CE224034 – QSPI F-RAM Access Using PSoC 6 MCU with SMIF in XIP Mode

Objective

This example demonstrates how to use the Serial Memory Interface (SMIF) Component in execute-in-place (XIP) mode with external F-RAM™ on PSoC® 6 MCU devices.

Overview

This example uses the SMIF Component in XIP mode to manipulate an array in an external FRAM device. The workspace contains two projects: FRAM_XIP and FRAM_XIP_DMA. The FRAM_XIP project uses the SMIF Component and API to carry out operations. The FRAM_XIP_DMA project uses the SMIF Component with DMA to perform transfers to and from F-RAM.

Requirements

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

Programming Language: C (Arm® GCC 5.4.1)

Associated Parts: All PSoC 6 MCU parts

Related Hardware: CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit

Hardware Setup

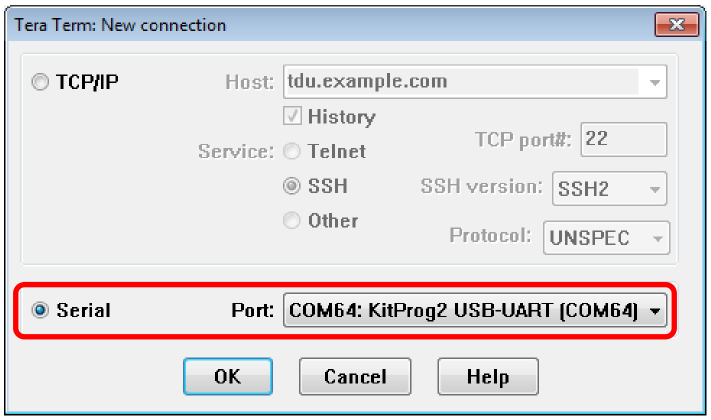
This example uses an external F-RAM device compatible with the CY8CKIT-062 and CY8CKIT-062 WIFI-BT kit. For either kit, ensure that footprint labeled U5 is populated.

Software Setup

This section describes how to set up a serial (UART) connection using Tera Term on PC to communicate with the PSoC 6 BLE Pioneer Kit. Tera Term is a free software terminal emulator for Windows, which can be downloaded here. Other software terminal emulator programs such as PuTTY can also be used.

1. Connect the PSoC 6 BLE Pioneer Kit to the PC using the USB cable.
2. After installing Tera Term, open the program and select the KitProg2 device in the Port drop down menu. Click **OK**.

Figure 1. Tera Term Port Selection



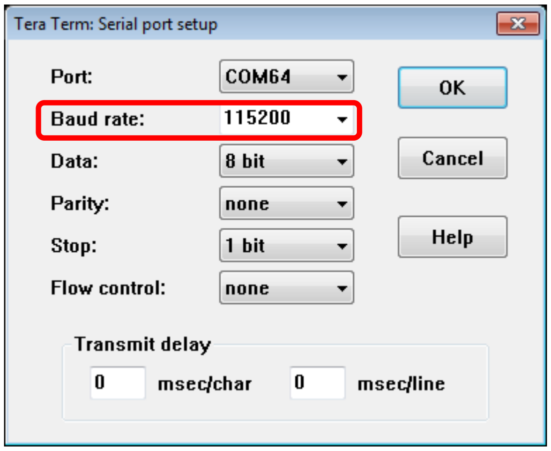
Tera Term: New connection

TCP/IP Host: tdu.example.com
History [✓]
Service: Telnet [✓ SSH] Other
TCP port#: 22
SSH version: SSH2
Protocol: UNSPEC

[✓ Serial] Port: COM64: KitProg2 USB-UART (COM64)

3. In Tera Term, select **Setup** > **Serial port** and set **Baud rate: 115200, Data: 8 bit**, **Parity: none**, **Stop 1 bit**, **Flow control: none**. Click **OK**.

Figure 2. Serial Port Configuration Settings



Tera Term: Serial port setup

Port: COM64
Baud rate: 115200
Data: 8 bit
Parity: none
Stop: 1 bit
Flow control: none

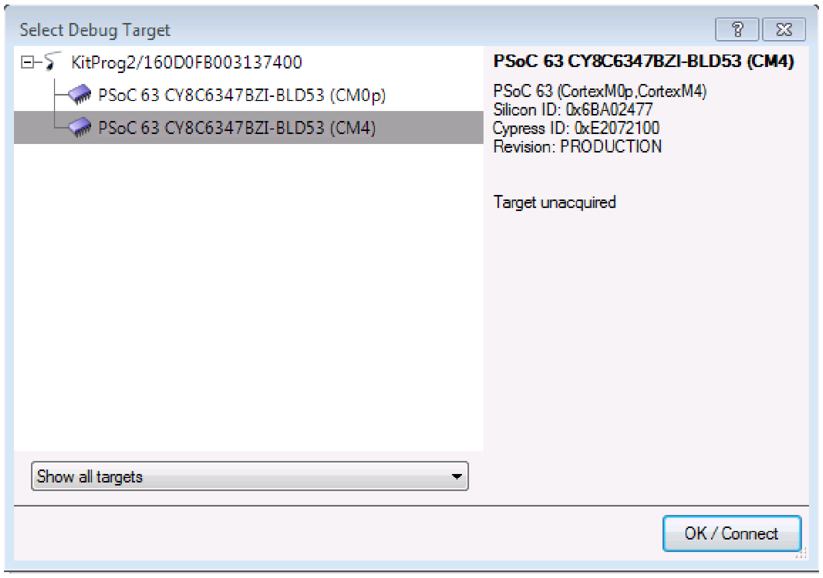
Transmit delay
0 msec/char 0 msec/line

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 Tera Term or 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 Tera Term setup. These settings must match the configuration of the PSoC Creator UART Component in the project.
3. Build and program either project 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 3. For more information on building a project or programming a device, see *PSoC Creator Help*.

