CE224073 – SPI F-RAM Access Using PSoC 6 MCU SMIF in Memory Mapped (XIP) Mode

Objective

This code examples demonstrates how to access the SPI F-RAM™ using the PSoC® 6 MCU's Serial Memory Interface (SMIF) in command or memory mapped IO (MMIO) mode as well as in memory mapped (or XIP, eXecute-In-Place) mode.

Overview

This code example implements the SPI host controller on PSoC 6 MCU using the SMIF Component and demonstrates accessing the SPI F-RAM writes and reads using the memory mapped IO (MMIO) mode and memory mapped (or XIP, eXecute-In-Place) mode of the SMIF component. The code example results can be monitored through an UART terminal on the PC. This code example also demonstrates switching between the XIP to MMIO mode and vice versa to execute various memory and non-memory accesses of the SPI F-RAM device. Since F-RAM is a nonvolatile memory with SRAM like accesses (symmetrical write and read at bus speed), depending on the application use case, PSoC 6 firmware can enable F-RAM access either in MMIO or the XIP mode. The MMIO mode only access is discussed in detail in *CE222460 - SPI F-RAM Access Using PSoC 6 MCU SMIF.*

MMIO vs Memory Mapped (XIP)

In the command or MMIO mode, the SMIF access is supported through software writes to transmit (Tx) FIFO and software reads from receive (Rx) FIFO. The FIFOs are mapped on SMIF registers. This interface provides flexibility to implement all types of access to the external memory device. For example, memory, registers, and low power modes accesses can be executed in the MMIO mode. This mode is suitable for data logging applications with memory write/read in burst mode (access more than one byte) as well as other non-memory accesses such as entering low power mode, read device ID, read serial number, and so on.

In the memory mapped or XIP mode, the external memory (SPI F-RAM in this code example) is directly mapped to the CPU's internal memory space. Hence, any write to the external memory or read from the external memory is like any register writes and reads in the program. Data transfers from the host controller to the external memory device are automatically translated to the memory device writes and reads by the SMIF hardware. However, other non-memory accesses such as entering low power mode, read device ID, read serial number, and so on can't be executed in XIP mode. Hence, the PSoC 6 host must switch to MMIO mode in between XIP mode memory writes and reads to execute non-memory accesses. The memory mapped or XIP mode provides an efficient method for random memory writes and reads. Hence, the XIP mode is suitable for code execution, scratchpad buffer, and DMA transfer use cases.

Requirements

Tool: PSoC Creator™ 4.2; Peripheral Driver Library (PDL) 3.0.1

Programming Language: C (Arm® GCC 5.4-2016-q2-update, Arm MDK Generic)

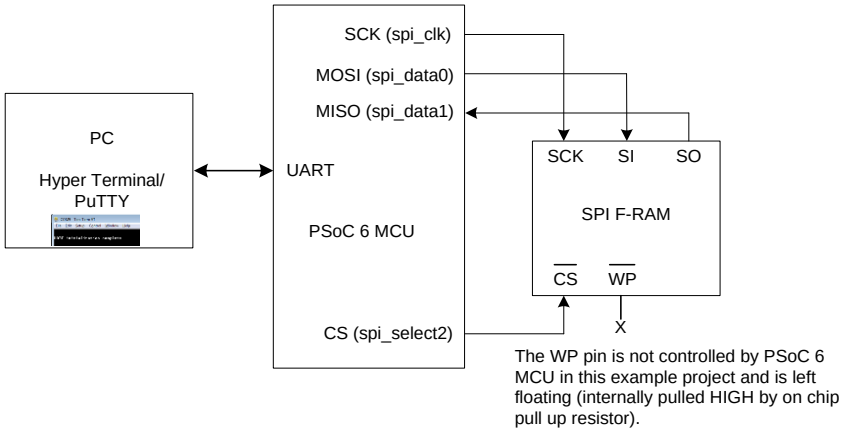
Associated Parts: All PSoC 6 MCU parts

Related Hardware: CY8CKIT-062-BLE- PSoC 6 BLE Pioneer Kit with SPI/QSPI F-RAM

Hardware Setup

The hardware setup includes connecting the SPI F-RAM with PSoC 6 MCU as shown in Figure 1. You can use either dedicated hardware as described in the Requirements section or connect via jumper wires by tapping the SMIF SPI control pins and connect to the SPI pins of an external SPI F-RAM. This example uses the PSoC 6 BLE Pioneer kit's default configuration. See the kit guide to make sure the kit is configured correctly.

