SD Specifications
Part 1
PHYSICAL LAYER

Simplified Specification

Version 1.10

March 18, 2005

SD Group
Matsushita Electric Industrial CO., Ltd.
SanDisk Corporation
Toshiba Corporation

SD Card Association
Technical Committee

Conditions for publication

Publisher:

SD Card Association
719 San Benito St., Suite C
Hollister, CA 95023 USA
Phone: +1 831 636 7322
Fax: +1 831 623 2248
E-mail: president@sdcard.org

Copyright Holders:

SD Group
Matsushita Electric Industrial Co. ,Ltd. (MEI)
SanDisk Corporation (SanDisk)
Toshiba Corporation (Toshiba)
SD Card Association

Note:
The copyright of all previous versions (Version 1.00 and 1.01) and all corrections or non-material changes thereto are owned by SD Group. The copyright of material changes to the previous versions (Version 1.01) are owned by SD Card Association.

Exemption:

The information contained herein is presented only as a standard specification for SD Card and SD Host products. No responsibility is assumed by SD Group and SD Card Association for any damages, any infringements of patents or other right of the third party, which may result from its use. No license is granted by implication or otherwise under any patent or rights of SD Group and SD Card Association or others.

Revision History

Date	Version	Changes compared to previous issue
March 18, 2005	1.10	Base version initial release.

Table of Content

Table of Figures

Table of Tables

1. General description

SD Memory Card is a 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.

SD memory cards may also support a second security system based on commonly used standards, such as ISO-7816, which can be used to interface the SD memory card into public networks and other systems supporting mobile e-commerce and digital signature applications.

The physical form factor, pin assignment and data transfer protocol are forward compatible with the MultiMediaCard[1] with some additions.

In addition to the SD Memory Card there is SD I/O (SDIO) Card. The SDIO Card specification is defined in separate specification named: "SDIO Card Specification" that can be obtained from the SD Association. The SDIO Specification defines SD card that may contain interfaces between various IO units and SD Host. The SDIO card may contain memory storage capability as well as its IO functionality. The Memory portion of SDIO card shall be fully compatible to the given SD Memory Card specification. The SDIO card is based on and compatible with the SD Memory card. This compatibility includes mechanical, electrical, power, signalling and software. The intent of the SD I/O card is to provide high-speed data I/O with low power consumption for mobile electronic devices. A primary goal is that an I/O card inserted into a non-SDIO aware host will cause no physical damage or disruption of that device or it's software. In this case, the I/O card should simply be ignored. Once inserted into an SDIO aware host, the detection of the card will be via the normal means described in the given SD Physical Specification with some extensions that are described in the SDIO Specification.

The SD Memory Card communication is based on an advanced 9-pin interface (Clock, Command, 4xData and 3xPower lines) designed to operate in at maximum operating frequency of 50MHz and 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.

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

[1] The SD Memory card is specified based on MultiMediaCard Version 2.11. The term of MultiMediaCard in this specification shall be interpreted to Version 2.11 only. Compatibility to the later Version 2.11 should be established on own responsibility of the Manufacturers of Hosts or Ancillary Products.

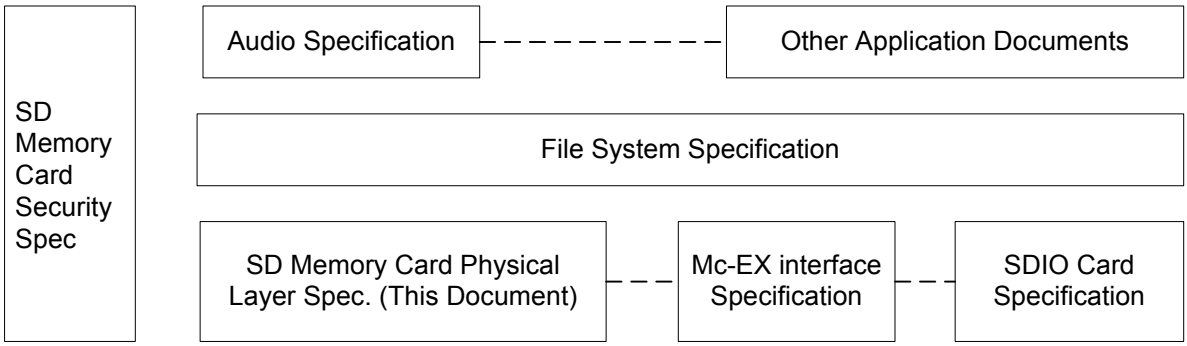


Figure 1: SD Memory Card Documentation Structure

- **SD Memory Card Audio 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 7 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 implementer.

- **Mc-EX Interface Specification:** (This section is newly added in version 1.10)

Part A1 of the SD memory card specification (refer to Figure 1) serves as an extension to the SD card Physical Layer Specification and provides all the definitions required to transfer the Mobile Commerce Extension (Mc-EX) command packets from the Mc-EX host to the Mc-EX enabled SD memory card, and vice versa.

2. System features

- **Targeted for portable and stationary applications**
- **Voltage range:**
  SD Memory Card -
  Basic communication (CMD0, CMD15, CMD55, ACMD41): 2.0 – 3.6V[1]
  Other commands and memory access: 2.7 - 3.6V
  SDLV Memory Card (low voltage) - Operating voltage range: 1.6 – 3.6V[2]
- **Designed for read-only and read/write cards.**
- **Default mode: Variable clock rate 0 - 25 MHz , up to 12.5MB/sec interface speed (using 4 parallel data lines)**
- **High-Speed mode: Variable clock rate 0 - 50 MHz , up to 25MB/sec interface speed (using 4 parallel data lines)**
- **Switch function command supports High-Speed, eCommerce and future functions**
- **Correction of memory field errors**
- **Card removal during read operation will never harm the content**
- **Forward compatibility**[3] **to MultiMediaCard Version 2.11**
- **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:**

[1] It is highly recommended not to use voltage range 2.0-2.7 for basic communication

[2] It is highly recommended not to use voltage range 1.6-2.7

[3] Compatibility means SD host can support both SD memory card and MultiMediaCard Version 2.11

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

- **SD Memory Card thickness is defined as either 2.1mm (normal) and 1.4mm (Thin SD Memory Card) .**

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.

In terms of capacity, two types of SD Memory Cards are defined:

- Standard capacity, SD Memory Cards with up to 2G bytes (2^{31} bytes) of memory. Part 1 Physical Layer Specification versions 1.01 and 1.10 are defining the standard capacity cards.
- High capacity, SD Memory Cards with more than 2G bytes (2^{31} bytes) of memory. Part 1 Physical Layer Specification version 2.0 will define the high capacity 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. In High-Speed mode, only one card can be connected to the bus.

3.1.1 SD bus

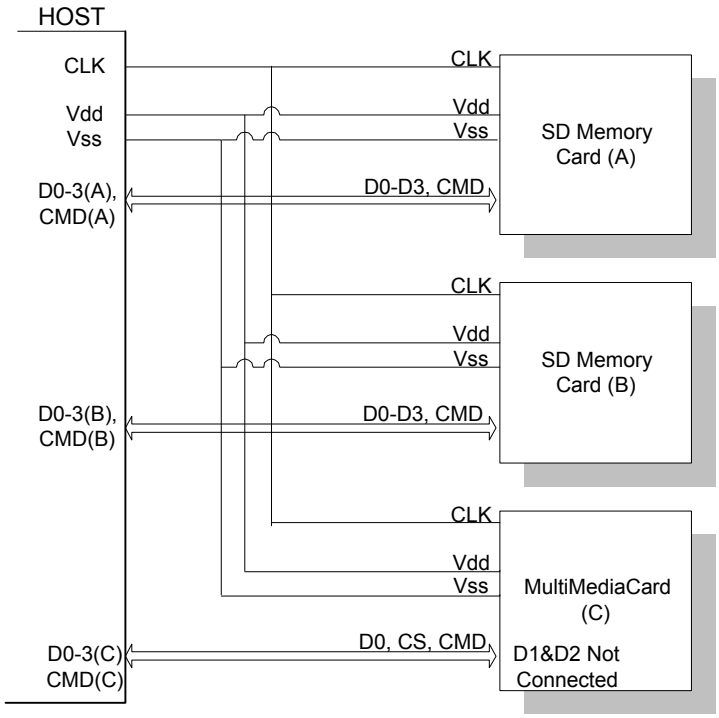


Figure 2: SD Memory Card system bus Topology

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. **Note that while DAT1-DAT3 are not in use, the related Host's DAT lines should be in tri-state (input mode). For SDIO cards DAT1 and DAT2 are used for signaling.**

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.

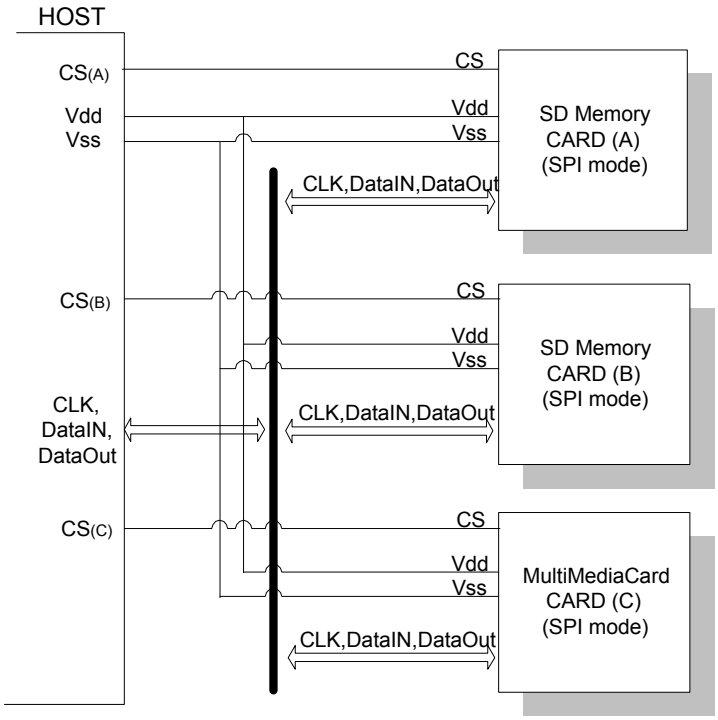


Figure 3: SD Memory Card system (SPI mode) bus topology

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.

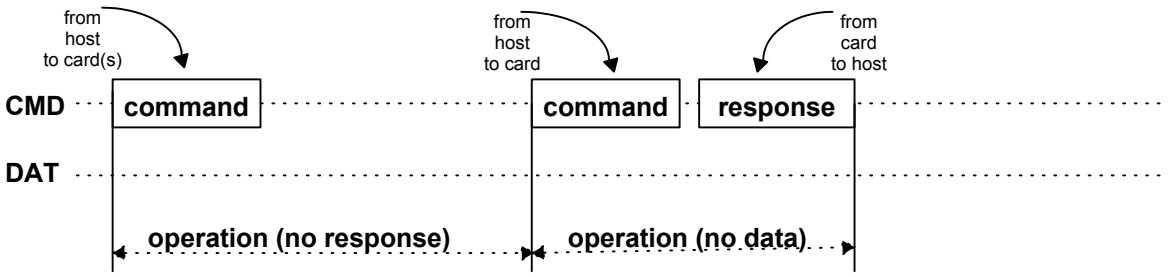


Figure 4: "no response" and "no data" operations

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.

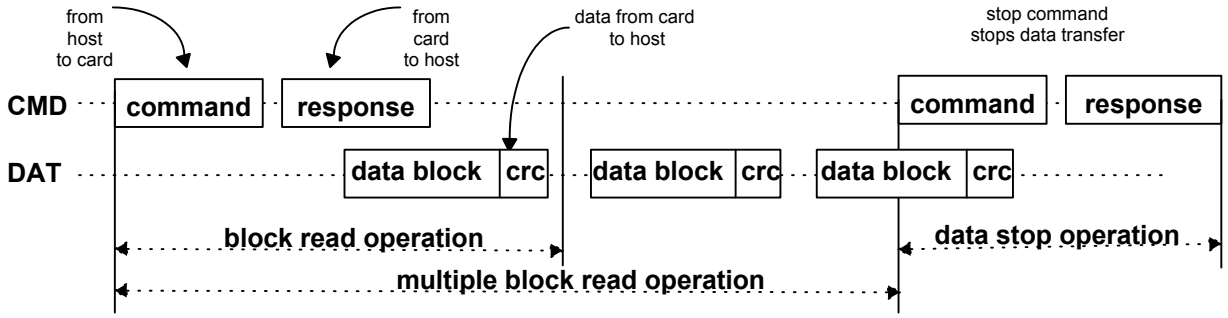


Figure 5: (Multiple) Block read operation

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.

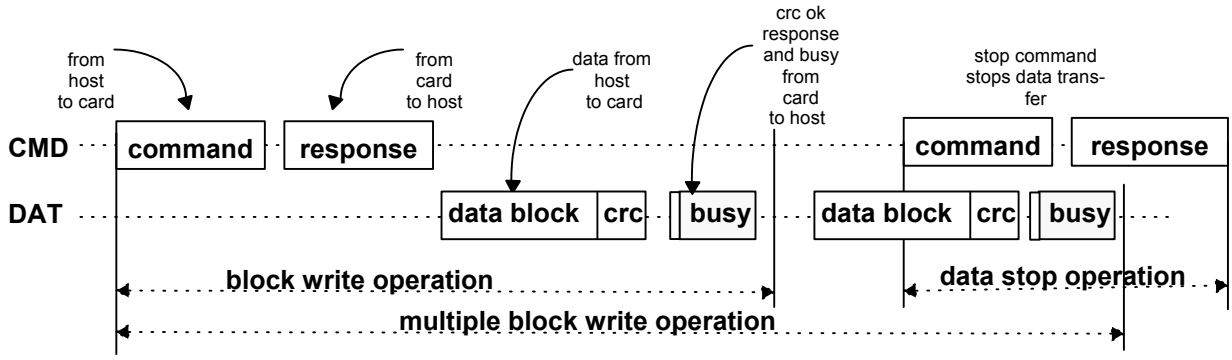


Figure 6: (Multiple) Block write operation

Command tokens have the following coding scheme:

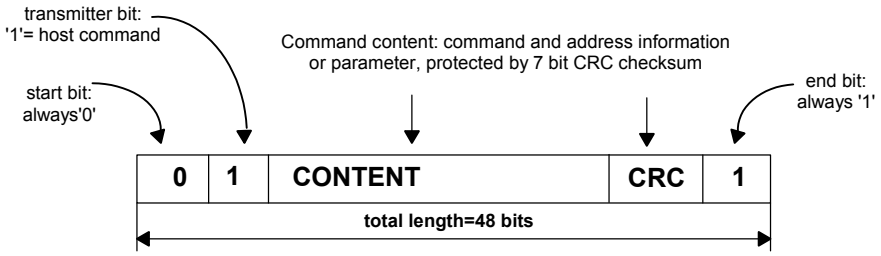


Figure 7: Command token format

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.

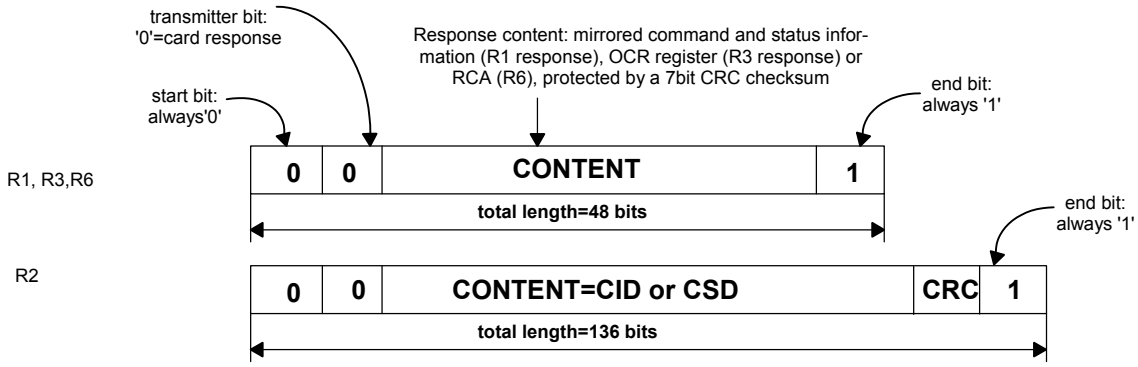


Figure 8: Response token format

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

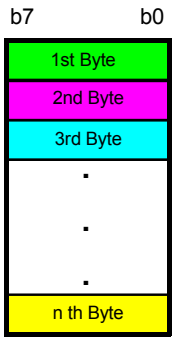
when the wide bus option is used, the data is transferred 4 bits at a time (refer to Figure 10). Start and end bits, as well as the CRC bits, are transmitted for every one of the DAT lines. CRC bits are calculated and checked for every DAT line individually. The CRC status response and Busy indication will be sent by the card to the host on DAT0 only (DAT1-DAT3 during that period are don't care).