Figure 3. PSoC 6 MCU Programming (CM4)



Select Debug Target

KitProg2/160D0FB003137400
PSoC 63 CY8C6347BZI-BLD53 (CM0p)
PSoC 63 CY8C6347BZI-BLD53 (CM4)

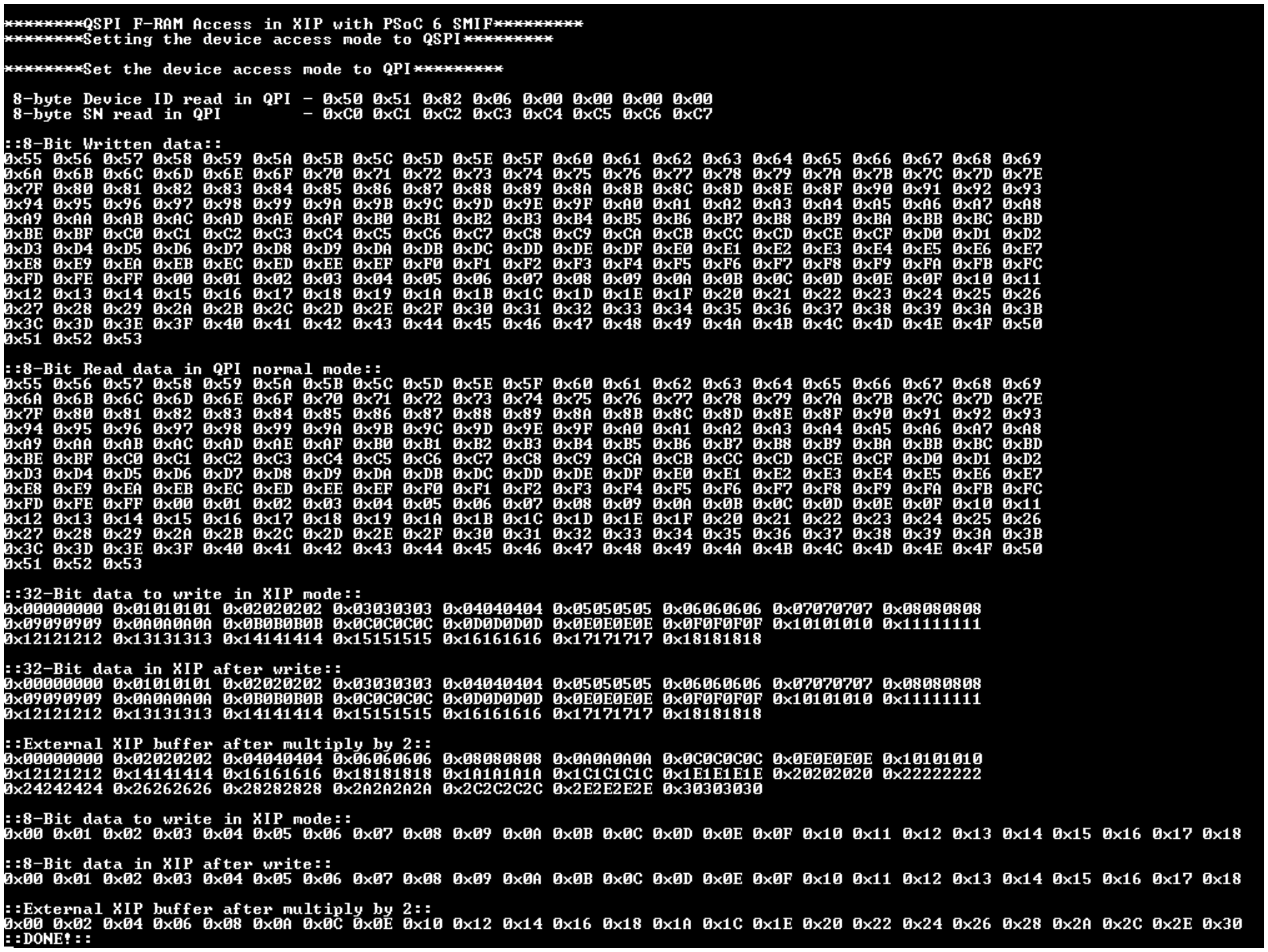
PSoC 63 CY8C6347BZI-BLD53 (CM4)
PSoC 63 (CortexM0p,CortexM4)
Silicon ID: 0x6BA02477
Cypress ID: 0xE2072100
Revision: PRODUCTION

Target unacquired

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

5. If you are using the Excelon™-Ultra (QSPI F-RAM) on the CY8CKIT-062-BLE kit to execute this code example, 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.

Figure 4. Terminal Output in Tera Term



```
*********QSPI F-RAM Access in XIP with PSoC 6 SMIF**********
*********Setting the device access mode to QSPI**********

*********Set the device access mode to QPI**********

 8-byte Device ID read in QPI - 0x50 0x51 0x82 0x06 0x00 0x00 0x00 0x00
 8-byte SN read in QPI        - 0xC0 0xC1 0xC2 0xC3 0xC4 0xC5 0xC6 0xC7

::8-Bit Written data::
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 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

::8-Bit Read data in QPI normal mode::
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 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

::32-Bit data to write in XIP mode::
0x00000000 0x01010101 0x02020202 0x03030303 0x04040404 0x05050505 0x06060606 0x07070707 0x08080808
0x09090909 0x0A0A0A0A 0x0B0B0B0B 0x0C0C0C0C 0x0D0D0D0D 0x0E0E0E0E 0x0F0F0F0F 0x10101010 0x11111111
0x12121212 0x13131313 0x14141414 0x15151515 0x16161616 0x17171717 0x18181818

::32-Bit data in XIP after write::
0x00000000 0x01010101 0x02020202 0x03030303 0x04040404 0x05050505 0x06060606 0x07070707 0x08080808
0x09090909 0x0A0A0A0A 0x0B0B0B0B 0x0C0C0C0C 0x0D0D0D0D 0x0E0E0E0E 0x0F0F0F0F 0x10101010 0x11111111
0x12121212 0x13131313 0x14141414 0x15151515 0x16161616 0x17171717 0x18181818

::External XIP buffer after multiply by 2::
0x00000000 0x02020202 0x04040404 0x06060606 0x08080808 0x0A0A0A0A 0x0C0C0C0C 0x0E0E0E0E 0x10101010
0x12121212 0x14141414 0x16161616 0x18181818 0x1A1A1A1A 0x1C1C1C1C 0x1E1E1E1E 0x20202020 0x22222222
0x24242424 0x26262626 0x28282828 0x2A2A2A2A 0x2C2C2C2C 0x2E2E2E2E 0x30303030

::8-Bit data to write in XIP mode::
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18

::8-Bit data in XIP after write::
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F 0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17 0x18

::External XIP buffer after multiply by 2::
0x00 0x02 0x04 0x06 0x08 0x0A 0x0C 0x0E 0x10 0x12 0x14 0x16 0x18 0x1A 0x1C 0x1E 0x20 0x22 0x24 0x26 0x28 0x2A 0x2C 0x2E 0x30
::DONE!::
```

