

SD Card Reader Using the M9S08JM60 Series

Designer Reference Manual

Devices Supported:

HCS08

Microcontrollers

Document Number: DRM104

Rev. 0

07/2008



How to Reach Us:

Home Page:

www.freescale.com

Web Support:

<http://www.freescale.com/support>

USA/Europe or Locations Not Listed:

Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:

Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:

Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:

Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or +1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals", must be validated for each customer application by customer's technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.



Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2008. All rights reserved.

DRM104
Rev. 0
07/2008

Chapter 1 Overview

1.1	Targets	1
1.2	Birds-Eye View of an SD/MMC Card Reader	1
1.3	Features	3

Chapter 2 Hardware Description

2.1	System Architecture	5
2.2	MC9S08JM60 Microcontroller	5
	2.2.1 MC9S08JM60 Series Features	6
	2.2.2 USB Module Features	6
	2.2.3 SPI Module Features	6
2.3	Secure Digital (SD) Card	7
	2.3.1 Targets and Functions	7
	2.3.2 Interface	7
	2.3.3 Access	8

Chapter 3 Firmware

3.1	Overview	9
	3.1.1 Architecture	9
	3.1.2 System Startup Flowchart	9
	3.1.3 Normal User Application Code	10
3.2	USB Driver	10
	3.2.1 Targets	10
	3.2.2 USB Endpoints Used	11
	3.2.3 USB Enumeration	11
	3.2.4 Driver Requirement	11
	3.2.5 USB Descriptors	11
	3.2.6 Bulk-Only Transport Protocol	13
	3.2.7 Bulk-Only Software Flowchart (SD Card Reader Working Process)	14
3.3	SCSI Handler	15
	3.3.1 Functions	15
	3.3.2 SCSI Commands Used	15
3.4	SD Card Driver	16
	3.4.1 SPI Mode	16
	3.4.2 SD Card in SPI Mode Protocol	17
	3.4.3 SD Card Initialization in SPI Mode	18
	3.4.4 SD Card I/O Functions	18
	3.4.5 SPI Driver	19

Chapter 4

Operational Description

4.1	Procedure	21
4.1.1	Target	21
4.1.2	Preparation	21
4.1.3	Procedure	21

Appendix A

SD Card Driver Codes

Appendix B

Bill of Materials

Appendix C

Schematics of SD Card Reader

Chapter 1 Overview

1.1 Targets

In this document we provide an SD card reader solution that uses the MC9S08JM60 microcontroller.

This reference design serves as an example of a USB mass-storage device using the Freescale Flexis JM USB microcontroller family. The MC9S08JM60 acts as a bridge between a PC USB host and an SD card.

All hardware schematic diagrams and firmware source codes are available in reference materials.

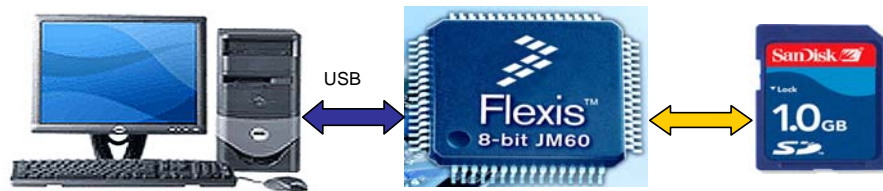


Figure 1-1. System Introduction

1.2 Birds-Eye View of an SD/MMC Card Reader

Figure 1-2 is the top view of the SD/MMC card reader and Figure 1-3 is the bottom view.



Figure 1-2. Top View of SD/MMC Card Reader



Figure 1-3. Bottom View of SD/MMC Card Reader

1.3 Features

Features of this SD/MMC card reader:

- Small and easy to use
- Plug-and-play functionality
- USB 2.0 full-speed capability
- Free drivers for Linux, Windows 2000, Windows XP, and later versions of Windows
- SD card insert detection and write protection
- USB access for reading and writing
- USB bus power

Chapter 2 Hardware Description

2.1 System Architecture

The SD card reader is controlled by the MC9S08JM60 microcontroller. This system consists of three main parts:

- MC9S08JM60 MCU
- SD card socket
- Voltage regulator

Figure 2-1 shows the block diagram. The whole system operates at 3.3 V. Because the SD card supports only 3.3 V operating voltage but the board is supplied with 5 V through the USB port, a 5 V to 3.3 V regulator is needed.

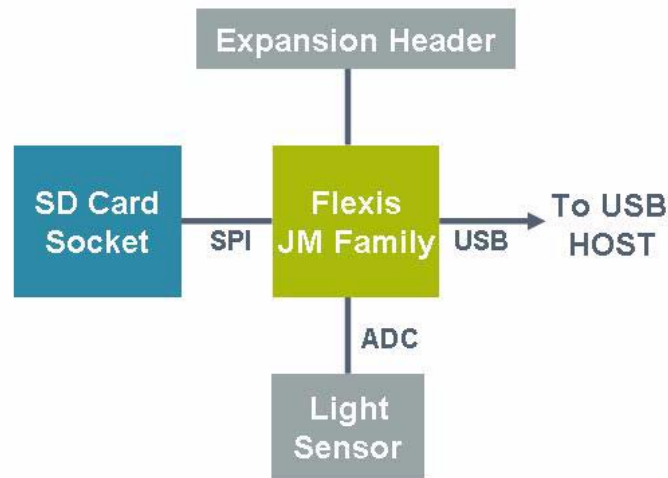


Figure 2-1. SD Card Reader Block Diagram

2.2 MC9S08JM60 Microcontroller

The MC9S08JM60 is a low-cost, high-performance HCS08 architecture 8-bit microcontroller with a USB module.