There are two types of Data packet format for the SD card.

(1) Usual data (8 bit width)The usual data (8 bit width) are sent in LSB (Least Significant Byte) first, MSB (Most Significant Byte) last manner. But in the individual byte it is MSB (Most Significant Bit) first, LSB (Least Significant Bit) last.

(2) Wide width data (SD Memory Register)The wide width data is shifted from MSB bit.

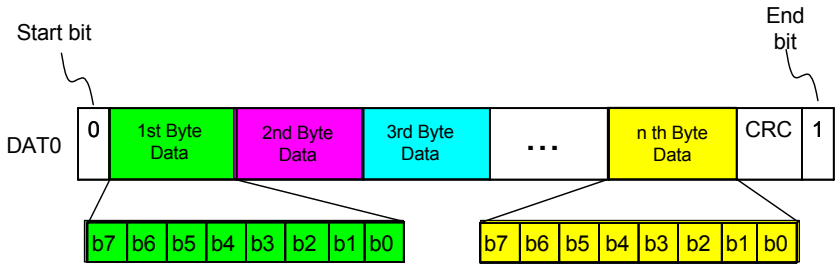
1. Data packet format for usual data (8bit width)



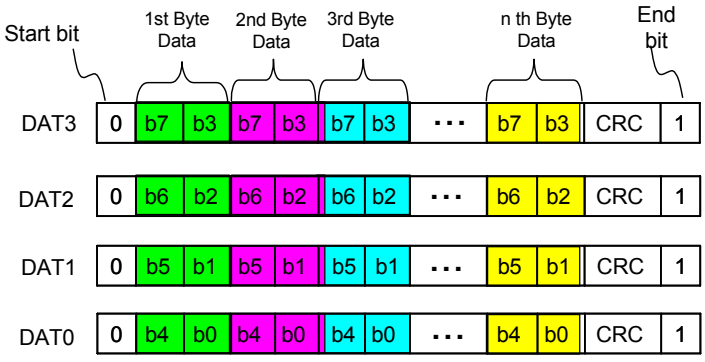
8bit width Data

Ex

[SDIO]
CMD53
[SD memory]
CMD17, CMD18,
CMD24, CMD25,
ACMD18, ACMD25,
etc



Data packet format for Standard bus (only DAT0 used)



Data packet format for Wide bus (all four lines used)

Figure 9: Data packet format - Usual data

2. Data packet format for wide width data (Ex. ACMD13)

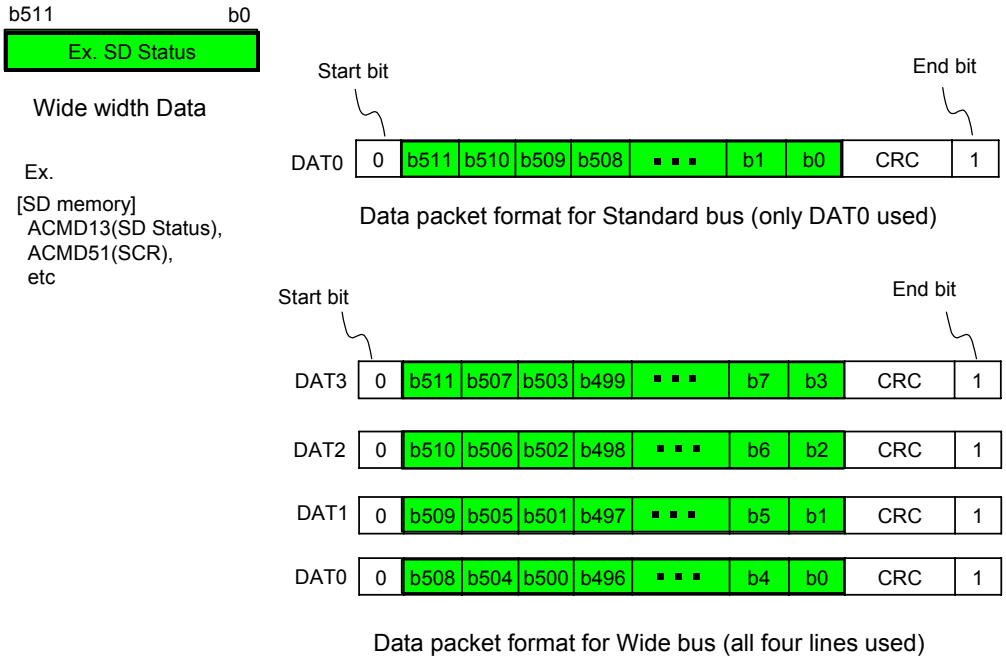


Figure 10: Data packet format - Wide width data

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. 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.

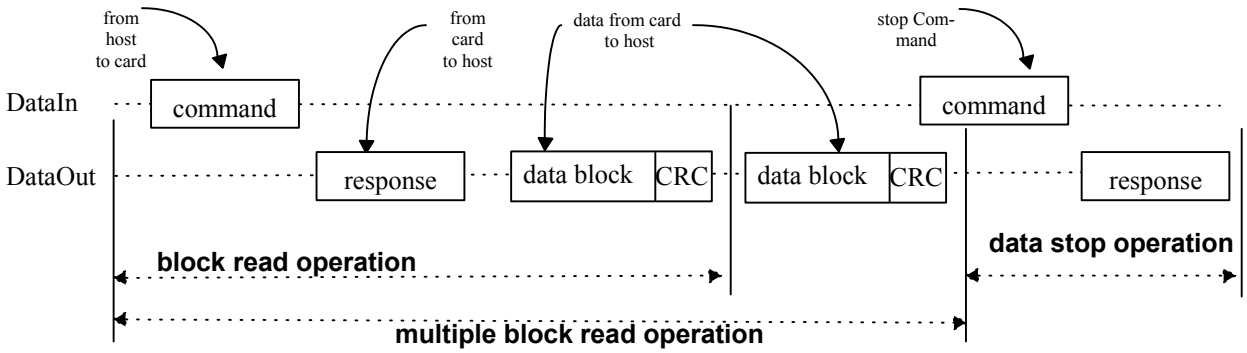


Figure 11: Read operation

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 12 shows a data read operation which terminated with an error token rather than a data block.

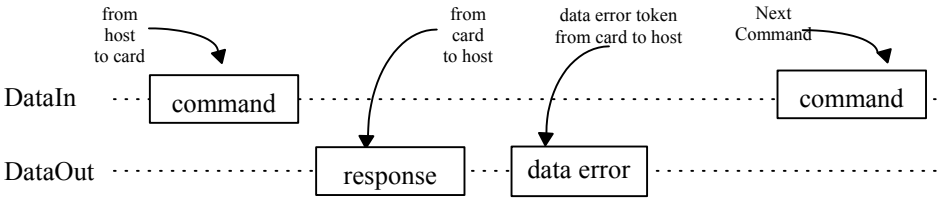


Figure 12: Read operation - data error

- **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 13).

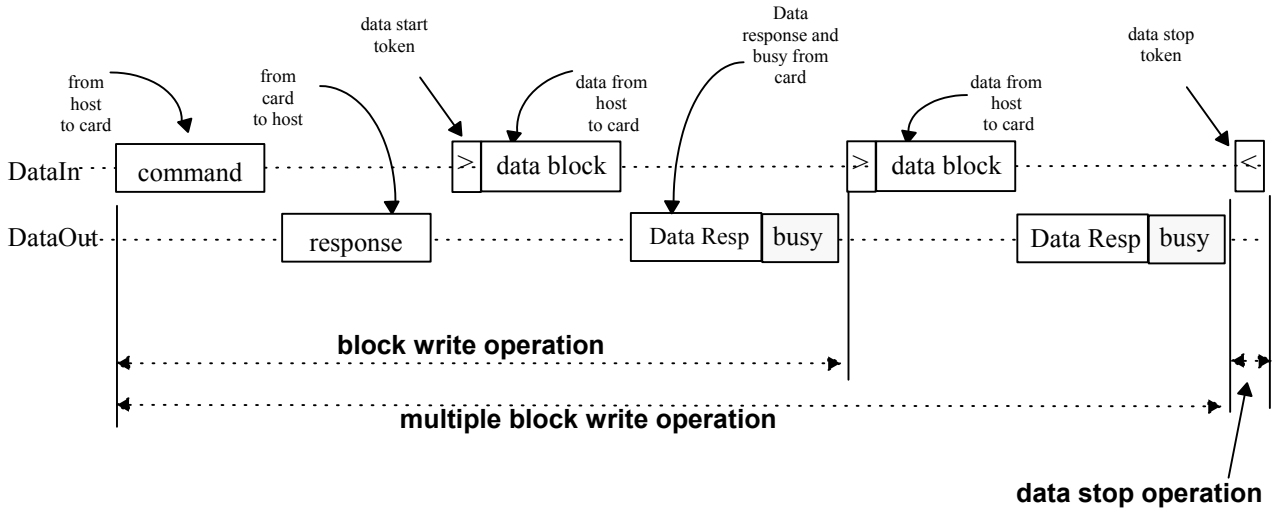


Figure 13: Write operation

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 Registers

Each card has a set of information registers (see also Chapter 5 in the Physical Layer Specification Version 1.10):

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.**

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

Table 1: SD Memory Card registers

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 14 for the card identification mode and in Figure for the data transfer mode. The commands are listed in the command tables (Table 14-Table 22). The dependencies between current state, received command and following state are listed in Table 24. 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 14) 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 2: 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. 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.

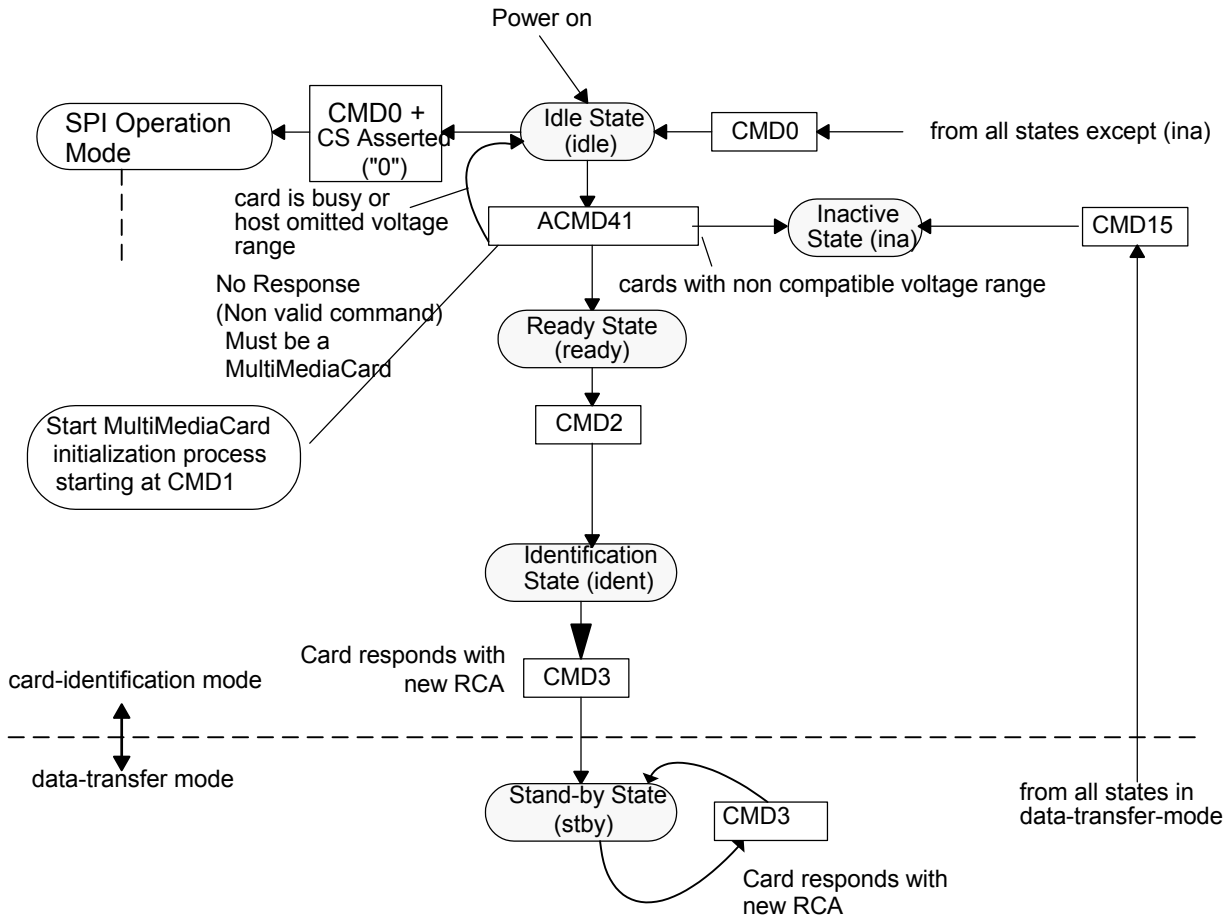


Figure 14: SD Memory Card state diagram (card identification mode)

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} (Maximum 400KHz). 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 end of Card Identification Mode the host must remain at f_{OD} frequency because some cards may have operating frequency restrictions during the card identification mode. In Data Transfer Mode the host may operate the card in f_{PP} frequency range (Up to 25MHz in default mode and up to 50MHz in high speed mode). The host issues SEND_CSD (CMD9) to obtain the Card Specific Data (CSD register), e.g. block length, card storage capacity, etc.

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* (Note that it is the responsibility of the Host to reserve the RCA=0 for card de-selection - refer to Table 14, CMD7

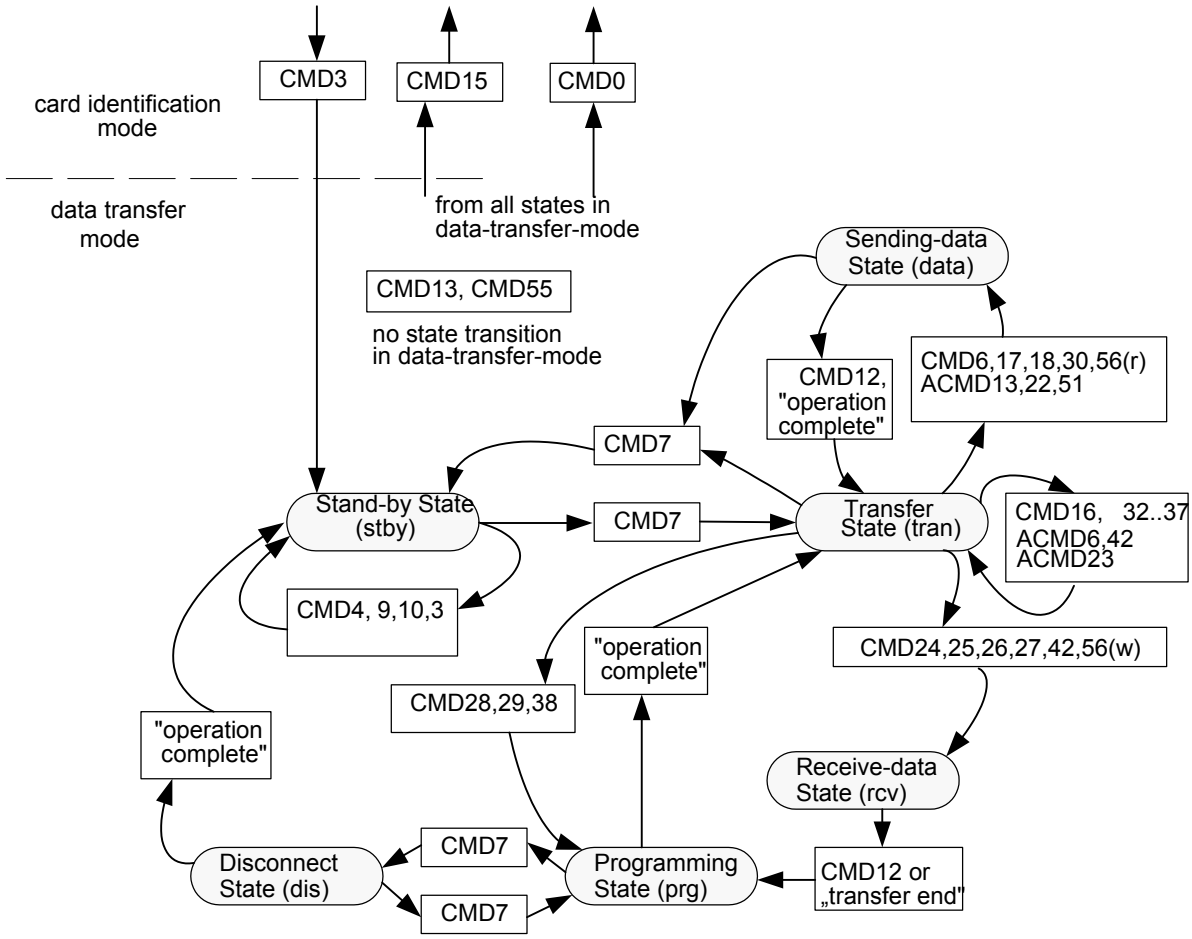


Figure 15: SD Memory Card state diagram (data transfer mode)

description). 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 (ACMD41, CMD2, 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 necessity 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.

