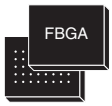
NAND08GAH0A
NAND16GAH0D

1 Gbyte, 2 Gbyte, 1.8 V/3 V supply, NAND Flash memories with MultiMediaCard™ interface

Preliminary Data

Features

- Packaged NAND Flash memory with MultiMediaCard interface
- 1, 2 Gbytes of formatted data storage
- eMMC/MultiMediaCard system specification, compliant with V4.1
- Full backward compatibilty with previous MultiMediaCard system specification
- Bus mode
  - High-speed MultiMediaCard protocol
  - SPI protocol
  - Three different data bus widths:1 bit, 4 bits, 8 bits
  - Data transfer rate: up to 52 Mbyte/s
- Operating voltage range:
  - V_{CCQ} =1.8 V/3 V
  - V_{CC} = 3 V
- Supported clock frequencies: 0 to 52 MHz
- Multiple Block Read (x 8 at 52 MHz): up to 3.5 Mbyte/s
- Multiple Block Write (x 8 at 52 MHz): up to 8.5 Mbyte/s
- Power dissipation
  - Standby current: down to 200 µA
  - Read current: down to 30 mA
  - Write current: down to 30 mA



LFBGA169 12 x 16 x 1.4 mm (ZA)

- Error free memory access
  - Internal enhanced data management algorithm (wear levelling, bad block management, garbage collection)
  - Internal error correction code
- Data integrity
  - Data reliability: less than 1 non-recoverable error per 10^{14} bits read
  - Endurance: more that 2,000,000 Program/Erase cycles
- Security
  - Password protection of data
  - Built-in write protection (permanent or temporary)

Contents

List of tables

List of figures

1 Description

NAND08GAH0A and NAND16GAH0D are embedded Flash memory storage solutions with MultiMediaCard interface (eMMC™). The eMMC™ was developed for universal low cost data storage and communication media. They can be considered as high speed MultiMediaCards embedded in LFBGA169 12 x 16 x 1.4 mm, 0.5 mm pitch package instead of an MMC. The devices are fully compatible with MMC bus and hosts.

NAND08GAH0A and NAND16GAH0D communications are made through an advanced 13-pin bus. The bus can be either 1-bit, 4-bit, or 8-bit bus width. The devices operate in high-speed mode at clock frequencies equal or higher than 20 MHz. The communication protocol is defined as a part of this MMC standard and referred to as MultiMediaCard mode. For compatibility with existing controllers the devices may offer, in addition to the MultiMediaCard mode, an alternate communication protocol which is based on the SPI standard.

The devices are designed to cover a wide area of applications such as smart phones, cameras, organizers, PDA, digital recorders, MP3 players, pagers, electronic toys, etc. They feature high performance, low power consumption, low cost and high density.

To meet the requirements of embedded high density storage media and mobile applications, Numonyx NAND08GAH0A and NAND16GAH0D support both 3 V supply voltage (V_{CC}), and 1.8 V/3 V input/output voltage (V_{CCQ}).

The devices have a built-in intelligent controller which manages interface protocols, data storage and retrieval, wear leveling, bad block management, garbage collection, internal ECC.

In order to meet environmental requirements, Numonyx offers the NAND08GAH0A and NAND16GAH0D in ECOPACK® packages. ECOPACK packages are Lead-free. The category of second Level Interconnect is marked on the package and on the inner box label, in compliance with JEDEC Standard JESD97. The maximum ratings related to soldering conditions are also marked on the inner box label. ECOPACK is an Numonyx trademark.

The system performance and characteristics are given in *Table 1*, *Table 2*, *Table 3*, and *Table 4*.

Related documentation

- eMMC™/MultiMediaCard system specification, version 4.1.

Table 1. System performance

System performance	Typical value		Unit
	NAND08GAH0A	NAND16GAH0D	
Reset to Ready	100	100	ms
Multiple Block Read	8.9	8.5	Mbyte/s
Single Block Read	1.0	1.0	Mbyte/s
Multiple Block Write	2.8	3.5	Mbyte/s
Single Block Write	0.1	0.1	Mbyte/s

Table 2. Current consumption

Operation	Test conditions	Current consumption				Unit
		NAND08GAH0A		NAND16GAH0D		
		Typ.	Max.	Typ.	Max.	
Read	V_{CC}= 3 V±5% V_{CCQ}=3 V±5% or 1.8 V±5%	20		25		mA
Write		40		60		mA
Standby	V_{CC}= 3 V±5%	10		20		
	V_{CCQ}=3 V±5% or 1.8 V±5%	100	200	100	200	µA

Table 3. System reliability and maintenance

MTBF	> 3 million hours
Preventive maintenance	None
Data reliability	less than 1 non-recoverable error per 10^{14} bits read
Endurance	2 000 000

Table 4. Communication channel performance

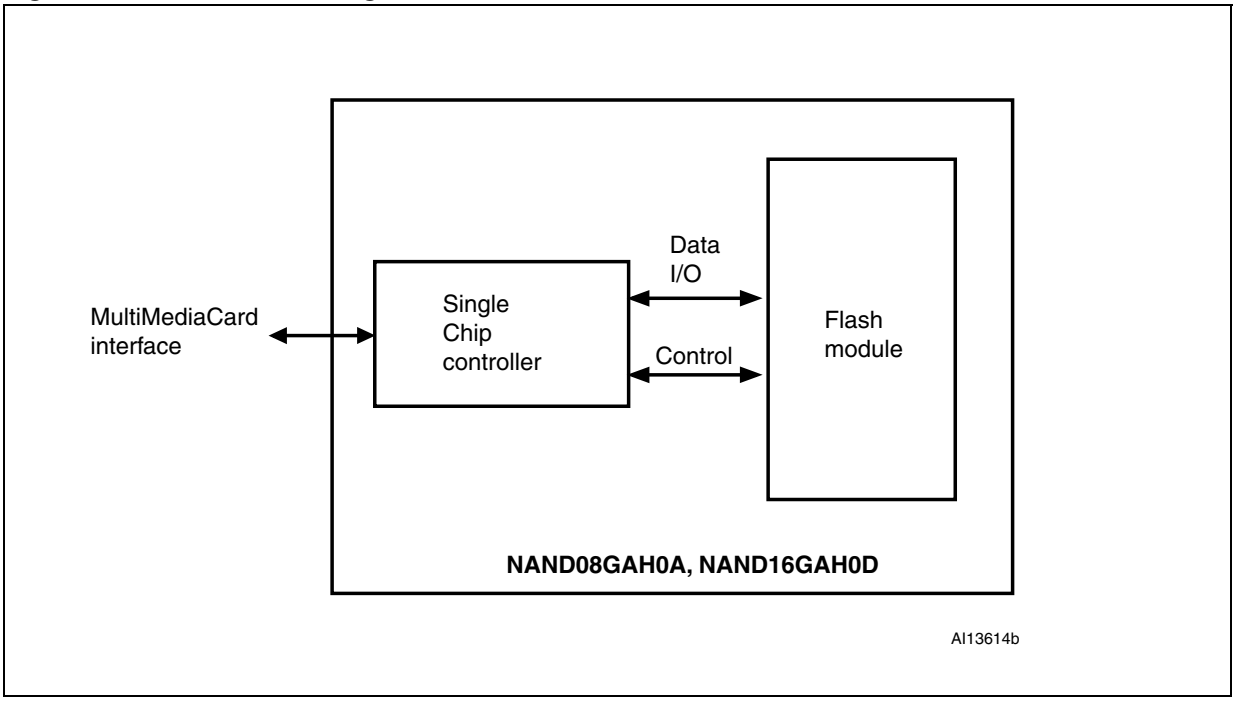
MultiMediaCard communication channel performance
Three-wire serial data bus (Clock, command, data)
Variable clock rate 0, 26, 52 MHz
Easy card identification
Error protected data transfer
Sequential and single/multiple block oriented data transfer

2 Device physical description

The NAND08GAH0A and NAND16GAH0D contain a single chip controller and Flash memory module, see *Figure 1: Device block diagram*. The microcontroller interfaces with a host system allowing data to be written to and read from the Flash memory module. The controller allows the host to be independent from details of erasing and programming the Flash memory.

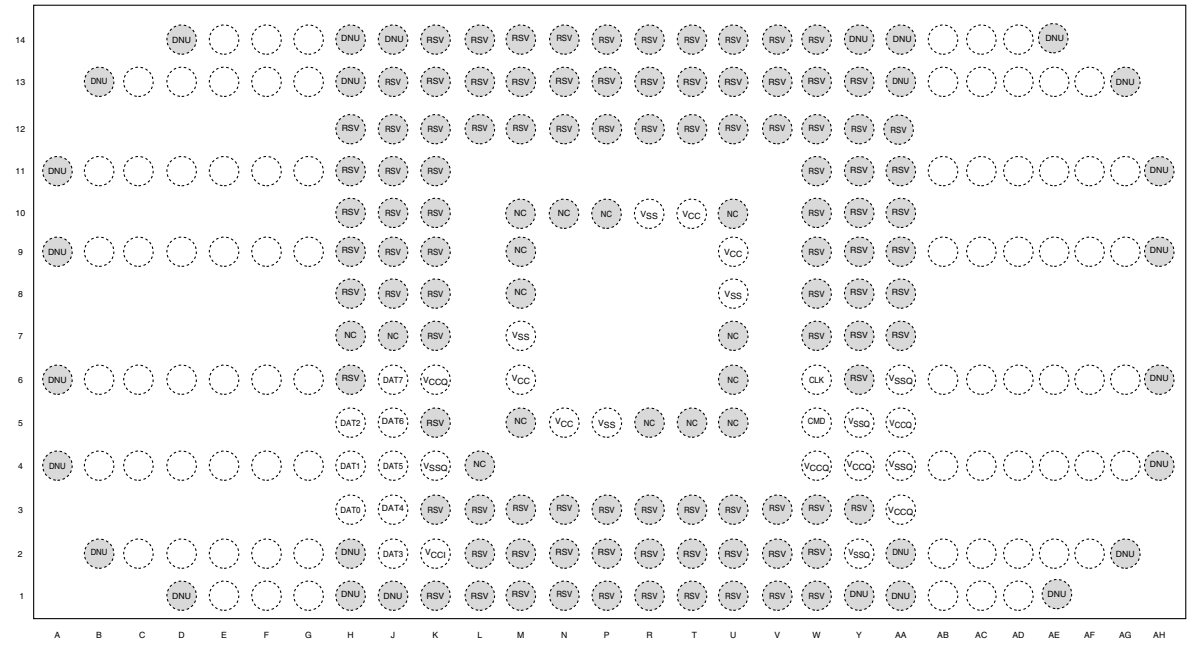
Figure 2 shows the package connections. See *Table 5: Signal names* for the description of the signals corresponding to the balls.

Figure 1. Device block diagram



2.1 Package connections

Figure 2. LFBGA169 package connections (top view through package)



1. The ball corresponding to V_{CCI} must be decoupled with capacitance C5 (see *Table 6*).

2.2 Form factor

The ball diameter, d, and the ball pitch, p, for LFBGA169 12 x 16 x 1.4 mm package are:

- d = 0.30 mm (solder ball diameter)
- p = 0.5 mm (ball pitch)

Figure 3. Form factor

3 Memory array partitioning

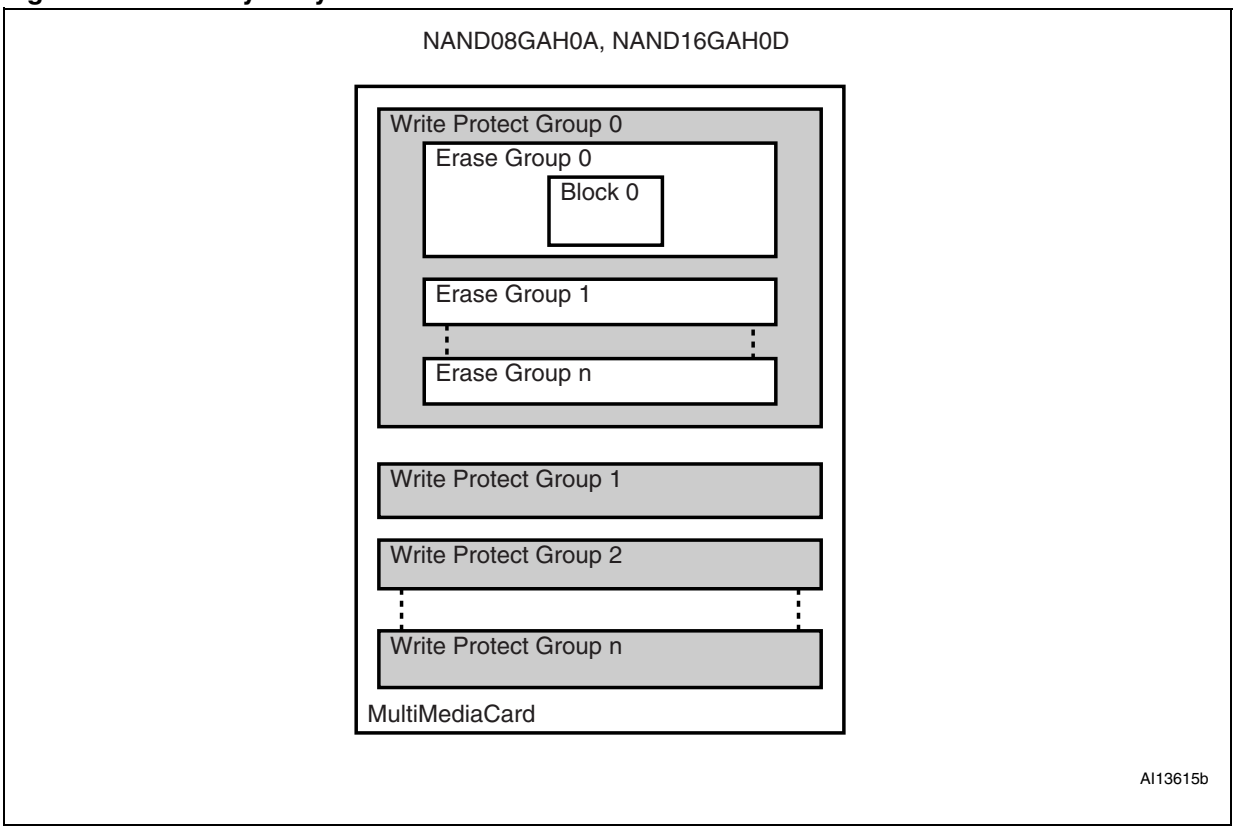
The basic unit of data transfer to/from the device is one byte. All data transfer operations which require a block size always define block lengths as integer multiples of bytes. Some special functions need other partition granularity.

For block oriented commands, the following definitions are used:

- **Block**: the unit which is related to the block oriented read and write commands. Its size is the number of bytes which are transferred when one block command is issued 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 register.
- **Erase Group**: the unit which is related to special erase and write commands defined for R/W cards. Its size is the smallest number of consecutive write blocks which can be addressed for erase. The size of the Erase Group depends on each device and is stored in the CSD.
- **Write Protect Group**: the smallest unit that may be individually write protected. Its size is defined in units of erase groups. The size of a WP-group depends on each device and is stored in the CSD.

Figure 4 shows NAND08GAH0A and NAND16GAH0D memory array organization.

Figure 4. Memory array structure



1. n = number of last Erase Group or last Write Protect Group.

4 MultiMediaCard interface

The signal/pin assignments are listed in *Table 5*. Refer to this table in conjunction with *Figure 2* and *Figure 3: Form factor*.

4.1 Signals description

4.1.1 Clock (CLK)

The Clock input, CLK, is used to synchronize the memory to the host during command and data transfers. Each clock cycle gates one bit on the command and on all the data lines. The Clock frequency, f_{PP}, may vary between zero and the maximum clock frequency.

4.1.2 Command (CMD)

The CMD signal is a bidirectional command channel used for device initialization and command transfer. The CMD signal has two operating modes: open-drain and push-pull. The open-drain mode is used for initialization, while the push-pull mode is used for fast command transfer. Commands are sent by the MultiMediaCard bus master (or host) to the device who answers by sending back responses.

4.1.3 Input/outputs (DAT0-DAT7)

DAT0 to DAT7 are bidirectional data channels. The signals operate in push-pull mode. The NAND08GAH0A and NAND16GAH0D include internal pull ups for all data lines. These signals cannot be driven simultaneously by the host and the NAND08GAH0A device.

By default, after power-up or hardware reset, only DAT0 is used for data transfers. The host can configure the device to use a wider data bus, DAT0, DAT0-DAT3 or DAT0-DAT7, for data transfer.

4.1.4 V_{CC} core supply voltage

V_{CC} provides the power supply to the internal core of the memory device. It is the main power supply for all operations (read, program and erase).

4.1.5 V_{SS} ground

Ground, V_{SS}, is the reference for the power supply. It must be connected to the system ground.

4.1.6 V_{CCQ} input/output supply voltage

V_{CCQ} provides the power supply to the I/O pins and enables all outputs to be powered independently from V_{CC}.

The input/output voltage (V_{CCQ}) can be either within 1.65/1.7 V and 1.95 V (low voltage range) or 2.7 V and 3.6 V (high voltage range).

4.1.7 V_{SSQ} supply voltage

V_{SSQ} ground is the reference for the input/output circuitry driven by V_{CCQ}.

Table 5. Signal names

Name	Type(1)	Description
DAT0	I/O (PP)	Data
DAT1	I/O (PP)	Data
DAT2	I/O (PP)	Data
DAT3	I/O (PP)	Data
DAT4	I/O (PP)	Data
DAT5	I/O (PP)	Data
DAT6	I/O (PP)	Data
DAT7	I/O (PP)	Data
CMD	I/O (OD or PP)	Command
CLK	I (PP)	Clock
V_{CCQ}		Input/output power supply
V_{CC}		Core power supply
V_{SSQ}		Input/output ground
V_{CCI}	I	Must be decoupled with capacitance C5 (see *Table 6*)
V_{CC}		Core power supply
NC	NC	Not connected(2)
RSV	RSV	Reserved for future use(2)
DNU	DNU	Do not use(2)

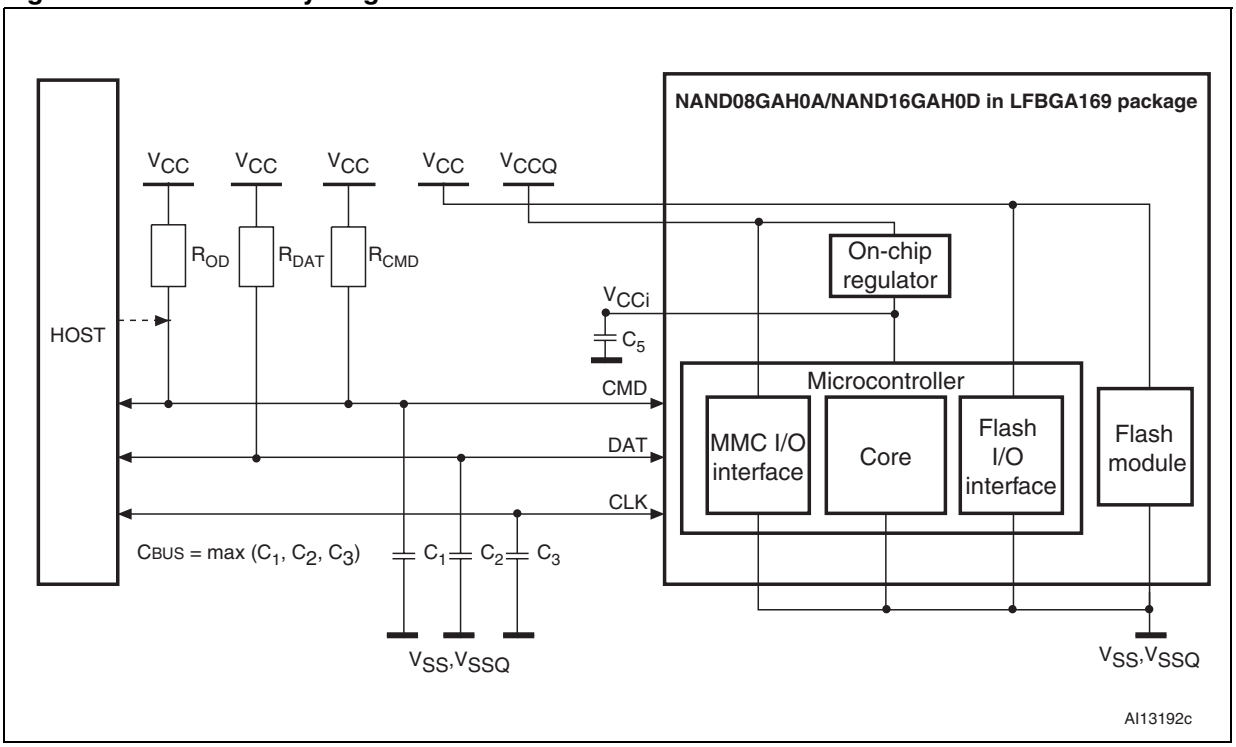
1. I: input; O: output, OD: open drain, PP: push-pull.
2. NC, RSV and DNU pins can be connected to ground or left floating.

4.2 Bus topology

Figure 5 shows the bus circuitry required for the device. The resistor R_{OD} is switched on and off, synchronously, by the host for the open-drain and push-pull mode transitions. R_{DAT} and R_{CMD} are pull-up resistors that are used to stop the CMD and DAT signals floating when no device is inserted or when all the device drivers are in a high impedance state.

A constant current source can replace R_{OD} and achieve a better performance (constant slopes for the signal's rising and falling edges). If the host does not allow a switchable R_{OD} to be implemented, a fixed R_{CMD} can be used. Consequently, the maximum operating frequency in the open drain mode has to be reduced if the value of R_{CMD} is higher than the minimum given in *Table 6: Bus operating conditions*.

Figure 5. Bus circuitry diagram



1. See *Table 6* for the values of R_{OD}, RD_{AT}, RCM_D, C1, C2, C3, and C5.

4.3 Power-up and power-down

4.3.1 Power-up

The power-up and hot insertion (e.g. inserting the device when the bus is operating) are handled locally in each device and in the bus master.

V_{CC} must be powered up before or simultaneously with V_{CCQ}. No delay must be respected between V_{CC} and V_{CCQ} ramp up (see *Figure 8*).

After power-up the device enters the Idle state until the CMD1 command is received. The bus master must get the device out of the Idle state. Since the power-up time and the supply voltage ramp up time depend on application parameters such as the bus length and the power supply unit, the host must ensure that the supply voltage has reached the operating level specified in CMD1 before issuing a CMD1 command.

CMD1 is a special synchronization command for the host to poll the device states until the power-up is completed correctly. The response of CMD1 contains a busy flag which indicates that a device is not ready. The host has to wait until this flag is cleared. The time for this flag to be cleared is the Identification delay (see *Figure 6*).

After power-up the host starts the clock and sends the initializing sequence on the CMD line (see *Figure 6*). This sequence is a contiguous stream of logic 1 s. The sequence length is either 1 ms, 74 clock cycles or the supply ramp up time, whichever is the longest. The additional 10 clocks (after the 64 clocks after which the device should be ready for communication) are provided to avoid power-up synchronization problems.

The device ignores all commands until the commands CMD1, CMD2 are issued and the RCA of the device is initialized.

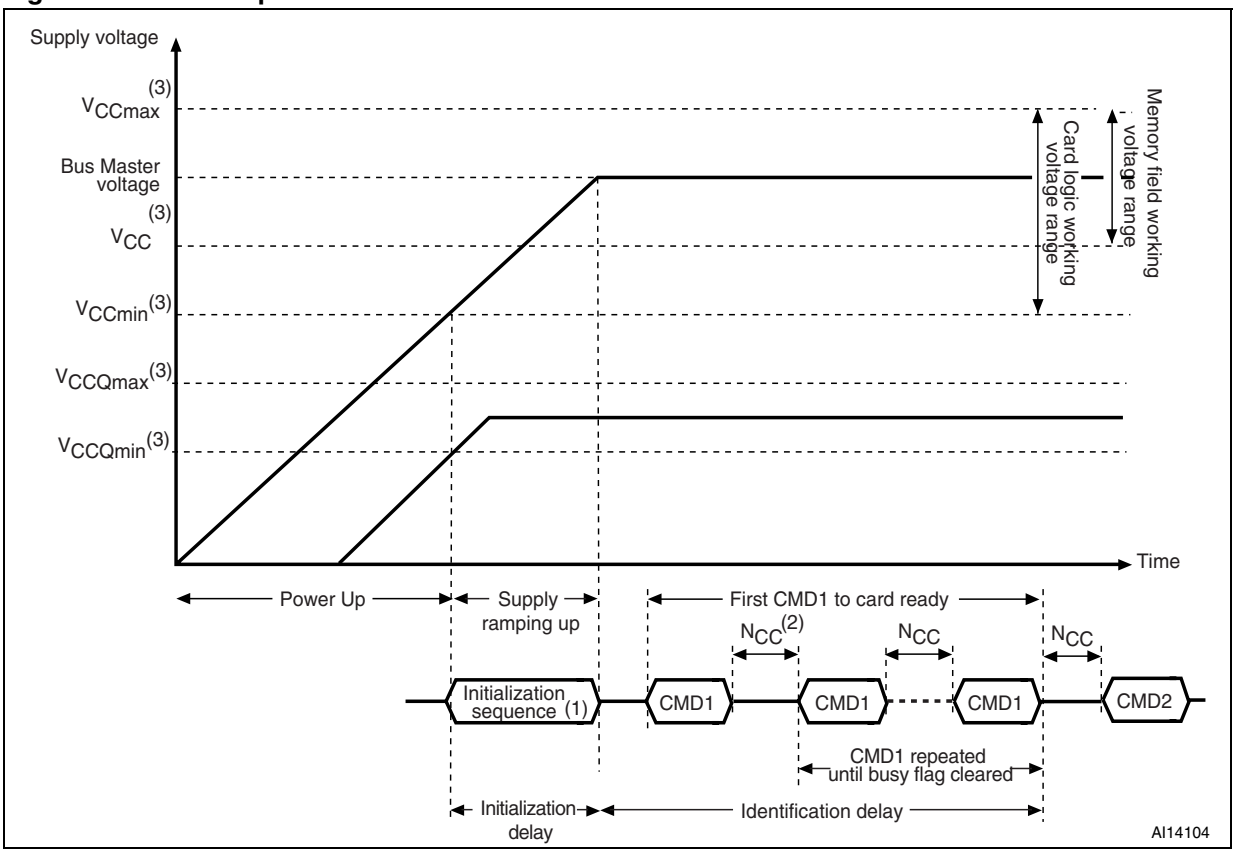
The initialization delay is relevant only after power-up, the identification delay is relevant for both power-up and hot insertion.

After power-up, the maximum initial load the NAND08GAH0A and NAND16GAH0D can present on the V_{CC} line is C4, in parallel with a minimum of R4. During operation, device capacitance on the V_{CC} line must not exceed 10 µF.