2.2.1 MC9S08JM60 Series Features

The main features of the MC9S08JM60 series MCUs are:

- High performance HCS08 core with up to 48 MHz system clock
- Up to 60 KB flash and 4 KB SRAM
- Integrated full-speed USB 2.0 module
- Two SPIs, full-duplex or single-wire bidirectional, with master or slave mode functionality
- 12-channel, 12-bit-resolution ADC with built-in temperature sensor
- Two 16-bit timers — one 6-channel and one 2-channel
- Two SCIs with LIN functionality
- One 8-bit RTC
- Up to 51 GPIOs
- Supply voltage: 2.7 V to 5.5 V
- Operating temperature range: $-40\text{ }^{\circ}\text{C}$ to $85\text{ }^{\circ}\text{C}$

The MC9S08JM60 is a member of the Flexis JM USB family of microcontroller devices. The Flexis JM family consists of 8-bit to 32-bit software-compatible and pin-compatible USB-enabled devices. It is easy for a user to migrate quickly from the MC9S08JM60 platform to other Flexis JM devices, such as the MC9S08JM32 or MCF51JM128.

2.2.2 USB Module Features

The main characteristics of the USB module are:

- USB 2.0 full-speed (12 Mbps)
- On-chip 3.3 V regulator
- Internal pullup resistor
- 256 bytes dual-port RAM
- Seven endpoints (EP0-EP6)
 - EP0: bidirectional
 - EP1–6: directional (in or out)
- Control, interrupt, bulk, and isochronous transfer types supported

This USB module is suitable for many types of USB devices, such as HID class, CDC class, and mass-storage class.

2.2.3 SPI Module Features

The main characteristics of the SPI module are:

- Master or slave mode functionality
- Full-duplex and single-wire bidirectional mode
- Programmable baud rate

- Double-buffered transmit-and-receive data register
- Serial clock phase and polarity options
- Slave output selection
- Mode fault error flag with CPU interrupt capability
- Selectable MSB-first or LSB-first shifting
- Programmable 8-bit or 16-bit data transmission length

The SPI module is used as the interface between the MC9S08JM60 and the SD card.

2.3 Secure Digital (SD) Card

2.3.1 Targets and Functions

The secure digital (SD) card standard was introduced by Toshiba, Matsushita Electric, and SanDisk in 1999. According to the *SanDisk SD Card Product Manual*, the SD card is “a flash-based memory card specifically designed to meet the security, capacity, performance, and environmental requirements inherent in audio and video consumer electronic devices.” Encryption for protected content is a key part of its design, thus ensuring that copyrighted material is distributed securely.

2.3.2 Interface

The SD card communication is based on an advanced nine-pin interface (clock, command, 4 × data and 3 × power lines) designed to operate in a low voltage range, as shown in [Figure 2-2](#).

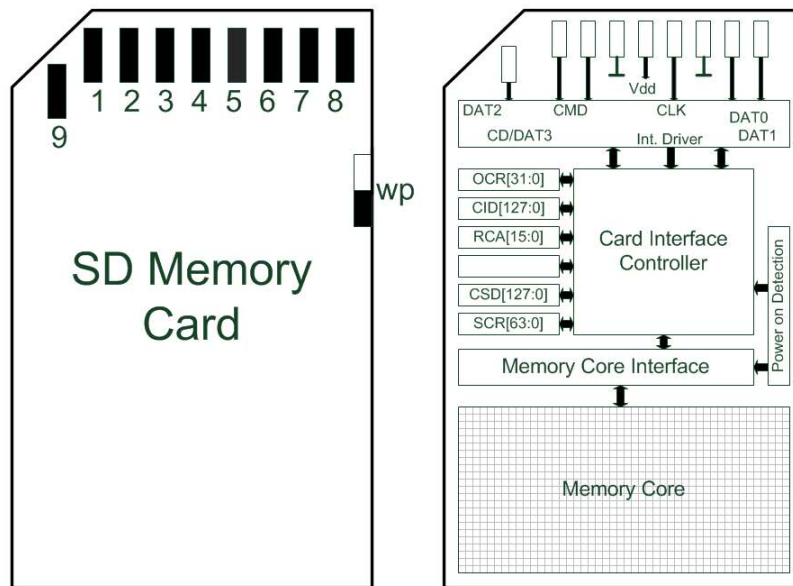


Figure 2-2. SD Card Interface

2.3.3 Access

The SD card can be accessed via the SPI-based MultiMediaCard (MMC) protocol or its own SD card protocol (the fastest, four bits parallel). [Table 2-1](#) shows the SPI mode pad definition.