Design and Implementation

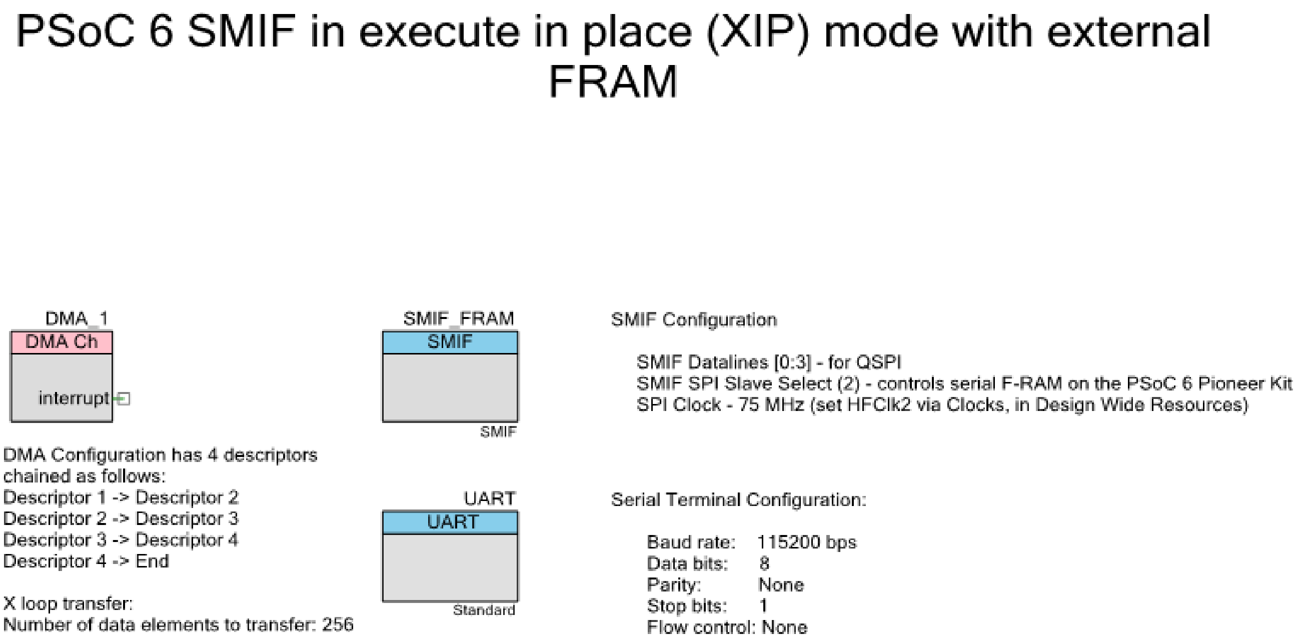
Figure 5 shows the schematic with the Components for this example. An instance of the SMIF Component enables communication with the QSPI-based F-RAM. The SMIF Component is configured in Quad-SPI mode with 4 data lines, a single slave select line, and the SPI clock at 75 MHz. The UART Component prints out the data being written to and read from the F-RAM. The UART also prints information about any failures that occur during operation. A direct memory access (DMA) Component is configured with four descriptors that transfer data between three arrays during operation.

8-bit and 32-bit arrays are initialized at the address of the external F-RAM. These arrays are manipulated in XIP mode and then printed out over UART. To demonstrate XIP reads and writes to the external arrays, a sequence of transfers is performed in firmware.

First, the firmware writes 8-bit values to a local array. Then, the local array is copied over to the external array. Finally, each value in the external array is doubled and the values are printed over the serial terminal. This sequence is repeated with the 32-bit array.

To demonstrate using DMA with data in the F-RAM, a second project is included in the workspace. The FRAM_XIP_DMA project initializes descriptors at the addresses of each of the data arrays. On a firmware trigger, the DMA transfers data from the local 32-bit array to an external array that the SMIF Component has mapped into the CPU memory space. To demonstrate manipulation of the data in XIP mode, this data is then doubled and printed over the UART.

Figure 5. F-RAM XIP Schematic


PSoC 6 SMIF in execute in place (XIP) mode with external FRAM
DMA_1
DMA Ch
interrupt
SMIF_FRAM
SMIF
SMIF
UART
UART
Standard

DMA Configuration has 4 descriptors chained as follows:
Descriptor 1 -> Descriptor 2
Descriptor 2 -> Descriptor 3
Descriptor 3 -> Descriptor 4
Descriptor 4 -> End

X loop transfer:
Number of data elements to transfer: 256

SMIF Configuration

SMIF Datalines [0:3] - for QSPI
SMIF SPI Slave Select (2) - controls serial F-RAM on the PSoC 6 Pioneer Kit
SPI Clock - 75 MHz (set HFClk2 via Clocks, in Design Wide Resources)

Serial Terminal Configuration:

Baud rate:	115200 bps
Data bits:	8
Parity:	None
Stop bits:	1
Flow control:	None

Components and Settings

Table 1 lists the PSoC Creator Components used in this example, how they are used in the design, and the non-default settings required so they function as intended.

Table 1: PSoC Creator Components

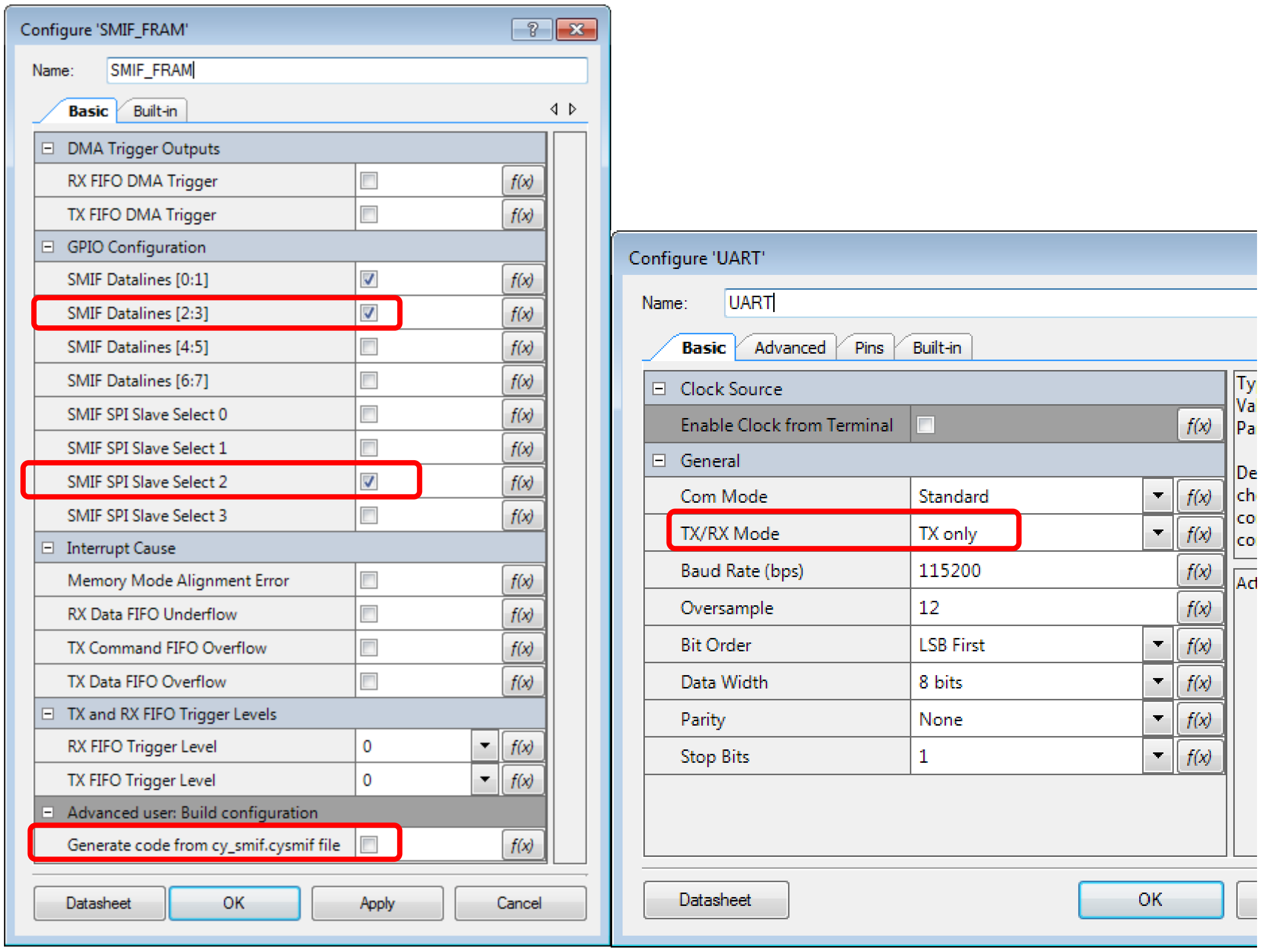
Component	Instance Name	Purpose
SMIF	SMIF_FRAM	Enables SPI-based communication with an external memory
UART (SCB)	UART	Enables communication with a serial terminal
DMA	DMA_1	Enables direct memory access

For information on the hardware resources used by a Component, see the Component datasheet.

Parameter Settings

The Component configuration settings are shown in the following figures. Figure 6 shows the configuration settings for the SMIF and UART Components.

Figure 6. SMIF and UART Component Configuration


Configure 'SMIF_FRAM'

Name: SMIF_FRAM

Basic | Built-in

DMA Trigger Outputs	
RX FIFO DMA Trigger	☐
TX FIFO DMA Trigger	☐
GPIO Configuration	
SMIF Datalines [0:1]	✓
SMIF Datalines [2:3]	✓
SMIF Datalines [4:5]	☐
SMIF Datalines [6:7]	☐
SMIF SPI Slave Select 0	☐
SMIF SPI Slave Select 1	☐
SMIF SPI Slave Select 2	✓
SMIF SPI Slave Select 3	☐
Interrupt Cause	
Memory Mode Alignment Error	☐
RX Data FIFO Underflow	☐
TX Command FIFO Overflow	☐
TX Data FIFO Overflow	☐
TX and RX FIFO Trigger Levels	
RX FIFO Trigger Level	0
TX FIFO Trigger Level	0
Advanced user: Build configuration	
Generate code from cy_smif.cysmif file	☐

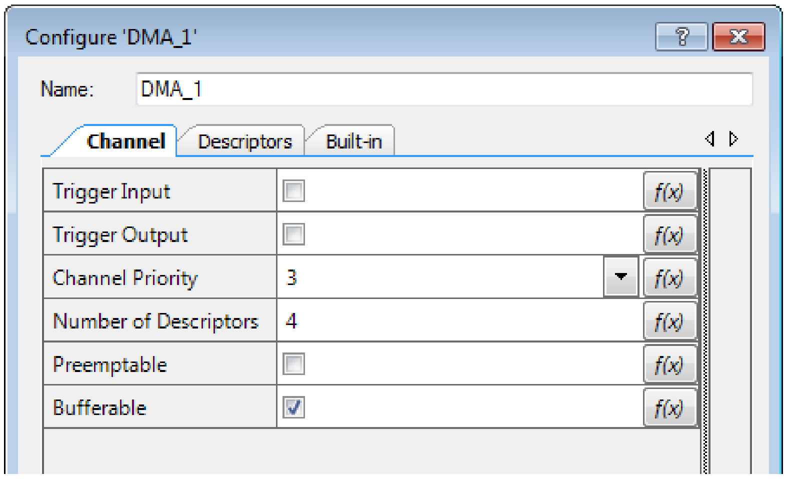
Configure 'UART'

Name: UART

Basic | Advanced | Pins | Built-in

Clock Source	
Enable Clock from Terminal	☐
General	
Com Mode	Standard
TX/RX Mode	TX only
Baud Rate (bps)	115200
Oversample	12
Bit Order	LSB First
Data Width	8 bits
Parity	None
Stop Bits	1

Figure 7. DMA Settings

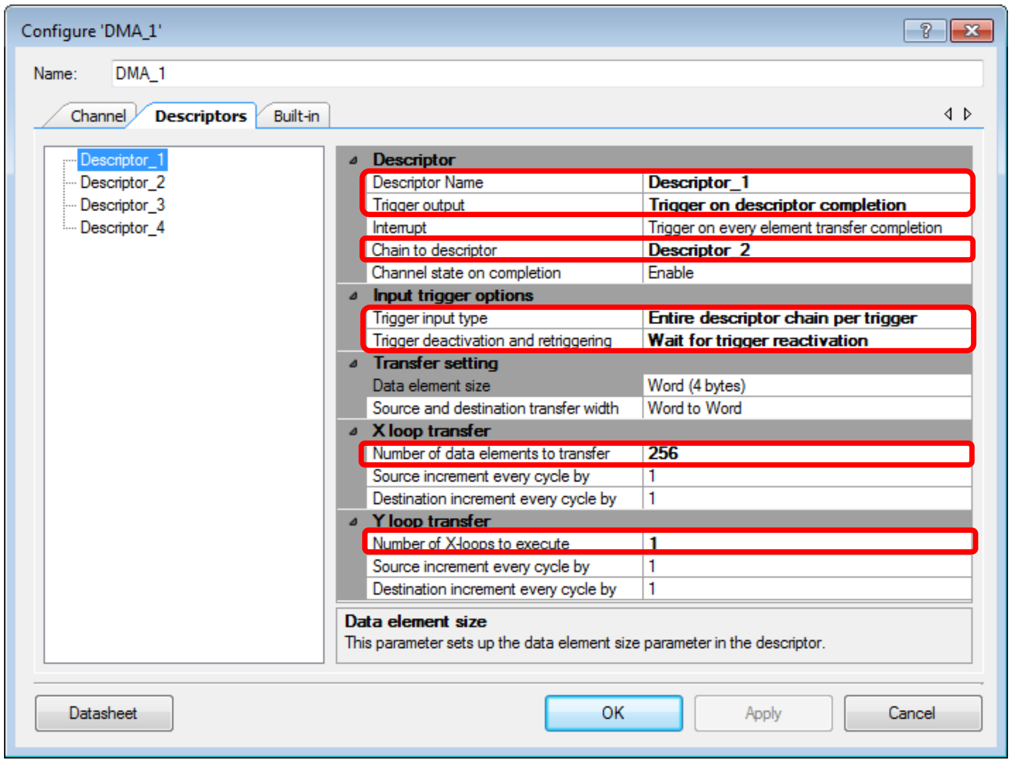

Configure 'DMA_1'

Name: DMA_1

Channel | Descriptors | Built-in

Trigger Input	☐
Trigger Output	☐
Channel Priority	3
Number of Descriptors	4
Preemptable	☐
Bufferable	✓

Figure 8. DMA Descriptor Settings

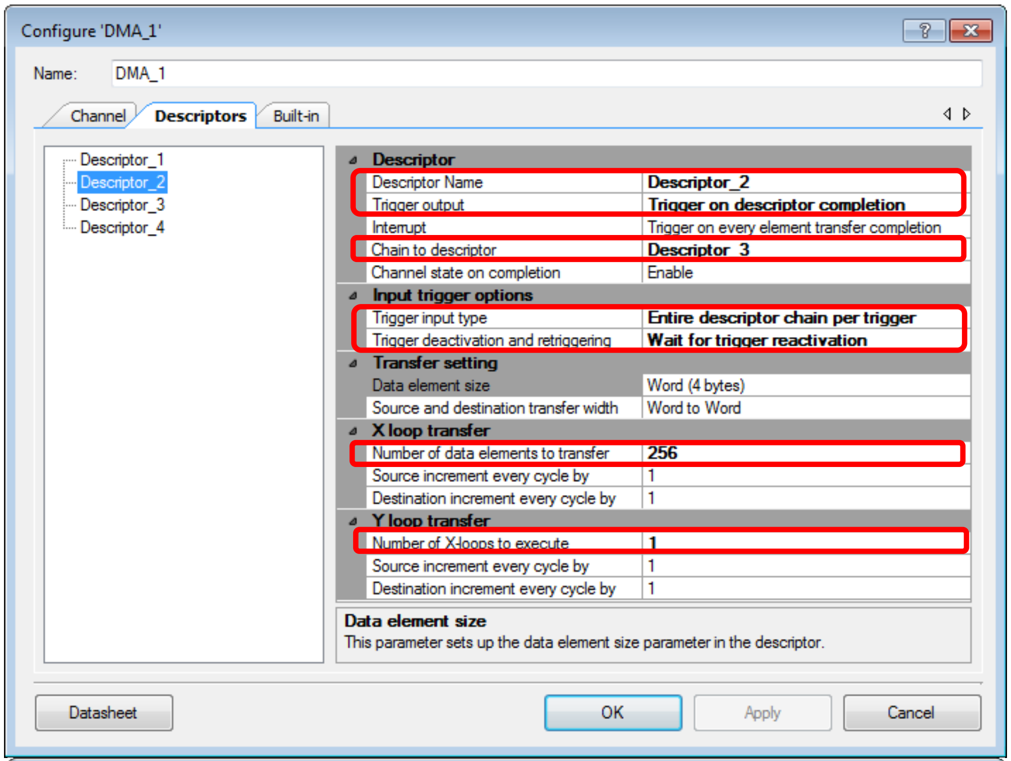


Configure 'DMA_1'

Name: DMA_1

Descriptor	
Descriptor Name	**Descriptor_1**
Trigger output	**Trigger on descriptor completion**
Interrupt	Trigger on every element transfer completion
Chain to descriptor	**Descriptor_2**
Channel state on completion	Enable
Input trigger options	
Trigger input type	**Entire descriptor chain per trigger**
Trigger deactivation and retriggering	**Wait for trigger reactivation**
Transfer setting	
Data element size	Word (4 bytes)
Source and destination transfer width	Word to Word
X loop transfer	
Number of data elements to transfer	**256**
Source increment every cycle by	1
Destination increment every cycle by	1
Y loop transfer	
Number of X-loops to execute	**1**
Source increment every cycle by	1
Destination increment every cycle by	1

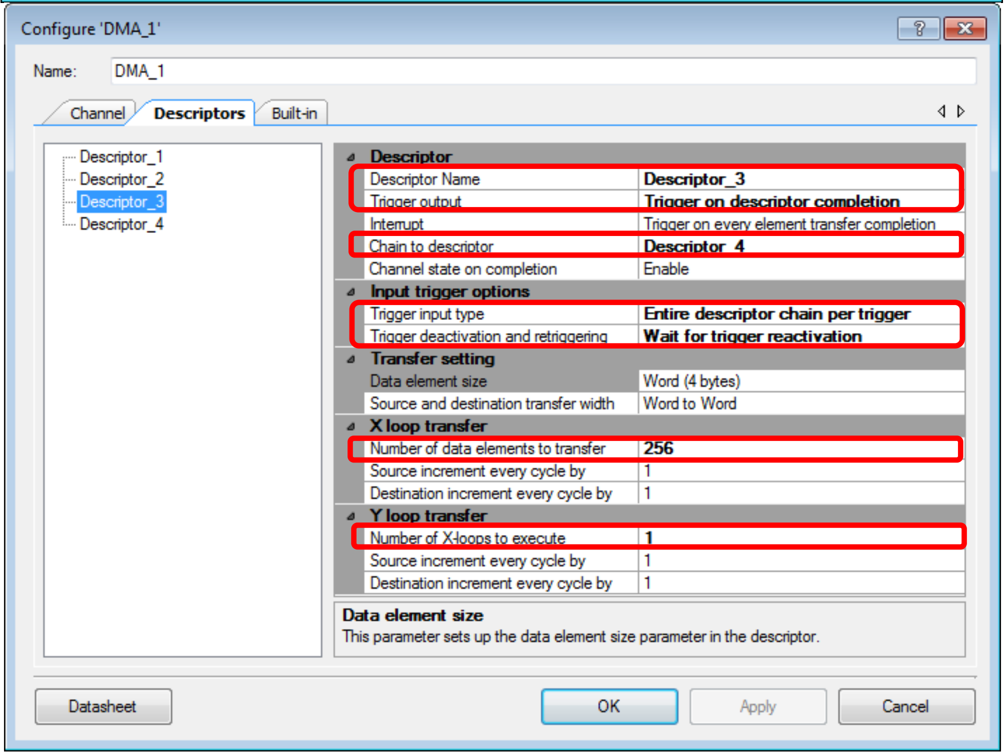
Data element size
This parameter sets up the data element size parameter in the descriptor.



Name: DMA_1

Descriptor	
Descriptor Name	**Descriptor_2**
Trigger output	**Trigger on descriptor completion**
Interrupt	Trigger on every element transfer completion
Chain to descriptor	**Descriptor_3**
Channel state on completion	Enable
Input trigger options	
Trigger input type	**Entire descriptor chain per trigger**
Trigger deactivation and retriggering	**Wait for trigger reactivation**
Transfer setting	
Data element size	Word (4 bytes)
Source and destination transfer width	Word to Word
X loop transfer	
Number of data elements to transfer	**256**
Source increment every cycle by	1
Destination increment every cycle by	1
Y loop transfer	
Number of X-loops to execute	**1**
Source increment every cycle by	1
Destination increment every cycle by	1

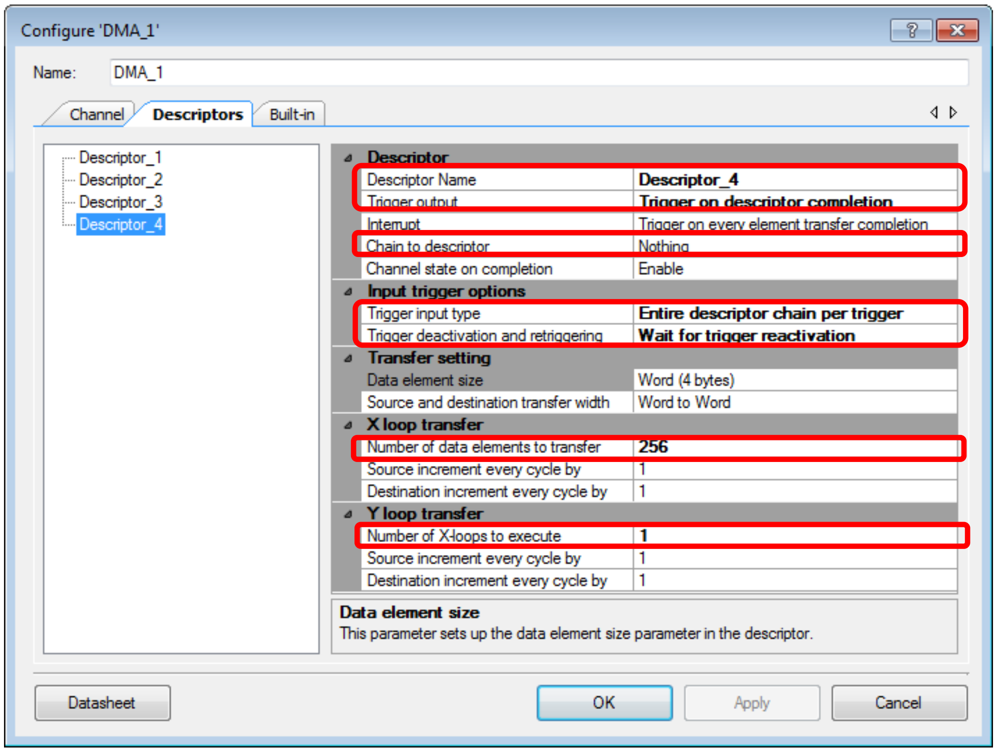
Data element size
This parameter sets up the data element size parameter in the descriptor.



Name: DMA_1

Descriptor	
Descriptor Name	**Descriptor_3**
Trigger output	**Trigger on descriptor completion**
Interrupt	Trigger on every element transfer completion
Chain to descriptor	**Descriptor_4**
Channel state on completion	Enable
Input trigger options	
Trigger input type	**Entire descriptor chain per trigger**
Trigger deactivation and retriggering	**Wait for trigger reactivation**
Transfer setting	
Data element size	Word (4 bytes)
Source and destination transfer width	Word to Word
X loop transfer	
Number of data elements to transfer	**256**
Source increment every cycle by	1
Destination increment every cycle by	1
Y loop transfer	
Number of X-loops to execute	**1**
Source increment every cycle by	1
Destination increment every cycle by	1

Data element size
This parameter sets up the data element size parameter in the descriptor.



Configure 'DMA_1'

Name: DMA_1

Channel | Descriptors | Built-in

- Descriptor_1
- Descriptor_2
- Descriptor_3
- Descriptor_4

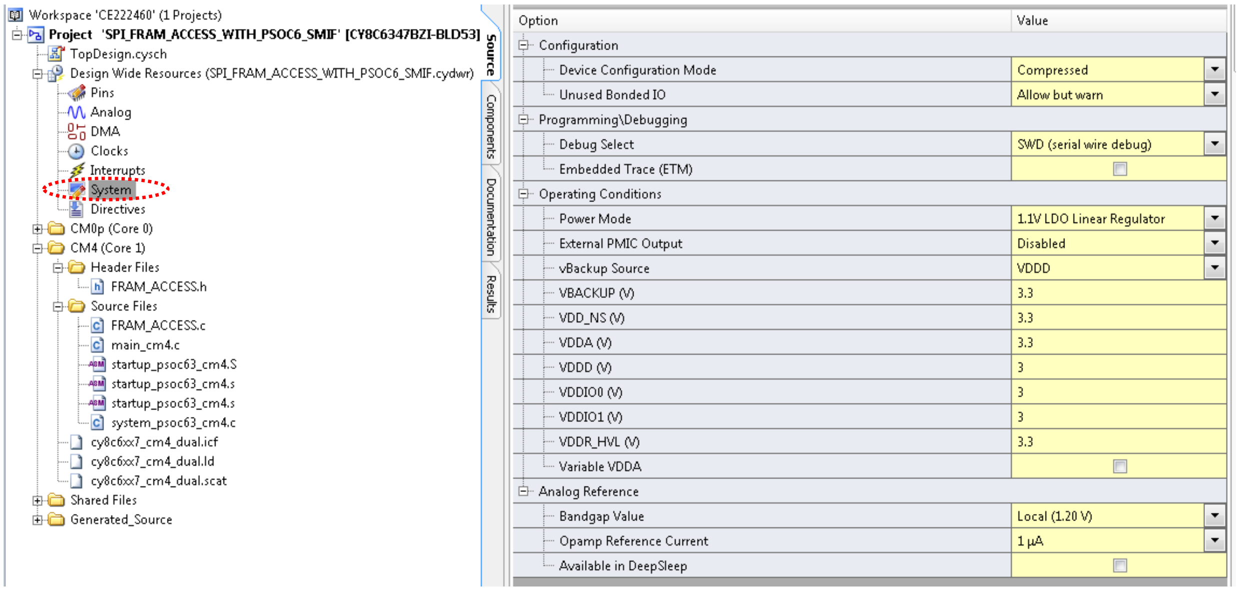
Descriptor	
Descriptor Name	**Descriptor_4**
Trigger output	**Trigger on descriptor completion**
Interrupt	Trigger on every element transfer completion
Chain to descriptor	Nothing
Channel state on completion	Enable
Input trigger options	
Trigger input type	**Entire descriptor chain per trigger**
Trigger deactivation and retriggering	**Wait for trigger reactivation**
Transfer setting	
Data element size	Word (4 bytes)
Source and destination transfer width	Word to Word
X loop transfer	
Number of data elements to transfer	**256**
Source increment every cycle by	1
Destination increment every cycle by	1
Y loop transfer	
Number of X-loops to execute	**1**
Source increment every cycle by	1
Destination increment every cycle by	1

Data element size
This parameter sets up the data element size parameter in the descriptor.

