Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

CE222460 - SPI F-RAM Access Using PSoC 6 MCU SMIF

Objective

CE222460 demonstrates accessing the SPI F-RAM™ using the Serial Memory Interface (SMIF) block in PSoC® 6 MCU and ModusToolbox™ IDE.

Requirements

Tool: ModusToolbox IDE 1.1

Programming Language: C

Associated Parts: All PSoC 6 MCU parts

Related Hardware: PSoC 6 WiFi-BT Pioneer Kit (CY8CKIT-062-WiFi-BT)

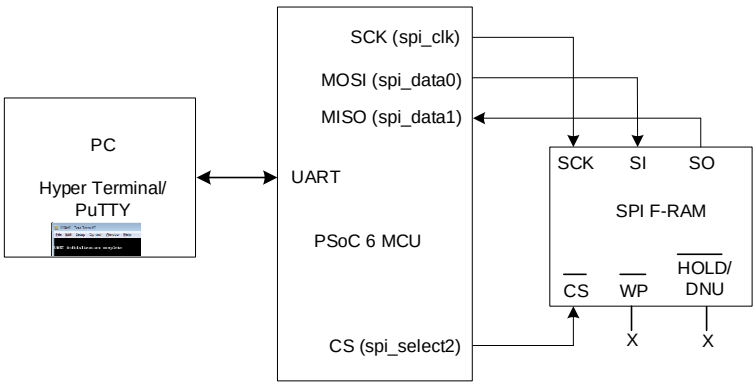
Overview

CE222460 provides a code example that implements the SPI host controller on PSoC 6 MCU using the SMIF resource and demonstrates accessing different features of an external SPI F-RAM using ModusToolbox IDE. The result is displayed by driving the RGB LED which turns green when the result is a pass and turns red when the result is a fail. The code example also enables the UART interface to connect to a PC to monitor the result.

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 any PSoC 6 MCU DVK connected to an external SPI F-RAM via jumper wires. This example uses the PSoC 6 WiFi-BT Pioneer kit's default configuration.

Figure 1. Hardware Setup Block Diagram



Note: The $\overline{WP}$ and $\overline{HOLD}$ pins, as applicable, are not controlled by PSoC 6 MCU in this code example; therefore, they are left floating (internally pulled HIGH). Some SPI F-RAMs do not provide internal pull-up on its $\overline{WP}$ and $\overline{HOLD}$ pins; in that case, the $\overline{WP}$ and $\overline{HOLD}$ pins must be driven to a logic HIGH externally either via pull-up or a GPIO for proper device operation. See the respective SPI F-RAM datasheet for details.

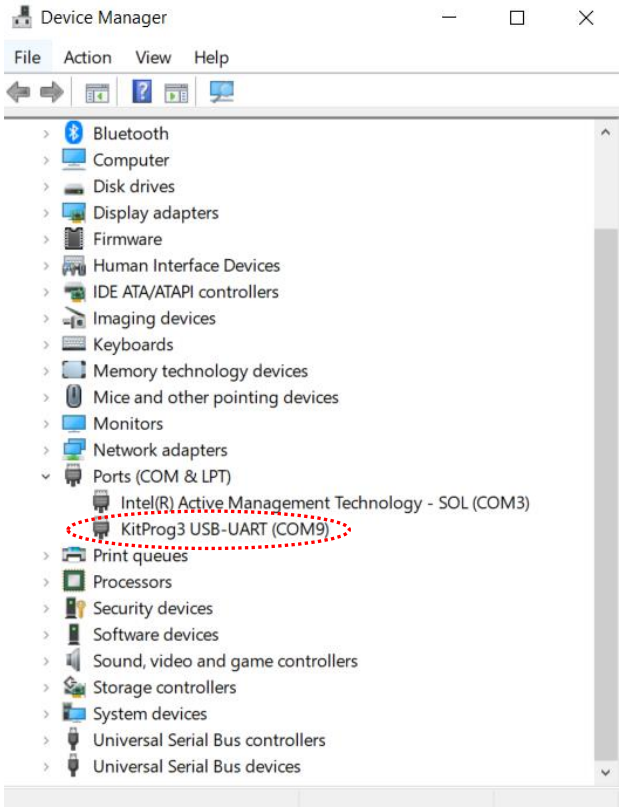
Note: PSoC 6 BLE and PSoC 6 WiFi-BT Pioneer kits are shipped with KitProg2. ModusToolbox only works with KitProg3. Therefore, make sure that the kit is upgraded to KitProg3 before using this code example. See ModusToolbox **Help** > **ModusToolbox IDE Documentation** > **User Guide**; section "PSoC 6 MCU KitProg Firmware Loader". If you do not upgrade, you will see an error like "unable to find CMSIS-DAP device" or "KitProg firmware is out of date".