Table 2-1. SPI Mode Pad Definition

Pin No.	Name	Type	SPI Description
1	\overline{CS}	I	Chip Select (Active Low)
2	DataIn	I	Host-to-Card Commands and Data
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	DataOut	O	Card-to-Host Data and Status
8	RSV ₍₂₎	I	Reserved
9	RSV ₍₂₎	I	Reserved

Chapter 3 Firmware

3.1 Overview

3.1.1 Architecture

The firmware includes two parts:

- USB bootloader code
- Normal user code

3.1.2 System Startup Flowchart

Figure 3-1 shows the system startup flowchart. After reset, a piece of determination code is run to determine whether the system enters bootloader mode or user mode.

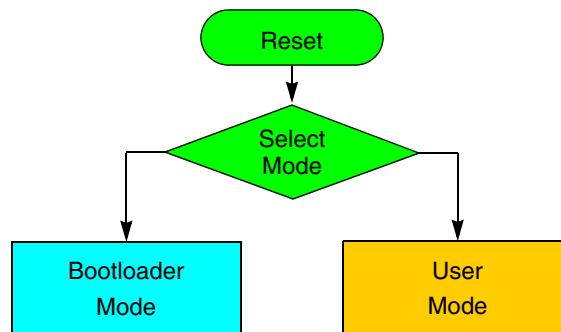


Figure 3-1. System Startup Flowchart

The bootloader function is embedded in firmware that allows in-circuit programming. In bootloader mode, the user can update the code via the USB port. For details on the USB bootloader, please refer to Freescale application note AN3561, “USB Bootloader for the MC9S08JM60.”

User mode is the normal mode. In this mode, the system jumps to the user application code. There the system will run the SD card reader application code.

One GPIO (PTB5) is used in this application to determine which mode the system enters.

3.1.3 Normal User Application Code

The normal user application code can be divided into five parts, as shown in [Figure 3-2](#).

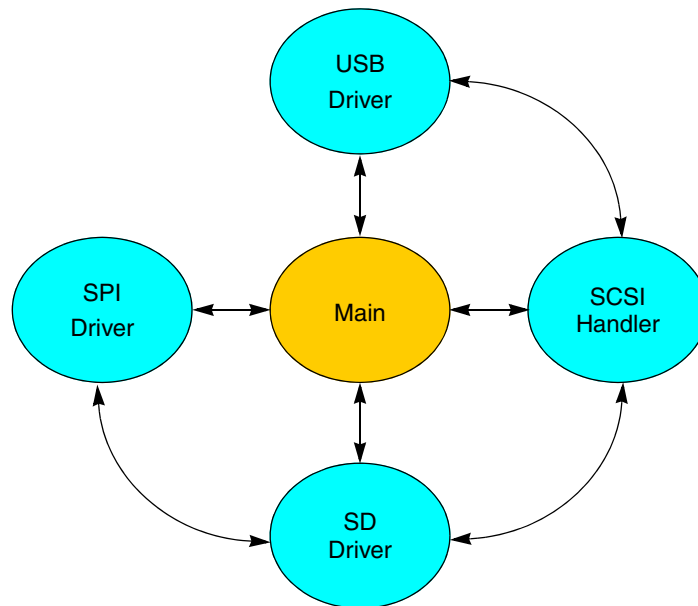


Figure 3-2. Firmware Chart

- **Main** — The main function takes care of system initialization and system management. It initializes the MCU, the SPI, the SD card, and USB. Then it manages the whole system and calls the SCSI handler if there is a command from the USB host.
- **USB driver** — The USB driver is based on the Freescale USB-MINI Stack. It is adapted for a mass-storage application. It manages USB enumeration at the beginning, then decodes the commands from the USB host and responds with corresponding data.
- **SCSI handler** — The SCSI command set is used in this application. The SCSI handler decodes SCSI commands from the host and acts according to the commands.
- **SD card driver** — The SD card driver is composed of SD card initialization, the SD command processor, and block read/write functions.
- **SPI driver** — It controls SPI module reading and writing when communicating with the SD card.

3.2 USB Driver

3.2.1 Targets

The Freescale USB-MINI Stack is highly optimized and ideal for HID-class and other general applications. Because a mass-storage-class device must be able to manage high-volume data, the USB-MINI Stack is modified to make it suitable for this application.

3.2.2 USB Endpoints Used

Three USB endpoints are used in this application:

- EP0
- EP1
- EP2

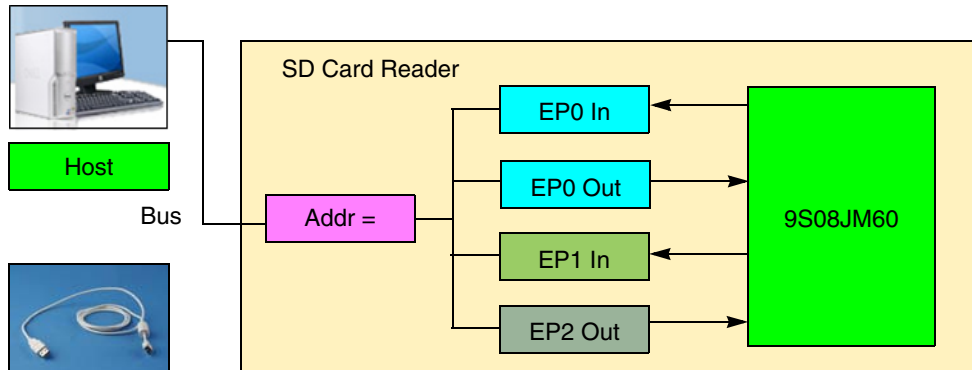


Figure 3-3. USB Endpoint

EP0 is the control transfer endpoint for USB enumeration and standard command operations.

EP1 is configured as the bulk transfer endpoint for the in token.

EP2 is configured as the bulk transfer endpoint for the out token.

The configuration for EP1 and EP2 is processed during USB enumeration.

3.2.3 USB Enumeration

USB enumeration is a process for the host to identify the USB device. The process has two phases.

In phase 1, the host learns about the newly arrived device by reading the descriptors, then loads the appropriate device driver.

In phase 2, the device driver configures the device and makes it ready for data transfer.

3.2.4 Driver Requirement

Microsoft Windows 2000 and above has a mass-storage device driver, so there is no driver requirement from the host.

3.2.5 USB Descriptors

The device must provide descriptors to the host.

According to the USB specification, the standard descriptors include:

- Device descriptors
- Configuration descriptors

- Interface descriptors
- Endpoint descriptors

3.2.5.1 Device Descriptor

The device descriptor gives an overview of this USB device, with information such as vendor ID and product ID. The vendor ID for this SD card reader is 0x15A2 (designating Freescale) and the product ID is 0x0037.

3.2.5.2 Configuration Descriptor

The configuration descriptor explains the number of interfaces for this device. If it is bus-powered, the device power consumption is also provided.

3.2.5.3 Interface Descriptor

The interface descriptor provides the class code and subclass code for this device. [Table 3-1](#) shows the interface descriptor of the SD card reader.

Table 3-1. Interface Descriptor of the SD Card Reader

Offset	Field	Size	Value	Description
0	bLength	1	0x09	Size of descriptor in bytes
1	bDescriptorType	1	0x04	Interface descriptor type
2	bInterfaceNum	1	0x00	Number of interface
3	bAlternateSetting	1	0x00	Value used to select alternative setting
4	bNumEndpoints	1	0x02	Number of endpoints used for this interface
5	bInterfaceClass	1	0x08	Class code (mass storage)
6	bInterfaceSubClass	1	0x06	Subclass code (SCSI transparent command set)
7	bInterfaceProtocol	1	0x50	Protocol code (bulk-only transport)
8	iInterface	1	0x00	Index of string descriptor describing this interface

The class code for mass storage is 0x08. Mass storage has six subclasses, such as reduced block commands, SFF-8020i, QIC-157, and the SCSI transparent command set. In this application, the SCSI transparent command set is used, so the subclass code is 0x06.

The SCSI transparent command set has two main protocols:

- Control/bulk/interrupt protocol
- Bulk-only transport protocol

Bulk-only is the most commonly used protocol, and is selected with a protocol code of 0x50.

3.2.5.4 Endpoint Descriptor

The endpoint descriptor describes:

- endpoint number
- endpoint type — in or out
- endpoint attribute — bulk
- maximum packet size

3.2.6 Bulk-Only Transport Protocol

In bulk-only transport protocol, there is no dedicated endpoint for command and status. All data, commands, and status values are transferred by EP1 or EP2. The command is wrapped in the command block wrapper (CBW) while status is wrapped in the command status wrapper (CSW).

3.2.6.1 Bulk-Only Transport Flow

Figure 3-4 shows the bulk-only (command/data/status) transport flowchart.

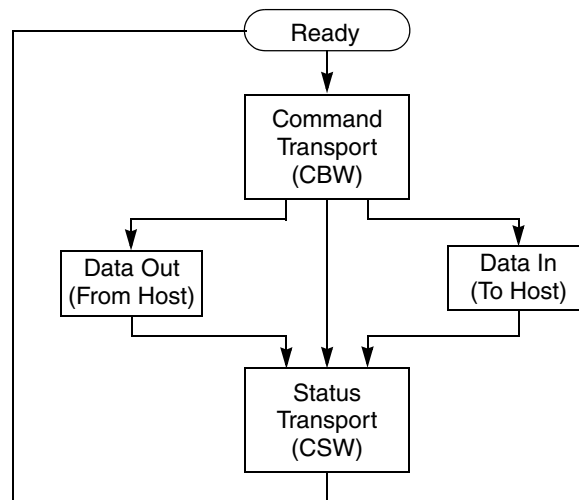


Figure 3-4. Bulk-Only Transport Flowchart

The USB host sends a command by the CBW to the device via the out endpoint (EP2 in this application). The device decodes the CBW to identify which command the host has sent, then manages this command. If there is more data that needs to be transferred between host and device, the data will be transferred via the in endpoint (for in token) or the out endpoint (for out token). After it has finished processing the command, the device sends a status value by the CSW back to the host via the in endpoint.