4.3.2 Power-down

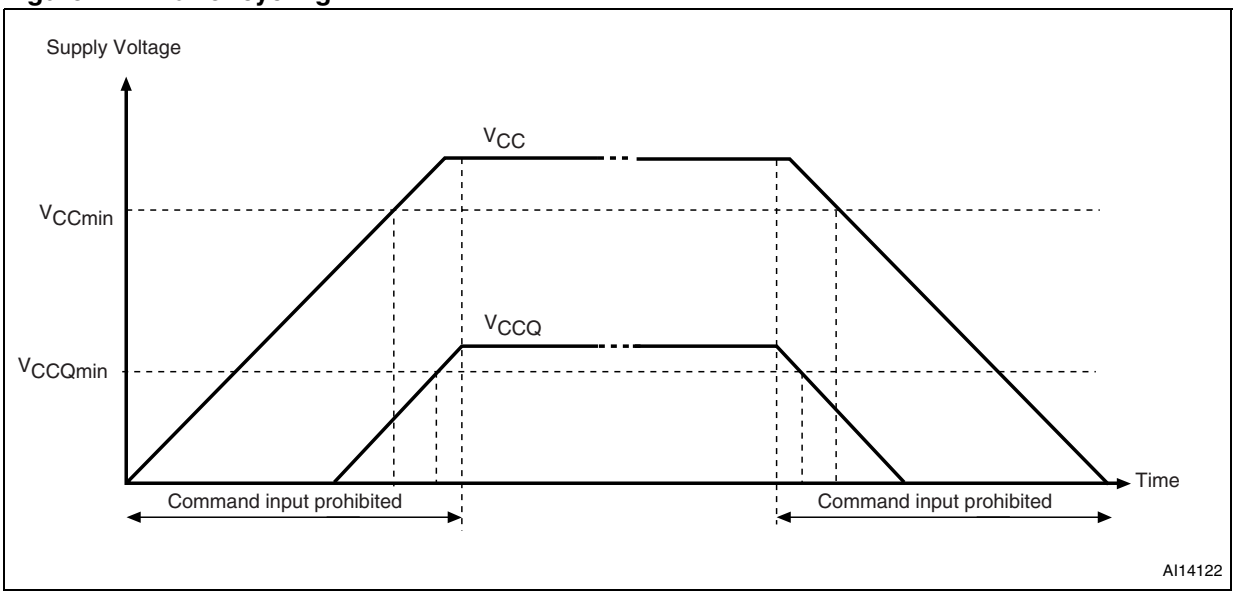
At power-down, V_{CCQ} must go Low before or simultaneously with V_{CC} going Low (see *Figure 9*). Commands from the bus master are accepted till V_{CCQ} and V_{CC} start to ramp down.

Figure 6. Power-up



1. The initialization sequence is a contiguous stream of logic 1's. Its length is either 1 ms, 74 clocks or the supply ramp up time, whichever is the longest. The device shall complete its initialization within 1 second from the first CMD1 with a valid V range.
2. N_{CC} is the number of clock cycles. Refer to *Table 61* for its value.
3. Refer to *Section 8.1: Operation conditions register (OCR)* for details on voltage ranges.

Figure 7. Power cycling



4.4 Electrical specifications

Table 6 defines the bus operating conditions for the device.

The total capacitance C_L of each line of the bus is given by the below equation:

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

where C_{HOST} is the bus master capacitance, C_{BUS} the bus capacitance itself and C_{CARD} the capacitance of the device connected to this line. The sum of the host and bus capacitance, CHOST+CBUS, must not to exceed 20 pF.

As the bus can be supplied with a variable supply voltage, all bus signal levels are related to the supply voltage (see *Figure 8*, *Table 7*, and *Table 8*).

Table 6. Bus operating conditions(1)(2)

Symbol	Parameter	Min	Max	Unit
	Peak voltage on all lines	-0.5	3.6	V
	Input leakage current on all inputs (before initialization sequence and/or internal pull up resistors connected)(3)	-100	100	µA
	Input leakage current on all inputs (after initialization sequence and/or internal pull up resistors connected)(3)	-10	10	µA
	Output leakage current on all outputs (before initialization sequence)	-100	100	µA
	Output leakage current on all outputs (after initialization sequence)	-10	10	µA
V_{CCQ}	Low supply-voltage range (MultiMediaCard v. 4.1)	1.65	1.95	V
	High supply-voltage range	2.7	3.6	
V_{CC}	Input/output supply voltage	2.7	3.6	V
V_{SS}	Supply voltage ground	-0.5	0.5	V
R_{DAT}	Pull-up resistance (to prevent bus floating)	50	100	kΩ
R_{CMD}	Pull-up resistance (to prevent bus floating)	4.7	100	kΩ
R_{INT}	Internal pull up resistance DAT1-DAT7 (to prevent unconnected line floating)	50	150	kΩ
R4	Load resistance on V_{CC} line after power-up or hot insertion	330		Ω
C4	Load capacitance on V_{CC} line after power-up or hot insertion		10	µF
C_L	Bus signal line capacitance		30	pF
C_{CARD}	Single card capacitance		7	pF
C1	Load capacitance on CMD input	TBD	TBD	pF
C2	Load capacitance on DAT input	TBD	TBD	pF
C3	Load capacitance on CLK input	TBD	TBD	pF
C5	Decoupling capacitance on V_{CCI} input		1	pF
	Maximum signal line inductance ($f_{PP} \leq 52$ MHz)		16	nH

1. The current consumption of the device for the different configurations is defined in the POWER_CLASS field of the EXT_CSD register (see *Section 8.4*).
2. TBD stands for 'to be defined'.
3. See *Section 4.3: Power-up and power-down*.

Figure 8. Bus signal levels

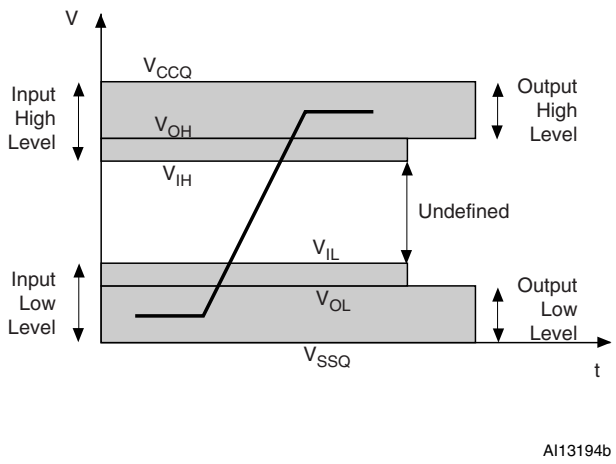


Table 7. Open-drain mode bus signal level(1)

Symbol	Parameter	Conditions	Min	Max	Unit
V_{OH}	Output High voltage	I_{OH} = –100 µA	V_{CCQ}–0.2		V
V_{OL}	Output Low voltage	I_{OL} = 2 mA		0.3	V

1. The values of V_{IH} and V_{IL} are identical in Open-drain and Push-pull mode (see *Table 8: Push-pull mode bus signal level*).

Table 8. Push-pull mode bus signal level(1)

Symbol	Parameter	Conditions	V_{CCQ}				Unit
			1.65 to 1.95 V		2.7 to 3.6 V		
			Min	Max	Min	Max	
V_{OH}	Output High voltage	I_{OH} = –100 µA at V_{CCQ}min	V_{CCQ}–0.2		0.75 V_{CCQ}		V
V_{OL}	Output Low voltage	I_{OL} = 100 µA at V_{CCQ}min		0.2		0.125V_{CCQ}	V
V_{IH}	Input High voltage		0.7V_{CCQ}	V_{CC} + 0.3	0.625V_{CCQ}	V_{CCQ} + 0.3	V
V_{IL}	Input Low voltage		V_{SSQ}–0.3	0.3V_{CCQ}	V_{SS}–0.3	0.25V_{CCQ}	V

1. In accordance with the JEDEC specification JESD8-1A, the device input and output voltages should be within the specified ranges for the whole V_{CC} range.

Figure 9. Timing diagram data input/output referenced to clock

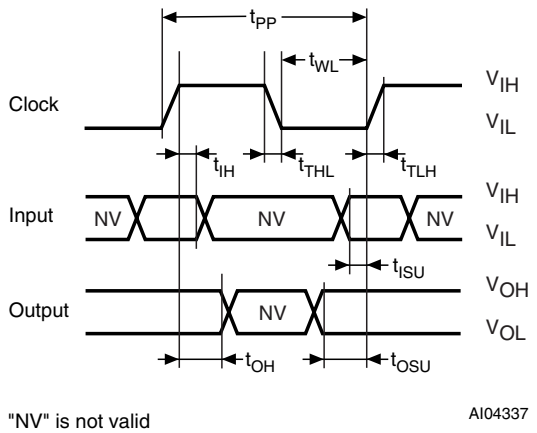


Table 9. Bus AC timings

Symbol	Parameter	20 MHz		26/52 MHz(1)		Unit
		Min	Max	Min	Max	
Clock CLK(2)						
f_{PP}	Clock frequency Data Transfer mode (PP)(3)(4)	0	20	0	26/52(1)	MHz
f_{OD}	Clock frequency Identification mode (OD)(5)	0	400	0	400	kHz
t_{WL}	Clock Low time(3)	10		6.5		ns
t_{TLH}	Clock Rise time(3)(6)		10		3	ns
t_{THL}	Clock Fall time(3)(6)		10		3	ns
Input CMD, DAT (referenced to CLK)						
t_{ISU}	Input Set-up time(3)	3		3		ns
t_{IH}	Input Hold time(3)	3		3		ns
Output CMD, DAT (referenced to CLK)						
t_{OSU}	Output Set-up time(3)	13.1		5		ns
t_{OH}	Output Hold time(3)	9.7		5		ns

1. f_{PP}=52 MHz is available for V_{CC}=2.7 to 3.6 V
2. All timing values are measured relatively to 50% of the voltage level.
3. Parameter measured with a bus line load capacitance, C_L, lower than 30 pF.
4. f_{PP} is measured with a tolerance of 100 KHz.
5. f_{OD} is measured with a tolerance of 20 KHz.
6. Rise and fall times are measured from 10% to 90% of the voltage level for High clock frequencies (26 and 52 MHz). They are measured from V_{IL}(max) to V_{IH} (min) of the voltage level for standard clock frequency (20 MHz).

5 High speed MultiMediaCard operation

5.1 Overview

All communication between the host and the device is controlled by the host (master). The host sends two types of command:

- Broadcast commands intended for all MultimediaCard devices. They are kept for backwards compatibility to previous MultiMediaCard systems, where more than one device was allowed on the bus.
- Addressed (point-to-point) commands which are sent to the addressed device and cause it to respond.

A general overview of the command flow is shown in *Figure 10* for the Card Identification mode and in *Figure 12* for the Data Transfer mode. The commands are listed in the command tables (*Table 20*, *Table 21*, *Table 22*, *Table 23*, and *Table 24*). The relation between the current device state, the received command and the resulting state are listed in *Table 27*. The different operating modes are presented in the following sections, together with the restrictions for controlling the clock signal, and device commands, state transitions and timings.

Three operating modes are defined for MultiMediaCard devices:

- **Card Identification mode**
  The host enter Card Identification mode after reset and while it is looking for new devices connected to the bus.
  MultiMediaCard devices enter this mode after reset until the SET_RCA command (CMD3) is received.
- **Interrupt mode (**not supported)
- **Data Transfer mode**
  The device enters Data Transfer mode once an RCA is assigned to it.
  The host will enter data transfer mode after identifying all the devices on the bus.

Table 10 shows the relations between bus modes, operation modes and device states. Each state in the device state diagrams is associated with a bus mode and an operation mode (*Figure 10* and *Figure 12*).

A command received with an incorrect CRC is ignored. If the command was issued during an operation (for example block read), the device continues the operation until it receives a correct host command.

Table 10. Bus modes overview

Device state	Operation mode	Bus mode
Inactive (ina)	Inactive	Open-drain
Idle	Card Identification mode	
Ready		
Identification (ident)		
Standby (stby)	Data Transfer mode	Push-pull
Transfer (tran)		
Bus state test (btst)		
Sending-data (data)		
Receiving-data (rcv)		
Programming (pgr)		
Disconnect		
Wait-IRQ (irq)	Interrupt mode	Push-pull

5.2 Card Identification mode

When in Card Identification mode, the host resets the device, validates the operating voltage range and the access mode, identifies the device and assigns a Relative Card Address (RCA) to it.

In Card Identification mode all data communications are performed using only the command line (CMD).

5.2.1 Card reset

After power-up, the device is in Idle state and defaults to operate in MultiMediaCard mode, even if it was previously in the Inactive state.

The GO_IDLE_STATE (CMD0) command performs a software reset and puts the device in Idle state. It is also used to switch the device into SPI mode (see *Section 10: Serial peripheral interface (SPI) mode*).

After power-up or a CMD0 command, all outputs are high impedance, and the device is initialized with a default RCA (0x0001) and default Driver Stage Register (DSR) settings. The host starts the device identification process in open-drain mode with the clock frequency set to the identification clock frequency f_{OD} (see *Table 9: Bus AC timings*).

CMD0 is valid in all states, with the exception of the Inactive state. While in Inactive state the device does not accept CMD0 commands unless it is used to switch the device into SPI mode.

5.2.2 Input/output voltage range validation

All device communicate with the host using an input/output voltage in the V_{CCQmin} and V_{CCQmax} range. In Card Identification mode, the minimum and maximum values for V_{CCQ} are defined in the operation condition register (OCR) and may not cover the whole range.

The SEND_OP_COND (CMD1) command is designed to provide hosts with a mechanism to identify and reject devices which do not match the desired V_{CCQ} range. This is performed by the host sending the required V_{CCQ} range as the operand of the CMD1 command (see *Section 8.1: Operation conditions register (OCR)*). If the device can not perform data transfer in the specified range, it switches into the Inactive state. Otherwise, the device answers sending back its V_{CCQ} range.

By omitting the voltage range when issuing the CMD1 command (by setting CMD1 argument to '0'), the host queries the device about its input/output voltage range. This bus query should be used if the host is able to select a common voltage range, or if the application needs to be notified of non usable devices connected to the bus. The host then chooses an operating voltage, and reissues the CMD1 together with this condition, sending incompatible devices into the Inactive state.

5.2.3 From Busy to Ready state

The busy flag in the CMD1 response can be used by the device to notify the host that the power-up/reset sequence is still ongoing and that the device is not ready for communication. In this case the host must reissue the CMD1 command until the busy flag is cleared.

During the initialization procedure, the host should not change the operating voltage range or access mode settings. Any change in the operating conditions is ignored by the device. If this case, the host must reset the device by issuing a CMD0 command, and restart the initialization sequence. However, a hardware reset must be performed for accessing devices that are already in the Inactive state.

The GO_INACTIVE_STATE (CMD15) command can be used to send an addressed device into the Inactive state. This command is used when the host explicitly wants to de-activate a device by changing its V_{CC} range into a range which is known not to be supported by this device.

5.2.4 Card Identification process

This process is valid when multiple MultiMediaCard devices are connected to the bus.

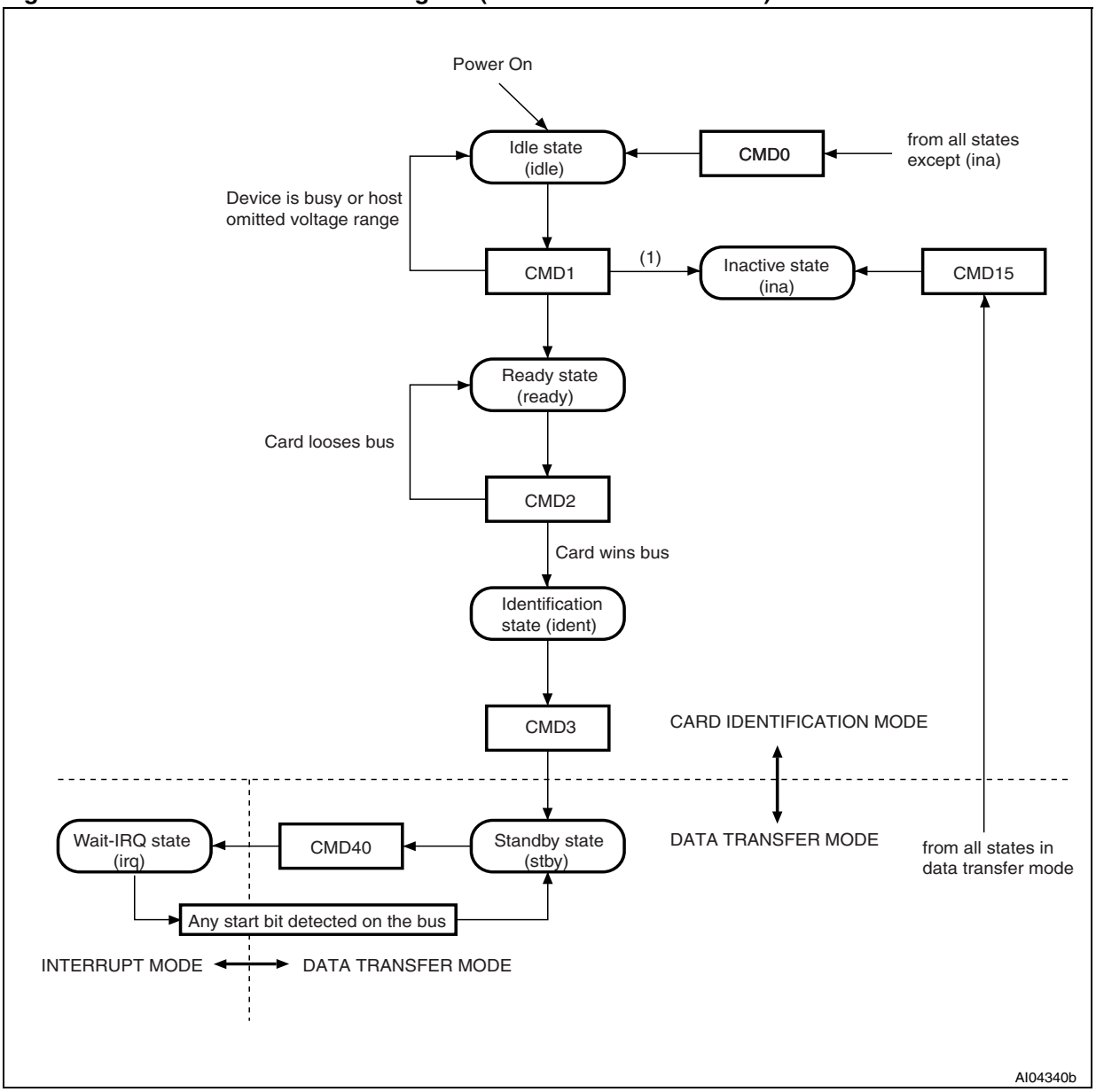
The host starts the card identification process in open-drain mode with the identification clock rate f_{OD} (see *Table 9: Bus AC timings*). The open drain driver stages on the CMD line allow parallel operation during card identification.

After the bus is activated, the host will request the devices to send its valid operating conditions (CMD1). The response to CMD1 is the 'wired and' operation on the condition restrictions of all devices in the system. Incompatible devices are sent into Inactive state. The host then issues the broadcast command CMD2 and asks all devices for their unique Card Identification (CID) number. All remaining unidentified devices simultaneously start sending their CID numbers serially, while monitoring their outgoing bit stream. The devices, whose outgoing CID bits do not match the corresponding bits on the command line, stop sending their CID immediately and wait for the next identification cycle (devices stay in the Ready state). Since CID numbers are unique for each device, there should be only one device which successfully sends its full CID-number to the host. This device then goes into the Identification state.

The host issues CMD3 to assign this device a relative card address (RCA) which will be used to address the device in future data transfer communication. Once the RCA is received the device goes to the Standby state and does not react to further identification cycles. The device also switches its output drivers from open-drain to push-pull.

The host repeats the identification process as long as it receives a response (CID) to its identification command (CMD2). When no more devices respond to this command, all devices have been identified.

Figure 10. MultiMediaCard state diagram (Card Identification mode)



1. Incompatible V_{CCQ} voltage range.
2. See *Table 10: Bus modes overview* for the definition of the abbreviated forms corresponding to the device state.

5.3 Data Transfer mode

The device enters data transfer mode once an RCA is assigned to it. When the device is in Standby mode, issuing the CMD7 command along with the RCA selects the device and puts it into the Transfer state.

The host enters Data Transfer mode after identifying all the MultiMediaCard devices on the bus. When all devices are in Standby state, communication over the CMD and DAT lines will be in push-pull mode (see *Table 10: Bus modes overview*).

The device supports two Read/Write modes as shown in *Figure 11: Data transfer formats*.

- Single Block mode
  - In this mode the host reads or writes one data block of a pre-specified length. The data block transmission is protected with a 16 bit Cyclic Redundancy Check (CRC).
- Multiple Block mode
  - This mode is similar to the single block mode, but the host can read/write multiple data blocks (all have the same length) which will be stored or retrieved from contiguous memory addresses.

The host issues CMD9 to obtain the Card Specific Data (CSD register). MultiMediaCard devices which already have an RCA do not respond to the identification command flow in this mode. Until the content of all CSD registers is known by the host, the f_{PP} clock rate must remain at f_{OD} because some devices may have operating frequency restrictions.

The relationship between the various operation modes is summarized in *Figure 12: MultiMediaCard state diagram (Data Transfer mode)*.

5.3.1 Active command set selection

By default, the device uses the MultiMediaCard standard command set after a power-up or software reset (CMD0). The host can change the active command set by issuing the SWITCH command (CMD6) with the ‘Command Set’ access mode selected.

The supported command sets, as well as the currently selected command set, are defined in the EXT_CSD register.

5.3.2 High speed mode selection

The device operates in high-speed mode (HS-MMC) at clock frequencies higher than 20 MHz.

The host must first check whether the Numonyx NAND08GAH0A and NAND16GAH0D comply with eMMC™/MultiMediaCard system specification version 4.1.

The high speed mode of the device must then be enables, before changing the clock frequency to a frequency higher than 20 MHz. This is done by using the SWITCH command to write 0x01 to the HS_TIMING byte, in the modes segment of the EXT_CSD register.

5.3.3 Power class selection

After checking whether the NAND08GAH0A and NAND16GAH0D complies with eMMC™/MultiMediaCard system specification version 4.0 or higher, the host can change the device power class.

After power-up or software reset (CMD0), the device defaults to operate in power class 0 which corresponds to the minimum current consumption for the card type (either Low or High V_{CCQ} voltage range).

The PWR_CL_ff_vvv bytes of the EXT_CSD register report the power consumption levels of the device, for a 4-bit or 8-bit bus width, at the supported clock frequencies (26 or 52 MHz). The host can read the PWR_CL_ff_vvv bytes by issuing a SEND_EXT_CSD command, and determine if it will allow the device to use a higher power class.

The power class can be changed by using the SWITCH command to program the POWER_CLASS Byte, in the modes segment of the EXT_CSD register.

The valid values for the EXT_CSD register are defined in (see *Section 8.4.3: PWR_CL_ff_vvv*). If the value programmed by the host is invalid, the POWER_CLASS byte remains unchanged and the SWITCH_ERROR bit is set.

5.3.4 Bus test procedure

The host can detect the bus functional lines by issuing CMD19 and CMD14 commands.

The following steps are required to test the bus functional signals:

1. The host must issue a CMD19 command, followed by a specific data pattern on each selected data lines (see *Table 11*). The data pattern sent by the host may optionally include a CRC16 checksum, which is ignored by the device. The data pattern to be sent per data line is defined in *Table 12*, *Table 13* and *Table 14*, according to the bus width.
2. The host must then requests the device to send back the reversed data pattern. This is done by issuing a CMD14 command. The device detects the start bit on DAT0 and synchronizes accordingly the reading of all data inputs. It ignores all data pattern bits except for the first two bits. The device buffer size consequently does not limit the maximum length of the data pattern. The minimum length of the data pattern is two bytes, of which the first two bits of each data line are sent back reversed by the device.
3. The host detects the bus functional lines by comparing the initial data pattern with the reversed pattern sent back by the device. The host ignores all bits except for the first two bits of the reversed data pattern. The length of the reversed data pattern is eight bytes and is always sent using all the device DAT lines (see *Table 12*, *Table 13* and *Table 14*). The reversed data pattern sent by the device may optionally include a CRC16 checksum, which is ignored by the host.

The device has internal pull-up resistor on DAT1-DAT7 lines. If the device is connected to 1-bit or 4-bit high-speed MMC system, the input value of the upper bits (e.g. DAT1-DAT7 or DAT4-DAT7) are detected as logic “1” by the device.

Table 11. Data format

Start bit	Data pattern	Checksum bit	End bit
0	1 0 x x x x ... x x	CRC16	1

Table 12. 1-bit bus test pattern

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

Table 13. 4-bit bus test pattern

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

Table 14. 8-bit bus test pattern

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

5.3.5 Bus width selection

After checking the bus functional lines, the host must change the bus width configuration accordingly. This is done by using the SWITCH command to program the BUS_WIDTH byte in the modes segment of the EXT_CSD register.

The BUS_WIDTH byte is write only.

By default (after power-up or software reset (CMD0)), the contents of the BUS_WIDTH byte is set to 0x00.

The valid values for this register are defined in *Section 8.4.11: BUS_WIDTH*.

If the value programmed by the host is invalid, the BUS_WIDTH byte remains unchanged and the SWITCH_ERROR bit is set.

