Technical notes on using Analog Devices DSPs, processors and development tools
Contact our technical support at dsp.support@analog.com and at dsptools.support@analog.com
Or visit our on-line resources http://www.analog.com/ee-notes and http://www.analog.com/processors

Interfacing MultiMediaCard™ with ADSP-2126x SHARC® Processors

Contributed by Aseem Vasudev Prabhugaonkar and Jagadeesh Rayala *Rev 1 – March 11, 2005*

Introduction

This application note describes how to implement the interface between an ADSP-2126x SHARC® processor and a MultiMediaCard™ (MMC). The application note also describes the MMC command format and demonstrates with example code how an MMC card can be interfaced seamlessly with the SHARC processor's SPI port. Example code supplied with this application note implements the most commonly used commands of MultiMediaCard.

About MultiMediaCard

The MMC was introduced in 1998 and had an amazing reduction in cubic capacity compared with CompactFlash™. MMC cards are now widely used in digital cameras, smart cell phones, PDAs, and portable MP3 players. Their intended use is to store information and content.

The MMC consists of a 7-pin interface and supports two serial data transfer protocols viz. the MMC (MultiMediaCard) mode and SPI (Serial Peripheral Interface) mode. The maximum operating clock frequency used for serial communication in both modes can go up to 20 MHz. The data written in any of these modes can be read by host in either mode. The advantage of MMC supporting SPI mode is that MMC can be interfaced seamlessly to many controllers or DSP processors, which have on-chip support for SPI. Most MMCs have a communication voltage from 2.0 to 3.6 V, a memory access voltage of 2.7 to 3.6 V, and a capacity from 4 MB to the gigabyte range.

About the ADSP-2126x SPI Port

The ADSP-2126x processor is equipped with a synchronous serial peripheral interface port that is compatible with the industry-standard Serial Peripheral Interface (SPI). The SPI port supports communication with a variety of peripheral devices including codecs, data converters, sample rate converters, S/PDIF or AES/EBU digital audio transmitters and receivers, LCDs, shift registers, micro-controllers, and FPGA devices with SPI emulation capabilities.

Important features of ADSP-2126x SPI port include:

- Simple four-wire interface, consisting of two data pins, a device select pin, and a clock pin
- Full duplex operation, allowing simultaneous data transmission and reception on the same SPI port
- Data formats to accommodate little and big endian data, different word lengths, and packing modes
- Master and slave modes as well as multimaster mode in which the ADSP-2126x processor can be connected to up to four other SPI devices
- Open drain outputs to avoid data contention and to support multimaster scenarios

- Programmable bit rates, clock polarities, and phases
- DMA capability, allowing transfer of data without core overhead
- Master or slave booting from a master SPI device

The MultiMediaCard Interface

In SPI mode, four signals (clock, data in, data out and chip select) are used for the interface. The clock is used to drive data out on the data out pin and receive data on the data in pin. The host drives commands and data to the MMC over the MMC's data in pin. The host receives response and data from the MMC on its data out pin. The chip select signal is used to enable the MMC during data and command transfer. The chip select signal is also used initially to drive the MMC in SPI mode. Note that in SPI mode, data is transferred in units of eight clock cycles.

Pin	Name	Type	Function
1	CS#	Input	Chip select (active low)
2	Din	Input	Data input
3	VSS1	Power	GND
4	VDD	Power	VCC
5	CLK	Input	Clock input
6	VSS2	Power	GND
7	Dout	Output	Data output

Table 1. MultiMediaCard Pin Assignment

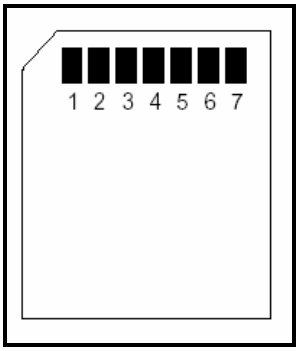


Figure 1. MultiMediaCard Pin Assignments

The MMC pin assignments in SPI mode are shown in Table 1 and Figure 1. Figure 2 shows the MMC interface with the ADSP-2126x SPI port.