- 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 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), the DAT0 line will be kept low (BUSY).
- There is no buffering option for write CSD, 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 2GByte Card

To make 2GByte card, the Maximum Block Length (READ_BL_LEN=WRITE_BL_LEN) shall be set to 1024 bytes. But Block Length set by CMD16 shall be up to 512 bytes to keep consistency with 512 bytes Maximum Block Length cards (Less than and equal 2GByte cards).

4.3.3 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 28-Figure 30). 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.

Read command is rejected if BLOCK_LEN_ERROR or ADDRESS_ERROR occurred and no data transfer is performed

- **Block Read**

Block read is block oriented data transfer. The basic unit of data transfer is a block whose maximum size is always 512 bytes. Smaller blocks whose starting and ending address are entirely contained within 512 bytes boundary may be transmitted.

Block Length set by CMD16 can be set up to 512 bytes regardless of READ_BL_LEN.

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. The following table defines the card behavior when partial block accesses is enabled.

CSD value			Current Blocklen *1	Read CMD Start Address
Max block size READ_BL_LEN	Misalign	Partial		
512Bytes	0 (Disable)	1 (Enable)	1- 512 bytes	Any address is forgiven. *2
1kBytes	0 (Disable)	1 (Enable)	1- 512 bytes	Any address is forgiven. *2

2kBytes	0 (Disable)	1 (Enable)	1- 512 bytes	Any address is forgiven. *2

Table 3 - Read command blocklen

*1 : “Current Blocklen" size is set or changed by CMD16. If value is less than or equal 512 bytes (there are no relations with Misalign and Partial option), it is set with no error.

*2 : When the Blocklen size data range crosses 512 bytes block boundary, card outputs the data until the 512 bytes block boundary” and then the data becomes invalid and CRC error also may occur. The card will send “ADDRESS_ERROR" on the next command response. Host should issue CMD12 to recover.

4.3.4 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 4.5) for each received data block prior to the write operation. By this mechanism, writing of erroneously transferred data can be prevented.

Write command is rejected if BLOCK_LEN_ERROR or ADDRESS_ERROR occurred and no data transfer is performed

- **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 be required that Block Length set by CMD16 shall be 512 bytes regardless of WRITE_BL_LEN is set to 1k or 2k bytes.

The following table defines the card behavior when partial block accesses is disabled (WRITE_BL_PARTIAL = 0).

CSD value			Current Blocklen *1	Write CMD Start Address
Max block size WRITE_BL_LEN	Misalign	Partial		
512Bytes	0 (Disable)	0 (Disable)	512 bytes *2	n * 512 bytes *3 (n : Integer)
1kBytes	0 (Disable)	0 (Disable)	512 bytes *2	n * 512 bytes *3 (n : Integer)
2kBytes	0 (Disable)	0 (Disable)	512 bytes *2	n * 512 bytes *3 (n : Integer)

Table 4 - Write command blocklen

*1 : “Current Blocklen” size is set or changed by CMD16. If value is less than 512 bytes (there are no relations with Misalign and Partial option), it is set with no error. And then “Current Blocklen” size is tested when write command execution.

*2 : If the current Blocklen is other than this value, the card indicates “BLOCK_LEN_ERROR” on the Write command response.

*3 : If start address is other than this value, the card will send “ADDRESS_ERROR” on the Write command response.

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 misalignment 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 CSD register does not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD 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 inter-leaving 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 Application 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.

4.3.5 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_BL_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'.

4.3.6 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 7) 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.

- **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-protect 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.

4.3.7 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, CMD16 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 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 5: Lock card data structure

- **ERASE:** '1' Defines Forced Erase Operation. In byte 0 bit 3 will be set to "1" (all other bits shall be '0'). All other bytes of this command will be ignored by the card.
- **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. The password length is up to 16 bytes. In case of password change the total length of the old password and the new password can be up to 32 bytes.

- **Password data:** In case of set new password, it contains the new password. In case of password change, it contains the old password followed by new password.

The data block size shall be defined by the host before it sends the card lock/unlock command. The block length shall be set to greater than or equal required data structure of lock/unlock command. In the following explanation, changing block size by CMD16 is not mandatory requirement for the lock/unlock command.

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 password data 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 subtracting 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 the sent old password is correct (equal in size and content) then the given new password and its size will be saved in the PWD and PWD_LEN registers, 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 itself . 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 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, unless it was done during a password definition or change operations.

- **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 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, unless it was done during a password definition or change operations.

- **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 set in the data field, 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.

- **Parameter and the Result of CMD42:**

The block length shall be greater than or equal required data structure of CMD42; otherwise, the result of CMD42 is undefined and the card may be in the unexpected locked state. Table 6 clarifies the behavior of CMD42. The reserved bits in the parameter (bit7-4) of CMD42 shall be don't care. In the case of CMD42 requires the password, it is assumed that the old password and the new password are set correctly; otherwise the card indicates error regardless of Table 6. If the password length is 0 or greater than 128 bits, the card indicates error. If errors occur during execution of CMD42, the LOCK_UNLOCK_FAILED (Bit24 of Card Status) shall be set to 1 regardless of Table 6. The CARD_IS_LOCKED (Bit25 of Card Status) in the response of CMD42 shall be the same as Current Card State in Table 6. In the field of Card Status, "0" to 1" means the card change to Locked and "1 to 0" means the card change to Unlocked after execution of CMD42. It can be seen in the response of CMD13 after the CMD42. The LOCK_UNLOCK_FAILED (Bit24 of Card Status) as the result of CMD42 can be seen in the response of either CMD42 or following CMD13.

CMD42 Parameter in the data
Bit3: ERASE
Bit2: LOCK_UNLOCK
Bit1: CLR_PWD
Bit0: SET_PWD

Related bits in the Card Status
Bit25: CARD_IS_LOCKED
Bit24: LOCK_UNLOCK_FAILED

CMD42 Parameter				Current Card State	PWD_LEN and PWD	Result of the Function	Card Status	
Bit3	Bit2	Bit1	Bit0				Bit25	Bit24
After Power On					Exist	The card is locked	1	0
					Cleared	The card is unlocked	0	0
1	0	0	0	Locked	Exist	Force Erase (Refer to Table 7)	Table 7	Table 7
1	0	0	0	Unlocked	Exist	Error	0	1
1	0	0	0	Unlocked	Cleared	Error	0	1
0	1	0	0	Locked	Exist	Error	1	1
0	1	0	0	Unlocked	Exist	Lock the card	0 to 1	0
0	1	0	0	Unlocked	Cleared	Error	0	1
0	1	0	1	Locked	Exist	Replace password and the card is still locked	1	0
0	1	0	1	Unlocked	Exist	Replace password and the card is locked	0 to 1	0
0	1	0	1	Unlocked	Cleared	Set Password and lock the card	0 to 1	0
0	0	1	0	Locked	Exist	Clear PWD_LEN and PWD and the card is unlocked	1 to 0	0
0	0	1	0	Unlocked	Exist	Clear PWD_LEN and PWD	0	0
0	0	1	0	Unlocked	Cleared	Error (Note *4 Refer to Table 9)	0	1
0	0	0	1	Locked	Exist	Replace password and the card is unlocked	1 to 0	0
0	0	0	1	Unlocked	Exist	Replace password and the card is unlocked	0	0
0	0	0	1	Unlocked	Cleared	Set password and the card is still unlocked	0	0
0	0	0	0	Locked	Exist	Unlock the card	1 to 0	0
0	0	0	0	Unlocked	Exist	Error	0	1
0	0	0	0	Unlocked	Cleared	Error	0	1
Other combinations				Don't care	Don't care	Error (Note *1 Refer to Table 9)	0 or 1	1

Table 6 : Lock Unlock Function (Basic Sequence for CMD42)

Application Note:
To replace password, the host should consider following cases. When PWD_LEN and password data exist, the card assumes old and new passwords are set in the data structure. When PWD_LEN and PWD are cleared, the card assumes only new password is set in the data structure. In this case, the host shall not set old password in the data structure; otherwise, unexpected password is set.

- **Force Erase Function to the Locked Card:**

Table 7 clarifies the relation between force erase and Write Protection. The force erase does not erase the secure area. The card shall keep locked state during the erase execution and change to unlocked state after the erase of all user area is completed. Similarly, The card shall keep Temporary and Group Write Protection during the erase execution and clear Write Protection after the erase of all user area is completed. In the case of erase error occur, the card can continue force erase if the data of error sectors are destroyed.

Write Protections
PWP: Permanent Write Protect (CSD Bit13)
TWP: Temporary Write Protect (CSD Bit12)
GWP: Group Write Protect (CMD28, CMD29, CMD30)

CMD42 Parameter				PWP	TWP	Result of the Function	Card Status	
Bit3	Bit2	Bit1	Bit0		GWP		Bit25	Bit24
1	0	0	0	Yes	don't care	Error (Note *2 Refer to Table 9)	1	1
1	0	0	0	No	Yes	Execute force erase and clear Temporary Write Protect and Group Write Protect. (Note *3 Refer to Table 9)	1 to 0	0
1	0	0	0	No	No	Execute force erase.	1 to 0	0

Table 7 : Force Erase Function to the Locked card (Relation to the Write Protects)

- **Relation Between ACMD6 and Lock / Unlock State:**

ACMD6 is rejected when the card is locked and bus width can be changed only when the card is unlocked.

Table 8 shows the relation between ACMD6 and Lock / Unlock state.

Card State	Bus mode	Result of the Function
Unlocked	1-bit mode	ACMD6 is accepted
Locked	1-bit mode	ACMD6 is rejected and still in 1-bit mode
Unlocked	4-bit mode	ACMD6 is accepted
Locked	4-bit mode	ACMD6 is rejected and still in 4-bit mode. CMD0 change to 1-bit mode

Table 8 : Relation Between ACMD6 and Lock / Unlock State

Application Note: After power on (in 1-bit mode), if the card is locked, the SD mode host shall issue CMD42 in 1-bit mode. If the card is locked in 4-bit mode, the SD mode host shall issue CMD42 in 4-bit mode.

- **Commands accepted for Locked card:**

The locked card shall accept commands listed below and return response with setting CARD_IS_LOCKED.

1) Basic class (0)
2) Lock card class (7)
3) CMD16
4) ACMD41
5) ACMD42

All other commands including security commands are treated as illegal commands.

Application Note: After power on, the host can recognize the card lock/unlock state by the CARD_IS_LOCKED in the response of CMD7 or CMD13.

- **Two types of Lock / Unlock Card:**

There will be two types of lock / unlock function-supported card. The Type 1 is earlier version of SD memory card and the Type 2 is new version defined in this version of the specification. Table 9 shows the difference between these types of card. The SD memory cards that support Lock / Unlock and comply with Version 1.01, can take either Type 1 or Type 2. The SD memory cards that support Lock / Unlock and comply with Version 1.10, shall take Type 2.

Notes	Type 1 Card (Earlier version)	Type 2 Card (New version)
*1 in Table 6	Treat CMD42 Parameter=0011b as 0001b. Treat CMD42 Parameter=0111b as 0101b. Treat CMD42 Parameter=0110b as 0010b. Results of other combinations are Error.	All results are Error
*2 in Table 7	Execute force erase and set Permanent Write Protect. If force erase is completed, the CARD_IS_LOCKED is changed from 1 to 0. A priority is given to force erase from Permanent Write Protect.	The result is Error A priority is given to Permanent Write Protect from force erase.
*3 in Table 7	Execute force erase but Temporary Write Protect and Group Write Protect are not cleared. It is in need of the host clear.	Execute force erase and clear Temporary Write Protect and Group Write Protect.
*4 in Table 6	CMD42 Parameter=0010 and CMD42 Parameter=0110 The result is no error. Card status Bit24 will be 0	The result is Error. Card status Bit24 will be 1

Table 9 : Difference of earlier version and new version of lock / unlock function

Application Note:
The host can use both types of card without checking difference by taking account of following points.
(1) The host should not set the parameters of CMD42 that return error in Table 6. (For *1)
(2) The host should not issue force erase command if the Permanent Write Protect is set to 1, otherwise the Type 1 card cannot be used any more even if the user remembers the pass word. (For *2)
(3) After the force erase, if the Temporary Write Protect is not cleared, the host should clear it. (For *3)

4.3.8 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.

4.3.9 Application specific commands:

The SD Memory Card is defined to be protocol forward compatible to the MultiMediaCard Version 2.11 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.

4.3.10 Switch function command (This chapter is newly added in version 1.10)

Switch function command (CMD6)[1] is used to switch or expand memory card functions. Currently there are two function groups defined:

- Card access mode - 12.5MB/sec interface speed (default) or 25MB/sec interface speed. (high-speed)
- Card command system - Standard command set (default) or eCommerce command set or Vendor Specific Command set.

This is a new feature, introduced in SD physical Layer Specification Version 1.10. Therefore, cards that are compatible with earlier versions of the spec do not support it. The host shall check the "SD_SPEC" field in the SCR register to recognize what version of the spec the card complies with before using CMD6. It is mandatory for SD memory card of Ver1.10 to support CMD6.

CMD6 is valid under the "Transfer State". Once selected, via the switch command, all functions only return to the default function after a power cycle, CMD6 (Mode 1 operation with Function 0 in each function group) or CMD0. Executing a power cycle or issuing CMD0, will cause the card to reset to the "idle" state and all the functions to switch back to the default function.

As a response to CMD6, the SD Memory Card will send R1 response on the CMD line, and 512 bits of status on the DAT lines. From the SD bus transaction point of view, this is a standard single block read transaction and the time out value of this command is 100ms, same as in read command. If CRC error occurs on the status data the host should issue a power cycle.

CMD6 function switching period is within 8 clocks after the end bit of status data. When CMD6 changes the bus behavior (i.e. access mode) the host is allowed to use the new functions

1. CMD6 is defined for memory card. SDIO card will use CCCR to switch functions.

(increase/decrease CLK frequency beyond the current max CLK frequency), at least 8 clocks after at the end of the switch command transaction (see Figure 16).

In response to CMD0, the switching period is within 8 clocks after the end bit of CMD0. When CMD6 has changed the bus behavior (i.e. access mode) the host is allowed to start the initialization process, at least 8 clocks after at the CMD0.

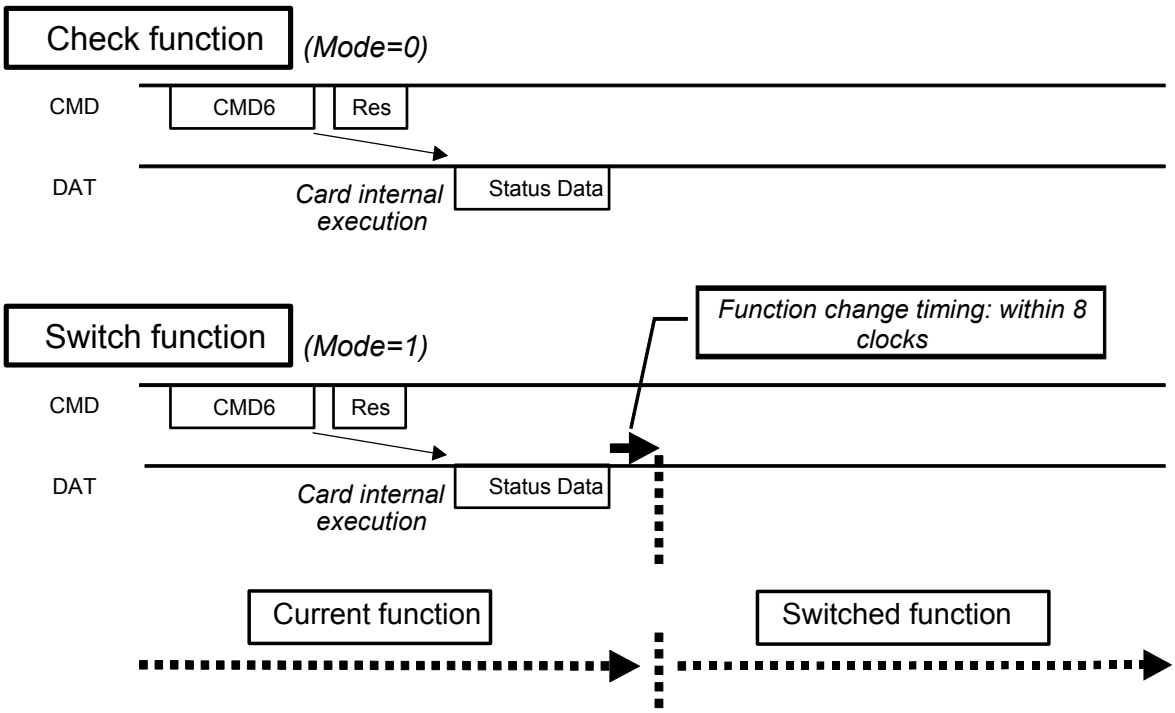


Figure 16: Usage of Switch command

CMD6 supports six function groups, and each function group supports sixteen branches (functions). Only one function can be chosen and active in a given function group. Function 0 in each function group is the default function (compatible with Spec. 1.01).