Figure 11. Data transfer formats

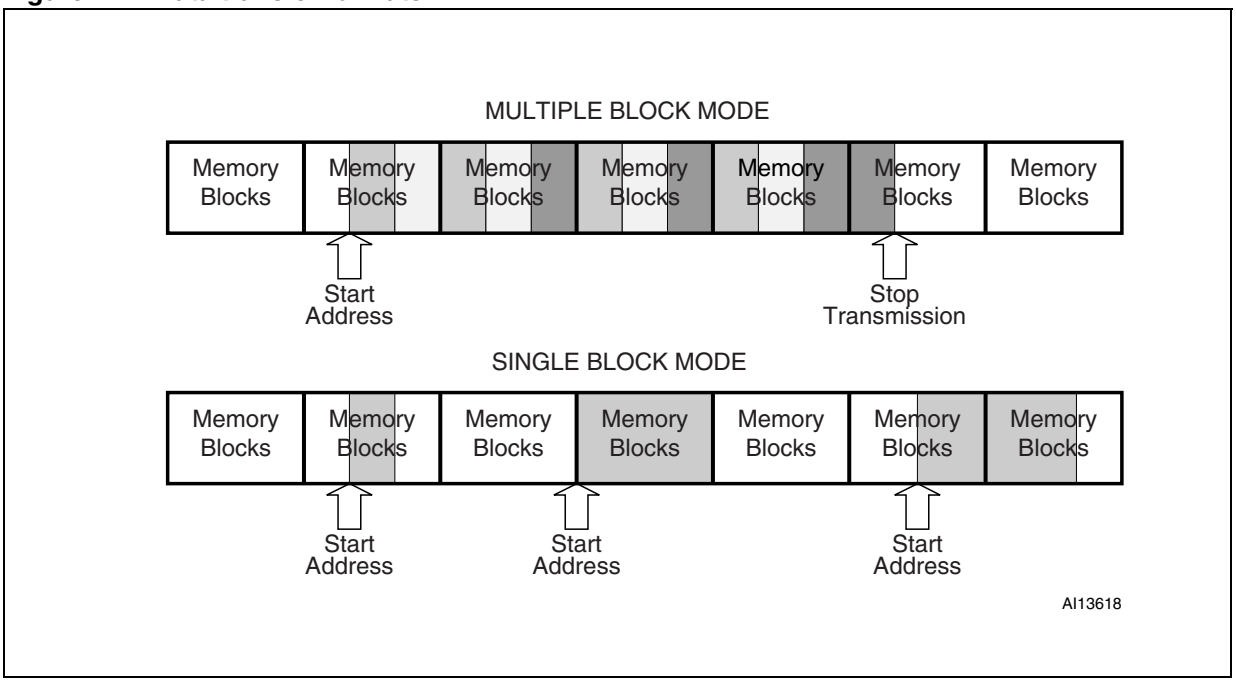
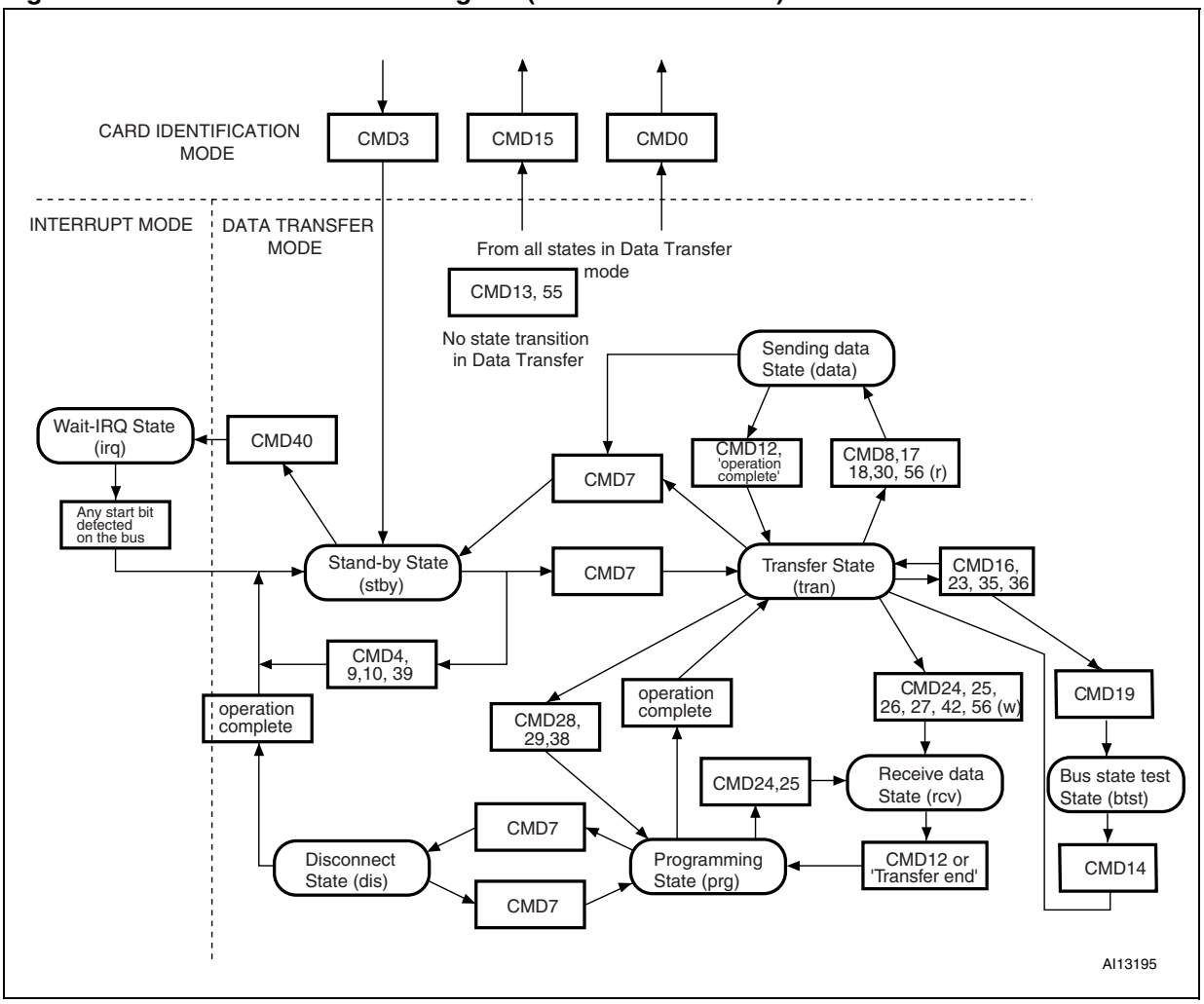


Figure 12. MultiMediaCard state diagram (Data Transfer mode)



1. See *Table 10: Bus modes overview* for the definition of the abbreviated forms corresponding to the device state.
2. ‘r’ and ‘w’ stand for read and write.
3. If the device was previously selected and was in Transfer state, its connection with the host is released and it moves back to the Standby state when a CMD7 command is issued along with any address different from the device own RCA.
4. Issuing the CMD7 command along with the reserved RCA 0x0000 returns to Standby state.
5. CMD7 commands issued along with the device RCA while the device is in Transfer state are ignored and may be treated as illegal commands.
6. After the device is assigned an RCA it will not respond to identification commands, CMD1, CMD2, or CMD3.
7. The SET_DSR (CMD4) broadcast command configures the device driver stages. It programs its DSR register according to the application bus length and the data transfer rate. The clock rate must then be switched from f_{OD} to f_{PP}.
8. The busy (Dat0=Low) is always active when the device is in Programming state. A host should not send CMD24/CMD25 while the device is in the Programming state and busy is active. However to ensure compatibility with previous MultiMediaCard specification, the device treats CMD24 and CMD25 as legal or illegal commands when in Programming state (while busy is active).

5.3.6 Data Read

The DAT0-DAT7 input/outputs are High when no data is transmitted.

Data Reads allow data to be transferred from the device to the host. All Data Read commands can be aborted at any time by the STOP_TRANSMISSION command (CMD12), which will terminate the data transfer and return the device to the Transfer state.

The DAT bus line is High when no data is transmitted. A transmitted data block consists of a start bit (Low), followed by a continuous data stream. The data stream contains the net payload data (and error correction bits if a non-embedded Error Correction is used). The data stream ends with an end bit (High) (see *Figure 19*, *Figure 20*, and *Figure 21*). The data transmission is synchronous to the clock signal.

The payload for block oriented data transfer is preserved by a CRC (Cyclic Redundancy Check) check sum.

5.3.7 Single Block/Multiple Block Read

The command CMD17 starts a single Block Read at the address specified in the command. After completion of the Single Block Read command, the device returns to the Transfer state.

The command CMD18 starts a Multiple Block Read where several consecutive blocks of data are read. The starting address is specified in the command. The blocks will be continuously transferred until a STOP-TRANSMISSION command (CMD12) is issued. Note that the host CMD12 command has an execution delay due to the serial command transmission. The data transfer stops after the end bit of the CMD12 command.

The start address for a read operation can be any byte address in the valid address space of the memory card.

During Single or Multiple Block Read operations, the basic unit of data transferred is a block whose maximum size is defined in the CSD Register. If READ_BL_PARTIAL is set, smaller blocks whose starting and ending address are contained within one physical block may also be transmitted. A 16 bit CRC (Cyclic Redundancy Check) is appended to the end of each block ensuring data transfer integrity.

Multiple Block Read operations can be of two types:

- **Open-ended Multiple Block Read operations**
  the number of blocks is not defined and the device keeps transferring data blocks until a STOP_TRANSMISSION command is issued.
- **Multiple Block Read with pre-defined block count**
  The number of blocks to be transferred is pre-determined so the operation stops after the pre-set number of blocks has been transmitted. When the block count is pre-defined, the STOP_TRANSMISSION command is not required unless an error occurs. To issue the Multiple Block Read operation with pre-defined block count, the READ_MULTIPLE_BLOCK command must be preceded by the SET_BLOCK_COUNT (CMD23) command, failing which the initiated Multiple Block Read operation will be open-ended.
  If all the arguments of the CMD23 command are set to 0, the command is accepted. However, a subsequent read will follow the open-ended READ_MULTIPLE_BLOCK operation protocol (STOP_TRANSMISSION command is required)
  If the host sends a STOP-TRANSMISSION command after the last block of a multiple block operation with a pre-defined number of blocks is transmitted, it is regarded as an illegal command, since the device is no longer in sending data state (Data).

If either one of the following errors is detected when the CMD17/CMD18 command is received, the device rejects the CMD17/CMD18 command, remains in Transfer state and sets the corresponding error bit:

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

If the device detects an error (e.g. address out of range, address misalignment, internal error, etc.) during a Multiple Block Read operation, it stops data transmission and remains in the sending data state (Data). The host must then abort the operation by sending the STOP-TRANSMISSION command. The read error is reported in the response to the STOP-TRANSMISSION command.

When the host uses partial blocks, if block misalignment is not allowed, the device returns a block misalignment condition (ADDRESS_MISALIGN bit set to '1') if the total length of the partial blocks is not block aligned, and returns to Transfer state.

If the host sets the argument of the SET_BLOCK_COUNT command (CMD23) to 0, the command is accepted; however, a subsequent read will follow the open-ended Multiple Block Read protocol (STOP_TRANSMISSION command - CMD12 - is required).

5.3.8 Data Write

Data Writes allow data to be transferred from the host to the device. All data write commands can be aborted any time by the CMD12 command. As soon as the data transfer has completed, the device exits the Data Write state and switches either to the Programming state (transfer successful) or Transfer state (transfer failed).

The Data Write 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 device performs a CRC check for each data block received prior to a write operation. The polynomial is the same as the one used for a read operation.

Read and Parameter Set commands are not allowed while the device is programming.

Moving another MultiMediaCard from Standby to Transfer state (using CMD7) does not terminate a programming operation. The device switches to the Disconnect state and releases the DAT line. The device can be reselected using CMD7. In this case it moves to the Programming state and reactivates the busy indication.

The device provides buffering for Block Write. This means that the next block can be sent to the device while the previous is being programmed. If all the write buffers are full, and the device is in the Programming state, the DAT line will be kept Low.

There is no buffering option for Write CSD, Write CID and Erase. This means that while the device is busy servicing any one of these commands, no other data transfer commands will be accepted. The DAT line will be kept Low as long as the device is busy and in the Programming state.

Note: Care should be taken by the host not to reset a device (using CMD0 or CMD15) during any pending or active programming operation. This will terminate the operation and may destroy the data stored on the device.

5.3.9 Single Block/Multiple Block Write

Single or Multiple Block Write (CMD24-27) allows one or more blocks of data to be transferred from the host to the device with a CRC bit appended to the end of each block by the host. A device supporting Block Write must always be able to accept a block of data defined by WRITE_BL_LEN. If the CRC fails, the device indicates the failure on the DAT line; the transferred data will be discarded and all further transmitted blocks (Multiple Block Write mode) will be ignored.

Multiple Block Write operations are initiated by issuing the WRITE_MULTIPLE_BLOCK command (CMD25). There are two types of Multiple Block Write operations:

- **Open-ended Multiple Block Write**
  The number of blocks is not defined and the host terminates device programming by sending a STOP-TRANSMISSION command.
- **Multiple Block Write with pre-defined block counts**
  The number of blocks to be programmed is pre-determined so the host does not need to send a STOP-TRANSMISSION command to stop the operation. To issue the Multiple Block Write operation with a pre-defined block count, the WRITE_MULTIPLE_BLOCK command must be preceded by the SET_BLOCK_COUNT (CMD23) command, failing which the initiated Multiple Block Write operation will be open-ended.
  If all the arguments of the CMD23 command are set to 0, the command is accepted. However, a subsequent write will follow the open-ended WRITE_MULTIPLE_BLOCK operation protocol (STOP_TRANSMISSION command is required).
  If a Multiple Block Write with pre-defined block count is aborted by a STOP-TRANSMISSION command, the data in the remaining blocks are invalid.
  If the host sends a STOP-TRANSMISSION command after the last block of a Multiple Block Write operation with a pre-defined number of blocks is programmed, it is regarded as an illegal command, since the device is no longer in Receiving data state.

If either one of the following errors is detected when the CMD24-27 command is received, the device rejects the command, remains in Transfer state and sets the corresponding error bit:

- The address provides by the host as an argument to either CMD24-27 is out of range. ADDRESS_OUT_OF_RANGE is set.
- The currently defined block length is illegal for a write operation. BLOCK_LEN_ERROR is set.
- The address/block-length combination positions the first data block is misaligned to the device physical blocks. ADDRESS_MISALIGN is set.

If the device detects an error (e.g. write protect violation, address out of range, address misalignment, internal error, etc.) during a Multiple Block Write operation, it stops data transmission and remains in the Receiving data state. The host must then abort the operation by sending the STOP-TRANSMISSION command. The write error is reported in the response to the STOP-TRANSMISSION command.

When the host uses partial blocks and block misalignment is not allowed (WRITE_BLK_MIS-ALIGN parameter not set in CSD Register), the total length of the partial blocks must be block aligned otherwise the device detects the misalignment, return an error data response, ignore subsequent incoming data blocks, and return to Transfer state.

The block length does not need to be set prior to programming the CID and CSD registers. The Data transferred to the CID and CSD registers is also CRC protected. If a part of the

CSD or CID register is stored in ROM it will not be overwritten and the device does not check the ROM data with the content of the received buffer.

Some devices may require a long time (1 s max) to write a block of data. After receiving a block of data and completing the CRC check, the device begins writing and hold DAT Low if its write buffer is full and unable to accept new data from a new Block Write command. The host may poll the status of the device with a SEND_STATUS command at any time, and the device responds with its status. The status bit READY_FOR_DATA indicates whether the device can accept new data or whether the write process is still in progress. The host may deselect the device by issuing CMD7 (to select a different device) which will place the device in the Disconnect state and release the DAT line without interrupting the write operation. When re-selecting the device, it will reactivate the busy indication by pulling DAT to Low if programming is still in progress and the write buffer is unavailable. If a block write operation is stopped and the block length and CRC of the last block are valid, the data will be programmed.

5.3.10 Group Erase

The device supports Group Erase. The size of the Erase Group is defined in the CSD register. To select an Erase Group, a first command with the starting address is followed by a second command with the final address. After a range is selected, the erase operation is performed by issuing an ERASE command (CMD38).

The address field in the CMD35/CMD36/CMD38 commands is the group address in units of bytes. The device will ignore all LSBs below the group size.

The host must adhere to the following command sequence:

- ERASE_GROUP_START (CMD35)
- ERASE_GROUP_END (CMD36)
- ERASE (CMD38)

If an CMD35/CMD36/CMD38 command is received out of sequence the device sets the ERASE_SEQ_ERROR error bit in the Status Register and reset the whole sequence.

If the host provides an out of range address as an argument of the CMD35 or CMD36 command, the device rejects the command, returns the ADDRESS_OUT_OF_RANGE error set, and resets the whole erase sequence.

If an non-erase command is received (different form CMD35, CMD36, CMD38 or CMD13), the device returns the ERASE_RESET error, resets the erase sequence and executes the last command.

Commands not addressed to the selected card do not abort the erase sequence.

If the erase group includes write protected blocks, only unprotected blocks are erased. In this case, the WP_ERASE_SKIP status bit of the Status Register is set.

The device indicates that an erase operation is in progress by holding DAT to Low.

5.4 Write protection

The device supports two levels of write protection commands to protect data against erase or write operations:

- The entire memory array may be write protected by setting the permanent or temporary write protect bits of the CSD register.
- Specific segments of the memory may be write protected. The segment size is defined in units of WP_GRP_SIZE erase groups as specified in the CSD register. The SET_WRITE_PROT (CMD28) command sets the write protection of the addressed write-protect group, and the CLR_WRITE_PROT (CMD29) command clears the write protection of the addressed write-protect group.

The SEND_WRITE_PROT (CMD30) command is similar to a single block read command. The device sends 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 device will ignore all LSBs below the group size.

If the host provides an out of range address as an argument to CMD28, CMD29 or CMD30, the device rejects the command, returns the ADDRESS_OUT_OF_RANGE error and remains in the Transfer state.

5.5 Device locking/unlocking (password protection)

The password protection feature enables the host to lock the device by providing a password, which later will be used for unlocking the device. 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-up cycle will not erase them.

The host is allowed to reset, initialize, select, query for status, etc., but not to access data on the device. If the password has been previously set (PWD_LEN value is not ‘0’), the device is automatically locked after power-up.

A locked device answers and executes all commands belonging to the basic class (class 0) and to the Lock Card class (class 7). The host can consequently reset, initialize, select, query for status, etc., but cannot access the data stored in the device.

As for the CSD and CID register write commands, the Lock/Unlock commands are available in Transfer state only. This means that they do not require any address argument and that the device has to be selected before issuing any of these commands.

The device Lock/Unlock commands have the same structure and comply with the same command/response transaction as single block write commands. The transferred data block

includes all the required information of the command (password setting mode, password, Lock/Unlock etc.) (see *Table 15*):

- **ERASE** bit: this bit must be set to '1' (all other bits shall be '0') to perform a Forced Erase operation, and only the Cmd byte is sent.
- **LOCK/UNLOCK** bit: The device is locked and unlocked by setting the Lock/Unlock bit to '1' or '0', respectively. Setting this bit together with the SET_PWD bit is allowed, while setting it together with CLR_PWD bit is forbidden.
- **CLR_PWD** bit: this bit must be set to '1' to clear the password.
- **SET_PWD** bit: this bit must be set to '1' to set a new password.
- **PWD_LEN**: this byte contains the password length expressed in bytes. Valid password length ranges from 1 to 16 bytes. PWD_LEN indicates if a password is currently set. If its value is equal to zero, no password is set, if it is different from zero, the device is locked after power-up.
- **Password (PWD):** these bytes contain the new or current password (depending on the command). The data block size is defined by the host before sending the device Lock/Unlock command. Different password sizes are allowed.

The Lock/Unlock command sequences are described in the following paragraphs.

Table 15. Lock/Unlock data block

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

5.5.1 Setting the password

The following steps are required to set the password:

1. Select the device if it has not been not previously selected. This is performed by issuing a CMD7 command.
2. Configure the block length by issuing a CMD16 command. The block length is given by the 8-bit Lock/Unlock mode, the 8-bit password size (in bytes), and the number of bytes of the new password. In case of password replacement, the block size must take into account the fact that both the old and the new passwords are sent with the command.
3. Send a Lock/Unlock command on the data line, along with the data block of the appropriate size and 16-bit CRC. The data block must contain the mode (SET_PWD), the length (PWD_LEN) and the password itself. If a password replacement is performed, the length value (PWD_LEN) must take into account both old and the new password length, and the PWD field must be composed of the current password followed by the new password.

If a password replacement is attempted with PWD_LEN set to the length of the current password only, or the current password is not correct (different size and content), then the LOCK_UNLOCK_FAILED error bit will be set in the Status Register and the current password is not changed.

In case that PWD matches the sent old password then the given new password and its size will be saved in the PWD and PWD_LEN fields, respectively.

Note: The device can be locked immediately after setting the password by programming the Lock/Unlock bit to '1' while setting the password or by sending an additional Lock command.

5.5.2 Resetting the password

The following steps are required to reset the password:

1. Select the device if it has not been not previously selected. This is performed by issuing a CMD7 command.
2. Configure the block length by issuing a CMD16 command. The block length is given by the 8-bit Lock/Unlock mode, the 8-bit password size (in bytes), and the number of bytes of the current password.
3. Send the Lock/Unlock command on the data line, along with the data block of the appropriate size and 16- bit CRC. The data block must contain the mode (CLR_PWD), the length (PWD_LEN) and the password itself. (PWD). If the PWD and PWD_LEN content match the 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 is set in the Status Register.

5.5.3 Locking the device

The following steps are required to lock the device:

1. Select the device if it has not been not previously selected. This is performed by issuing a CMD7 command.
2. Configure the block length by issuing a CMD16 command. The block length is given by the 8-bit Lock/Unlock mode, the 8-bit password size (in bytes), and the number of bytes of the current password.
3. Send the Lock/Unlock command on the data line, along with the data block of the appropriate size and 16- bit CRC. The data block must indicate the Lock mode, the length (PWD_LEN) and the password (PWD) itself. If the PWD content is identical to the device password, the device is locked and the locked status bit set in the Status Register. If the password is not correct, the LOCK_UNLOCK_FAILED error bit is set in the Status Register.

If the password was previously set (PWD_LEN is not '0'), the device is automatically locked after power-up.

An attempt to lock a locked device or to lock a device that does with no password defined will fail and the LOCK_UNLOCK_FAILED error bit will be set in the Status Register.

Note: It is possible to set the password and to lock the device in the same sequence. In this case the host must go through all the steps required to set the password (see Section 5.5.1: Setting the password) and set the Lock bit while the new password command is issued.

5.5.4 Unlocking the device

The following steps are required to lock the device:

1. Configure the block length by issuing a CMD16 command. The block length is given by the 8-bit Lock/Unlock mode, the 8-bit password size (in bytes), and the number of bytes of the current password.
2. Send the Lock/Unlock command on the data line, along with the data block of the appropriate size and 16- bit CRC. The data block indicates the Unlock mode, the length (PWD_LEN) and the password (PWD) itself. If the PWD content is identical to the device password, the device is unlocked and the locked status bit is cleared in the Status Register. If the password is incorrect, the LOCK_UNLOCK_FAILED error bit is set in the Status Register.

An attempt to unlock an unlocked device will fail and LOCK_UNLOCK_FAILED error bit will be set in the Status Register.

Note: The device is unlocked for the current power session only. As long as the PWD is not cleared, the device will be automatically locked after the next power-up. The only way to unlock the device is to clear the password.

5.5.5 Performing a Forced Erase

If the user forgets the password, it is possible to erase all the device data along with the PWD content. This operation is called a Forced Erase.

The following steps are required to perform a Forced Erase operation on the device:

1. Select the device if it has not been not previously selected. This is performed by issuing a CMD7 command.
2. Configure the block length to 1 byte (8-bit Lock/unlock command) by issuing a CMD16 command.
3. Send the Lock/Unlock command on the data line, along with the data block of the appropriate size and 16- bit CRC. The data block indicates the ERASE mode (the ERASE bit must be the only bit set to '1'). **The whole memory content is then erased** including the password (PWD) and PWD_LEN register, and the locked device is unlocked. In addition, if the device was temporary write protected, it is unprotected (write enabled), and the CSD temporary-write-protect bit and all Write-Protect-Groups are cleared. If other bits than the ERASE bit are set to '1', the LOCK_UNLOCK_FAILED error bit is set and the Forced Erase operation fails.

An attempt to force erase on an unlocked device will fail and LOCK_UNLOCK_FAILED error bit will be set in the Status Register.

Issuing a Forced Erase command on a permanently-write-protected device will fail, the device will remain locked, and the LOCK_UNLOCK_FAILED error bit will be set.

The Forced Erase timeout is specified in *Table 16*.

5.5.6 Application specific commands

The NAND08GAH0A and NAND16GAH0D devices support two application specific commands:

- APP_CMD (CMD55)
- GEN_CMD (CMD56)

APP_CMD command (CMD55)

Receiving a CMD55 command from the host causes the device to interpret the next command as an application specific command, ACMD. The ACMD command has the same structure as a regular MultiMediaCard standard commands and may have the same CMD number. The device recognizes it as an application specific command because it follows the APP_CMD command.

If the application specific version of the command that follows the APP_CMD command is supported, the non standard version is used. If it is not supported, the standard version is used.

Let us take the example of a device accepting ACMD13 but not ACMD7. When receiving the APP_CMD command immediately followed by command 13, the device will interpret it as the non standard command ACMD13. Whereas it will interpret command 7 as the standard command CMD7.

To use one of the application specific ACMD commands, the host must follow the steps described below:

- Send the APP_CMD command. The device will respond with APP_CMD bit (new status bit) of the response set to '1' to signal to the host that ACMD is now expected.
- Send the required ACMD command. The device will respond with APP_CMD bit set, indicating that the accepted command was interpreted as an ACMD command.
  - If a non-ACMD command is sent, then the device will handle it as a normal command, and the APP_CMD bit in the Card status will remain set to '0'.
  - If a non valid command is sent (neither ACMD nor CMD) then the device will handle it as a standard MultiMediaCard illegal command.

GEN_CMD command (CMD56)

Bus operation during a GEN_CMD command is identical as Single Block Read or Write commands (CMD24 or CMD17). The only difference is that the argument indicates 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 must be selected (Transfer state) before the host sends a CMD56 command. The data block size is specified in the BLOCK_LEN defined with CMD16. The response to CMD56 will be of R1 type.

5.6 Clock control

The device bus clock signal can be used by the host to set the device to energy saving mode or to control the data flow on the bus. The host is allowed to lower the clock frequency or shut it down.

There are a few restrictions the host must follow:

- The bus frequency can be changed at any time, under the restrictions of maximum data transfer frequency and the identification frequency
- The clock must be running for the device to output data or response tokens
- After the last bus transaction, the host is required to provide 8 clock cycles for the device to complete before shutting down the clock.

The host is allowed to shut down the clock of a busy device. The device will complete the programming operation regardless of the host clock. However, the host must provide a clock edge for the device to turn off its busy flag. Without a clock edge the MultiMediaCard (unless previously disconnected by a deselect command CMD7) will force the DAT line Low, permanently.

5.7 Error conditions

5.7.1 CRC and illegal commands

All commands are protected by CRC (Cyclic Redundancy Check) bits. If the addressed device CRC check fails, the device does not respond and the command is not executed. The device does not change its state, and the COM_CRC_ERROR bit is set in the Status Register.

Similarly, if an illegal command has been received, the device will not respond or change its state and will set the ILLEGAL_COMMAND error bit in the Status Register. Error conditions are not shown in the state diagrams (*Figure 10* and *Figure 12*). Refer to *Table 27* for a complete state transition description.

There are different kinds of illegal commands:

- Commands which belong to classes not supported by the device (e.g. write commands in read only devices)
- Commands not allowed in the current state (e.g. CMD2 in Transfer state)
- Undefined commands (e.g. CMD44).

5.7.2 Read, Write and Erase timeout conditions

The times after which a timeout condition for read/write/erase operations occurs are 10 times longer than the typical access/program times for these operations. A device will complete the command within this time, or give up and return an error message. If the host does not get a response within the defined timeout it should assume the device is not going to respond and reset the device.

Table 16 gives the formulae required to calculate typical access and program times.

Table 16. Formulae to calculate typical access and program times(1)

Time	Unit	Formula	Description
Read Access time	clock cycles	(TAAC + NSAC)	These parameters define the typical delay between the end bit of the Read command and the start bit of the Data Block.
Block Write time	clock cycles	(Read Access time * R2W_FACTOR)	This applies to all Write/Erase commands
Erase time	clock cycles	Number of Erase groups * Block Write time	This gives an approximate value
Forced Erase time	min	3	Duration of the Forced Erase operation using CMD42 command

1. See *Section 8.3: Card specific data register (CSD)* for the definition of the parameters used to calculate the maximum clock frequency.

6 Commands

There are four kinds of commands defined on the bus:

- Broadcast commands (bc)—sent on CMD, no response
- Broadcast commands with response (bcr)— sent on CMD, response (all devices simultaneously) on CMD
- Addressed (point-to-point) commands (ac)—sent on CMD, response on CMD
- Addressed (point-to-point) data transfer commands (adtc)—sent on CMD, response on CMD, data transfer on DAT.

All commands are 48 bits long, and are protected by a CRC. The command transmission always starts with the MSB (see *Table 17: MultiMediaCard command format*).

6.1 Command classes

The command set of the device is divided into several classes (See *Table 18* and *Table 19*). Each class supports a set of MultiMediaCard functions.

Class 0 is mandatory and supported by all MultiMediaCards. The other classes are optional and can be interpreted as a tool box. By using different classes, several configurations can be chosen (e.g. a block writable device). The supported Card Command Classes (CCC) are coded as a parameter in the Card Specific Data (CSD) register of each device, providing the host with information on how to access the device.

Table 20, *Table 21*, *Table 22*, *Table 23* and *Table 24* define in detail the device bus commands. *Table 27* defines the device state transitions depending on the command received.