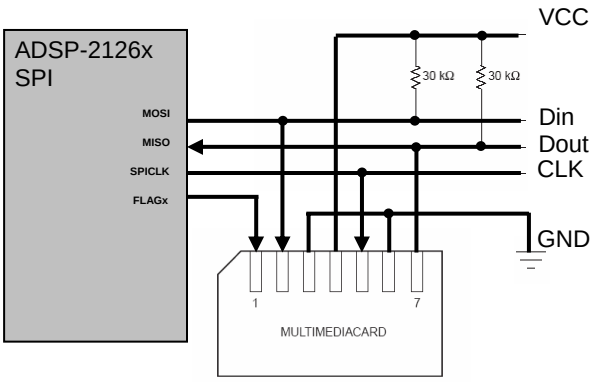


Figure 2. MultiMediaCard Interface with SPI Port

The MultiMediaCard Protocol

The SHARC processor's SPI issues commands to the MMC over the data in (`Din`) pin of the MMC. The data in pin of the MMC is connected to `MOSI` of the SPI. The data is also written to the MMC over the data in signal of the MMC. Based on the received command, the MMC sends response or data on the data out (`Dout`) pin. The data out pin of the MMC is connected to `MISO` of the SHARC processor's SPI port. The processor's SPI port uses one of the Programmable Flag pins (`FLAGx`) to drive `CS#` of the MMC. The communication is initiated by different commands sent from the SHARC processor to the MMC. All commands are six bytes long and are transmitted MSB first. Refer to Figure 3 for generic transfer protocol between the MMC and the SHARC processor's SPI port.

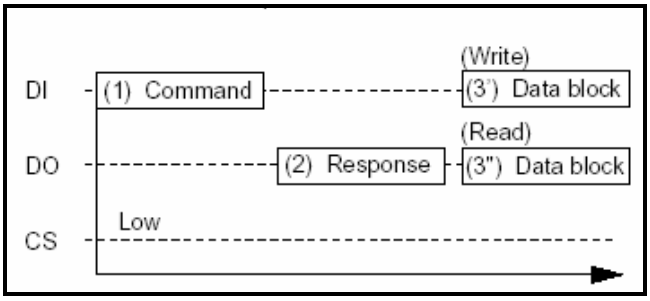


Figure 3. MultiMediaCard Transfer Protocol

Commands and Responses

Table 2 lists the MultiMediaCard's most commonly used commands in SPI mode. The command format is shown in Figure 6.

Each MMC command consists of 48 bits (6 bytes) comprising a start bit (always 0), a transfer bit (always 1), a 6-bit command field, a 4-byte (32-bit) argument field, a 7-bit CRC field, and an end bit (always 1). The argument field contains the necessary information (card relative address, read address, write address, etc.) for issuing that command. For every received command (except the SEND_STATUS command), the MMC responds with a token value.

The R1 response token is 1-byte long and its MSB is always 0. The other bits in the response indicate error conditions. The structure of an R1 response is shown in Figure 4. The R1 format byte description is given in Table 3.

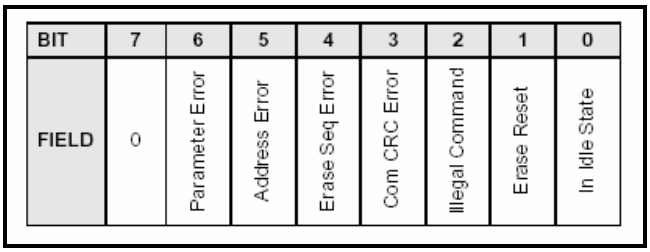


Figure 4. The MultiMediaCard R1 Response Format

Response format R2 is 2 bytes long. The response token is sent by the card as a response to the SEND_STATUS command. The format of the R2 status is shown in Figure 5.