3.2.6.2 Command Block Wrapper (CBW)

The command block wrapper is a packet containing a command block and associated information, as shown in Figure 3-5. The CBW starts on a packet boundary and ends as a short packet with exactly 31

(0x1F) bytes transferred. All CBW transfers shall be ordered with the LSB (byte 0) first (little endian). The signature field contains the value 0x43425355 (little endian), indicating a CBW.

Byte	Bit							
	7	6	5	4	3	2	1	0
0–3	dCBWSignature = 0x43425355 (Little Endian)							
4–7	dCBWTag							
8–11	dCBWDataTransferLength							
12	bmCBWFlags							
13	Reserved				bCBWLun			
14	Reserved			BCBWCBLength				
15–30	CBWCB							

Figure 3-5. Command Block Wrapper

3.2.6.3 Command Status Wrapper (CSW)

The command status wrapper is a packet that contains the status of a command block, as shown in Figure 3-6. The CSW starts on a packet boundary and ends as a short packet with exactly 13 (0x0D) bytes transferred. All CSW transfers must be ordered with the LSB (byte 0) first (little endian). The signature field contains the value 0x53425355h (little endian), indicating that this is the CSW.

Byte	Bit							
	7	6	5	4	3	2	1	0
0–3	dCSWSignature = 0x53425355 (Little Endian)							
4–7	dCSWTag							
8–11	dCSWDataResidue							
12	bCSWStatus							

Figure 3-6. Command Status Wrapper (CSW)

3.2.7 Bulk-Only Software Flowchart (SD Card Reader Working Process)

The SD card reader software flowchart illustrates the bulk-only protocol. After enumeration is successful, the USB driver waits for commands from the host. If the command is a CBW, the USB driver will pass the CBW to SCSI_Handler to process. At the end of the process, one CSW will be sent to the host to provide the status. According to the SCSI command type, the DATA IN/OUT will be transferred if needed.

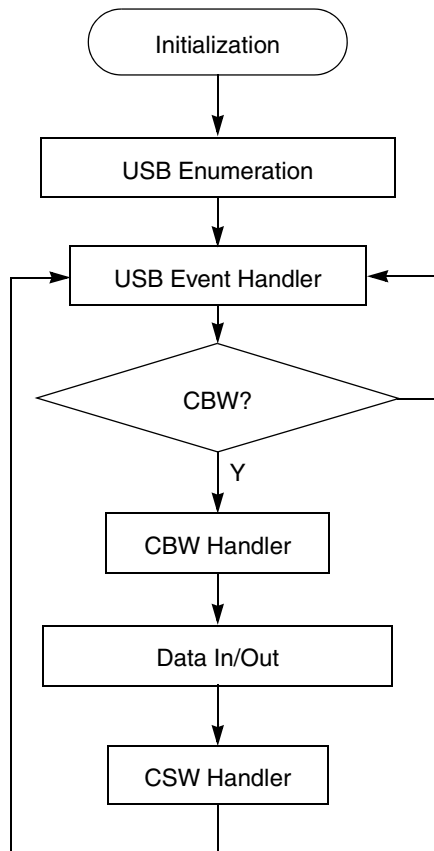


Figure 3-7. Bulk-Only Software Flowchart

3.3 SCSI Handler

3.3.1 Functions

The SCSI primary command set 2 (SPC-2) is designed to provide efficient peer-to-peer operation of SCSI devices (disks, tapes, printers, etc.) by an operating system.

3.3.2 SCSI Commands Used

The SCSI command set has around 30 commands for all types of devices, but only seven commands are used in the SD card reader application. [Table 3-2](#) shows these commands.

Table 3-2. SCSI commands for SD Card Reader

Command Name	Opcode	Description
Test Unit Ready	0x00	Check to see if device is ready
Request Sense	0x03	Request sense data
Inquiry	0x12	Request information about the target

Table 3-2. SCSI commands for SD Card Reader (continued)

Command Name	Opcode	Description
Mode Sense(6)	0x1A	Report selected information about the device, such as write protection
Read Capacity	0x25	Request current capacity of the mass storage device
Read (10)	0x28	Allow host to read data from the device
Write (10)	0x2A	Allow host to write data to the device

- Test unit ready command allows the host to poll whether the device is ready.
- Request sense command requests that the device transfer data to the host.
- Inquiry command allows the host to request additional information about the device or about the SCSI command device support.
- Mode sense command provides a means for the device to report parameters to the host, such as page mode or write protection.
- Read capacity command retrieves the capacity of the device.
- Read command allows the host to read data from the device.
- Write command allows the host to write data to the device.

3.4 SD Card Driver

SD cards are one of the most popular flash-based removable storage devices. Low cost, small size, and low power consumption make the SD card the favorite storage for devices in the consumer market.

There are two ways to manage the information in the SD card: using the SD protocol or using a simple SPI module.

This design reference manual shows how to interface an SD card with a low-end microcontroller using the SPI protocol, because almost all Freescale low-end microcontrollers include at least one SPI module.

3.4.1 SPI Mode

The behavior of SD cards in SPI mode is basically the same as for any SPI slave device.

The maximum transfer rate of the SD card in SPI mode is 25 Mbps, but in the initialization process the transfer rate must be less than 375 kbps. This is because the SPI mode of the SD cards is compatible with the MMC cards, and MMC cards can only reach 375 kbps. After initialization, the SPI clock can be changed to 25 Mbps.