Figure 1. Hardware Setup Block Diagram

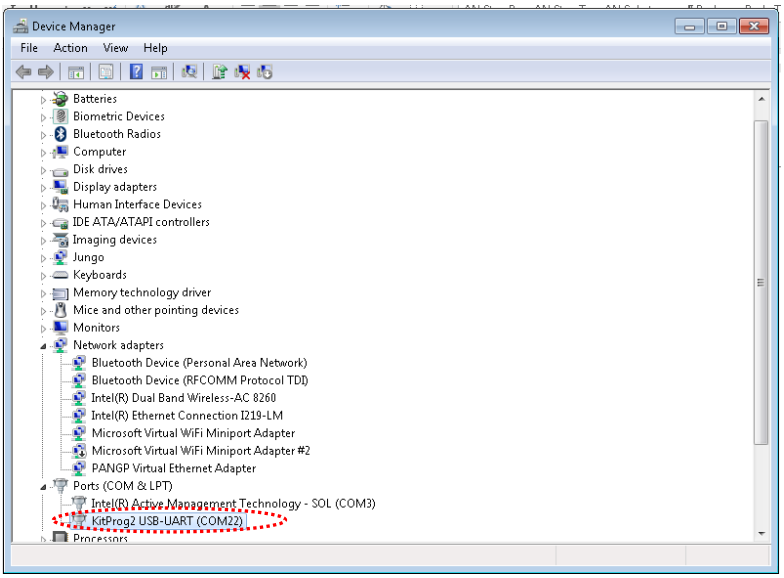


Software Setup

This section demonstrates the procedure to setup the serial (UART) connection using PuTTY on a PC to communicate with the PSoC 6 Pioneer Kit. PuTTY is a free SSH and telnet client for Windows. You can download PuTTY from www.putty.org. Follow these instructions to determine the COM port number and setup the PuTTY to monitor the code example outputs on PC.

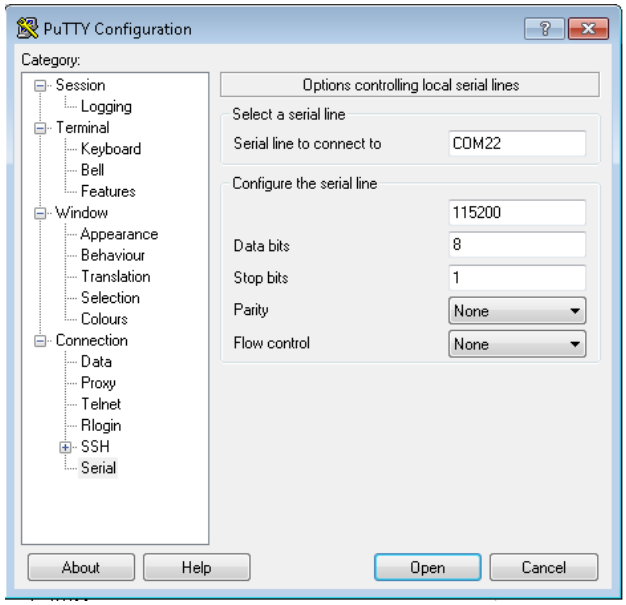
1. Connect the PSoC 6 Pioneer Kit to the PC using an USB cable. The kit enumerates as KitProg2 USB-UART and is available under the **Device Manager** > **Ports (COM & LPT)**. A communication port (COMx) is assigned to KitProg2 USB-UART; for example, COM22 is assigned to PSoC 6 Pioneer Kit on the sample setup, as shown in Figure 2.

Figure 2. KitProg2 USB-UART in Device Manager



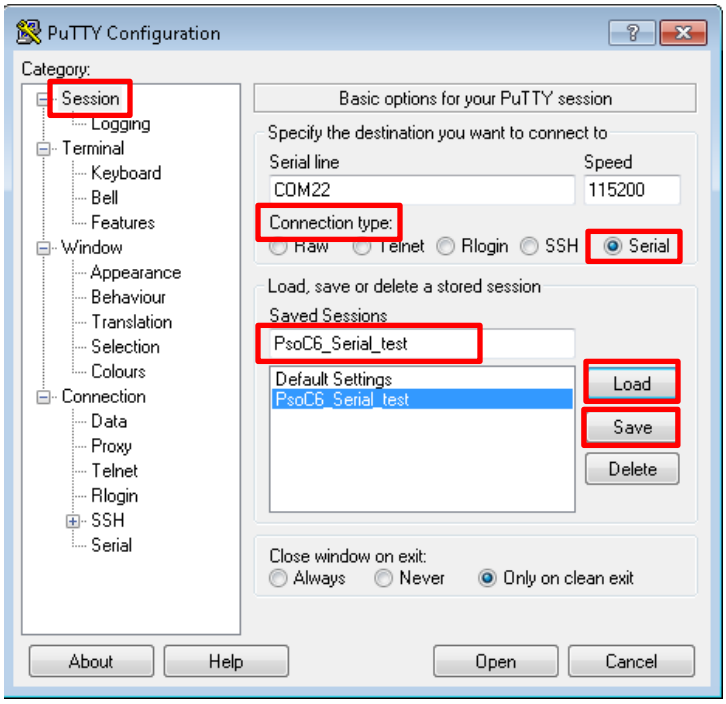
2. After you download and install PuTTY, double-click the PuTTY icon and select **Serial** under **Connection**.
3. A new window opens, as shown in Figure 3, where you can select the communication port. Do the following in the **Options controlling local serial lines** section:
   - In the **Serial line to connect to** field, enter the PSoC 6 Port (COM & LPT), COMx, in. This code example uses **COM22**. Verify the COM setting for your setup and select the appropriate COMx.
   - Enter the **Speed (baud)**, **Data bits**, and **Stop bits**.
   - Select the **Parity** and **Flow control**.

