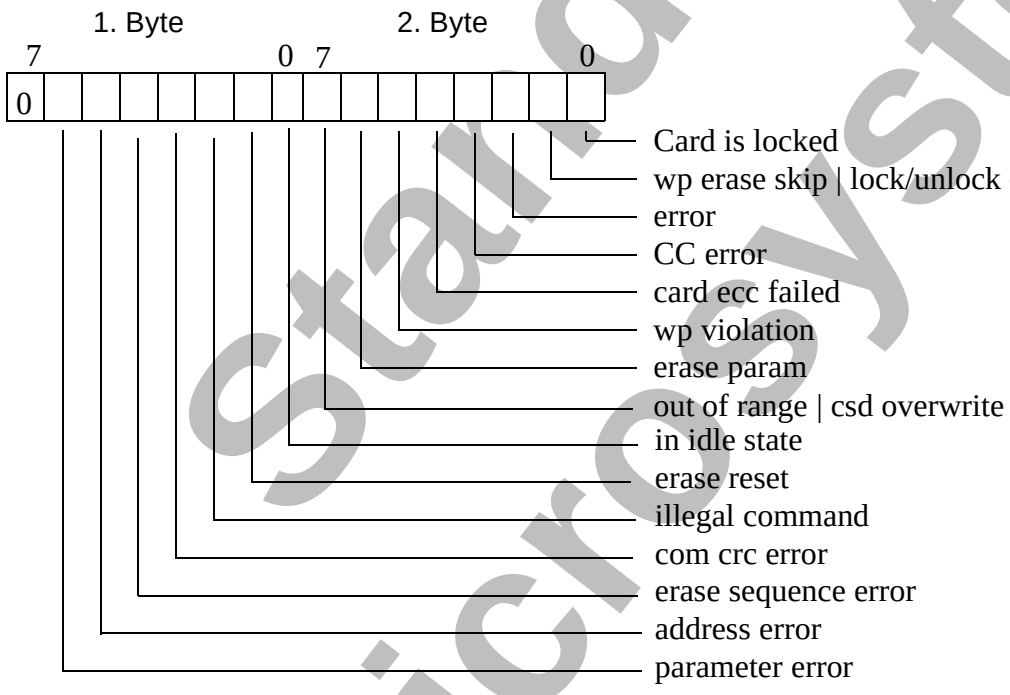


Figure 48: R2 response format

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

- **Erase param**: An invalid selection, sectors or 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.
- **CC error**: Internal card controller error
- **Error**: A general or an unknown error occurred during the operation.
- **Write protect erase skip | lock/unlock command failed**: This status bit has two functions over-loaded. It is set when the host attempts to erase a write protected sector or makes a sequence or password error during card lock/unlock operation.
- **Card is locked**: Set when the card is locked by the user. Reset when it is unlocked.

- **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 49). The structure of the first (MSB) byte is identical to response type R1. The other four bytes contain the OCR register.

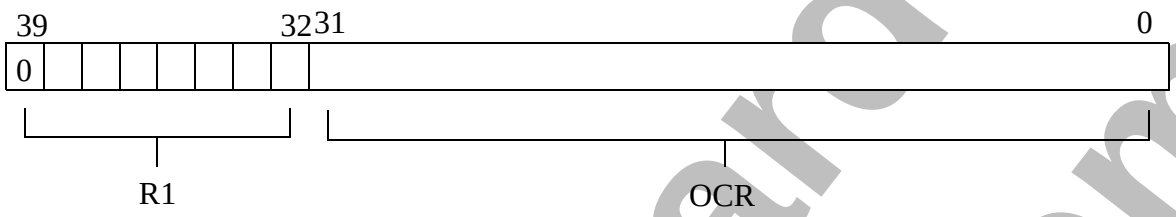
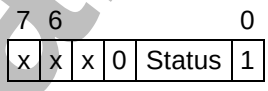


Figure 49: R3 Response Format

- **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:



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

If the host gets status field'110' it shall stop the data transmission using CMD12 and send CMD13 in order to verify which write problem caused the Write Error indication. ACMD22 can be used to find the number of well written write blocks.

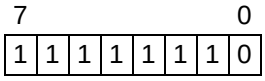
7.3.3 Data Tokens

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 515 bytes long and have the following format:

For Single Block Read, Single Block Write and Multiple Block Read:

- First byte: Start Block

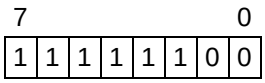


- Bytes 2-513 (depends on the data block length): User data
- Last two bytes: 16 bit CRC.

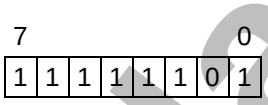
For Multiple Block Write operation:

- First byte of each block:

  If data is to be transferred then - Start Block



  If Stop transmission is requested - Stop Tran



Note that this format is used only for Multiple Block Write. In case of Multiple Block Read the stop transmission is done using STOP_TRAN Command (CMD12).

7.3.4 Data Error Token

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:

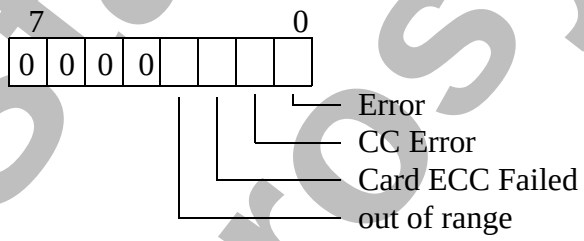


Figure 50: Data Error Token

The 4 least significant bits (LSB) are the same error bits as in the response format R2.

7.3.5 Clearing Status Bits

As described in the previous paragraphs, in SPI mode, 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 multi-

ple response types - e.g Card ECC failed)

As in the SD mode, error bits are cleared when read by the host, regardless of the response format. State indicators are either cleared by reading or in accordance with the card state.

The following table summarizes the set and clear conditions for the various status bits:

.

Identifier	Included in resp	Type[1]	Value	Description	Clear Condition[2]
Out of range	R2 DataErr	E R X	'0'= no error '1'= error	The command argument was out of the allowed range for this card.	C
Address error	R1 R2	E R X	'0'= no error '1'= error	A misaligned address which did not match the block length was used in the command.	C
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 error in the parameters of the erase command sequence	C
Parameter error	R1 R2	E R X	'0'= no error '1'= error	An error in the parameters of the command	C
WP violation	R2	E R 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 previous command failed.	C
Illegal command	R1 R2	E R	'0'= no error '1'= error	Command not legal for the card state	C
Card ECC failed	R2 DataEr	E X	'0'= success '1'= failure	Card internal ECC was applied but failed to correct the data.	C
CC error	R2 dataEr	E R X	'0'= no error '1'= error	Internal card controller error	C
Error	R2 dataEr	E R X	'0'= no error '1'= error	A general or an unknown error occurred during the operation.	C
CID/ CSD_OVERWRITE	R2	E R 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
WP erase skip	R2	S X	'0'= not protected '1'= protected	Only partial address space was erased due to existing write protected blocks.	C

Table 59: SPI mode status bits

Identifier	Included in resp	Type[1]	Value	Description	Clear Condition[2]
Lock/Unlock cmd failed	R2	X	'0'= no error '1'= error	Sequence or password error during card lock/unlock operation	C
Card is locked	R2	S X	'0' = card is not locked '1' = card is locked	Card is locked by a user password and	A
Erase reset	R1 R2	S R	'0'= cleared '1'= set	An erase sequence was cleared before executing because an out of erase sequence command was received	C
In Idle state	R1 R2	S R	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

Table 59: SPI mode status bits

1) Type:

E: Error bit.

S: State bit.

R: Detected and set for the actual command response.

X: Detected and set during command execution. The host must poll the card by issuing the status command in order to read these bits.

2) Clear Condition:

A: According to the card current state.

C: Clear by read

7.4 Card Registers

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

7.5 SPI Bus Timing Diagrams

All timing diagrams use the following schematics and abbreviations:

H	Signal is high (logical '1'
L	Signal is low (logical '0')
X	Don't care
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 Table 60. 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.

7.5.1 Command / Response

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

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

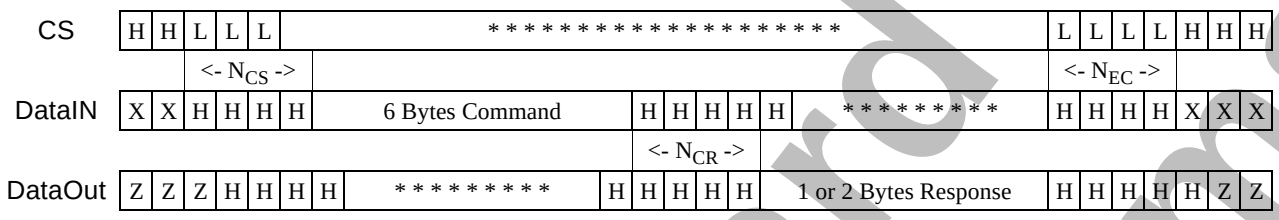


Figure 51: Basic command response

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

The following timing diagram describes the command response transaction for commands when the card responses which the R1b response type (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.

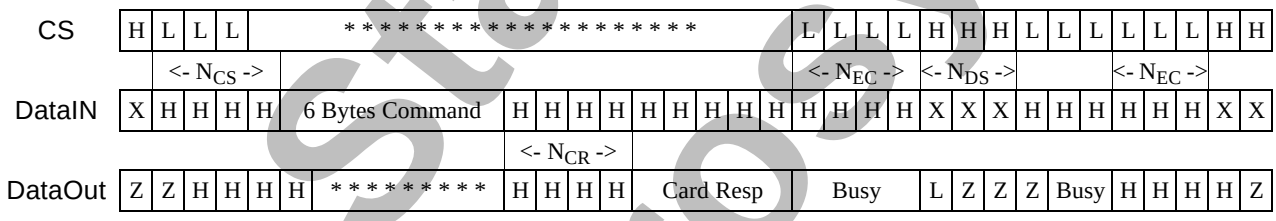


Figure 52: Command response with busy indication (R1b)

- **Card Response to Host Command**

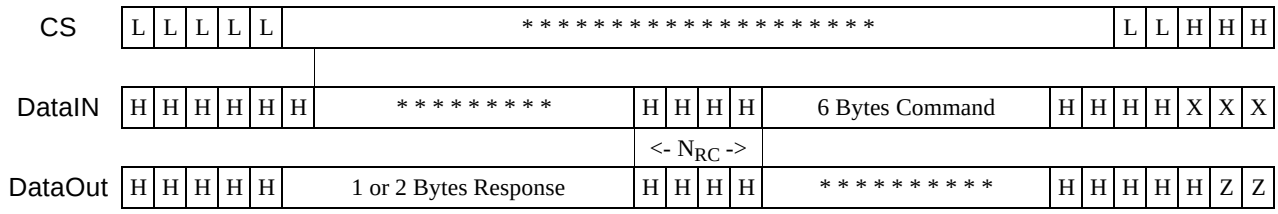


Figure 53: Timing between card response to new host command

7.5.2 Data read

- The following timing diagram describes all single block read operations with the exception of SEND_CSD command.

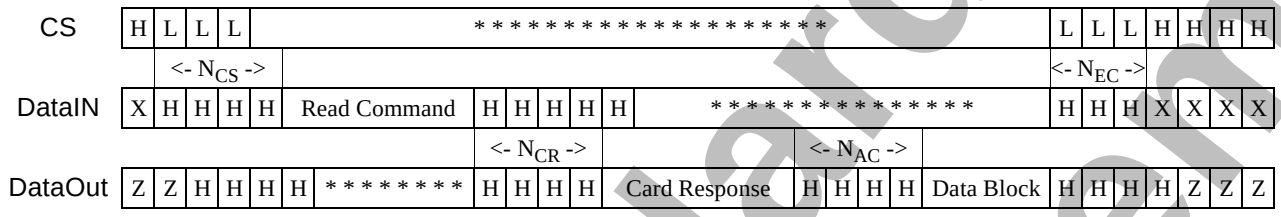


Figure 54: Read Single Block operations - bus timing

The following table describes Stop transmission operation in case of Multiple Block Read.

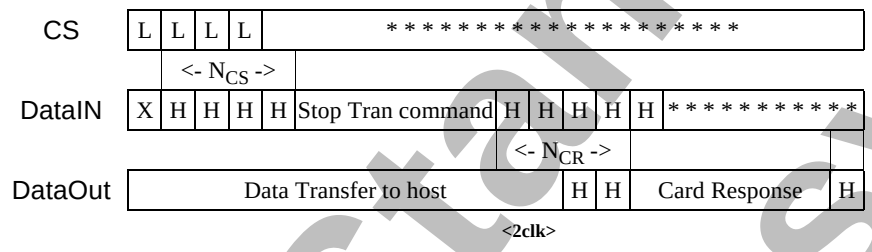


Figure 55: Stop Transmission in Read Multiple Block

- **Reading the CSD register**

The following timing diagram describes the SEND_CSD command bus transaction. The timeout values for the response and the data block are Ncr (Since the Nac is still unknown).