CMD6 can be used in two modes:

- Mode 0 (Check function) is used to query if the card supports a specific function or functions.
- Mode 1 (set function) is used to switch the functionality of the card.

- **Mode 0 Operation - Check Function**

CMD6 is used in mode 0 to query which functions the card supports, and to identify the maximum current consumption of the card under the selected functions.

Refer to Table 23: Switch function commands (class 10) for the argument definition of CMD6.

A query is done by setting the argument field of the command, as follows:

- Setting the Mode bit to "0"
- Selecting only one function in each function group. Selection of default function is done by setting the function to 0x0. Selecting a specific function is done by using appropriate values from Table 10. Selecting 0xF will keep the current function that has been selected for the function group.

In response to a query, the switch function will return the following 3 statuses (see Table 11):

- The functions that are supported by each of the function groups
- The function that the card will switch to, in each of the function groups. This value is identical to the provided argument if the host made a valid selection or 0xF if the selected function was invalid.
- Maximum current consumption under the selected functions. If one of the selected functions was wrong the return value will be 0.

- **Mode 1 Operation - Set Function**

CMD6 is used in mode 1 to switch the functionality of the card.

Switching to a new functionality is done by:

- Setting the Mode bit to "1"
- Selecting only one function in each function group. Selection of default function is done by setting the function to 0x0. It is recommended to specify 0xF (no influence) for all selected functions, except for functions, which need to be changed. Selecting 0xF will keep the current function for the function group.

In response to a set function, the switch function will return the following 3 statuses:

- The functions that are supported by each of the function groups
- The function, which is the result of the switch command. In case of invalid selection of one function or more, all set values are ignored and no change will be done (identical to the case where the host selects 0xF for all functions groups). The response to an invalid selection of function will be 0xF.
- Maximum current consumption under the selected functions. If one of the selected functions was wrong the return value will be 0.

Arg. Slice	[23:20]	[19:16]	[15:12]	[11:8]	[7:4]	[3:0]
Group No.	6	5	4	3	2	1
Function name	reserved	reserved	reserved	reserved	Command system	Access mode
0x0	Default (Ver. 1.01)					
0x1	Reserved	Reserved	Reserved	Reserved	For eC	High-Speed
0x2	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0x3	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0x4	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0x5	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0x6	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0x7	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0x8	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0x9	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0xA	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0xB	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0xC	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0xD	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0xE	Reserved	Reserved	Reserved	Reserved	Vendor specific	Reserved
0xF	No influence					

Table 10: Functions

- **Switch function status**

The switch function status is the returned data block that contains function and current consumption information. The block length is predefined to 512 bits and the use of SET_BLK_LEN command is not necessary. Table 11 describes the status data structure.

The status bits of the response contain the information of the function group. Maximum current consumption will be used only for the new function added through this command. In this case VDD_R_CURR_MIN, VDD_W_CURR_MIN, VDD_R_CURR_MAX and VDD_W_CURR_MAX values in the CSD register provides the current consumption when all card functions are set to the default state and can be used by spec 1.01 compatible hosts.

Bits	Description	Width
511:496	Maximum current consumption (0:Error, 1:1mA, 2:2mA… , 65,535:65,535mA) under the function shown with [399:376] bits. The voltage to calculate current consumption is defined by ACMD41 (SD memory card) or CMD5 (SD I/O card). Maximum current consumption indicates the total card current(memory portion) if the functions are switched. The host should check the maximum current consumption and verify that it can supply the necessary current before mode 1 operation. Maximum current consumption is average over 1second.	16
495:480	Function group 6, information . If a bit i is set, function i is supported	16
479:464	Function group 5, information. If a bit i is set, function i is supported	16
463:448	Function group 4, information . If a bit i is set, function i is supported	16
447:432	Function group 3, information . If a bit i is set, function i is supported	16
431:416	Function group 2, information. If a bit i is set, function i is supported	16
415:400	Function group 1, information. If a bit i is set, function i is supported	16
399:396	mode 0 - The function which will be switched in function group 6. mode 1 - The function which is result of the switch command, in function group 6. 0xF shows function set error with the argument.	4
395:392	mode 0 - The function which will be switched in function group 5. mode 1 - The function which is result of the switch command, in function group 5. 0xF shows function set error with the argument.	4

Bits	Description	Width
391:388	mode 0 - The function which will be switched in function group 4. mode 1 - The function which is result of the switch command, in function group 4. 0xF shows function set error with the argument.	4
387:384	mode 0 - The function which will be switched in function group 3. mode 1 - The function which is result of the switch command, in function group 3. 0xF shows function set error with the argument.	4
383:380	mode 0 - The function which will be switched in function group 2. mode 1 - The function which is result of the switch command, in function group 2. 0xF shows function set error with the argument.	4
379:376	mode 0 - The function which will be switched in function group 1. mode 1 - The function which is result of the switch command, in function group 1. 0xF shows function set error with the argument.	4
375:0	Reserved (All '0')	376

Table 11: Status data structure

4.3.10.1 Relationship between CMD6 data & other commands

The card may accept the commands using only CMD line (CMD12, CMD13, etc) during CMD6 transaction but its response and result is undefined.

Application Note: The host is advised not to issue any command during CMD6 transaction. If the host cannot get valid data of CMD6, it advised to issue CMD0 and try re-initialization.

- **Relationship between CMD6 data &CMD12**

Case 1: Not complete case (The card does not output all data.)

In case that the host sends the end bit of CMD12 before CRC bit 15, CMD6 is stopped by CMD12, card shall terminate data transfer of CMD6. The card behavior is not guaranteed and re-initialization from CMD0 is the only way to recover from undefined sate. The end bit of the host command is followed, on the data line, with one more data bit and one end bit.

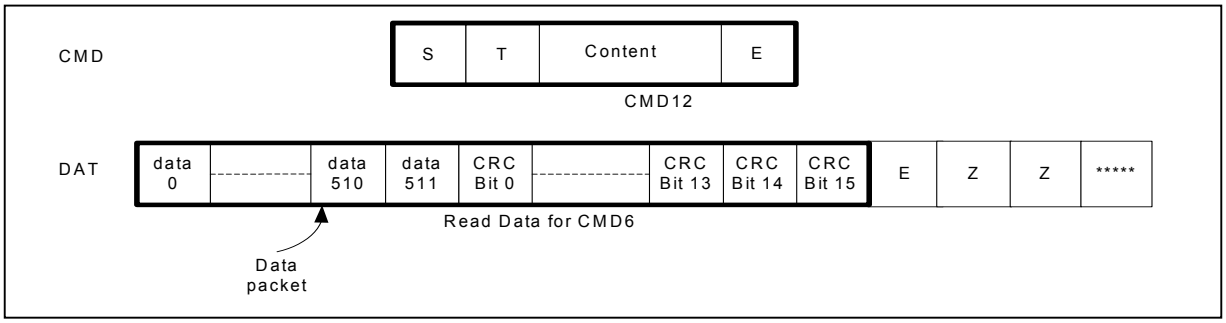


Figure 17: CMD12 during CMD6; Case 1

Case 2: Complete case (The card outputs all data.)

The card shall complete CMD6 execution and its behavior is guaranteed. Complete case includes the later timing of CMD12 than Figure 18. The end bit of the host command is followed, by the end bit on the data line.

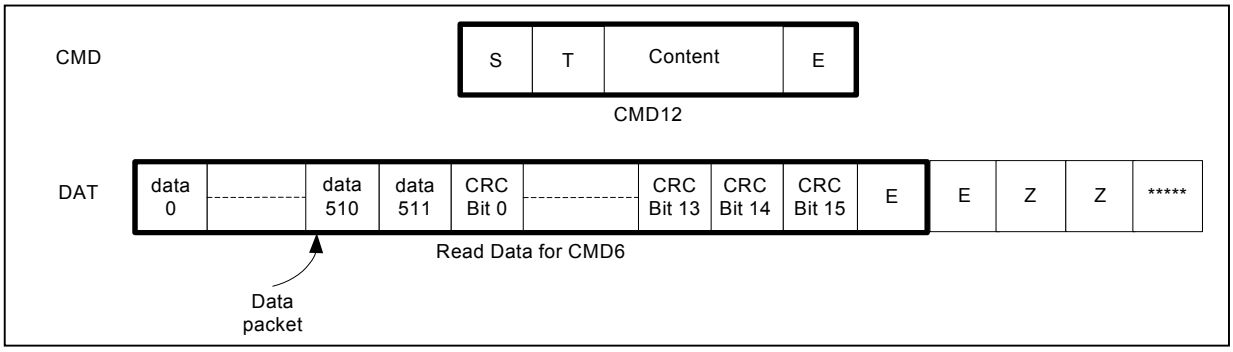


Figure 18: CMD12 during CMD6; Case 2

Application Note:
The host is advised not to issue CMD12 during CMD6 transaction.

- **Switch function flow example**

Application Note: The host is recommended to take the following flow for switching the function.

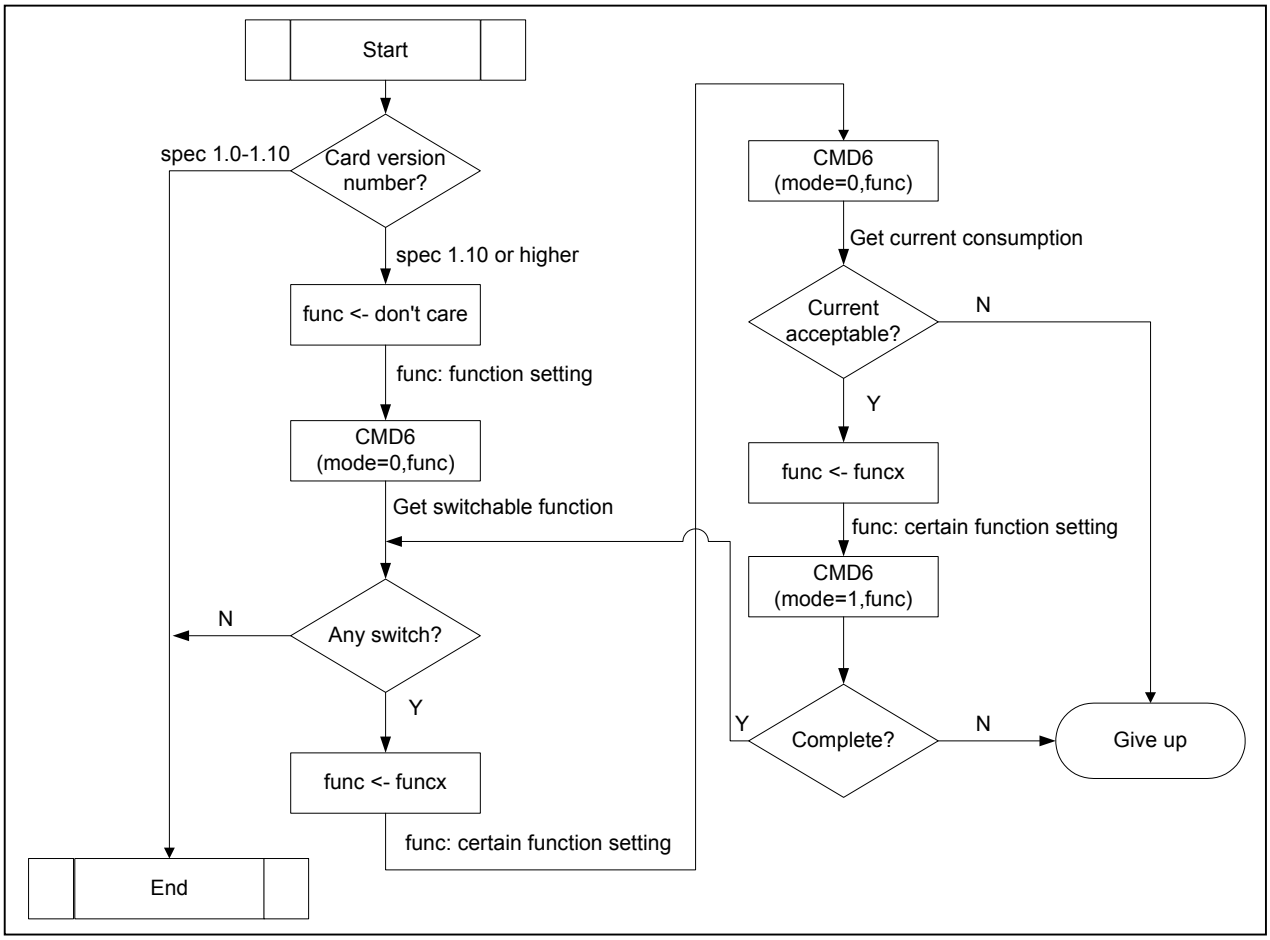


Figure 19: Switching function flow

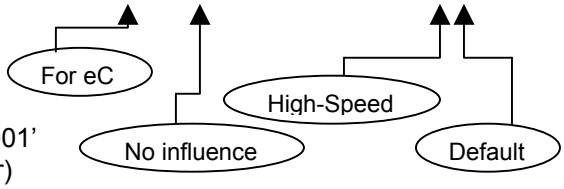
4.3.10.2 Example for checking

Card condition

Support function = command system : For eC(0x1) , access mode : High-Speed(0x1)
Current function = command system : For eC(0x1) , access mode : Default(0x0)
Switch example : command system : For eC => Default, access mode : Default => High-Speed

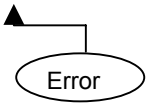
Case (1) - Check function with no error

CMD6 argument = '0000 0000 1111 1111 1111 1111 **0000 0001**'
Read Data = [511:496] = '0000 0000 0010 0000' (=64mA)
[495:400] = '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 00**1**1' & '**1**000 0000 0000 00**11**'
[399:376] = '0000 0000 0000 0000 0000 0001'
[375:0] = Reserved (All '0')



Case (2) - Check function with error

CMD6 argument = '0000 0000 1111 1000 1111 0010 0000 0001'
Read Data = [511:496] = '0000 0000 0000 0000' (means error)
[495:400] = '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 0011' & '1000 0000 0000 0011'
[399:376] = '0000 **1111** 0000 **1111** 0000 0001'
[375:0] = Reserved (All '0')



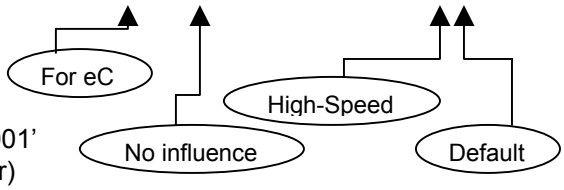
4.3.10.3 Example for switching

Card condition

Support function = command system : For eC(0x1), access mode : High-Speed(0x1)
Current function = command system : For eC(0x1), access mode : Default(0x0)
Switch example : command system : For eC => Default, access mode : Default => High-Speed

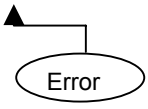
Case (3) - Switch function with no error

CMD6 argument = '1000 0000 1111 1111 1111 1111 **0000 0001**'
Read Data = [511:496] = '0000 0000 0010 0000' (=64mA)
[495:400] = '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 00**1**1' & '**1**000 0000 0000 00**11**'
[399:376] = '0000 0000 0000 0000 0000 0001'
[375:0] = Reserved (All '0')



Case (4) - Switch function with error

CMD6 argument = '1000 0000 1111 1000 1111 0010 0000 0001'
Read Data = [511:496] = **'0000 0000 0000 0000'** (means error)
[495:400] = '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 0001' & '1000 0000 0000 0011' & '1000 0000 0000 0011'
[399:376] = '0000 **1111** 0000 **1111** 0001 0000'
[375:0] = Reserved (All '0')



4.3.11 High-Speed mode (25MB/sec interface speed) (This chapter is newly added in version 1.10)

Though the Rev 1.01 SD memory card supports up to 12.5MB/sec interface speed, the speed of 25MB/sec is necessary to support increasing performance needs of the host and because of memory size which continues to grow.

To achieve 25MB/sec interface speed, clock rate is increased to 50MHz and CLK/CMD/DAT signal timing and circuit conditions are reconsidered and changed from Physical Layer Specification Version 1.01.

After power up, the SD memory card is in the default speed mode, and by using Switch Function command(CMD6)), the rev 1.10 (and greater) SD memory card can be placed in High-Speed mode. The High-Speed function is a function in the access mode group (see Table 10). Supporting High-Speed mode is optional.