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

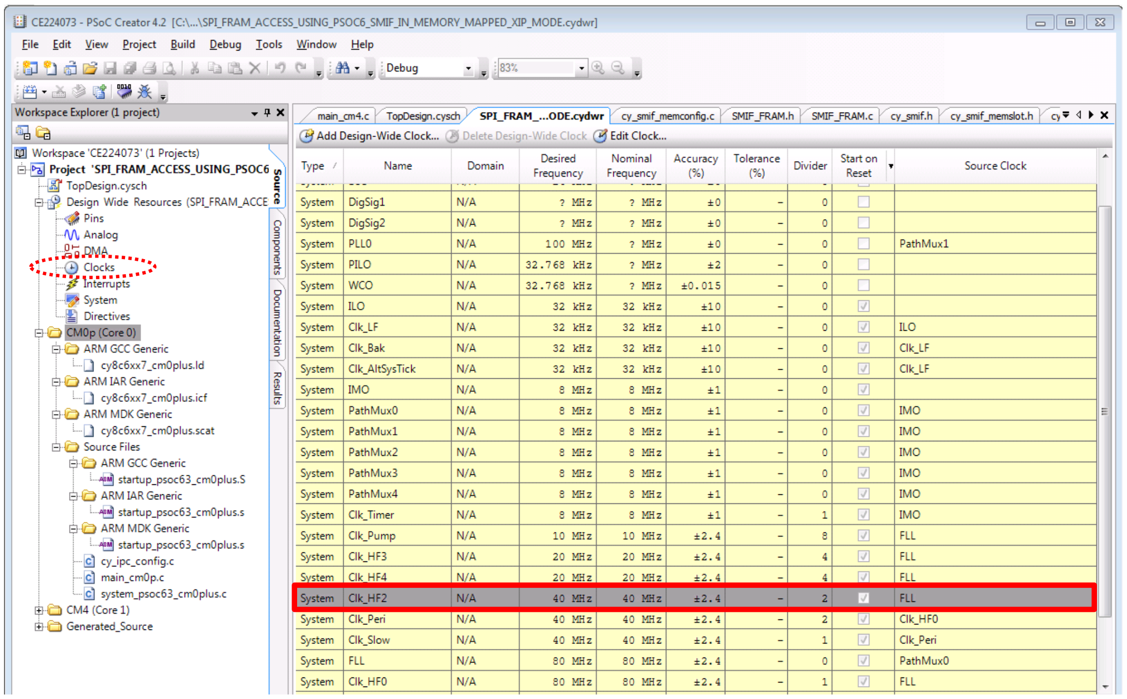


Workspace 'CE222460' (1 Projects)
- Project 'SPI_FRAM_ACCESS_WITH_PSOC6_SMIF' [CY8C6347BZI-BLD53]
  - TopDesign.cysch
  - Design Wide Resources (SPI_FRAM_ACCESS_WITH_PSOC6_SMIF.cydwr)
    - Pins
    - Analog
    - DMA
    - Clocks
    - Interrupts
    - System
    - Directives
  - CM0p (Core 0)
  - CM4 (Core 1)
    - Header Files
      - FRAM_ACCESS.h
    - Source Files
      - FRAM_ACCESS.c
      - main_cm4.c
      - startup_psoc63_cm4.S
      - startup_psoc63_cm4.s
      - startup_psoc63_cm4.s
      - system_psoc63_cm4.c
    - cy8c6xx7_cm4_dual.icf
    - cy8c6xx7_cm4_dual.ld
    - cy8c6xx7_cm4_dual.scat
  - Shared Files
  - Generated_Source

Option	Value
Configuration	
Device Configuration Mode	Compressed
Unused Bonded IO	Allow but warn
Programming\Debugging	
Debug Select	SWD (serial wire debug)
Embedded Trace (ETM)	☐
Operating Conditions	
Power Mode	1.1V LDO Linear Regulator
External PMIC Output	Disabled
vBackup Source	VDDD
VBACKUP (V)	3.3
VDD_NS (V)	3.3
VDDA (V)	3.3
VDDD (V)	3
VDDIO0 (V)	3
VDDIO1 (V)	3
VDDR_HVL (V)	3.3
Variable VDDA	☐
Analog Reference	
Bandgap Value	Local (1.20 V)
Opamp Reference Current	1 µA
Available in DeepSleep	☐

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**, and double-click **Clocks**, as shown in Figure 10.