Figure 3-8 shows how to interface the SD card pins with the SPI module of the MCU.

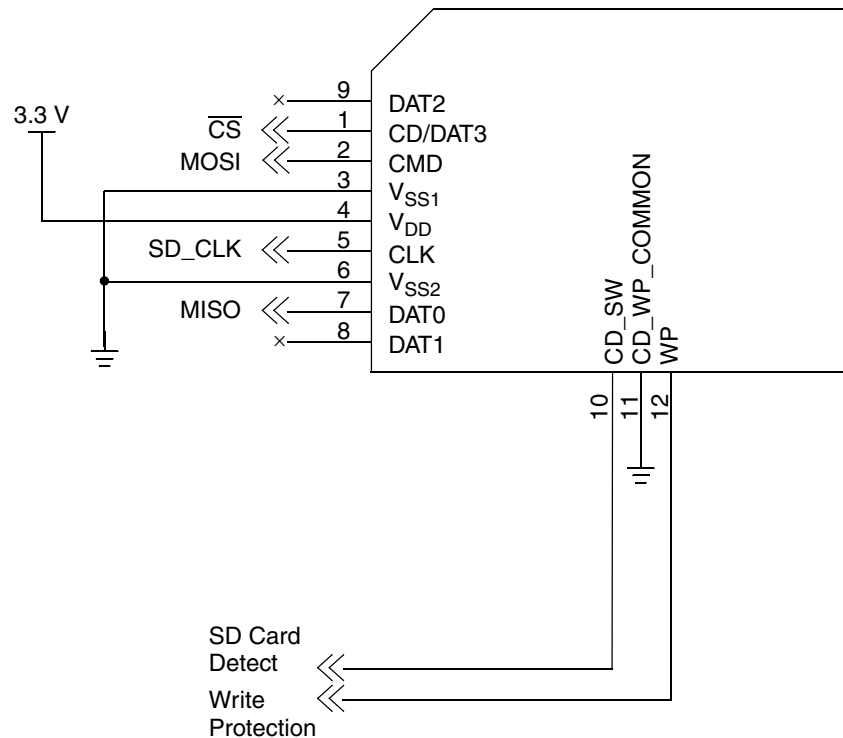


Figure 3-8. SD Card Connection in SPI Mode

3.4.1.1 Signal Description

- MISO: Master input, slave output
- MOSI: Master output, slave input
- CLK: SPI clock
- \overline{SS} : Slave select
- WP: Write protect
- SDI: SD card insertion (if an SD card has been inserted in the socket)

3.4.2 SD Card in SPI Mode Protocol

The SD SPI protocol is basically a simple command-response protocol. All commands are initiated by the master (the MCU in this case) and the SD responds with a response frame, followed by a master token indicating that the SD card is ready to send/receive data frames.

3.4.3 SD Card Initialization in SPI Mode

SD cards are very similar to MMC cards. Basically the main differences from a software point of view are the initialization sequence and the access speed. An SD card requires a specific initialization sequence to enter SPI mode.

1. Set the SPI clock to 375 kbps. This is required for compatibility across a wide range of SD and MMC cards.
2. Provide at least 75 SPI clock cycles with the \overline{SS} signal asserted to ensure that the SD card internal state machine is initialized.
3. Send 16 SPI clocks (\overline{SS} unasserted).
4. Send reset command to restart the SD card in SPI mode.
5. The card is continuously polled with the initialize and block-length commands until the idle bit becomes clear, indicating that the card is fully initialized and ready to respond to general commands.
6. The SPI clock is set to the maximum supported by the MCU and allowed by the SD card.

3.4.4 SD Card I/O Functions

SD cards are divided into physical data blocks; the access functions can read or write only one entire block at a time. That is, even if the application needs read/write access only to a single byte, the entire block must be accessed for the byte operation.

The user's access to the SD card driver is through a simple API that is composed of four call functions: read, write, CSD, and CID. These call functions allow users to read/write blocks and retrieve the SD card information needed for file system management (managed by the USB host operating system in this case).

3.4.4.1 Read Function

The block read command is a bulk data command. The command response is followed by a delay, then a start-of-block token, and finally the actual block itself.

The function `SD_Read_Block` reads a single block of the SD card. Also this function returns an error code in case the read function was not executed properly (see appendix A, "[SD Card Driver Codes](#)," for error codes).

```
UINT8 SD_Read_Block(UINT32 u32SD_Block,UINT8 *pu8DataPointer)
U32SD_Block: indicate what block needs to be read
pu8DataPointer: base pointer to store the data from SD card
```

3.4.4.2 Write Function

The `SD_Write_Block` function allows the user to write on a single SD card block. This function returns an error code in case the write operation fails (see appendix A, "[SD Card Driver Codes](#)," for error codes).

```
UINT8 SD_Write_Block(UINT32 u32SD_Block,UINT8 *pu8DataPointer)
U32SD_Block: indicate what block needs to be written
pu8DataPointer: base pointer of data to be stored in SD card
```

3.4.4.3 CSD Function

SD_ReadCSD function reads the internal CSD register of the SD card. This CSD register contains the storage information of the SD card. This CSD information will be used by the file system of the OS.

```
UINT8 SD_ReadCSD(void)
```

3.4.4.4 CID Function

The CSD register of the SD card contains the basic information of the card, such as version, manufacturer, serial number, etc.

```
UINT8 SD_GetCID(void)
```

3.4.5 SPI Driver

The SPI driver is the only one that has direct contact with the hardware (SPI module); basically the SPI driver API contains simple byte read/write functions and SPI module initialization.