Figure 3. Open New Connection



4. From the Category panel, select **Session**. Select **Serial** as the **Connection type** as shown in Figure 4. You can save this current session and load the settings when required. Enter a name in **Saved Sessions** and click **Save**. Click **Open** to proceed.

Figure 4. Select Communication Type in PuTTY



5. The COM terminal window then displays the code example results as shown in Figure 5 after building and programming the code example project, as described in the Operation section. You may have to reprogram PSoC 6 MCU with the code example hex file or reset the PSoC 6 MCU (already programmed) to restart the code execution and monitor the result.

Figure 5. Result Displayed on PuTTY

```
********SPI F-RAM Access using PSoC 6 SMIF - Code Example (CE224073)*********
Read Device ID (RDID 0x9F)
 0x50 0x51 0x82 0x06 0x00 0x00 0x00 0x00 0x7F

Write data (256-Byte) in MMIO mode:
Write Address: 0x00 0x00 0x00
Write Data:
0x0A 0x0B 0x0C 0x0D 0x0E 0x0F 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19
0x1A 0x1B 0x1C 0x1D 0x1E 0x1F 0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29
0x2A 0x2B 0x2C 0x2D 0x2E 0x2F 0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39
0x3A 0x3B 0x3C 0x3D 0x3E 0x3F 0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49
0x4A 0x4B 0x4C 0x4D 0x4E 0x4F 0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59
0x5A 0x5B 0x5C 0x5D 0x5E 0x5F 0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69
0x6A 0x6B 0x6C 0x6D 0x6E 0x6F 0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79
0x7A 0x7B 0x7C 0x7D 0x7E 0x7F 0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89
0x8A 0x8B 0x8C 0x8D 0x8E 0x8F 0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99
0x9A 0x9B 0x9C 0x9D 0x9E 0x9F 0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9
0xAA 0xAB 0xAC 0xAD 0xAE 0xAF 0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9
0xBA 0xBB 0xBC 0xBD 0xBE 0xBF 0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9
0xCA 0xCB 0xCC 0xCD 0xCE 0xCF 0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9
0xDA 0xDB 0xDC 0xDD 0xDE 0xDF 0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9
0xEA 0xEB 0xEC 0xED 0xEE 0xEF 0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9
0xFA 0xFB 0xFC 0xFD 0xFE 0xFF 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09


Read Data (256-Byte) in MMIO mode:
Read Address: 0x00 0x00 0x00
Read Data:
0x0A 0x0B 0x0C 0x0D 0x0E 0x0F 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18 0x19
0x1A 0x1B 0x1C 0x1D 0x1E 0x1F 0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27 0x28 0x29
0x2A 0x2B 0x2C 0x2D 0x2E 0x2F 0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x39
0x3A 0x3B 0x3C 0x3D 0x3E 0x3F 0x40 0x41 0x42 0x43 0x44 0x45 0x46 0x47 0x48 0x49
0x4A 0x4B 0x4C 0x4D 0x4E 0x4F 0x50 0x51 0x52 0x53 0x54 0x55 0x56 0x57 0x58 0x59
0x5A 0x5B 0x5C 0x5D 0x5E 0x5F 0x60 0x61 0x62 0x63 0x64 0x65 0x66 0x67 0x68 0x69
0x6A 0x6B 0x6C 0x6D 0x6E 0x6F 0x70 0x71 0x72 0x73 0x74 0x75 0x76 0x77 0x78 0x79
0x7A 0x7B 0x7C 0x7D 0x7E 0x7F 0x80 0x81 0x82 0x83 0x84 0x85 0x86 0x87 0x88 0x89
0x8A 0x8B 0x8C 0x8D 0x8E 0x8F 0x90 0x91 0x92 0x93 0x94 0x95 0x96 0x97 0x98 0x99
0x9A 0x9B 0x9C 0x9D 0x9E 0x9F 0xA0 0xA1 0xA2 0xA3 0xA4 0xA5 0xA6 0xA7 0xA8 0xA9
0xAA 0xAB 0xAC 0xAD 0xAE 0xAF 0xB0 0xB1 0xB2 0xB3 0xB4 0xB5 0xB6 0xB7 0xB8 0xB9
0xBA 0xBB 0xBC 0xBD 0xBE 0xBF 0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7 0xC8 0xC9
0xCA 0xCB 0xCC 0xCD 0xCE 0xCF 0xD0 0xD1 0xD2 0xD3 0xD4 0xD5 0xD6 0xD7 0xD8 0xD9
0xDA 0xDB 0xDC 0xDD 0xDE 0xDF 0xE0 0xE1 0xE2 0xE3 0xE4 0xE5 0xE6 0xE7 0xE8 0xE9
0xEA 0xEB 0xEC 0xED 0xEE 0xEF 0xF0 0xF1 0xF2 0xF3 0xF4 0xF5 0xF6 0xF7 0xF8 0xF9
0xFA 0xFB 0xFC 0xFD 0xFE 0xFF 0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09


Set the SMIF memory mapped (XIP) mode

Read in memory mapped (XIP) read - 32 bit AHB access
Read Address: (@MEMBASE = 0x18000000)
Read Data:
0x0D 0x0C 0x0B 0x0A
0x11 0x10 0x0F 0x0E
0x15 0x14 0x13 0x12
0x19 0x18 0x17 0x16
0x1D 0x1C 0x1B 0x1A
0x21 0x20 0x1F 0x1E
0x25 0x24 0x23 0x22
0x29 0x28 0x27 0x26
0x2D 0x2C 0x2B 0x2A
0x31 0x30 0x2F 0x2E
```