Software Setup

This section demonstrates the procedure to setup the serial (UART) connection using PuTTY on 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 PSoC 6 Pioneer Kit to the PC using USB cable. The kit enumerates as KitProg3 USB-UART and is available under the **Device Manager** > **Ports (COM & LPT)**. A communication port (COMx) is assigned to KitProg3 USB-UART; for example, COM9 is assigned to PSoC 6 Pioneer Kit on the sample setup, shown in Figure 2.

Figure 2. KitProg3 USB-UART in Device Manager

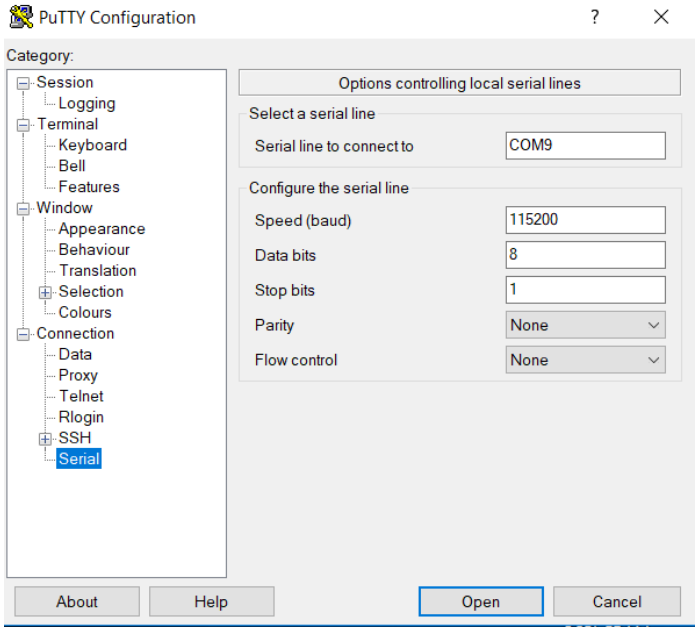


2. After you download and install PuTTY, double-click the PuTTY icon and select **Serial** under **Connection**.

3. A new window, as shown in Figure 3, opens where the communication port can be selected. Do the following in the **Options controlling local serial lines** section:

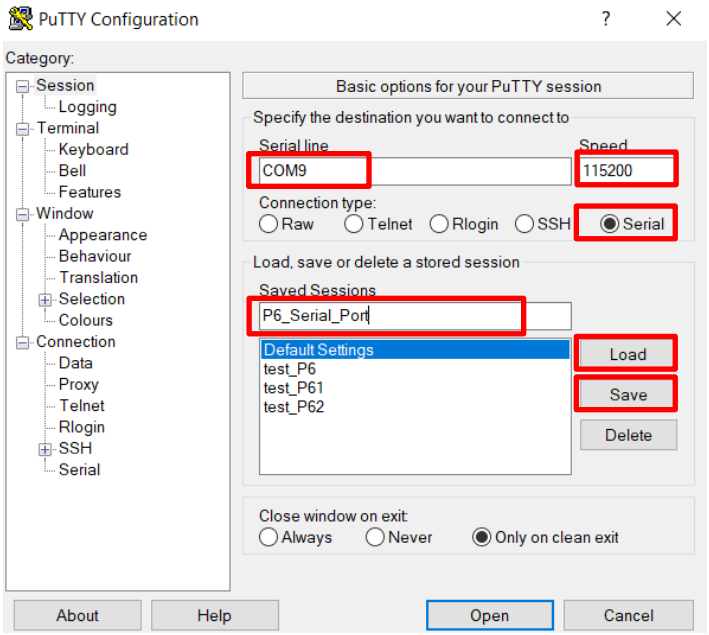
   - Enter the PSoC 6 Port (COM & LPT), COMx, in **Serial line to connect to**. This code example uses **COM9**. Verify the COM setting for your setup and select the appropriate COMx.
   - Enter **Speed (baud):** 115200, **Data bits:** 8, and **Stop bits:** 1.
   - Set **Parity** and **Flow control to** None.