Chapter 4 Operational Description

4.1 Procedure

4.1.1 Target

This procedure explains how to use the SD card reader to manage the SD card.

The SD card reader reference design is basically a simple SD card reader similar to the kind that can be found in any electronics store.

4.1.2 Preparation

If you want to do a write operation or a remove operation, set the write-protect switch to the on position. Otherwise, the only operation you can perform is to read the data in the SD card.

4.1.3 Procedure

Use these steps to process the SD card.

1. Insert the SD card in the SD card reader board.

NOTE

Insert the card in the board before connecting the card reader to the computer.

2. Connect the SD card reader to a Windows or Linux-based computer.
3. Read or write files in the SD card reader folder created by the operating system.

NOTE

If the write operation fails, check the SD card write protection.

4. Disconnect the SD card reader using the safe removal procedure (the exact procedure varies depending on the OS used).

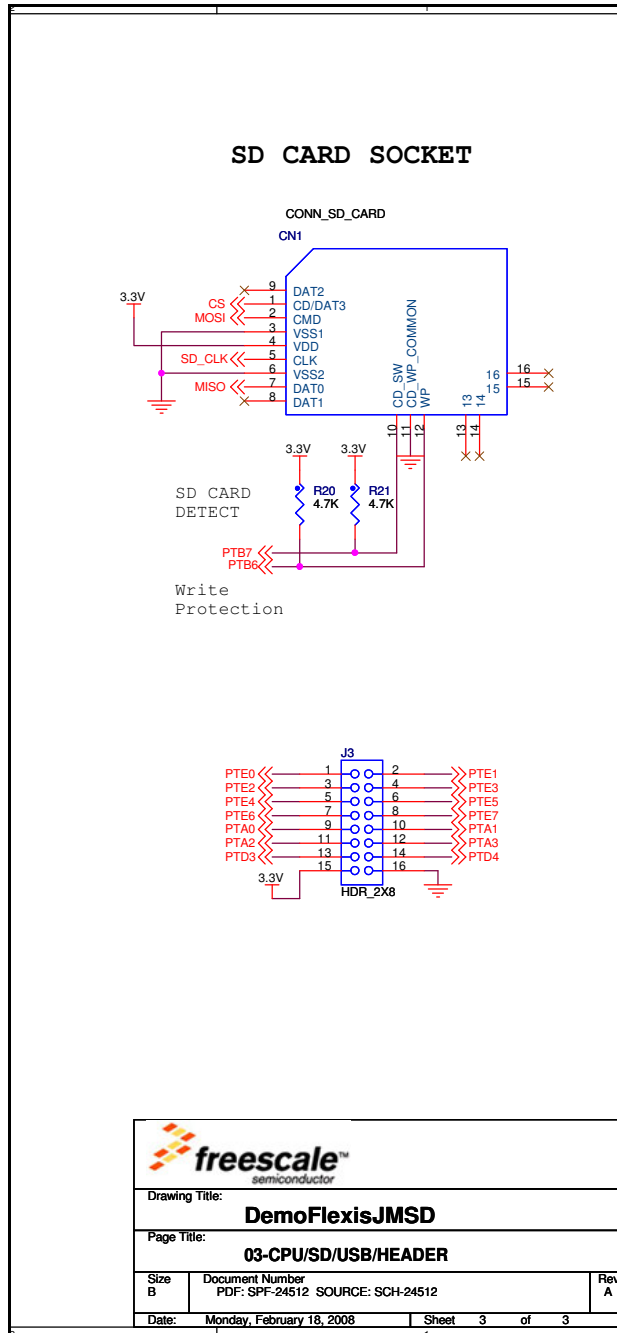
Appendix A SD Card Driver Codes

0x00	OK (operation completed successfully)
0x01	COMMAND_FAILS
0x02	INIT_FAILS
0x03	WRITE_COMMAND_FAILS
0x04	WRITE_DATA_FAILS
0x05	READ_COMMAND_FAILS
0x06	READ_DATA_FAILS
0x07	NO_SD_CARD