Table 17. MultiMediaCard command format

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	Host	Command	Argument	CRC7	End bit

Table 18. Device command classes (CCCs) - supported commands 0 to 27

Device command class (CCC)	Class description	Supported commands, CMD																							
		0	1	2	3	4	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	23	24	25	26	27
Class 0	Basic	+	+	+	+	+		+		+	+		+	+	+	+				+						
Class 2	Block Read																+	+	+			+				
Class 4	Block Write																+					+	+	+	+	+
Class 5	Erase																									
Class 6	Write Protection																									
Class 7	Lock																+									
Class 9	I/O mode																									

Table 19. Card command classes (CCCs) - supported commands 28 to 56

Card command class (CCC)	Class description	Supported commands, CMD										
		28	29	30	35	36	38	39	40	42	55	56
Class 0	Basic											
Class 2	Block Read											
Class 4	Block Write											
Class 5	Erase				+	+	+					
Class 6	Write Protection	+	+	+								
Class 7	Lock									+		
Class 9	I/O mode							+	+			

6.2 Detailed command description

The following tables provide a detailed description of MultiMediaCard commands. The responses R1 to R4 are defined in *Section 7: Responses*.The registers CID, CSD, EXT_CSD and DSR are described in *Section 8: Device registers*.

Table 20. Basic commands for read-only devices (class 0)

Cmd Index	Type	Argument	Response	Abbreviation	Command description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets all devices to Idle state. All devices in Idle state after power-up
CMD1	bcr	[31:0] OCR without busy	R3	SEND_OP_COND	Asks all devices in Idle state to send their OCR content in the response on CMD line
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks all devices in Ready state to send their CID(1) numbers on CMD line
CMD3	ac	[31:16] RCA [15:0] stuff bits	R1	SET_RELATIVE_ADDR	Assigns relative address to the device in identification state
CMD4	bc	[31:16] DSR [15:0] stuff bits	-	SET_DSR	Programs the DSR of all devices in Standby state
CMD5	Reserved				
CMD6	ac	[31:26] Set to '0' [25:24] Access [23:16] Index [15:8] Value [7:3] Set to '0' [2:0] Cmd Set	R1b	SWITCH	Switches the device operating mode or modifies the EXT_CSD register (see *Section 8.4: Extended CSD register*)
CMD7	ac	[31:16] RCA [15:0] stuff bits	R1/R1b(2)	SELECT/DESELECT_ CARD	Command toggles a device between the Standby and Transfer states or between the Programming and Disconnect states. In both cases the device is selected by its own relative address and deselected by any other address; address 0 deselects all
CMD8	adtc	[31:0] stuff bits	R1	SEND_EXT_CSD	Ask the address device to send back its EXT_CSD register as a data block
CMD9	ac	[31:16] RCA [15:0] stuff bits	R2	SEND_CSD	Asks the addressed device to send its card specific data, CSD, on CMD line.
CMD10	ac	[31:16] RCA [15:0] stuff bits	R2	SEND_CID	Asks the addressed device to send its card identification data, CID, on CMD line.
CMD11	Reserved				

Table 20. Basic commands for read-only devices (class 0) (continued)

Cmd Index	Type	Argument	Response	Abbreviation	Command description
CMD12	ac	[31:0] stuff bits	R1/R1b(3)	STOP_TRANSMISSION	Forces the device to stop transmission.
CMD13	ac	[31:16] RCA [15:0] stuff bits	R1	SEND_STAT	Asks the addressed device to send its Status Register.
CMD14	adtc	[31:0] stuff bits	R1	BUSTEST_R	Reads the reversed bus test data pattern from a device.
CMD15	ac	[31:16] RCA [15:0] stuff bits	-	GO_INACTIVE_STATE	Sets the device to Inactive state to prevent communication breakdowns in the stack of devices.
CMD19	adtc	[31:0] stuff bits	R1	BUSTEST_W	Sends the bus test data pattern to the device.

1. The addressing capability for 8-bit address resolution is 2^{32} = 4 Gbytes.
2. The response is R1 when the selecting from Standby to Transfer state, and R1b when selecting from Disconnected state to Programming state.
3. The response is R1 and R1b, for read and write operations, respectively.

Table 21. Block oriented Read commands (class 2)

Cmd index	Type	Argument	Response	Abbreviation	Command description
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	Selects block length (in bytes) for all following block commands (Read and Write)(1)
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command(2)
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously send blocks of data until interrupted by a Stop command.

1. The default block length is as specified in *Section 8.3: Card specific data register (CSD)*.
2. The data transferred must not cross a physical block boundary unless RD_BLK_MISALIGN is set in the CSD.

Table 22. Block oriented Write commands (class 4)

Cmd index	Type	Argument	Response	Abbreviation	Command description
CMD23	ac	[31:16] set to 0 [15:0] number of blocks	R1	SET_BLOCK_COUNT	Defines the number of blocks which are going to be transferred in the next Multiple Block Read or Write command
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until interrupted by a Stop command
CMD26	adtc	[31:0] stuff bits	R1	PROGRAM_CID	Programs the CID register - done once per device and is normally reserved for the manufacturer. The device contains hardware to prevent further programming
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	Programs the programmable bits of the CSD

Table 23. Block oriented Write commands (class 6)

Cmd index	Type	Argument	Response	Abbreviation	Command description
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	If the device 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 device 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 device provides write protection features, this command asks the device 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 are set to zero.

Table 24. Erase commands (class 5)

Cmd index	Type	Argument	Response	Abbreviation	Command description
CMD35	ac	[31:0] data address	R1	ERASE_GROUP_START	Sets the address of the first Erase Group within a range to be selected for erase.
CMD36	ac	[31:0] data address	R1	ERASE_GROUP_END	Sets the address of the last Erase Group within a continuous range to be selected for erase.
CMD38	ac	[31:0] stuff bits	R1b	ERASE	Erases all previously selected groups

Table 25. I/O mode commands (class 9)

Cmd index	Type	Argument	Response	Abbreviation	Command description
CMD39	ac	[31:16] RCA [15:15] register write flag [14:8] register address [7:0] register data	R4	FAST_IO	Used to write and read 8 bit (register) data fields. The command addresses a device and a register and provides the data for writing if the write flag is set. This command accesses application dependent registers which are not defined in the MultiMediaCard standard.
CMD40-41	Reserved				

Table 26. Lock (class 7)

Cmd index	Type	Argument	Response	Abbreviation	Command description
CMD42	adtc	[31:0] stuff bits	R1	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the device. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43... CMD54	Reserved				
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the device 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 device or to get a data block from the device for general purpose / application specific commands. The size of the data block shall be set by issuing a SET_BLOCK_LEN command

1. RD/WR is set to ‘1’ if the host receives a data block from the device, and to ‘0’ if the host sends a data block to the device.

6.3 Device state transition

Table 27: Device state transition gives the device state transitions according to the command received from the host.

Table 27. Device state transition

Command	Current state											
	idle	ready	ident	stby	tran	data	btst	rcv	prg	dis	ina	irq
	changes to											
Class independent												
CRC error	–	–	–	–	–	–	–	–	–	–	–	stby
Command not supported	–	–	–	–	–	–	–	–	–	–	–	stby
class 0												
CMD0	idle	idle	idle	idle	idle	idle	idle	idle	idle	idle	–	stby
CMD1, V_{CC} range compatible	ready	–	–	–	–	–	–	–	–	–	–	stby
CMD1, device busy	idle	–	–	–	–	–	–	–	–	–	–	stby
CMD1, V_{CC} range not compatible	ina	–	–	–	–	–	–	–	–	–	–	stby
CMD2, device wins bus	–	ident	–	–	–	–	–	–	–	–	–	stby
CMD2, device loses bus	–	ready	–	–	–	–	–	–	–	–	–	stby
CMD3	–	–	stby	–	–	–	–	–	–	–	–	stby
CMD4	–	–	–	stby	–	–	–	–	–	–	–	stby
CMD6	–	–	–	–	prg	–	–	–	–	–	–	stby
CMD7, device addressed	–	–	–	tran	–	–	–	–	–	prg	–	stby
CMD7, device not addressed	–	–	–	–	stby	stby	–	–	dis	–	–	stby
CMD8	–	–	–	–	data	–	–	–	–	–	–	stby
CMD9	–	–	–	stby	–	–	–	–	–	–	–	stby
CMD10	–	–	–	stby	–	–	–	–	–	–	–	stby
CMD12	–	–	–	–	–	tran	–	prg	–	–	–	stby
CMD13	–	–	–	stby	tran	data	btst	rcv	prg	dis	–	stby
CMD14	–	–	–	–	–	–	tran	–	–	–	–	stby
CMD15	–	–	–	ina	ina	ina	ina	ina	ina	ina	–	stby
CMD19	–	–	–	–	btst	–	–	–	–	–	–	stby

Table 27. Device state transition (continued)

Command	Current state											
	idle	ready	ident	stby	tran	data	btst	rcv	prg	dis	ina	irq
	changes to											
class 2												
CMD16	–	–	–	–	tran	–	–	–	–	–	–	stby
CMD17	–	–	–	–	data	–	–	–	–	–	–	stby
CMD18	–	–	–	–	data	–	–	–	–	–	–	stby
CMD23	–	–	–	–	tran	–	–	–	–	–	–	stby
class 4												
CMD16	see class 2											
CMD23	see class 2											
CMD24	–	–	–	–	rcv	–	–	–	rcv	–	–	stby
CMD25	–	–	–	–	rcv	–	–	–	rcv	–	–	stby
CMD26	–	–	–	–	rcv	–	–	–	–	–	–	stby
CMD27	–	–	–	–	rcv	–	–	–	–	–	–	stby
class 6												
CMD28	–	–	–	–	prg	–	–	–	–	–	–	stby
CMD29	–	–	–	–	prg	–	–	–	–	–	–	stby
CMD30	–	–	–	–	data	–	–	–	–	–	–	stby
class 5												
CMD35	–	–	–	–	tran	–	–	–	–	–	–	stby
CMD36	–	–	–	–	tran	–	–	–	–	–	–	stby
CMD38	–	–	–	–	prg	–	–	–	–	–	–	stby
class 7												
CMD16	see class 2											
CMD42	–	–	–	–	rcv	–	–	–	–	–	–	stby
class 8												
CMD55	–	–	–	stby	tran	rcv	btst	rcv	prg	dis	–	irq
CMD56, RD/WR = 0	–	–	–	–	rcv	–	–	–	–	–	–	stby
CMD56, RD/WR = 1	–	–	–	–	data	–	–	–	–	–	–	stby
class 9												
CMD39	–	–	–	stby	–	–		–	–	–	–	stby
CMD60-CMD63	Reserved for manufacturer											

7 Responses

All responses are sent via the CMD command line. The response transmission always starts with the left bit of the bit string corresponding to the response code word. 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 (device = '0'). A value denoted by 'x' in the tables below indicates a variable entry. All responses except for the R3 type (see *Section 7.4*) are protected by a CRC. Every command code word is terminated by the end bit (always '1').

There are five response types: R1, R1b, R2, R3, R4.

7.1 R1 response (normal response command)

R1 response code length is 48 bits. Bits 45 to 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 device is coded in 32 bits. See *Table 28* for a full description of R1 responses.

Table 28. R1 response

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	Device status	CRC7	End bit

7.2 R1b response

R1b response is identical to R1, except that it has an additional busy flag sent via the DAT line as defined in the MultiMediaCard specification.

7.3 R2 response (CID, CSD register)

R2 response code length is 136 bits. The content of the CID register is sent as a response to CMD2, CMD9 and CMD10 commands. Only bits 127 to 1 of the CID and CSD registers are transferred. The reserved bit (bit 0) of these registers is replaced by the end bit of the response. See *Table 29* for a full description of R2 responses.

Table 29. R2 response

Bit position	135	134	[133:128]	[127:1]	0
Width (bits)	1	1	6	127	1
Value	'0'	'0'	'111111'	x	'1'
Description	Start bit	Transmission bit	Reserved	CRC7	End bit

7.4 R3 response (OCR register)

R3 response code length is 48 bits. The contents of the OCR register is sent as a response to CMD1 commands.

See *Table 30* for a full description of R3 responses.

Table 30. R3 response

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

7.5 R4 response (Fast I/O)

R4 response code length is 48 bits. The argument field contains the RCA of the addressed device, the register address to be read out or written to, and its contents.

See *Table 31* for a full description of R4 responses.

Table 31. R4 response

Bit position	47	46	[45:40]	[39:8] Argument field			[7:1]	0
Width (bits)	1	1	6	16	8	8	7	1
Value	‘0’	‘0’	‘100111’	x	x	x	x	‘1’
Description	Start bit	Transmission bit	CMD39	RCA [31:16]	Register addr. [15:8]	Read register contents	CRC7	End bit

8 Device registers

There are five different registers within the device interface:

- Operation conditions register (OCR)
- Card identification register (CID)
- Card specific data register (CSD)
- Relative card address register (RCA)
- DSR (driver stage register)
- Extended card specific data register (EXT_CSD)

These registers are used for the serial data communication. The device does not implement the DSR register.

The MultiMediaCard has a status register to provide information about the device current state and completion codes for the last host command.

8.1 Operation conditions register (OCR)

The 32-bit operation conditions register stores the V_{CCQ}, the input/output voltage of the Flash memory component. The device is capable of communicating (identification procedure and data transfer) with any MultiMediaCard host using any operating voltage within 1.65 V and 1.95 V (low-voltage range) or 2.7 V and 3.6 V (high-voltage range) depending on the voltage range supported by the host. The 31 least significant bits are constant. Bit 32 is the busy flag as defined in the MultiMediaCard specification document.

If the host tries to change the OCR values during an initialization procedure the changes in the OCR content will be ignored.

The level coding of the OCR register is as follows:

- Restricted voltage windows = Low
- Device busy = Low

Table 32. OCR register definition

OCR bit	Description	MultiMediaCard
6 to 0	Reserved	000 0000b
7	Low V_{CCQ}	1b
14 to 8	2.0 - 2.6	000 0000b
23 to 15	2.7 - 3.6 (High V_{CCQ} range)(1)	1 1111 1111b
30 to 24	Reserved	000 0000b
31	Power-up status bit (busy) (2)	

1. The voltage for internal Flash memories (V_{CCi}) should be in the 2.7 V to 3.6 V range.
2. This bit is set to Low if the device has not finished the power-up routine.

8.2 Card identification (CID) register

The CID register is 16 bytes long and contains a unique card identification number used during the card identification procedure. It is a 128 bit wide register with the content as defined in *Table 33*. It is programmed during device manufacturing and can not be changed by MultiMediaCard hosts.

Table 33. Card identification (CID) register

Name	Field	Width	CID - slice	CID - value
Manufacture ID	MID	8	[127:120]	0x33
OEM/application ID	OID	16	[119:104]	0x5354
Product name	PNM	48	[103:56]	eMMC01
Product revision	PRV	8	[55:48]	1.0
Product serial number	PSN	32	[47:16]	TBD
Manufacturing date	MDT	8	[15:8]	Manufacturing date
CRC7 checksum	CRC	7	[7:1]	TBD
Not used, always ‘1’	-	1	[0:0]	-

8.3 Card specific data register (CSD)

All the configuration information required to access the device data is stored in the CSD register. The MSB bytes of the register contain the manufacturer data and the two least significant bytes contains the host controlled data (the device Copy, Write Protection and the user ECC register).

The host can read the CSD register and alter the host controlled data bytes using the SEND_CSD and PROGRAM_CSD commands.

In *Table 34*, the cell type column defines the CSD field as Read only (R), One Time Programmable (R/W) or erasable (R/W/E). The programmable part of the register (entries marked by W or E) can be changed by command CMD27.

The Copy bit in the CSD can be used to mark the device as an original or a copy. Once set it cannot be cleared. The device can be purchased with the copy bit set (copy) or cleared, indicating the device is a master.

The One Time Programmable (OTP) characteristic of the Copy bit is implemented in the MultiMediaCard controller firmware and not with a physical OTP cell.

Table 35 to *Table 47* describe the CSD fields and the relevant data types. If not otherwise defined, all bit strings are interpreted as binary coded numbers starting with the left bit first.

Table 34. Card specific data register

Name	Field	Width [bits]	Cell type	CSD-slice	CSD-value	
CSD structure	CSD_STRUCTURE	2	R	[127:126]	0x3 (see *Table 49: Extended CSD*)	
MultiMediaCard protocol version	SPEC_VERS	4	R	[125:122]	0x4 (version 4.0, 4.1, 4.2)	
Reserved		2	R	[121:120]	TBD(1)	
Data Read Access-time-1	TAAC	8	R	[119:112]	0x5E (TAAC=5000 µs, NSAC=0 cycles)	
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]	0x2A (20 Mbit/s)	
Command classes	CCC	12	R	[95:84]	0x1F5 (classes 0, 2, 4, 5, 6, 7, 8)	
Max. Read Data Block Length	READ_BL_LEN	4	R	[83:80]	NAND08GAH0A	512 bytes
					NAND16GAH0D	1024 bytes
Partial Blocks for Read allowed	READ_BL_PARTIAL	1	R	[79:79]	1 (Yes)	
Write Block misalignment	WRITE_BLK_MISALIGN	1	R	[78:78]	0 (No)	
Read Block misalignment	READ_BLK_MISALIGN	1	R	[77:77]	0 (No)	
DSR implemented	DSR_IMP	1	R	[76:76]	0 (No)	
Reserved		2	R	[75:74]	0 (No)	
Device size	C_SIZE	12	R	[73:62]	According to device density	
Max. Read current at V_{CC}(min)	VDD_R_CURR_MIN	3	R	[61:59]	35 mA	
Max. Read current at V_{CC}(max)	VDD_R_CURR_MAX	3	R	[58:56]	45 mA	
Max. Write current at V_{CC}(min)	VDD_W_CURR_MIN	3	R	[55:53]	35 mA	
Max. Write current at V_{CC}(max)	VDD_W_CURR_MAX	3	R	[52:50]	45 mA	
Device size multiplier	C_SIZE_MULT	3	R	[49:47]	According to device density	
Erase group size	ERASE_GRP_SIZE	5	R	[46:42]	32 Erase groups	
Erase group size multiplier	ERASE_GRP_MULT	5	R	[41:37]	NAND08GAH0A	4
					NAND16GAH0D	8
Write Protect group size	WP_GRP_SIZE	5	R	[36:32]	32	
Write Protect Group Enable	WP_GRP_ENABLE	1	R	[31:31]	1 (Yes)	
Manufacturer Default ECC	DEFAULT_ECC	2	R	[30:29]	0 (None)	
Write speed factor	R2W_FACTOR	3	R	[28:26]	32	

Table 34. Card specific data register (continued)

Name	Field	Width [bits]	Cell type	CSD-slice	CSD-value	
Max. Write Data Block Length	WRITE_BL_LEN	4	R	[25:22]	NAND08GAH0A	512 bytes
					NAND16GAH0D	1024 bytes
Partial Blocks for Write Allowed	WRITE_BL_PARTIAL	1	R	[21:21]		0 (No)
Reserved				[20:20]		
Content protection application	CONTENT_PROT_APP	1	R	[16:16]		0 (No)
FileFormatGroup	FILE_FORMAT_GROUP	1	R/W	[15:15]		0 (No)
Copy Flag (OTP)	COPY	1	R/W	[14:14]		0 (No)
Permanent Write Protection	PERM_WRITE_PROTECT	1	R/W	[13:13]		0 (No)
Temporary Write Protection	TMP_WRITE_PROTECT	1	R/W/E	[12:12]		0 (No)
FileFormat	FILE_FORMAT	2	R/W	[11:10]		HD (Hard disk-like file system with partition table)
ECC Code 2 R/W/E None 0	ECC	2	R/W/E	[9:8]		0 (None)
CRC	CRC	7	R/W/E	[7:1]		0x10
Not used, always ‘1’		1	-	[0:0]		1

1. TBD stands for ‘to be defined’.

8.3.1 CSD_STRUCTURE

This field describes the version of the CSD structure.

Table 35. CSD register structure

CSD_STRUCTURE	CSD structure version	Valid for system specification version
0	CSD version No. 1.0	Allocated by MMCA
1	CSD version No. 1.1	Allocated by MMCA
2	CSD version No. 1.2	version 4.1 - 4.2
3	Version is coded in the CSD_STRUCTURE byte in the EXT_CSD register	

8.3.2 SPEC_VERS

Defines the MultiMediaCard system specification version supported by the device.

Table 36. System specification version

SPEC_VERS	System specification version number
0	Allocated by MMCA
1	Allocated by MMCA
2	Allocated by MMCA
3	Allocated by MMCA
4	Version 4.1 - 4.2
5 - 15	Reserved

8.3.3 TAAC

Defines the asynchronous part of the data access time.

Table 37. TAAC access time definition

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

8.3.4 NSAC

Defines the typical case for the clock dependent factor of the data access time. The unit for NSAC is 100 clock cycles. Therefore, the maximal value for the clock dependent part of the data access time is 25.5k clock cycles.

The total access time N_{AC} as expressed in the *Table 61: Timing values* is calculated based on TAAC and NSAC. It has to be computed by the host for the actual clock rate. The read access time should be interpreted as a typical delay for the first data bit of a data block.

8.3.5 TRAN_SPEED

Table 38 defines the clock frequency when not in high speed mode. For devices supporting version 4.0, and higher, of the specification, the value shall be 20 MHz (0x2A):

Table 38. Maximum bus clock frequency definition

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

8.3.6 CCC

The MultiMediaCard command set is divided into subsets (command classes). The card command class register CCC defines which command classes are supported by this device. A value of '1' in a CCC bit means that the corresponding command class is supported. For command class definition refer to *Table 18* and *Table 19*.

Table 39. Supported card command classes

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

8.3.7 READ_BL_LEN

The purpose of this field is to indicate the maximum data block length that is supported by the device when performing read operations.

The data block length is computed as $2^{READ\_BL\_LEN}$. The block length can therefore range from one to 16 Kbytes.

After power-up or software reset, the device defaults to operate in 512-byte block length.

Table 40. Data block length

READ_BL_LEN	Block length	Comment
0	2^0= 1 bytes	
1	2^1= 2 bytes	
...	...	...
10(1)	2^{10}= 1024 bytes	
11- 15	'00000'	

1. This bit is only available for the NAND16GAH0D. It must be set to '0' for the NAND08GAH0A.

8.3.8 READ_BL_PARTIAL

Defines whether partial block sizes can be used in block read commands.

- NAND08GAH0A and NAND16GAH0D (densities ≤2 Gbytes, byte access mode):
  - READ_BL_PARTIAL = '0' means that only the 512-byte and the READ_BL_LEN size can be used for block oriented data transfers.
  - READ_BL_PARTIAL = '1' means that smaller blocks can be used as well. The minimum block size will be equal to minimum addressable unit (one byte).

8.3.9 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 cross physical block boundaries are invalid.

WRITE_BLK_MISALIGN=1 signals that cross physical block boundaries are allowed.

8.3.10 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 cross physical block boundaries are invalid.

READ_BLK_MISALIGN=1 signals that cross physical block boundaries are allowed.

8.3.11 DSR_IMP

This parameter allows to select the configurable driver stage on the device. If set, a driver stage register (DSR) must be implemented also.

Table 41. DSR implementation code

DSR_IMP	DSR type
0	DSR is not implemented
1	DSR implemented

8.3.12 C_SIZE

This parameter is used to compute the NAND08GAH0A and NAND16GAH0D memory density.

The memory capacity of the device is computed from the entries C_SIZE, C_SIZE_MULT and READ_BL_LEN as follows:

Memory capacity = BLOCKNR * BLOCK_LEN, where:

- BLOCKNR = (C_SIZE+1) * MULT
- MULT = $2^{C\_SIZE\_MULT+2}$ (C_SIZE_MULT < 8)
- BLOCK_LEN = $2^{READ\_BL\_LEN}$, (READ_BL_LEN < 12)

Therefore, the maximal capacity which can be coded is 4096*512*2048 = 4 Gbytes.

Example: A 4 Mbyte device with BLOCK_LEN = 512 can be coded by C_SIZE_MULT = 0 and C_SIZE = 2047.

8.3.13 VDD_R_CURR_MIN, VDD_W_CURR_MIN

The minimum values for read and write currents at the minimum V_{CC} power supply (V_{CC}min) are coded as follows:

Table 42. Current consumption at V_{CC}min

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

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

8.3.14 VDD_R_CURR_MAX, VDD_W_CURR_MAX

The maximum values for read and write currents at the maximum V_{CC} power supply (V_{CC}max) are coded as follows:

Table 43. Current consumption at V_{CC}max

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

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

8.3.15 C_SIZE_MULT

This parameter is used for coding a factor MULT for computing the total device size (see *Section 8.3.12: C_SIZE*). The factor MULT is defined as $2^{C\_SIZE\_MULT+2}$.

Table 44. Multiply factor for the device size

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

8.3.16 ERASE_GRP_SIZE

The contents of this register is a 5 bit binary coded value used to calculate the size of the erasable unit of the device. The size of the erase unit (also referred to as erase group) is determined by the ERASE_GRP_SIZE and the ERASE_GRP_MULT entries of the CSD, using the following equation:

size of erasable unit = (ERASE_GRP_SIZE+1) * (ERASE_GRP_MULT +1)

This size is given as the minimum number of write blocks that can be erased in a single erase command.

8.3.17 ERASE_GRP_MULT

A 5 bit binary coded value used for calculating the size of the erasable unit of the device. See *Section 8.3.16: ERASE_GRP_SIZE* for detailed description.

8.3.18 WP_GRP_SIZE

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

8.3.19 WP_GRP_ENABLE

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

8.3.20 DEFAULT_ECC

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

8.3.21 R2W_FACTOR

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

Table 45. R2W_FACTOR

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

8.3.22 WRITE_BL_LEN

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

8.3.23 WRITE_BL_LEN

Block length for write operations. See *Section 8.3.7: READ_BL_LEN* for field coding.

Note that the support for 512 byte write access is mandatory for all cards. And that the cards has to be in 512 byte block length mode by default after power-up, or software reset. The purpose of this register is to indicate the maximum write data block length supported.

Defines whether partial block sizes can be used in block write commands.

- NAND08GAH0A and NAND16GAH0D (densities ≤2 Gbytes, byte access mode):
  - WRITE_BL_PARTIAL='0' means that only the 512 bytes and the WRITE_BL_LEN block size can be used for block oriented data write.
  - WRITE_BL_PARTIAL='1' means that smaller blocks can be used as well. The minimum block size is one byte.

8.3.24 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 46: File formats*.

8.3.25 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 device contents as a copy. The COPY bit is a one-time-programmable bit.

8.3.26 PERM_WRITE_PROTECT

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

8.3.27 TMP_WRITE_PROTECT

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

8.3.28 CONTENT_PROT_APP

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

8.3.29 FILE_FORMAT

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

Table 46. File formats

FILE_FORMAT_GRP	FILE_FORMAT	Type
0	0	Hard disk-like file system with partition table.
0	1	DOS FAT (floppy-like) with boot block only (no partition table).
0	2	Universal file format
0	3	Others/unknown
1	0,1,2,3	Reserved

8.3.30 ECC

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

Table 47. ECC type

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

8.3.31 CRC

The CRC field carries the check sum for the CSD contents. It is computed according to *Section 11: Error protection*. 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.

Table 48. CSD field command classes

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

8.4 Extended CSD register

The Extended CSD Register defines the device properties and selected modes. It is 512 bytes long. The 320 most significant bytes are the properties segment that defines the device capabilities and cannot be modified by the host. The 192 lower bytes are the modes segment that defines the configuration the device is working in.

These modes can be changed by the host by means of the SWITCH command.