Figure 3. Open New Connection



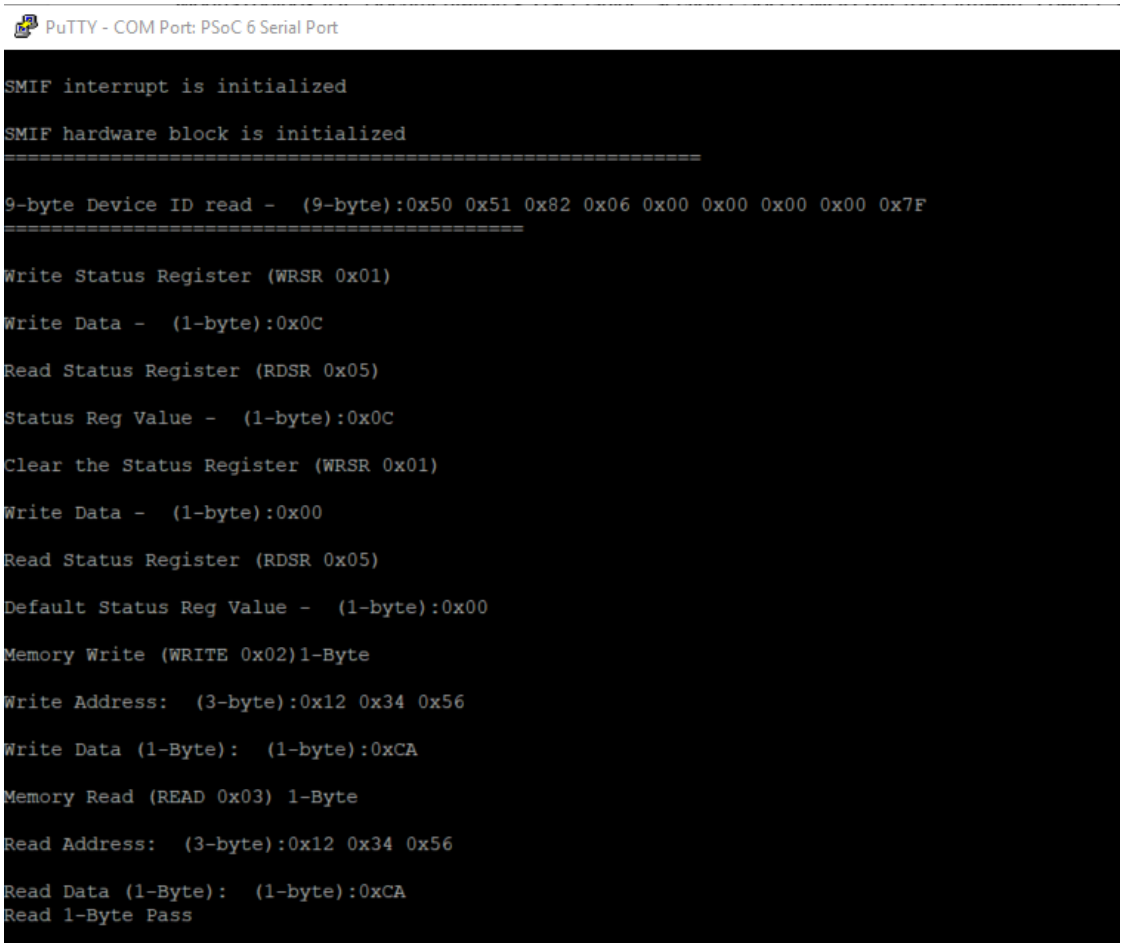
4. Select **Session** under **Category.** Select **Serial** under **Connection type** as shown in Figure 4. You can save this current session and **load** the settings when required. Enter a name in **Saved Sessions** and click **Save**. Click **Open** to proceed.

Figure 4. Select Communication Type in PuTTY



5. The COM terminal window then displays the code example results as shown in Figure 5. You may have to reprogram the PSoC 6 MCU device with the code example or reset the PSoC 6 MCU device (already programmed) to restart code execution and monitor the result.

Figure 5. Result Displayed on PC Terminal



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

Operation

This code example demonstrates the following features of the SPI F-RAM:

- Write and read access to the Status Register
- Random (1-Byte) memory write, read, and verify from an address
- Burst (256-Byte) memory write, read, and verify from a start address
- Burst (256-Byte) special sector write, burst read, and verify from an address
- Serial Number write, read, and verify

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

1. Connect the CY8CKIT-062-WiFi-BT 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, Parity and Flow control: None.
3. Import the application into a new workspace. See KBA225201.

