The MultiMediaCard

Based On System Specification Version 4.1

System Summary

MMCA Technical Committee

Table Of Contents

1 General Description

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

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

To provide for the forecasted migration of CMOS power (V_{DD}) requirements and for compatibility and integrity of MultiMediaCard systems, two types of MultiMediaCards are defined in this standard specification, which differ only in the valid range of system V_{DD}. These two card types are referred to as High Voltage MultiMediaCard and Dual Voltage MultiMediaCard.

The purpose of the system specification is the definition of the MultiMediaCard, its environment and handling. It gives guidelines for a system designer. The system specification also defines a tool box (a set of macro functions and algorithms) which contributes to reducing the design-in costs.

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

2 System Features

The MultiMediaCard System has a wide variety of system features, whose comprehensive elements serves several purposes, which include:

- Covering a broad category of applications from smart phones and PDAs to digital recorders and toys
- Facilitating the work of designers wno seek to develop applications with their own advanced and enhanced features
- Maintainng compatibility and compliance with current electronic, communication, data and error handling standards.

The following list identifies the main features of the MultiMediaCard System, which:

- **Is targeted for portable and stationary applications**
- **Has these System Voltage (V_{DD}) Ranges:**

	High Voltage MultiMediaCard	Dual Voltage MultiMediaCard
Communication	**2.7 - 3.6**	**1.65 - 1.95, 2.7 - 3.6**[1]
Memory Access	**2.7 - 3.6**	**1.65 - 1.95, 2.7 - 3.6**

1)V_{DD} range: 1.95V - 2.7V is not supported.

- **Includes MMCplus and MMCmobile definitions**
- **Is designed for read-only, read/write and I/O cards**
- **Supports card clock frequencies of 0-20MHz, 0-26MHz or 0-52MHz**
- **Has a maximum data rate up to 416Mbits/sec.**
- **Has a defined minimum performance**
- **Maintains card support for three different data bus width modes: 1bit (default), 4bit and 8bit**
- **Includes password protection of data**
- **Supports basic file formats for high data interchangeability**
- **Includes application specific commands**
- **Enables correction of memory field errors**
- **Has built-in write protection features, which may be permanent or temporary**
- **Includes a simple erase mechanism**
- **Maintains full backward compatibility with previous MultiMediaCard systems (1 bit data bus, multi-card systems)**
- **Ensures that new hosts retain full compatibility with previous versions of MultiMediaCards (backward compatibility).**
- **Supports two form factors: Normal size(24mm x 32mm x 1.4mm) and Reduced size (24mm**

x 18mm x 1.4mm)

- **Supports multiple command sets**
- **Includes attributes of the available operation modes:**

MultiMediaCard Mode	**SPI Mode**
Ten-wire bus (clock, 1 bit command, 8 bit data bus)	Three-wire serial data bus (clock, dataIn, dataOut) + card specific CS signal.
Card selection is done through an assigned unique card address to maintain backwards compatibility to prior versions of the specification	Card selection via a hardware CS signal
One card per MultiMediaCard bus	Card requires a dedicated CS signal.
Easy identification and assignment of session address	Not available. Card selection via a hardware CS signal
Error-protected data transfer	Optional. A non-protected data transfer mode is available.
Sequential and Single/Multiple block Read/Write commands	Single/Multiple block Read/Write commands

Table 1: MMC System Operational Modes

3 MultiMediaCard System Concept

The main design goal of the MultiMediaCard system is to provide a very low cost mass storage product, implemented as a ‘card’ with a simple controlling unit, and a compact, easy-to-implement interface. These requirements lead to a reduction of the functionality of each card to an absolute minimum.

Nevertheless, since the complete MultiMediaCard system has to have the functionality to execute tasks (at least for the high end applications), such as error correction and standard bus connectivity, the system concept is described next. It is based on modularity and the capability of reusing hardware over a large variety of cards.

Figure 1 shows four typical architectures of possible MultiMediaCard systems.

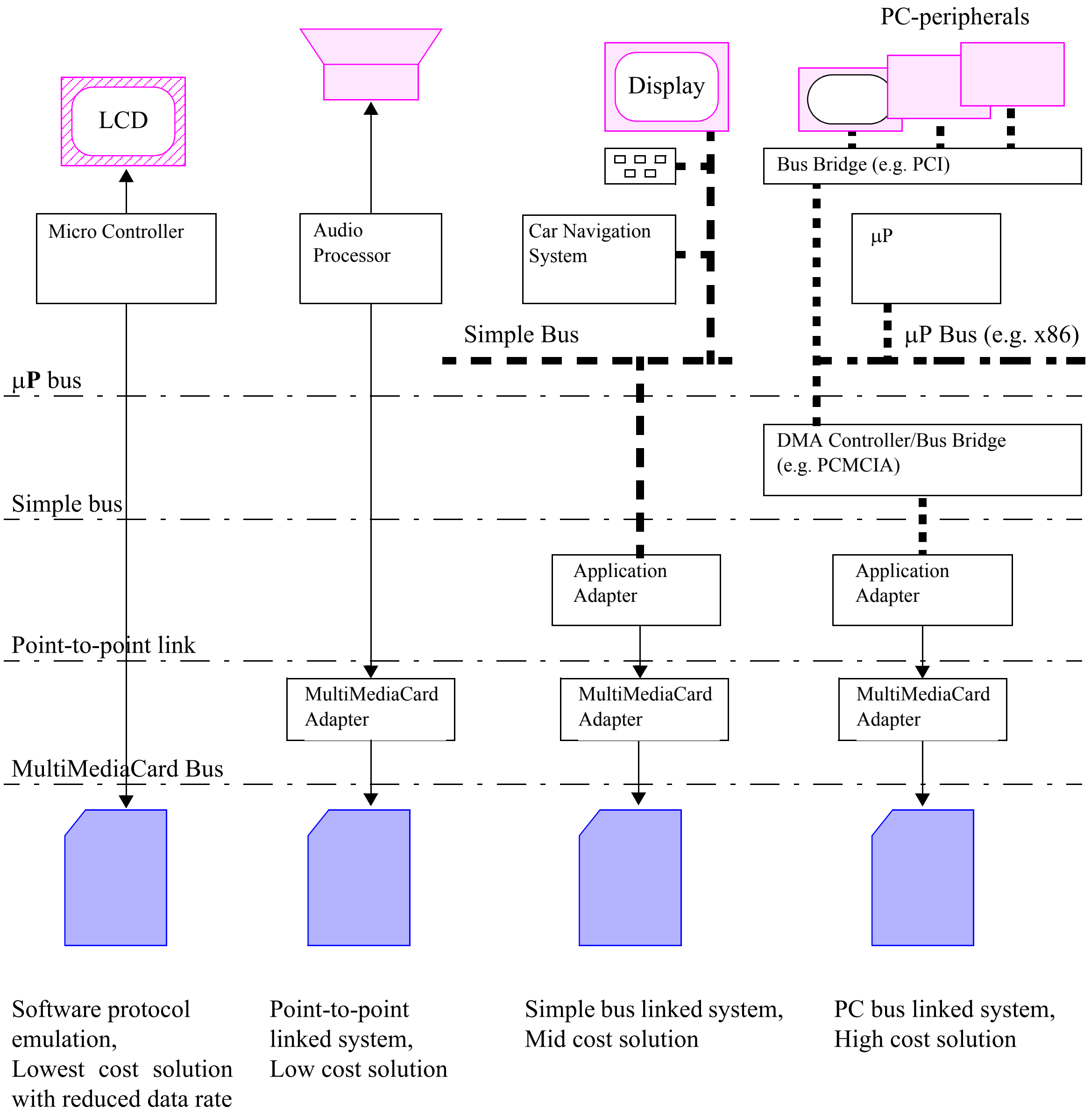

Figure 1: Topology Of MultiMediaCard Systems

Four typical types of MultiMediaCard systems can be derived from the diagram shown in Figure 2. The typical systems include:

- Software emulation: reduced data rate, typically 100-300 kbit per second, restricted by the host
- Point to point linkage: full data rate (with additional hardware)
- Simple bus: full data rate, part of a set of addressable units
- PC bus: full data rate, addressable, extended functionality, such as DMA capabilities

In the first variant, the MultiMediaCard bus protocol is emulated in software using up to ten port pins of a microcontroller. This solution requires no additional hardware and is the cheapest system in the list. The other applications extend the features and requirements, step by step, towards a sophisticated PC solution. The various systems, although different in their feature set, have a basic common functionality, as can be seen in Figure 2. This diagram shows a system partitioned into hierarchical layers of abstract ('virtual') components. It describes a logical classification of functions which cover a wide variety of implementations. (See also Figure 1 on page 7.) It does not imply any specific design nor specify rules for implementing parts in hardware or software.