Because it is not possible to control two cards or more in case that each of them have a different timing mode (Default and High-Speed mode) and in order to satisfy severe timing, host shall drive only one card. CLK/CMD/DAT signal shall be connected in 1 to 1 between the host and the card.

Maximum current consumption for SD memory cards operating in High-Speed mode is 200mA (average over 1 second). VDD_R_CURR_MIN, VDD_W_CURR_MIN, VDD_R_CURR_MAX and VDD_W_CURR_MAX values in CSD register are meaningless in High-Speed mode.

In High-Speed mode, TRAN_SPEED value in CSD register is 0_1011_010b (05Ah) which is equal to 50MHz.

If the host wants to know the current consumption, the current consumption indicated in the switch function status returned by the Switch Function command (CMD6) shall be checked.

4.3.12 Command system (This chapter is newly added in version 1.10)

SD commands CMD34-37, CMD50, CMD57 are reserved for SD command system expansion via the switch command. Switching between the various functions of the command system function group, will change the interpretation and associated bus transaction (i.e. command without data transfer, single block read, multiple block write, etc.) of these commands. Supporting Command system is optional

- When the "standard command set" (default function 0x0) is selected these commands will not be recognized by the card and will be considered as illegal commands (as defined in revision 1.01 of the SD physical layer specification)
- When the “vendor specific” (function 0xE) is selected the behavior of these commands are vendor specific. They are not defined by this standard and may change for different card vendors.
- When the “mobile e-commerce” (function 0x1) is selected the behavior of these commands is governed by the SD memory card spec part A1: Mobile Commerce Extension Specification.

When either one of these extensions is used, special care should be given to proper selection of the command set function, otherwise the host command may be interpreted incorrectly.

All other commands of the SD memory card (not reserved for the switch commands) are always available and will be executed as defined in this document regardless of the currently selected commands set.

4.4 Clock Control

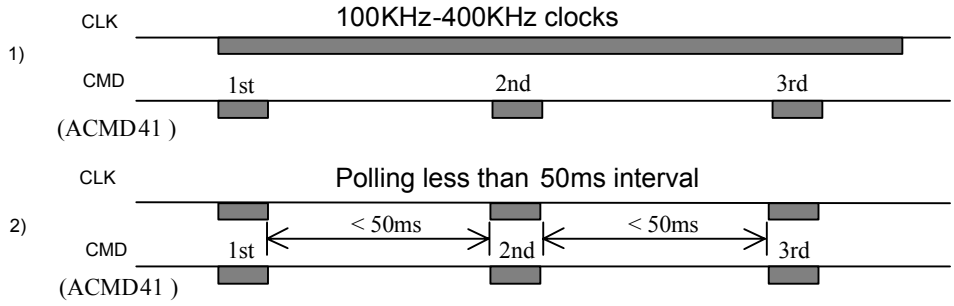
The SD Memory Card bus clock signal can be used by the host to turn the cards into energy saving mode or to control the data flow (to avoid under-run or over-run conditions) on the bus. The host is allowed to lower the clock frequency or shut it down. For example, in a case that a host with 512Bytes 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 and the identification frequency defined by the specification document).
- An exemption to the above is ACMD41(SD_APP_OP_COND). After issuing command ACMD41 the following 1) or 2) procedures shall be done by the host until the card becomes ready.

  1) Issue continues clock in frequency range of 100KHz-400KHz.

  2) If the host wants to stop the clock, poll busy bit by ACMD41 command at less than 50ms intervals.



- 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).

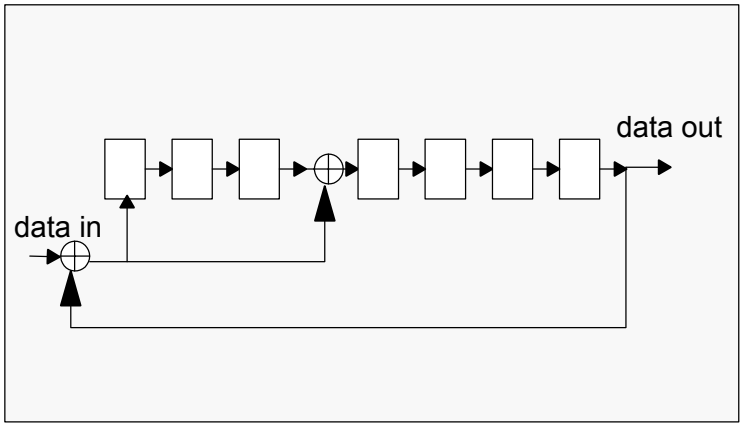


Figure 20: CRC7 generator/checker

- **CRC7 examples**

The CRC section of the command/response is bolded.

CMD0 (Argument=0) --> 01 000000 00000000000000000000000000000000 "**1001010**" 1

CMD17 (Argument=0) --> 01 010001 00000000000000000000000000000000 "**0101010**" 1

Response of CMD17 --> 00 010001 00000000000000000000100100000000 "**0110011**" 1

- **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 $G(x) = x^{16} + x^{12} + x^5 + 1$

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

$CRC[15...0] = \text{Remainder } [(M(x) * x^{16}) / G(x)]$

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.

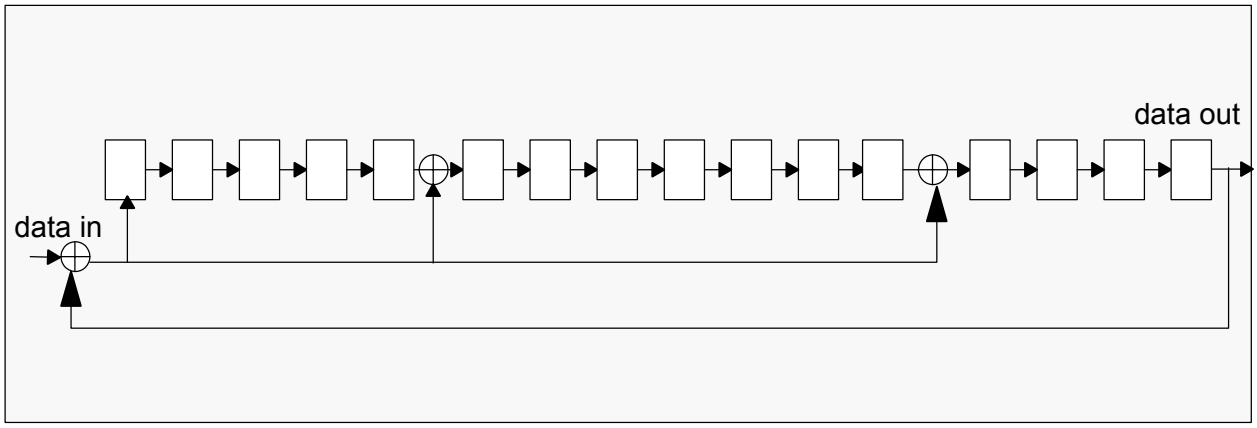


Figure 21: CRC16 generator/checker

- **CRC16 example**

512 bytes with 0xFF data --> CRC16 = 0x7FA1

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-erro-

neous state branches are shown in the state diagrams (see Figure 14 to Figure). Table 24 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 operations occurs are (card independent) **either 100 times longer** than the typical access times for these operations given below **or 100ms (the lower of them)**. The times after which a time-out condition for Write/Erase operations occurs are (card independent) **either 100 times longer** than the typical program times for these operations given below **or 250ms (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(CLR)_WRITE_PROTECT, PROGRAM_CSD 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 12: 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 4.5 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 14-Table 22.

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 13). 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**	**10**	**11**
Supported commands	**class description**	basic	reserved	block read	reserved	block write	erase	write protection	lock card	application specific	I/O mode	switch	reserved
CMD0	Mandatory	+											
CMD2	Mandatory	+											
CMD3	Mandatory	+											
CMD4	Mandatory	+											
CMD5	Optional										+		
CMD6(2)	Mandatory											+	
CMD7	Mandatory	+											
CMD9	Mandatory	+											
CMD10	Mandatory	+											
CMD12	Mandatory	+											
CMD13	Mandatory	+											
CMD15	Mandatory	+											
CMD16	Mandatory			+		+			+				
CMD17	Mandatory			+									
CMD18	Mandatory			+									
CMD24	Mandatory(1)					+							
CMD25	Mandatory(1)					+							
CMD27	Mandatory(1)					+							
CMD28	Optional							+					
CMD29	Optional							+					
CMD30	Optional							+					
CMD32	Mandatory(1)						+						
CMD33	Mandatory(1)						+						
CMD34-37(2)	Optional											+	
CMD38	Mandatory(1)						+						
CMD42	Optional								+				
CMD50(2)	Optional											+	
CMD52	Optional										+		
CMD53	Optional										+		
CMD55	Mandatory									+			
CMD56	Mandatory									+			

	Card Command Class (CCC)	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Supported commands	**class description**	basic	reserved	block read	reserved	block write	erase	write protection	lock card	application specific	I/O mode	switch	reserved
CMD57(2)	Optional											+	
ACMD6	Mandatory									+			
ACMD13	Mandatory									+			
ACMD22	Mandatory(1)									+			
ACMD23	Mandatory(1)									+			
ACMD41	Mandatory									+			
ACMD42	Mandatory									+			
ACMD51	Mandatory									+			

Table 13: Card Command Classes (CCCs)

Note (1): The write related commands are mandatory only for the Writable type of Cards (OTP and R/W).

Note (2): This command is newly defined in version 1.10

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 for I/O cards (refer to "SDIO Card Specification")				
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. In case that the RCA equal 0 then the host may do one of the following: - Use other RCA number to perform card deselection. - Re-send CMD3 to change its RCA number to other than 0 and then use CMD7with RCA=0 for card de-selection.
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 identi-fication (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				

CMD INDEX	type	argument	resp	abbreviation	command description
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 com-munication breakdowns.

Table 14: 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, write, lock). Default block length is fixed 512Bytes. If block length is set bigger than 512Bytes, the card will set the BLOCK_LEN_ERROR bit. 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_B LOCK	continuously transfers data blocks from card to host until interrupted by a STOP_TRANSMISSION command.
CMD19 ... CMD23	reserved				

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

Table 15: Block oriented read commands (class 2)

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, write, lock). Default block length is specified in the CSD. Supported only if Partial block RD/WR operation are allowed in CSD.
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.

CMD INDEX	type	argument	resp	abbreviation	command description
CMD26	Reserved For Manufacturer				
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	programming of the programmable bits of the CSD.

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).

Table 16: Block oriented write commands (class 4)

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				

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

Table 17: Block oriented write protection commands (class 6)

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.

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 18: Erase commands (class 5)

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, write, lock). Default block length is specified in the CSD. Supported only if Partial block RD/WR operation are allowed in CSD.
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.
CMD43-49 CMD51	reserved				

Table 19: 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.
CMD58-59	reserved				
CMD60-63	reserved for manufacturer				

Table 20: 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 20) are supported if Class 8 is allowed (mandatory in SD Memory Card).

CMD INDEX	type	argument	resp	abbreviation	command description
CMD52... CMD54	reserved for I/O mode (refer to "SDIO Card Specification")				

Table 21: I/O mode commands (class 9)

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 31.
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. If WRITE_BL_PARTIAL='0', the unit of ACMD22 is always 512byte. If WRITE_BL_PARTIAL='1', the unit of ACMD22 is a block length which was used when the write command was executed.
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]

ACMD INDEX	type	argument	resp	abbreviation	command description
ACMD38	--	--	--	--	Reserved for SD security applications[1]
ACMD39 to ACMD40	reserved				
ACMD41	bcr	[31:0]OCR without busy	R3	SD_SEND_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.
ACMD43 ACMD49	--	--	--	--	Reserved for SD security applications[1]
ACMD51	adtc	[31:0] stuff 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 22: Application Specific Commands used/reserved by SD Memory Card

Table 23 is newly added in version 1.10

CMD INDEX	type	argument	resp	abbreviation	command description
CMD6	adtc	[31] Mode 0:Check function 1:Switch function [30:24] reserved (All ’0’) [23:20] reserved for function group 6 (All ’0’ or 0xF) [19:16] reserved for function group 5 (All ’0’ or 0xF) [15:12] reserved for function group 4 (All ’0’ or 0xF) [11:8] reserved for function group 3 (All ’0’ or 0xF) [7:4] function group 2 for command system [3:0] function group 1 for access mode	R1	SWITCH_FUNC	Checks switchable function (mode 0) and switch card function (mode 1). see Chapter 4.3.10.
CMD34	Reserved for each command system set by switch function command (CMD6). Detail definition is refer to each command system specification.				
CMD35					
CMD36					
CMD37					
CMD50					

CMD INDEX	type	argument	resp	abbreviation	command description
CMD57					

Table 23: Switch function commands (class 10)

4.8 Card State Transition Table

Table 24 defines the card state transitions in dependency of the received command.

	current state									
	idle	**ready**	**ident**	**stby**	**tran**	**data**	**rcv**	**prg**	**dis**	**ina**
Trigger of state change	**changes to**									
class independent										
“Operation Complete”	-	-	-	-	-	-	-	tran	stby	-
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										
CMD32	-	-	-	-	tran	-	-	-	-	-

	current state									
	idle	**ready**	**ident**	**stby**	**tran**	**data**	**rcv**	**prg**	**dis**	**ina**
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	-	-	-	-	data	-	-	-	-	-
ACMD22	-	-	-	-	data	-	-	-	-	-
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										
CMD52-CMD54	refer to "SDIO Card Specification"									
class 10(1)										
CMD6	-	-	-	-	data	-	-	-	-	-
CMD34-37,50,57	-	-	-	-	tran	-	-	-	-	-
class 11										
CMD41; CMD43...CMD49, CMD58-CMD59	reserved									
CMD60...CMD63	reserved for manufacturer									

Table 24: Card state transition table

Note (1): Class 10 commands are newly defined in version 1.10

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 bit string 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 4.5 for the definition of CRC7). Every command codeword is terminated by the end bit (always '1').

There are four types of responses for SD Memory Card. The SDIO Card supports additional response types named R4 and R5. Refer to SDIO Card Spec for detailed information on the SDIO commands and 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 25: 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'

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 26: 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 27: 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 29)	CRC7	end bit

Table 28: 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 29 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 X	'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 X	'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 R 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

Bits	Identifier	Type	Value	Description	Clear Condition
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.	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 R 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
18	reserved				
17	reserved				
16	CSD_OVERWRITE	E R X	’0’= no error ’1’= error	can be either one of the following errors: - 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

Bits	Identifier	Type	Value	Description	Clear Condition
12:9	CURRENT_STATE	S X	0 = idle 1 = ready 2 = ident 3 = stby 4 = tran 5 = data 6 = rcv 7 = prg 8 = dis 9-14 = reserved 15 = reserved for I/O mode	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 signaling 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 for SD I/O Card				
3	AKE_SEQ_ERROR (SD Memory Card app. spec.)	E R	'0' = no error '1' = error	Error in the sequence of authenti-cation process	C
2	reserved for application specific commands				
1, 0	reserved for manufacturer test mode				

Table 29: 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		
6(2)	x						x		x	x	x	x	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	

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
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	x	
25	x	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 30: 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

(2): This command is newly defined in version 1.10

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	'00xxh'= SD Memory Cards as defined in Physical Spec Ver.1-1.10 ('x'=don't care). The following cards are currently defined: '0000'= Regular SD RD/WR Card. '0001'= SD ROM Card	In the future the 8LSBs will be used to define different variations of a SD Memory Card (Each bit will define different SD Type). The 8MSBs will be used to define SD Cards that do not comply with SD Memory Card as was defined in Spec Ver.1-1.10	A
479: 448	SIZE_OF_PROTECTED_AREA	SR	Size of protected area (in units of MULT*BLOCK_LEN refer to CSD register Table 10.3)	Shows the size of protected area. The actual area = (SIZE_OF_PROTECTED_AREA) * MULT * BLOCK_LEN.	A
447: 312	reserved				
311:0	reserved for manu-facturer				

Table 31: 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.