Table 49. Extended CSD(1)

Name	Field	Size (bytes)	Cell type	CSD-slice	CSD-slice value
Properties segment					
Reserved(2)		7		[511:505]	TBD
Supported command sets	S_CMD_SET	1	R	[504]	00
Reserved(2)		288		[503:216]	TBD
Reserved(2)		1		[211]	TBD
Minimum Write performance for 8 bit at 52 MHz	MIN_PERF_W_8_52	1	R	[210]	0x08
Minimum Read performance for 8 bit at 52 MHz	MIN_PERF_R_8_52	1	R	[209]	0x08
Minimum Write performance for 8 bit at 26 MHz / 4 bit at 52 MHz	MIN_PERF_W_8_26_4_52	1	R	[208]	0x08
Minimum Read performance for 8 bit at 26 MHz / 4 bit at 52 MHz	MIN_PERF_R_8_26_4_52	1	R	[207]	0x08
Minimum Write performance for 4 bit at 26 MHz	MIN_PERF_W_4_26	1	R	[206]	0x08
Minimum Read performance for 4 bit at 26 MHz	MIN_PERF_R_4_26	1	R	[205]	0x08
Reserved(2)		1		[204]	TBD
Power class for 26 MHz at 3.6 V	PWR_CL_26_360	1	R	[203]	00
Power class for 52 MHz at 3.6 V	PWR_CL_52_360	1	R	[202]	00
Power class for 26 MHz at 1.95 V	PWR_CL_26_195	1	R	[201]	00

Table 49. Extended CSD(1) (continued)

Name	Field	Size (bytes)	Cell type	CSD-slice	CSD-slice value
Power class for 52 MHz at 1.95 V	PWR_CL_52_195	1	R	[200]	00
Reserved(2)		3		[199:197]	TBD
Card type	CARD_TYPE	1	R	[196]	01
Reserved(2)		1		[195]	TBD
CSD structure version	CSD_STRUCTURE	1	R	[194]	02
Reserved(2)		1		[193]	TBD
Extended CSD revision	EXT_CSD_REV	1	R	[192]	01
Modes segment					
Command Set	CMD_SET	1	R/W	[191]	00
Reserved(2)		1		[190]	TBD
Command set revision	CMD_SET_REV	1	RO	[189]	00
Reserved(2)		1		[188]	TBD
Power class	POWER_CLASS	1	R/W	[187]	00
Reserved(2)		1		[186]	TBD
High speed interface timing	HS_TIMING	1	R/W	[185]	00
Reserved(2)		1		[184]	TBD
Bus Width mode	BUS_WIDTH	1	WO	[183]	00
Reserved(2)		1		[182]	TBD
Reserved(2)		181		[180:0]	TBD

1. TBD stands for ‘to be defined’.
2. Reserved bits should read as ‘0’.

8.4.1 S_CMD_SET

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

Table 50. Supported command sets

Bit	Command set
7-5	Reserved
4	Allocated by MMCA
3	Allocated by MMCA
2	Allocated by MMCA
1	Allocated by MMCA
0	Standard MMC

8.4.2 MIN_PERF_a_b_ff

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

Table 51. R/W access performance values

Value	Performance
0x00	For devices not reaching the 2.4 Mbyte/s minimum value
0x08	Class A: 2.4 Mbyte/s and is the lowest allowed value for MMCplus and MMCmobile(16 x 150 Kbyte/s)
0x0A	Class B: 3.0 Mbyte/s and is the next allowed value (20 x 150 Kbyte/s)
0x0F	Class C: 4.5 Mbyte/s and is the next allowed value (30 x 150 Kbyte/s)
0x14	Class D: 6.0 Mbyte/s and is the next allowed value (40 x 150 Kbyte/s)
0x1E	Class E: 9.0 Mbyte/s and is the next allowed value (60 x 150 Kbyte/s) This is also the highest class which any MMCplus or MMC mobile card is needed to support in low bus category operation mode (26 MHz with 4 bit data bus). An MMCplus or MMCmobile card supporting any higher class than this has to support this class also (in low category bus operation mode).
0x28	Class F: Equals 12.0 Mbyte/s and is the next allowed value (80 x 150 Kbyte/s)
0x32	Class G: Equals 15.0 Mbyte/s and is the next allowed value (100 x 150 Kbyte/s)
0x3C	Class H: Equals 18.0 Mbyte/s and is the next allowed value (120 x 150 Kbyte/s)
0x46	Class J: Equals 21.0 Mbyte/s and is the next allowed value (140 x 150 Kbyte/s) This is also the highest class which any MMCplus or MMCmobile card is needed to support in mid bus category operation mode (26 MHz with 8 bit data bus or 52 MHz with 4 bit data bus). An MMCplus or MMCmobile card supporting any higher class than this has to support this class (in mid category bus operation mode) and class E also (in low category bus operation mode)
0x50	Class K: Equals 24.0 Mbyte/s and is the next allowed value (160 x 150 Kbyte/s)
0x64	Class M: Equals 30.0 Mbyte/s and is the next allowed value (200 x 150 Kbyte/s)
0x78	Class O: Equals 36.0 Mbyte/s and is the next allowed value (240 x 150 Kbyte/s)
0x8C	Class R: Equals 42.0 Mbyte/s and is the next allowed value (280 x 150 Kbyte/s)
0xA0	Class T: Equals 48.0 MByte/s and is the last defined value (320 x 150 Kbyte/s)

8.4.3 PWR_CL_ff_vvv

These fields define the supported power classes by the device. By default, the device has to operate at maximum frequency using 1 bit bus configuration, within the default maximum current consumption, as stated in the table below. If 4 bit/8 bit bus configurations, require increased current consumption, it has to be stated in these registers.

By reading these registers the host can determine the power consumption of the device in different bus modes. Bits [7:4] code the current consumption for the 8 bit bus configuration. Bits [3:0] code the current consumption for the 4 bit bus configuration

The PWR_52_vvv registers are not defined for 26 MHz MultiMediaCards.

Table 52. Power classes

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

The measurement for maximum rms current is the average of rms current consumption over a period of 100 ms.

The maximum peak current is defined as the absolute maximum value not to be exceeded.

The conditions under which the power classes are defined are:

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

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

8.4.4 CARD_TYPE

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

Table 53. Card type

Bit	Card type
7:2	Reserved
1	High speed MultiMediaCard at 52 MHz
0	High speed MultiMediaCard at 26 MHz

8.4.5 CSD_STRUCTURE

This field is a continuation of the CSD_STRUCTURE field in the CSD Register.

Table 54. CSD Register structure

CSD_STRUCTURE	CSD structure version	Valid for system specification version
0	CSD version No. 1.0	Allocated by MMCA
1	CSD version No. 1.1	Allocated by MMCA
2	CSD version No. 1.2	Version 4.1 - 4.2
3-255	Reserved for future use	

8.4.6 EXT_CSD_REV

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

Table 55. Extended CSD revision

EXT_CSD_REV	Extended CSD revision
255-3	Reserved
2	Revision 1.2
1	Revision 1.1
0	Revision 1.0

8.4.7 CMD_SET

Contains the binary code of the command set that is currently active in the device. It is set to '0' (Standard MMC) after power up and can be changed by a SWITCH command. Note that while changing the command set with the switch command, values according to the S_CMD_SET Register should be used, for example, bit0 set=0x01 for standard MMC.

8.4.8 CMD_SET_REV

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

Table 56. Standard MMC command set revisions

Code	MMC revision
255-1	Reserved
0	v4.0

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

8.4.9 POWER_CLASS

This field contains the 4-bit value of the selected power class for the device. The power classes are defined in *Table 57*. The host should be responsible of properly writing this field with the maximum power class it allows the device to use. The device uses this information to, internally, manage the power budget and deliver an optimized performance.

This field is 0 after power-up or software reset.

Table 57. Power class code

Bits	Description
[7:4]	Reserved
[3:0]	Device power class code (See *Table 47*)

8.4.10 HS_TIMING

This field is 0 after power-up, or software reset, thus selecting the backwards compatibility interface timing for the device. If the host writes 1 to this field, the device changes its timing to high speed interface timing (see *Table 9*).

8.4.11 BUS_WIDTH

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

Table 58. Bus mode values

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

8.5 RCA (relative card address) register

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

8.6 DSR (driver stage register) register

The 16-bit driver stage register (DSR) can be optionally used to improve the bus performance for extended operating conditions (depending on parameters like bus length, transfer rate or number of devices on the bus).

The CSD register contains the information concerning the DSR register usage.

The default value of the DSR register is ‘0x404’.

8.7 Status register

The Status register provides information about the device current state and completion codes for the last host command. The device status can be explicitly read (polled) with the SEND_STATUS command. The MultiMediaCard Status register structure is defined in *Section 8.7: Status register*.

Each of the Status register bit has three attributes:

- **Type**
  There are two types of Status register bits:
  - Error bit (E): it signals an error condition detected by the device. Error bits are cleared as soon as the response reporting the error is sent back.
  - Status bit (S): it provides information on the device status and do not alter the execution of the command being responded to. Status bits are non-volatile. They are set and cleared according to the device status.
- **Detection mode**
  Exceptions can be detected by the device either during the command interpretation and validation phase (Response mode) or during command execution phase (Execution mode).
  - Response mode (R) exceptions are reported in the response to the command that raised the exception. The command is not executed and the associated state transition does not take place.
  - Execution mode (X) exceptions are reported in the response to a STOP_TRANSMISSION command used to terminate the operation or in the response to a GET_STATUS command issued while the operation is being carried out or after the operation is completed. When an error is detected in X mode, the error will be reported in the response to the next command. Note that ADDRESS_OUT_OF_RANGE and ADDRESS_MISALIGN exceptions may be detected both in Response and Execution modes. The conditions for each one of the modes are explicitly defined in *Table 59*.

- **Clear condition**
  Status Register bits clear condition can be of three types:
  – The bit is cleared according to the device current state (A).
  – The bit value is always related to the previous command (B). The bit is cleared at reception of a valid command (with a delay of one command).
  – The bit is cleared by read operation (C).

Table 59. Status register

Bits	Identifier	Type	Detection mode	Value	Description	Clear condition
31	ADDRESS_OUT_OF_RANGE	E	R	'0'= no error '1'= error	Command argument not allowed	B
			X		Single/multiple block operation attempting to read or write beyond the device address space	B
30	ADDRESS_MISALIGN	E	R	'0'= no error '1'= error	Command using misaligned address, not matching the block length.	B
			X		Multiple block read/write operation attempting to read or write data block which does not align with the device memory blocks.	
29	BLOCK_LEN_ERROR	E	R	'0'= no error '1'= error	Transferred block length not allowed, or number of bytes transferred not matching the block length.	B
28	ERASE_SEQ_ERROR	E	R	'0'= no error '1'= error	An error occurred in the erase command sequence.	B
27	ERASE_PARAM	E	X	'0'= no error '1'= error	Invalid selection of erase groups during erase operation	B
26	WP_VIOLATION	E	X	'0'= not protected '1'= protected	Attempt to write to a write protected block.	B
25	CARD_IS_LOCKED	S	R	'0' = unlocked '1' = locked	This bit is set when the device is locked by the host	A
24	LOCK_UNLOCK_FAILED	E	X	'0'= no error '1'= error	This bit is set when a sequence or password error has been detected during a Lock/Unlock command or if there was an attempt to access a locked device	B
23	COM_CRC_ERROR	E	R	'0'= no error '1'= error	CRC check of the previous command failed.	B
22	ILLEGAL_COMMAND	E	R	'0'= no error '1'= error	Illegal command not legal for the current state	B
21	CARD_ECC_FAILED	E	X	'0'= success '1'= failure	Card internal ECC performed but failed to correct the data	B
20	CC_ERROR	E	R	'0'= no error '1'= error	Internal device controller error	B

Table 59. Status register (continued)

Bits	Identifier	Type	Detection mode	Value	Description	Clear condition
19	ERROR	E	X	‘0’= no error ‘1’= error	General or unknown error occurred during operation	B
18	UNDERRUN	E	X	‘0’		B
17	OVERRUN	E	X	‘0’		B
16	CID/ CSD_OVERWRITE	E	X	‘0’= no error ‘1’= error	Can be one of the following errors: - The CID register has been already written and cannot be overwritten - The read only section of the CSD does not match the device content. - An attempt to reverse the copy (set as original) or permanent WP (unprotected) bits was made.	B
15	WP_ERASE_SKIP	E	X	‘0’=not protected ‘1’= protected	Partial address space erased due to existing write protected blocks	B
14	Reserved (must be set to 0)					
13	ERASE_RESET	E	R	‘0’= cleared ‘1’= set	Erase sequence cleared before executing because an out of erase sequence command was received	B
12-9	CURRENT_STATE	S	R	0 = idle 1 = ready 2 = ident 3 = stby 4 = tran 5 = data 6 = rcv 7 = prg 8 = dis 9 = btst 10-15 = reserved	The state of the device 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.	A
8	READY_FOR_DATA	S	R	‘0’= not ready ‘1’= read	Corresponds to buffer empty signaling on the bus	A
7	SWITCH_ERROR	E	X	‘0’= not error ‘1’= error	If set, the device did not switch to the expected mode as requested by the SWITCH command	B
6-4	Reserved					
5				‘1’		
3, 2	Reserved for application specific commands					
1, 0	Reserved for manufacturer test mode					

9 Timings

All timing diagrams use the abbreviations shown in *Table 60*.

Bit P is actively driven High by the device respective to the host output driver, while bit Z is driven High by the pull-up resistors R_{CMD} and R_{DAT}. Actively driven P bits are less sensitive to noise superposition.

All timing values are shown in *Table 61*.

Table 60. Timing symbols

S	Start bit (= 0)
T	Transmitter bit (Host = 1, Device = 0)
P	One-cycle Pull-up (= 1)
E	End bit (=1)
Z	High Impedance state
D	Data bits
*	Repeater
CRC	Cyclic Redundancy Check bits (7 bits)

Table 61. Timing values

Timing	Min	Max	Unit
N_{CR}	2	64	Clock cycles
N_{ID}	5	5	Clock cycles
N_{AC}	2	10(TAAC f_{OP} + 100NSAC)	Clock cycles
N_{RC}	8	-	Clock cycles
N_{CC}	8	-	Clock cycles
N_{WR}	2	-	Clock cycles
N_{ST}	2	2	Clock cycles

9.1 Command and response timings

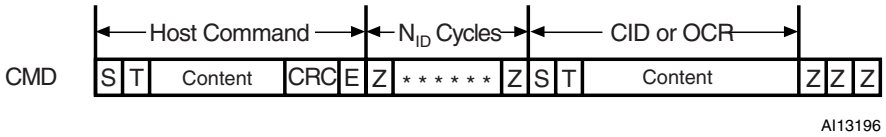
Both host command and device response are clocked out with the rising edge of the host clock.

9.1.1 Card identification and card operation conditions

The Card Identification (CMD2) and Card Operation Conditions (CMD1) commands are processed in the open-drain mode. The minimum delay between the host command and device response is N_{ID} clock cycles.

Figure 13 shows the identification timing diagram.

Figure 13. Identification timing diagram (Card Identification mode)

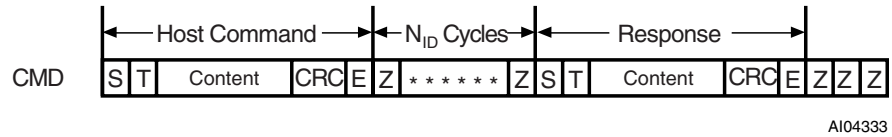


9.1.2 Assignment of relative card address

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

Figure 14 shows the SET_RCA timing diagram.

Figure 14. SET_RCA timing diagram (Card Identification mode)



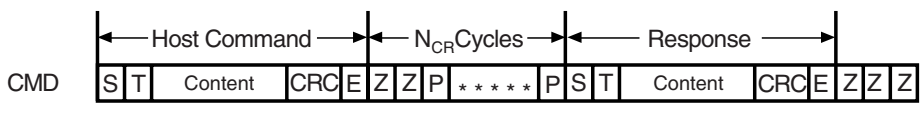
9.1.3 Data Transfer mode

After an RCA has been assigned to the device, it switches to Data Transfer mode. In this mode the CMD line is driven with push-pull drivers.

The command is followed by a two Z-bit period to allow direction switching on the bus, and by P bits pushed up by the responding device.

This timing diagram shown on *Figure 15* applies to all host command responses except for CMD1, CMD2, and CMD3.

Figure 15. Command response timing diagram (Data Transfer mode)

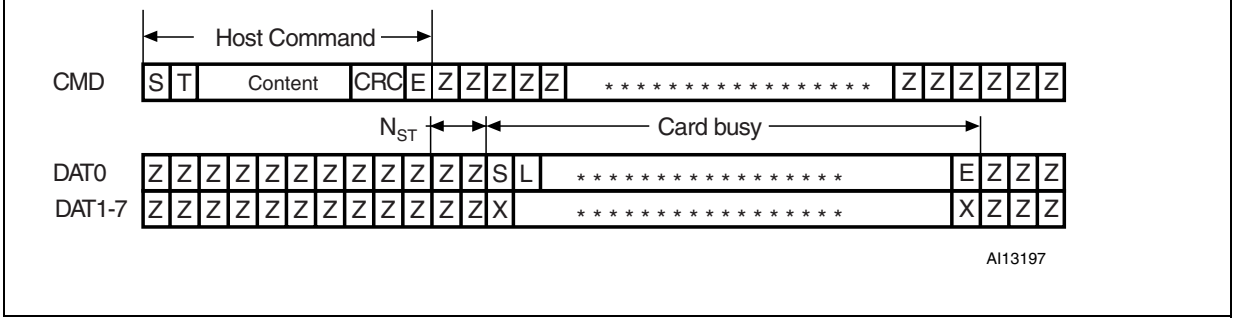


9.1.4 R1b responses

Some commands, like CMD6, may assert the busy flag and send back a R1 response. If the busy flag is asserted, this is done two clock cycles after the end bit of the command.

The DAT0 line is driven Low, DAT1-DAT7 lines are driven by the device though their values are not relevant.

Figure 16. R1b response timing diagram

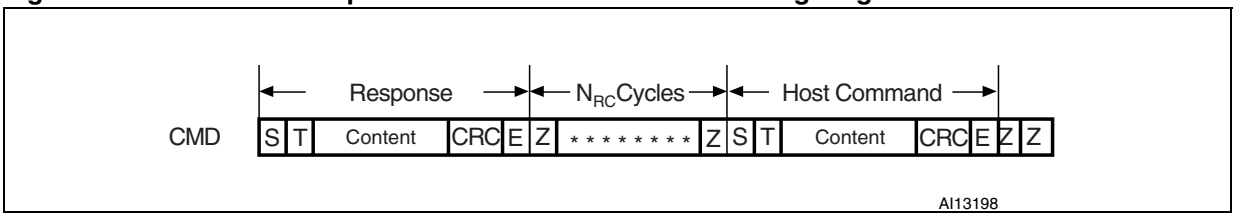


9.1.5 Last device response to Next Host command

After receiving the last device response, the host can start the next command transmission after a minimum delay of is N_{CR} clock cycles.

The timing diagram shown on *Figure 17* applies to all host commands.

Figure 17. Last device response to Next Host command timing diagram



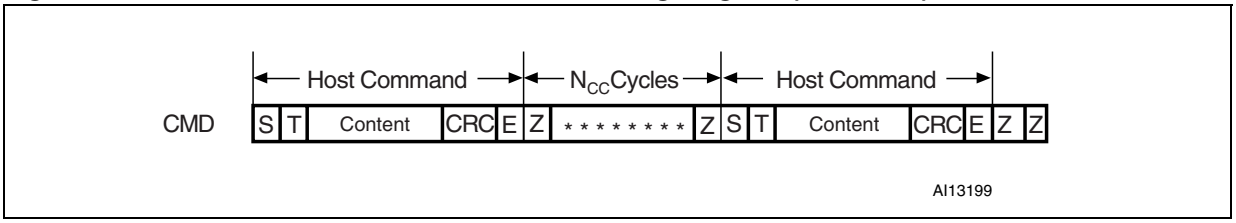
9.1.6 Last Host command to Next Host command

After the last command has been sent, the host can continue issuing the next command. A minimum delay of N_{CC} clock periods must be respected between the two commands.

If the device has not responded to the ALL_SEND_CID command after NID+1 clock periods, the host can conclude that no devices are present on the bus.

See *Figure 18* for a description of the timing diagram.

Figure 18. Command n end to CMD n+1 start timing diagram (all modes)



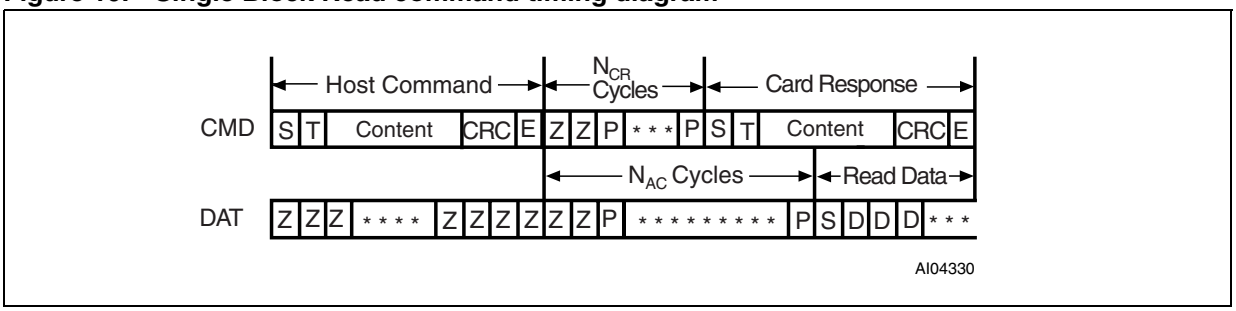
9.2 Data Read timings

9.2.1 Single Block Read

The host selects one device for data read operation by issuing a CMD7 command. It then sends a CMD16 command to set the valid block length for block oriented data transfer.

Figure 19 shows the timing diagram for a Single Block Read operation. 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. Data transmission starts N_{AC} clock cycles after the end bit of the host command. After the last data bit, the CRC check bits are suffixed to allow the host to check for transmission errors.

Figure 19. Single Block Read command timing diagram



9.2.2 Multiple Block Read

In Multiple Block Read mode, the device 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 20 describes the timing of the data blocks and *Figure 21* the response to a STOP_TRANSMISSION command. The data transmission stops two clock cycles after the end bit of the STOP_TRANSMISSION command.

Figure 20. Multiple Block Read command timing diagram

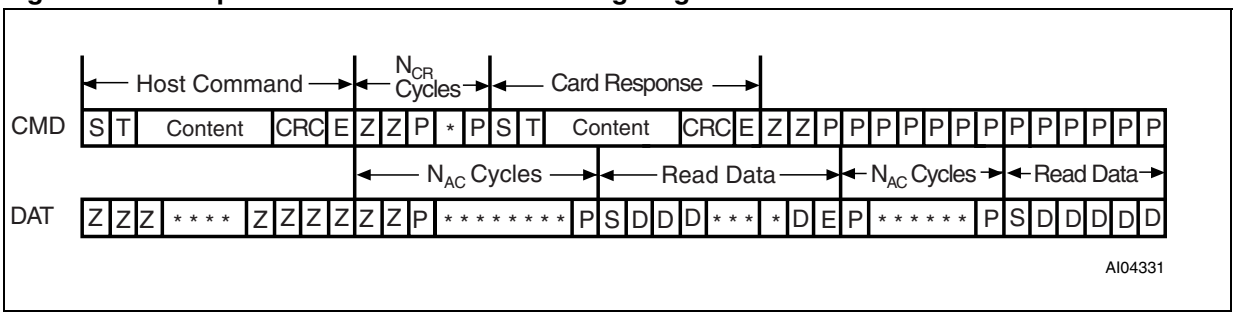
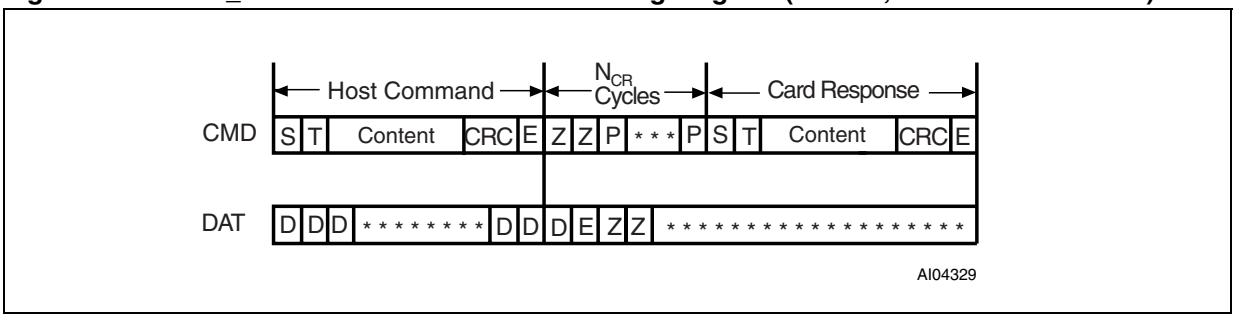


Figure 21. STOP_TRANSMISSION command timing diagram (CMD12, Data Transfer mode)



9.3 Data Write timings

9.3.1 Single Block Write

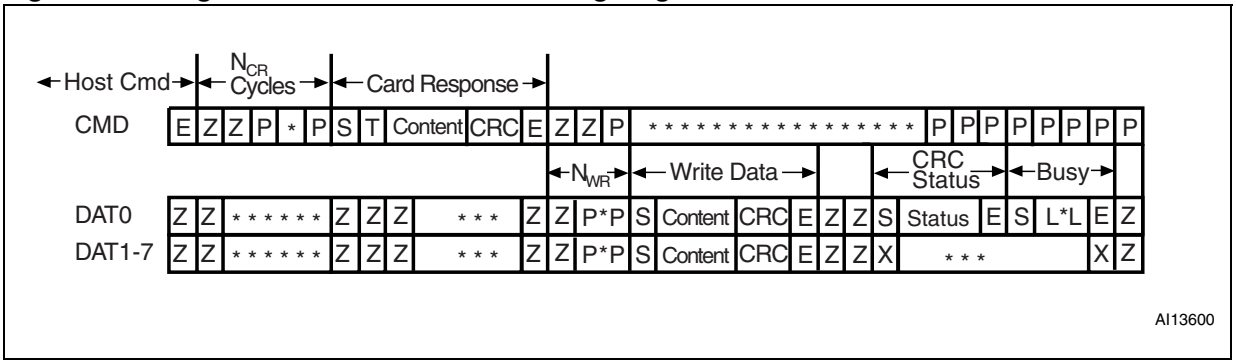
Before performing a data write operation, the host must select the device by issuing a CMD7 command, and set the valid block length for block oriented data transfer by issuing a CMD16 command.

The timing diagram for Single Block Write operation is given on *Figure 22*. The sequence starts with a single block write command (CMD24) which determines the start address. The data transfer from the host starts N_{WR} clock cycles after the device response was received.

The data is suffixed with CRC check bits to allow the device to check it for transmission errors. The device sends back the CRC check result as a CRC status token on the data line. In the case of transmission error the device sends a negative CRC status (‘101’). In the case of non erroneous transmission the device sends a positive CRC status (‘010’) and starts the data programming procedure.

During the write operation, the device notifies the host that it is busy by holding DAT0 Low. The data line goes High as soon as at least one receive buffer becomes free.

Figure 22. Single Block Write command timing diagram



9.3.2 Multiple Block Write

In Multiple Block Write mode, the device is sent a continuous flow of data blocks following the initial host write command. The data flow is terminated by a STOP_TRANSMISSION command (CMD12).
See *Figure 23* for a description of Multiple Block Write timing diagrams with and without device busy flag.