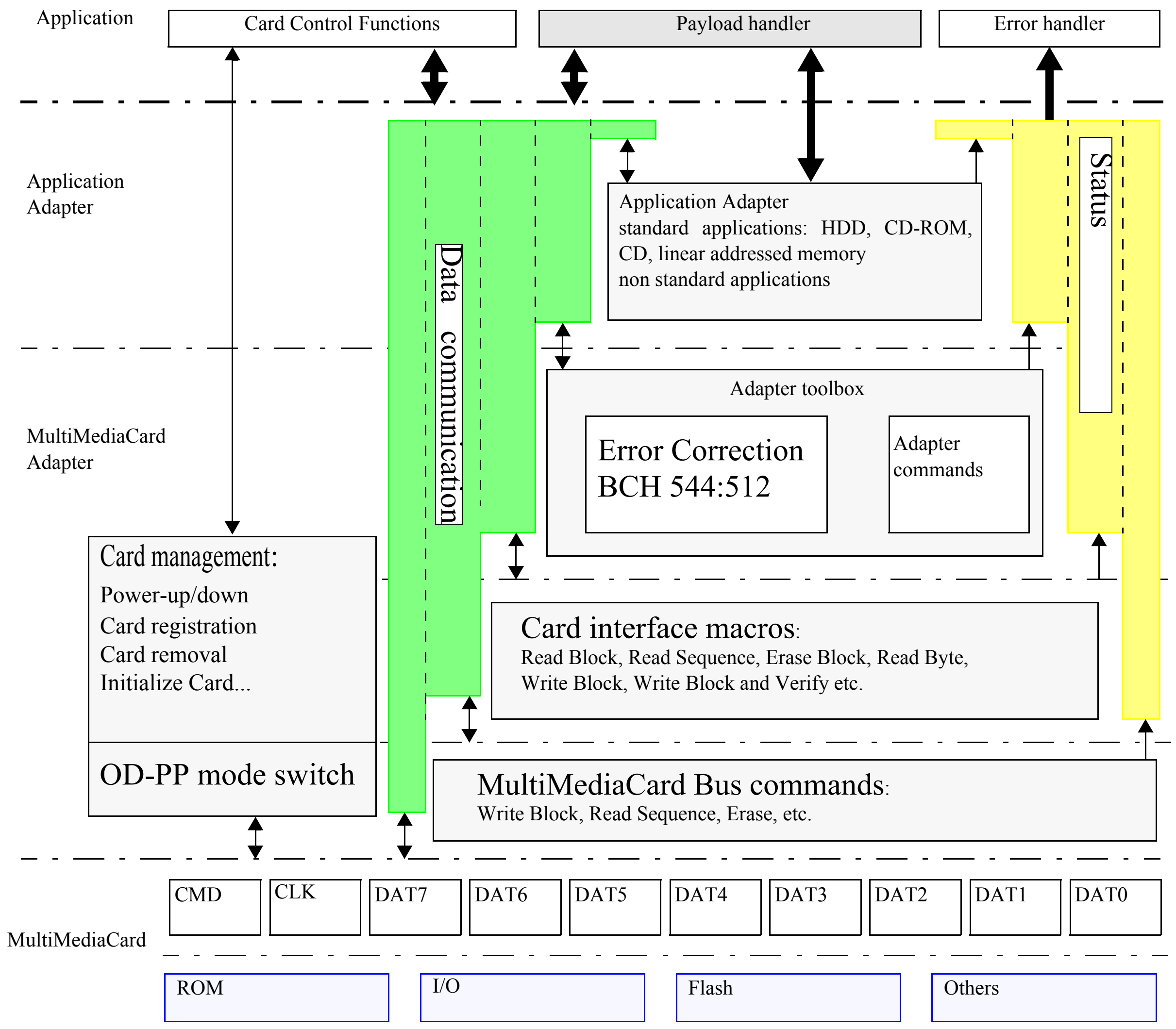

Figure 2: MultiMediaCard System Overview

Figure 3 is a specific design example based on the abstract layer model described in Figure 2 on page 8.

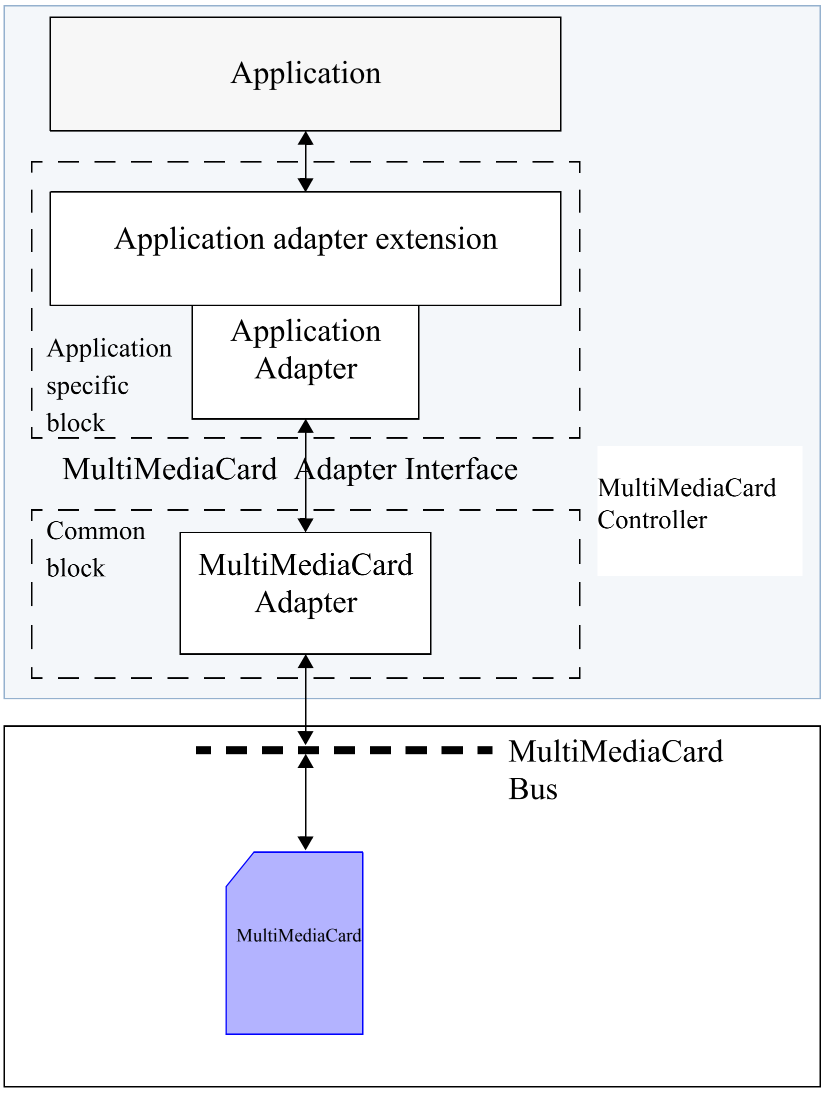
Application
Application adapter extension
Application specific block
Application Adapter
MultiMediaCard Adapter Interface
MultiMediaCard Controller
Common block
MultiMediaCard Adapter
MultiMediaCard Bus
MultiMediaCard

Figure 3: MultiMediaCard System Example

This MultiMediaCard system contains at least two components:

- The MultiMediaCard
- The MultiMediaCard controller

The MultiMediaCard controller is divided into two major blocks. In some implementations like the example shown in Figure 3, the controller may implement the whole application, while in others it may be divided into several physical components which, apart from the application itself, can be identified as:

1. Application adapter — the application specific block, for example, a microprocessor or an adapter to a standard bus like USB or ATA

    - Performs application oriented tasks, e.g., display controlling or input decoding for hand-held applications

- Typically connected as a bus slave for a standard bus

2. MultiMediaCard adapter — the common block

- Contains all card specific functions, such as initialization and error correction
- Serves as a bus master for the MultiMediaCard bus
- Implements the standard interface to the card.

3.1 Card Concept

The MultiMediaCard transfers data via a configurable number of data bus signals. The communication signals are:

- **CLK**: Each cycle of this signal directs a one bit transfer on the command and on all the data lines. The frequency may vary between zero and the maximum clock frequency.
- **CMD**: This signal is a bidirectional command channel used for card initialization and transfer of commands. The CMD signal has two operation modes: open-drain for initialization mode, and push-pull for fast command transfer. Commands are sent from the MultiMediaCard bus master to the card and responses are sent from the card to the host.
- **DAT0-DAT7**: These are bidirectional data channels. The DAT signals operate in push-pull mode. The MultiMediaCard includes internal pull ups for all data lines. Only the card or the host is driving these signals at a time. By default, after power up or reset, only DAT0 is used for data transfer. A wider data bus can be configured for data transfer, using either DAT0-DAT3 or DAT0-DAT7, by the MultiMediaCard controller.

MultiMediaCards can be grouped into several card classes which differ in the functions they provide (given by the subset of MultiMediaCard system commands):

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

The card is connected directly to the signals of the MultiMediaCard bus. The following table defines the card contacts:

Pin No.	Name[1]	Type[2]	Description
1	DAT3	I/O/PP	Data
2	CMD	I/O/PP/OD	Command/Response
3	V_{SS1}	S	Supply voltage ground
4	V_{DD}	S	Supply voltage
5	CLK	I	Clock
6	V_{SS2}	S	Supply voltage ground
7	DAT0[3]	I/O/PP	Data
8	DAT1	I/O/PP	Data
9	DAT2	I/O/PP	Data

Table 2: MultiMediaCard Interface Pin Configuration

Pin No.	Name[1]	Type[2]	Description
10	DAT4	I/O/PP	Data
11	DAT5	I/O/PP	Data
12	DAT6	I/O/PP	Data
13	DAT7	I/O/PP	Data

Table 2: MultiMediaCard Interface Pin Configuration

1)See Table 4 on page 26, for a combined table including SPI pin definitions
2)S: power supply; I: input; O: output; PP: push-pull; OD: open-drain; NC: Not connected (or logical high)
3)The DAT0-DAT7 lines for read-only cards are output only

The card initialization uses only the CMD channel and is, therefore, compatible for all cards.

Each card has a set of information registers (see also Chapter 5):

Name	Width (bytes)	Description	Implementation
CID	16	Card IDentification number, a card individual number for identification.	Mandatory
RCA	2	Relative Card Address, is the card system address, dynamically assigned by the host during initialization.	Mandatory
DSR	2	Driver Stage Register, to configure the card's output drivers.	Optional
CSD	16	Card Specific Data, information about the card operation conditions.	Mandatory
OCR	4	Operation Conditions Register. Used by a special broadcast command to identify the voltage type of the card.	Mandatory
EXT_CSD	512	Extended Card Specific Data. Contains information about the card capabilities and selected modes. Introduced in specification v4.0	Mandatory

Table 3: MultiMediaCard Registers

The host may reset the card by switching the power supply off and back on. The card shall have its own power-on detection circuitry which puts the card into a defined state after the power-on. No explicit reset signal is necessary. The card can also be reset by a special command.

3.1.1 MMC Plus and MMC Mobile

The specification further defines two card types, MMCplus and MMCmobile, to describe R/W or ROM cards with specifically defined mandatory features and attributes. Only cards meeting MMCplus or MMCmobile requirements are eligible to carry the MMCplus or MMCmobile name and logo.

- MMCplus is defined as normal size R/W or ROM cards that supports 2.7-3.6V operation, x1/x4/x8 bus widths, minimum of 2.4MB/s read/write performance and 26MHz (52MHz optional)
- MMCmobile is defined as reduced size R/W or ROM card that supports 1.65-1.95V and 2.7-3.6V operations, x1/x4/x8 bus widths, minimum of 2.4MB/s read/write performance and 26MHz (52MHz optional)

Both implementations are backwards compatible with MMCA System Specification versions 3.xx in max 20MHz clock frequency mode.

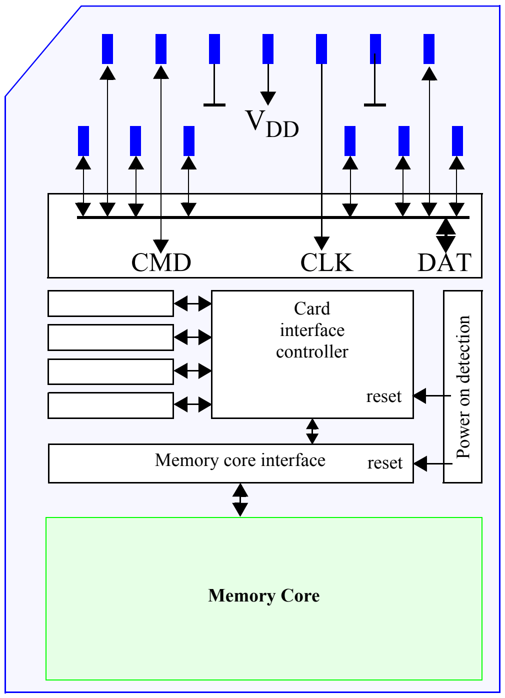

Figure 4: MultiMediaCard Architecture

3.1.2 Form Factors

The MultiMediaCard has two possible form factors. The normal size form factor is 24mm x 32mm x 1.4mm. The reduced size form factor is 24mm x 18mm x 1.4mm. To use a reduced size MMC in a normal size MMC socket, a special adaptor has to be used. Figure 5 shows the two form factors. The mechanical and electrical interface is identical in both form factors.

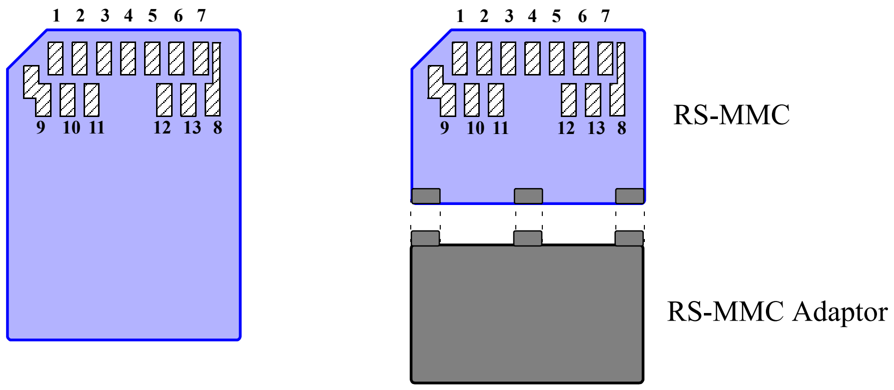

Figure 5: MultiMediaCard Form Factors (Bottom View)

3.2 Bus Concept

The MultiMediaCard bus is designed to connect either solid-state mass-storage memory or I/O-devices in a card format to multimedia applications. The bus implementation allows the coverage of application fields from low-cost systems to systems with a fast data transfer rate. It is a single master bus with a single slave. The MultiMediaCard bus master is the bus controller and the slave is either a single mass storage card (with possibly different technologies such as ROM, OTP, Flash etc.) or an I/O-card with its own controlling unit (on card) to perform the data transfer.

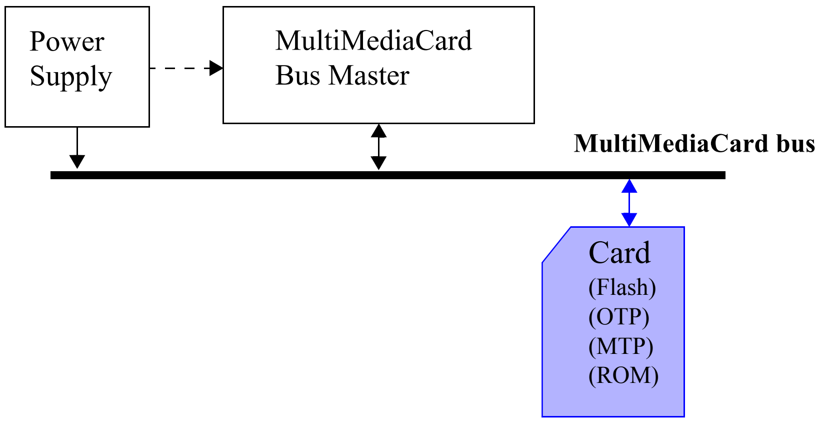

Figure 6: MultiMediaCard Bus System

The MultiMediaCard bus also includes power connections to supply the cards.

The bus communication uses a special protocol (MultiMediaCard bus protocol). The payload data transfer between the host and the card can be bidirectional.

3.2.1 Bus Lines

The bus lines can be divided into three groups:

- Power supply: V_{SS1} and V_{SS2},V_{DD} - used to supply the cards.
- Data transfer: CMD, DAT0-DAT7 - used for bidirectional communication.
- Clock: CLK - used to synchronize data transfer across the bus.

The bus line definitions and the corresponding pad numbers are described in Section 3.1.

3.2.2 Bus Protocol

After a power-on reset, the host must initialize the card by a special message-based MultiMediaCard bus protocol. Each message is represented by one of the following tokens:

- command: a command is a token which starts an operation. A command is sent from the host to a card. A command is transferred serially on the CMD line.
- response: a response is a token which is sent from the card to the host as an answer to a previously received command. A response is transferred serially on the CMD line.
- data: data can be transferred from the card to the host or vice versa. Data is transferred via the data lines. The number of data lines used for the data transfer can be 1(DAT0), 4(DAT0-DAT3) or 8(DAT0-DAT7).

Card addressing is implemented using a session address, assigned during the initialization phase, by the bus controller to the connected card. A card is identified by its CID number. This method requires the card to have an unique CID number. To ensure uniqueness of CIDs the CID register contains 24 bits (MID and OID fields - see Chapter 5) which are defined by the MMCA. Every card manufacturer is required to apply for an unique MID (and optionally OID) number.

The structure of commands, responses and data blocks is described in Chapter 4.

MultiMediaCard bus data transfers are composed of these tokens. One data transfer is a *bus operation*. There are different types of operations. Addressed operations always contain a command and a response token. In addition, some operations have a data token, the others transfer their information directly within the command or response structure. In this case no data token is present in an operation. The bits on the DAT0-DAT7 and CMD lines are transferred synchronous to the host clock.

Two types of data transfer commands are defined:

- Sequential commands[1]: These commands initiate a continuous data stream, they are terminated only when a stop command follows on the CMD line. This mode reduces the command overhead to an absolute minimum.
- Block-oriented commands: These commands send a data block succeeded by CRC bits. Both read and write operations allow either single or multiple block transmission. A multiple block transmission is terminated when a stop command follows on the CMD line similarly to the sequential read.

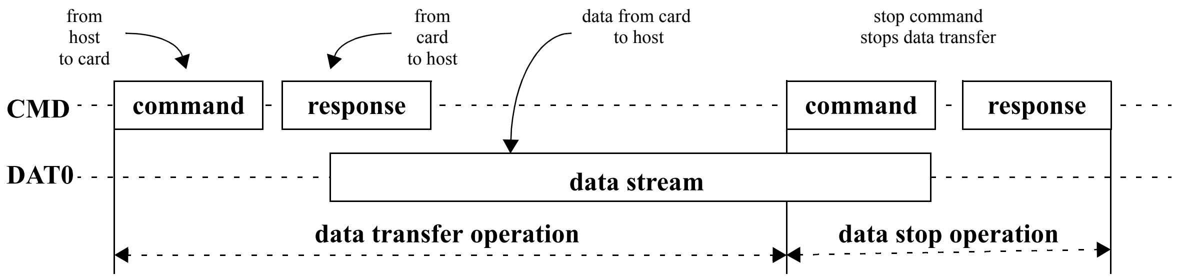

Figure 7: Sequential Read Operation

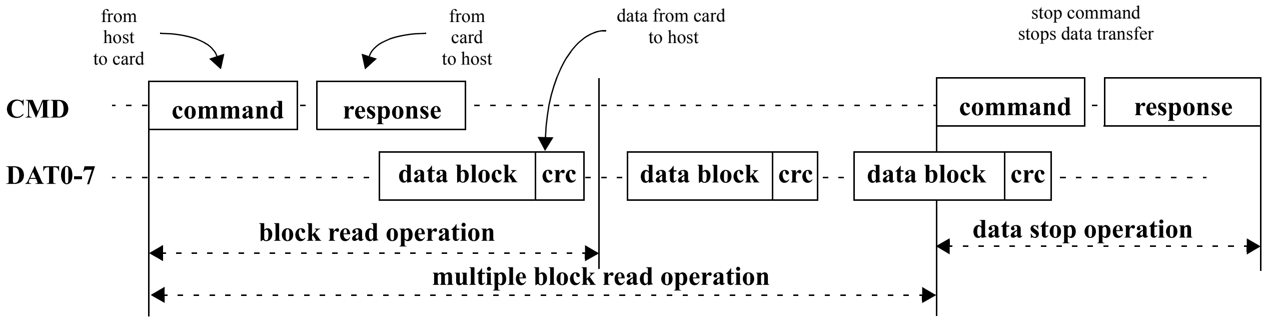

Figure 8: (Multiple) Block Read Operation

1. Sequential commands are supported only in 1 bit bus mode, to maintain compatibility with previous versions of this specification

.

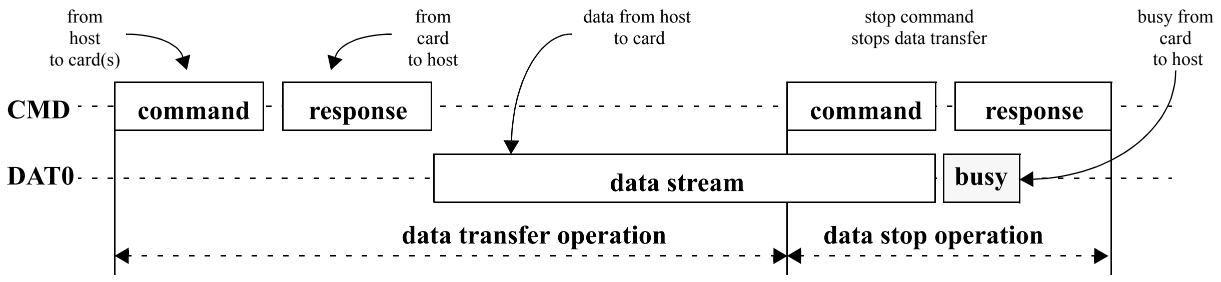

Figure 9: Sequential Write Operation

The block write operation uses a simple busy signalling of the write operation duration on the data (DAT0) line. (See Figure 10.)

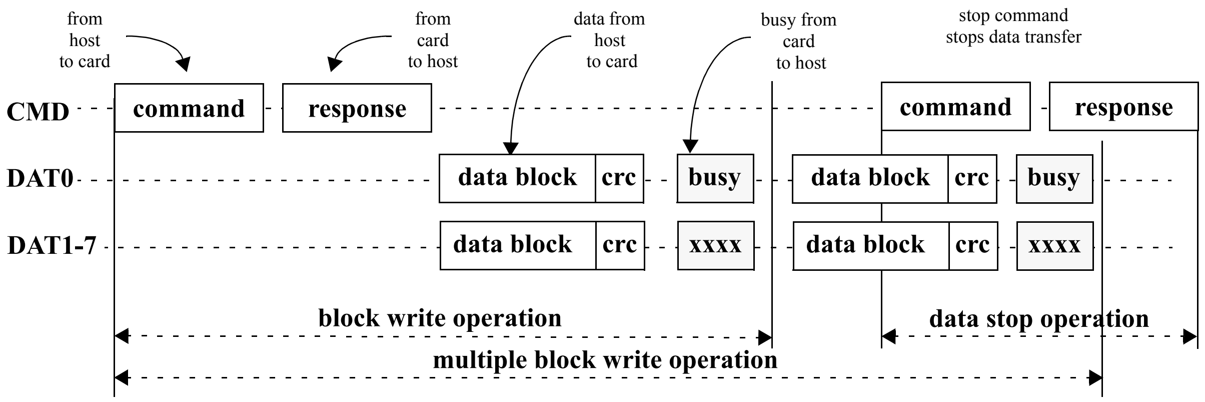

Figure 10: (Multiple) Block Write Operation

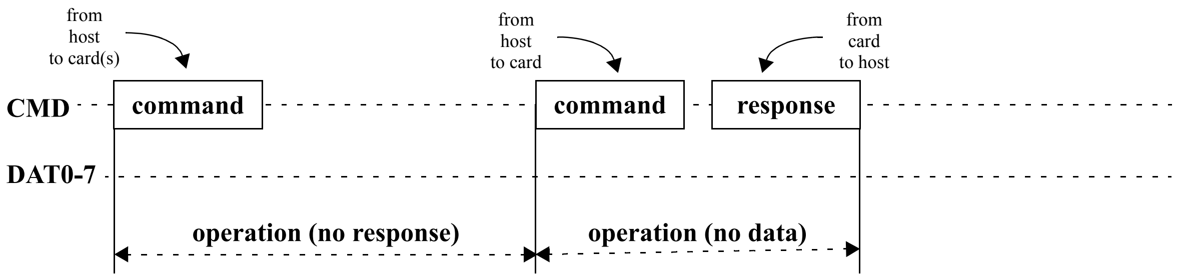

Figure 11: "no response" And "no data" Operations

Command tokens have the following coding scheme:

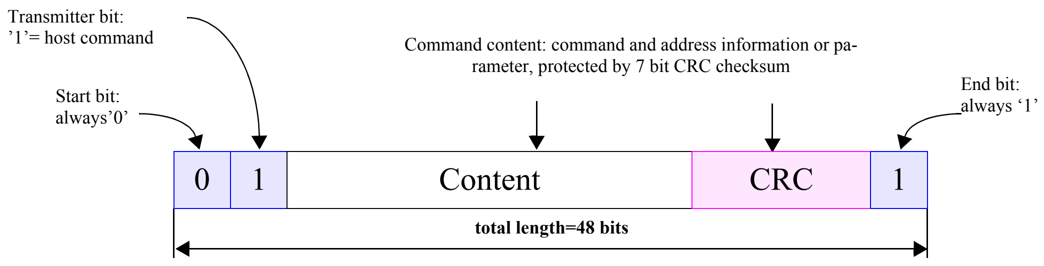

Figure 12: Command Token Format

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

Response tokens have five coding schemes depending on their content. The token length is either 48 or 136 bits. The detailed commands and response definition is given in Section 4.7 and Section 4.9.

Due to the fact that there is no predefined end in sequential data transfer, no CRC protection is included in this case. The CRC protection algorithm for block data is a 16 bit CCITT polynomial. All used CRC types are described in Chapter 7.

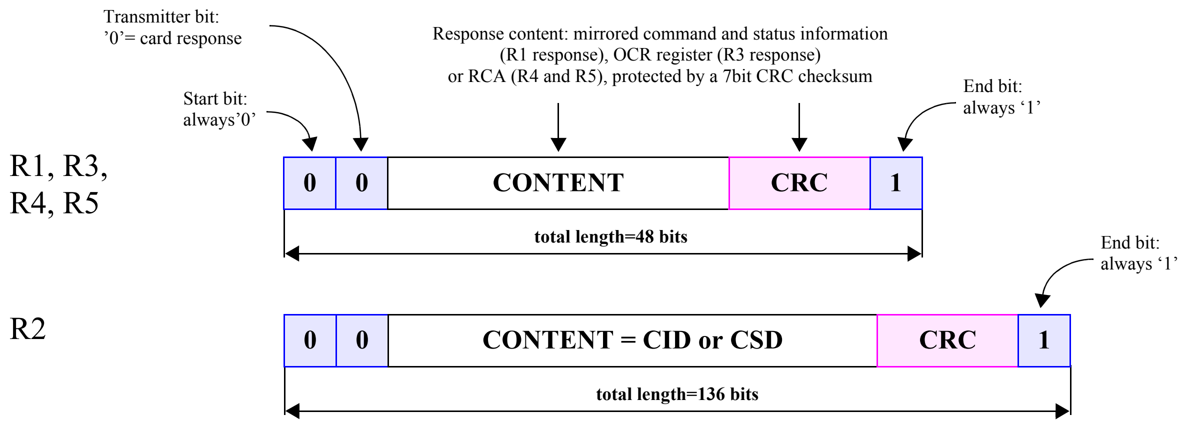

Figure 13: Response Token Format

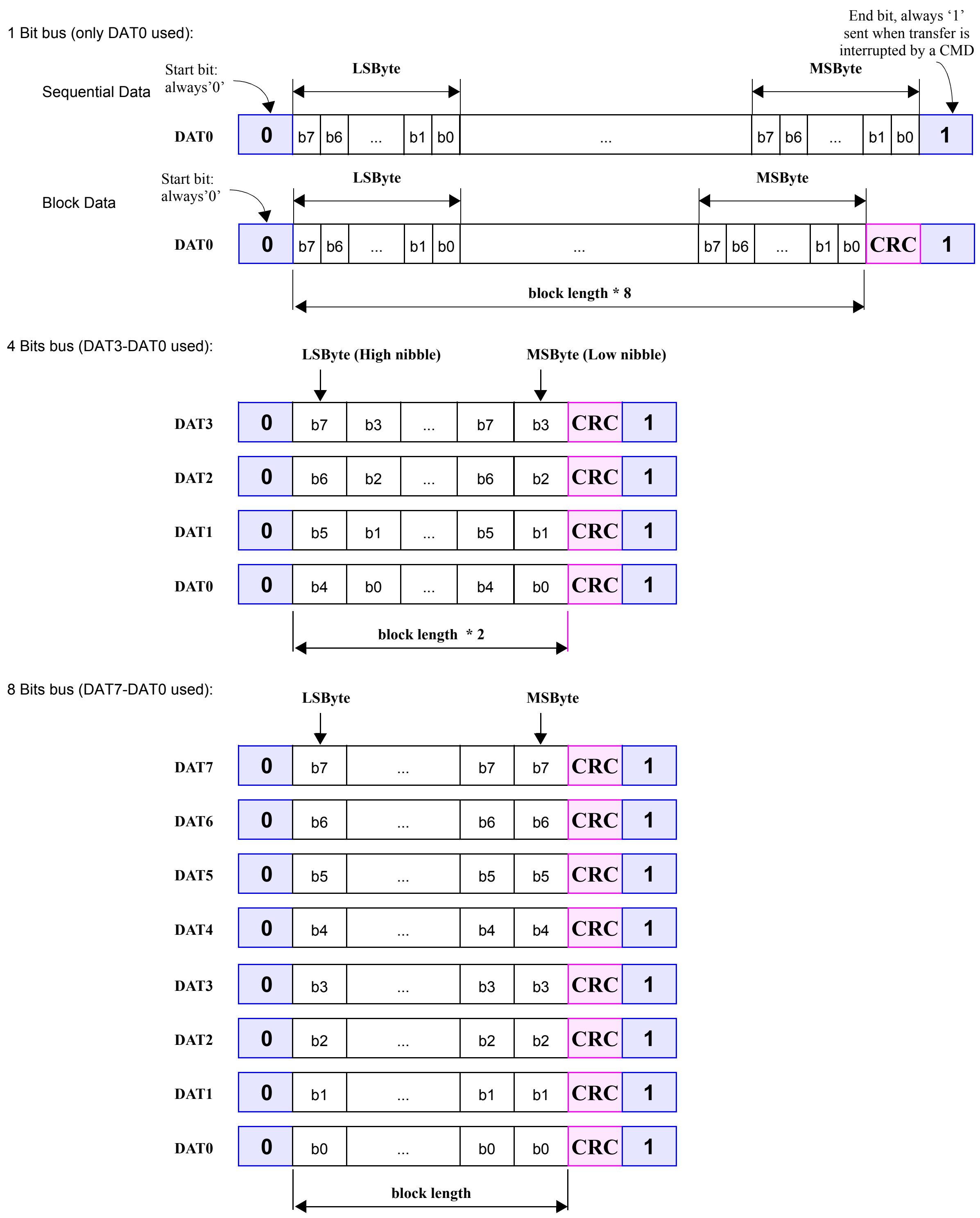

Figure 14: Data Packet Format

3.3 Controller Concept

The MultiMediaCard is defined as a low cost mass storage product. The shared functions have to be implemented in the MultiMediaCard system. The unit which contains these functions is called the MultiMediaCard controller. The following points are basic requirements for the controller:

- Protocol translation from standard MultiMediaCard bus to application bus
- Data buffering to enable minimal data access latency
- Macros for common complex command sequences

The MultiMediaCard controller is the link between the application and the MultiMediaCard bus with its card. It translates the protocol of the standard MultiMediaCard bus to the application bus. It is divided into two major parts:

- The application adapter: the application oriented part
- The MultiMediaCard adapter: the MultiMediaCard oriented part

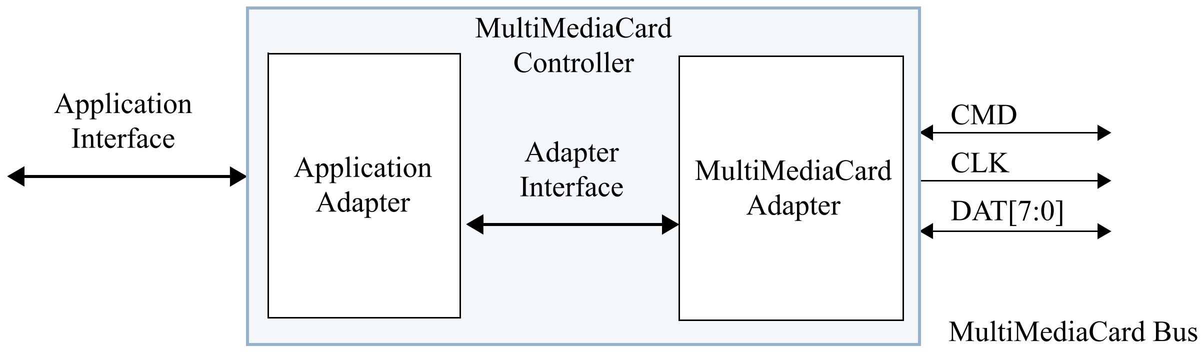

Figure 15: MultiMediaCard Controller Scheme

The application adapter consists at least of a bus slave and a bridge into the MultiMediaCard system. It can be extended to become a master on the application bus and support functions like DMA or serve application specific needs. Higher integration will combine the MultiMediaCard controller with the application.

Independently of the type and requirements of the application the MultiMediaCard bus requires a host. This host may be the MultiMediaCard adapter. On the MultiMediaCard bus side it is the only bus master and controls all activity on that bus. On the other side, it is a slave to the application adapter or to the application, respectively. No application specific functions shall be supported here, except for those that are common to most MultiMediaCard systems. It supports all MultiMediaCard bus commands and provides additionally a set of macro commands. The adapter includes error correction capability for non error-free cards. The error correction codes used are defined in Section 7.1.

Because the application specific needs and the chosen application interface are out of the scope of this specification, the MultiMediaCard controller defines an internal adapter interface. The two parts communicate across this interface. The adapter interface is directly accessible in low cost (point to point link) systems where the MultiMediaCard controller is reduced to an MultiMediaCard adapter.

3.3.1 Application Adapter Requirements

The application adapter enhances the MultiMediaCard system in the way that it becomes plug&play in every standard bus environment. Each environment will need its unique application adapter. For some bus systems standard, off the shelf, application adapters exist and can interface with the MultiMediaCard adapter. To reduce the bill of material it is recommended to integrate an existing application adapter with the MultiMediaCard adapter module, to form a MultiMediaCard controller.

The application adapter extension is a functional enhancement of the application adapter from a bus slave to a bus master on the standard application bus. For instance, an extended application adapter can be triggered to perform bidirectional DMA transfers.

3.3.2 MultiMediaCard Adapter Architecture

The architecture and the functional units described below are not implementation requirements, but general recommendations on the implementation of a MultiMediaCard adapter. The adapter is divided into two major parts:

- The controller: macro unit and power management
- The data path: Adapter interface, ECC unit, read cache, write buffer, CRC unit and MultiMediaCard bus interface

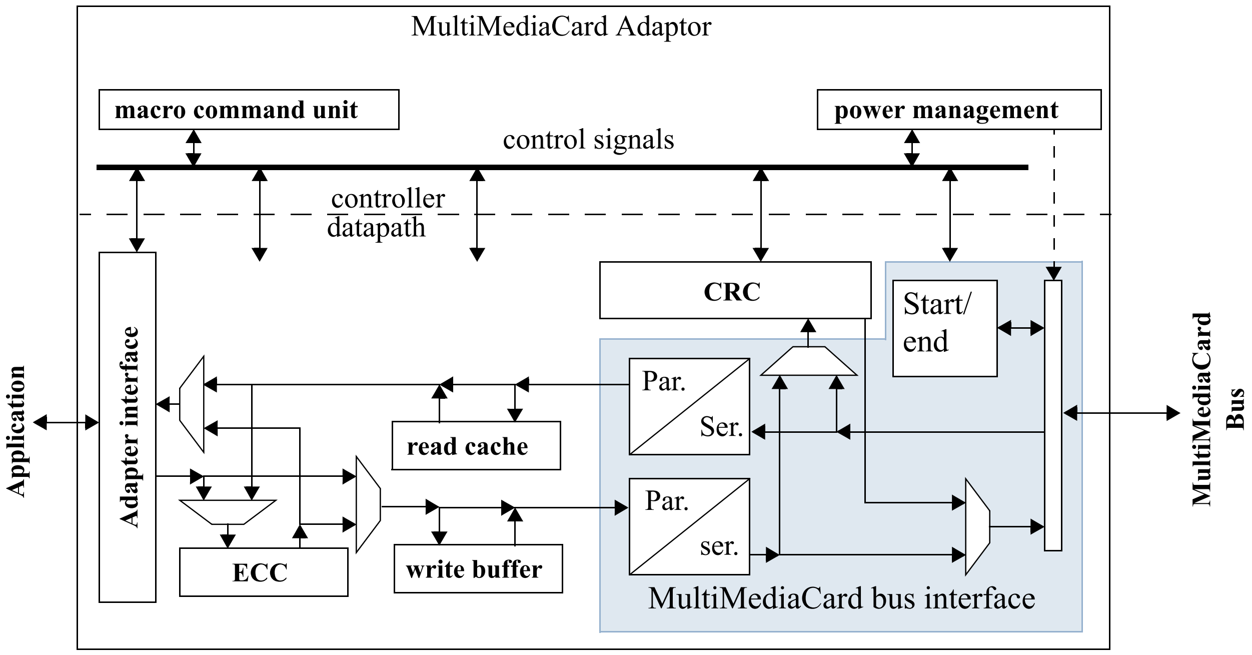

Figure 16: MultiMediaCard Adaptor Architecture

The data path units should be implemented in hardware to guarantee the full capabilities of the MultiMediaCard system. The controller part of the adapter can be implemented in hardware or software depending on the application architecture.

The width of the data path should be a byte; the units which are handling data should work on bytes or blocks of bytes. This requirement is derived from the MultiMediaCard bus protocol, which is organized in data blocks. Blocks are multiples of bytes. Thus, the smallest unit of a data access or control unit is a byte.

Commands for the MultiMediaCard bus follow a strict protocol. Each command is encapsulated in a syntactical frame. Each frame contains some special control information like start/end bits and CRC protection. Some commands include stuffing bits to enable simple interpreters to use a fixed frame length. This transport management information should be generated in the MultiMediaCard adapter. These functions are combined in the MultiMediaCard bus interface of the adapter.

The response delays of the MultiMediaCard system may vary; they depend on the type of cards. So the adapter interface must handle asynchronous mode via handshake signals(STB,ACK) or the host has to poll the state (busy/not busy) if no handshake signals are required (synchronous mode). This interface may be a general unit supporting most application protocols or can be tailored to one application.

It is recommended to equip the MultiMediaCard adapter with data buffers for write and read operation. It will, in most cases, improve the system level performance on the application side.The MultiMediaCard bus transports its data with a data rate up to 416 Mbit/sec. This may be slower than a typical applications CPU bus. Enabling the CPU to off load the data to the buffers will free up CPU time for system level tasks, while the MultiMediaCard adapter handles the data transfer to the card.

The access time for random access read operations from a card may be improved by caching a block of data in the read cache. After reading a complete block into the MultiMediaCard adapter cache, repeated accesses to that block can be done very fast. Especially read-modify-write operations can be executed in a very efficient way on a block buffer with the help of the SRAM swapper.

4 Card Registers

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

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

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

The Card-Specific Data (CSD) register provides information on how to access the card contents. The CSD defines the data format, error correction type, maximum data access time, data transfer speed, whether the DSR register can be used, etc.

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

The writable 16-bit relative card address (RCA) register carries the card address assigned by the host during the card identification. This address is used for the addressed host-card communication after the card identification procedure. The default value of the RCA register is 0x0001.

The 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 cards). The CSD register carries the information about the DSR register usage. The default value of the DSR register is 0x404.

5 The MultiMediaCard Bus

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

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

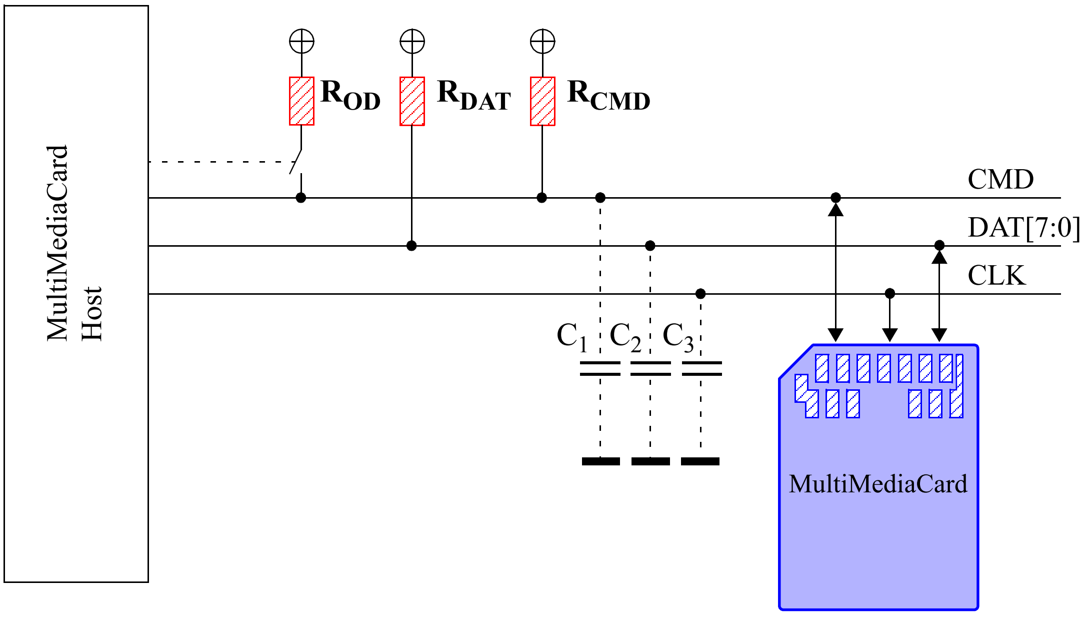

Figure 17: Bus Circuitry Diagram

The R_{OD} is switched on and off by the host synchronously to the open-drain and push-pull mode transitions. The host does not have to have open drain drivers, but must recognize this mode to switch on the R_{OD}. R_{DAT} and R_{CMD} are pull-up resistors protecting the CMD and the DAT lines against bus floating when no card is inserted or when all card drivers are in a high-impedance mode.

A constant current source can replace the R_{OD} by achieving a better performance (constant slopes for the signal rising and falling edges). If the host does not allow the switchable R_{OD} implementation, a fixed R_{CMD} can be used. Consequently the maximum operating frequency in the open drain mode has to be reduced if the used R_{CMD} value is higher than the minimal one.

To guarantee the proper sequence of card pin connection during hot insertion, the use of either a special hot-insertion capable card connector or an auto-detect loop on the host side (or some similar mechanism) is mandatory.

No card shall be damaged by inserting or removing a card into the MultiMediaCard bus even when the power (V_{DD}) is up. Data transfer operations are protected by CRC codes, therefore any bit changes induced by card insertion and removal can be detected by the MultiMediaCard bus master.

The inserted card must be properly reset also when CLK carries a clock frequency f_{PP}. Each card shall have power protection to prevent card (and host) damage. Data transfer failures induced by removal/insertion are detected by the bus master. They must be corrected by the application, which may repeat the issued command.

6 SPI Mode

6.1 Introduction

The SPI mode consists of a secondary, optional communication protocol which is offered by Flash-based MultiMediaCards. This mode is a subset of the MultiMediaCard protocol, designed to communicate with a SPI channel, commonly found in Motorola's (and lately a few other vendors') microcontrollers. The interface is selected during the first reset command after power up (CMD0) and cannot be changed once the part is powered on.

The SPI standard only defines the physical link and not the complete data transfer protocol. The MultiMedia-Card SPI implementation uses a subset of the MultiMediaCard protocol and command set. It is intended to be used by systems which typically require one card and have lower data transfer rates, compared to MultiMedi-aCard-protocol-based systems. From the application point of view, the advantage of the SPI mode is the capa-bility of using an off-the-shelf host, hence, reducing the design-in effort to minimum. The disadvantage is the loss of performance of the SPI mode versus MultiMediaCard mode (lower data transfer rate, hardware CS, etc.).

6.2 SPI Interface Concept

The Serial Peripheral Interface (SPI) is a general purpose synchronous serial interface originally found on cer-tain Motorola microcontrollers. A virtually identical interface can now be found on certain TI and SGS Thom-son microcontrollers as well.

The MultiMediaCard SPI interface is compatible with SPI hosts available on the market. As in any other SPI device, the MultiMediaCard SPI channel consists of the following four signals:

CS: Host to card Chip Select signal.

CLK: Host to card clock signal

DataIn: Host to card data signal.

DataOut: Card to host data signal.

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

6.3 SPI Bus Topology

The card identification and addressing methods are replaced by a hardware Chip Select (CS) signal. There are no broadcast commands. For every command, a card (slave) is selected by asserting (active low) the CS signal (see Figure 18).

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

The bidirectional CMD and DAT lines are replaced by unidirectional *dataIn* and *dataOut* signals. This elimi-nates the ability of executing commands while data is being read or written and, therefore, makes the sequen-tial and multi block read/write operations obsolete. Only single block read/write commands are supported by the SPI channel.

The MultiMediaCard pin assignment in SPI mode (compared to MultiMediaCard mode) is given in Table 4.

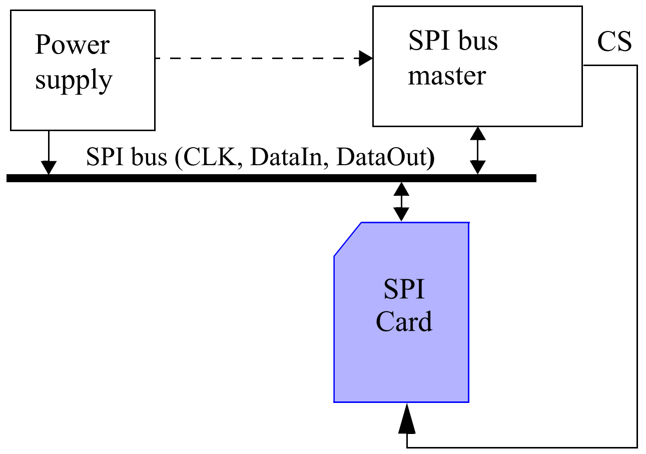

Figure 18: MultiMediaCard Bus System

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

Table 4: SPI Interface Pin Configuration

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

6.4 MultiMediaCard Registers in SPI Mode

The register usage in SPI mode is summarized in Table 5. Most of them are inaccessible.

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

Table 5: MultiMediaCard Registers In SPI Mode

6.5 SPI Bus Protocol

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

Similar to the MultiMediaCard protocol, the SPI messages consist of command, response and data-block tokens. All communication between host and card is controlled by the host (master). The host starts every bus transaction by asserting the CS signal low.

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

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

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

6.6 SPI Mode Transaction Packets

SPI mode transaction packets can be described by one of the following tokens:

- **Command tokens**: various formats and classes that support a set of card functions
- **Response tokens**: signals acknowledging the commands sent
- **Data tokens**: representing data transmission
- **Data error tokens**: identifying data read failure
- **Clearing Status bits**: a SPI mode status returned to the host

6.7 Card Registers

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

6.8 SPI Bus Timing Diagrams

The host must keep the clock running for at least N_{CR} clock cycles after receiving the card response. This restriction applies to both command and data response tokens. Timing diagrams for SPI bus mode use a well-specified set of schematics and abbreviations.

6.9 SPI Electrical Interface

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

6.10 SPI Bus Operating Conditions

The bus operating conditions are identical to MultiMediaCard mode.

6.11 Bus Timing

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

7 Error protection

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

In order to detect data defects on the cards the host may include error correction codes in the payload data. For error free devices this feature is not required. With the error correction implemented off card, an optimal hardware sharing can be achieved. On the other hand the variety of codes in a system must be restricted or one will need a programmable ECC controller, which is beyond the intention of a MultiMediaCard adapter.

If a MultiMediaCard requires an external error correction (external means outside of the card), then an ECC algorithm has to be implemented in the MultiMediaCard host. The shortened BCH (542,512) code was chosen for matching the requirement of having high efficiency at lowest costs.

8 File Formats for the MultiMediaCard

The file format specification, for the MultiMediaCard, starting with V4.0 of this document, has been moved into a separate document called the “File Formats Specifications For MultiMediaCards”.

9 MultiMediaCard Standard Compliance

The MultiMediaCard standard provides all the necessary information required for media exchangeability and compatibility.

- Generic card access and communication protocol (Chapter 4, Chapter 5)
- The description of the SPI mode (Chapter 6)
- Data integrity and error handling (Chapter 7)
- Mechanical iApril 2005uency
- Basic file formats for achieving high data interchangeability.

However, due to the wide spectrum of targeted MultiMediaCard applications—from a full blown PC based application down to the very-low-cost market segments—it is not always cost effective nor useful to implement every MultiMediaCard standard feature in a specific MultiMediaCard system. Therefore, many of the parameters are configurable and can be tailored per implementation.

A card is compliant with the standard as long as all of its configuration parameters are within the valid range. A MultiMediaCard host is compliant as long as it supports at least one MultiMediaCard class as defined below. Card classes include: Read Only Memory (ROM) cards, Read/Write (RW) cards and I/O cards. Every provider of MultiMediaCard system components is required to clearly specify (in its product manual) all the MultiMediaCard specific restrictions of the device.

MultiMediaCards (slaves) provide their configuration data in the Card Specific Data (CSD) register. The MultiMediaCard protocol includes all the necessary commands for querying this information and verifying the system concept configuration. MultiMediaCard hosts (masters) are required (as part of the system boot-up process) to verify host-to-card compatibility with each of the cards connected to the bus. The I/O card class characteristics and compliance requirements will be refined in coming revisions.

The following table summarizes the requirements from a MultiMediaCard host for each card class (CCC = card command class). The meaning of the entries is as follows:

- *Mandatory*: any MultiMediaCard host supporting the specified card class must implement this function.
- *Optional*: this function is an added option. The host is compliant to the specified card class without having implemented this function.
- *Not required*: this function has no use for the specified card class.

Function	ROM card class	R/W card class	I/O card class
26-52 MHz transfer rate	Optional	Optional	Optional
20-26 MHz transfer rate	Mandatory	Mandatory	Mandatory
0-20 MHz transfer rate	Mandatory	Mandatory	Mandatory
2.7-3.6 volts power supply	Mandatory	Mandatory	Mandatory
1.65-1.95 volts power supply	Optional	Optional	Optional
CCC 0 basic	Mandatory	Mandatory	Mandatory
CCC 1 sequential read	Optional	Optional	Optional
CCC 2 block read	Mandatory	Mandatory	Optional
CCC 3 sequential write	Not required	Optional	Optional

Function	ROM card class	R/W card class	I/O card class
CCC 4 block write	Not required	Mandatory	Optional
CCC 5 erase	Not required	Mandatory	Not required
CCC 6 write protection functions	Not required	Mandatory	Not required
CCC 7 lock card commands	Mandatory	Mandatory	Mandatory
CCC 8 application specific com- mands	Optional	Optional	Optional
CCC 9 interrupt and fast read/write	Not required	Optional	Mandatory
DSR	Optional	Optional	Optional
SPI Mode	Mandatory	Mandatory	Mandatory

Comments on the optional functions:

- The interrupt command is intended for reducing the overhead on the host side required during polling for some events.
- The setting of the DSR allows the host to configure the MultiMediaCard bus in a very flexible, application dependent manner
- The external ECC in the host allows the usage of extremely low-cost cards.
- The Card Status bits relevance, according to the supported classes, is defined in Table 21 in Chapter 4

10 Abbreviations and terms

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