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

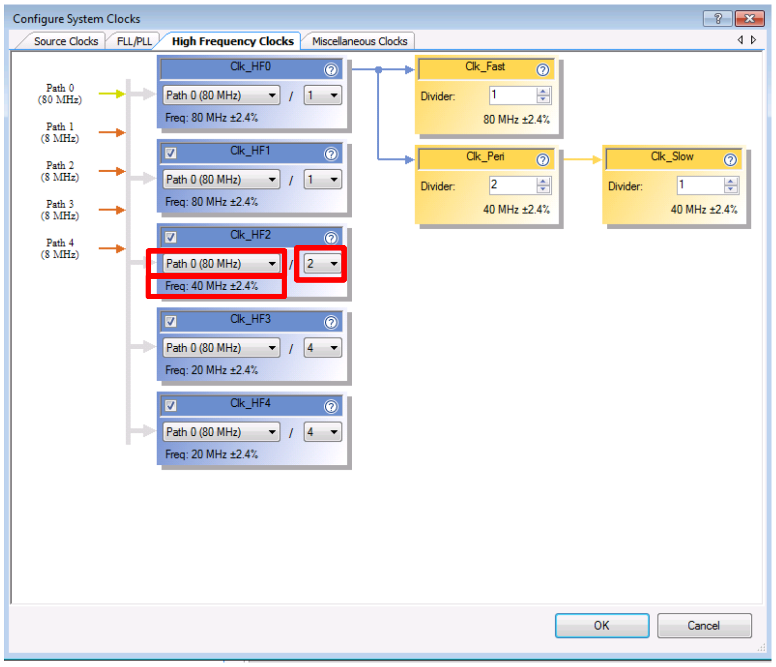


Type	Name	Domain	Desired Frequency	Nominal Frequency	Accuracy (%)	Tolerance (%)	Divider	Start on Reset	Source Clock
System	DigSig1	N/A	? MHz	? MHz	±0	–	0	☐	
System	DigSig2	N/A	? MHz	? MHz	±0	–	0	☐	
System	PLL0	N/A	100 MHz	? MHz	±0	–	0	☐	PathMux1
System	PILO	N/A	32.768 kHz	? MHz	±2	–	0	☐	
System	WCO	N/A	32.768 kHz	? MHz	±0.015	–	0	☐	
System	ILO	N/A	32 kHz	32 kHz	±10	–	0	✓	
System	Clk_LF	N/A	32 kHz	32 kHz	±10	–	0	✓	ILO
System	Clk_Bak	N/A	32 kHz	32 kHz	±10	–	0	✓	Clk_LF
System	Clk_AltSysTick	N/A	32 kHz	32 kHz	±10	–	0	✓	Clk_LF
System	IMO	N/A	8 MHz	8 MHz	±1	–	0	✓	
System	PathMux0	N/A	8 MHz	8 MHz	±1	–	0	✓	IMO
System	PathMux1	N/A	8 MHz	8 MHz	±1	–	0	✓	IMO
System	PathMux2	N/A	8 MHz	8 MHz	±1	–	0	✓	IMO
System	PathMux3	N/A	8 MHz	8 MHz	±1	–	0	✓	IMO
System	PathMux4	N/A	8 MHz	8 MHz	±1	–	0	✓	IMO
System	Clk_Timer	N/A	8 MHz	8 MHz	±1	–	1	✓	IMO
System	Clk_Pump	N/A	10 MHz	10 MHz	±2.4	–	8	✓	FLL
System	Clk_HF3	N/A	20 MHz	20 MHz	±2.4	–	4	✓	FLL
System	Clk_HF4	N/A	20 MHz	20 MHz	±2.4	–	4	✓	FLL
System	Clk_HF2	N/A	40 MHz	40 MHz	±2.4	–	2	✓	FLL
System	Clk_Peri	N/A	40 MHz	40 MHz	±2.4	–	2	✓	Clk_HF0
System	Clk_Slow	N/A	40 MHz	40 MHz	±2.4	–	1	✓	Clk_Peri
System	FLL	N/A	80 MHz	80 MHz	±2.4	–	0	✓	PathMux0
System	Clk_HF0	N/A	80 MHz	80 MHz	±2.4	–	1	✓	FLL

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



Clk_HF0: Path 0 (80 MHz) / 1 — Freq: 80 MHz ±2.4%

Clk_HF1: Path 0 (80 MHz) / 1 — Freq: 80 MHz ±2.4%

Clk_HF2: Path 0 (80 MHz) / 2 — Freq: 40 MHz ±2.4%

Clk_HF3: Path 0 (80 MHz) / 4 — Freq: 20 MHz ±2.4%

Clk_HF4: Path 0 (80 MHz) / 4 — Freq: 20 MHz ±2.4%

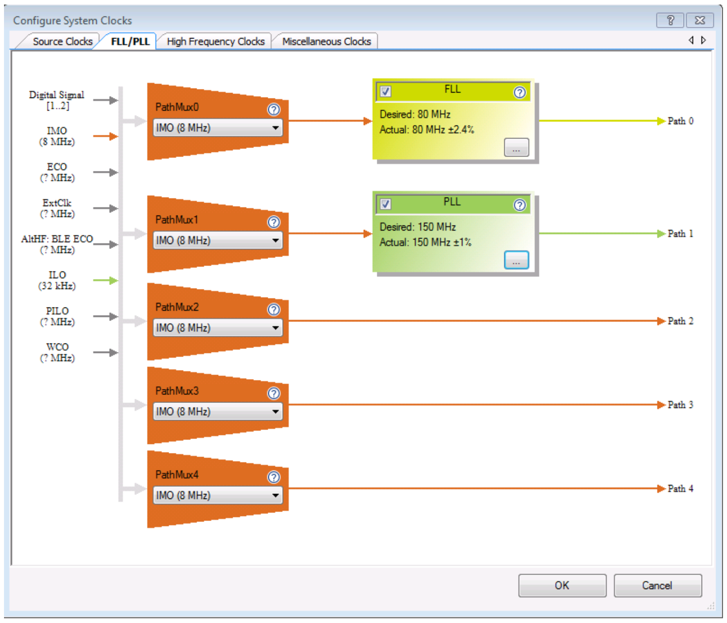
Clk_Fast: Divider: 1 — 80 MHz ±2.4%

Clk_Peri: Divider: 2 — 40 MHz ±2.4%

Clk_Slow: Divider: 1 — 40 MHz ±2.4%

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



PathMux0: IMO (8 MHz)
PathMux1: IMO (8 MHz)
PathMux2: IMO (8 MHz)
PathMux3: IMO (8 MHz)
PathMux4: IMO (8 MHz)

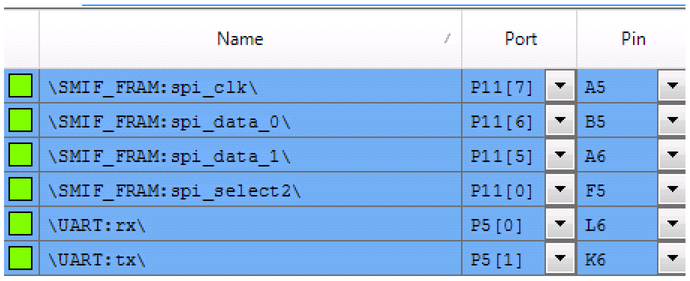
FLL: Desired: 80 MHz, Actual: 80 MHz ±2.4%

PLL: Desired: 150 MHz, Actual: 150 MHz ±1%

PSoC 6 Pin Assignment

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

Figure 13. PSoC 6 MCU 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 a serial F-RAM device 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 MCU 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 MCU SMIF Component to access the QSPI F-RAM in XIP mode.
- *CY_SMIF_FRAM_CONFIG.c* - This file contains the initialization for PSoC 6 MCU SMIF Component to access the QSPI F-RAM in XIP mode
- *FRAM_ACCESS.h* - This file contains variable and all API declarations to access the QSPI F-RAM in PSoC 6 MCU SMIF MMIO mode.
- *FRAM_ACCESS.c* - This file contains all API definitions to access the QSPI F-RAM in PSoC 6 MCU 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 such cases, 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.

Related Documents

Code Examples/ Application Notes	
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	CE222460 provides a code example that implements the SPI host controller on PSoC 6 MCU using the SMIF Component and demonstrates accessing different features of the SPI F-RAM.
CE224073 – SPI F-RAM Access Using PSoC 6 MCU SMIF in Memory Mapped (XIP) Mode	SPI F-RAM Access Using PSoC 6 MCU SMIF in Memory Mapped (XIP) 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
Direct Memory Access	Supports data transfer independent of the CPU
Device Documentation	
PSoC 6 MCU: PSoC 63 with BLE Datasheet	PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
EXCELON-ULTRA 4-MBIT (512K X 8) QUAD SPI F-RAM Datasheet	
Development Kit Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	

Document History

Document Title: CE224034 – QSPI F-RAM Access Using PSoC 6 MCU with SMIF in XIP Mode

Document Number: 002-24034

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6267846	BFMC	07/31/2018	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 Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

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