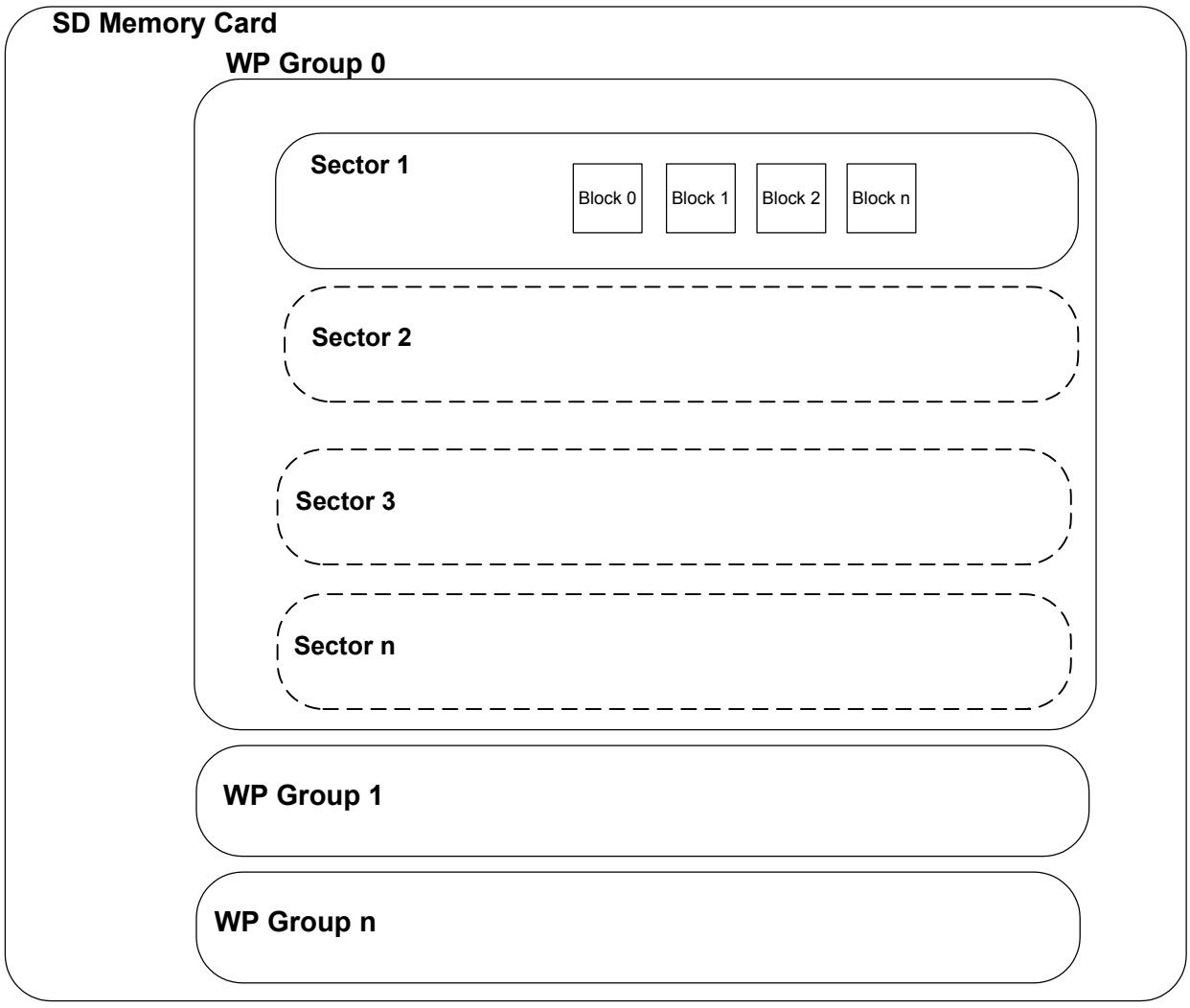


Figure 22: Write Protection hierarchy

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 32: 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 33.

4.12.1 Command and Response

Both host command and card response are clocked per the timing specified in

Section 6.8 (and Section 6.9 for high speed card)

- **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.

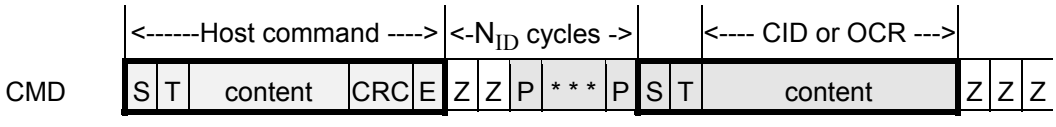


Figure 23: Identification timing (card identification mode)

- **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.

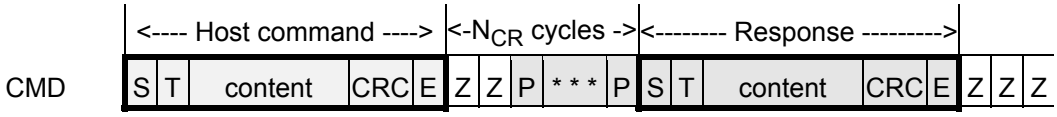


Figure 24: SEND_RELATIVE_ADDR timing

- **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:

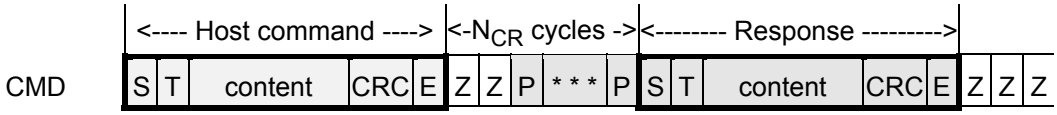


Figure 25: Command response timing (data transfer mode)

- **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.

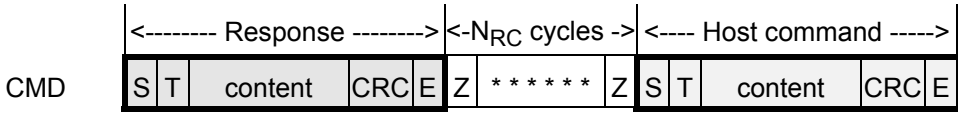


Figure 26: Timing response end to next CMD start (data transfer mode)

- **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.

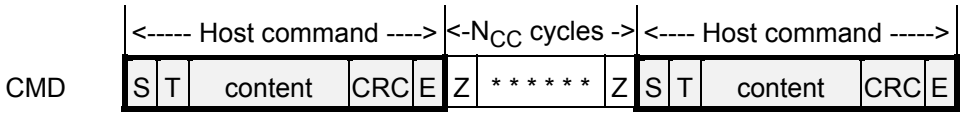


Figure 27: Timing of command sequences (all modes)

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 28. 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.

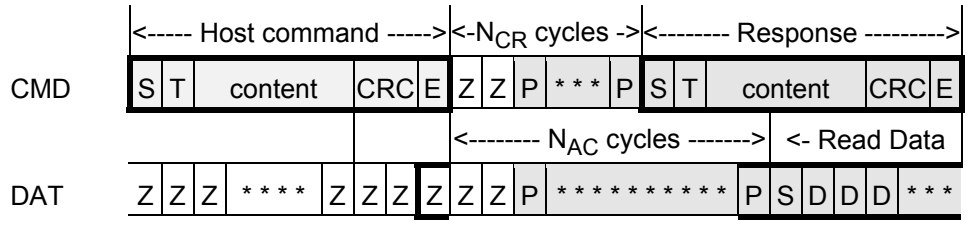


Figure 28: Timing of single block read

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 29 describes the timing of the data blocks and Figure 30 the response to a stop command. The data transmission stops two clock cycles after the end bit of the stop command.

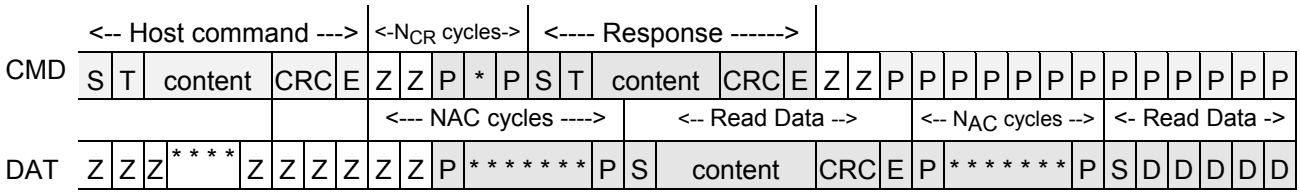


Figure 29: Timing of multiple block read command

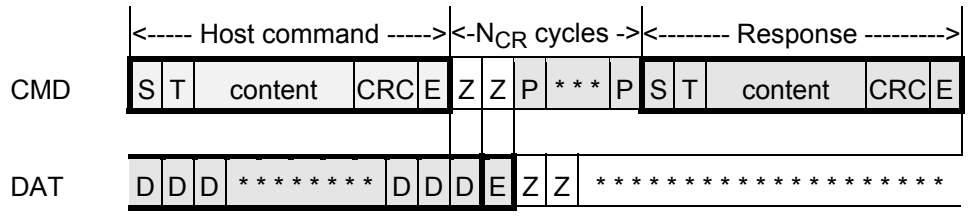


Figure 30: Timing of stop command (CMD12, data transfer mode)

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 30. 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‘).

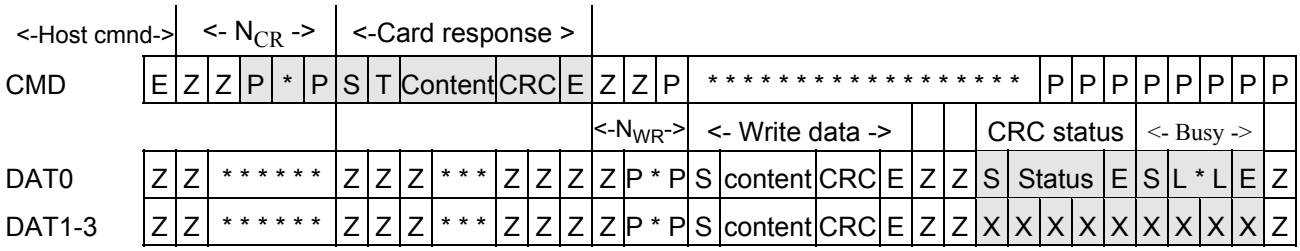


Figure 31: Timing of the block write command

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 signaling 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 32 describes the timing of the data blocks with and without card busy signal.

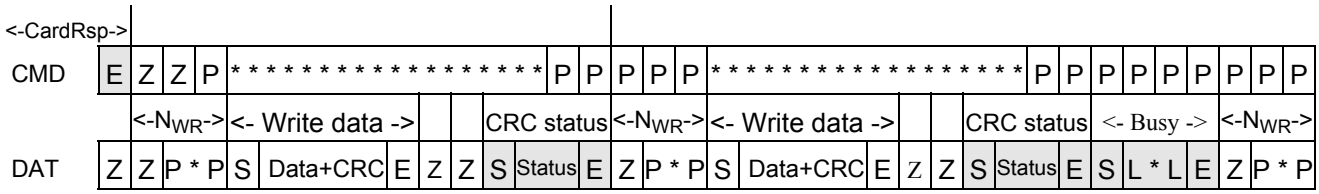


Figure 32: Timing of the multiple block write command

The stop transmission command works similar as in the read mode. Figure 33 to Figure 36 describe the timing of the stop command in different card states.

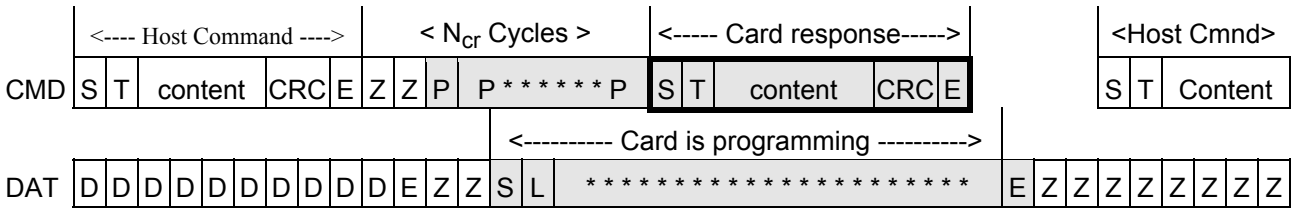
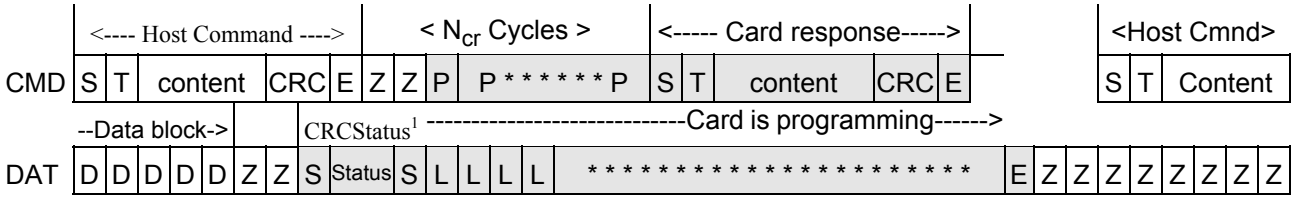


Figure 33: Stop transmission during data transfer from the host

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 34 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 and start of busy signaling. In that case there are no Z clocks, for switching the bus direction, because the bus direction is already towards the host. 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 34: 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.

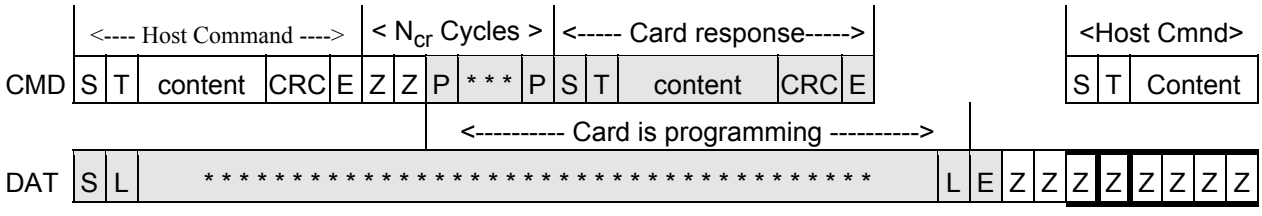


Figure 35: Stop transmission received after last data block. Card is busy programming.

:

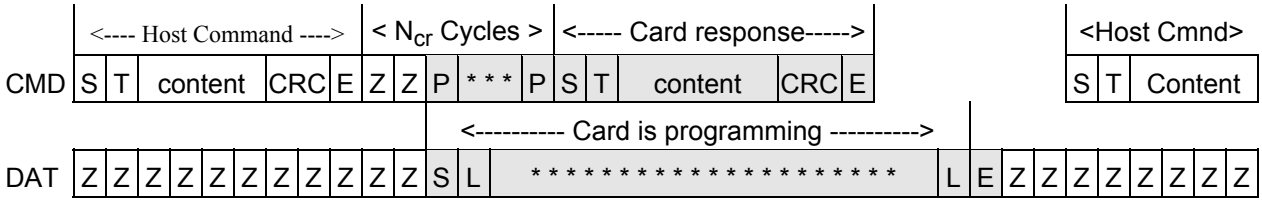


Figure 36: Stop transmission received after last data block. Card becomes busy.

- **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 36.

- **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 36.

4.12.4 Timing Values

Table 33 defines all timing values.

	Min	Max	Unit
N_{CR}	2	64	clock cycles
N_{ID}	5	5	clock cycles
N_{AC} [1]	2	-	clock cycles
N_{RC}	8	-	clock cycles
N_{CC}	8	-	clock cycles
N_{WR}	2	-	clock cycles

1) The maximum read access time shall be calculated by host as follows:

Nac(max)= 100 ((TAAC * fpp) + (100 * NSAC)) ;

fpp is the interface clock rate and TAAC & NSAC are given in the CSD (Chapter 5.3).

Table 33: 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.

OCR bit position	VDD voltage window
0-3	reserved
4	reserved
5	reserved
6	reserved
7	reserved
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 34: OCR register definition

The supported voltage range is coded as shown in Table 34. 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.

1)This bit is set to LOW if the card has not finished the power up routine

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 35: 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 2 ASCII string characters 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 binary number.

- **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 4.5.

5.3 CSD Register

The Card-Specific Data register provides information on how to access the card contents. The CSD defines the data format, error correction type, maximum data access time, 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 @VDD min	VDD_R_CURR_MIN	3	R	[61:59]
max. read current @VDD max	VDD_R_CURR_MAX	3	R	[58:56]
max. write current @VDD min	VDD_W_CURR_MIN	3	R	[55:53]
max. write current @VDD max	VDD_W_CURR_MAX	3	R	[52:50]

Name	Field	Width	Cell Type	CSD-slice
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 36: The CSD Register fields

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 1.0-1.10
1-3	reserved	

Table 37: 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 38: 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 33 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 39: Maximum data transfer rate definition

Note that for current SD Memory Cards that field must be always 0_0110_010b (032h) which is equal to 25MHz - the mandatory maximum operating frequency of SD Memory Card.

In High-Speed mode, that field must be always 0_1011_010b (05Ah) which is equal to 50MHz. And when the timing mode returns to the default by CMD6 or CMD0 command, its value will be 032h.

- **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 13.

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

Table 40: 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 41: Data block length

- **READ_BL_PARTIAL (always = 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 42: 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)

Maximum capacity of the card, compliant to SD Physical Specification Versoin1.01 shall be up to 2G bytes (2^{31} bytes) to be consistent with the maximum capacity (2G bytes) of SD Memory Card File System Specification Ver.1.01.

To indicate 2GByte card, BLOCK_LEN shall be 1024 bytes.

Therefore, the maximal capacity which can be coded is 4096*512*1024 = 2G bytes.

Example: A 32Mbyte 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}

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 43: $V_{DD,min}$ 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 44: $V_{DD,max}$ 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 45: Multiply factor for the device size

- **ERASE_BLK_EN**

The ERASE_BLK_EN defines the granularity of the unit size of the data to be erased. The erase operation can erase either one or multiple units of WRITE_BL_LEN or one or multiple units (or sectors) of SECTOR_SIZE (see definition below).

If ERASE_BLK_EN = ’0’, the host can erase one or multiple units of SECTOR_SIZE. The erase will start from the beginning of the sector that contains the start address to the end of the sector that contains the end address. For example, if SECTOR_SIZE=31 and the host sets the Erase Start Address to 5 and the Erase End Address to 40, the physical blocks from 0 to 63 will be erased as shown in **Figure 37**.

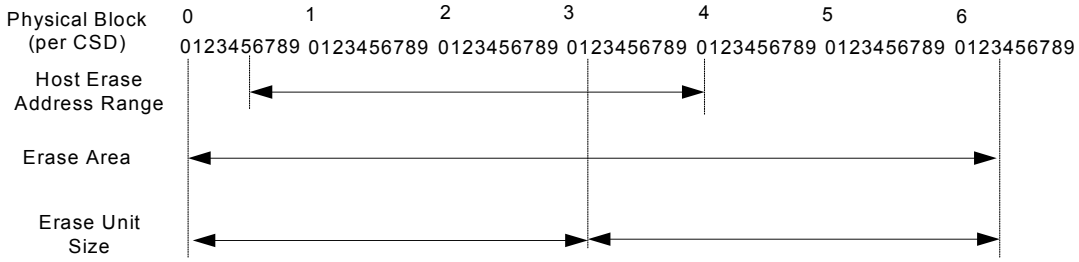


Figure 37: ERASE_BLK_EN = '0' example

If ERASE_BLK_EN = ’1’ the host can erase one or multiple units of 512 bytes. All blocks that contain data from start address to end address are erased. For example, if the host sets the Erase Start Address to 5 and the Erase End Address to 40, the physical blocks from 5 to 40 will be erased as shown in **Figure 38**.

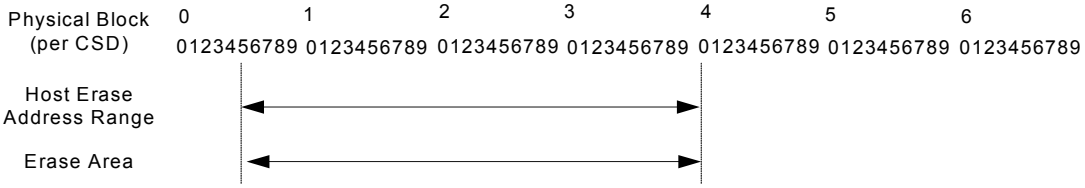


Figure 38: ERASE_BLK_EN = '1' example

- **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)
2	4
3	8
4	16
5	32
6,7	reserved

Table 46: 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	
.....		

WRITE_BL_LEN	Block length	Remark
11	2^{11} = 2048 Bytes	
12-15	reserved	

Table 47: 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 48 (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 48: 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 4.5. 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		+	+	+	+	+	+	
TRAN_SPEED		+	+					
CCC	+	+	+	+	+	+	+	+

	Command classes							
CSD Field	**0**	**2**	**4**	**5**	**6**	**7**	**8**	**9**
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 49: 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 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 widths supported	SD_BUS_WIDTHS	4	R	[51:48]
reserved	-	16	R	[47:32]
reserved for manufacturer usage	-	32	R	[31:0]

Table 50: The SCR Fields

- **SCR_STRUCTURE**

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

SCR_STRUCTURE	SCR structure version	Valid for SD Physical Layer Specification Version
0	SCR version No. 1.0	Version 1.0-1.10
1-15	reserved	

Table 51: SCR register structure version

- **SD_SPEC**

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

SD_SPEC	Physical Layer Specification Version Number
0	Version 1.0-1.01
1	Version 1.10
2-15	reserved

Table 52: 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 53: SD Supported security algorithm

Security Protocol 1.0 relates to Security Specification Version 0.96.

Security Protocol 2.0 relates to Security Specification Version 1.0.-1.01

Note that it is mandatory for a regular writable SD Memory Card to support Security Protocol. For ROM (Read Only) and OTP (One Time Programmable) type of SD Memory Card the security feature is optional

- **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 54: SD Memory Card Supported Bus Widths

Since SD Memory Card shall support at least the two bus modes 1bit or 4bit width then any SD Card shall set at least bits 0 and 2 (SD_BUS_WIDTH="0101").

6. SD Memory Card Hardware Interface

This Chapter is omitted from the simplified version of the physical layer specification.

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.

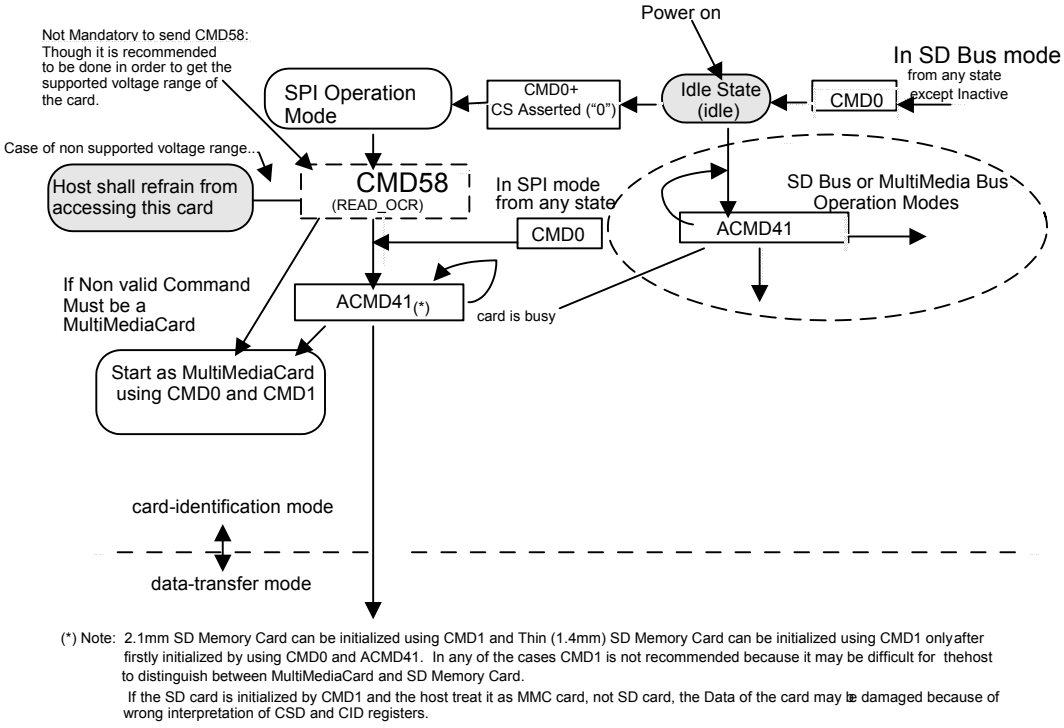


Figure 39: SPI Mode Initialization Flow

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.

During the initialization sequence, if the host gets Illegal Command indication for ACMD41 sent to the card, it may assume that the card is MultiMediaCard. In that case it should re-start the card as MultiMediaCard using CMD0 and CMD1.

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 52).

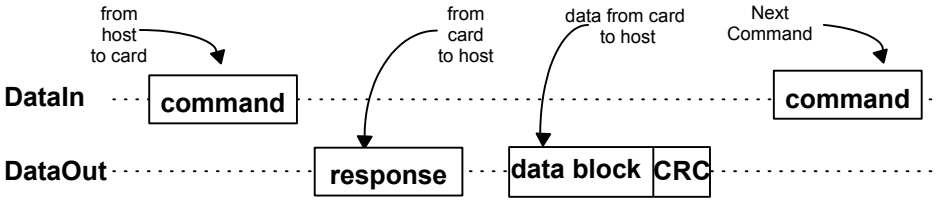


Figure 40: Single Block Read operation

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 53 shows a data read operation which terminated with an error token rather than a data block.

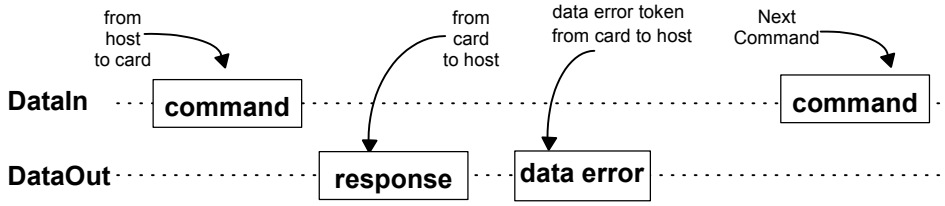


Figure 41: Read operation - data error

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).

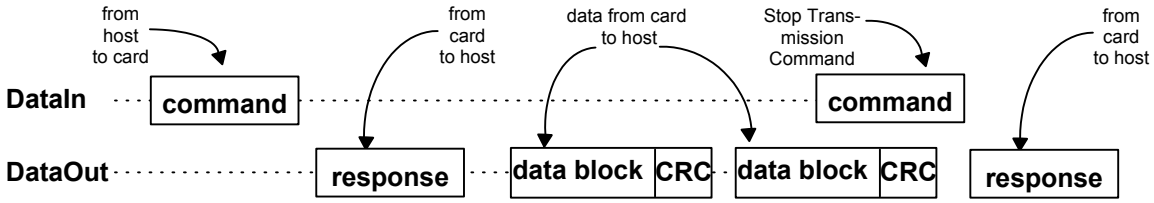


Figure 42: Multiple Block Read operation

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 55).

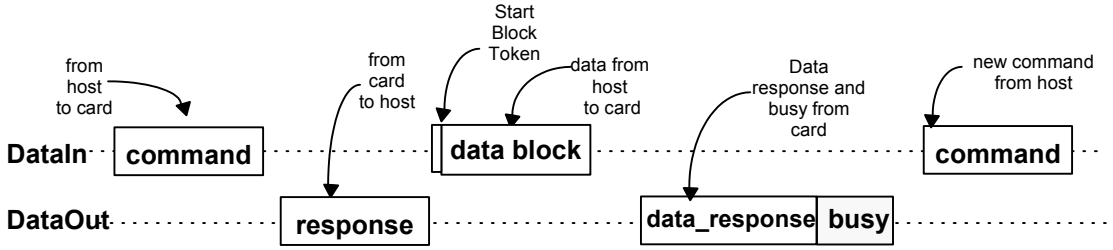


Figure 43: Single Block Write operation

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.

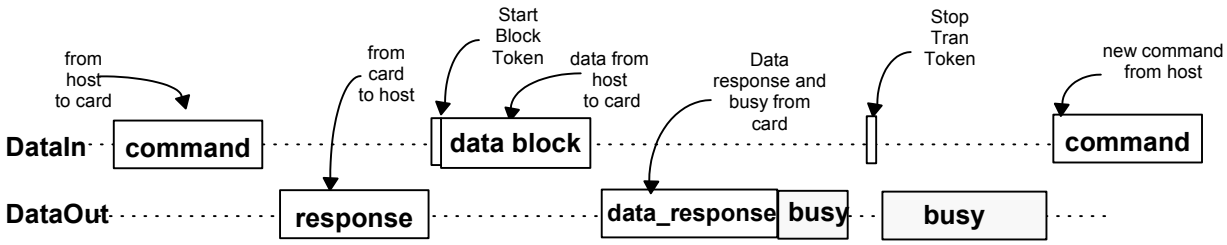


Figure 44: Multiple Block Write operation

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 57 illustrates a 'no data' bus transaction with and without busy signaling.

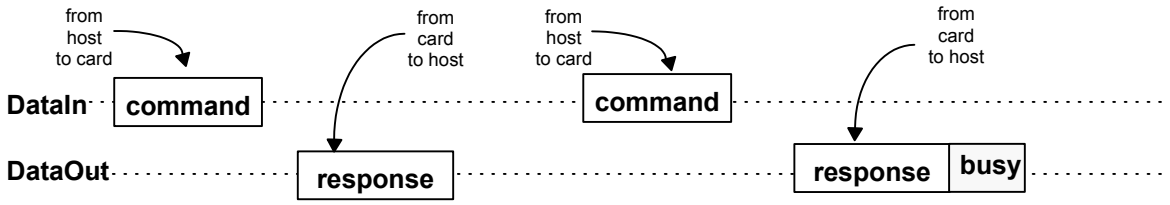


Figure 45: 'No data' operations

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 53) 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 valid host commands are ACMD41 (SD_SEND_OP_COND), CMD58 (READ_OCR) and CMD59 (CRC_ON_OFF). For the Thick (2.1mm) SD Memory Card - CMD1 (SEND_OP_COND) is also valid - that means that in SPI mode CMD1 and ACMD41 have the same behavior, though the usage of CMD41 is preferable since it allows easy distinguishing between SD Memory Card and MultiMediaCard. **For the Thin (1.4mm) SD Memory Card CMD1 (SEND_OP_COND) is illegal command during the initialization that is done after power on. After Power On, once the card accepted valid ACMD41, it will be able to accept also CMD1 even if used after re-initializing (CMD0) the card.** It was defined in such way in order to be able to distinguish between Thin SD Memory Card and MultiMediaCards (that supports CMD1 as well).

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, ACMD41 (or CMD1 as well, for 2.1mm-SD Memory Card) 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 in any one of the following cases:

- It is sent while the card is in read operation (except CMD12 which is legal).
- It is sent wile the card is in Busy.
- Card is Locked and it is other than Class 0 or 7 commands.
- It is not supported (illegal opcode).
- CRC check failed.
- It contains an illegal operand.
- It was out of sequence during an erase sequence.

Note that in case the host sends command while the card sends data in read operation then the response with an illegal command indication may disturb the data transfer.

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 4.3.7 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.

7.2.13 Switch function command (This chapter is newly added in spec version 1.10)

Same as for SD mode with two exceptions:

- The command is valid under the "not idle state".
- In SPI mode, CMD0 switching period is within 8 clocks after the end bit of the CMD0 command R1 response

7.2.14 High-Speed mode (25MB/sec interface speed) (This chapter is newly added in spec version 1.10)

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 4.5). 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**	**6**	**9**	**10**	**12**	**13**	**16**	**17**	**18**	**24**	**25**	**27**	**28**	**29**	**30**	**32**	**33**	**34**	**35**	**36**	**37**	**38**	**42**	**50**	**52**	**53**	**55**	**56**	**57**	**58**	**59**
class 0	Basic	+	+		+	+	+	+																								+	+
class 1	Not supported in SPI																																
class 2	Block read								+	+	+																						
class 3	Not supported in SPI																																

Card CMD Class (CCC)	Class Description	Supported commands																													
		0	1	6	9	10	12	13	16	17	18	24	25	27	28	29	30	32	33	34	35	36	37	38	42	50	52	53	55	56	57	58	59
class 4	Block write								+			+	+	+																		
class 5	Erase																	+	+					+								
class 6	Write-protection (Optional)														+	+	+															
class 7	Lock Card (Optional)								+																+							
class 8	Application specific																												+	+		
class 9	I/O mode																										+	+				
class 10(1)	Switch			+																+	+	+	+			+				+		
class 11	Reserved																															

Table 56: Command classes in SPI mode

Note (1) This command class is newly added in spec version 1.10

- **Detailed Command Description**

The following table provides a detailed description of the SPI bus commands. The responses are defined in Chapter 7.3.2. 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[1]	None	R1	SEND_OP_ COND	Activates the card’s initialization process (In Thin- 1.4mm SD Memory Card it is valid only if used after re-initializing a card - see note at Chapter 7.2.7)
CMD2	No				
CMD3	No				
CMD4	No				

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD5	reserved for I/O Mode (refer to "SDIO Card Specification")				
CMD6[8]	Yes	[31] Mode 0:Check function 1:Switch function [30:24] reserved (All '0') [23:20] reserved for function group 6 (All '0' or 0xF) [19:16] reserved for function group 5 (All '0' or 0xF) [15:12] reserved for function group 4 (All '0' or 0xF) [11:8] reserved for function group 3 (All '0' or 0xF) [7:4] function group 2 for command system [3:0] function group 1 for access mode	R1	SWITCH_FUNC	Checks switchable function (mode 0) and switches card function (mode 1). see Chapter 4.3.10.
CMD7	No				
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	None	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				

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
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).[2]
CMD17	Yes	[31:0] data address	R1	READ_SINGLE_ BLOCK	reads a block of the size selected by the SET_BLOCKLEN command.[3]
CMD18	Yes	[31:0] data address	R1	READ_MULTIPLE _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.[4]
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[5]	SET_WRITE_ PROT	if the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	Yes	[31:0] data address	R1b	CLR_WRITE_ PROT	if the card has write protection features, this command clears the write protection bit of the addressed group.
CMD30	Yes	[31:0] write protect data address	R1	SEND_WRITE_ PROT	if the card has write protection features, this command asks the card to send the status of the write protection bits.[6]
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.