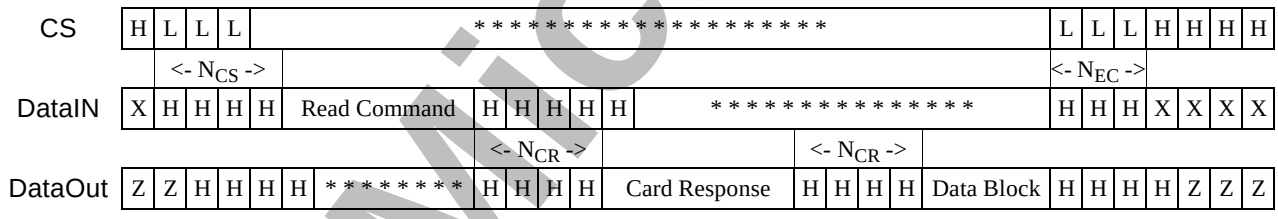


Figure 56: Read CSD - bus timing

7.5.3 Data 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 after the falling edge of CS.

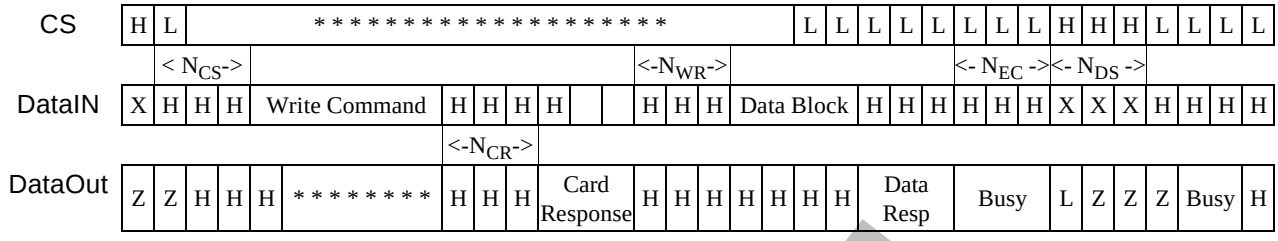


Figure 57: Write operation - bus timing

The following figure describes stop transmission operation in Multiple Block Write transfer.

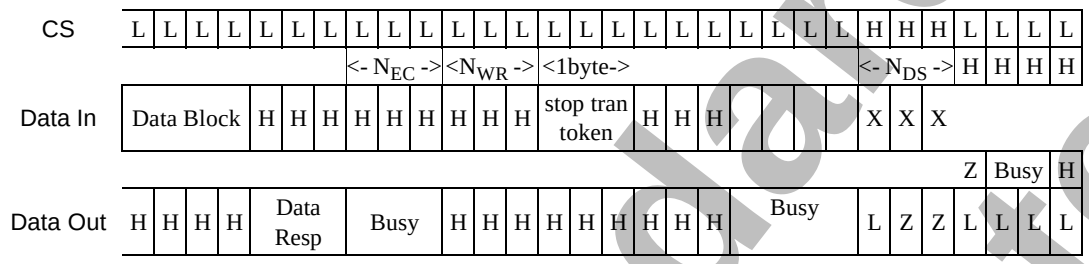


Figure 58: Stop Transmission in Write Multiple Block

7.5.4 Timing Values

	Min	Max	Unit
N_{CS}	0	-	8 clock cycles
N_{CR}	1	8	8 clock cycles
N_{RC}	1	-	8 clock cycles
N_{AC}	1	spec. in the CSD	8 clock cycles
N_{WR}	1	-	8 clock cycles
N_{EC}	0	-	8 clock cycles
N_{DS}	0	-	8 clock cycles

Table 60: Timing values

7.6 SPI Electrical Interface

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

7.7 SPI Bus Operating Conditions

Identical to SD mode

7.8 Bus Timing

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

8 SD Memory Card mechanical specification

This chapter describes the mechanical and electromechanical features of the SD Memory Card, and furthermore the minimal recommendations to the SD Memory Card connector. All technical drafts follow DIN ISO standard.

The functions of the card package are:

- protecting the chip
- easy handling for the end user
- reliable electrical interconnection
- reliable write protect/card detection capability
- bearing textual information and image
- appealing appearance

The functions of the connector are:

- attaching and fixing the card
- electrical interconnecting the card to the system board
- write protect/card detect indication
- optional: switch on/off power supply
- protection against card inverse insertion

8.1 Card package

Every card package shall have the characteristics described in the following sections.

8.1.1 External signal contacts (ESC)

Number of ESC	9
distance from front edge	1.2 mm
ESC grid	2.5mm
contact dimensions	1.7mm x 4.0mm
electrical resistance	30 mΩ (worst case: 100 mΩ)
micro interrupts	< 0.1 µs

Table 61: SD Memory Card Package - External Signal Contacts

8.1.2 Design and format

Dimensions SD Memory Card package	24mm x 32mm; (min. 23.9mm x 31.9mm; max.24.1mm x 32.1mm) other dimensions Figure 59 testing according to MIL STD 883, Meth 2016
thickness	2.1mm +- 0.15mm
label or printable area	whole card except contact area
surface	plain (except contact area)
edges	smooth edges, see Figure 60, Figure 61
inverse insertion	protection on left corner (top view) see Figure 62
position of ESC contacts	along middle of shorter edge

Table 62: SD Memory Card Package - Dimensions

8.1.3 Reliability and durability

temperature	operation: -25°C / 85°C (Target spec) storage: -40°C (168h) / 85°C (500h) junction temperature: max. 95°C
moisture and corrosion	operation: 25°C / 95% rel. humidity storage: 40°C / 93% rel. hum./500h salt water spray: 3% NaCl/35C; 24h acc. MIL STD Method 1009
durability	10.000 mating cycles; Test procedure: tbd.
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 <= -0.1 mm/cm² within contour; no cracks; no pollution (fat, oil dust, etc.)
Minimum moving force of WP switch	40gf (Ensures that the WP switch will not slide while it is inserted to the connector).
WP Switch cycles	t.b.d

Table 63: Reliability and Durability

8.1.4 Electrical Static Discharge (ESD) Requirements (Target spec)

ESD testing should be conducted according to ANSI EOS/ESD-S5.1 1998

Required ESD parameters are:

(1) Human body model +- 4 KV 100 pf / 1.5 Kohm

(2) Machine model +- 0.25 KV 200 pf / 0 ohm

Contact Pads:

+/- 4kV, Human body model according to ANSI EOS/ESD-S5.1-1998

Non Contact Pads area:

+/-4kV (coupling plane discharge)
+/-4kV (air discharge)
Human body model according to IEC61000-4-2

Voltages higher than 4kV will **T.B.D**.

8.1.5 Quality assurance

The product traceability shall be ensured by an individual card identification number.

8.2 Mechanical form factor

The following 3 technical drawings define the card package.

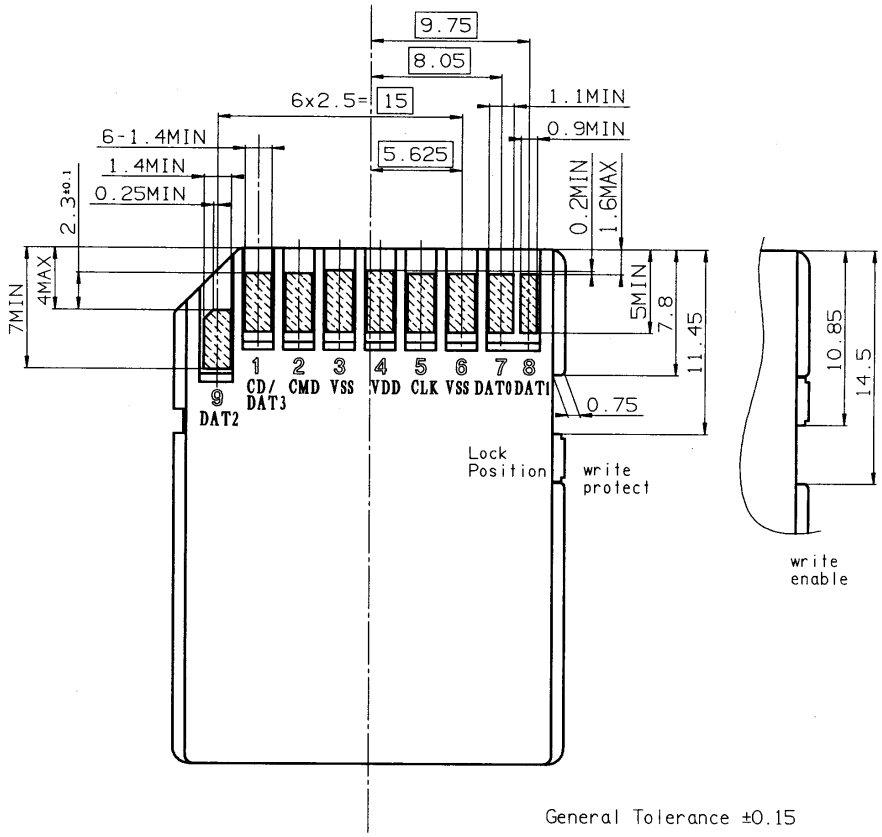


Figure 59: SD Memory Card - Mechanical Description (1 out of 3)

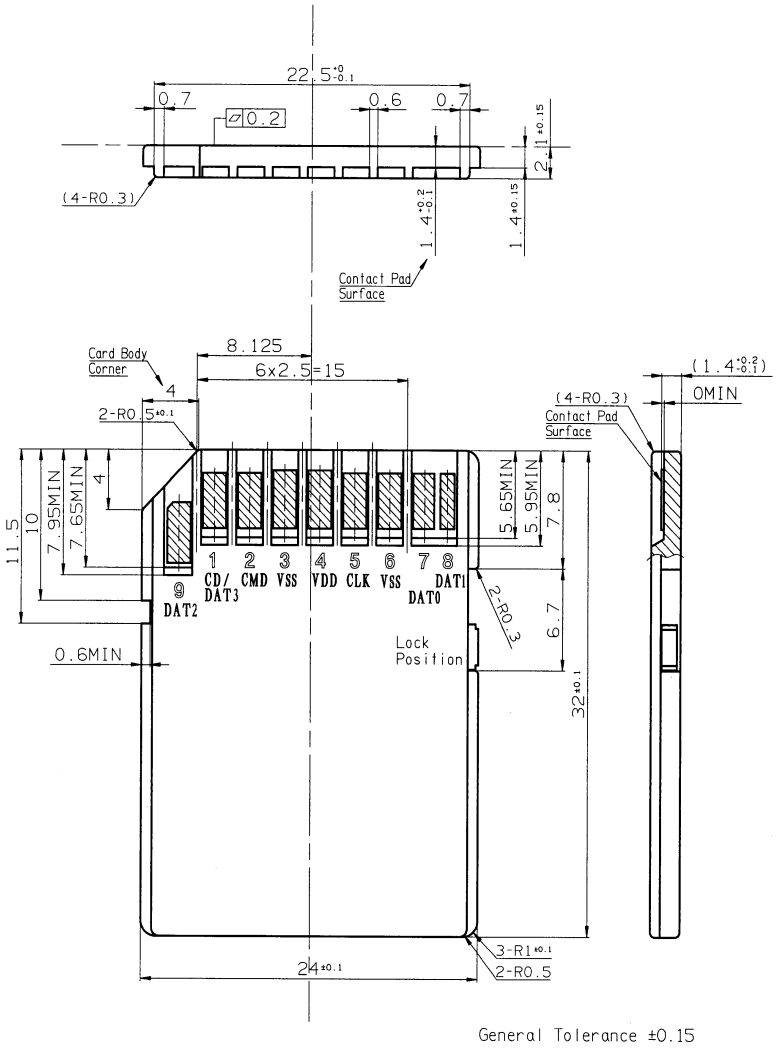


Figure 60: SD Memory Card - Mechanical Description (2 out of 3)

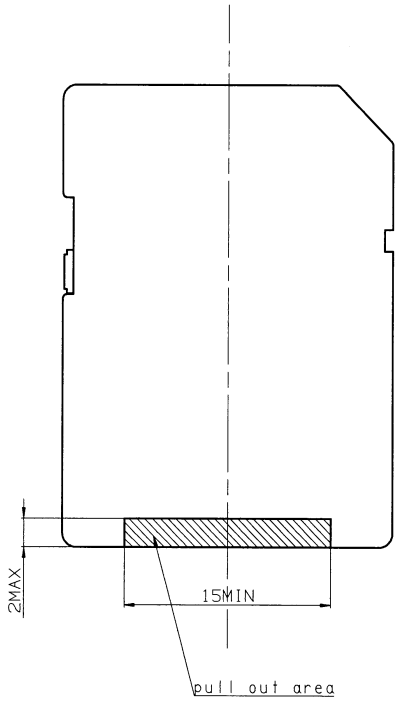


Figure 61: SD Memory Card - Mechanical Description (3 out of 3)

8.3 System: card and connector

The description of the connector is out of the scope of this document. However, minimal recommendations to the connector comprise the ability to guarantee Write Protect and Card Detection, hot insertion and removal of the card, and to prevent inverse insertion.