4. Build and program the application into the CY8CKIT-062-WiFi-BT Kit or CY8CKIT-062-BLE Kit which has serial F-RAM mounted on it.
5. Observe the result by monitoring the RGB LED. The LED toggles green when result is a pass and red when result is a fail.
6. Observe the result through UART message printed in the terminal window. Figure 5 shows a snapshot of a sample UART terminal output.

Debugging

You can debug the example to step through the code. Use the **Debug (KitProg3)** configuration. See KBA224621 to learn how to start a debug session with ModusToolbox IDE.

Design and Implementation

The Quad Serial Memory Interface (QSPI) resource implements a SPI-based communication for interfacing an external SPI F-RAM devices with PSoC 6 MCU. The QSPI resource is configured as the SPI with two data lines (MISO, MOSI), one SPI clock (SCK), and single slave select (SS/CS). This example executes a burst write of 256-byte data to an external SPI F-RAM. The written data is read back to check its integrity. The UART resource outputs debug/result information to a terminal window. A user LED (RGB) indicates the status of read and write operation.

Resources

Table 1 and Table 2 list the PSoC 6 MCU resources and their utilization in this code example.

Table 1. ModusToolbox Resources - Peripherals

Resource	Alias	Purpose	Non-Default Settings
Quad Serial Memory Interface (QSPI) 0	KIT_FRAM	Configured as the SPI host controller to communicate with the SPI F-RAM	Figure 6
Serial Communication Block (SCB) 5	KIT_UART	To print output results on a terminal window	Figure 7

Table 2 lists the resources used in this example, and how they are used in the design.

