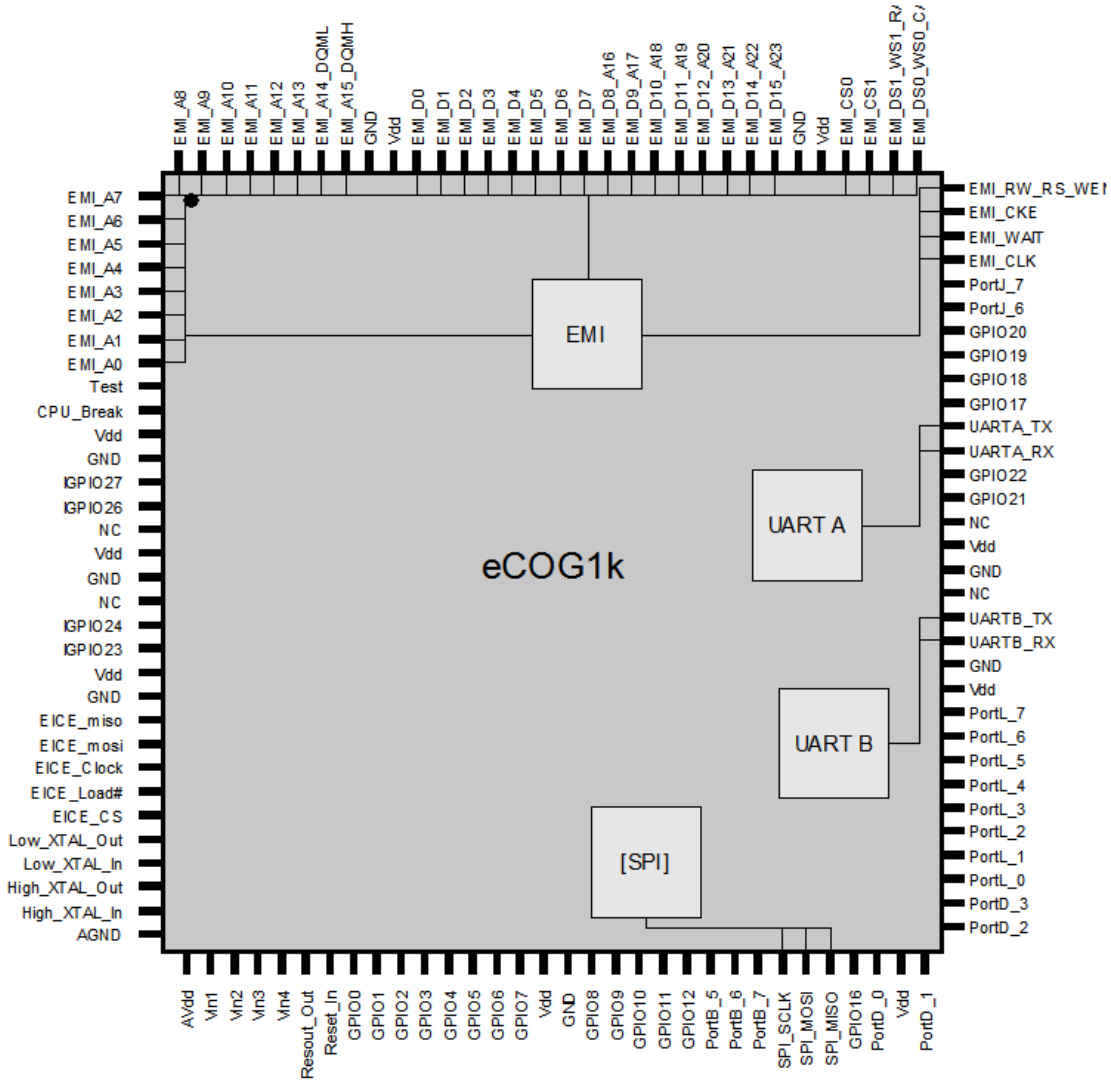
AN037 – Interfacing to an MMC or SD Card via SPI

Version 1.0

This application note describes an implementation of the multi media card (MMC) interface to the eCOG1k microcontroller, using the SPI bus protocol.



Confidential and Proprietary Information

Revision History

Version	Date	Notes
V1.0	22/03/2006	Initial Version

Contents

1 Introduction 5
2 Glossary 5
3 MMC and SD Card overview 5
4 eCOG1k Hardware 6
5 The MMC/SD Card Library on the eCOG1k 7
6 Example Software 7
6.1 Configuration 7
6.2 Example Software Operation 7
7 References 8
Appendix A Application Programming Interface 9
void mmcInit(void); *9*
unsigned int mmcReset(void); *9*
unsigned int mmcSendCommand(unsigned int cmd, unsigned long arg); *9*
*unsigned int mmcRead(unsigned long sector, unsigned char *buffer);* *10*
*unsigned int mmcWrite(unsigned long sector, unsigned char *buffer);* *10*

1 Introduction

This application note describes an implementation of the multi media card (MMC) interface to the eCOG1k microcontroller using the SPI bus protocol. The application software is based on public domain software that is covered by the GNU public licence agreement[1]. This updated version can be used to successfully mount and initialise the cards, as well as performing buffered sector reading and writing. While the software is predominantly designed to support MMC cards, the secure digital (SD) card is also supported in backward-compatibility mode.

2 Glossary

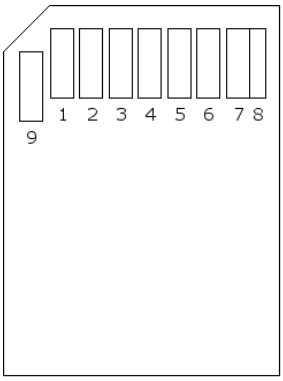
A table of abbreviations used in this document.

eCOG1	Cyan Technology target micro controller
MMC	Multi-Media Card
SD	Secure Digital
SPI	Serial Peripheral Interface
UART	Universal Asynchronous Receiver/Transmitter

3 MMC and SD Card overview

The MMC card and the SD card are flash memory storage based devices, and are physically very similar. Both card types support proprietary data transfer protocols using four data bits, and are compatible though having different initialisation. The major difference is that the SD card is designed to provide optional security by allowing encryption of the device contents. The MMC card supports additional bus widths (up to 8 bits). The SD card also supports several modes that are not present in the MMC card, including SDIO (secure digital input/output) that can be used as an external communications interface using the standard SD card format.

Both card types also support a basic SPI type interface for simple connection to embedded devices. The SD card specifications state a maximum clock frequency of 25MHz, and the MMC specifications state a maximum clock frequency of 52MHz depending on the device. Figure 1 shows the MMC/SD card connections when configured for SPI mode operation.



Pin	SPI Function
1	Chip select
2	MOSI
3	GND
4	VCC
5	SCLK
6	GND
7	MISO
8	NC (SD only)
9	NC (SD only)

Figure 1. MMC/SD Card Physical Connections in SPI mode.

The diagram shows the MMC/SD card when looking at the conductors (from the underside), and the table describes the connection signals.

1 Procyon AVRlib software, http://hubbard.engr.scu.edu/embedded/avr/avrlib/

4 eCOG1k Hardware

This application note is based on the use of an MMC/SD card interface daughter board, connected to the eCOG1k evaluation board. The daughter board contains a card socket, as well as the necessary connections for monitoring and control. Table 1 shows the required eCOG1k port configuration, Figure 2 shows the schematic of the daughter board, including eCOG1k specific connections, and Table 2 describes the physical connections to the daughter board.

The GPIO signals on port A (GPIO0-GPIO2) are routed to the evaluation board LED's to give an active display of software operation. GPIO signals on port J (GPIO17-GPIO18) provide hardware signaling to indicate the status of the MMC/SD card socket. If GPIO is used for interrupt control then the user should provide debouncing for mechanical detection mechanisms. Port K3 (GPIO22) is routed to the CS signal of the MMC/SD card to provide a chip select signal for SPI communication.

Port	A	B	C	D	E	F	G	H	I	J	K	L
Configuration	11	0	3	1	0	0	0	0	0	3	3	2

Table 1. eCOG1k Port Configuration

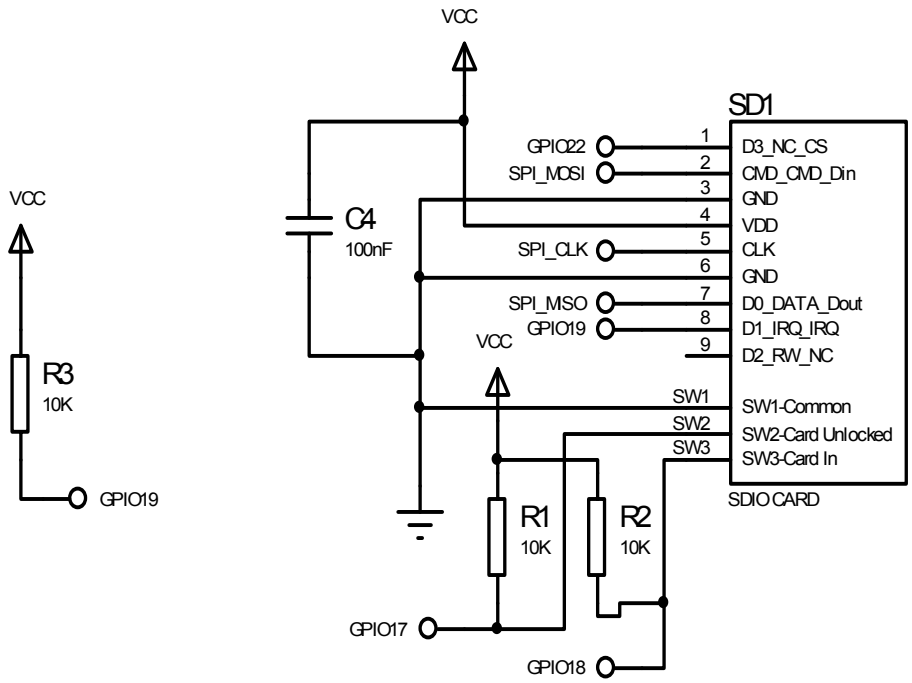


Figure 2. Connection Schematic for the MMC/SD card socket.

Note SW1, SW2 and SW3 provide hardware signals for the current status of the MMC/SD card socket. See Table 1 and Table 2 for a description of the connections.

Port	Pin	Operation	Function
A0	40	GPIO0	LED0, active low, indicates card in
A1	41	GPIO1	LED1, active low, indicates card locked
A2	42	GPIO2	LED2, active low, indicates SPI access
C0	58	SPI_SCLK	SPI clock
C1	59	SPI_MOSI	SPI data out
C2	60	SPI_MISO	SPI data in
J0	85	UARTA_RX	UART receive
J1	86	UARTA_TX	UART transmit
J2	87	GPIO17	Indicates status of manual card write lock
J3	88	GPIO18	Indicates status of card in signal
K3	84	GPIO22	SPI chip select signal[2]

Table 2. eCOG1k Physical Connections

5 The MMC/SD Card Library on the eCOG1k

This application is designed as a multi-media card (MMC) interface, and has been designed to also correctly support the secure digital (SD) card. The software was originally developed for use with MMC cards, and required additional software to correctly operate on SD cards. The SD card SPI protocol requires additional clock cycles (with chip select inactive) to be supplied after each command, to allow the device to complete the command execution. This is not implemented in the original software and has been updated. The remaining software is functionally the same as the original, apart from additional software for the control of evaluation board LEDs for status information, and interrupt-controlled card detection.