8.3.1 Card hot insertion

To guarantee a reliable initialization during hot insertion, some measures shall be taken on the host side. For instance, a special hot-insertion capable card connector may be used to guarantee the proper sequence of card pin connection.

The card contacts are contacted in three steps:

1) Ground Vss (pin 3) and supply voltage Vdd (pin 4).
2) CLK, CMD, DAT0, DAT1, DAT2 and Vss (pin 6).
3) CD / DAT3 (pin 1).

Pins 3 and 4 should make first contact when inserting, and release last when extracting.

As another method, a switch could ensure that the power is switched on only after all card pads are contacted. Of course, any other similar mechanism is allowed.

8.3.2 Inverse insertion

Inverse insertion is prevented by the reclining corners of SD Memory Card and connector.

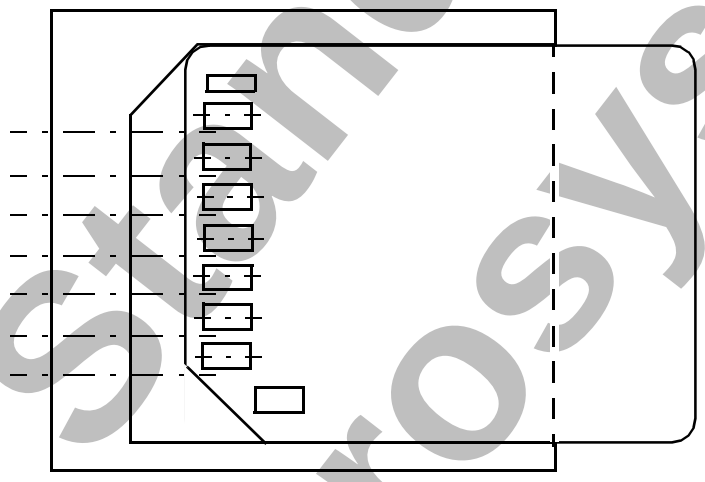


Figure 62: Inverse insertion

8.4 Thin (1.4mm) SD Memory Card (Preliminary)

SD Memory cards with mechanical dimensions that will be suitable for extra small applications will be available.

The "Thin SD Memory card" has exactly the same form factor as the SD Memory Card that is given in Chapter 8.2 above, except it thickness. The thickness of the "Thin SD Memory Card" is 1.4mm.

The mechanical drawing of "Thin SD Memory Card" was not finilized yet. A detailed drawing will appear in next revision of SD Physical Layer Specification.

9 Appendix

9.1 Power Supply Decoupling

The $V_{SS1,}$ V_{SS2} and V_{DD} lines supply the card with operating voltage. For this, decoupling capacitors for buffering current peak are used. These capacitors are placed on the bus side corresponding to Figure 63.

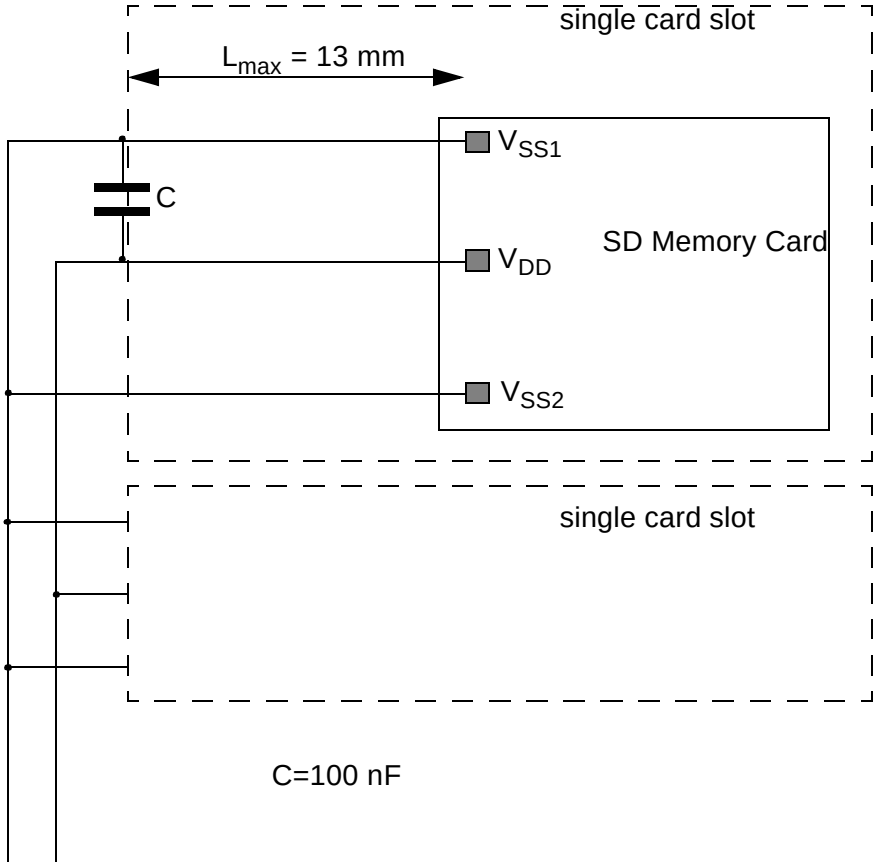


Figure 63: Power supply decoupling

The host controller includes a central buffer capacitor for V_{DD}. Its value is 1 µF/slot

9.2 Connector

The connector described in this chapter serves as an example and is subject to further changes.

9.2.1 General

The connector housing which accommodates the card is formed of plastic. Inside are 9 contact springs for contacting the pads of the inserted card. As an option a Write Protect/Card Detection

switch shall be part of the housing in order to be able to detect the position of the Write Protect sliding tablet on the card. Testing procedures are performed according to DIN IEC 68.

9.2.2 Card Insertion and Removal

Insertion of the SD Memory Card is only possible with the contact area of the card and the contact area of the connector in the correct position to each other. This is ensured by the reclining corners of the card and the connector, respectively.

To guarantee a reliable initialization during hot insertion, some measures must be taken on the host side. One possible solution is shown in Figure 64. It is based on the idea of a defined sequence for card contact connection during the card insertion process. The card contacts are contacted in three steps:

1) Ground Vss (pin 3) and supply voltage Vdd (pin 4).
2) CLK, CMD, DAT0, DAT1, DAT2 and Vss (pin 6).
3) CD / DAT3 (pin 1).

The pins 3 and 4 should make first contact when inserting and release last when extracting.

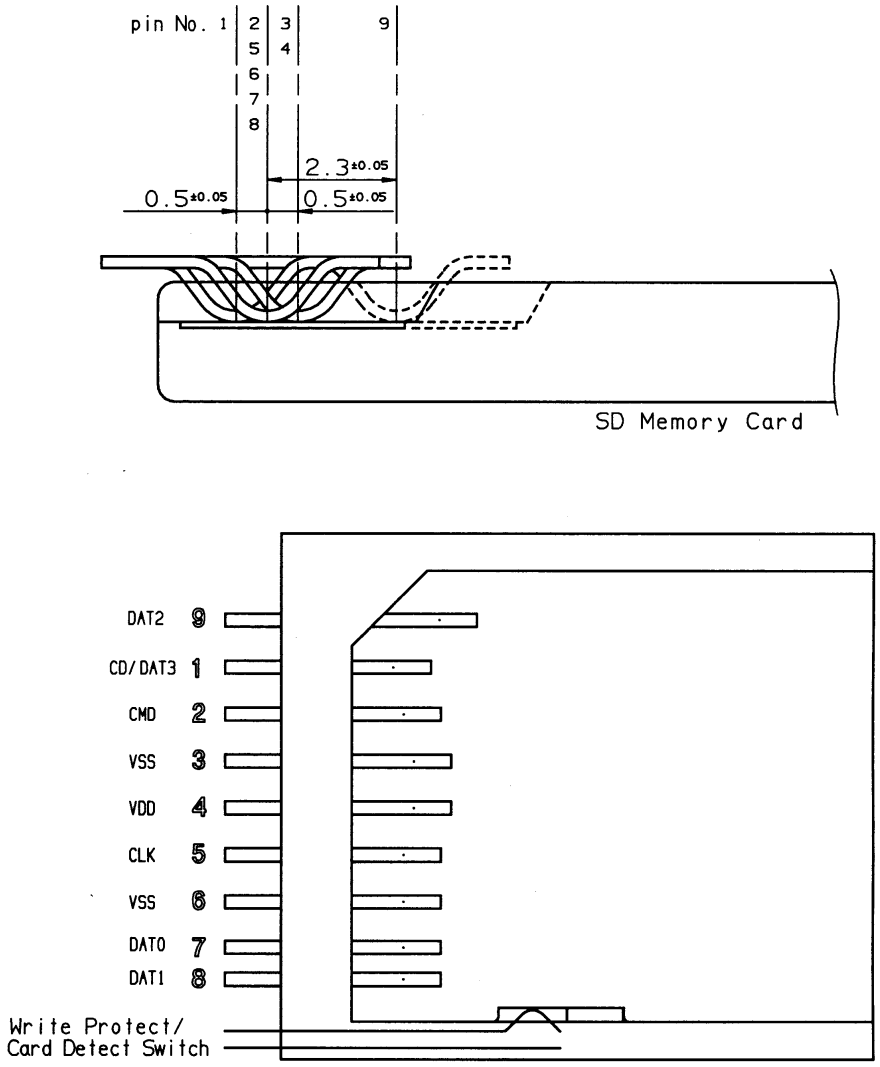


Figure 64: Modified SD Memory Card connector for hot insertion

9.2.3 Characteristics

The features described in the following must be considered when designing a SD Memory Card connector. The given values are typical examples.

- **Mechanical Characteristics**

  - Max. number of mating operations > 10000
  - Contact force 0.2.....0.4 N

- Total pulling force	min.2 N	DIN IEC 512 part 7
- Total insertion force	max. 40 N	DIN IEC 512 part 7
- Vibration and High Frequency		
- Mechanical frequency range	10.....2000 Hz	DIN IEC 512 part 2 and 4
- Acceleration	2 g	
- Shock:		
- Acceleration	5 g	

- **Electrical Characteristics**

DIN IEC 512

- Contact resistance	100 mΩ
- Current carrying capacity at 25°C	0.5 A
- Insulation resistance	> 1000 MΩ, > 100 MΩ after test
- Operating voltage	3.3 V
- Testing voltage	500 V
- Operating current	100 mA max.

- **Climatic Characteristics**

DIN IEC 512 part 6-9

- Operating temperature	-25°C.....90°C
- Storage temperature	-40°C.....90°C
- Humidity	95% max. non condensing

- This page was left empty intentionally -

10 Abbreviations and terms

block	a number of bytes, basic data transfer unit
broadcast	a command sent to all cards on the SD bus
CID	Card IDentification number register
CLK	clock signal
CMD	command line or SD bus command (if extended CMDXX)
CRC	Cyclic Redundancy Check
CSD	Card Specific Data register
DAT	data line
DSR	Driver Stage Register
ECC	Error Correction Code
Flash	a type of multiple time programmable non volatile memory
group	a number of sectors, composite erase and write protect unit
LOW, HIGH	binary interface states with defined assignment to a voltage level
NSAC	defines the worst case for the clock rate dependent factor of the data access time
MSB, LSB	the Most Significant Bit or Least Significant Bit
MTP	Multiple Time Programmable memory
OCR	Operation Conditions Register
open-drain	a logical interface operation mode. An external resistor or current source is used to pull the interface level to HIGH, the internal transistor pushes it to LOW
OTP	One Time Programmable memory
payload	net data
push-pull	a logical interface operation mode, a complementary pair of transistors is used to push the interface level to HIGH or LOW
RCA	Relative Card Address register
ROM	Read Only Memory
sector	a number of blocks, basic erase unit
stuff bit	filling bits to ensure fixed length frames for commands and responses
SPI	Serial Peripheral Interface
TAAC	defines the time dependent factor of the data access time
tag	marker used to select groups or sector to erase
TBD	To Be Determined (in the future)
three-state driver	a driver stage which has three output driver states: HIGH, LOW and high impedance (which means that the interface does not have any influence on the interface level)
token	code word representing a command
V_{DD}	+ power supply
V_{SS}	power supply ground

SD Memory Card Specifications

Part 2
FILE SYSTEM SPECIFICATION

Version 1.0

February 2000

SD Group

Matsushita Electric Industrial Co., Ltd. (MEI)

SanDisk Corporation

Toshiba Corporation

Conditions for publication

Publisher and Copyright Holder

SD Group(MEI, SanDisk, Toshiba)

Confidentiality

This document shall be treated as confidential under the Non Disclosure Agreement which has been signed by the obtainer. Reproduction in whole or in part is prohibited without prior written permission of SD Group.

Exemption

None will be liable for any damages from use of this document.

This page is intentionally left blank.