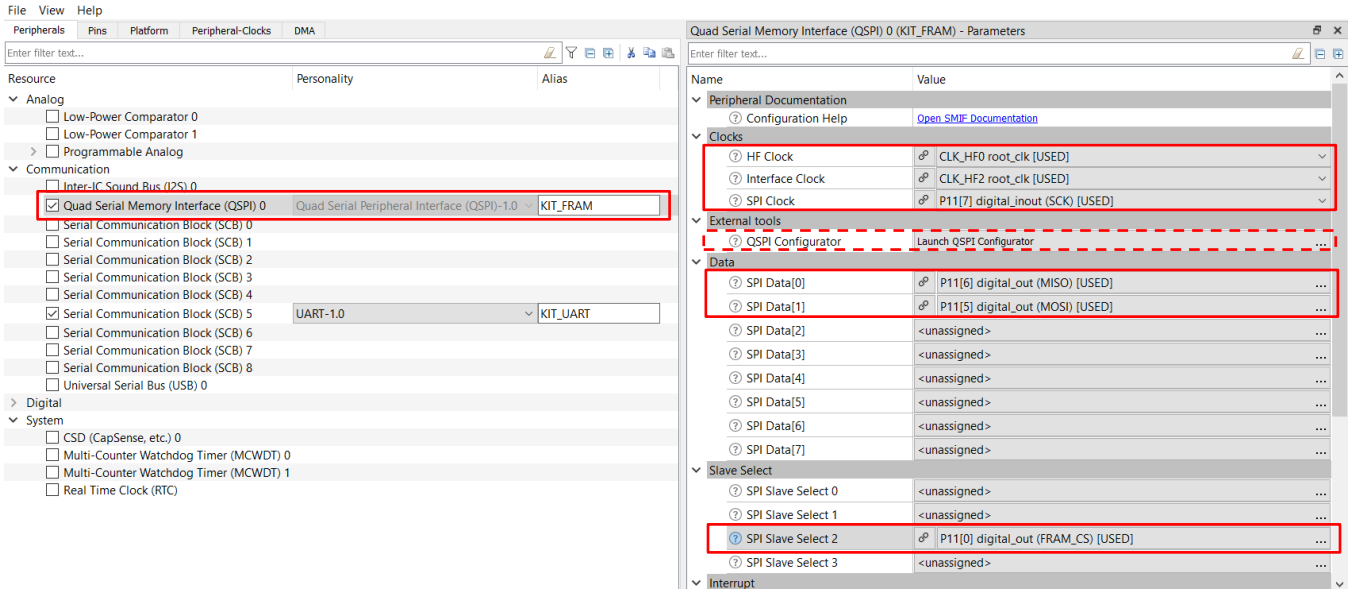
Table 2. ModusToolbox Resources - Pins

Resource	Port/Alias	Pin Drive Mode	Purpose	Non-Default Settings
GPIO – Port 11	P11[0]/FRAM_CS	Strong Drive, Input buffer off	–	Figure 8
	P11[5]/MOSI	Strong Drive, Input buffer on		
	P11[6]/MISO	Strong Drive, Input buffer on		
	P11[7]/SCK	Strong Drive, Input buffer off		
GPIO – Port 5	P5[0]/ KIT_UART_RX	Digital High-Z, input buffer on	Receives/transmits data packets from/to PC terminal	Figure 9
	P5[1]/ KIT_UART_TX	Strong Drive, Input buffer off		
GPIO – Port 0	P0[3]/RGB_RED	Strong Drive, Input buffer off	Drives the GPIO to glow red of the RGB LED to indicate the fail status	Figure 10