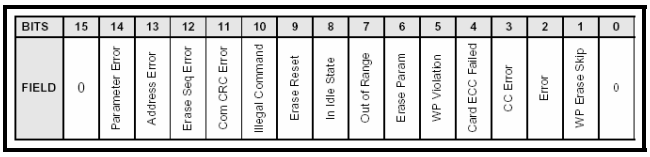
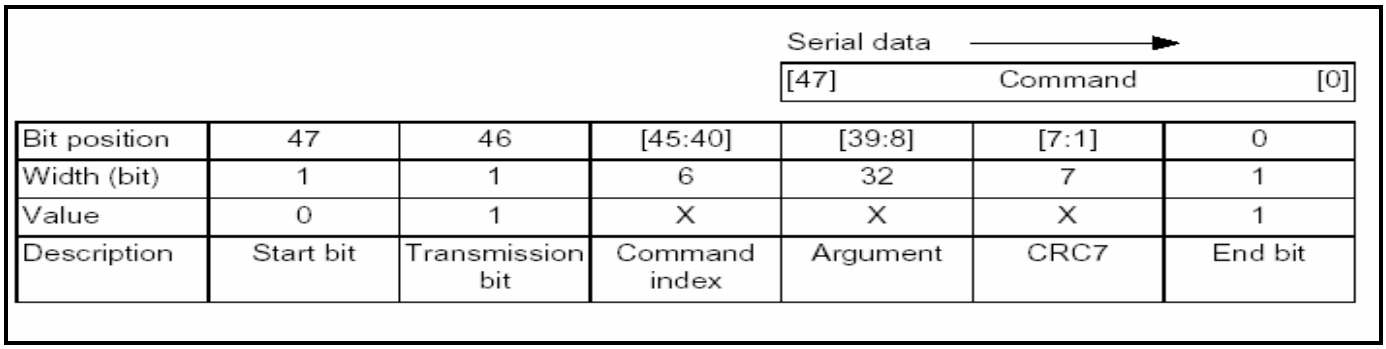


Figure 5. The MultiMediaCard R2 Response Format

The R2 format description is given in Table 4.

CMD INDEX	ARGUMENT	RESPONSE	ABBREVIATION	COMMAND DESCRIPTION
CMD0	None	R1	GO_IDLE_STATE	Resets the MultiMediaCard
CMD1	None	R1	SEND_OP_COND	Activates the card Initialization process
CMD13	None	R2	SEND_STATUS	Asks the selected card to send its status register
CMD16	[31:0]block length	R1	SET_BLOCKLEN	Selects a block length (in bytes) for all following block commands (read and write).
CMD17	[31:0]data address	R1	READ_SINGLE_BLOCK	Reads a block of size selected by the SET_BLOCKLEN command
CMD24	[31:0]data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command
CMD32	[31:0]data address	R1	TAG_SECTOR_START	Sets the address of the first sector of the erase group
CMD33	[31:0]data address	R1	TAG_SECTOR_END	Sets the address of the last sector in a continuous range within the selected erase group, or the address of a single sector to be selected for erase.
CMD34	[31:0]data address	R1	UNTAG_SECTOR	Removes one previously selected sector from the erase selection
CMD38	[31:0]don't care	R1b	ERASE	Erases all previously selected sectors
CMD59	[31:1]don't care [0:0]CRC option	R1	CRC_ON_OFF	Turns the CRC option on or off. A '1' in the CRC option bit will turn the option on. A '0' will turn it off.

Table 2. Most Commonly Used MultiMediaCard Commands



Bit position	47	46	[45:40]	[39:8]	[7:1]	0
Width (bit)	1	1	6	32	7	1
Value	0	1	X	X	X	1
Description	Start bit	Transmission bit	Command index	Argument	CRC7	End bit

Figure 6. The MultiMediaCard Command Format

BIT	NAME	DESCRIPTION
7	0	Fixed to '0'
6	Parameter Error	The command's argument (e.g. address, block length) was out of the allowed range for this card
5	Address Error	A misaligned address, which did not match the block length, was used in the command
4	Erase Seq Error	An error in the sequence of erase commands occurred
3	Com CRC Error	The CRC check of the last command failed
2	Illegal Command	An illegal command code was detected
1	Erase Reset	An erase sequence was cleared before executing because an out of erase sequence command was received
0	In Idle State	The card is in idle state and running initializing process

Table 3. MultiMediaCard R1 Response Format Description

BIT	NAME	DEFINITION
15	0	Fixed to '0'
14	Parameter Error	The command's argument (e.g. address, block length) was out of the allowed range for this card
13	Address Error	A misaligned address, which did not match the block length was used in the command
12	Erase Seq Error	An error in the sequence of erase commands occurred
11	Com CRC Error	The CRC check of the last command failed
10	Illegal Command	An illegal command code was detected
9	Erase Reset	An erase sequence was cleared before executing because an out of erase sequence command was received
8	In Idle State	The card is in idle state and running initializing process
7	Out of Range	
6	Erase Param	An invalid selection, sectors or groups, for erase
5	WP Violation	The command tried to write a write protected block
4	Card ECC Failed	Card internal ECC was applied but failed to the corrected data
3	CC Error	Internal card controller error
2	Error	A general or an unknown error occurred during the operation
1	WP Erase Skip	Only partial address space was erased due to existing WP blocks
0	0	Fixed to '0'

Table 4. MultiMediaCard R2 Response Format Description

The Algorithm

By default, the MMC starts in MMC mode after power up. To configure the card for SPI mode, the CS# is driven logic low while transmitting the CMD0 command. The following procedure is required to set up data transfer in SPI mode.