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD34. .. CMD37[8]	Reserved for each command system set by switch function command (CMD6). Detail definition is refer to each command system specification.				
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 4.3.7. The size of the Data Block is defined with SET_BLOCK_LEN command.
CMD43-49 CMD51	reserved				
CMD50[8]	Reserved for each command system set by switch function command (CMD6). Detail definition is refer to each command system specification.				
CMD52. .. CMD54	reserved for I/O Mode (refer to "SDIO Card Specification")				
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_[7]	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[8]	Reserved for each command system set by switch function command (CMD6). Detail definition is refer to each command system specification.				
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				

[1] CMD1 is valid command for the Thin (1.4mm) SD Memory Card only if used after re-initializing a card (not after power on reset).

[2] The default block length is as specified in the CSD.

[3] The data transferred must not cross a physical block boundary unless READ_BLK_MISALIGN is set in the CSD.

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

[5] R1b: R1 response with an optional trailing busy signal

[6] 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

[7] RD/WR_: “1” the Host shall get a block of data from the card.
“0” the host sends block of data to the card.

[8] This command is added in spec version 1.10

Table 57: Commands and arguments

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 31
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				
ACMD41	yes	None	R1	SD_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 CS (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 58. 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.

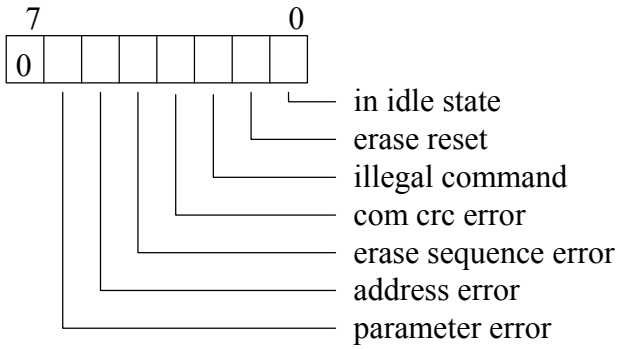


Figure 46: R1 Response Format

- **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 59.

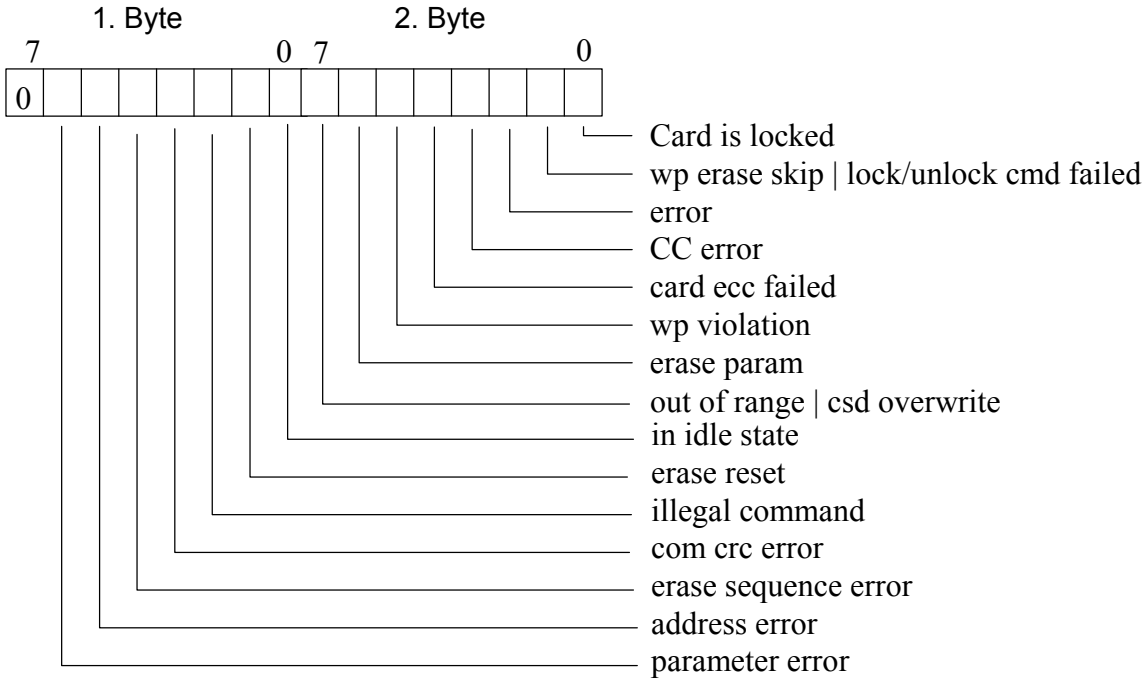


Figure 47: R2 response format

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 overloaded. 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 48). The structure of the first (MSB) byte is identical to response type R1. The other four bytes contain the OCR register.

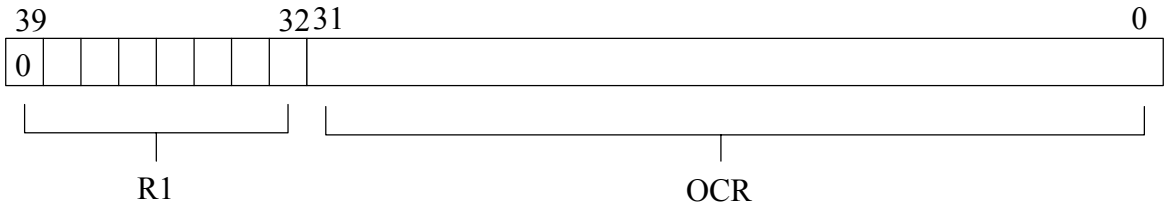


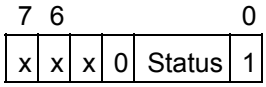
Figure 48: R3 Response Format

- **Formats R4 & R5**

Those response formats are reserved for I/O mode (refer to "SDIO Card Specification").

- **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

In case of any error (CRC or Write Error) during Write Multiple Block operation, the host shall stop the data transmission using CMD12. In case of Write Error (response ’110’) the host may send CMD13 (SEND_STATUS) in order to get the cause of the write problem. ACMD22 can be used to find the number of well written write blocks.

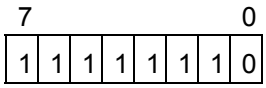
7.3.3 Data Tokens

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

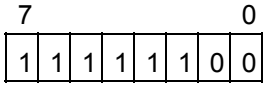


- Bytes 2-513 (depends on the data block length): User data
- Last two bytes: 16 bit CRC.

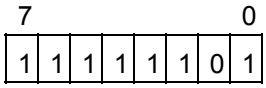
For Multiple Block Write operation:

- 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:

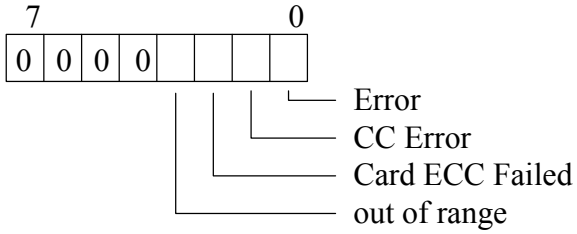


Figure 49: Data Error Token

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 multiple 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 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
CSD_OVERWRITE	R2	E R X	'0'= no error '1'= error	can be either one of the following errors: - 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
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.	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

Identifier	Included in resp	Type[1]	Value	Description	Clear Condition[2]
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.

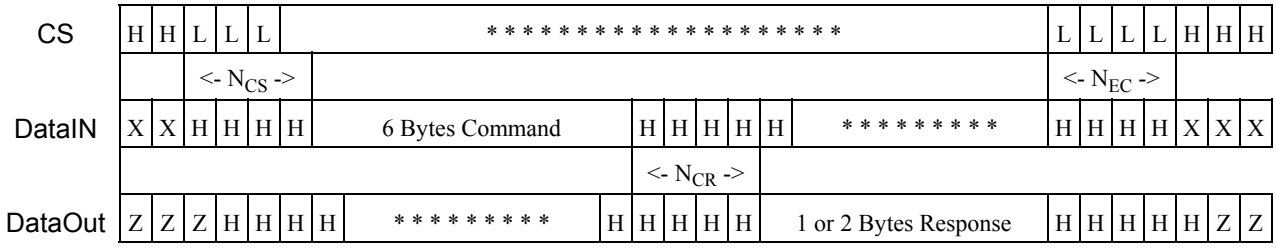


Figure 50: Basic command response

- **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.

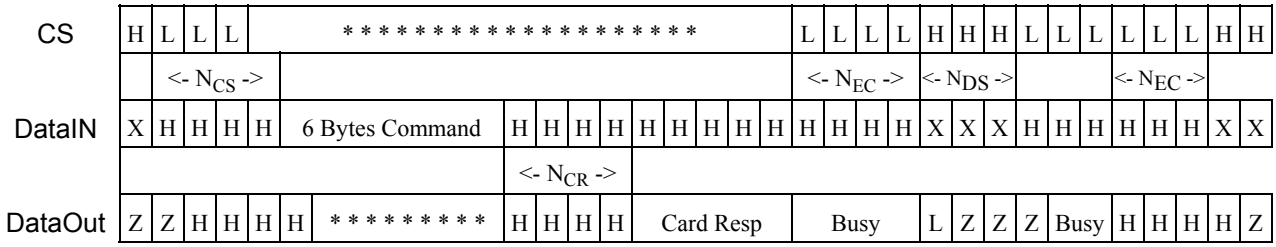


Figure 51: Command response with busy indication (R1b)

- **Card Response to Host Command**

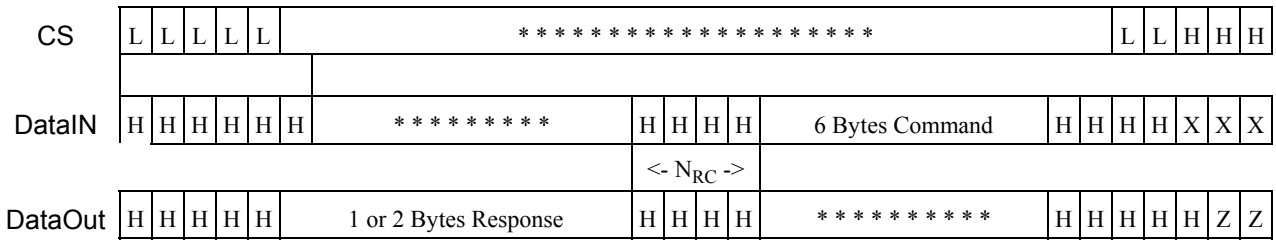


Figure 52: Timing between card response to new host command

7.5.2 Data read

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

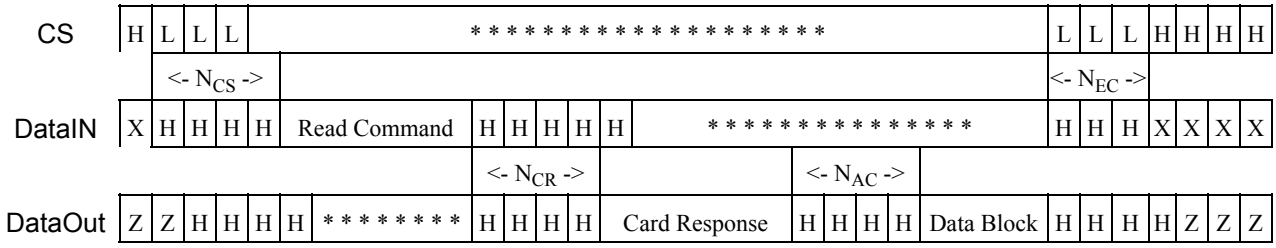


Figure 53: Read Single Block operations - bus timing

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

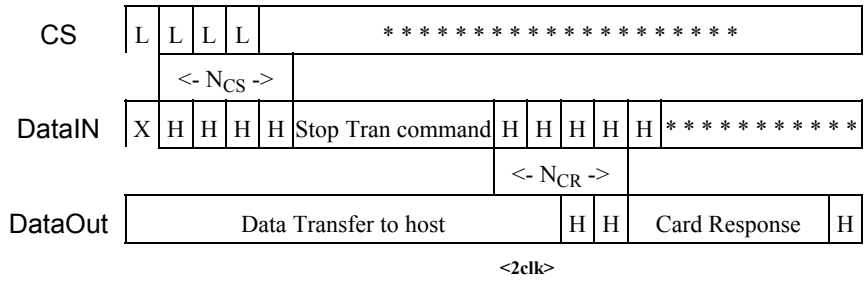


Figure 54: Stop Transmission in Read Multiple Block

- **Reading the CSD or CID register**

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

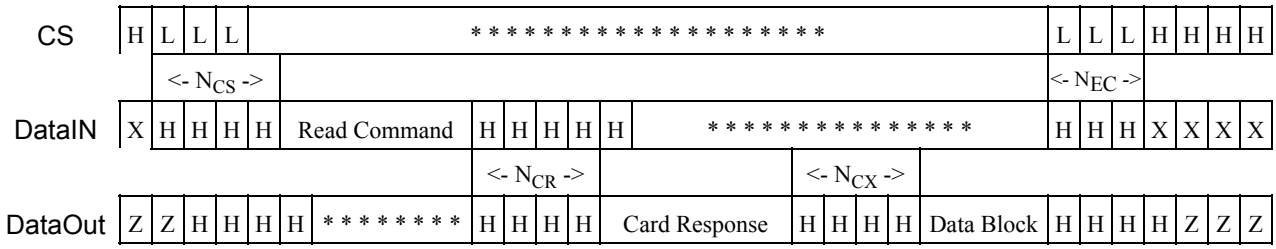


Figure 55: Read CSD / CID - bus timing

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.

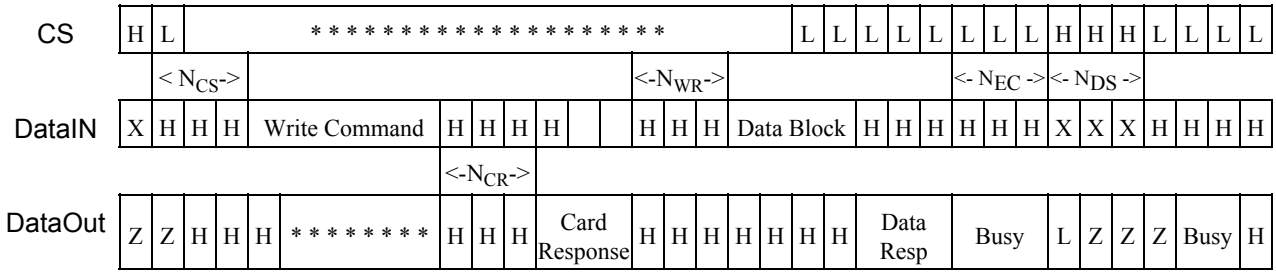
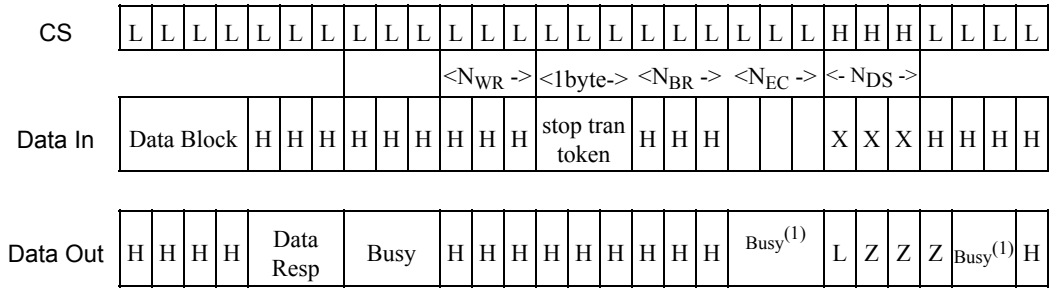


Figure 56: Write operation - bus timing

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



(1) The Busy may appear within N_{BR} clocks after Stop Tran Token. If there is no Busy the host may continue to the next command.

Figure 57: 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]	1	spec. in the CSD	8 clock cycles
N_{WR}	1	-	8 clock cycles
N_{EC}	0	-	8 clock cycles

1)The maximum read access time shall be calculated by host as follows:

Nac(max)= 100 ((TAAC * fpp) + (100 * NSAC)) ;

fpp is the interface clock rate and TAAC & NSAC are given in the CSD (Chapter 5.3).

	Min	Max	Unit
N_{DS}	0	-	8 clock cycles
N_{BR}	0	1	8 clock cycles
Ncx	0	8	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 is omitted from the simplified version of the physical layer specification.

9. Appendix

This Chapter is omitted from the simplified version of the physical layer specification.

10. Abbreviations and terms

block	a number of bytes, basic data transfer unit
broadcast	a command sent to all cards on the SD bus
Blocklen	Block Length set by CMD16
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

- This page was left empty intentionally -