GPIO – Port 1	P1[1]/RGB_GREEN	Strong Drive, Input buffer off	Drives the GPIO to glow green of the RGB LED to indicate the pass status	Figure 11

Parameter settings

Non-default settings for each resource is outlined in red in the following figures. Figure 6 and Figure 7 show the resource parameter settings for QSPI 0 and SCB 5 block to enable the SPI host and UART.

Figure 6. QSPI 0 (KIT_FRAM) Resource Parameter Settings



This code example doesn't use the external tool "QSPI Configurator", outlined as above with the dotted red line.

Figure 7. SCB 5 (KIT_UART) Resource Parameter Settings

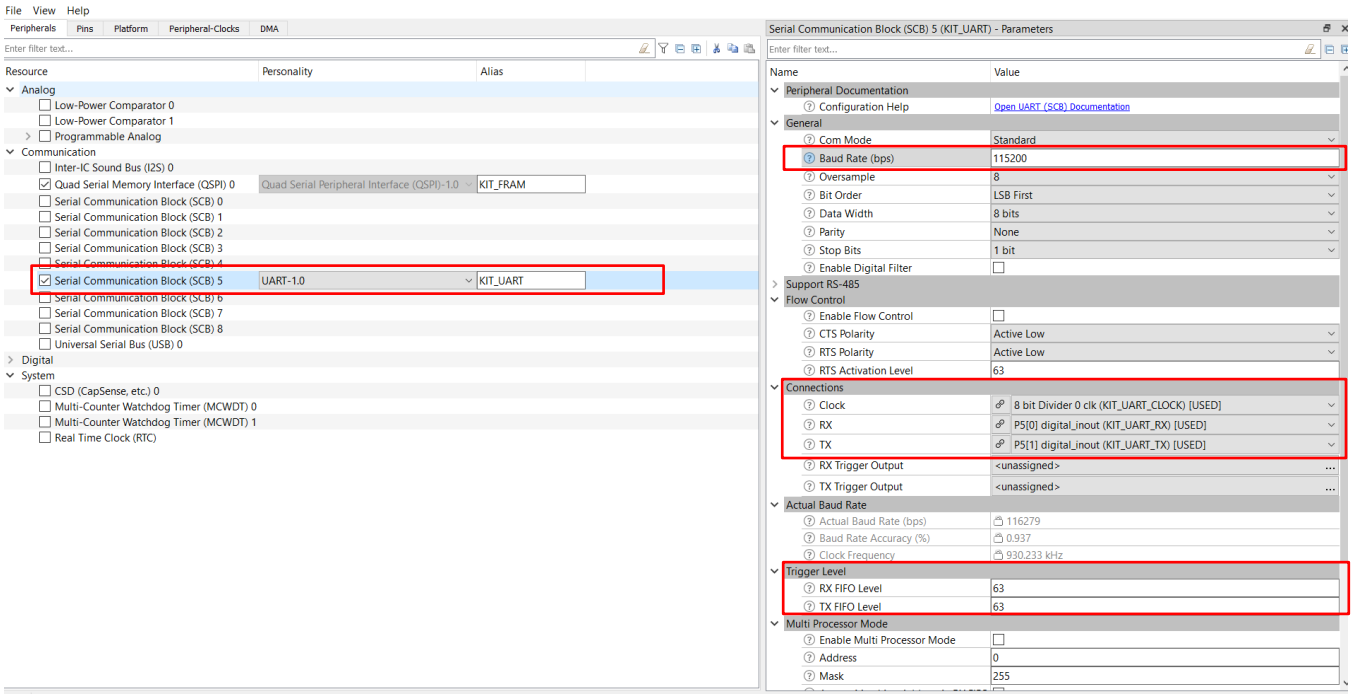


Figure 8 and Figure 9 show the QSPI and UART port pin and drive mode settings. Refer Table 2 for the drive mode setting for each PSoC 6 MCU pin used in this code example.