The STOP_TRANSMISSION command works in the same way as for Read operations. The device considers 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 25* is an example of an interrupted attempt to transmit the CRC status block. The end bit of the host command is followed, on the data line, with one more data bit, an end bit and two Z clock for switching the bus direction. In this case the received data block is considered incomplete and will not be programmed.

In an open-ended Multiple Block Write case the busy flag between the data blocks should be considered as buffer busy flag. As long as there is no free data buffer available the device should indicate this by pulling down the Dat0 line. The device stops pulling down DAT0 as soon as at least one receive buffer for the defined data transfer block length becomes free. After the device receives the stop command (CMD12), the following busy indication should be considered as programming busy and being directly related to the Programming state. As soon as the device completes the programming, it stops pulling down the Dat0 line.

In pre-defined Multiple Block Write case the busy flag between the data blocks should be considered as buffer busy flag similar to the open-ended multiple block case. After the device receives the last data block the following busy indication should be considered as programming busy and being directly related to the Programming state. The meaning of busy flag (from buffer busy to programming busy) changes at the same time with the state change (from rcv to prg). The busy flag remains “low” all the time during the process and is not released by the device between the state change from rcv to prg. As soon as the device completes the programming, it stops pulling down the Dat0 line.

See *Figure 24* and *Figure 25* show examples of timing diagrams corresponding to host stopping the data transmission during an active data transfer, while *Figure 26* and *Figure 27* describe scenarios of STOP_TRANSMISSION command received between data blocks transmission. In *Figure 26* the device is busy programming the last block while in *Figure 27* the device is idle. Unprogrammed data blocks remain in the input buffers and will be programmed as soon as the STOP_TRANSMISSION command is received and the device activates the busy flag.

Figure 23. Multiple Block Write command timing diagram

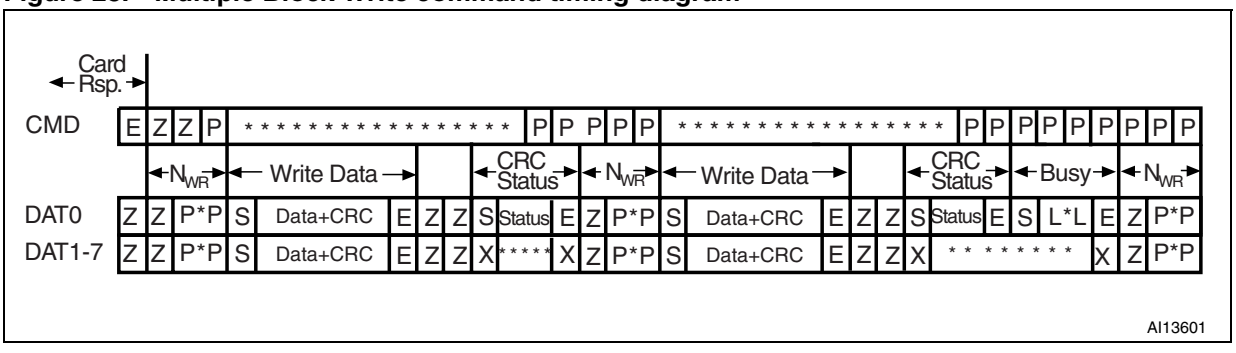


Figure 24. STOP_TRANSMISSION during data transfer from the host timing diagram

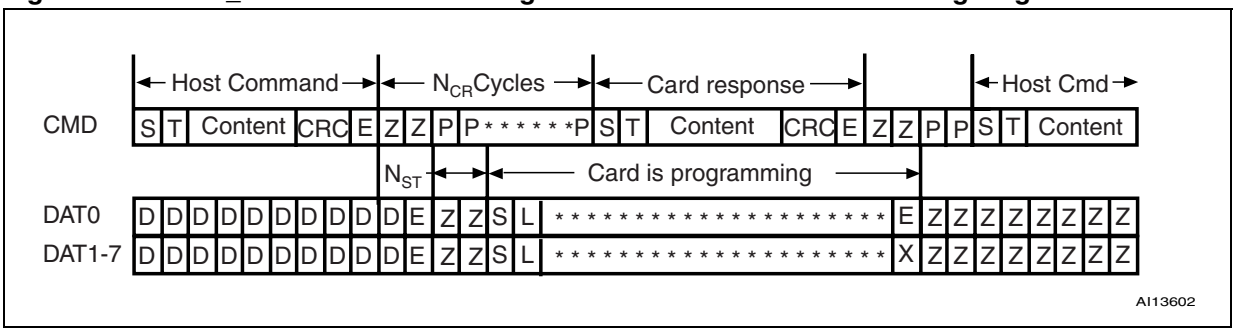
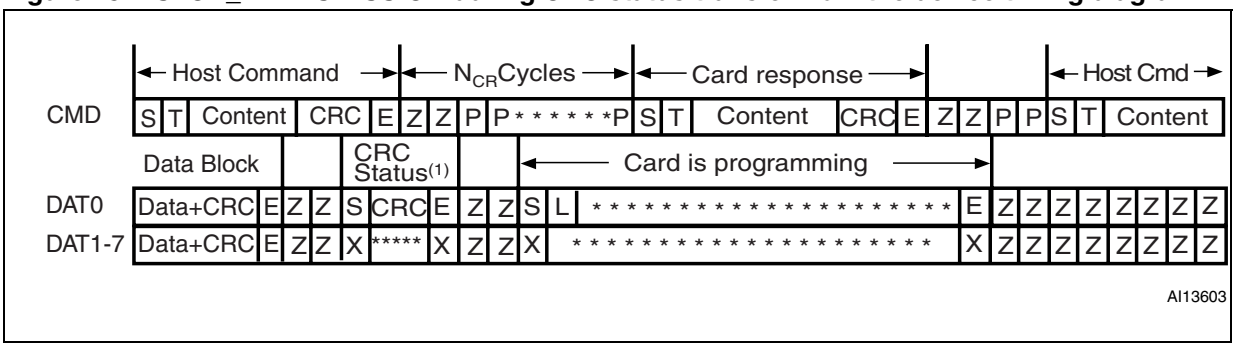


Figure 25. STOP_TRANSMISSION during CRC status transfer from the device timing diagram



1. The device CRC status response is interrupted by the host.

Figure 26. STOP_TRANSMISSION received after last data block (device busy)

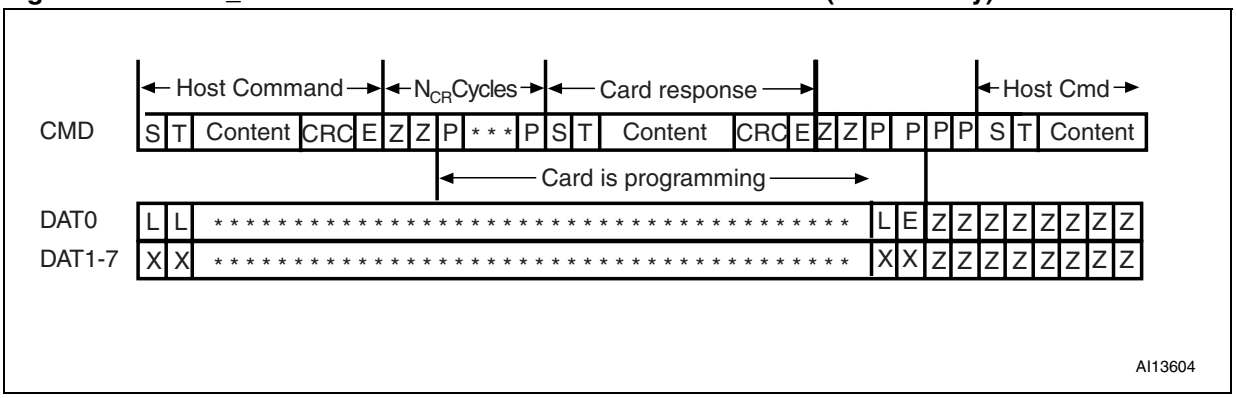
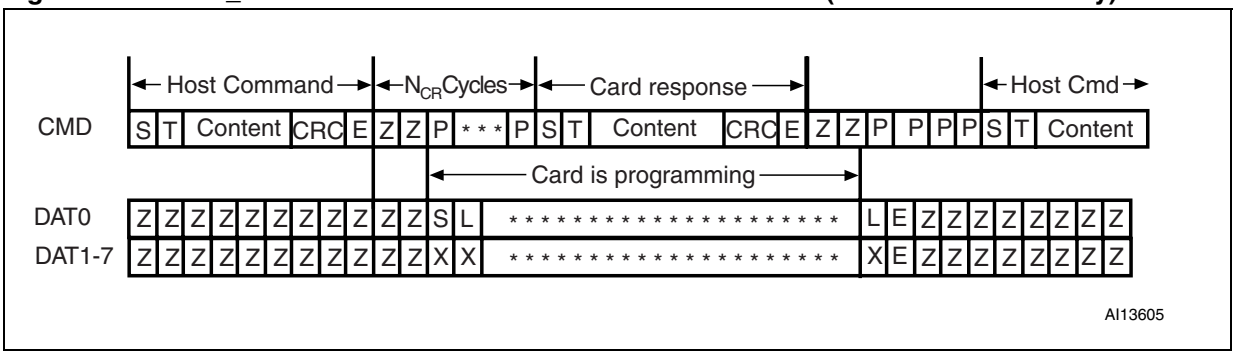


Figure 27. STOP_TRANSMISSION received after last data block (device becomes busy)



9.3.3 Erase, Set and Clear Write Protect

The host must first select the groups to be erased using the ERASE_GROUP_START and ERASE_GROUP_END commands (CMD35, CMD36). Once the ERASE command (CMD38) is issued, all the selected erase groups are erased.

Similarly, SET_WRITE_PROT and CLR_WRITE_PROT commands start a write operation as well. The device signals that it is busy by pulling the DAT0 line low for the duration of the erase or program operation. The timing diagrams are identical to the STOP_TRANSMISSION diagrams described in *Figure 27*.

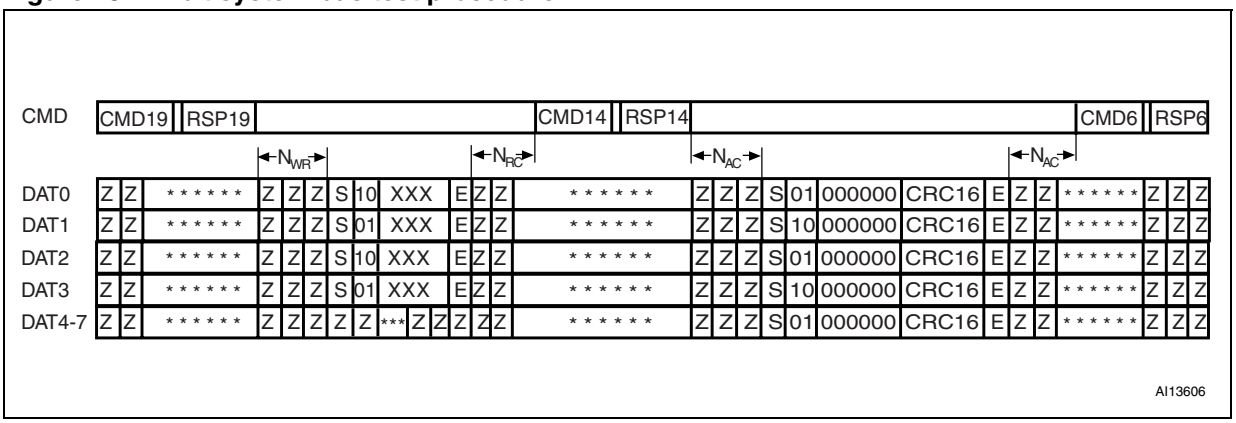
9.3.4 Reselecting a busy device

When a busy device which is currently in the disable state is reselected, it reinstate its busy signalling on the data line DAT0. The timing diagram for the command/response/busy transaction is given in *Figure 27*.

9.4 Bus test procedure timing

When in transfer state, the host can initiate a bus test procedure. This is done by issuing CMD19 command to the device. If there is no response to the CMD19, the host may assume that this function is not supported by the device and should read the status from the device by sending a CMD13 command.

Figure 28. 4-bit system bus test procedure



10 Serial peripheral interface (SPI) mode

The SPI mode is an optional communications protocol in Flash based MultiMediaCards. It is used to communicate with a microcontroller (host) through an SPI channel.

On power-up the MultiMediaCard defaults to the MultiMediaCard mode. The SPI mode is selected by asserting the $\overline{CS}$ signal during a reset command (CMD0). This may be after power-up but also anytime a reset command is issued. On entering the SPI mode, the device returns the SPI mode R1 response. Once the MultiMediaCard enters SPI mode it remains in this mode until the next power-up.

To run, the SPI mode implements a subset of the MultiMediaCard protocol and commands. This mode is intended for systems which require multiple devices, generally one, to operate, and have lower data transfer rates compared to MultiMediaCard protocol based systems.

The serial peripheral interface is a general purpose, synchronous interface designed to communicate with SPI hosts through four lines:

- $\overline{CS}$: host to device Chip Select line
- CLK: host to device clock line
- DataIn, DI: unidirectional host to device data line
- DataOut, DO: unidirectional device to host data line.

10.1 SPI bus topology

The host (master) selects a device (slave) by driving its $\overline{CS}$ pin Low. The $\overline{CS}$ pin must then be kept Low during the entire SPI communication process (command, response and data). During device programming, however, the host can de-assert the $\overline{CS}$ signal without affecting the programming operation.

Single block and multiple read and write operations are supported by the SPI channel as DataIn and DataOut are unidirectional.

Table 62 shows the MultiMediaCard pin assignment in SPI mode.

In SPI mode only the OCR, CSD and CID registers are accessible (see *Table 63*).

10.2 SPI electrical interface

The electrical interface in SPI mode is the same as the MultiMediaCard mode, except for the programmable device output drivers option which is not available in SPI mode.

10.3 SPI bus operating conditions

The SPI bus operating conditions are the same as for the MultiMediaCard mode.

Table 62. SPI interface pin configuration

MultiMediaCard mode			SPI mode		
Name	**Type(1)**	**Description**	**Name**	**Type**	**Description**
DAT3	I/O/PP	Data	$\overline{CS}$	I	Chip Select (active Low)
CMD	I/O/PP/ OD	Command/response	DI	I/PP	Data In
V_{SS1}	S	Supply voltage ground	V_{SS}	S	Supply voltage ground
V_{CC}, V_{CCQ}	S	Supply voltage	V_{CC}	S	Supply voltage
CLK	I	Clock	SCLK	I	Clock
V_{SS2}	S	Supply voltage ground	V_{SS2}	S	Supply voltage ground
DAT0	I/O/PP	Data	DO	O/PP	Data Out
DAT1	I/O/PP	Data	Not used		
DAT2	I/O/PP	Data	Not used		
DAT4	I/O/PP	Data	Not used		
DAT5	I/O/PP	Data	Not used		
DAT6	I/O/PP	Data	Not used		
DAT7	I/O/PP	Data	Not used		

1. S = power supply; I = input; O = output; PP = push-pull; OD = open drain; NC = Not connected (or logical High).

Table 63. MultiMediaCard registers in SPI mode

Name	Available	Width (in bytes)	Description
CID	Yes	16	Card identification data (serial number, manufacturer code, etc.)
RCA	No		
DSR	No		
CSD	Yes	16	Card specific data, card operation conditions
OCR	Yes	4	Operation condition register

10.4 SPI bus protocol

While the MultiMediaCard channel is based on command and data bit streams which are initiated by a start bit and terminated by a stop bit, the SPI channel is byte oriented. Every command or data block consists of 8-bit bytes and is byte aligned to the $\overline{CS}$ signal (i.e. the length is a multiple of 8 clock cycles).

Like in the MultiMediaCard protocol, the SPI messages consist of command, response and data-block tokens (see *Section 4*). All communication between host and device is controlled by the host (master). The host starts every bus transaction by asserting the $\overline{CS}$ signal Low.

The response behavior in the SPI mode differs from that in the MultiMediaCard mode in the three following aspects:

- The selected device always responds to the command
- Additional (8, 16 & 40 bit) response structures are used
- When the device encounters a data retrieval problem, it responds with an error response (which replaces the expected data block) instead of a timeout in the MultiMediaCard mode.

Only Single and Multiple Block Read/Write operations are supported in SPI mode (sequential mode is not supported).

In addition to the command response, a special data response token is returned for every data block sent to the device during write operations. A data block may be as big as one device write block and as small as a single byte. Partial block read/write operations are enabled by device options specified in the CSD Register.

10.4.1 Mode selection

The MultiMediaCard wakes up in the MultiMediaCard mode. It enters the SPI mode if the $\overline{CS}$ signal is asserted (negative) during the reception of the reset command (CMD0). The SPI mode can also be selected from other states than the Idle state (the state the device enters after power-up). Every time the device receives CMD0, even while the device is in the Inactive state, the $\overline{CS}$ signal is sampled.

If the device recognizes that the MultiMediaCard mode is required ($\overline{CS}$ signal is High), it does not respond to the command and remains in the MultiMediaCard mode. If the SPI mode is required ($\overline{CS}$ signal is Low), the device switches to the SPI mode and responds with the SPI mode R1 response.

The only way to return to the MultiMediaCard mode is by a power-down cycle (turn the power off an on). In SPI mode, the MultiMediaCard protocol state machine is not observed. All the MultiMediaCard commands supported in the SPI mode are always available.

10.4.2 Bus transfer protection

Every MultiMediaCard token transferred on the bus is protected by CRC (Cyclic Redundancy Check) bits. In SPI mode, the MultiMediaCard offers a non-protected mode which enables systems built with reliable data links to exclude the hardware or firmware required for implementing the CRC generation and verification functions.

In the non-protected mode, the CRC bits of the command, response and data tokens are still required in the tokens. However, they are defined as ‘don’t care’ for the transmitter and ignored by the receiver.

The SPI interface is initialized in the non-protected mode. However, the RESET command (CMD0), which is used to switch the device to SPI mode, is received by the device while in MultiMediaCard mode and, therefore, must have a valid CRC field.

Since CMD0 has no arguments, the content of all the fields, including the CRC field, are constants and need not be calculated in run time. A valid reset command is:

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

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

10.4.3 Data Read

The SPI mode supports Single and Multiple Block Read operations. The main difference between SPI and MultiMediaCard modes is that the data and the response are both transmitted to the host on the DataOut signal (refer to *Figure 29* and *Figure 30*). Therefore the device response to the STOP_COMMAND may cut-short and replace the last data block.

Figure 29. SPI Single Block Read operation

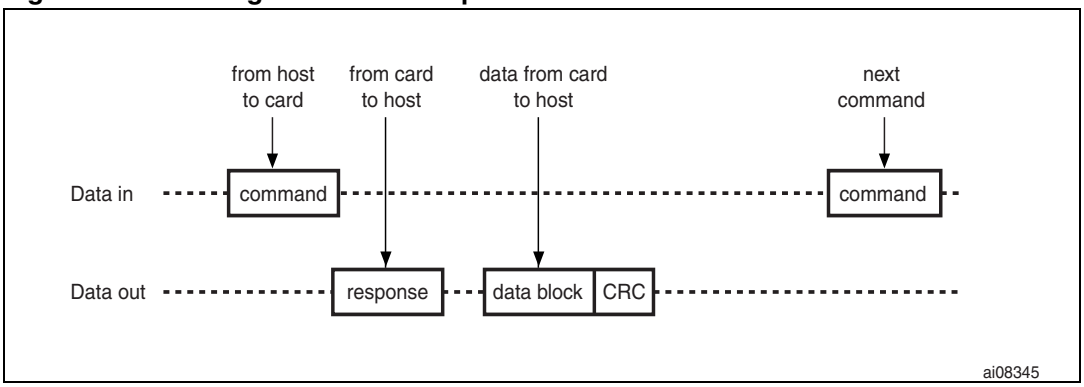
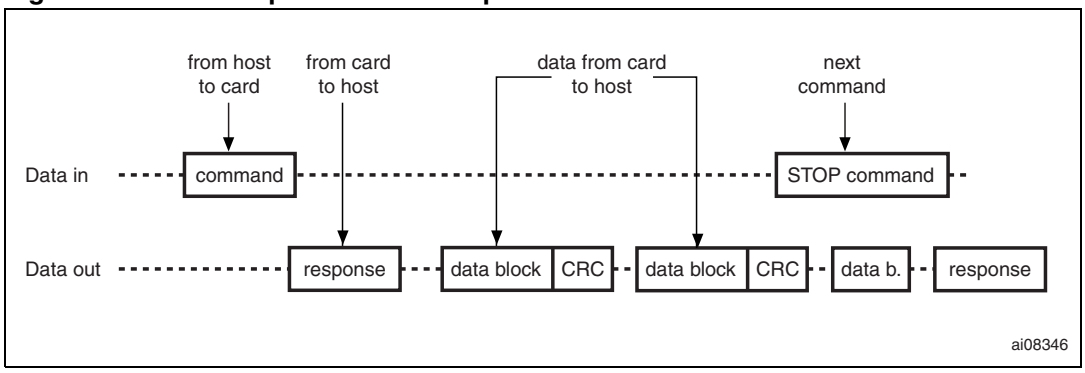


Figure 30. SPI Multiple Block Read operation



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

Open-ended Multiple Block Read

The number of blocks for the Multiple Block Read operation is not defined. The device continuously transfers data blocks until a stop transmission command is received.

Multiple Block Read with pre-defined block count

The device transfers the requested number of data blocks and terminates the transaction. A Stop command is not required at the end of this type of Multiple Block Read, unless it is terminated with an error.
In order to start a Multiple Block Read with pre-defined block count the host must use the SET_BLOCK_COUNT command (CMD23) just before issuing the READ_MULTIPLE_BLOCK (CMD18) command. Otherwise the device starts an open-ended Multiple Block Read that can be stopped using the STOP_TRANSMISION command.

The host can abort reading at any time, within a multiple block operation, regardless of the its type. Transaction abort is done by sending the STOP_TRANSMISION command.

If the host provides an out-of-range address as an argument to either CMD17 or CMD18, or if the currently defined block length is illegal for a read operation, the device rejects the command and responds with the ADDRESS_OUT_OF_RANGE or BLOCK_LEN_ERROR bit set, respectively.

If the host sets the argument of the SET_BLOCK_COUNT command (CMD23) to all 0's, then the command is accepted, however, a subsequent read follows the open-ended Multiple Block Read protocol (STOP_TRANSMISSION command - CMD12 - is required).

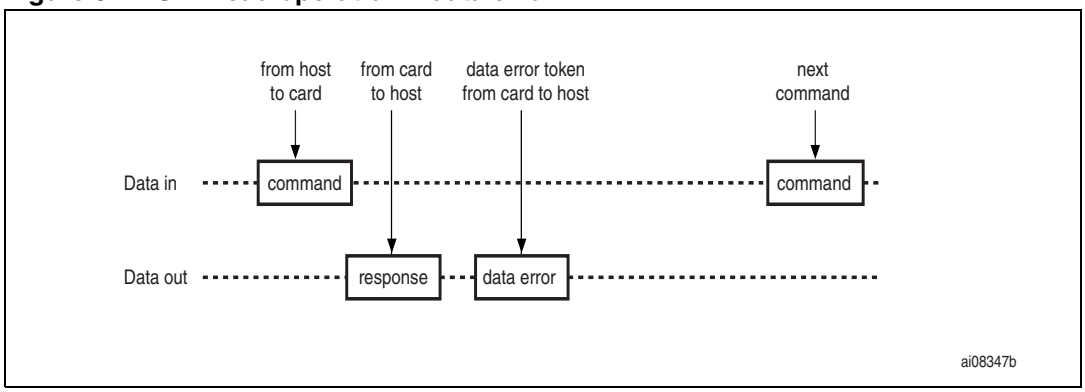
In case of a data retrieval error (such as out of range, address misalignment or internal error) detected during data transfer, the device does not transmit any data. Instead (as opposed to MultiMediaCard mode where the device times out), a special data error token is sent to the host. *Figure 31* shows a single block read operation terminated with an error token and not a data block.

Multiple Block Read operation can be terminated in the same way, with the error token replacing a data block anywhere in the sequence. The host must then abort the operation by sending the STOP_TRANSMISSION command.

If the host sends a STOP_TRANSMISSION command after the device transmitted the last block of a Multiple Block Read with a pre-defined number of blocks, it is responded to as for an illegal command.

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

Figure 31. SPI Read operation – data error



10.4.4 Data Write

The SPI mode supports single block and Multiple Block Write commands. Upon reception of a valid write command (CMD24 or CMD25), the device responds with a response token and waits for the host to send a data block. The 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 29*). If a CRC error is detected it is reported in the data-response token and the data block is programmed.

Figure 32. SPI Single Block Write operation

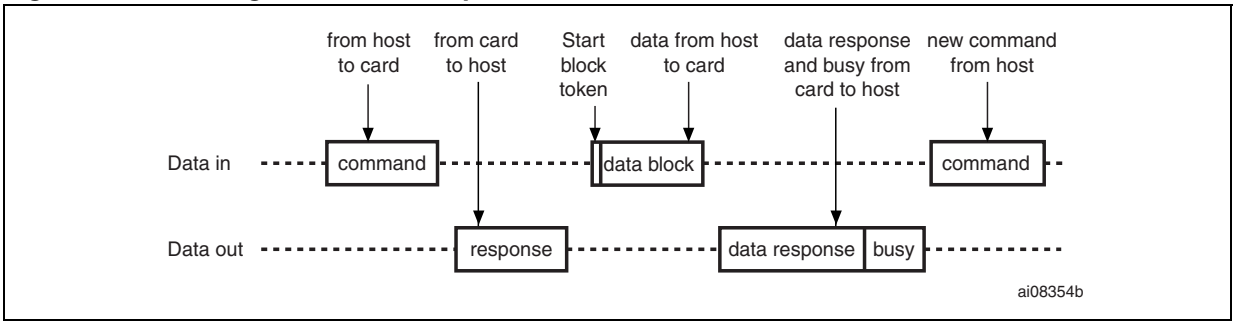
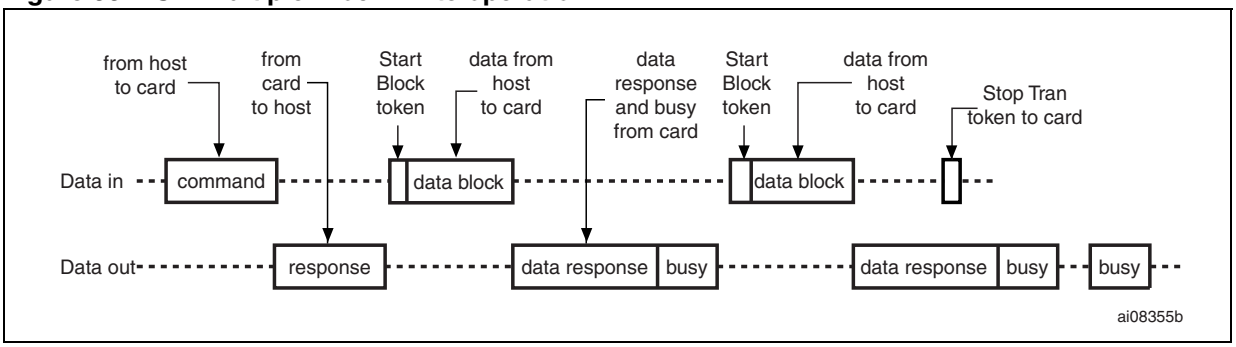


Figure 33. SPI Multiple Block Write operation



Every data block has a ‘Start block’ token prefix of one byte.