1. After power up, drive CS# inactive (logic high). This disables the card .
2. Issue at least 80 dummy clock cycles for MMC initialization.
3. Drive CS# low and transmit a CMD0 command. At this point, card mode changes from default MMC mode to SPI mode.
4. Wait for an R1 response from the MMC. The R1 response from the MMC should be 0x01. Any other value indicates an error condition. Re-execution starting from powering on may be required.
5. Issue a CMD1 command and wait for an R1 response. The R1 response should be 0x00. Any other response value indicates of error condition.
6. After a correct R1 response for CMD1, data transfer can occur. Note that until the CMD1 command is successful, the SPI baud rate (clock frequency) should be less than 400 kHz. Before data transfer can begin, the SPI baud rate can be increased.
7. Commands such as set block length can now be issued.
8. Issue commands to read/write the MMC card.
9. When powering off, check that the card is in the Ready state and drive CS# high (inactive) before turning off the power supplies.

The commands implemented in the supplied code are:

- MMC initialization
- CMD0 (GO_IDLE_STATE)
- CMD1 (SEND_OP_COND)
- CMD16 (SET_BLOCKLEN)
- CMD24 (WRITE_BLOCK)
- CMD59 (CRC_ON_OFF)
- CMD17 (READ_SINGLE_BLOCK)

The following are screen captures for some of the MMC commands:

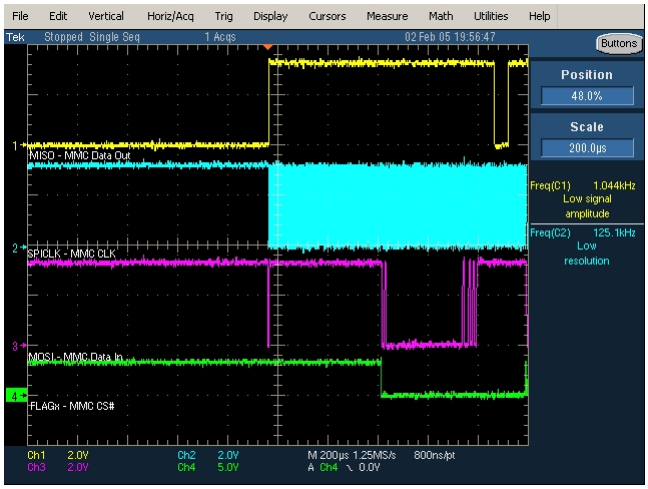


Figure 7. Initialization with 80 CLK Cycles, CMD0 Command and R1 Response – 0x01

Figure 8. CMD1 Command and R1 Response - 0x00

Figure 9. Write Data Completed – 0x00 (BUSY) Followed by 0xFF(High) Indicating Write Complete

Figure 10. Data Being Read After Receiving Read Token 0xFE from the MMC

The example code is supplied in the file `EE264v01.zip` associated with this EE-Note. For further information on commands, responses, and code flow, refer to the `readme.txt` file included in the ZIP file.

References

[1] *ADSP-2126x SHARC DSP Peripherals Manual.* Rev 2.0, January 2004. Analog Devices, Inc.

[2] *ADSP-21262 DSP EZ-KIT Lite Evaluation System Manual.* Rev 1.2, March 2004, Analog Devices, Inc.

[3] *Interfacing a MultiMediaCard to the LH79520 System-On-Chip.* SHARP Application Note.

[4] *MultiMediaCard User's Manual, ADE-603-002B.* Rev.3.0, 3/20/2003 Hitachi, Ltd.

Document History

Revision	Description
Rev 1 – March 11, 2005 *by Aseem Vasudev Prabhugaonkar and Jagadeesh Rayala*	Initial Release