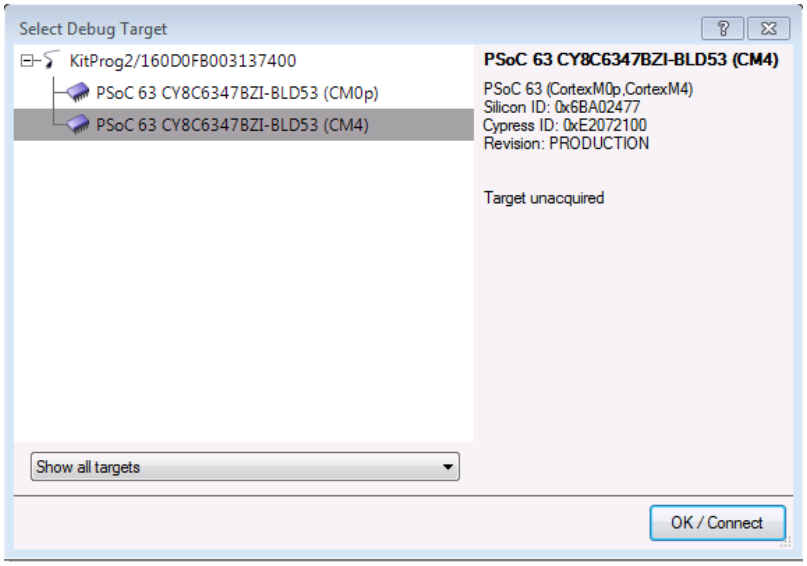
Alternatively, you can run the HyperTerminal, if supported on your PC, to monitor the result.

Operation

Do the following to execute the code example project. See the Design and Implementation section for more details.

1. Connect the CY8CKIT-062-BLE Pioneer Kit to a USB port on your PC. Set the V_{DD}; select either 1.8 V or 3.3 V using the switch SW5 on PSoC 6 Pioneer Kit. The SPI/QSPI F-RAM supports wide operating range V_{DD} = 1.8 V to 3.6 V.
2. Open a serial port communication program such as PuTTY and select the corresponding COM port. Configure the terminal to match the UART: 115200 baud rate, 8N1, and Flow control – None. See the Software Setup section for the PuTTY setup. These settings must match the configuration of the PSoC Creator UART Component in the project.
3. Build and program the application into the CY8CKIT-062-BLE Kit or CY8CKIT-062 Kit, which has serial F-RAM mounted on it. Select the CM4 option for programming, as shown in Figure 6. For more information on building a project or programming a device, see *PSoC Creator Help*.

Figure 6. PSoC 6 Programming (CM4)



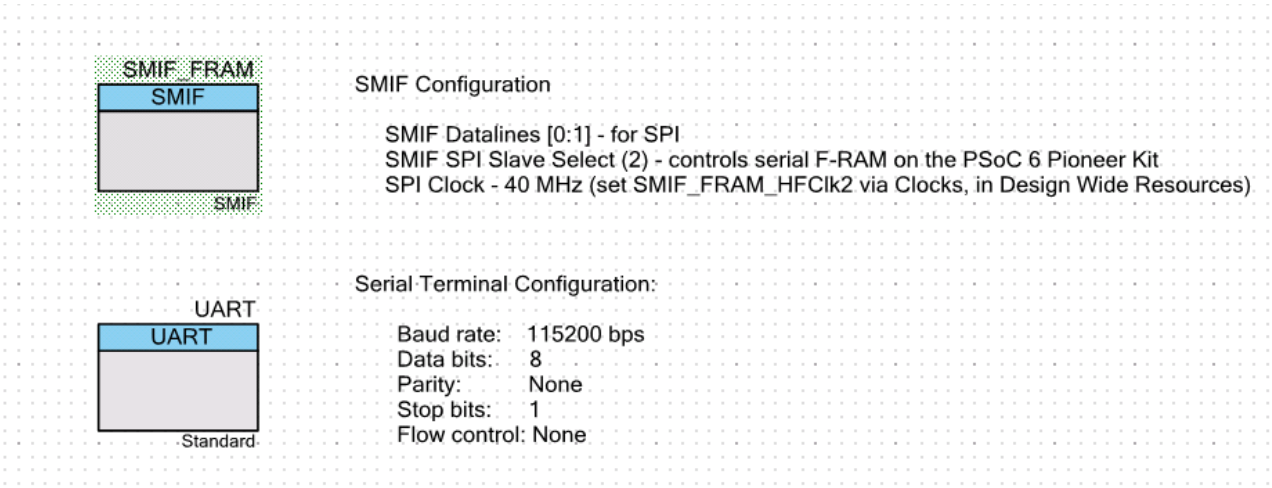
4. Observe the UART example header message printed in the terminal window. Figure 5 shows an example of the output.

Note: If you are using the ExcelonTM-Ultra (QSPI F-RAM) on the CY8CKIT-062-BLE kit to execute this code example, then make sure that the Excelon-Ultra device access mode is set to SPI and the memory and the register latencies are set to 0. See the code example *CE222967 - Excelon™-Ultra QSPI F-RAM Access Using PSoC 6 MCU SMIF* to change the access mode and the latency settings in the Excelon-Ultra device using PSoC 6 MCU.

Design and Implementation

Figure 7 shows the design for this code example. The SMIF Component implements a SPI-based communication for interfacing with an external SPI F-RAM with PSoC 6 as the host MCU. The SMIF Component is configured with two data lines (input SI, output SO), single slave select line, and the SPI clock (SCK) at 40 MHz. The UART Component outputs debug information to a terminal window. It is configured for 8N1, transmit only, at 115.2 kbps.

Figure 7. CE224073 Design Schematic in PSoC Creator



Cypress SPI F-RAMs support SPI clock frequency (SCK) up to 50 MHz. See the SPI F-RAM datasheet for more details on the access speed of the specific SPI F-RAM part used in your application.

Components and Settings

Table 1 lists all PSoC Creator Components used in the three examples.

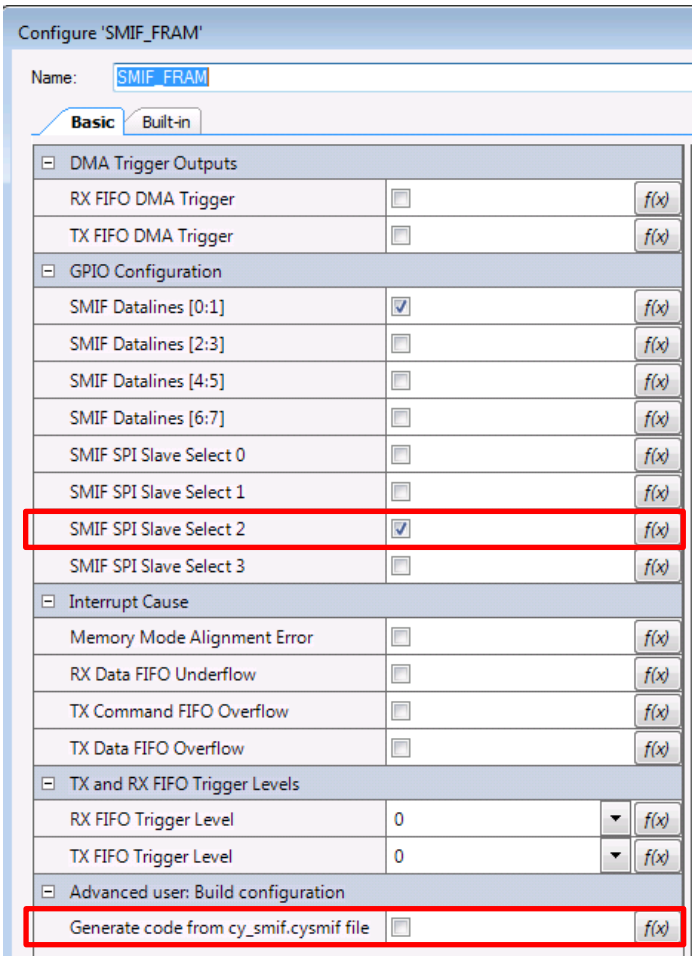
Table 1. PSoC Creator Components

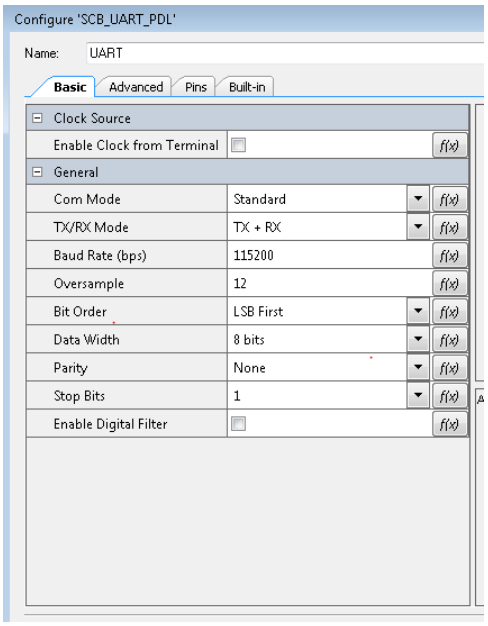
Component	Instance Name	Purpose
SMIF(SMIF_PDL)	SMIF_FRAM	The SMIF peripheral block. Configures the SPI host controller in the design
UART (SCB_UART_PDL)	UART	Handles communication with the terminal window

Parameter Settings

Figure 8 shows the parameter settings for the SMIF_FRAM Component and the UART (SCB). Non-default settings for each Component are outlined in red.

Figure 8. Parameters Setting for SMIF and UART Components (Non-default settings are highlighted)