After receiving a data block, the device responds with a data-response token. If the data block has been received without errors, it is programmed. As long as the device is busy programming, a continuous stream of busy tokens is sent to the host (effectively holding the DataOut line Low).

In Multiple Block Write operations the host stops the transmission by sending the ‘Stop tran’ token instead of the ‘Start block’ token at the beginning of the next block.

Two types of Multiple Block Write transactions, identical to the Multiple Block Read, are defined (the host can use either of them at any time):

Open-ended Multiple Block Write

The number of blocks for the Multiple Block Write operation is not defined. The device accepts and programs all received data blocks until it receives a ‘Stop tran’ token.

Multiple Block Write with pre-defined block count

The device accepts the requested number of data blocks and then terminates the transaction. The ‘Stop tran’ token is not required at the end of this type of Multiple Block Write operation, unless the operation is terminated with an error. In order to start a Multiple

Block Write operation with pre-defined block count the host must issue the SET_BLOCK_COUNT command (CMD23) just before sending the WRITE_MULTIPLE_BLOCK (CMD25) command. Otherwise the device starts an open-ended Multiple Block Write operation that can be stopped using the ‘Stop tran’ token.

The host can abort writing at any time, within a Multiple Block Write operation, regardless of its type. Transaction abort is done by sending the ‘Stop tran’ token. If a Multiple Block Write operation with pre-defined block count is aborted, the data in the remaining blocks is not defined.

If the host provides an out-of-range address as an argument to either CMD17 or CMD18, or if the currently defined block length is illegal for a read operation, the device rejects the command, remains in Tran state and responds with the ADDRESS_OUT_OF_RANGE or BLOCK_LEN_ERROR bit set, respectively.

If the host sets the argument of the SET_BLOCK_COUNT command (CMD23) to all 0’s, then the command is accepted, however, a subsequent write follows the open-ended Multiple Block Write protocol (STOP_TRANSMISSION command - CMD12 - is required).

If the device detects a CRC error or a programming error (such as write protect violation, out of range, address misalignment or internal error) during a Multiple Block Write operation (both types) it reports the failure in the data-response token and ignores any further incoming data blocks. The host must than abort the operation by sending the ‘Stop tran’ token.

If the host uses partial blocks whose accumulated length is not block aligned, and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the device detects the block misalignment error upon reception of the first misaligned block, aborts the write operation, and ignores all further incoming data. The host must abort the operation by sending the ‘Stop tran’ token. The devices then responds by setting the ADDRESS_MISALIGN bit.

Once the programming operation has completed (either successfully or with an error), the host must check the results of the programming (or the cause of the error if already reported in the data-response token) using the SEND_STATUS command (CMD13).

If the host sends a ‘Stop tran’ token after the device received the last data block of a multiple block operation with pre-defined number of blocks, it is interpreted as the beginning of an illegal command and the device responds accordingly.

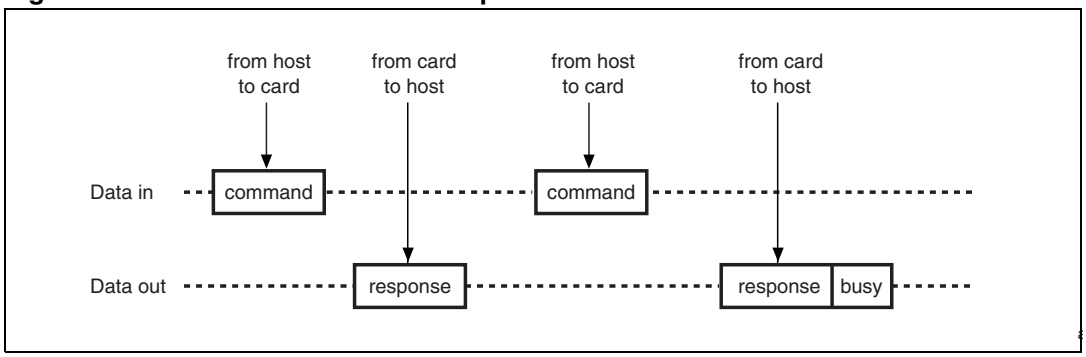
When the device is busy, resetting the $\overline{CS}$ signal does not terminate the programming process. The device simply releases the DataOut line (tri-state) and continues with the programming. If the device is reselected before the programming is finished, the DataOut line is forced back to Low and all commands are rejected.

Resetting a device (using CMD0) terminates any pending or active programming operations. This may destroy the data formats on the device. It is the responsibility of the host to prevent this.

10.4.5 Erase and Write Protect management

The Erase and Write Protect management procedures in the SPI mode are identical to those of the MultiMediaCard mode. While the device is erasing or changing the write protection bits of the predefined erase groups list, it is in a busy state and holds the DataOut line Low. *Figure 34* illustrates a 'no data' bus transaction with and without busy signalling.

Figure 34. Erase and Write Protect operations



10.4.6 Read the CID and CSD registers

Unlike the MultiMediaCard protocol (where the register contents is sent as a command response), reading the contents of the CSD and CID registers in SPI mode is a simple read-block transaction. The device responds with a standard response token (see *Figure 31*) 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 device TAAC since this value is stored in the CSD.

Refer to *Table 71* for timing values. For consistency, read CID transaction is identical to read CSD.

10.4.7 Reset sequence

The MultiMediaCard requires a defined reset sequence. After power-up reset or CMD0 (software reset) the device enters an idle state. In this state the only legal host commands are CMD1 (SEND_OP_COND) and CMD58 (READ_OCR).

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

In SPI mode, as opposed to MultiMediaCard mode, CMD1 has no operands and does not return the contents of the OCR register. Instead, the host may use CMD58 (available in SPI mode only) to read the OCR register.

Furthermore, it is of the responsibility of the host to refrain from gaining access to a device that does not support its voltage range.

The usage of CMD58 is not restricted to the initializing phase, the command can be issued at any time. The host must poll the device (by repeatedly sending CMD1) until the 'in-idle-state' bit in the device response indicates (by being cleared to 0) that the device has completed its initialization processes and is ready for the next command.

10.4.8 Clock control

The SPI bus clock signal can be used by the SPI host to put the device into energy saving mode or to control the data flow (to avoid under-run or over-run conditions) on the bus. The host is allowed to change the clock frequency or shut it down.

There are a few restrictions the SPI host must follow:

- The bus frequency can be changed at any time (under the restrictions of maximum data transfer frequency, defined by the MultiMediaCards)
- The clock must be running for the MultiMediaCard to output data or response tokens. After the last SPI bus transaction, the host must provide 8 (eight) clock cycles for the device to complete the operation before shutting down the clock. Throughout this 8-clock period the state of the $\overline{CS}$ signal is irrelevant, it can be asserted or de-asserted.

The various SPI bus transactions are listed below:

- A command / response sequence, 8 clocks after the device response end bit. The $\overline{CS}$ signal can be asserted or de-asserted during these 8 clocks
- A read data transaction, 8 clocks after the end bit of the last data block
- A write data transaction, 8 clocks after the CRC status token
- The host is allowed to shut down the clock of a “busy” device. The MultiMediaCard completes the programming operation regardless of the host clock. However, the host must provide a clock edge for the device to turn off its busy flag. Without a clock edge the MultiMediaCard (unless previously disconnected by de-asserting the $\overline{CS}$ signal) will force the DataOut line Low, permanently.

10.4.9 Error conditions

CRC and illegal command

All commands are (optionally) protected by CRC (cyclic redundancy check) bits. If the addressed MultiMediaCard's CRC check fails, the COM_CRC_ERROR bit is set in the device response. Similarly, if an illegal command has been received the ILLEGAL_COMMAND bit is set in the device response.

There are different kinds of illegal commands:

- Commands that belong to classes not supported by the MultiMediaCard (like interrupt and I/O commands)
- Commands not allowed in SPI mode
- Commands that are not defined (like CMD47).

10.4.10 Read, Write, Erase and Forced Erase timeout conditions

The time period after which a timeout condition for read/write/erase operations occurs are (device independent) 10 times longer than the typical access/program times for the operations listed below. A device must complete the command within this time period, or give up and return an error message. If the host does not get a response within the defined timeout time it must assume that the device is not going to respond and try to recover (e.g. reset the device, power cycle, reject).

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. These device parameters define the typical delay between the end bit of the read command and the start bit of the data block. This number is device dependent.

- Write

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

- Erase

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

- Forced Erase

  The Forced Erase timeout is specified in *Table 16*.

- Read ahead in Multiple Block Read operation

  In Multiple Block Read operations, in order to improve the read performance, the device may fetch data from the memory array, ahead of the host. In this case, when the host is reading the last addresses of the memory, the device attempts to fetch data beyond the last physical memory address and generates an ADDRESS_OUT_OF_RANGE error.

  Therefore, even if the host times the stop transmission command to stop the device immediately after the last byte of data was read, the device may already have generated the error, and it will show in the response to the stop transmission command. The host should ignore this error.

10.4.11 Memory array partitioning

It is the same as in the MultiMediaCard mode (see *Section 3*).

10.4.12 Lock/Unlock commands

In SPI mode the Lock and Unlock commands are the same as in MultiMediaCard mode (see *Section 5.5.3* and *Section 5.5.4*).

10.4.13 Application specific commands

The only difference between the MultiMediaCard and SPI modes is the APP_CMD status bit which is not available in SPI mode.

10.5 SPI mode commands

All the SPI commands are 6 bytes long. The command always starts with the left bit of the string, which corresponds to the command code. See *Table 64* for details of the command format.

The commands in SPI mode are divided into several classes as in MultiMediaCard mode. However, the supported and available classes are different for each mode. See *Table 65* for details.

Table 66 gives a detailed description of the commands supported in SPI mode. If no argument is required in the command, the value of the field should be set to '0'. Reserved commands are reserved in both MultiMediaCard and SPI modes. The contents of the command index field is binary: for example it is '000000' for CMD0 and '100111' for CMD39.

Table 64. Command format

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
Descriptions	Start bit	Transmission bit	Command index	Argument	CRC	End bit

Table 65. Command classes in SPI mode

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

Table 66. Commands and arguments

CMD INDEX	SPI mode	Argument	Response	Abbreviation	Command description
CMD0	Yes	None	R1	GO_IDLE_STATE	Resets the device
CMD1	Yes	None	R1	SEND_OP_COND	Activates the device Initialization process
CMD2	No				
CMD3	No				
CMD4	No				
CMD5	Reserved				
CMD6	Yes	[31:26] Set to '0' [25:24] Access [23:16] Index [15:8] Value [7:3] Set to '0' [2:0] Cmd Set	R1b	SWITCH	Switches the device operating mode or modifies the EXT_CSD register
CMD7	No				
CMD8	Yes	[31:0] stuff bits	R1	SEND_EXT_CSD	Ask the address device to send back its EXT_CSD register as a data block
CMD9	Yes	None	R1	SEND_CSD	Asks the selected device to send its Card Specific Data (CSD)
CMD10	Yes	None	R1	SEND_CID	Asks the selected device to send its Card Identification Data (CID)
CMD12	Yes	None	R1	STOP_TRANSMISSION	Stops transmission on Multiple Block Read
CMD13	Yes	None	R2	SEND_STATUS	Asks the selected device to send its Status Register
CMD14	Illegal command				
CMD15	No				
CMD16	Yes	[31:0] block length	R1	SET_BLOCKLEN	Selects a block length (in Bytes) for all following block commands (read and write)(1)
CMD17	Yes	[31:0] data address(2)	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command(3)

Table 66. Commands and arguments (continued)

CMD INDEX	SPI mode	Argument	Response	Abbreviation	Command description
CMD18	Yes	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously transfers data blocks from device to host until interrupted by a stop command or the requested number of data blocks have been transmitted
CMD19	Illegal command				
CMD20	No				
CMD21 CMD22	Reserved				
CMD23	Yes	[31:16] set to 0 [15:0] number of blocks	R1	SET_BLOCK_COUNT	Defines the number of blocks which are going to be transferred in the next Multiple Block Read or Write command
CMD24	Yes	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKEN command(4)
CMD25	Yes	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a Stop tran prefix or the requested number of blocks have been received
CMD26	No				
CMD27	Yes	None	R1	PROGRAM_CSD	Programs the programmable bits of the CSD
CMD28	Yes	[31:0] data address	R1b(5)	SET_WRITE_PROT	If the device 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 register (WP_GRP_SIZE)
CMD29	Yes	[31:0] data address	R1b	CLR_WRITE_PROT	If the device 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 device has the write protection features, this command asks the device to send the status of the write protection bits(6)

Table 66. Commands and arguments (continued)

CMD INDEX	SPI mode	Argument	Response	Abbreviation	Command description
CMD31	Reserved				
CMD32 ... CMD34	Reserved These command indexes cannot be used for reasons of backward compatibility with older versions of the MultiMediaCard				
CMD35	Yes	[31:0] data address	R1	ERASE_GROUP_ START	Sets the address of the first erase group within a range to be selected for erase
CMD36	Yes	[31:0] data address	R1	ERASE_GROUP_ END	Sets the address of the last erase group within a continuous range to be selected for erase
CMD37	Reserved These command indexes cannot be used for reasons of backward compatibility with older versions of the MultiMediaCard				
CMD38	Yes	[31:0] stuff bits	R1b	ERASE	Erases all previously selected erase groups
CMD39	No				
CMD40	No				
CMD41	Reserved				
CMD42	Yes	[31:0] stuff bits	R1b	LOCK_UNLOCK	Used to set/reset the password or Lock/Unlock the device. The size of the data block is defined by the SET_BLOCK_LEN command
CMD43 ... CMD54	Reserved				
CMD55	Yes	[31:0] stuff bits	R1	APP_CMD	Notifies the device 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 device or to read a data block from the device for general purpose/application specific commands. The size of the data block is defined by the SET_BLOCK_LEN command.
CMD57	Reserved				

Table 66. Commands and arguments (continued)

CMD INDEX	SPI mode	Argument		Response	Abbreviation	Command description
CMD58	Yes	NAND08GAH0A NAND16GAH0D	None	R3	READ_OCR	Reads the OCR register of a device
CMD59	Yes		[31:1] stuff bits [0] CRC option	R1	CRC_ON_OFF	Turns the CRC option on or off CRC option bit = '1' on CRC option bit = '0' off
CMD60 -63	No					

1. See *Section 8.3: Card specific data register (CSD)* for the value of the default block length.
2. Data address on the NAND08GAH0A and the NAND16GAH0D is a 32-bit byte address.
3. The data transferred must not cross a physical block boundary unless READ_BLK_MISALIGN is set in the CSD register.
4. The data transferred must not cross a physical block boundary unless WRITE_BLK_MISALIGN is set in the CSD register.
5. R1b: R1 response with an optional busy flag (see *Section 10.6.2*).
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 are set to zero.
7. RD/WR is set to '1' if the host receives a data block from the device, and to '0' if the host sends a data block to the device.

10.6 SPI mode responses

In SPI mode like in MultiMediaCard mode, responses are transmitted most significant bit (MSB) first.

There are different types of responses:

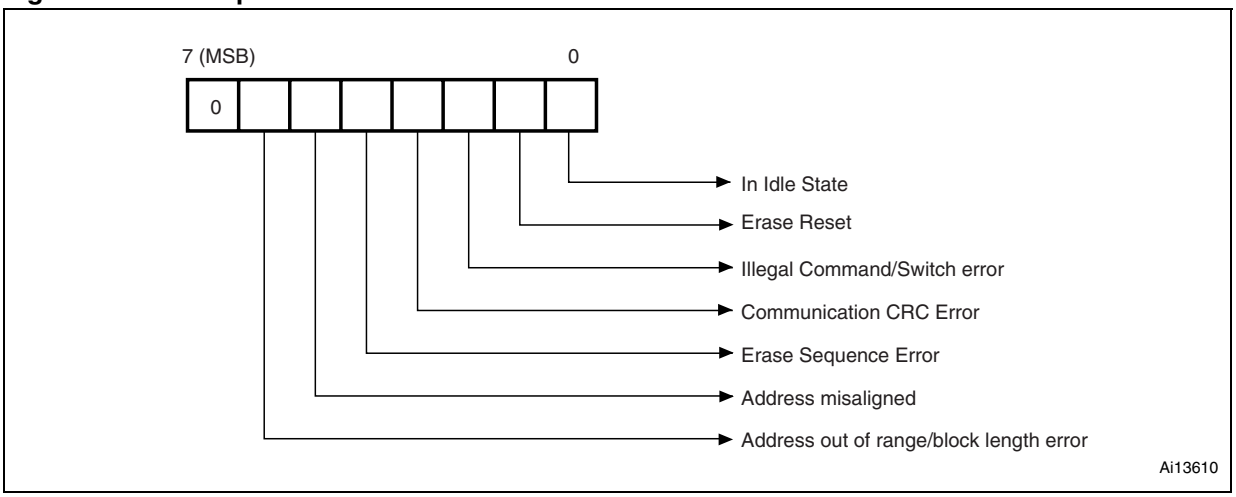
10.6.1 R1 format

This is the format of the response sent by the device to any command received except for the SEND_STATUS and READ_OCR commands (See R2 and R3, respectively).

R1 format responses are one byte long. Their most significant bit is always ‘0’ and the other bits are error bits. Any one of the error bits going High, (set to ‘1’) indicates an error.

Refer to *Figure 35* for the structure of the R1 response format and to *Table 68* for the meaning of the bits.

Figure 35. R1 response format

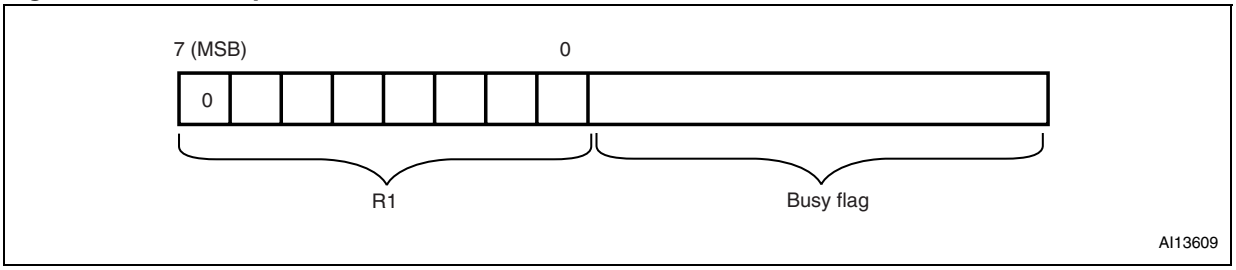


10.6.2 R1b format

It consists of the R1 response plus an optional busy flag.

The busy flag does not have a fixed number of bytes. When its value is zero, the device is busy. When its value is different from zero, the device is ready.

Figure 36. R1b response format

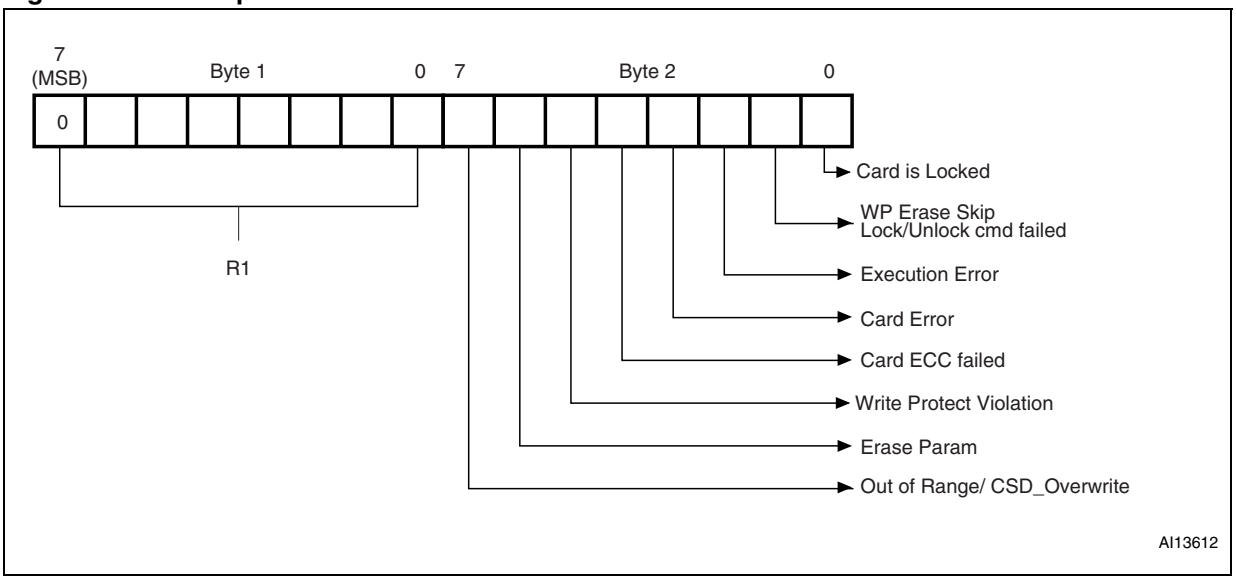


10.6.3 R2 format

This is the format of the response sent by the device to the SEND_STATUS command. R2 format responses are two bytes long. The first byte is identical to the R1 response byte. All the bits in the second byte are error bits; any one of them going High, (set to '1') indicates an error.

Refer to *Figure 37* for the structure of the R2 response format and to *Table 68* for the meaning of the bits.

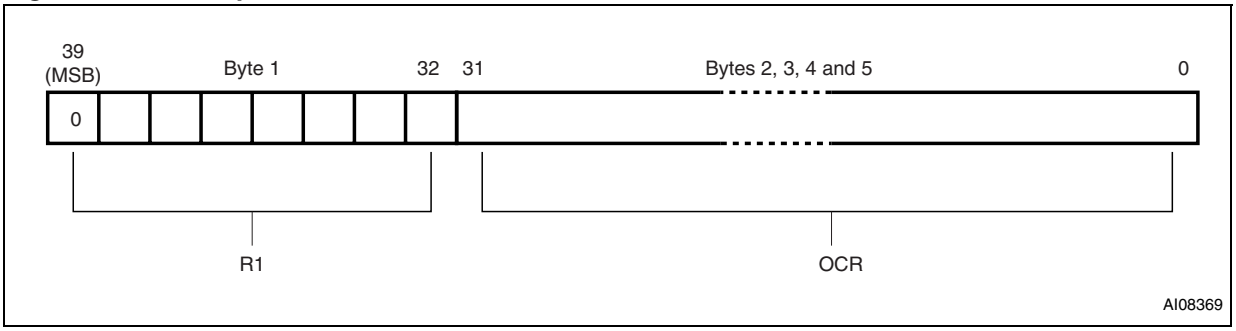
Figure 37. R2 response format



10.6.4 R3 format

This is the format of the response sent by the device to the READ_OCR command. R3 format responses are five bytes long. The first byte is identical to the R1 response byte. The other four bytes contain the OCR Register. *Figure 38* shows the structure of the R3 response format.

Figure 38. R3 response format



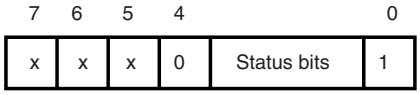
10.6.5 Data response format

This is the format of the acknowledgement sent by the device for each data block written to it. The data response is one byte long and contains three Status bits. If the Status bits in the data response are:

- '010', it means that the data is accepted
- '101', it means that a CRC error occurred and the data is rejected
- '110', it means that a Write Error occurred and the data is rejected.

If the CRC or Write error occurs during a Write Multiple Block operation, the host must abort the operation by sending a Stop tran prefix. *Figure 39* shows the structure of the Data response format.

Figure 39. Data response format



10.6.6 Data messages

Data is transferred in the form of data messages during Read and Write transactions. Data bytes are always transmitted most significant bit (MSB) first. The data messages are 4 to N+3 bytes long, where N is the data block length set using the SET_BLOCK_LENGTH command. The first byte indicates the type of transaction (see *Table 67* for details), the bytes 2 to N+1 contain user data and the last two bytes consist of the 16 CRC bits.

Table 67. Data message first byte

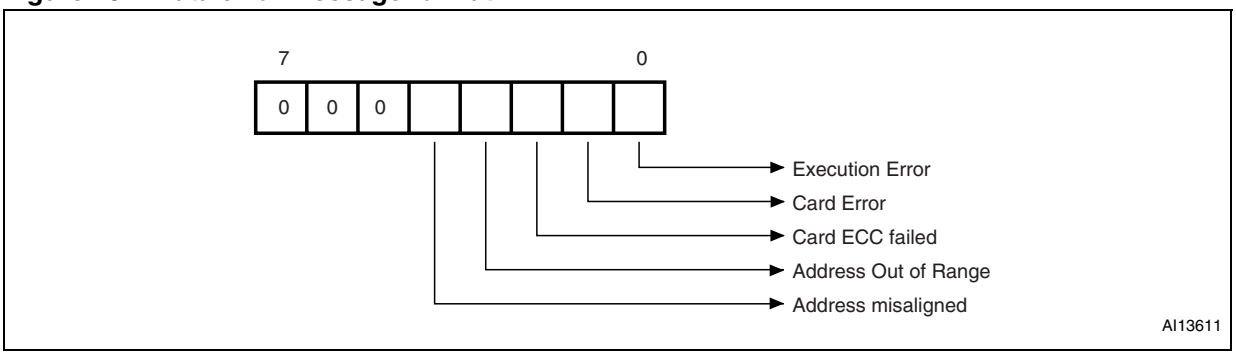
Message type	Transaction type	Bit position							
		7	6	5	4	3	2	1	0
Start Block	Single Block Read	1	1	1	1	1	1	1	0
Start Block	Multiple Block Read	1	1	1	1	1	1	1	0
Start Block	Single Block Write	1	1	1	1	1	1	1	0
Start Block	Multiple Block Write	1	1	1	1	1	1	0	0
Stop tran	Multiple Block Write	1	1	1	1	1	1	0	1

10.6.7 Data error messages

If a read operation fails, a data error message is returned to the host instead of the requested data. Data Error Messages are one byte long.

Refer to *Figure 40* for the structure of the data error message format and to *Table 68* for the meaning of the bits.

Figure 40. Data error message format



10.7 Clearing Status Register bits

In SPI mode, Error and Status bits are reported to the host in three different formats:

- Response R1
- Response R2
- Data error message

Note: The same bits may exist in multiple response types (for example Address out of range).

All Error bits defined in MultiMediaCard mode, with the exception of Underrun and Overrun, have the same meaning and usage in SPI mode. There are some differences in the Status bits due to the different protocol (for example Current state is not defined in SPI mode).

The detection mode and clear condition of Error and Status bits are identical to MultiMediaCard mode, except for Error bits which are cleared when read by the host, regardless of the response format.

Not all status bits are meaningful all the time. The relevant bits depend on the classes supported by the device. If all the classes that affect a status/error, are not supported by the device, the bit is not relevant and can be ignored by the host.

See *Table 68* for details of how the status and error bits are set and cleared in SPI mode, and *Table 69* for a description of relevant bits according to classes and commands.