INDEX

Part2 FILE SYSTEM SPECIFICATION

Annex

List of Figures

List of Tables

This page is intentionally left blank.

1. General

1.1 Scope

This part specifies the volume structure and file structure of the SD Memory Card (Secure Digital Memory Card). The SD Memory Card file system that defines the logical structure uses FAT file system based on ISO/IEC 9293 standard.

It also supports two areas: one for the Data Area that user can access without mutual authentication and one for the Protected Area that user can access after mutual authentication.

1.2 Normative References

1) ISO/IEC646:1991
Information technology - ISO 7-bit code character set for information interchange

2) ISO/IEC9293:1994
Information technology - Volume and file structure of disk cartridges for information interchange

1.3 Definitions

1.3.1 byte

A string of binary digits operated upon as a unit.

1.3.2 defective sector

A sector that cannot be read or written.

1.3.3 descriptor

A recorded structure containing information about the volume or a file.

1.3.4 file

A named collection of information.

1.3.5 sector / block

A unit of data that can be accessed independently of other units on the SD Memory Card.

1.3.6 partition

An extent of sectors within a volume.

1.3.7 user

A person or other entity that causes the invocation of the services provided by an implementation.

1.3.8 volume

A sector address space as specified in the relevant standard for recording.

1.4 Notations

1.4.1 Numerical notation

Numbers in decimal notation are represented by decimal digits, namely 0 to 9.

Numbers in hexadecimal notation are represented as a sequence of one or more hexadecimal digits namely 0 to 9 and A to F, prefixed by "0x".

ZERO represents a single bit with the value 0.

1.4.2 Arithmetic notation

The notation *ip(x)* shall mean the integer part of *x*.

The notation *ceil(x)* shall mean the minimum integer that is greater than *x*.

1.4.3 Character strings

A value for a sequence of bytes may be specified by a quoted sequence of characters, encoded according to the ISO/IEC 646 standard.

1.4.4 List of acronyms

BP : Byte Position within a certain field, starting with 0 from the first byte of the field.
FAT : File Allocation Table.
FDC : Flexible Disk Cartridge.

1.5 Data types

1.5.1 Numerical values in one-byte fields

A numerical value in a one-byte field shall be recorded as an 8-bit number in one-byte field.

1.5.2 Numerical values in two-byte fields

A numerical value in a two-byte field shall be recorded in the little endian representation. It shall be recorded according to ISO/IEC 9293.

1.5.3 Numerical values in four-byte fields

A numerical value in a four-byte field shall be recorded in the little endian representation. It shall be recorded according to ISO/IEC 9293.

1.5.4 Pairs of 12-bit integers

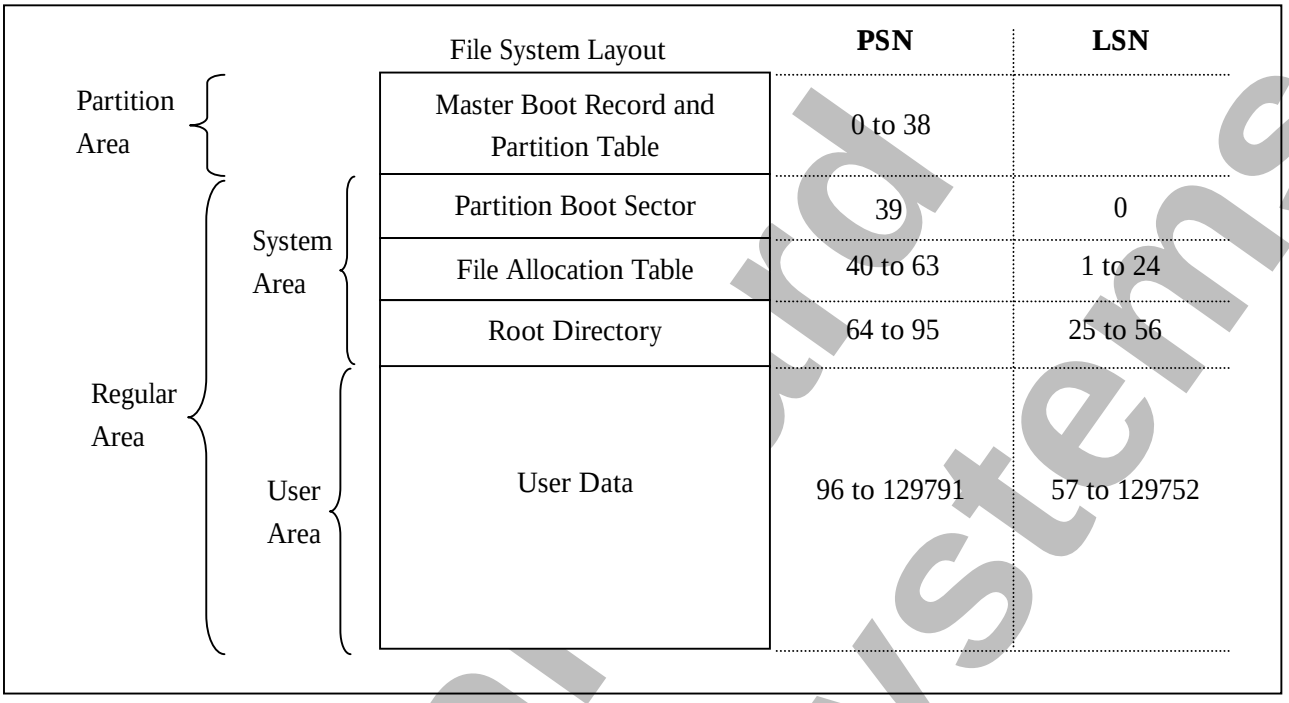
A pair of 12-bit numbers shall be recorded in three-byte field according to ISO/IEC 9293.

This page is intentionally left blank.

2. Volume Structure

The volume structure of the SD Memory Card is specified in this section. It defines the logical structure of the Data Area. For the identification of the Data Area as a partition, the first sector has Master Boot Record and Partition Table. And the SD Memory Card file system uses the FAT file system (ISO/IEC 9293) and supports both FAT12 and FAT16 as the file system type.

◊ **Figure 2-1 : Example of Volume Structure for Data Area**



PSN : Physical Sector Number

LSN : Logical Sector Number

2.1 Arrangement of the Data Area

2.1.1 Physical Address

Each sector shall be identified by a Physical Address comprising the parameters of SD Memory Card's own.

2.1.2 Physical Sector Number

Each sector on a volume shall be identified by a Physical Sector Number. There shall be a one-to-one correspondence between Physical Address and Physical Sector Number. The Physical Sector Numbers shall be assigned in an ascending sequence, beginning with 0.

2.1.3 Logical Sector Number

Each sector on a partition shall be identified by a Logical Sector Number. The first sector of the partition shall be assigned 0 as Logical Sector Number. There shall be a one-to-one correspondence between Physical Sector Number.

2.1.4 Partition Area and Regular Area

The space on Data Area shall be divided into two parts: Partition Area and Regular Area. And the Regular Area shall be divided into System Area and User Area.

The Partition Area shall occupy sectors with the Physical Sector Numbers 0 to *NOM-1*, where *NOM* is the number of sectors in the Master Boot Record and Partition Table.

The Regular Area is a partition of the volume, and divided into System Area and User Area.

The System Area shall occupy sectors with the Physical Sector Numbers *NOM* to *NOM+SSA*-1, where *SSA* is the number of sectors in the System Area. The System Area shall contain Descriptors that specify the recording format of the Regular Area. No part of any file shall be contained in the System Area.

The User Area shall occupy sectors with the Physical Sector Numbers starting with *NOM+SSA*. The User Area shall contain files and directories, and be recorded user data.

2.2 Arrangement of the User Area

2.2.1 Clusters

The User Area shall be organized into units of allocation called clusters. Each cluster shall consist of the same number of sectors. Each cluster shall be identified by a unique Cluster Number. Cluster Numbers shall be assigned integer number starting with 2.

2.2.2 Status of clusters

A status shall be assigned to each cluster, and shall be one of the following:

- allocated to a file
  The cluster is already allocated.
- available for allocation
  The cluster is prepared for allocate.
- defective
  The cluster is defective. This cluster cannot be allocated.

The status of each cluster shall be identified according to ISO/IEC 9293.

2.3 Arrangement of the Partition Area

The first sector of the Data Area has a Master Boot Record that includes executable codes and Partition Table that includes the information to identify the partition.

◊ **Table 2.3-1 Master Boot Record and Partition Table**

BP	Length	Field Name	Contents
0	446	Master Boot Record	Not Restricted
446	16	Partition Table (partition1)	Refer to Table2.3-2
462	16	Partition Table (partition2)	All 0x00
478	16	Partition Table (partition3)	All 0x00
494	16	Partition Table (partition4)	All 0x00
510	2	Signature Word	0x55, 0xaa

(BP 0 to 445) Master Boot Record

The content of this field is not specified by this specification.

(BP 446 to 461) Partition Table (partition1)

This field shall specify the information of first partition in the volume. This partition means Regular Area that user can access without mutual authentication. It shall be recorded according to Table 2.3-2.

(BP 462 to 477) Partition Table (partition2)

This field shall be recorded as ZEROs, as a volume shall consist of single Regular Area.

(BP 478 to 493) Partition Table (partition3)

This field shall be recorded as ZEROs, as a volume shall consist of single Regular Area.

(BP 494 to 509) Partition Table (partition4)

This field shall be recorded as ZEROs, as a volume shall consist of single Regular Area.

(BP 510 and 511) Signature Word

This field shall be recorded as 0x55 (BP 510) and 0xaa (BP 511).

◊ **Table 2.3-2 Partition Table**

BP	Length	Field Name	Contents
0	1	Boot Indicator	0x00 or 0x80
1	1	Starting Head	Numeric Value
2	2	Starting Sector / Starting Cylinder	Numeric Value
4	1	System ID	0x01 or 0x04 or 0x06
5	1	Ending Head	Numeric Value
6	2	Ending Sector / Ending Cylinder	Numeric Value
8	4	Relative Sector	Numeric Value
12	4	Total Sector	Numeric Value

(BP 0) Boot Indicator
This field shall be recorded as 0x80 if SD Memory Card is used for boot. Otherwise, this field shall be recorded as 0x00.

(BP 1) Starting Head
This field shall specify the starting head of the partition.

(BP 2 and 3) Starting Sector / Starting Cylinder
This field shall specify the starting sector and cylinder of the partition. 6 bits (Bit 0 to Bit 5 in BP 2) in this field shall be used for starting sector. 10 bits (Bit 6 and Bit 7 in BP 2, Bit 0 to Bit 7 in BP 3) in this field shall be used for starting cylinder.

(BP 4) System ID
This field shall specify the type of file system. It shall be recorded as 0x01 if the partition size is less than 32680 sectors. And it shall be recorded as 0x04 if the one is less than 65536 sectors. Otherwise, it shall be recorded as 0x06.

(BP 5) Ending Head
This field shall specify the ending head of the partition.

(BP 6 and 7) Ending Sector / Ending Cylinder
This field shall specify the ending sector and cylinder of the partition. 6 bits (Bit 0 to Bit 5 in BP 6) in this field shall be used for ending sector. 10 bits (Bit 6 and Bit 7 in BP 6, Bit 0 to Bit 7 in BP 7) in this field shall be used for ending cylinder.

(BP 8 to 11) Relative Sector
This field shall specify the number of sectors existing before the starting sector of this partition.

(BP 12 to 15) Total Sector
This field shall specify the number of sectors on the partition.

2.4 Arrangement of the System Area

2.4.1 System Area

The System Area shall contain the Partition Boot Sector, the Root Directory and the File Allocation Table (FAT) recorded twice.

2.4.2 Partition Boot Sector

The first sector of the System Area shall contain the Partition Boot Sector including the FDC Descriptor. The FDC Descriptor shall contain the parameters for the partition.

2.4.3 File Allocation Table (FAT)

The FAT shall contain a Format Identifier and some entries, each of which indicates cluster of the User Area. These entries shall be numbered consecutively starting with 2 and the Entry Number shall be equal to the Cluster Number of the corresponding cluster.
Each entry in the FAT shall indicate the status of the corresponding cluster. The FAT entries shall be used to identify the set of clusters that are allocated to each file.

2.4.4 Root Directory

The Root Directory shall be recorded in the System Area following the second occurrence of the FAT.

3. File Structure

3.1 Partition Boot Sector

There is a Partition Boot Sector at the head of a partition and it contains an FDC Descriptor or an Extended FDC Descriptor. The FDC Descriptor and the Extended FDC Descriptor are compliant to ISO/IEC 9293. The Extended FDC is used for the default.

◊ **Table 3.1-1 FDC Descriptor**

BP	Length	Field Name	Contents
0	3	Jump Command	bytes
3	8	Creating System Identifier	a-characters
11	2	Sector Size	Numeric Value
13	1	Sectors per Cluster	Numeric Value
14	2	Reserved Sector Count	Numeric Value
16	1	Number of FATs	Numeric Value
17	2	Number of Root-directory Entries	Numeric Value
19	2	Total Sectors	Numeric Value
21	1	Medium Identifier	0xf8
22	2	Sectors per FAT	Numeric Value
24	2	Sectors per Track	Numeric Value
26	2	Number of Sides	Numeric Value
28	2	(Reserved for future standardization)	0x0000
30	480	(Reserved for system use)	Not Restricted
510	2	Signature Word	0x55, 0xaa

◊ **Table 3.1-2 Extended FDC Descriptor**

BP	Length	Field Name	Contents
0	3	Jump Command	bytes
3	8	Creating System Identifier	a-characters
11	2	Sector Size	Numeric Value
13	1	Sectors per Cluster	Numeric Value
14	2	Reserved Sector Count	Numeric Value
16	1	Number of FATs	Numeric Value
17	2	Number of Root-directory Entries	Numeric Value
19	2	Total Sectors	Numeric Value
21	1	Medium Identifier	0xf8
22	2	Sectors per FAT	Numeric Value
24	2	Sectors per Track	Numeric Value
26	2	Number of Sides	Numeric Value
28	4	Number of Hidden Sectors	Numeric Value
32	4	Total Sectors	Numeric Value
36	1	Physical Disk Number	0x80
37	1	Reserved	0x00
38	1	Extended Boot Record Signature	0x29
39	4	Volume ID Number	Numeric Value
43	11	Volume Label	d-characters
54	8	File System Type	d-characters
62	448	(Reserved for system use)	Not Restricted
510	2	Signature Word	0x55, 0xaa

(BP 0 to 2) Jump Command
This field shall specify the jump command to the boot program. It shall be recorded as 0xeb (BP 0), 0xXX (BP 1) and 0x90 (BP 2), or 0xe9 (BP 0), 0xXX (BP 1) and 0xXX (BP 2). 0xXX means that the value is not specified in this specification.

(BP 3 to 10) Creating System Identifier
This field shall specify identification for the system. This field shall be recorded using a-characters and according to ISO/IEC 9293 9.

(BP 11 and 12) Sector Size
This field shall specify the size of a sector in bytes. It shall be recorded as the number 512.

(BP 13) Sectors per Cluster
This field shall specify the number of sectors per cluster. It shall be recorded the following number: 1, 2, 4, 8, 16, 32 or 64. The Cluster Size shall be the multiple size of the erase block size determined by the physical layer. If the erase block size is larger than 32KB, the Cluster Size shall be 32KB and this field shall be recorded 64.

(BP 14 and 15) Reserved Sector Count
This field shall specify the number of sectors reserved for system use. It shall be recorded as the number 1.

(BP 16) Number of FATs
This field shall specify the number of FATs. It shall be recorded as the number 2.

(BP 17 and 18) Number of Root-directory Entries
This field shall specify the number of entries in the Root Directory. It shall be recorded as the number 512.

(BP 19 and 20) Total Sectors
This field shall specify the number of sectors on the partition. It shall be recorded according to ISO/IEC 9293 9.

(BP 21) Medium Identifier
This field shall be recorded as 0xf8 for this specification.

(BP 22 and 23) Sectors per FAT
This field shall specify the number of sectors that shall be occupied by each FAT. It shall be recorded according to ISO/IEC 9293 9.

(BP 24 and 25) Sectors per Track
This field shall specify the number of sectors in each track. This parameter depends on the SD Memory Card's parameter. It shall be recorded according to ISO/IEC 9293 9.

(BP 26 and 27) Number of Sides
This field shall specify the number of sides that can be recorded. This parameter depends on the SD Memory Card's parameter. It shall be recorded according to ISO/IEC 9293 9.

(BP 28 and 29) Field reserved for future standardization
This field shall be reserved for future standardization. It shall contain only ZEROs.

(BP 30 to 509) Field reserved for system use
This field shall be reserved for system use. It shall be not specified in this specification.

(BP 510 and 511) Signature Word
This field shall be recorded as 0x55 (BP 510) and 0xaa (BP 511).

(Extended FDC Descriptor BP 28 to 31) Number of Hidden Sectors
This field shall specify the number of sectors existing before the starting sector of this partition.

(Extended FDC Descriptor BP 32 to 35) Total Sectors
This field shall specify the number of sectors on the partition if the field in BP 19 and 20 is recorded as ZEROs. It shall be recorded according to ISO/IEC 9293 9.

(Extended FDC Descriptor BP 36) Physical Disk Number
This field shall specify the BIOS physical disk number. This field shall be recorded as 0x80.

(Extended FDC Descriptor BP 37) Reserved
This field shall be reserved for future standardization. It shall be recorded as ZEROs. It shall be recorded according to ISO/IEC 9293 9.

(Extended FDC Descriptor BP 38) Extended Boot Record Signature
This field shall be used to identify the descriptor type in the Extended FDC Descriptor when either BP 19 or BP 20 is not recorded as ZEROs. This field shall be recorded as 0x29.

(Extended FDC Descriptor BP 39 to 42) Volume ID Number
This field shall specify the volume identification number. It shall be recorded according to ISO/IEC 9293 9.

(Extended FDC Descriptor BP 43 to 53) Volume Label
This field shall specify the volume label. It shall be recorded according to ISO/IEC 9293 9.

(Extended FDC Descriptor BP 54 to 61) File System Type
This field shall specify the type of the file system. It shall be recorded according to ISO/IEC 9293 9.

(Extended FDC Descriptor BP 62 to 509) Field reserved for system use
This field shall be reserved for system use. It shall be not specified in this specification.

(Extended FDC Descriptor BP 510 and 511) Signature Word
This field shall be recorded as 0x55 (BP 510) and 0xaa (BP 511).

3.2 File Allocation Table

The File Allocation Table supports both the 12-bit FAT and the 16-bit FAT. The FAT structure is compliant to ISO/IEC 9293. The FAT type shall be determined by the number of clusters that depends on the parameter from the physical layer. If the cluster number is less than 4085, FAT12 shall be used. Otherwise, FAT16 shall be used. The first byte of the FAT shall specify the format identifier and be recorded 0xf8. In case of FAT12, the byte 2 and 3 shall be recorded as 0xff each. In case of FAT16, the byte 2, 3 and 4 shall be recorded as 0xff each. The sectors of the FAT may include unused area, because the number of clusters shall determine the FAT size in byte. This unused area shall be recorded as ZEROs.

◊ Table 3.2-1 FAT Entry Value

FAT Entry Value		Contents
FAT12	FAT16	
000	0000	Indicates that the corresponding cluster is not in use and may be allocated to a file or a directory.
002 to MAX	0002 to MAX	Indicates that the corresponding cluster is already allocated. The value of the entry is the cluster number of the next cluster following this corresponding cluster. Max shall be the Maximum Cluster Number.
MAX+1 to FF6	MAX+1 to FFF6	Shall be reserved for future standardization and shall not be used.
FF7	FFF7	Indicates that the corresponding cluster has a defective cluster.
FF8 to FFF	FFF8 to FFFF	The corresponding cluster is already allocated, and it is the final cluster of the file.

3.3 File directories

3.3.1 Characteristics

A Directory is a Descriptor that shall contain a set of Directory entries each of which identifies a file, a Volume Label, another Directory or is unused. A Directory can contain the 65536 Directory entries.
The format of the file name for the Directory entries should supports 8.3 format. Although the Long File Name (LFN) can exist in the Directory entries, the SD Memory Card file system may ignore these entries, and refers to only the file name of 8.3 format that is stored with the LFN.
The character code in the Directory entry can be used the code which is permitted by the ISO/IEC 9293.

3.3.2 Directory entry types

Directory entries shall contain descriptive information about the files recorded on the partition. There are some types of these entries as below:

- File Entry

  A File Entry shall specify information of a file.

- Volume Label Entry

  A Volume Label Entry shall specify the volume label of the partition.

- Sub-directory Pointer Entry

  A Sub-directory Pointer Entry shall specify information of a directory.

- Sub-directory Identifier Entry

  A Sub-directory Identifier Entry shall identify a file as a Sub-directory.

- Sub-directory Parent Pointer Entry

  A Sub-directory Parent Pointer Entry shall specify information of its parent directory.

- Not-currently-in-use Entry

  A Not-currently-in-use Entry shall specify the entry is not used and able to allocate.

- Never-used Entry

  A Never-used Entry shall specify the end of the directory. It shall not appear before any other type of Directory entry.

These entries shall be recorded according to ISO/IEC 9293 11.

3.3.3 General definition of Directory entry fields

Table 3.3.3-1 indicates the structure of the Directory entry field.

◊ **Table 3.3.3-1 Directory Entry Field**

BP	Length	Field Name	Contents
0	8	Name	Depends on entry type
8	3	Name Extension	d-characters
11	1	Attributes	8 bits
12	10	Reserved Field	bytes
22	2	Time Recorded	Numeric Value
24	2	Date Recorded	Numeric Value
26	2	Starting Cluster Number	Numeric Value
28	4	File Length	Numeric Value

(BP 0 to 7) Name
The content and the description of this field shall depend on the entry type. It shall be recorded according to ISO/IEC 9293 11.

(BP 8 to 10) Name Extension
The content and the description of this field shall depend on the entry type. The content of this field shall be d-characters. It shall be recorded according to ISO/IEC 9293 11.

(BP 11) Attributes
This field shall specify the attributes of the entry. It shall be recorded according to ISO/IEC 9293 11.

(BP 12 to 21) Reserved Field
The content of this field shall depend on the entry type. If this entry is LFN entry, this field shall not be specified in this specification. Otherwise, it shall be recorded as ZEROs.

(BP 22 and 23) Time Recorded
This field shall contain a 16-bit integer representing a time. It shall be recorded according to ISO/IEC 9293 11.

(BP 24 and 25) Date Recorded
This field shall contain a 16-bit integer representing a date. It shall be recorded according to ISO/IEC 9293 11.

(BP 26 and 27) Starting Cluster Number
The content of this field shall depend on the entry type. It shall be recorded according to ISO/IEC 9293 11.

(BP 28 to 31) File Length
The content of this field shall depend on the entry type. It shall be recorded according to ISO/IEC 9293 11.

3.4 User Area

The User Area shall be organized into clusters. Each cluster has a Cluster Number respectively. The first cluster in the User Area is corresponding to Cluster Number 2.

Although it is available to read/write by the sector, it is necessary to transact reading/writing with the unit whose minimum size is the same as that of the recommended reading/writing at the physical layer. Other than that, there are no special restrictions for the SD Memory Card file system.

Annex

Annex A: File System Layout

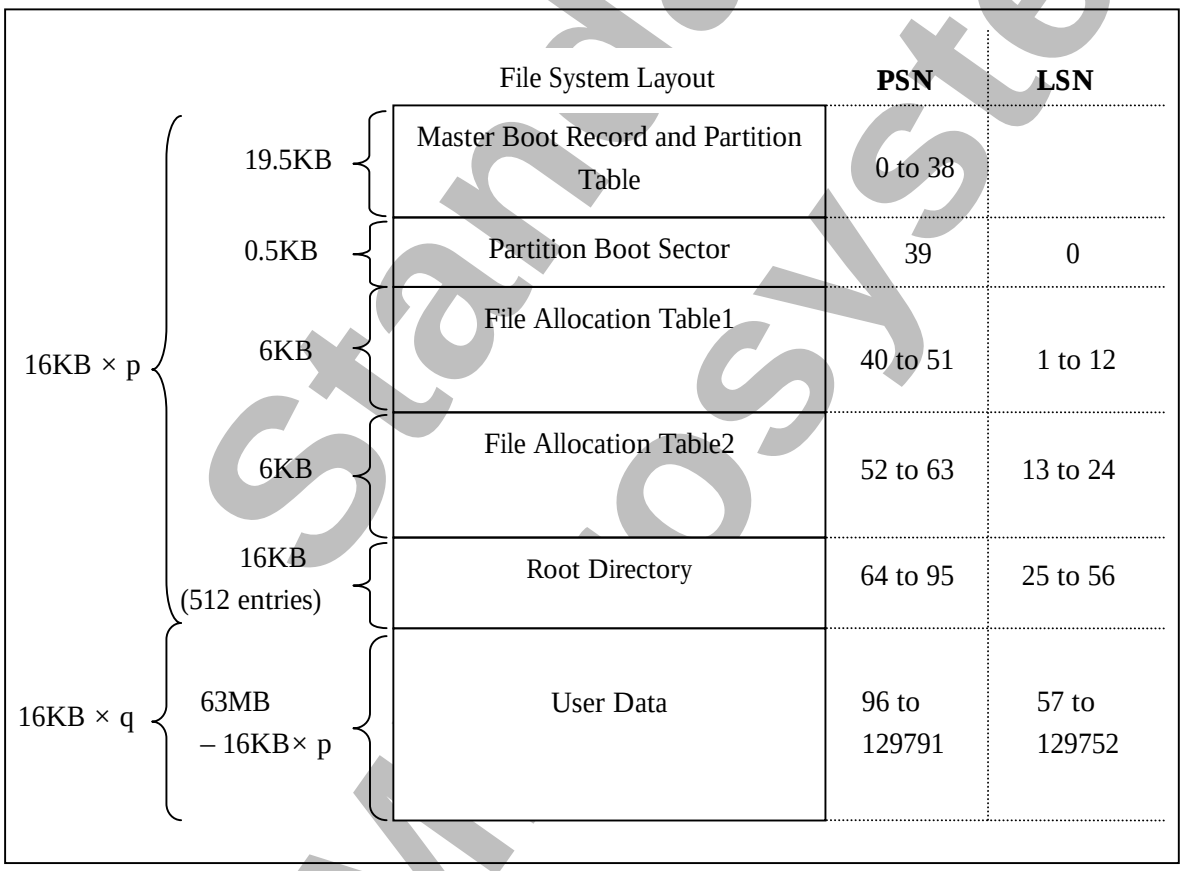
Reading/writing of the User Area should be done with the unit whose minimum size is the same as the recommended size of reading/writing at physical layer. Therefore, Cluster Size shall be determined considering the recommended size. And the head sector of the User Area must be started with the multiple offset of the Cluster Size.

The structure of the file system should be implemented as follows.

1. The combined size of Master Boot Record, Partition Table, Partition Boot Sector, File Allocation Table and Root Directory is a multiple size of the Cluster Size.
2. The number of the sectors before Partition Boot Sector adjusts the above size.
3. Master Boot Record and Partition Boot Sector belong to different cluster.
4. The first sector of the Master Boot Record and the first sector of the User Data are always placed on the cluster boundary.

The following is an example of a partition when the size of the cluster is 16KB.

◊ **Figure A-1 : Example of File System Layout**



p,q : a natural number

PSN : Physical Sector Number

LSN : Logical Sector Number

Annex B: CHS Recommendation

The following table shows the recommendation for CHS parameter. In this table, Card Capacity means the total size of Data Area and Protected Area.

◊ **Table B-1 CHS Recommendation**

Card Capacity	Number of Heads	Sectors per Track
~2MB	2	16
~16MB	2	32
~32MB	4	32
~128MB	8	32
~256MB	16	32
~504MB	16	63
~1008MB	32	63
~2016MB	64	63
~2048MB	128	63

The following is an example of a Partition Table when the size of the Data Area 63MB.

◊ **Table B-2 Example of Partition Table**

BP	Length	Field Name	Contents
0	1	Boot Indicator	0x00
1	1	Starting Head	1
2	2	Starting Sector / Starting Cylinder	8 / 0
4	1	System ID	0x06
5	1	Ending Head	7
6	2	Ending Sector / Ending Cylinder	32 / 506
8	4	Relative Sector	39
12	4	Total Sector	129753

Annex C: Sectors per Cluster and Boundary Unit Recommendation for Data Area

The following table shows the recommendation for Sectors per Cluster and Boundary Unit of Data Area.

Number of sectors before the starting sector of User Data is multiple size of Boundary Unit. In this table, Card Capacity means the total size of Data Area size and Protected Area size.

◊ **Table C-1 Sectors per Cluster and Boundary Unit Recommendation (Data Area)**

Card Capacity	Sectors per Cluster	Boundary Unit
~8MB	16	16
~64MB	32	32
~256MB	32	64
~1024MB	32	128
~2048MB	64	128

Maximum Data Area size and format parameters are shown in following table.

◊ **Table C-2 Maximum Data Area size and format parameters**

Card Capacity	Sectors per Cluster	Max Data Area size(sector)	an example(values depend on Data Area size)				
			Clusters	FAT Sec	Hidden	FAT bits	User Data Offset
~4MB	16	8032	498	2	27	12	64
~8MB	16	16224	1010	3	25	12	64
~16MB	32	32448	1011	3	57	12	96
~32MB	32	64896	2025	6	51	12	96
~64MB	32	129792	4053	12	39	12	96
~128MB	32	259584	8106	32	95	16	192
~256MB	32	519168	16216	64	95	16	256
~512MB	32	1038336	32432	127	225	16	512
~1024MB	32	2076672	64872	254	227	16	768
~2048MB	64	4153344	64884	254	227	16	768

However,

Card Capacity…SD Card Capacity.

Sectors per Cluster…number of sectors per cluster. This parameter is defined from Card Capacity.

Max Data Area size…maximum number of sectors for Data Area.

Clusters…number of clusters in User Data. This parameter varies with the Data Area size.

FAT Sec…number of sectors per FAT. This parameter varies with the Data Area size.

Hidden…number of sectors existing before Partition Boot Sector. This parameter varies with the Data Area size.

FAT bits…If the area is formatted with FAT12, FAT bits is 12. And If the area is formatted with FAT16, FAT bits is 16. This parameter varies with the Data Area size.

User Data Offset…number of sectors existing before the starting sector of User Data.

Sectors per Cluster is defined from Card Capacity. Clusters, FAT Sec, Hidden, FAT bits, and User Data Offset vary with the Data Area size (Use the parameters in Table C-1 for calculation).

Annex D: Format Parameter Computations

In this section, format parameter computations is proposed. Data Area should be formatted by the following steps.

1. Sectors per Cluster(SC) is determined from the area size.
2. Number of Root-directory Entries(RDE) is 512.
3. Sector Size(SS) is 512.
4. Reserved Sector Count(RSC) is 1.
5. Total Sectors(TS) is the number of all sectors of the area.
6. FAT bits(12[FAT12], 16[FAT16]) is determined by SC and TS.
7. Sectors per FAT(SF) is computed as following:

$$SF = ceil\{ \frac{TS/SC \times FAT\ bits}{SS \times 8} \}$$

8. Number of sectors in the system area(SSA) is computed as following:

$$SSA = RSC + 2 \times SF + ceil(\frac{32 \times RDE}{SS})$$

9. Number of Sectors in Master Boot Record(NOM) is computed as following:

$$NOM + SSA = BU \times n$$

   Here, n means the minimum natural number satisfying above expression. And BU means the Boundary Unit determined by Annex C.

10. If NOM isn't equal to BU, NOM is added BU.
11. Maximum Cluster Number(MAX) is computed as following:

$$MAX = ip(\frac{TS - NOM - SSA}{SC}) + 1$$

12. Sectors per FAT(SF') is recalculated as following:

$$SF' = ceil\{ \frac{[2 + (MAX - 1)] \times FAT\ bits}{SS \times 8} \}$$

    In this formula, 'MAX-1' means the number of clusters. And '2+(MAX-1)' means the number of FAT entries including two signature entries.

13. If SF' isn't equal to SF, SF' is used as SF. And recalculate from step 8.
14. If SF' is equal to SF, parameter computing is complete.

Example of a SD Memory Card including 63MB Data Area:

- TS=129792 Sectors

- SC=32 Sectors
- RDE=512 Entries
- SS=512 B
- RSC=1 Sectors
- FAT bits=12[FAT12]
- SF=12 Sectors
- SSA=57 Sectors
- NOM= 39 Sectors
- MAX=4054

Example of Extend FDC Descriptor for the Above Example:

◊ **Table D-1 Extended FDC Descriptor (Regular Area)**

BP	Length	Field Name	Contents
0	3	Jump Command	0xEB,0x00,0x90
3	8	Creating System Identifier	"SYSTEMID"
11	2	Sector Size	512
13	1	Sectors per Cluster	32
14	2	Reserved Sector Count	1
16	1	Number of FATs	2
17	2	Number of Root-directory Entries	512
19	2	Total Sectors	0
21	1	Medium Identifier	0xF8
22	2	Sectors per FAT	12
24	2	Sectors per Track	32
26	2	Number of Sides	8
28	4	Number of Hidden Sectors	39
32	4	Total Sectors	129753
36	1	Physical Disk Number	0x80
37	1	Reserved	0x00
38	1	Extended Boot Record Signature	0x29
39	4	Volume ID Number	0x01234567
43	11	Volume Label	"VOLUME1 "
54	8	File System Type	"FAT12 "
62	448	(Reserved for system use)	Not Restricted
510	2	Signature Word	0x55aa

SD Memory Card Specifications

Part 3
SECURITY SPECIFICATION

Version 1.0

February 2000

SD Group

Matsushita Electric Industrial Co., Ltd. (MEI)

SanDisk Corporation

Toshiba Corporation

This page is intentionally left blank

Conditions for publication

Publisher and Copyright Holder

SD Group(MEI, SanDisk, Toshiba)

Confidentiality

This document shall be treated as confidential under the Non Disclosure Agreement which has been signed by the obtainer. Reproduction in whole or in part is prohibited without prior written permission of SD Group.

Exemption

None will be liable for any damages from use of this document.

This page is intentionally left blank

Part 3 SECURITY SPECIFICATION

This page is intentionally left blank

1.General

1.1 Scope

The main objectives of the Security specifications of SD Memory Card are:

- To protect the copyrighted data recorded on the SD Memory Card from unauthorized use (for reproduction and duplication).
- To give independent protection for different pieces of copyrighted data of different applications (electronic music distribution EMD, electronic books, etc.).

This document and *"Content Protection for Recordable Media Specification SD Memory Card Book"*, that is developed by 4C Entity, LLC(IBM,Intel,MEI,Toshiba), contain the information on the functions required of the SD Memory Card to achieve the above objectives.

This document especially contains the security specification that is depending on the implementation of the SD Memory Card, more concretely,

(A) Data Element (SD-Card Specific)
Media Identifier

(B) Security Command set of SD Memory Card

(C) Random Number Generation on SD Memory Card

(D) File system (volume structure) of Protected Area on SD Memory Card

The following technologies are offered by *Content Protection for Recordable Media Specification SD Memory Card: Book:*

- Content and key encryption algorithm (C2 encryption),
- Revocation scheme of the unauthorized accessing device (Media Key Block),
- Authentication and Key Exchange mechanism (AKE) between SD Memory Card and the accessing devices.
- Data Structure on SD Memory Card
- File System (directory and file format) of Protected Area on SD Memory Card
- Content Encryption Format

etc.

1.2 References

4C Entity, LLC, *[Content Protection for Recordable Media Specification SD Memory Card Book, available soon.]*

SD Group, SD Memory Card Specifications Part1: Physical Layer Specifications

SD Group, SD Memory Card Specifications Part2: File System Specifications

2.Data Element

This Section describes the SD Memory Card specific 'data element'.

SD Memory Card non-specific data element is described in *Content Protection for Recordable Media Specification SD Memory Card Book*.

2.1 Media Identifier

SD Memory Card shall contain a 64-bit Media Identifier (ID_{media}), a part of which is unique by each SD Memory Card.

The Media Identifier logical format is shown in Table 2-1. As shown in Table 2-1, the least significant 56-bit(Byte"1" to Byte"7") of the Media Identifier is a SD Memory Card Specific part.

In Table2-1,

- The 4C Entity, LLC assigns each SD Memory Card Manufacturer a unique 1-byte value as the Manufacture ID field. (The detail is defined in *Content Protection for Recordable Media Specification SD Memory Card Book*)
- The SD Group assigns each SD Memory Card Manufacturer a unique 2-byte value as the OEM/Application ID value
- Each SD Memory Card Manufacturer assigns a unique 5-byte value as the Serial Number, which consists of 1-byte Product Revision (PRV) value, and 4-byte Product serial number (PSN) value.

Table 2.1: Media Identifier for SD Memory Card

Bit / Byte	7	6	5	4	3	2	1	0
0	Manufacturer ID (MID:1byte) assigned by 4C Entity, LLC							
1	OEM/Application ID(OID:2byte) assigned by SD Group							
2								
3	Product Revision(PRV:1byte)							
4	Product serial number(PSN:4byte)							
5								
6								
7								

3. Security Command set for copyright protection

3.1 Security Command List

In order to support a new set of commands that will be 'behind' the MULTIMEDIACARD standard, the new commands will be an Application Specific given commands and shall be preceded with CMD55 (APP_CMD). First describes the new set of commands. Note that all the commands do not use RCA (Relative Card Address). Therefor those commands shall be used after the card was selected (in '*tran_state*'). Note that the SD Memory CARD supports only a fix block size of 512 Bytes per block (Sector Size). The Sector Size is able to change by Set_Block_Size command, though it is not executable except for 512 Bytes Sector size because of restriction of ATA.