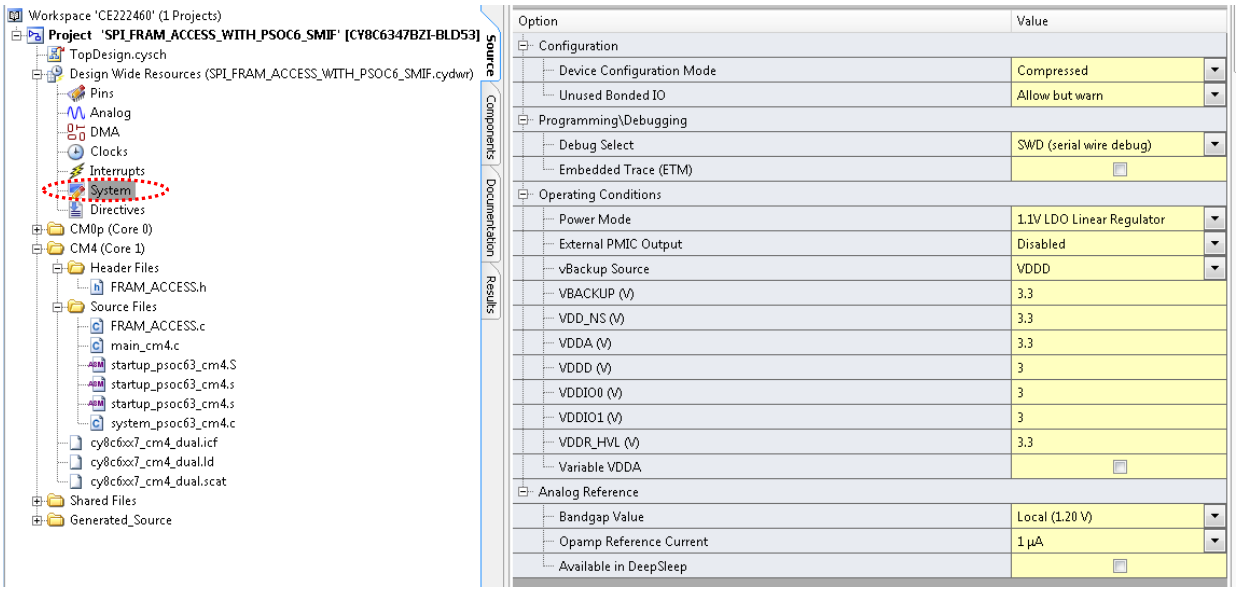




Design-Wide Resources

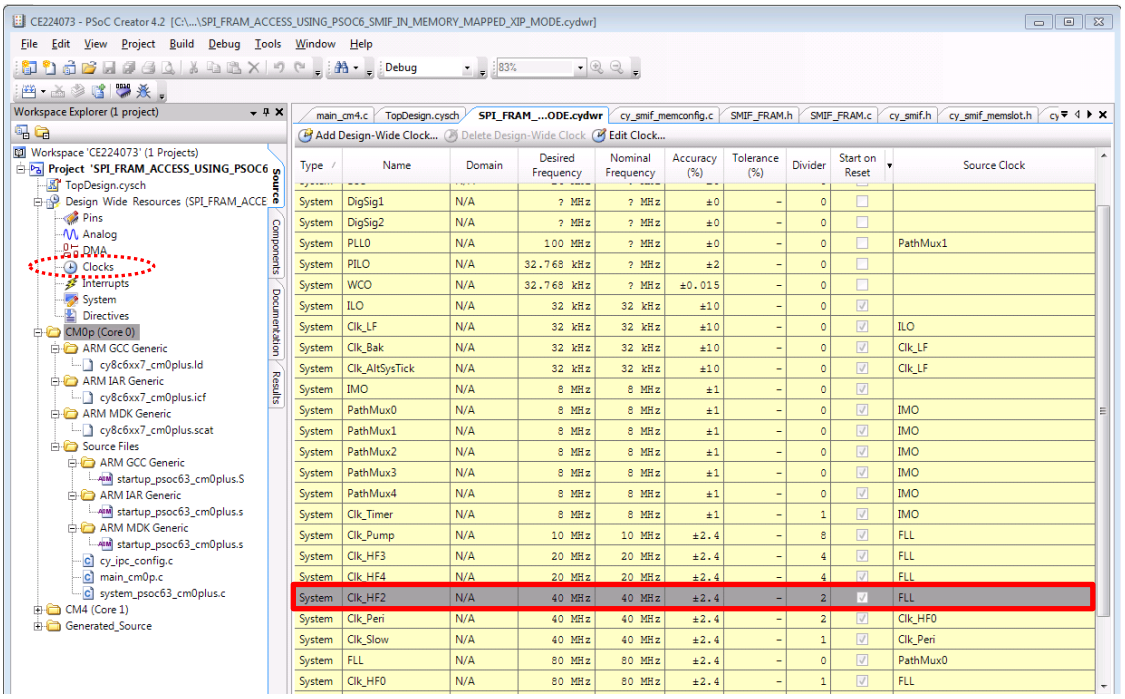
Make sure that V_{DDD} (**PSoC Creator** > **Design Wide Resources** > **System** tab) is set to 2.7 V or above, as shown in Figure 9, to drive the status LED (if any) used in the application. Also, make sure that PSoC 6 MCU I/O voltage is set correctly to match the SPI F-RAM operating range (V_{DD}/ V_{CC}).

Figure 9. V_{DD} Setting using Design Wide Resources



Make sure that SMIF SPI clock frequency is set at 50 MHz or below. To change the SMIF clock frequency, go to **PSoC Creator** > **Design Wide Resources** > **Clocks**, as shown in Figure 10, and double-click.

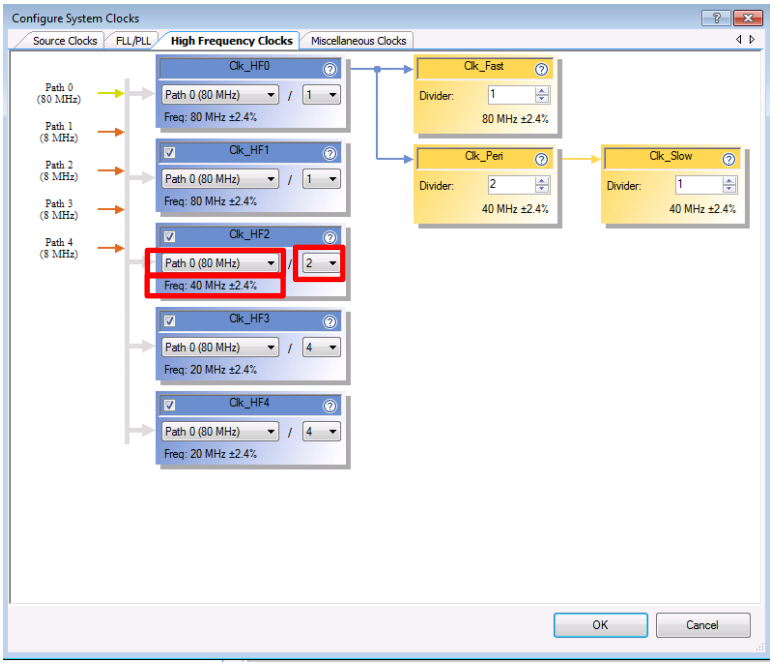
Figure 10. SMIF SPI Clock Setting using Design Wide Resources – Step 1



Double-click anywhere in the **Clk_HF2** row, as highlighted in Figure 10. A new Configure System Clocks window opens as shown in Figure 11.

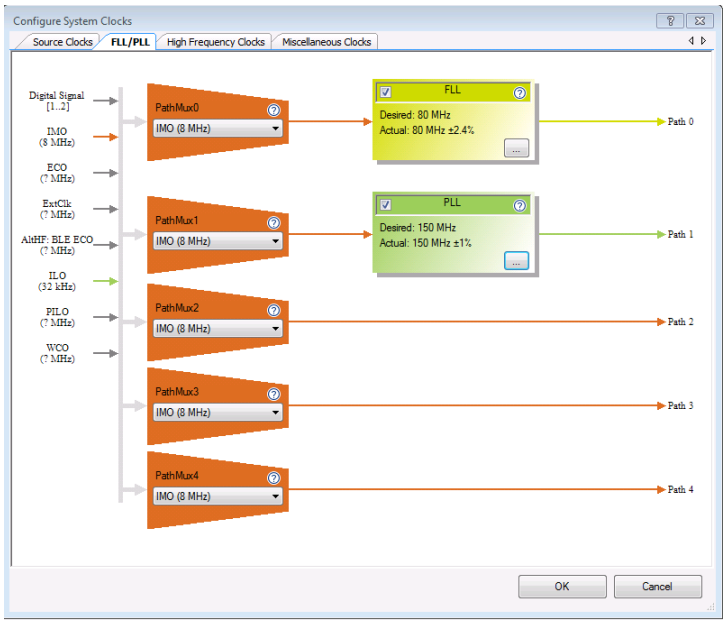
Go to the **High Frequency Clocks** tab and select the appropriate clock path and the clock divider from corresponding drop-down lists to set the frequency to 50 MHz or below (see Figure 11). This code example sets the SPI clock frequency to 40 MHz. Select **OK**.