Table 68. Status bits definition in SPI mode(1)

Identifier	Response	Type	Detection mode	Value	Description
Address out of range	R1 R2 DataErr	E	R	‘0’= no error ‘1’= error	Command argument not allowed
			X		A Multiple Block Read/Write operation attempted to read or write beyond the device capacity (although it started at a valid address)
Address misaligned	R1 R2 DataErr	E	R	‘0’= no error ‘1’= error	Command using misaligned address, not matching the block length
			X		Multiple block Read/Write operation attempting to read or write data block which does not align with the device memory blocks
Erase sequence error	R1 R2	E	R	‘0’= no error ‘1’= error	An error occurred in the Erase command sequence
Erase param	R2	E	X	‘0’= no error ‘1’= error	Invalid selection of erase groups during erase operation
Block length error	R1 R2	E	R	‘0’= no error ‘1’= error	Transferred block length not allowed, or number of bytes transferred not matching the block length
WP violation	R2	E	X	‘0’= not protected ‘1’= protected	Attempt to write to a write protected block
Communication CRC error	R1 R2	E	R	‘0’= no error ‘1’= error	CRC check of the previous command failed
Illegal command	R1 R2	E	R	‘0’= no error ‘1’= error	Illegal command not legal for the current state
Switch error	R1 R2	E	X	‘0’= not error ‘1’= error	If set, the device did not switch to the expected mode as requested by the SWITCH command
Card ECC failed	R2 DataErr	E	X	‘0’= success ‘1’= failure	Internal ECC performed but failed to correct the data
Card error	R2 DataErr	E	R	‘0’= no error ‘1’= error	Internal device controller error
Execution error	R2 DataErr	E	X	‘0’= no error ‘1’= error	General or unknown error occurred during operation
WP Erase skip	R2	S	X	‘0’=not protected ‘1’= protected	Partial address space erased due to existing Write protected blocks
Lock/Unlock failed	R2	E	X	‘0’= no error ‘1’= error	This bit is set when a sequence or password error has been detected during a Lock/Unlock command or if there was an attempt to access a locked device

Table 68. Status bits definition in SPI mode(1) (continued)

Identifier	Response	Type	Detection mode	Value	Description
Card locked	R2	S	-	'0' = unlocked '1' = locked	This bit is set when the device is locked by the host
Erase reset	R1 R2	E	R	'0'= cleared '1'= set	Erase sequence cleared before executing because an out of erase sequence command was received
In idle state	R1 R2	S	-	'0'= Ready '1'= Idle	The device enters idle state after power-up or reset command. It exits this state and become ready upon completion of the initialization sequence
CSD overwrite	R2	E	X	'0'= no error '1'= error	Can be one of the following errors: - The CID register has been already written and cannot be overwritten - The read only section of the CSD does not match the device content. - An attempt to reverse the copy (set as original) or permanent WP (unprotected) bits was made.

1. 'R' or 'X' mean the Error/Status bit may be affected by the respective command (using R or X detection mechanism respectively). 'S' indicates Status bits.

Table 69. Status bits versus commands and classes (SPI mode)

Comd/ classes	R2 response bits																Data Error bit							
	R1 response bits																							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
Bit is valid for classes	-	1, 2, 3, 4, 5, 6	2, 4	5	All	All	All	All	All	5	3, 4	1, 2	All	All	3, 4	All				All	All	1, 2	All	All
CMD0					R			S					R	X		S								
CMD1					R	R		S						X										
CMD6					R	R/ X		S					R	X		S								
CMD8					R	R		S					R	X		S								
CMD9					R	R	R	S					R	X		S								
CMD10					R	R	R	S					R	X		S								
CMD10					R	R	R	S					R	X		S								
CMD12					R	R		S					R	X		S								
CMD13					R	R		S					R	X		S								
CMD16					R	R	R	S					R	X		S								
CMD17		R	R		R	R	R	S				X	R	X		S				X	X	X	R	X
CMD18		R	R		R	R	R	S				X	R	X		S				X	X	X	R	X
CMD23					R	R	R	S					R	X		S								
CMD24		R	R		R	R	R	S			X		R	X		S								
CMD25		R	R		R	R	R	S			X		R	X		S								
CMD27					R	R	R	S	X				R	X		S								
CMD28		R			R	R	R	S					R	X		S								
CMD29		R			R	R	R	S					R	X		S								
CMD30		R			R	R	R	S					R	X		S								
CMD35		R		R	R	R		S		X			R	X		S								
CMD36		R		R	R	R		S		X			R	X		S								
CMD38				R	R	R		S					R	X	X	S								
CMD42					R	R	R	S					R	X	X	S								
CMD55					R	R	R	S					R	X		S								
CMD56					R	R	R	S					R	X		S								
CMD58					R	R	R	S					R	X		S								
CMD59					R	R	R	S					R	X		S								

10.8 Device registers

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

10.9 SPI bus timings

Figure 41 illustrates the basic command/response transaction in SPI mode (that is, when the device is ready). *Figure 42* describes a command/response transaction when the device is busy (R1b response format). Refer to *Table 71* for the timing values.

Table 70. SPI timing symbols

S	Start bit (= 0)
T	Transmitter bit (Host = 1, Device = 0)
P	One-cycle Pull-up (= 1)
E	End bit (=1)
Z	High Impedance state
D	Data bits
*	Repeater

Table 71. SPI timing values

Timing	Min	Max	Unit
N_{CS}	0		8 clock cycles
N_{CR}	1	8	8 clock cycles
N_{CX}	0	8	8 clock cycles
N_{RC}	1		8 clock cycles
N_{AC}	1	10/8(TAAC f_{OP} + 100NSAC)(1)	8 clock cycles
N_{WR}	1		8 clock cycles
N_{EC}	0		8 clock cycles
N_{DS}	0		8 clock cycles
N_{BR}	1	1	8 clock cycles

1. f_{OP} is the device clock frequency the host is using for the read operation.

10.9.1 Command/response timings

Figure 41. Host command to device response timing diagram (device ready)

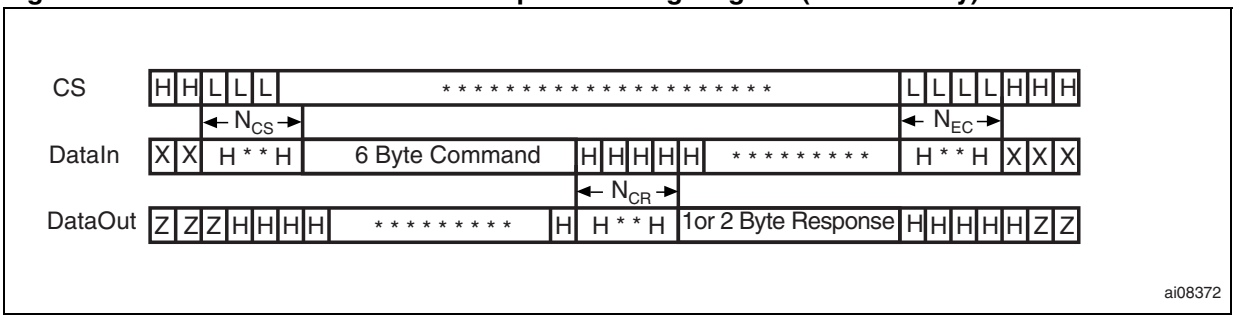


Figure 42. Host command to device response timing diagram (device busy)

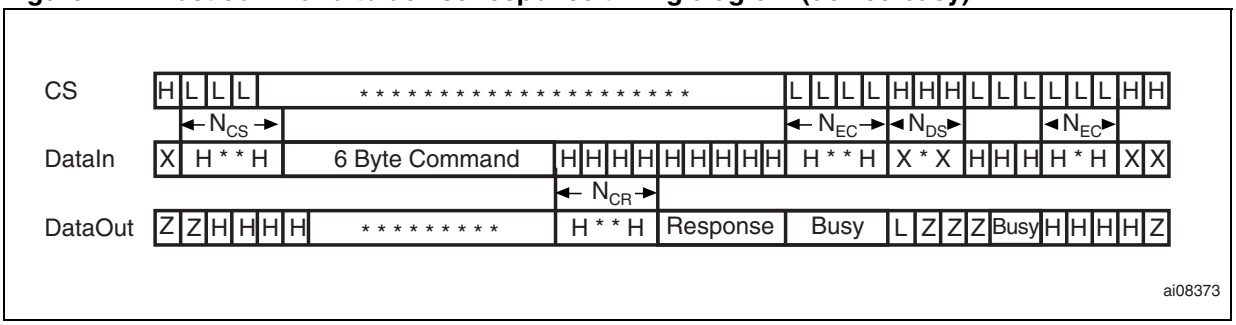
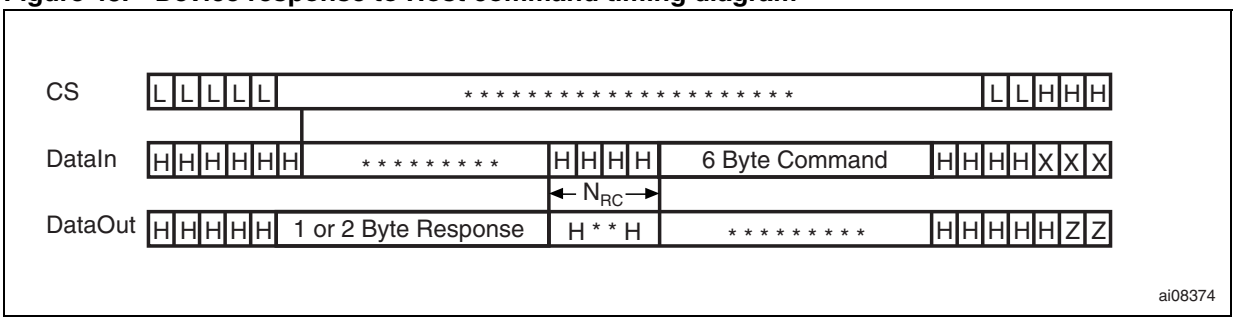


Figure 43. Device response to Host command timing diagram



10.9.2 Data Read timings

The timing diagram for deselecting the device (by de-asserting $\overline{CS}$ after the last device response) corresponds to a standard command-to-response timing diagram as illustrated in *Figure 41*.

During open-ended Multiple Block Read operations, the STOP_TRANSMISSION command may be sent while the device is transmitting data to the host. In this case, the device stops transmitting the data block within two clock cycles (the bits in the first byte may not all be set to '1') and returns the response message after a time measured in numbers of clock cycles (N_{CR}).

Refer to *Table 71* for timing values and to *Figure 44*, *Figure 45*, *Figure 46*, and *Figure 47* for a description of Data Read timing diagrams.

Figure 44. Single Block Read timing diagram

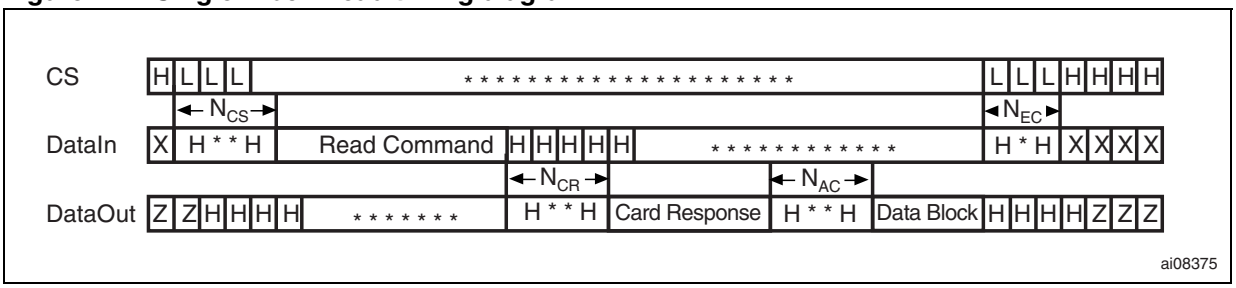


Figure 45. STOP_TRANSMISSION between blocks in Multiple Block Read timing diagram

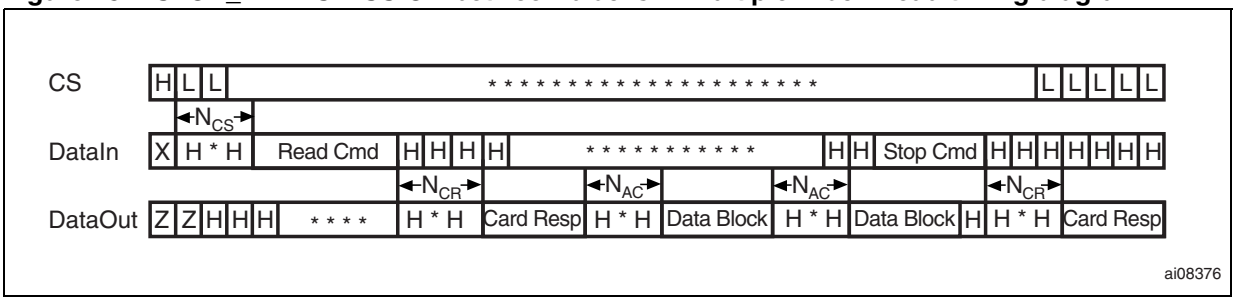


Figure 46. STOP_TRANSMISSION within a block in Multiple Block Read timing diagram

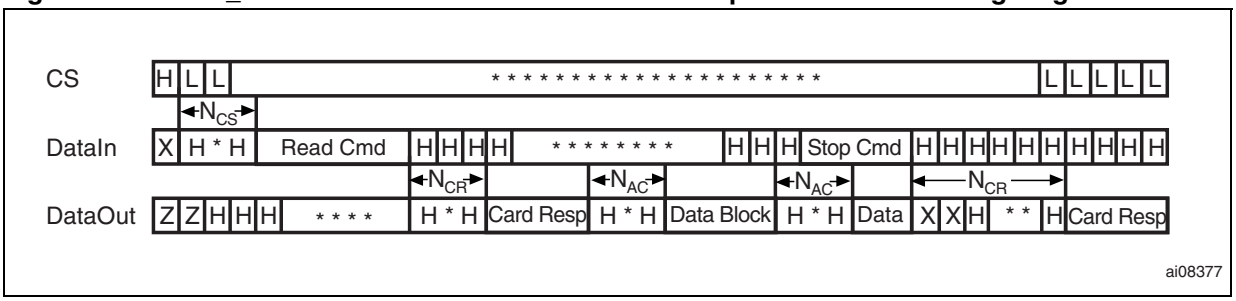
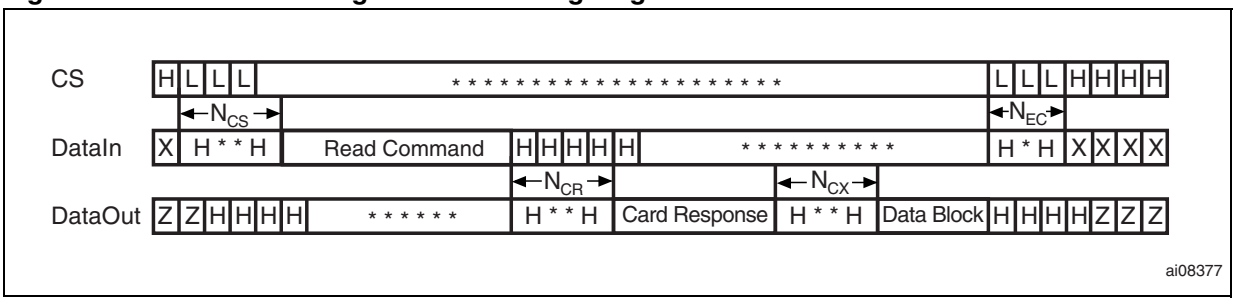


Figure 47. CSD and CID register Read timing diagram



1. The timeout between the device response and the data block is N_{CX} as N_{AC} is not known yet.

10.9.3 Data Write timings

The host may deselect a device at any moment during single and Multiple Block Write operations. The device will release the DataOut line one clock cycle after it is deselected ($\overline{CS}$ High). To check whether the device is still busy, the host must reselect it by reasserting $\overline{CS}$ Low. The device will then take control of the DataOut line one clock cycle after being reselected.

In Multiple Block Write operations, the timings from the command being issued to the first data block being transmitted by the device are the same as for single block Write operations (see *Figure 48* and *Figure 49* for details). The timing of STOP TRANSMISSION prefixes is the same as that of data blocks. After the Stop tran is received by the device, the data on the DataOut line is undefined for one byte (N_{BR}), then a Busy message may be sent by the device.

Refer to *Table 71* for timing values.

Figure 48. Single Block Write timing diagram

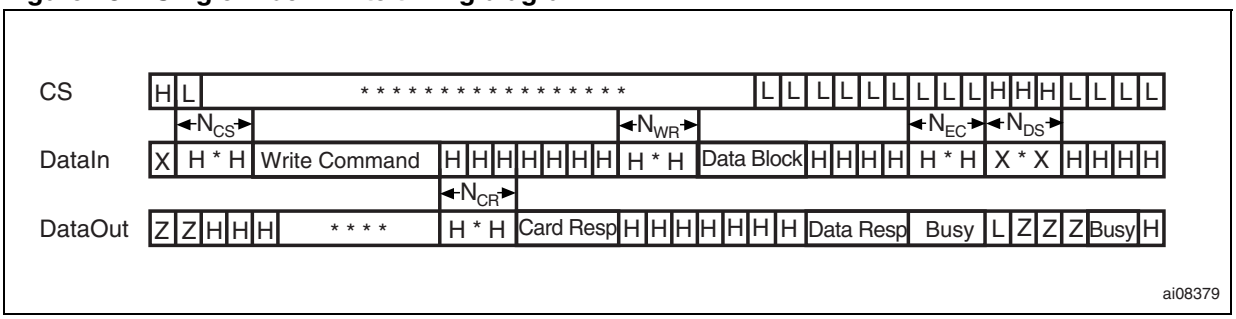
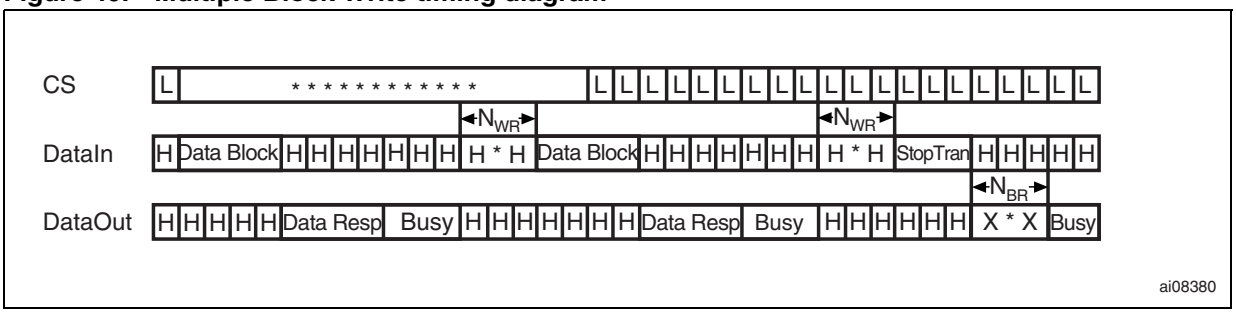


Figure 49. Multiple Block Write timing diagram



11 Error protection

All commands, responses and data transfers are protected against transmission errors by CRC (cyclic redundancy check) codes.

One CRC is generated for each command and checked for each response transferred through the CMD line. For data blocks, one CRC per transferred block and per data line is generated.

If the addressed device CRC check fails, the device does not respond, and the command is not executed.

11.1 CRC7

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

It is computed as follows:

The generator polynomial is $G(x) = x^7 + x^3 + 1$

$M(x) = \text{firstbit} \times x^n + \text{secondbit} \times x^{n-1} + ... + \text{lastbit} \times x^0$

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

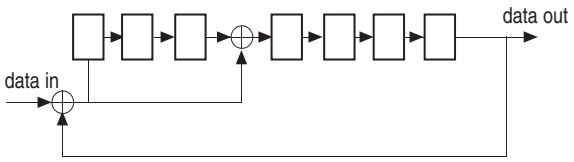
All CRC registers are initialized to zero.

The first bit is the outmost left bit of the corresponding bit string of the command, response, CID or CSD).

The degree *n* of the polynomial is the number of CRC protected bits decreased by one.

The number of bits to be protected is 40 for commands and responses (n = 39), and 120 for the CSD and CID (n = 119).

Figure 50. CRC7 generator/checker



11.2 CRC16

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

It is computed as follows:

The generator polynomial is

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

$$M(x) = firstbit \times x^n + secondbit \times x^{n-1} + ... + lastbit \times x^0$$

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

All CRC registers are initialized to zero.

The first bit is the first data bit of the corresponding block.

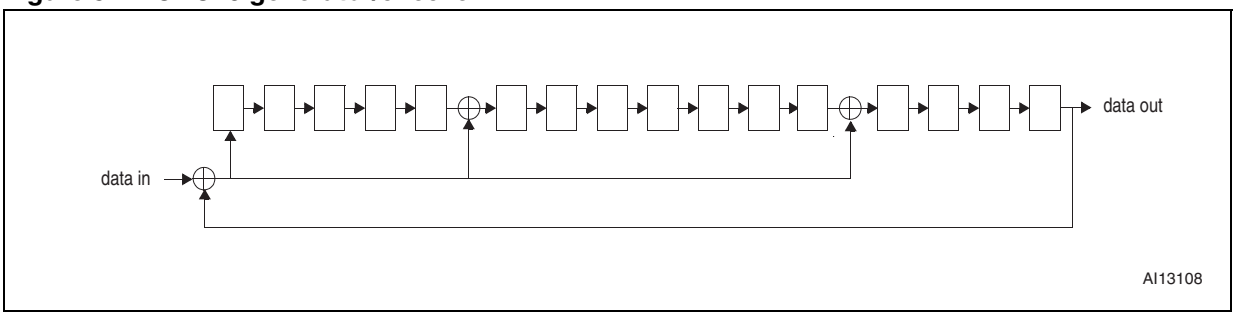
The degree n of the polynomial is the number of bits of the data block decreased by one (e.g. n = 4095 for a block length of 512 bytes).

The generator polynomial is a standard CCITT polynomial.

The code has a minimal distance d set to 4 and is used for a payload length of up to 2048 bytes (n <= 16383).

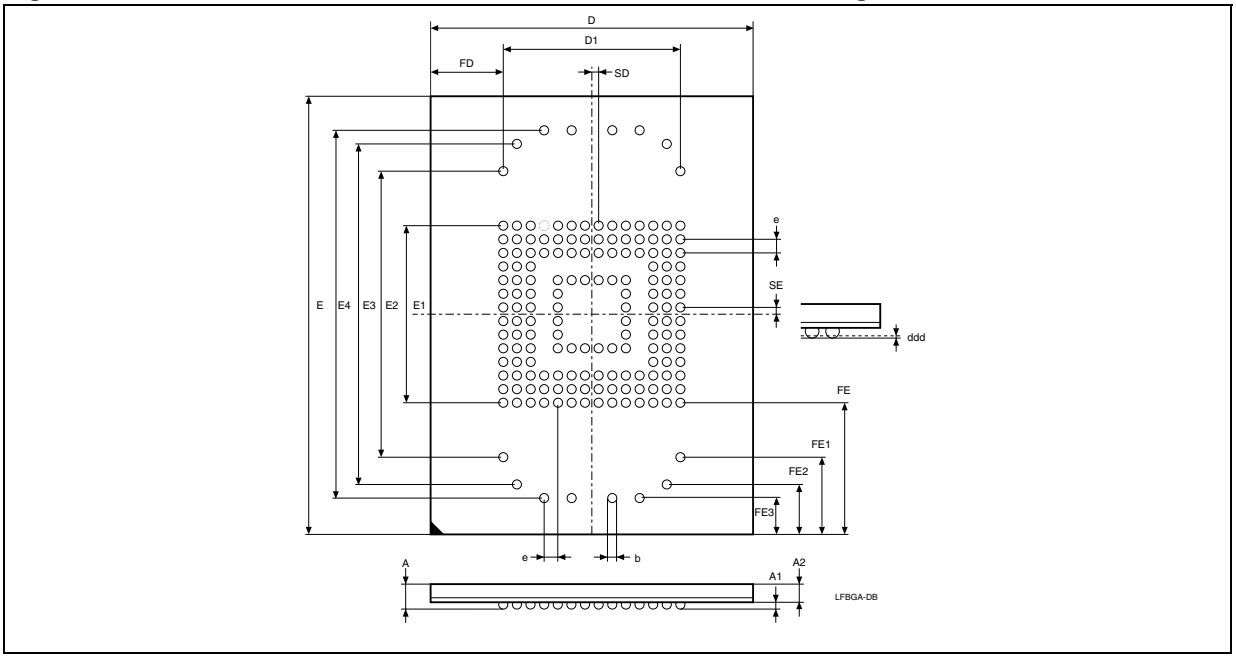
The same CRC16 calculation is used for all bus width. In 4-bit and 8-bit bus width, the CRC16 is calculated for each line separately. Sending the CRC is synchronized so the CRC code is transferred at the same time in all lines.

Figure 51. CRC16 generator/checker



12 Package mechanical

Figure 52. LFBGA169 12 x 16 x 1.4 mm 132+21+16 3R14 0.50 mm, package outline



1. Drawing is not to scale.

Table 72. LFBGA169 12 x 16 x 1.4 mm 132+21+16 3R14 0.50 mm, package mechanical data

Symbol	millimeters			inches		
	Typ	Min	Max	Typ	Min	Max
A			1.40			0.055
A1		0.15			0.006	
A2	1.00			0.039		
b	0.30	0.25	0.35	0.012	0.010	0.014
D	12.00	11.90	12.10	0.472	0.469	0.476
D1	6.50			0.256		
ddd			0.08			0.003
E	16.00	15.90	16.10	0.630	0.626	0.634
E1	6.50			0.256		
E2	10.50			0.413		
E3	12.50			0.492		
E4	13.50			0.531		
e	0.50	–	–	0.020	–	–
FD	2.75			0.108		
FD1	3.25			0.128		
FD2	4.25			0.167		
FD3	5.25			0.207		

Table 72. LFBGA169 12 x 16 x 1.4 mm 132+21+16 3R14 0.50 mm, package mechanical data (continued)

Symbol	millimeters			inches		
	Typ	Min	Max	Typ	Min	Max
FE	4.75			0.187		
FE1	2.75			0.108		
FE2	1.75			0.069		
FE3	1.25			0.049		
SD	0.25	–	–	0.010	–	–
SE	0.25	–	–	0.010	–	–

13 Part numbering

Table 73. Ordering information scheme

Example: NAND08GAH0AZA5F

Device Type

NAND Flash memory

Density

08G = 8 Gbits (1Gbyte)

16G = 16 Gbits (2 Gbytes)

Operating voltage

A = V_{CC}= 3 V, V_{CCQ} = 1.8 V or 3 V

Memory type

H = eMMC

Device options

0 = No option

Device composition

A = Version A (1 Gbyte)

D = Version D (2 Gbytes)

Package

ZA = LFBGA169 12 x 16 x 1.4 mm

Temperature range

5 = –25 to 85 °C

Packing

E = ECOPACK package, standard packing

F = ECOPACK package, tape & reel packing

Note: Other digits may be added to the ordering code for preprogrammed parts or other options.

Devices are shipped from the factory with the memory content bits erased to ’1’. For further information on any aspect of the device, please contact your nearest Numonyx Sales Office.

14 Revision history

Table 74. Document revision history

Date	Revision	Changes
07-Dec-2006	0.1	Initial release.
20-Aug-2007	1	Document status promoted from Target Specification to Preliminary Data. Change of names from NAND08GAH to NAND08GAH0A, and from NAND16GAH to NAND16GAH0D. Removed the 4 Gbyte density, the LFBGA169 (ZB) and LFBGA153 (ZC) packages. Section 8.4.2: SEC_COUNT, section 5.2.3: Access mode validation, and section 8.4.13: ERASED_MEM_CONT removed. *Section 8.1: Operation conditions register (OCR)*, *Table 1: System performance*, *Table 2: Current consumption*, and *Table 3: System reliability and maintenance* updated. Removed the Erased Memory Content from *Table 49: Extended CSD*. Removed note 1 below *Table 61: Timing values*. Small text changes.
10-Dec-2007	2	Applied Numonyx branding.

Please Read Carefully: