Interfacing MMC (MicroSD) Cards with SPI Communication

This document was created with the intention of providing beginner users with some baseline knowledge on how to interface with common MMC cards via SPI for quick, simple, low-cost storage.

Contents

Table of Figures

Table of Tables

MMC Packaging and Standards

The MMC (MultiMediaCard) standard is a NAND based flash-memory system commonly used in multimedia applications. Though the original card spec required a 1-wire serial interface it is now common for devices to use 2, 4 or 8 bits of parallel serialized data for offload. This guide focuses on the 1-wire exchange via SPI.

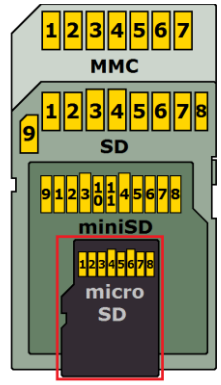


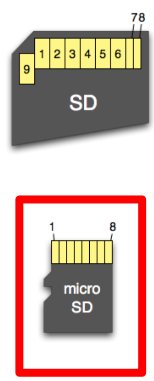
Figure 1: The MMC/MicroSD Form Factors

Secure Digital (SD) and its Subclasses

From the year 2000 onward the SD (Secure Digital) Association (https://www.sdcard.org/home/) has come to represent a significant portion of the MMC market. Though SD does offer some additional functionality on top of basic MMC storage it will not be discussed in this document. The MicroSD form factor will be the targeted application.

Generally speaking each SD/MMC device has a 7-9 pin footprint, but as previously mentioned we will focus on the 8-pin MicroSD form factor for discussion here. Both the SD and MicroSD form factors feature two modes of operation:

1. **SD Bus** in which a single bi-directional command line and one/multiple bi-directional data channels are used to transfer data
2. **SPI Bus** in which a standard SIMO,SOMI, SCLK, and CS structure is used to transfer data and commands to/from the card



Pin	SD	SPI
1	CD/DAT3	CS
2	CMD	DI
3	VSS1	VSS1
4	VDD	VDD
5	CLK	SCLK
6	VSS2	VSS2
7	DAT0	DO
8	DAT1	X
9	DAT2	X

Pin	SD	SPI
1	DAT2	X
2	CD/DAT3	CS
3	CMD	DI
4	VDD	VDD
5	CLK	SCLK
6	VSS	VSS
7	DAT0	DO
8	DAT1	X

Figure 2: SD vs. MicroSD Pin-outs

It should be noted that the MicroSD card has two unused (unconnected) pins in this SPI configuration, labeled as DAT1 and DAT2 in the diagram, these are the parallel data transfer lines which we will leave out for the sake of ease of implementation.

Included below for the sake of comparison is a table of the various MMC form-factors with affiliated physical dimensions, SPI, 1, 4, and 8-bit bus modes, max clock rates, operating voltages, and maximum capacities.

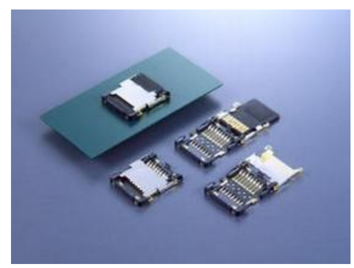
Table 1: MMC Device Summary

Type	MMC	RS-MMC	MMCplus	SecureMMC	SDIO	SD	miniSD	microSD
SD Socket	Yes	Extender	Yes	Yes	Yes	Yes	Adapter	Adapter
Pins	7	7	13	7	9	9	11	8
Width	24 mm	24 mm	24 mm	24 mm	24 mm	24 mm	20 mm	11 mm
Length	32 mm	18 mm	32 mm	32 mm	32 mm+	32 mm	21.5 mm	15 mm
Thickness	1.4 mm	1.4 mm	1.4 mm	1.4 mm	2.1 mm	2.1 mm (most) 1.4 mm (rare)	1.4 mm	1 mm
SPI bus mode	Optional	Optional	Optional	Yes	Yes	Yes	Yes	Yes
1-bit bus mode	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
4-bit bus mode	No	No	Yes	No	Optional	Optional	Optional	Optional
8-bit bus mode	No	No	Yes	No	No	No	No	No
Interrupts	No	No	No	No	Optional	No	No	No
Max clock rate	20 MHz	20 MHz	52 MHz	20 MHz?	50 MHz	208 MHz	208 MHz	208 MHz
Max transfer rate	20 Mbit/s	20 Mbit/s	416 Mbit/s	20 Mbit/s?	200 Mbit/s	832 Mbit/s	832 Mbit/s	832 Mbit/s
Max SPI transfer	20 Mbit/s	20 Mbit/s	52 Mbit/s	20 Mbit/s	50 Mbit/s	25 Mbit/s	50 Mbit/s	50 Mbit/s
DRM	No	No	No	Yes	N/A	Yes	Yes	Yes
User encrypt	No	No	No	Yes	No	No	No	No
Simplified spec	Yes	Yes	No	Not yet?	Yes	Yes	No	No
Membership cost	JEDEC: US$4,400/yr, optional				SD Card Association: US$2,000/year, general; US$4,500/year, executive			
Specification cost	Free			?	Simplified spec: Free. Full spec: Free to members, US$1,000/year to R&D non-members			
Host license	No	No	No	No	Yes, US$1,000/year. No, if using SPI-mode only.			
Card royalties	Yes	Yes	Yes	Yes	Yes, US$1,000/year	Yes	Yes	Yes
Open source compatible	Yes	Yes	Yes?	Yes?	Yes	Yes	Yes	Yes
Nominal operating voltage(s)	3.3 V	1.8 V/3.3 V	1.8 V/3.3 V[5][6]	1.8 V/3.3 V	3.3 V	3.3 V (SDSC), 1.8/3.3 V (SDHC), 1.8/3.3 V (SDXC)	3.3 V (miniSD), 1.8/3.3 V (miniSDHC)	3.3 V (SDSC), 1.8/3.3 V (microSDHC), 1.8/3.3 V (microSDXC)
Maximum available capacity	128 GB	2 GB	128 GB?	128 GB?	?	4 GB (SD), 32 GB (SDHC), 256 GB (SDXC)	4 GB (miniSD), 16 GB (miniSDHC)	4 GB (microSD), 32 GB (microSDHC), 128 GB (microSDXC)
Type	MMC	RS-MMC	MMCplus	SecureMMC	SDIO	SD	miniSD	microSD

Common Sockets and Connectors

The MicroSD form-factor can be connected to using a variety of styles of connectors (sometimes referred to as sockets) available through common distributors such as DigiKey or Mouser. There are 3 classes of connectors that are commonly used. These are summarized in the table below:

Table 2: MicroSD Card Connectors

Connector Name	Description	Example Part	Image
Push-Push	Push card in until it locks to insert, push in until it pops out to remove	Hirose DM3AT Surface Mount Connector DigiKey Part Page: http://www.digikey.com/product-detail/en/DM3AT-SF-PEJM5/HR1964CT-ND/2533566	
Push-Pull	Push card in to insert, pull card out to remove	Hirose DM3D Surface Mount Connector DigiKey Part Page: http://www.digikey.com/product-detail/en/DM3D-SF/HR1941TR-ND/1786510	
Hinged	Open hinge and slide card in to insert, open hinge and pull card out to remove	JAE Electronics ST1W008S4FR2000 Surface Mount Hinged Connector Mouser Part Page: http://www.mouser.com/ProductDetail/JAE-Electronics/ST1W008S4FR2000/?qs=sGAEpiMZZMuJakaoiLiBpgH5xmmIbPf4Y69%2fMnRM%252bYY%3d	

It is worth noting that in addition to the 8-pins provided from the MMC card which each of these connectors allows us to place on a PCB, we are commonly also given what is referred to as a Card Detect (CD) line to represent the presence of a MicroSD card in the connector. This CD is normally a small integrated, mechanical switch that requires two external connections (input and output).

Figure 3 shows a common MicroSD connector foot print captured from a schematic using a Hirose push-push connector. Note the connections to the CD lines (the MMC_CD net should be pulled high and potentially de-bounced to provide full CD functionality).

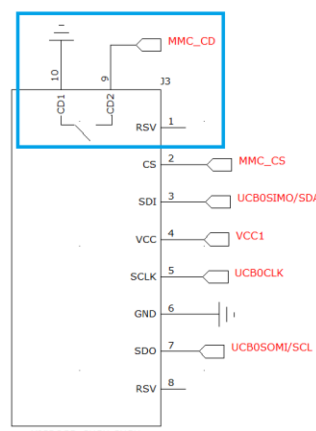


Figure 3: MicroSD Connector Schematic Symbol

The additional dimensioning information needed to create the affiliated PCB footprint is typically provided by the manufacturer in a mechanical drawing included as a part of the connector data sheet. An example is provided for the Hirose DM3 Series: http://www.hirose.co.jp/cataloge_hp/e60900232.pdf.

It is worth noting that schematic symbols and layout footprints for both the Hirose DM3AT and JAE Electronics ST1W008S42 series have been created by the UVa INERTIA group previously for the Mentor Graphics PADs Design and Layout suite. These are available upon request for new UVa-based MMC hardware developers.

Electrical Specs and SPI Interfacing

This section will focus on minimal considerations for getting an MMC platform powered, clocked, and communicating via the on-board SPI interface with minimal overhead.

Power Considerations

The MicroSD form factor is a convenient embedded data logging form factor as it works at both 1.8 and 3.3V, consumes acceptable power on read/write (when compared with something like radio TX/RX power) and has incredibly low quiescent (non-read/write) currents.

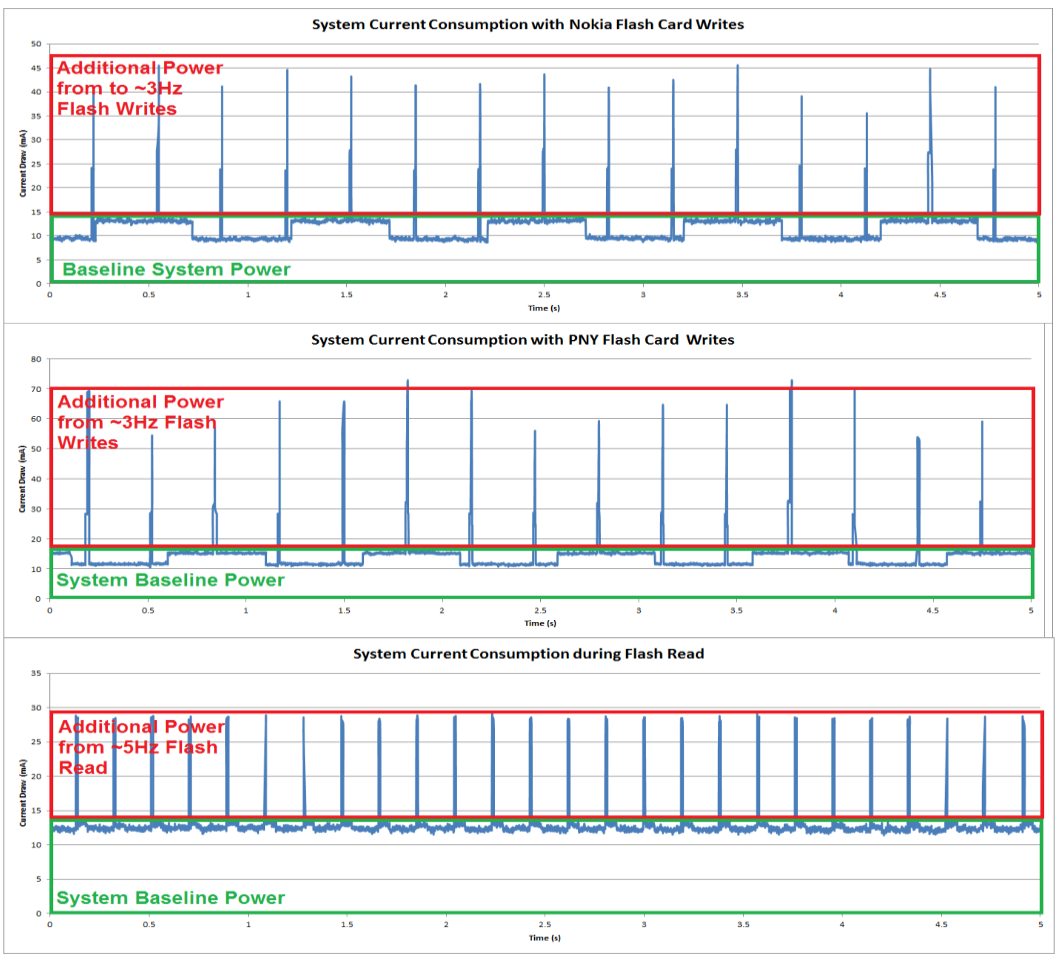


Figure 4: MicroSD Write/Read Currents

Though the maximum power (typically between 80-200mW) is only drawn for a fraction of a second it is worth assuring that any supply circuitry connected to the card is capable of sourcing this type of current instantaneously. Spurious or sub-nominal voltage levels on the SPI bus (across multiple cards) are a good indicator that supply currents are not sufficient, this can be verified by probing the voltage input line for large dips on flash read/write operations.

Clocking Considerations

It is worth briefly stating some of the clocking considerations for a typical SPI-based MMC setup. Though no dedicated external clock need be provided to the card (other than SPI clocking, commonly referred to as SCLK) there are several constraints on what rates can be used for commands/data transfers.

Though initialization procedure will be summarized in a later section it is worth mentioning that card initialization (see Power-On Procedure) must occur at a lower rate than the remainder of card operation. For the purpose of initialization an SCLK rate of 100-400 kHz is recommended.

NOTE: If an SCLK rate higher than 400 kHz is used to initialize the card a number of errors may occur. For all example code an SCLK rate of about 230 kHz appeared to work well in successfully initializing the cards.

Once initialized the SCLK rate can be increased significantly, though some companies advertise data transfer rates into 100’s of MHz we typically use 25MHz as an upper-bound for the SCLK rate.

SPI Interface

The MMC SPI interface conforms to a standard SPI structure and accepts two of the commonly proposed clock phase/polarity disciplines.

Standard SPI connections are made as follows:

Table 3: SPI Functionality Summary

Signal Name	Signal Description	MMC Signal Function
MOSI	Slave input, Master Output	Data input to MMC
MISO	Slave output, Master Input	Data output from MMC
SCLK	Clock	Clock input to MMC
CS	Chip select (active low)	Logic input to MMC

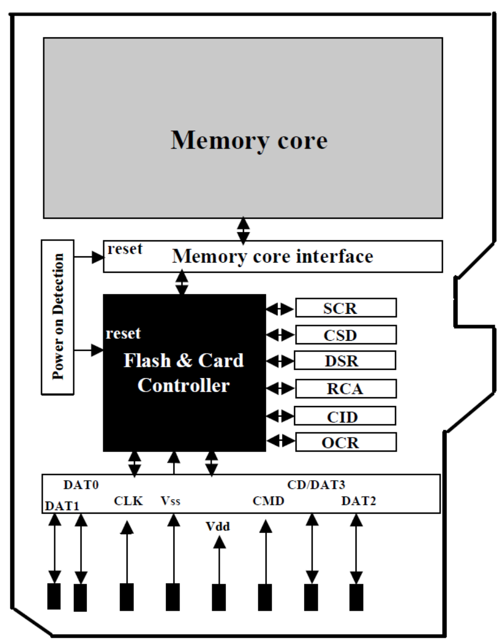


Figure 5: Internal MMC SPI Structure

A diagram of the correction connections between the desired controller device and the MMC card is included below. With the standard SPI connections made between the card and the controller device there are a few additional considerations we may want to explore based upon more controller-specific information, this is included in the following section.

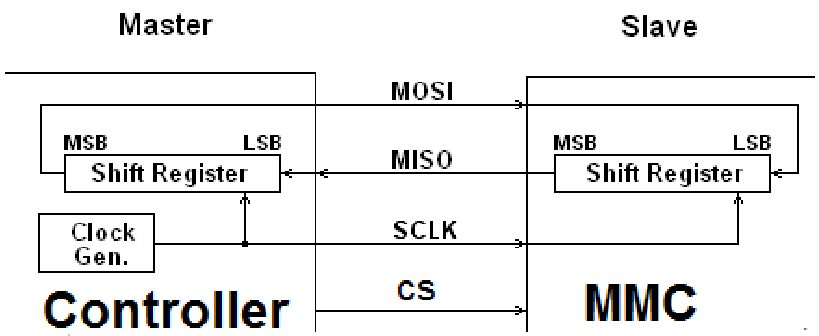


Figure 6: Example Controller/MMC Connection Diagram

SPI Modes: SCLK Phase and Polarity

When providing the SCLK signal to the MMC card we need to adhere to certain rules regarding the phase and polarity of the SCLK relative to the MOSI and MISO lines. Generally speaking there are 4 "Modes" in which the SPI protocol works.

In Figure 7 the 4 "modes" of operation are summarized. Though it is convenient to think of these modes as universal descriptors of SCLK operation, **the convention for naming often differs manufacturer to manufacturer**. Make sure to verify the operation of your SCLK against either mode 0 (the recommended mode) or mode 3 included below.