Table 3.1 Security Command List

CMD INDEX	Type	Argument	Resp	Abbriviation	Command Description
ACMD43	Adtc	[31:24]UNIT_COUNT: [23:16] MKB_ID: [15:0]UNIT_OFFSET:	R1	GET_MKB	Reads Media Key Block from the System Area of SD Memory Card. -'UNIT_COUNT' specifies the Number of units to read. (here, a unit=512 byte (fixed)) - 'MKB_ID' specifies the application unique number. - 'UNIT_OFFSET' specifies the start address(offset) to read.
ACMD44	Adtc	[31:0] stuff bits	R1	GET_MID	Reads Media ID from the System Area of SD Memory Card.
ACMD45	Adtc	[31:0] stuff bits	R1	SET_CER_RN1	AKE Command: Writes random number RN1 as challenge1 in AKE process. (See Note(1)(2))
ACMD46	Adtc	[31:0] stuff bits	R1	GET_CER_RN2	AKE Command: Reads random number RN2 as challenge2 in AKE process. (See Note (1))
ACMD47	Adtc	[31:0] stuff bits	R1	SET_CER_RES2	AKE Command: Writes RES2 as response2 to RN2 in AKE process. (See Note (1))
ACMD48	Adtc	[31:0] stuff bits	R1	GET_CER_RES1	AKE Command: Reads RES1 as response1 to RN1 in AKE process. (See Note (1))

ACMD18	Adtc	[31:0] stuff bits	R1	SECURE_READ_MULTI_BLOCK	Protected Area Access Command: Reads continuously transfer data blocks from Protected Area of SD Memory Card. The (essential) argument of this command as shown below is transferred securely in AKE command(ACMD45) (See Note (2)). - [31:24] 'Unit_Count' specifies the number of blocks to transfer. Block Size is fixed 512bytes. -[23] 'Reserve'.(This value shall be set to '0' for the future extension.) - [22:0] 'UNIT_Address' specifies the start address to read. ([ ] shows bit position of the (essential) argument)

ACMD25	Adtc	[31:0] stuff bits	R1	SECURE_WRITE_MULT_BLOCK	Protected Area Access Command: Writes continuously transfer data blocks to Protected Area of SD Memory Card. (See Note (4)) The (essential) argument of this command as shown below is transferred securely in AKE command(ACMD45) (See Note (2)). - [31:24] 'Unit_Count' specifies the number of blocks to transfer. Block Size is fixed 512bytes. -[23] Mode specifies the following: - Mode=0: This mode shall be used to write a data which should be shared by all applications, such as FAT associated data (e.g. Master boot record, Partition table, FAT directory entry,). -Mode= 1: This mode shall be used to write a data which should be protected from other application such as content associated data(e.g. Title Key, CCI). - [22:0] 'UNIT_Address' specifies the start address to write. ([ ] shows bit position of the (essential) argument)

ACMD38	Ac	[31:0] stuff bits	R1b	SECURE_ERASE	Protected Area Access Command: Erase a specified region of the Protected Area of SD Memory Card. (See Note(4).) The (essential) argument of this command as shown below is transferred securely In AKE command(ACMD45) (See Note (2)). - [31:24] 'Unit_Count' specifies the number of blocks to transfer. Block Size is fixed 512bytes. - [23:0] 'UNIT_Address' specifies the start address to erase. ([ ] shows bit position of the (essential) argument)

ACMD49	Ac	[31:0] stuff bits	R1b	CHANGE_SECURE_AREA	Protected Area Access Command: Change size of the Protected Area. See Note(3). The value of Unit_address [23:0] (discribed bellow) is given in units of MULT*BLOCK_LEN/512-1. Error occurs in the case that the Protected Area size (in Bytes) is set to a value other than multiplies of MULT*BLOCK_LEN. Note that the minimum User Data Area size is MULT*BLOCK_LEN. In that case Unit_address[23:0]=MULT*BLOCK_LEN/512-1. (About MULT*BLOCK_LEN, see chapter5.3 of SD-Card Specifications Part1: Physical Specifications). The (essential) argument of this command as shown below is transferred securely In AKE command(ACMD45) (see Note(2)). - [23:0]'UNIT_Address' The Protected Area follows the User Data Area in such a way that the first unit address (unit=512 byte) of the Protected Area follows the last unit address of the User Data Area and the highest unit address of the Protected Area is the highest unit address of the memory area. Under those conditions, [23:0]'UNIT_Address' is an address, in units of 512byte, of the end of the User Data Area. - [31:24] stuff bits ([ ] shows bit position of the (essential) argument)

ACMD26	Adtc	[31:0] stuff bits	R1	SECURE_WRITE_MKB	Protected Area Access Command: Overwrite the exist Media Key Block(MKB) on the System Area of SD Memory Card with new MKB.(See Note(4)) The (essential) argument of this command as shown below is transferred securely in AKE command (ACMD45) (see Note(2)). -[31:24] 'Unit_Count' specifies the total number of units to be transferred (up to max of 128K bytes). Unit Size is fixed 512bytes. -[23:16] 'MKB_ID' -[15:0] reserved. (This value shall be set to '0' for the future extension.) ([ ] shows bit position of the (essential) argument)

Note:

(1) AKE Commands (ACMD45-48) are always executed in conjunction with either of "Protected Area Access Command" (ACMD18,ACMD25, ACMD26,ACMD38,ACMD49).

(2) In AKE command (ACMD45), challenge1 (random number RN1) is generated by encrypting an (essential) argument of the following "Protected Area Access Command" (shown in Table3.1). SD Memory Card gets the (essential) argument of the following "Protected Area Access Command" by decrypting received challenge1. Regarding the generation method of challenge1, please see 3.2.1 of *Content Protection for Recordable Media Specification SD Memory Card Book*.

(3) Change_Secure_Area command(ACMD49) is restricted to execute as follows:

-The use of this command in end-user application is prohibited.

-The use of this command is allowed only in special authorized applications or devices.
(e.g. manufacturer specific application which is used to make a custom SD-card.)

and the following process shall be executed:

- If the new Protected Area is larger than the former Protected Area, the region of the new Protected Area shall be erased.
- If the new Protected Area is smaller than the former Protected Area, the region of the former Protected Area shall be erased.

(4) It is possible to send ACMD38 (SECURE_ERASE) before

ACMD25(SECURE_WRITE_MULTI_BLOCK) or ACMD26(SECURE_WRITE_MKB) for high-speed purpose.In this case, AKE commands must be done before each secured command (ACMD38, ACMD25, ACMD26) is executed.

(5) The card will send "OUT_OF_RANGE" when the MKB_ID number is bigger than the amount of MKBs saved in the card.

(6) The host shall send the stop transmission command described in the SD-Card Specification Part1 in case that there was an error while Read or Write operation.

(7)In Protected Area Access Command(ACMD18,ACMD25,ACMD38), UNIT_Address starts at "0"(It is not "the top address of User Data Area +1"). And if data accessing is out_of_range, the operation (write,read or erase) will be performed up to the 'end' of range and then indicates 'out_of_range'.

(8) Write Protect Group, Permanent Write Protection and Temp Write Protection(see chapter4.3.5 of SD-Card Specifications Part1:Physical Specifications) do not affect operations on Protected area.

(9) CHANGE_SECURE_AREA command will set WP_VIOLATON flag in Card Status (and change of protected area will not be performed) in case that the card is Permanent Write Protected, Temp Write Protected or if the requested secure area fall into Write Protected Group area.
In case that there is Write Protected area but the new requested Protected Area does not fall into the Write Protected Area then there will not be an error and the new Protected Area shell be defined.

(10) CHANGE_SECURE_AREA command will set LOCK_UNLOCK_FAILED flag in Card Status (and change of protected area will not be performed) in case that the card is LOCKED (with Password). It is always the responsibility of the host to verify successful operation by sending SEND_STATUS (CMD13) command.

(11) As shown in the chapter 3.9.2 of Content Protection for Recordable Media Specification SD Memory Card Book, the data field of Secure_Write_MKB is begun with "Size of MKB" field, followed by "MKB" field , 0 or 4 bytes “0 padding” field, "Kmu" field, 1byte "0 padding" field and "RCC" field.
And to simplify the calculation of RCC, further “0 padding” fields are added in the unit data. Those “0 padding” data has no meaning to RCC value.
Here, in "Size of MKB" field, byte length of MKB shall be stored by big-endian as well as CPRM (which means that byte 0 is a most significant byte).
SECURE_WRITE_MKB data format.

*** 1st unit of 512 bytes contains the following data ***

8 Bytes	Size of MKB (N bytes)
504 Bytes	'0 padding' (*1)

*** A succession of M units of 512 bytes each.(where M is MKB size in units of 512 bytes) ***

N Bytes	MKB data
M * 512 - N Bytes	'0 padding' (*1)

*** (M+2)th unit of 512 bytes data ***

7 Bytes	Kmu
1 Byte	'0 Padding'
8 Bytes	RCC
512 - 16 = 496 Bytes	'0 Padding' (*1)

*** After (M+3)th units are '0 padding' data ***

*1) To simplify the calculation of RCC remaining data in the unit should be "0".

(12) About 'Unit_Count' in the Security R/W/E, GET_MKB and SECURE_WRITE_MKB.

'Unit_Count=0' means 256 units.

3.2 Usage of Security command

Regarding the usage of Security command, please refer to the chapter 3.3 of *Content Protection for Recordable Media Specification SD Memory Card Book*.

3.3 SD Memory Card State Diagram on Authentication

Figure 3.1 shows the SD Memory Card State Diagram on Authentication.

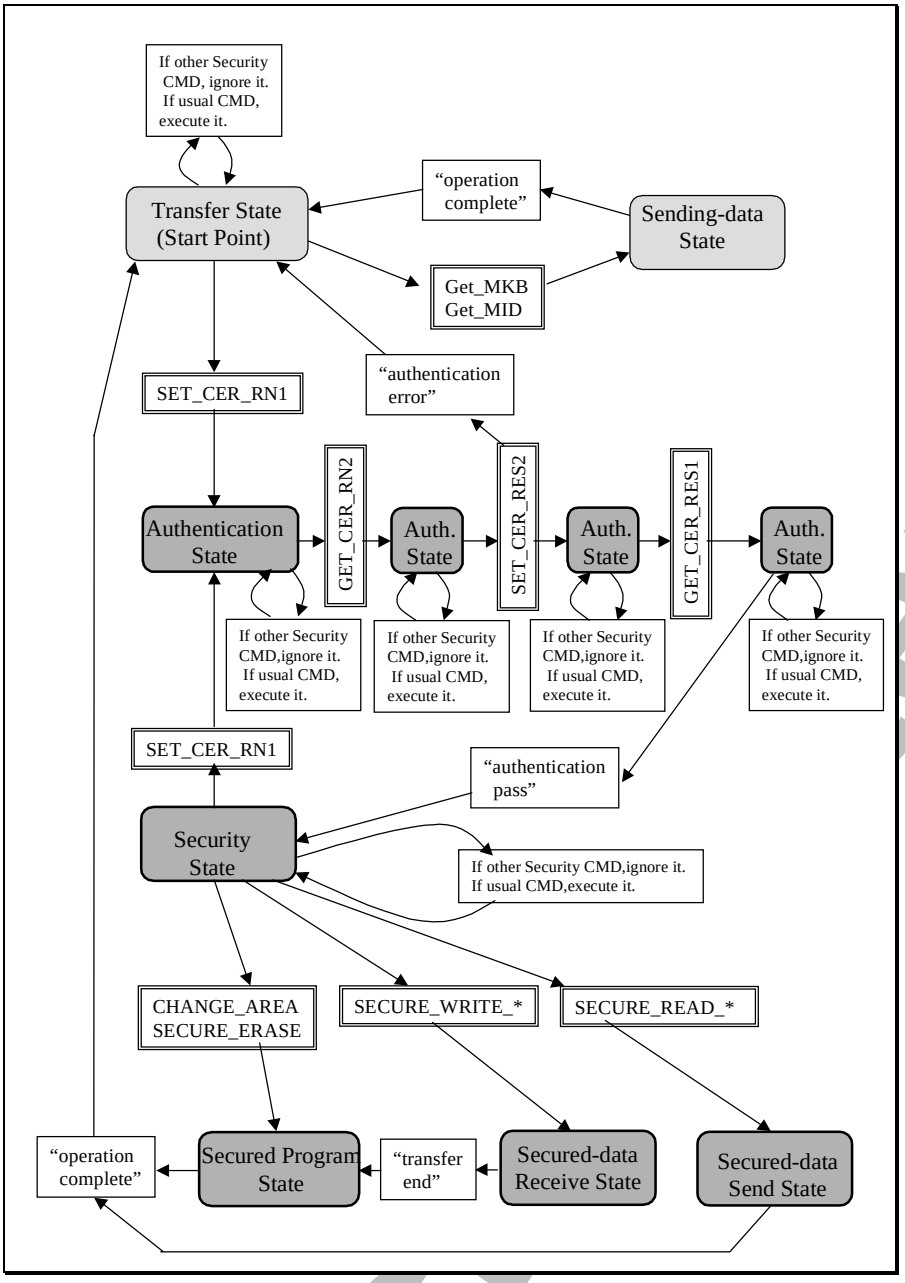


Figure 3.1 State Diagram

4. Random Number Generation(RNG)

In AKE process, SD Memory Card and the accessing device can use the following random number generation scheme. In this scheme, each licensee assigns two 56-bit random numbers as Random Number Key (RNK) pair (c_1, c_2) and pre-stores (c_1, c_2) on Hidden Area of SD Memory Card.

(1) Seed Generation

The 64-bit seed v_t is kept secret in RAM. More concretely, it shall be diffcult to access the seed v_t from outside of SD Memory Card.

