TB3174

How to Add USB Mass Storage Device (MSD) Functionality Using the MPLAB Harmony Configurator (MHC)

Introduction

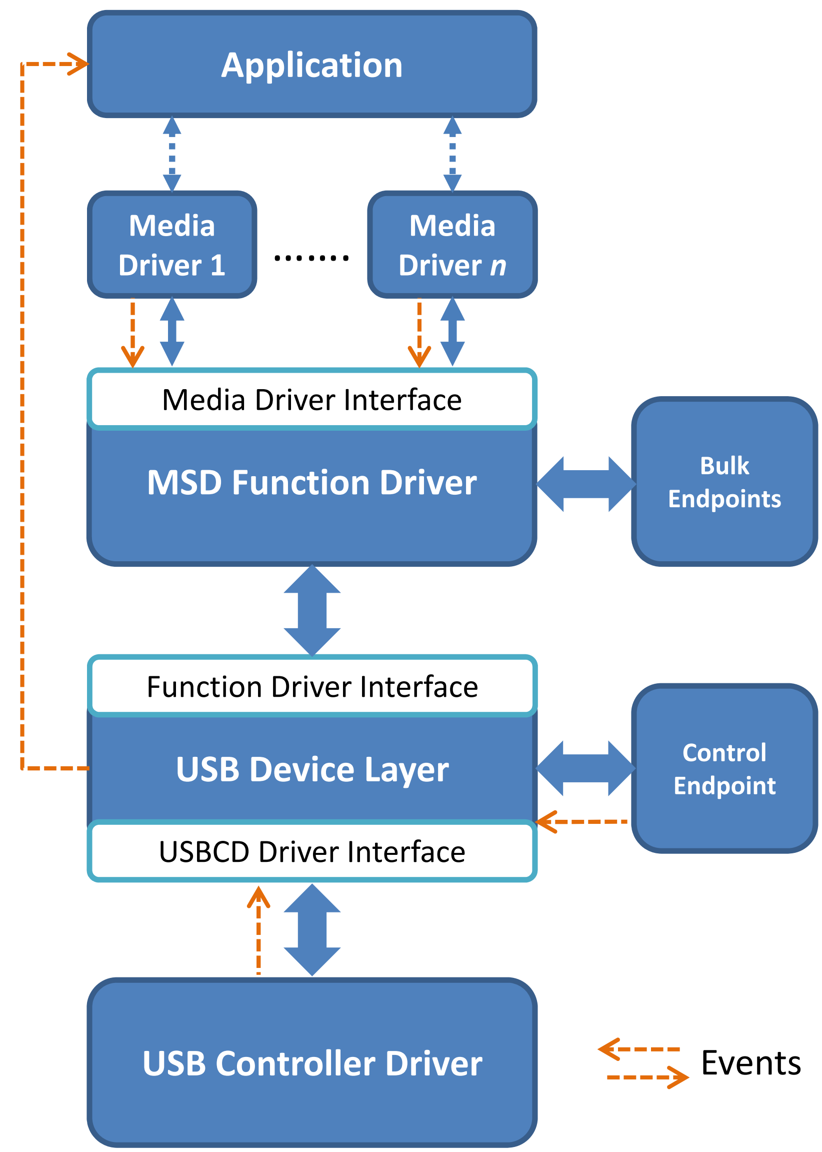
The Universal Serial Bus (USB) protocol is widely used to interface storage devices to a USB Host computer. Any device that allows access to its internal storage using the Mass Storage Class protocol can be connected as a Mass Storage Device (MSD) to the Host computer using USB interface. This document briefly discusses the different components of the MPLAB® Harmony USB Mass Storage Device (MSD) stack. This is followed by an example to demonstrate the addition of the USB MSD functionality to an existing MPLAB Harmony project.

Table of Contents

1. USB MSD Library Components

The USB Device Library consists of the following three major components, as shown in Figure 1-1.

Figure 1-1. USB Mass Storage Device Library Components



- USB Controller Driver (USBCD)
- USB Device Layer
- Function Driver

USB Controller Driver (USBCD)

- Manages the USB peripheral
- Provides access to the USB peripheral to the USB Device Layer by implementing the USBCD Driver Interface
- Provides the Device Layer with the USB events

USB Device Layer

- Opens the USB Controller Driver.
- Registers an event handler with the USBCD to receive events from the USBCD
- Registers an event handler to handle transmit and receive complete events from the Control Endpoint
- Responds to enumeration requests issued by the USB Host
- Handles Standard Device and Endpoint requests
- Calls the application registered event handler to notify the USB Device Layer events such as, `USB_DEVICE_EVENT_CONFIGURED`, `USB_DEVICE_EVENT_POWER_DETECTED`, `USB_DEVICE_EVENT_POWER_REMOVED` to the application
- Initializes and runs the state machine of the MSD Function Driver

MSD Function Driver

- Implements the USB Device Mass Storage class functionality
- Exposes its functionality to the USB Device Layer by implementing the Function Driver interface
- Handles the Standard and Class specific Interface requests
- Interacts with the Media through the Media Driver Interface to process the data read and write requests it receives from the USB Host. Handles Media Driver events.

Application

- The application neither have to interact with the MSD Function Driver nor the MSD Function Driver provide any application callable functions.
- The application must open the USB Device Layer and handle the Device Layer events to attach/detach the USB device.
- It is possible that the application may also open the Media Driver while they are already opened by the MSD Function Driver. If the application and the MSD Function Driver try to write to the same media, the result could be unpredictable. It is recommended that the application restricts write access to the media while the USB Device is plugged into the Host.

2. Adding USB MSD Functionality Using the MHC

Using the MHC, add USB MSD functionality to an existing MPLAB Harmony project. The example shown here uses the PIC32MZ Embedded Connectivity with Floating Point Unit (EF) Starter Kit and the Multimedia Expansion Board II (MEB II) with SD Card as the media. MPLAB X IDE v3.61 and MPLAB Harmony v2.04 were used for this example.

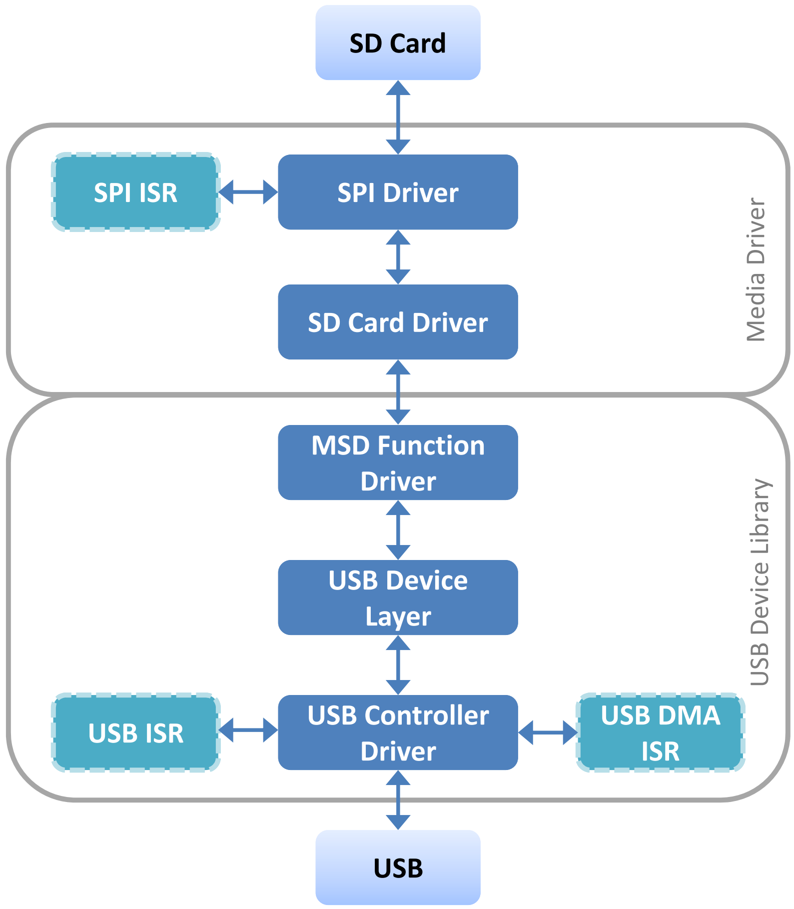
The SD Card driver uses the SPI driver to communicate with the SD Card. Figure 2-1 shows the interaction between various drivers.

Note: Other media like the NVM (internal Flash) and the SPI Flash are also supported in the MHC. Users can also develop their own media drivers and plug into the USB MSD Function Driver by implementing the Media Driver Interface specified by the MSD Function Driver.

Adding USB MSD functionality to an existing MPLAB Harmony application mainly consists of the following steps:

1. Using the MHC, configure the USB stack for MSD functionality.
2. Using the MHC, configure the media.
   2.1. Configure the SD Card Driver.
   2.2. Configure the SPI Driver.
   2.3. Configure the I/O pins used by the SPI Driver.
3. Generate Code.
4. Add application code.

Figure 2-1. Interaction Between Various Drivers



Step 1: Using the MHC, Configure USB Stack for MSD Functionality (see Figure 2-2 and Figure 2-3).

1. Open the MHC in your existing project by selecting *Tools > Embedded*.
2. Go to *Options > Harmony Framework Configuration > USB Library* and then select the *Use USB Stack?* option.
3. The *Interrupt Mode* is already selected. The USB driver state machine will be run from the interrupt context.
4. Expand *Select Host or Device Stack*. The *USB Device* stack is already selected.
5. Set the *Number of Endpoints Used* to 2. The USB MSD uses bulk-only transport (BOT) protocol. One endpoint is the control endpoint (EP0) used for control requests and the second endpoint is the bulk endpoint (bulk IN and bulk OUT) used for data transfer between the USB Host and the device.
6. Retain the *Endpoint 0 Buffer Size* to 64. For High-Speed devices, the EP0 size is fixed to 64. For Full-Speed devices, the EP0 size can be 8, 16, 32, or 64 bytes.
7. Expand the *USB Device Instance 0*, which is selected by default.
8. Keep the Device Speed to the default - USB_SPEED_HIGH. PIC32MZ devices support both Full-Speed and High-Speed operation. Selecting High-Speed will allow the device to work at both Full-Speed and High-Speed.
9. Retain the default value of 1 for the *Number of Functions Registered to this Device Instance* as there is only the MSD Function Driver registered to the USB Device Instance.
10. Expand the *Function 1* which is already selected. Configure the *Function 1* for USB MSD operation.

11. Set the *Device Class* to MSD.
12. Retain the *Configuration Value* of 1. The MSD Function Driver is tied to configuration value 1. The USB device task will run the state machine for the MSD Function Driver when it receives the SET Configuration control command with the configuration value set to 1 from the USB Host.
13. Retain the value 0 for the *Start Interface Number*. This indicates that interface 0 is managed by the MSD Function Driver. This will allow the standard and class specific requests for interface 0 to be forwarded to the MSD Function Driver.
14. The *Speed* member specifies the Device speeds for which this Function driver should be initialized. This can be set to USB_SPEED_FULL, USB_SPEED_HIGH or a logical OR combination of both. The Device Layer will initialize the function if the devices' attach speed matches the speed mentioned in the Speed member of the entry. To allow for both High-Speed and Full-Speed operation, set it to USB_SPEED_HIGH|USB_SPEED_FULL.
15. Retain the value 1 for the *Bulk Endpoint Number*. This indicates that Endpoint 1 will be used for Bulk IN and Bulk OUT transfers.
16. Retain the value 1 for *Max number of sectors to buffer*. This will set aside a buffer of size 512 x 1 bytes. This value may be changed to allow buffering of data read from the media which helps in increasing the overall throughput at the expense of increased RAM size.
17. Retain the value 1 for *Number of Logical Units*.
18. *LUN0* (Logical Unit 0) is already selected. Expand it, and set the *Media Type* to SDCARD.
19. The *Product ID Selection* is set to msd_basic_sdcard_demo. This sets the Product ID (PID) to 0x0009.
20. Retain the default values of *Enter Vendor ID*, *Enter Product ID*, *Manufacturer String* and *Product String*.
21. Retain the default priorities for the USB interrupt and the USB DMA interrupts.

Figure 2-2. USB Stack Configuration

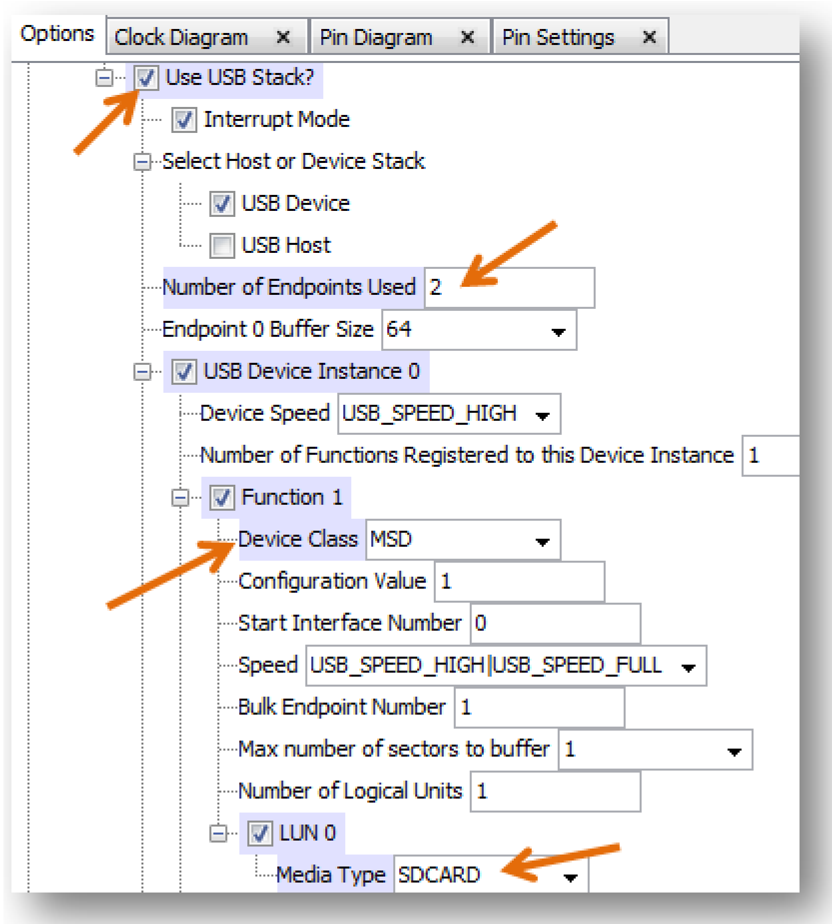
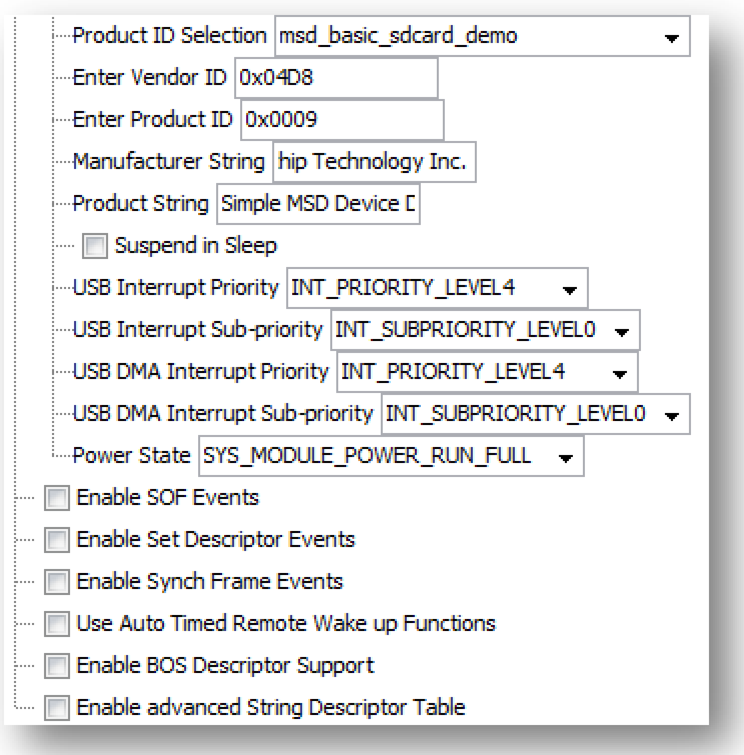


Figure 2-3. USB Stack Configuration (Continued)

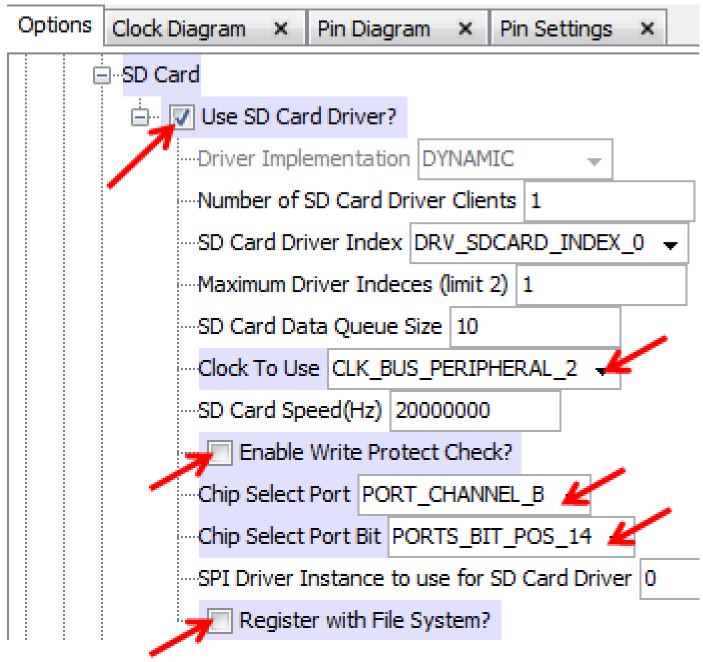


Step 2: Using MHC, Configure the Media

1. Configure the SD Card Driver, see Figure 2-4.
   - 1.1. Expand *Options* > *Harmony Framework Configuration* > *Drivers* > *SD Card*. The *Use SD Card Driver?* is already selected as the LUN0 in the USB device stack is configured to use the SD Card media.
   - 1.2. Retain the value 1 for *Number of SD Card Driver Clients* as the SD Card Driver is used only by the USB Function Driver.
   - 1.3. Retain the value DRV_SDCARD_INDEX_0 to index into the (only) SD Card driver instance.
   - 1.4. Retain the value 1 for *Maximum Driver Indeces (limit 2)* as only one instance of SD Card driver is needed.
   - 1.5. Retain the value 10 for the *SD Card Data Queue Size*.
   - 1.6. *Clock To Use* specifies the clock source for the SPI peripheral used by the SD Card. Set it to CLK_BUS_PERIPHERAL_2.
   - 1.7. *SD Card Speed (Hz)* specifies the communication speed of the SD Card. Retain the default value of 20 MHz. This value must be less than the maximum SPI frequency and must be supported by the SD Card used.
   - 1.8. Micro SD Cards do not have the write protection line. Clear the *Enable Write Protect Check?* option.
   - 1.9. The chip select line of the SD Card (DAT[3]/CD) is connected to the RB14 port pin of the PIC32 device. Set the *Chip Select Port* and *Chip Select Port Bit* to PORT_CHANNEL_B and PORT_BIT_POS_14, respectively.
   - 1.10. Clear the *Register with File System?* option. The USB MSD Function Driver accesses the SD Card directly without a file system. The data on an SD Card is organized as a file system where first several blocks of data provide information about the file system type and

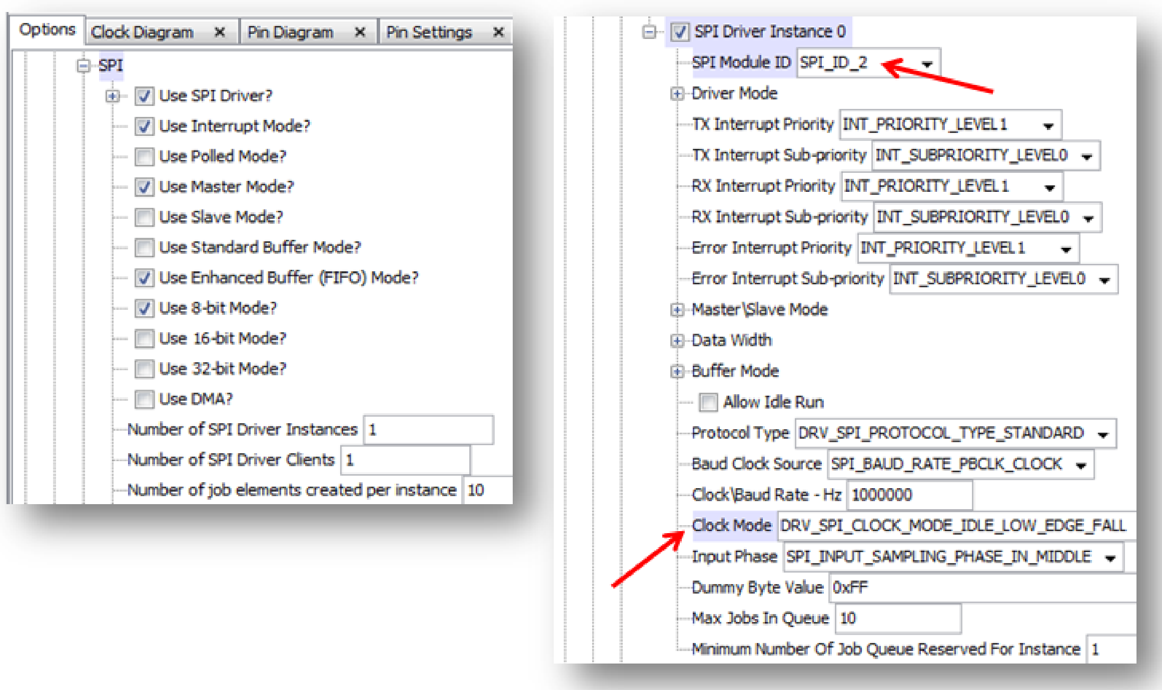
how the data is organized. The USB Host reads these blocks of data to identify the file system on the SD Card and mounts a suitable file system to access (read/write) the SD Card.

Figure 2-4. SD Card Driver Configuration



2. Configure the SPI Driver, see Figure 2-5.
   - 2.1. Expand *Options* > *Harmony Framework Configuration* > *Drivers* > *SPI*. The *Use SPI Driver?* is already selected as the SD Card is configured to use the SPI driver. The SPI driver is configured to use dynamic implementation of the driver in interrupt mode.
   - 2.2. Expand the *SPI Driver Instance 0* and change the SPI Module ID to SPI_ID_2 as the SD Card is interfaced to SPI2. The *SPI Driver Instance 0* is already configured as SPI Master and for Interrupt mode operation.
   - 2.3. SD Card supports SPI Mode 0 (i.e., Clock Polarity = 0, Clock Phase = 0). Set the *Clock Mode* to DRV_SPI_CLOCK_MODE_IDLE_LOW_EDGE_FALL.
   - 2.4. Accept the default values for all the other configurations for the SPI driver.

Figure 2-5. SPI Driver Configuration

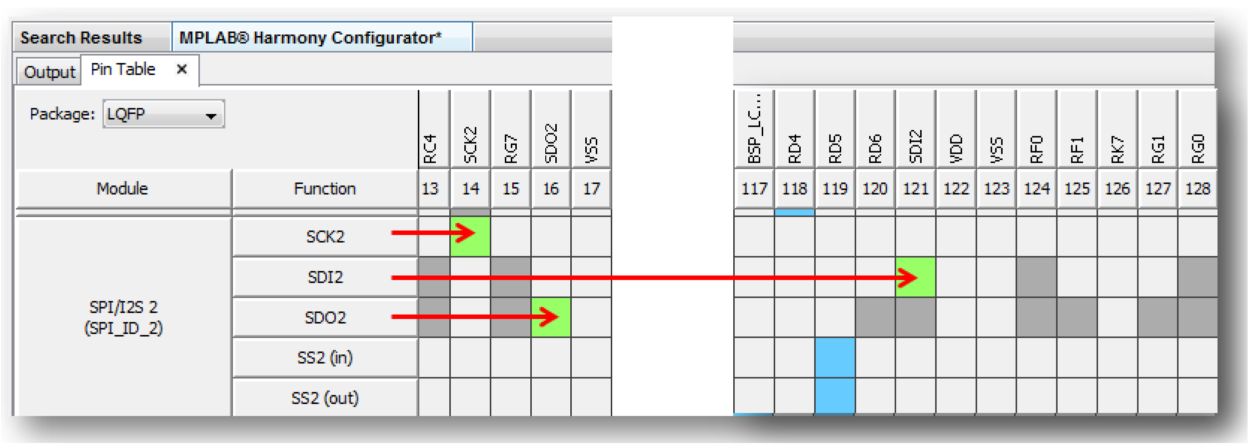


3. Configure the I/O pins used by the SPI Driver.
   Click the Pin Table and configure the I/O pins used by the SPI_ID_2, as shown in Figure 2-6. Refer to the *PIC32MZ Embedded Connectivity with Floating Point Unit (EF) Starter Kit User's Guide* (DS70005230) for details on pin mapping. This document is available for download from the Microchip web site at: www.microchip.com.

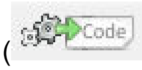
Table 2-1. SPI Driver Pin Configuration

SD Card Pin	SPI Mode Function	Pin Mapping
CLK	SCK2	RG6 (pin 14)
CMD	SDI2	RD7 (pin 121)
DATA0	SDO2	RG8 (pin 16)

Figure 2-6. SPI I/O Pins Mapping



Step 3: Generate Code

Save the configuration and click **Generate Code** ().

When the MHC generates code, it adds the USB framework files to the project, as shown in Figure 2-7. Figure 2-8 shows the `SYS_Tasks()` routine that runs the SD Card Driver task, USB Driver task, and the USB Device task routines. The `system_interrupts.c` file contains the interrupt handlers for the SPI, USB, and the USB DMA.

Figure 2-7. Project Files and Folder Structure

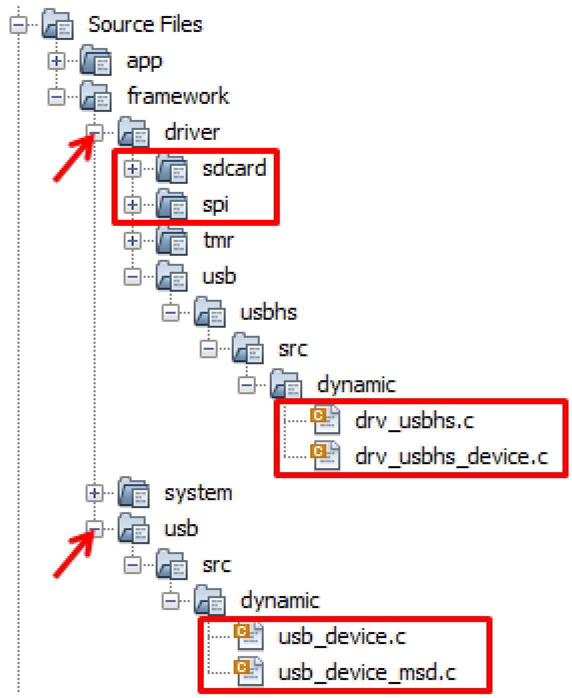
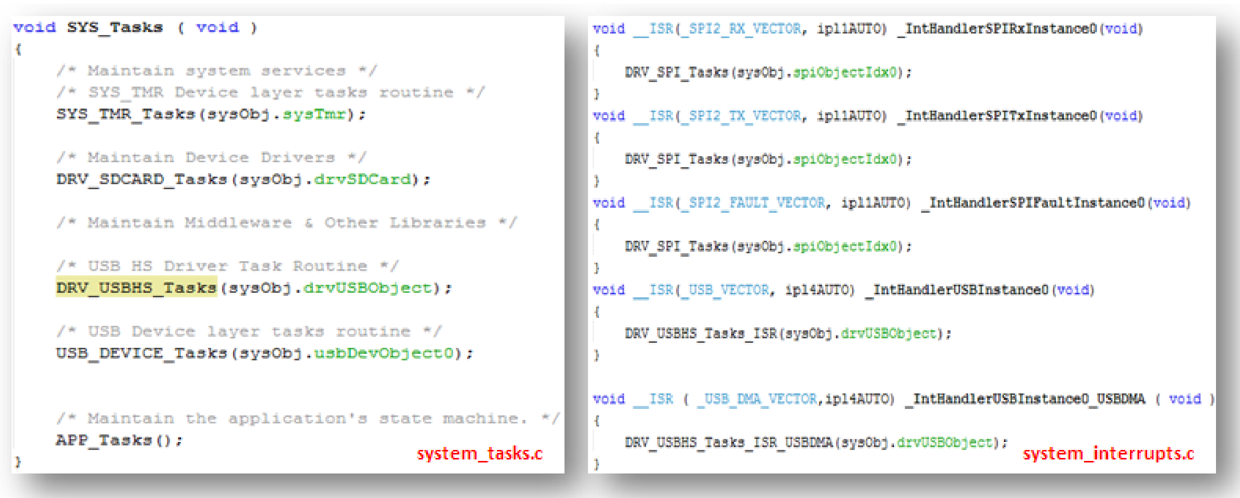


Figure 2-8. Generated Code in system_tasks.c and system_interrupts.c Files



```
void SYS_Tasks ( void )
{
    /* Maintain system services */
    /* SYS_TMR Device layer tasks routine */
    SYS_TMR_Tasks(sysObj.sysTmr);

    /* Maintain Device Drivers */
    DRV_SDCARD_Tasks(sysObj.drvSDCard);

    /* Maintain Middleware & Other Libraries */

    /* USB HS Driver Task Routine */
    DRV_USBHS_Tasks(sysObj.drvUSBObject);

    /* USB Device layer tasks routine */
    USB_DEVICE_Tasks(sysObj.usbDevObject0);


    /* Maintain the application's state machine. */
    APP_Tasks();
}
```

system_tasks.c

```
void __ISR(_SPI2_RX_VECTOR, ipl1AUTO) _IntHandlerSPIRxInstance0(void)
{
    DRV_SPI_Tasks(sysObj.spiObjectIdx0);
}
void __ISR(_SPI2_TX_VECTOR, ipl1AUTO) _IntHandlerSPITxInstance0(void)
{
    DRV_SPI_Tasks(sysObj.spiObjectIdx0);
}
void __ISR(_SPI2_FAULT_VECTOR, ipl1AUTO) _IntHandlerSPIFaultInstance0(void)
{
    DRV_SPI_Tasks(sysObj.spiObjectIdx0);
}
void __ISR(_USB_VECTOR, ipl4AUTO) _IntHandlerUSBInstance0(void)
{
    DRV_USBHS_Tasks_ISR(sysObj.drvUSBObject);
}


void __ISR ( _USB_DMA_VECTOR,ipl4AUTO) _IntHandlerUSBInstance0_USBDMA ( void )
{
    DRV_USBHS_Tasks_ISR_USBDMA(sysObj.drvUSBObject);
}
```

system_interrupts.c

Step 4: Add Application Code

The application must first open the USB Device Layer, and then register an event handler with the USB Device Layer to handle the USB Device Layer events.

1. Open the USB Device Layer. Once a valid handle to the USB Device Layer is obtained, register an event handler to receive the USB Device Layer events, as shown in Figure 2-9.
2. Handle the USB Device layer events, and attach and detach the USB device within the USB_DEVICE_EVENT_POWER_DETECTED and the USB_DEVICE_EVENT_POWER_REMOVED events, as shown in Figure 2-9.

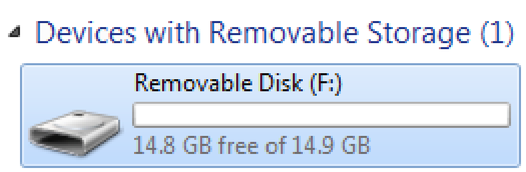
Figure 2-9. Application Code - Opening and Handling the USB Device Layer

```
appData.usbDevHandle = USB_DEVICE_Open(USB_DEVICE_INDEX_0, DRV_IO_INTENT_READWRITE);

if(appData.usbDevHandle != USB_DEVICE_HANDLE_INVALID)
{
    /* Set the Event Handler. We will start receiving events after
     * the handler is set */
    USB_DEVICE_EventHandlerSet(appData.usbDevHandle, APP_USBDeviceEventHandler, (uintptr_t)&appData);
```

```
void APP_USBDeviceEventHandler( USB_DEVICE_EVENT event, void * pEventData, uintptr_t context )
{
    /* This is an example of how the context parameter
       in the event handler can be used.*/

    APP_DATA* appData = (APP_DATA*)context;

    switch( event )
    {
        case USB_DEVICE_EVENT_RESET:
        case USB_DEVICE_EVENT_DECONFIGURED:

            /* Device was reset or deconfigured. Update LED status */
            BSP_LEDOn ( BSP_LED_1 );
            BSP_LEDOn ( BSP_LED_2 );
            BSP_LEDOn ( BSP_LED_3 );
            break;


        case USB_DEVICE_EVENT_POWER_DETECTED:

            /* VBUS is detected. Attach the device. */
            USB_DEVICE_Attach(appData->usbDevHandle);
            break;

        case USB_DEVICE_EVENT_POWER_REMOVED:

            /* VBUS is not detected. Detach the device */
            USB_DEVICE_Detach(appData->usbDevHandle);
            break;
```

Build and program the PIC32MZ EF Starter Kit connected to the MEB II. Insert a micro SD card in the SD Card slot J8 on the MEB II. Connect a Micro-B USB cable between the USB port J4 on the PIC32MZ EF Starter Kit and the PC.

Once enumerated, the device should appear as a mass storage device on the PC (USB Host).



3. References

For additional information on MHC and MPLAB Harmony, download the MPLAB Harmony Integrated Software Framework from the Microchip web site: http://www.microchip.com/mplab/mplab-harmony.

Detailed documentation on the USB Mass Storage Device is included in the installation of MPLAB Harmony, available in the following folder: `<harmony-install-path>/doc/`. PDF, Compiled Help (CHM), and HTML help file formats are available.

Demonstrations on USB Mass Storage Device are located within the installation of MPLAB Harmony within the following folder: `<harmony-install-path>/apps/usb/device`.

The Microchip Web Site

Microchip provides online support via our web site at http://www.microchip.com/. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQ), technical support requests, online discussion groups, Microchip consultant program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Customer Change Notification Service

Microchip's customer notification service helps keep customers current on Microchip products. Subscribers will receive e-mail notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, access the Microchip web site at http://www.microchip.com/. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

Technical support is available through the web site at: http://www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip’s code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Trademarks

ISBN: 978-1-5224-2347-8

Quality Management System Certified by DNV

ISO/TS 16949

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon

Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

China - Hong Kong SAR
Tel: 852-2943-5100
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-3326-8000
Fax: 86-21-3326-8021

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

ASIA/PACIFIC

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-3019-1500

Japan - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

Japan - Tokyo
Tel: 81-3-6880- 3770
Fax: 81-3-6880-3771

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

France - Saint Cloud
Tel: 33-1-30-60-70-00

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-67-3636

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-7289-7561

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820