It should be noted the significant constraint on the SCLK discipline here is that the rising edge of SCLK latches in data, while the falling edge shifts it out.

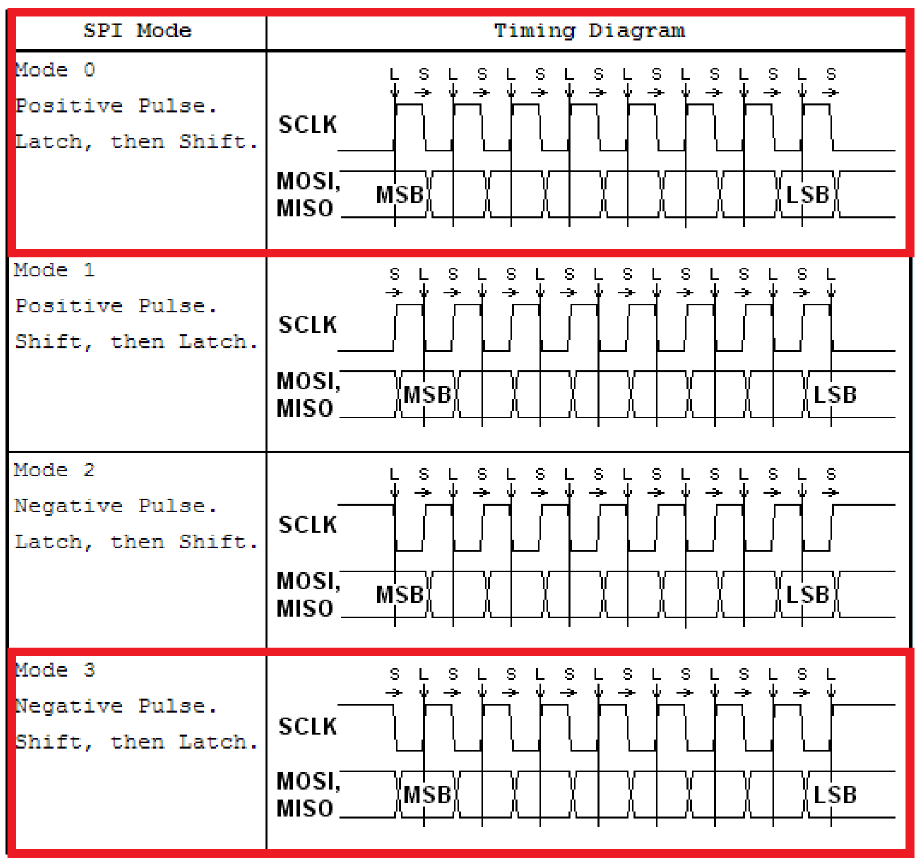


Figure 7: 4 Modes of SPI SCLK Operation

Devices which do not use this rising/falling discipline will **not** be able to successfully communicate with MMC cards regardless of remaining SPI structure. If a device without this SCLK discipline is to be used as a controller the SPI can be emulated either through a bit-banging interface or with a parallel-input shift register and split clocking scheme.

The Command Interface

A developer interacts with the MMC card over SPI via the SPI command set. When an MMC card first boots (sometimes also referred to as power-on, or insertion) it needs to be setup for SPI initialization. Following this procedure initialization occurs, which generally takes some time on the order of 1-100ms, the card is ready to accept data transfers or queries for register values.

Power-On Procedure

When an MMC card is inserted into a socket (CD triggered low/high correspondingly), with at least 2.2V present on the VCC line it is typically ready to accept the pre-initialization routine within 1ms (VCC stabilization period). For this the SCLK will need to be set between 100 and 400 kHz as previously mentioned (see Clocking Considerations).

Following SCLK establishment, the **CS line to the device should be set high** (de-selecting the card) and **at least 74 clock pulses** must be applied to the device. Generally speaking 10 1-byte transfers (corresponding to 80 SCLK cycles) can be used for this procedure.

Once these pulses are applied the card enters its native operating mode and becomes ready to accept common commands. Before any other command is issued to the device a `CMD0`, or device reset, is typically sent. Next, CMD1, or device initialization, must be written. The R1 response code (outlined in a later section) then indicates the idle state in response this CMD1 until initialization has completed. This is summarized in the scope capture included below:

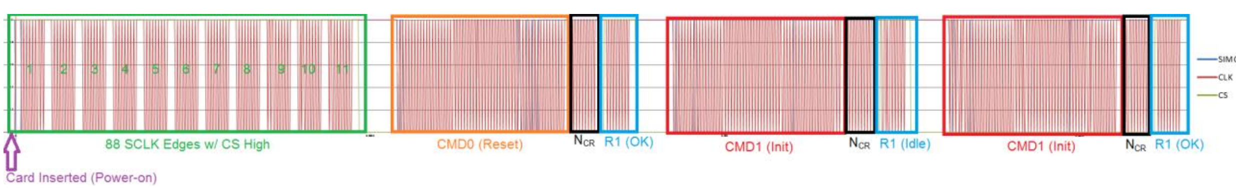


Figure 8: MMC Power-On Procedure

MMC SPI Commands and Command Set

The MMC SPI mode uses a standard format command structure with a 6-byte out-going message, followed by a variable length clocking period, followed by a 1-byte response. This structure is outlined below.

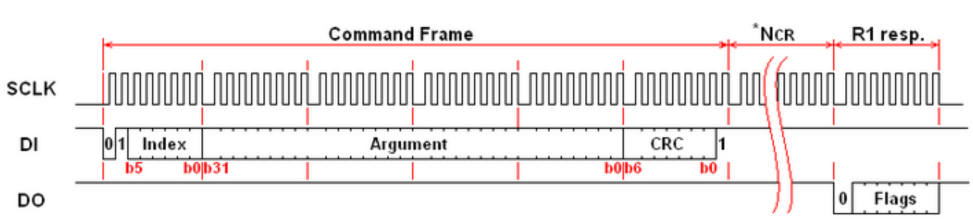


Figure 9: MMC SPI Command and Response

In Figure 9 the N_{CR} value is the time-to-command response, for SD cards this time is typically 0-8 bytes of SCLK, while for MMC cards this time is typically 1-8 bytes of SCLK. It is also worth noting that the CRC field is optional in SPI mode.

The MMC command structure (based on command indexes) is summarized in the table below. It should be noted that the SDC only commands (sometimes prefixed with ACMD) are omitted from this table.

Table 4: MMC SPI Command Set

Index	Argument	Res.	D	Name	Command Description
CMD0	None (0)	R1	N	GO_IDLE_STATE	Software Reset
CMD1	None (0)	R1	N	SEND_OP_COND	Initiate Init Process
CMD9	None (0)	R1	Y	SEND_CSD	Read Card Specific Data Register
CMD10	None (0)	R1	Y	SEND_CID	Read Card ID Register
CMD12	None (0)	R1b	N	STOP_TRANSMISSION	Stop to read data
CMD16	Block Len [15:0]	R1	N	SET_BLOCKLEN	Change R/W block size
CMD17	Address [31:0]	R1	Y	READ_SINGLE_BLOCK	Read a block
CMD18	Address [31:0]	R1	Y	READ_MULTIPLE_BLOCK	Read multiple blocks
CMD23	Number of blocks [15:0]	R1	N	SET_BLOCK_COUNT	For MMC only. Define number of blocks to transfer with next multi-block operation
CMD24	Address [31:0]	R1	Y	WRITE_BLOCK	Write a block
CMD25	Address [31:0]	R1	Y	WRITE_MULTIPLE_BLOCK	Write multiple blocks
CMD58	None (0)	R3	N	READ_OCR	Read operating conditions register

In the table above the **Res.** Column corresponds to the type of response code the card will provide upon receiving the command it is worth noting that for all but reads of the operating conditions register (OCR) this is an **R1** type response. In addition the **D** column indicates whether or not the command will require the additional clocking in/out of data to complete its function. A 'Y' in the D column means some amount of additional SCLK edges is required for this command to be completed.

Bit #	0	1	2	3	4	5	6	7
Value	0	Param Error	Addr Error	Erase Seq Error	CRC Error	Illegal Command	Erase Reset	In Idle State

Figure 10: R1 Response Code

In Figure 10 above the R1 response code is outlined in terms of its potential error fields. Upon reception, this response can be decoded to find any significant errors which have occurred during the command execution.

An R1b response is simply an R1 response whose SCLK time to return may be larger than N_{CR}, while an R3 response indicates a normal R1 response followed by 4 bytes (32 bits) of OCR values being placed on the bus.

MMC Card ID Exchange and Connection Testing

ADD SECTION HERE

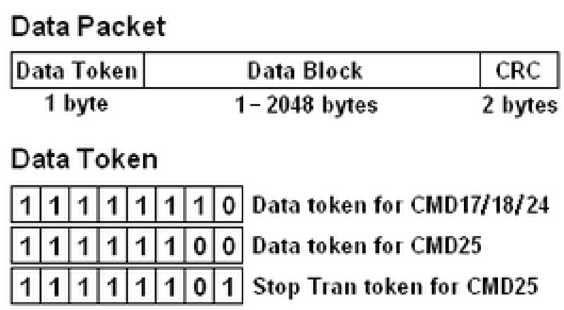
MMC Data Exchange

When large data blocks are to be written to/from the card using CMD17/18 or CMD24/25 several basic primitives can be followed. Depending on the amount of data transfer needed for the targeted application the single/multi-block read/write implementation should be determined based on user specs.

The Data Packet

A single block is written to the MMC card in the form of a data packet, which has fixed format and length for a given block size. This block includes a data token (for multi-block control), data block, and CRC field. The **stop tran** token shown at right need not be followed by a data block or CRC as it signals the end of a multiple block write event.

Figure 11: Data Packet Format



The Data Response

During a write as each data packet is provided to the MMC card it responds with an affiliated data response code summarizing the status of the packet which was just attempted to be written. This data response is responsible for flagging data rejections due to CRC errors along with more generic write errors.

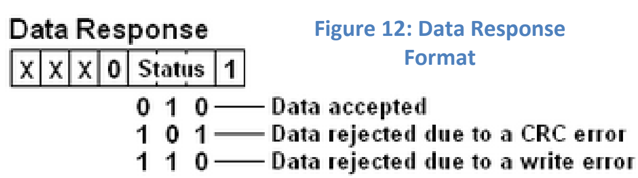


Figure 12: Data Response Format

Single Block Read/Write Operations

When the user requests a single block read/write operation using CMD17 or CMD24 the card provides the correct R1 command response, then either clocks in or out the requested sector to be written or read respectively. When a write occurs the data packet is followed by a single data response which indicates the status of the data to be written, while for a single block read no additional information is provided at the end of the received block.

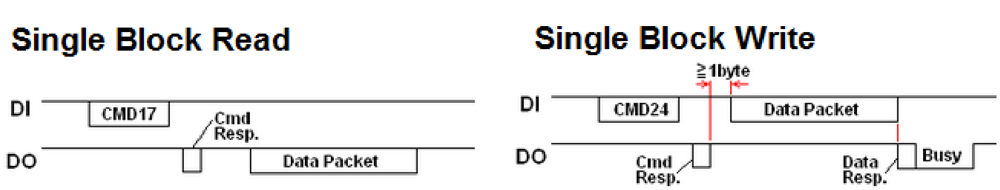


Figure 13: Single Block Read/Write Interaction

Multi-Block Reads

If a developer needs to stream a large amount of information from the MMC card to a desired endpoint without performing additional command requests in between data packet receptions. The multi-block read works in 2 modes, 1 with a desired length to read, set with CMD23, and another with unfixed length (sometimes called open-ended multi-block read) set by calling CMD25 without first setting the block size using CMD23. When the open-ended multi-block read is used the stream of data being clocked back in from the MMC device needs to be interrupted using CMD12, the stop transmission command.

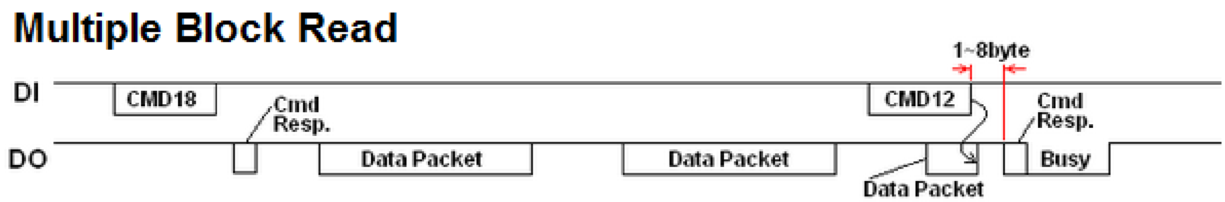


Figure 14: Multiple Block Read

Multi-Block Writes

Similarly, if a developer desires to stream a large amount of data packets into a card without requiring repeated command mode operation the multi-block write procedure is used to accomplish this task. Upon start of a multi-block write the card will accept data (in the standard data packet format) until it receives a data packet whose data token matches the **Stop Tran** token, the remainder of this data packet will then be ignored and the write completed.

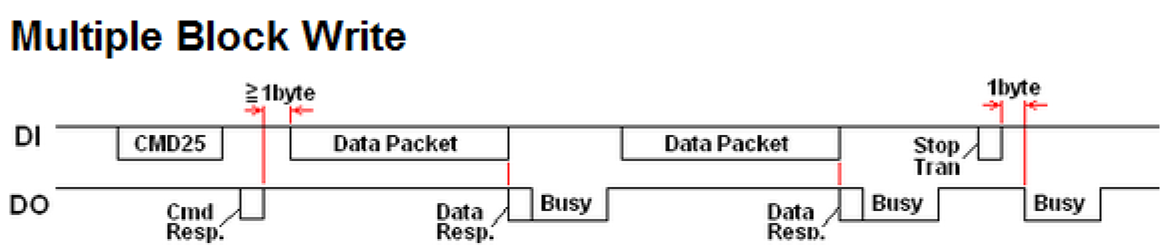


Figure 15: Multiple Block Write

The MMC C Library (for MSP430)

For the sake of simplified MMC development for a commonly used platform, an MMC library has been created for the Texas Instruments MSP430 line of microcontrollers. This library began as a set of TI example code and has grown to a full-featured MMC interface that relies only on 3 basic SPI communication primitives.

These three primitives are outlined in the table below. Though this C code should compile and run for any MSP430 series MCU provided these primitives are implemented correctly there is a possibility of reuse of this library if these primitives are correctly implemented for another MCU architecture.

Table 5: SPI Primitives for MMC C Libraries

Primitive Name	Primitive Function	Arguments/Returns
`spiSendByte(outByte)`	SPI Byte Swap	outByte – a single byte to clock out Returns – a single byte clocked back in
`spiSendFrame(*data, len)`	SPI Frame Transmit	*data – a data pointer to start of frame len – length of the frame to transmit
`spiReadFrame(*data, len)`	SPI Frame Receive	*data – a write-back pointer to start of storage for frame len – length of the frame to receive

From these 3 basic primitives we built the full functionality of the MMC communication interface. Simplified functions for initialization, setting/checking the idle state, getting R1-R3 responses, reading/writing blocks, setting the block length, and reading device size/registers are included. A function outline is provided below:

Useful Resources

Below is a list of resources that helped to produce this document and the information included:

SparkFun MicroSD Spec: http://www.sparkfun.com/datasheets/Prototyping/microSD_Spec.pdf

ELM How to Use MMC/SDC: http://elm-chan.org/docs/mmc/mmc_e.html

SAMSUNG microSD Card Product Datasheet [Download]:
https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CDIQFjAA&url=http%3A%2F%2Fkazus.ru%2Fforums%2Fattachment.php%3Fattachmentid%3D17256%26d%3D1295798733&ei=tpwiUbW5KK3r0QGCm4G4Dg&usg=AFQjCNGTMANKFVFLU_KAw6ltUOhn1fX1qA&bvm=bv.42553238,d.dmQ&cad=rja

SAMSUNG MMC Host Algorithm Guideline:
http://www.eetasia.com/ARTICLES/2004NOV/A/2004NOV26_MSD_AN.PDF

SAMSUNG SD & MicroSD Card Product Family: http://www.kosmodrom.com.ua/pdf/RASPBERRY-PI-PROG-4GB-SDCARD.pdf

Hirose Connectors Memory Card Page: http://www.hirose-connectors.com/connectors/H203SeriesCategorySearch.aspx?p1=1&cat=05&key=SD%20CARD&keym=2

JAE Connectors Memory Card Catalog Page: http://jae-connectors.com/en/pdf/2008-17-SF6.pdf

Physical Description and Pinout: http://www.interfacebus.com/MicroSD_Card_Pinout.html

Wikipedia MMC: http://en.wikipedia.org/wiki/MultiMediaCard

Wikipedia SD: http://en.wikipedia.org/wiki/Secure_Digital

FreeScale Setup and use of MMC: http://cache.freescale.com/files/32bit/doc/app_note/AN3049.pdf