Figure 8. Port 11 (QSPI Controls) Resource Parameter Settings

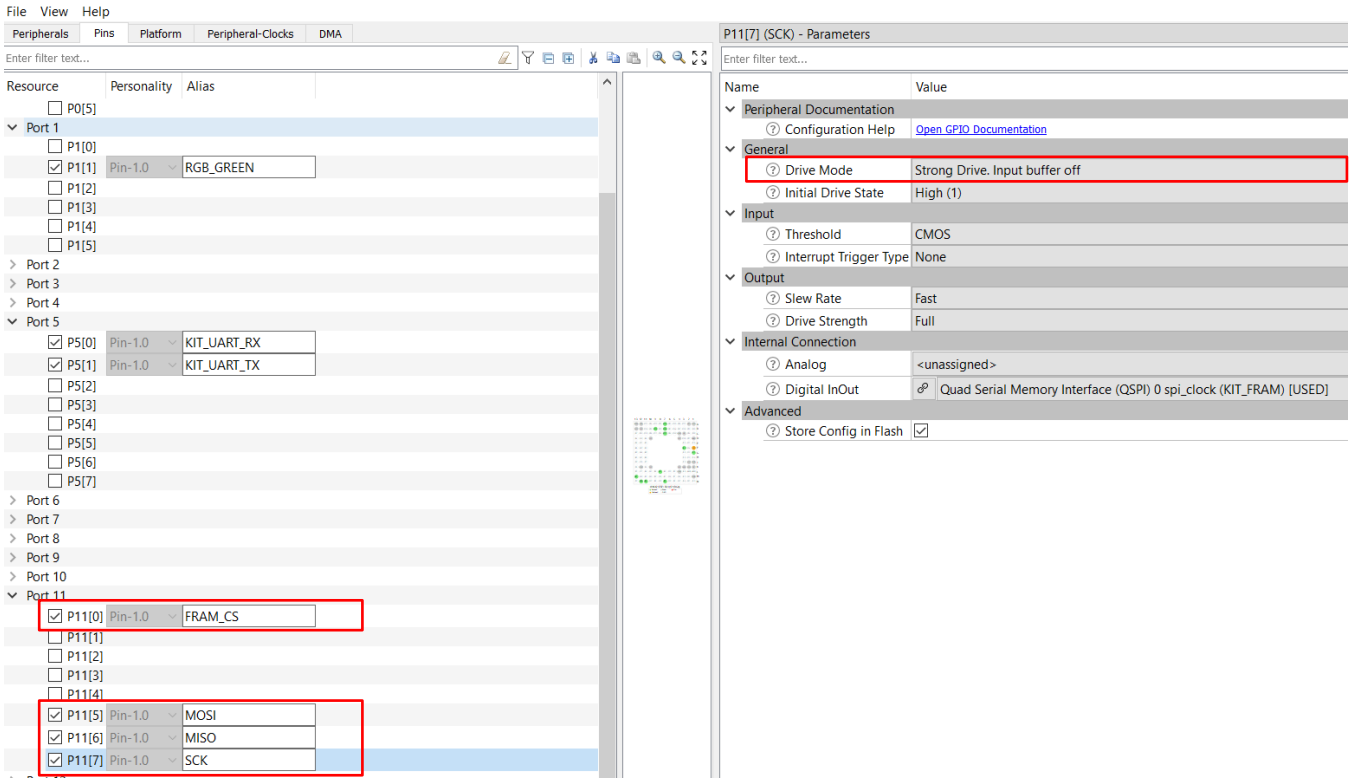


Figure 9. Port 5 (KIT_UART Control) Resource Parameter Settings

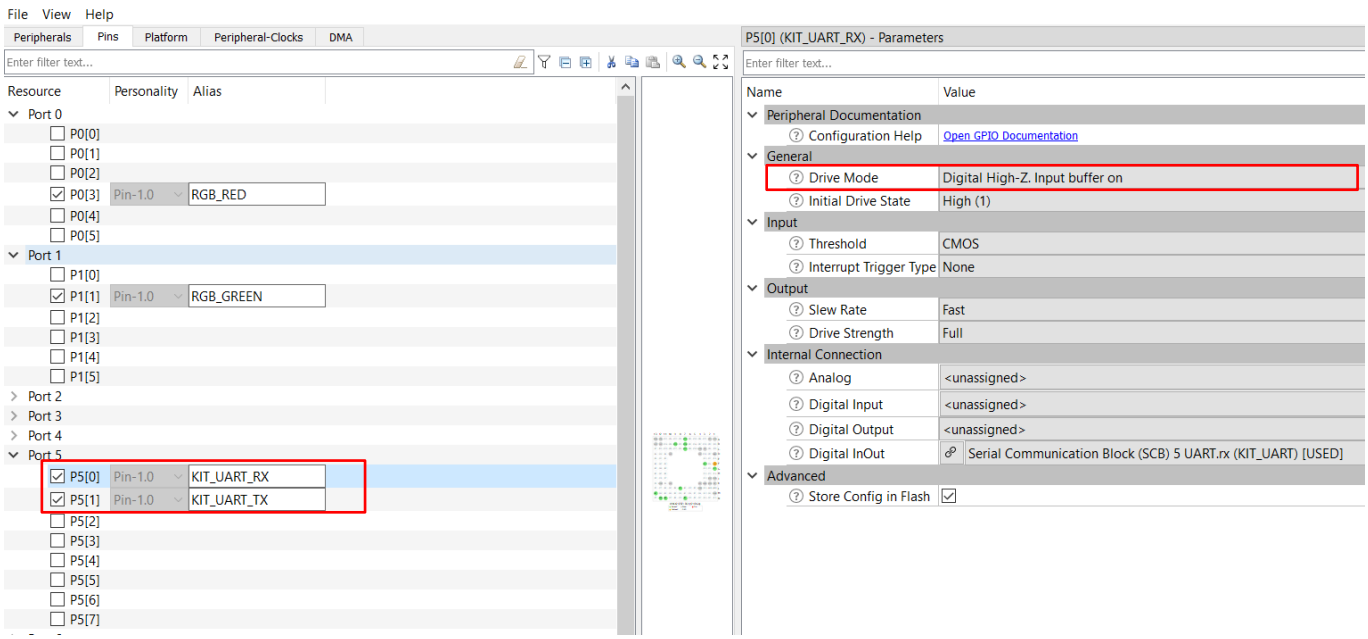


Figure 10 and Figure 11 show the RGB LED port pin and drive strength settings to drive the RGB LED red when the code executes with fail output and drives green when the code executes with pass (expected) output.