Figure 11. SMIF SPI Clock Setting using Design Wide Resources – Step 2



You can use FLL/PLL, as shown in Figure 12, to configure other frequency options for Path 0.

Figure 12. FLL/PLL Setting examples – Step 3



PSoC 6 Pin Assignment

Figure 13 shows the pin assignment for the code example. The following PSoC 6 pins control the respective SPI F-RAM control pins for the SPI communication.

Figure 13. PSoC 6 Pin Assignments for Code Example

Name	Port	Pin
\SMIF_FRAM:spi_clk\	P11[7]	A5
\SMIF_FRAM:spi_data_0\	P11[6]	B5
\SMIF_FRAM:spi_data_1\	P11[5]	A6
\SMIF_FRAM:spi_select2\	P11[0]	F5
\UART:rx\	P5[0]	L6
\UART:tx\	P5[1]	K6

Reusing This Example

This example is designed for the CY8CKIT-062-BLE Pioneer Kit with serial F-RAM mounted. To port the design to a different PSoC 6 MCU device, kit, or both, change the target device using the Device Selector and update the pin assignments in Design Wide Resources Pins settings as needed. For single-core PSoC 6 MCU devices, port the code from *main_cm4.c* to *main.c*.

This code example also includes SPI F-RAM-specific PSoC 6 SMIF driver files (*.c* and *.h*) that can be used as in another project. The driver file description is follows. For more details on driver files content, see the code example project.

- *CY_SMIF_FRAM_CONFIG.h* - This file contains all defines for PSoC 6 SMIF component to access the SPI F-RAM in XIP mode.
- *CY_SMIF_FRAM_CONFIG.c* - This file contains the initialization for PSoC 6 SMIF component to access the SPI F-RAM in XIP mode
- *FRAM_ACCESS.h* - This file contains variable and all API declaration to access the SPI F-RAM in PSoC 6 SMIF MMIO mode.
- *FRAM_ACCESS.c* - This file contains all API definitions to access the SPI F-RAM in PSoC 6 SMIF MMIO mode.

In some cases, a resource used by a code example (for example, an IP block) is not supported on another device. In that case the example will not work. If you build the code targeted at such a device, you will get errors. See the PSoC device datasheet for information on the supported device.

Upgrade Information

None. See the Requirements section for details on tools and hardware requirements to successfully execute this code example.

Related Documents

Application Notes/Code Examples	
CE220823 – PSoC® 6 MCU SMIF Memory Write and Read Operation	This example demonstrates the write and read operations to the Serial Memory Interface (SMIF) in PSoC 6 MCU.
CE222460 - SPI F-RAM Access Using PSoC 6 MCU SMIF	This example demonstrates accessing the SPI F-RAM™ using PSoC® 6 MCU's Serial Memory Interface (SMIF) Component.
CE222967 - Excelon™-Ultra QSPI F-RAM Access Using PSoC 6 MCU SMIF	CE222967 demonstrates accessing the Excelon™-Ultra QSPI F-RAM™ using PSoC® 6 MCU's Serial Memory Interface (SMIF) Component in memory mapped I/O (MMIO) mode.
AN304 – SPI Guide for F-RAM™	AN304 provides the functional description, timing, and example code for SPI F-RAMs.
PSoC Creator Component Datasheets	
UART	UART communications interface
SMIF	Serial Memory Interface
Control Register	Allows the firmware to set values for to use for digital signals
General-Purpose Input / Output	Supports Analog, Digital I/O and Bidirectional signal types
Device Documentation	
PSoC® 6 MCU: PSoC® 63 with BLE Datasheet	PSoC® 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
SPI F-RAM (CY15B104QN) Datasheet	3 V, 50 MHz SPI, 4 Mb SPI FRAM datasheet
Development Kit (DVK) Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	

Document History

Document Title: CE224073 – SPI F-RAM Access Using PSoC 6 MCU SMIF in Memory Mapped (XIP) Mode

Document Number: 002-24073

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6214702	ZSK	06/22/2018	New code example

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Arm® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6 MCU

Cypress Developer Community

Community | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709