The Media Unique Key (56-bit) is used for 56-bit(lsb) of the initial seed v_0 and 8-bit "0" is concatenated as the 8-bit(msb) of the initial seed v_0 when the Card is manufactured. Before shipment, the circuit of RNG freely runs. This makes temporary seed different from K_{mu}.

When the first AKE process is executed after the power of SD Memory Card is turned ON, the seed v_t stored in flash memory is transferred as a temporary seed to RAM. After that, the 64-bit seed (v_t) and 56-bit RNK (c_1) are input to C2 One-way function(C2_G), and 64-bit output (v_{t+1}) of C2_G is stored in flash memory as a next seed (v_{t+1}). Fig. 4.1 shows the procedure of seed generation.

Seed generation is executed only when the first AKE process is executed after the power of SD Memory Card is turned ON.

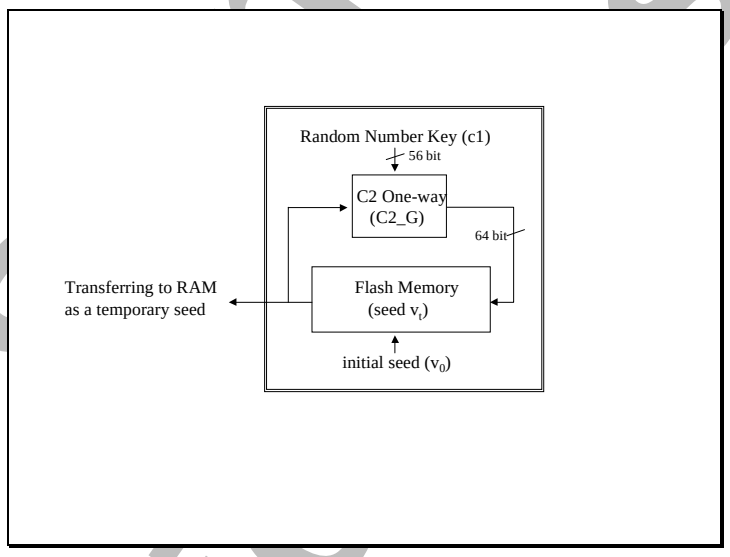


Figure 4.1 Seed Generation

(2) Random Number Generation:

When the first AKE process is executed after the power of SD Memory Card is turned ON, RAM receives the seed from flash memory as a temporary seed (r_{i-1}). After that, the 64-bit temporary seed (r_{i-1}) and 56-bit RNK (c_2) are input to C2 One-way function(C2_G), and 64-bit output (r_i) of C2_G is used as a 64-bit random number and stored in RAM as a next temporary seed(r_i).

Next, Random Number Generation is executed using new temporary seed (r_i). This process is executed repeatedly until the power of SD Memory Card is turned OFF. Fig. 4.2 shows the procedure of random number generation.

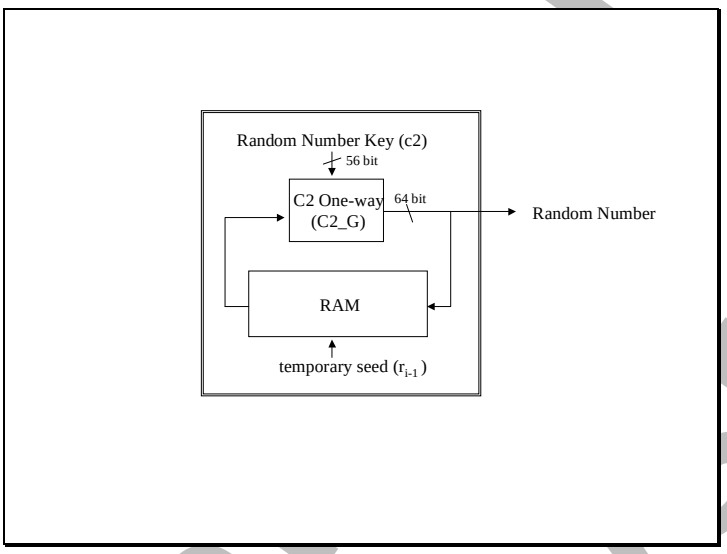


Figure 4.2 Random Number Generation

5. File System

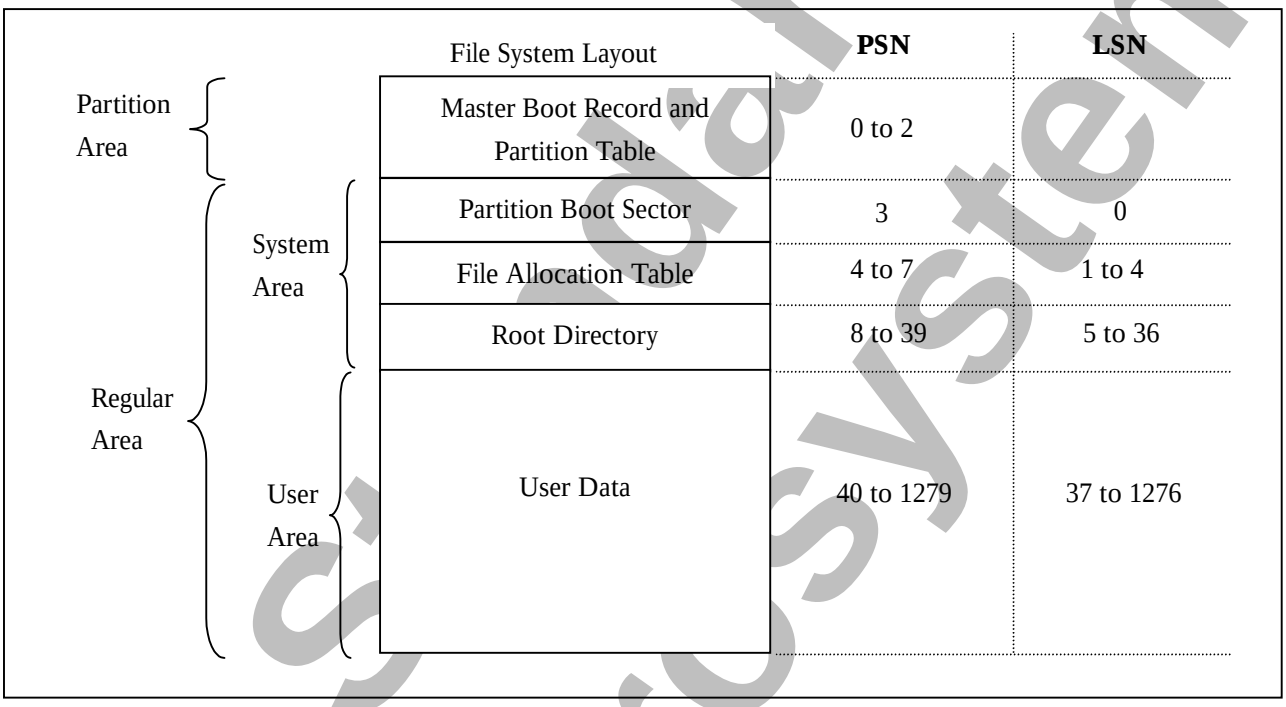
5.1 General

This section defines volume structure regarding Protected Area in SD Memory Card. As well as User Data Area, File System in Protected Area uses ISO/IEC 9293-compliant FAT file system. Furthermore, following is a major point that File System in Protected Area differs from one in User Data Area (defined in SD-Card Specifications Part2: File System Specifications):

- Cluster size of Protected Are is not necessarily a multiple of SD Memory Card erase block size.
- Since it does not require high-speed write for Protected Area, the starting address of User Data is not necessarily located on the boundary of SD Memory Card erase block.

Figure 5.1 shows the example of volume structure for Protected Area.

◊ **Figure 5.1 : Example of Volume Structure for Protected Area**



PSN : Physical Sector Number

LSN : Logical Sector Number

5.2 Master Boot Record and Partition Table

See SD Memory Card Specifications Part2: File System Specifications as reference since this Master Boot Record and Partition Table are defined in the same way as Master Boot Record and Partition Table in Data Area.

5.3 Partition Boot Sector

Since each field in Partition Boot Sector is almost the same as the one in Partition Boot Sector of Data Area, see SD-Card Specifications Part2: File System Specifications as reference. Fields that differ from the ones described in Specifications above mentioned are explained as follows:

(BP13) Sectors per Cluster

This field shall specify the number of sectors per cluster. It shall be recorded the following number: 1, 2, 4, 8, 16, 32 or 64.

5.4 File Allocation Table

See SD-Card Specifications Part2: File System Specifications as reference since this File Allocation Table is defined in the same way as File Allocation Table in User Data Area.

5.5 Root Directory

See SD-Card Specifications Part2: File System Specifications as reference since this Root Directory is defined in the same way as Root Directory in User Data Area.

5.6 User Data

The minimum read/write access size is defined in the units of Sector.

Annex

A Test command Requirement

Each SD Memory card manufacturer can individually define the new test command for setting up and testing or analyzing the devices in SD card. But such defined new test command must comply the following security requirement. CMD60,61,62,63 is reserved for manufacturer for test purpose.

Requirement:

(1) cannot read nor update the data stored in a Hidden Area

(2) cannot update the data stored in a System Area

(3) cannot read nor update the data stored in a Protected Area without successful authentication.

B Sectors per Cluster and Boundary Unit Recommendation for Protected Area

The following table shows the recommendation for Sectors per Cluster and Boundary Unit of Protected Area.

◊ **Table B-1 Sectors per Cluster and Boundary Unit Recommendation (Protected Area)**

Protected Area size	Sectors per Cluster	Boundary Unit
~256KB	1	1
~1MB	2	2
~4MB	8	8
~1024MB	32	32
~2048MB	64	64

NOTE: The Table B-1 is not based on the Card Capacity.

Minimum Protected Area size and format parameters for Protected Area are shown in following table.

◊ **Table B-2 Minimum Protected Area size and format parameters**

Card Capacity	Min Protected Area size(sector)	Sectors per Cluster	an example(values depend on Protected Area size)				
			Clusters	FAT Sec	Hidden	FAT bits	User Data Offset
~4MB	160	1	124	1	1	12	36
~8MB	160	1	124	1	1	12	36
~16MB	320	1	284	1	1	12	36
~32MB	640	2	301	1	3	12	38
~64MB	1280	2	620	2	3	12	40
~128MB	2560	8	314	1	13	12	48
~256MB	5120	8	634	2	11	12	48
~512MB	10240	32	317	1	61	12	96
~1024MB	20480	32	637	2	59	12	96
~2048MB	40960	32	1277	4	55	12	96

However,

Card Capacity…SD Card Capacity.

Min Protected Area size…minimum number of sectors for Protected Area. This parameter is defined from the Card Capacity, using Table B-2.

Sectors per Cluster…number of sectors per cluster. This parameter is defined from the Protected Area size, using Table B-1.

Clusters…number of clusters in User Data. This parameter varies with the Protected Area size.

FAT Sec…number of sectors per FAT. This parameter varies with the Protected Area size.

Hidden…number of sectors existing before Partition Boot Sector. This parameter varies with the Protected Area size.

FAT bits…If the area is formatted with FAT12, FAT bits is 12. And If the area is formatted with FAT16, FAT bits is 16. This parameter varies with the Protected Area size.

User Data Offset…number of sectors existing before the starting sector of User Data.
Sectors per Cluster is defined from the Protected Area size. Clusters, FAT Sec, Hidden, FAT bits, and User Data Offset vary with the Protected Area size (Use the parameters in Table B-1 for calculation).

C Type of 16 MKBs on SD Memory Card

This annex shows the type of 16 MKBs pre-stored in the SD Memory Card. As shown in Table C-1, first eight MKBs (MKB "#0" to MKB "#7") are read-only and not updateable by the "dynamic MKB update scheme". Here, the "dynamic MKB update scheme" is described in Chapter 3.9 of *Content Protection for Recordable Media Specification SD Memory Card Book*. Next seven MKBs (MKB "#8" to MKB "#14") are updateable MKB by using the "dynamic MKB update scheme". A last MKB (MKB "#15") is a master MKB which is used in a special authorized accessing device (e.g., a Kiosk), which is allowed to execute the "dynamic MKB update scheme". MKB "#0" is used in SD-Audio application. MKB "#1" to MKB "#14" are reserved for other applications. Application which uses reserved MKB (MKB "#1" to MKB "#14") is added in the future.

◊ **Table C-1 Type of 16MKBs**

Number of MKB	Type	Application
MKB0	Read only	SD-Audio
MKB 1	Read only	Reserved
MKB 2	Read only	Reserved
MKB 3	Read only	Reserved
MKB 4	Read only	Reserved
MKB 5	Read only	Reserved
MKB 6	Read only	Reserved
MKB 7	Read only	Reserved
MKB 8	Updateable	Reserved
MKB 9	Updateable	Reserved
MKB 10	Updateable	Reserved
MKB 11	Updateable	Reserved
MKB 12	Updateable	Reserved
MKB 13	Updateable	Reserved
MKB 14	Updateable	Reserved
MKB 15	Master	Reserved