Interfacing the MSP430
With MMC/SD Flash Memory Cards

Stefan Schauer, Christian Speck   *MSP430*

ABSTRACT

This application report and the associated source code files demonstrate the implementation of a serial peripheral interface (SPI) between the MSP430F161x microcontroller and an MMC or SD flash memory card used in SPI mode. The provided information can be used with any MSP430 device with a hardware SPI interface.

The sample software described in this application report can be downloaded from http://www.ti.com/lit/zip/slaa281.

1   Hardware Description

The MSP430F161x is used to communicate with the MMC or SD card via the SPI interface. SPI is a fast and efficient protocol that allows for simultaneous bidirectional data transfer. Serial data is transmitted and received by the MSP430 using the USART module in SPI mode. The hardware interconnection for the master-slave configuration operating on a single supply voltage is shown in Figure 1.

The associated MSP430F161x pins P5.3 and P5.4 are configured as GPIO to control the chip-select pin and read the memory card detect signal. The USART1 hardware peripheral of the MSP430F161x is configured in the 3-pin SPI mode. Pins P5.1, P5.2, and P5.3 of the MSP430 provide the SIMO1, SOMI1, and UCLK1 interface to the MMC card.
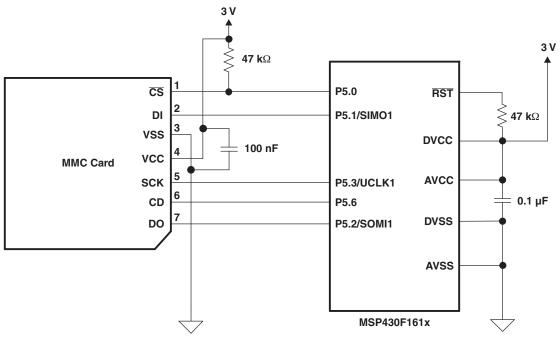

Figure 1. Connection Between MSP430 and MMC Card

2 Software Description

The code associated with this application note is designed as a driver set for communication between an MSP430 and an MMC card or an SD card via the SPI bus as illustrated in the preceding hardware section. The code is written modularly and can be reused easily. Note that only a subset of the available card commands is used, based on the limitations of the SPI interface and the secure functions of the SD card. This sample code can be downloaded from http://www.ti.com/lit/zip/slaa281.

An example main() function is provided to illustrate proper use of the driver functions. It initializes the MSP430F161x in the 3-pin SPI mode with the function mmcInit(). The MSP430 then polls until an MMC card is detected and tries to read the device memory size. Upon the completion, the MSP430 prepares a buffer with 512 bytes of data and writes it into two different sectors of the card. Finally, the MSP430 reads the data back that was written to each memory segment.

After uncommenting the code line

```
//#define withDMA
```

the MSP430F161x DMA module is used for data transmission between the MSP430 and the MMC card, resulting in higher communication speed and less CPU load.

If the software is adapted to a different USART or a different device in which the USART is connected to other port pins, the following functions must be checked for the correct settings:

- void initSPI (void)
- char mmcInit (void)
- mmc.h file

3 Function Description

3.1 char mmcInit (void);

Initialize the port, SPI, and the MMC card. This routine may need to be adjusted if the library is adapted to another MSP430 type or to another USART module. Also, some of the control signals can be set to a different port, if necessary.

Parameter:	None	
Return:	Status	Error/success code

3.2 char mmcping(void);

Check if MMC card is present.

Parameter:	None	
Return:	Status	Error/success code

3.3 void mmcSendCmd (const char cmd, unsigned long data, const char crc);

Send a command to the MMC card.

Parameter:	cmd	Command that should be sent to the MMC card
	data	Data for the command
	crc	Checksum for the command
Return:	None	

3.4 *char mmcGoIdle();*

Set the MMC card in idle mode to save current.

Parameter:	None	
Return:	Status	Error/success code

3.5 *char mmcSetBlockLength (const unsigned long);*

Set the MMC block length of count = 2^n byte. Normally this command is not required. The default block length is 512 bytes.

Parameter:	Block Length of count = 2^n Byte	
Return:	Status	Error/success code

3.6 *char mmcReadBlock(const unsigned long address, const unsigned long count, unsigned char *pBuffer);*

#define mmcReadSector(sector, pBuffer) mmcReadBlock(sector × 512, 512, pBuffer)

Read a size byte big block beginning at the address.

Parameter:	address	Start address of data to read on the card
	count	Number of bytes to read
	pBuffer	Pointer to read buffer
Return:	Status	Error/success code

3.7 *char mmcWriteBlock (const unsigned long address, const unsigned long count, unsigned char *pBuffer);*

#define mmcWriteSector(sector, pBuffer) mmcWriteBlock(sector × 512, 512, pBuffer).

Write a 512 byte big block beginning at the (aligned) address

Parameter:	address	Start address of data to write on the card
	count	Number of bytes to write
	pBuffer	Pointer to write buffer
Return:	Status	Error/success code

3.8 char mmcReadRegister (const char cmd_register, const unsigned char length, unsigned char *pBuffer);

Read the Register arg1 with Length arg2 (into the buffer).

Parameter:	cmd_register	Register to read
	length	Number of bytes to read
	pBuffer	Pointer to read buffer
Return:	Status	Error/success code

3.9 unsigned long mmcReadCardSize(void);

Read the card size from the CSD Register.

Parameter:	None
Return:	Detected card size

4 References

1. *MSP430x169 Mixed Signal Microcontroller* Data Sheet (SLAS368)
2. *MSP430x1xx Family User's Guide* (SLAU049)
3. *SanDisk MultiMediaCard Product Manual* (SanDisk, 2001)

IMPORTANT NOTICE