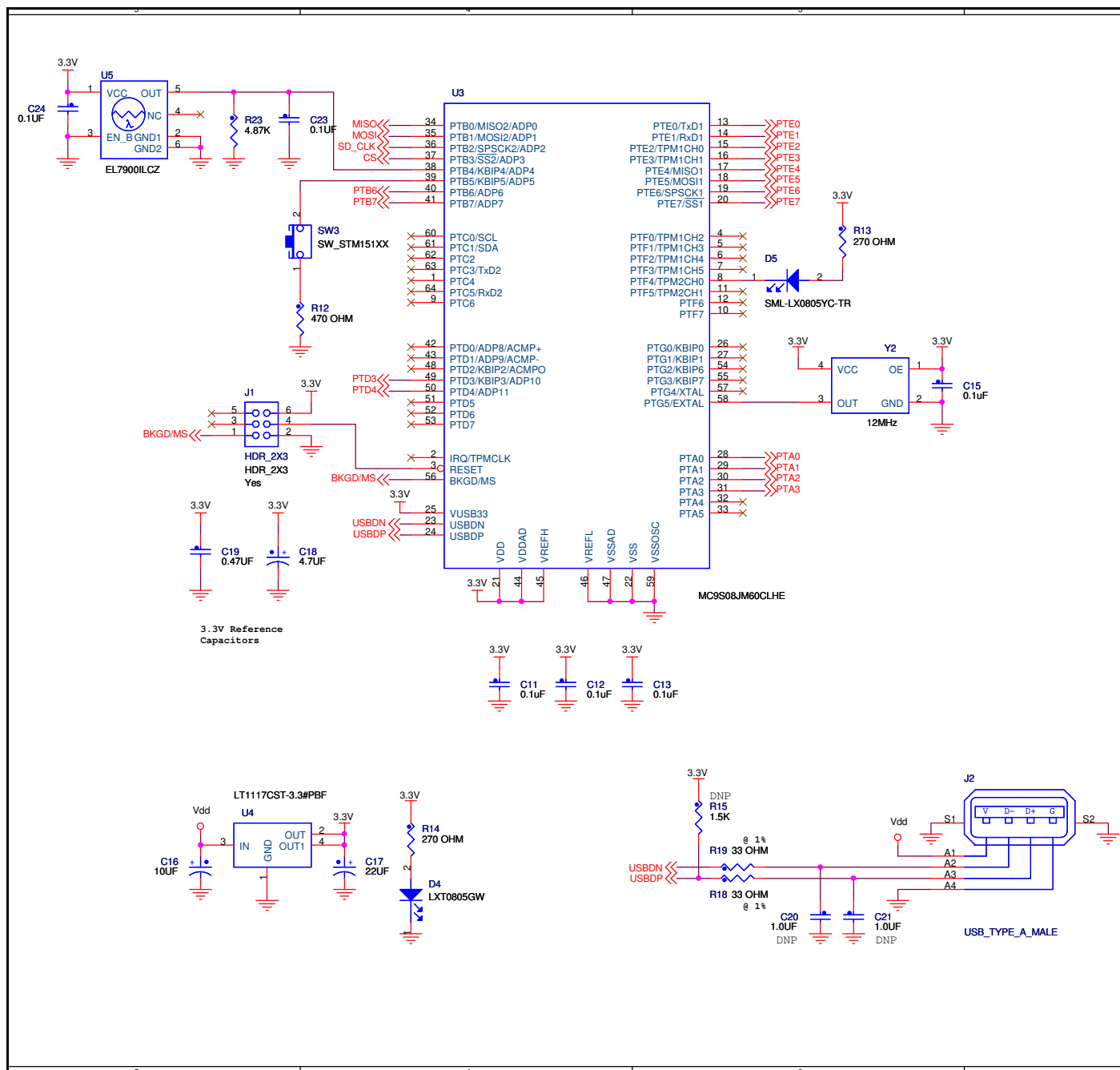
Appendix B Bill of Materials

Component ID	Value	Manufacturer	Mfg. Part Number
C11–C13, C15	0.1 μ F	SMEC	MCCE104J2NRTF
C16	10 μ F	Vishay Intertechnology	293D106X9010A2TE3
C17	22 μ F	AVX	TAJA226K010R
C18	4.7 μ F	AVX	TACL475K010R
C19	0.47 μ F	Venkel Company	C0805X7R250474KNE
C23, C24	0.1 μ F	Murata	GRM188R71E104KA01D
CN1	CONN_SD_CARD	Molex	0676000004
D4	LXT0805GW	Citizen	CL-170G-CD-T
D5	SML-LX0805YC-TR	Lumex	SML-LX0805YC-TR
J1	HDR_2X3	Samtec	TSW-103-07-S-D
J2	USB_TYPE_A_MALE	Samtec	USB-AM-S-S-B-SM1
J3	HDR_2X8	Samtec	TSW-108-07-G-D
R12	470 Ω	Bourns	CR0805-JW-471ELF
R13, R14	270 Ω	SMEC	RC73L2D271JTF
R18, R19	33 Ω	Vishay Intertechnology	CRCW080533R0FKEA
R20, R21	4.7 k Ω	Venkel Company	CR0805-8W-472JT
R23	4.87 k Ω	Koa Speer	RK73H1JT4871F
SW3	SW_STM151XX	Silcomp Sp. Z O.o.	STM151HS
U3	MC9S08JM60CLHE	Freescale Semiconductor	MC9S08JM60CLHE
U4	LT1117CST-3.3#PBF	Linear Technology	LT1117CST-3.3#PBF
U5	EL7900ILCZ	Intersil	EL7900ILCZ
Y2	12 MHz	Jauch Quartz GMBH	O12.0-VX3JQ-LF

Appendix C Schematics of SD Card Reader



Schematics of SD Card Reader



SD Card Reader Using the M9S08JM60 Series, Rev. 0