6 Example Software

6.1 Configuration

Terminal software is required for operation. The PC serial port should be connected to UART A on the evaluation board, and configured to use 115,200 baud, 8 data bits, no parity, one stop bit and no flow control. Once the software is downloaded to the eCOG1k, the initialisation routine is started by inserting an MMC/SD card or by using the menu system.

6.2 Example Software Operation

An example application is included that uses the MMC/SD hardware/software interface to allow monitoring of and access to a connected device. Basic sector reading and writing is provided, and a menu system is provided via UART A.

The interface to the MMC/SD card firmware is via UART A of the eCOG1k. A basic menu system is provided to allow the user to initialise a card and to perform sector reading and writing. Command format is a single character per command and Table 3 lists the available commands.

2 SPI interface limitation requires GPIO control of the SPI CS signal.

Command	Function
i	Reset and initialise MMC/SD card for operation
r	Read the current sector to the buffer, and display the results
w	Write the buffer to the current sector
+	Increment the current sector number by 1
-	Decrement the current sector number by 1
*****	Increment the current sector number by 512
/	Decrement the current sector number by 512

Table 3. Example software commands

The current status of the MMC/SD card socket is interrupt controlled. An interrupt is generated whenever a rising edge is detected on the card in signal, and the interrupt service routine sets LED0 and LED1 to indicate (respectively) whether a card has been inserted and whether the physical write protect lock has been enabled. When a card is inserted and detected, a global flag is set that instructs the firmware to attempt to initialise the card. If the initialisation routine fails, then LED0 is cleared to indicate that the firmware has not detected a valid MMC/SD card. Upon successful initialization, LED0 remains active.

7 References

1. System Summary 4.1: A short version of the MMCA System Specification 4.1. http://www.mmca.org/compliance/buy_spec/MMCA_System_SummaryV41.pdf
2. SD Card Physical Layer Specification: Simplified version V1.01. http://www.sdcard.org/sdio/Simplified Physical Layer Specification.PDF

Appendix A Application Programming Interface

`void mmcInit(void);`

Initialises the MMC/SD SPI card interface, and enables the required signals.

Example

```
mmcInit();
```

`unsigned int mmcReset(void);`

Resets the MMC/SD card interface, and configures the device for operation.

Configures the connected MMC/SD card to use SPI mode, turns off CRC checking and sets the block length to 512 bytes.

Returns

0 if configuration was successful and the card is operational.
–1 if the configuration was unsuccessful.

Example

```
if (!mmcReset())
{
   printf("Found SD Memory Card\n\r");
}
else
{
   printf("\n\rProblem Configuring Card!!\n\r");
}
```

`unsigned int mmcSendCommand(unsigned int cmd, unsigned long arg);`

Sends a command to the MMC/SD card.

Parameters

cmd	A one byte command
arg	The arguments for the command

Returns

The command response from the MMC/SD card.

Example

```
int i;
r1 = mmcCommand(MMC_READ_SINGLE_BLOCK, sector << 9);  // Start block read
for (i = 0; i < 512; i++)
{
   // Read in 512 bytes
   buffer[i] = spiTransferByte(0xff);
}
```

`unsigned int mmcRead(unsigned long sector, unsigned char *buffer);`

Read a block of data from the MMC/SD card into a buffer.

Parameters

sector	The MMC/SD block to read
**buffer*	Character pointer to the buffer used to store the read data

Returns

0 if the read was successful, otherwise returns the response from the MMC/SD card.

Example

```
mmcRead(sector, buffer);      // Read the sector to the buffer
```

`unsigned int mmcWrite(unsigned long sector, unsigned char *buffer);`

Write the buffer data to the MMC/SD card block.

Parameters

sector	The MMC/SD block to read
**buffer*	Character pointer to the buffer used to store the data to write

Returns

0 if the write was successful, otherwise returns the response from the MMC/SD card.

Example

```
mmcWrite(sector, buffer);     // Write the buffer to the specified sector
```