Figure 10. P0[3] (RGB_RED) Resource Parameter Settings

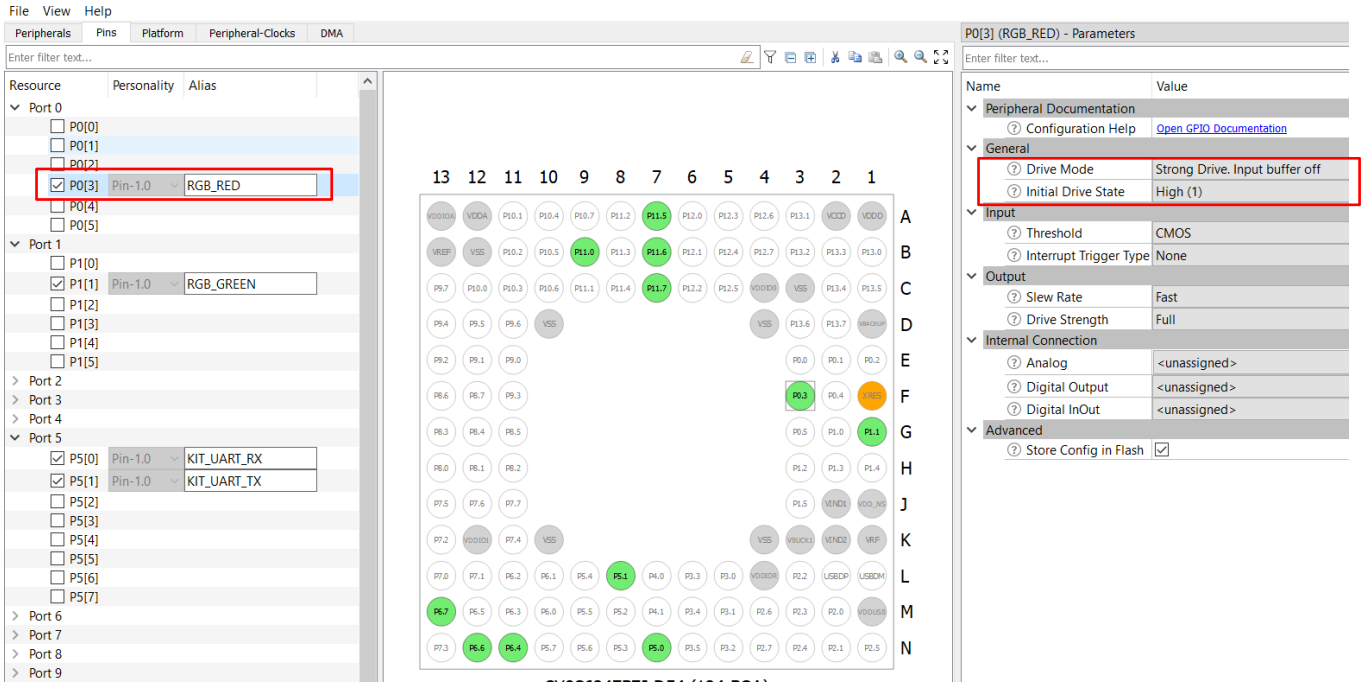


Figure 11. P1[1] (RGB_GREEN) Resource Parameter Settings

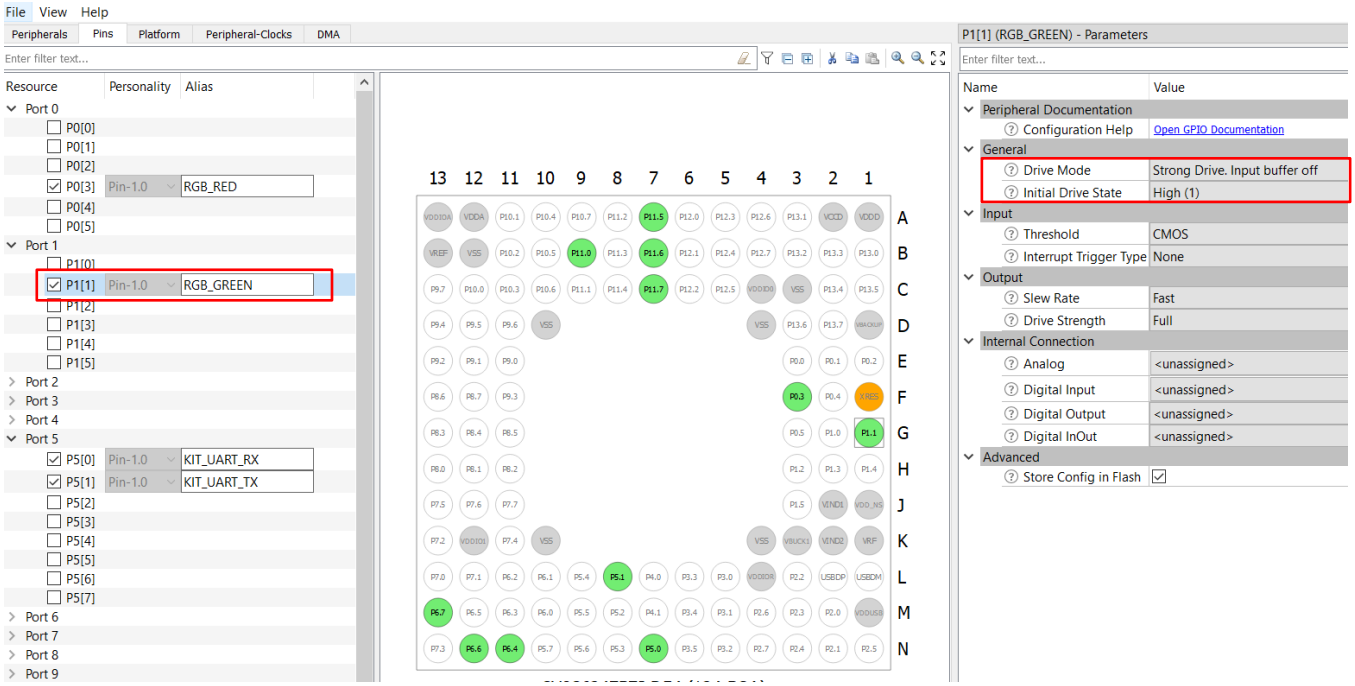


Figure 12 and Figure 13 show the 40-MHz SPI clock (CLK_HF2) setting. Cypress Excelon™ SPI F-RAMs can operate up to 50 MHz (max).

Figure 12. CLK_HF2 Resource Parameter Settings – PLL Frequency

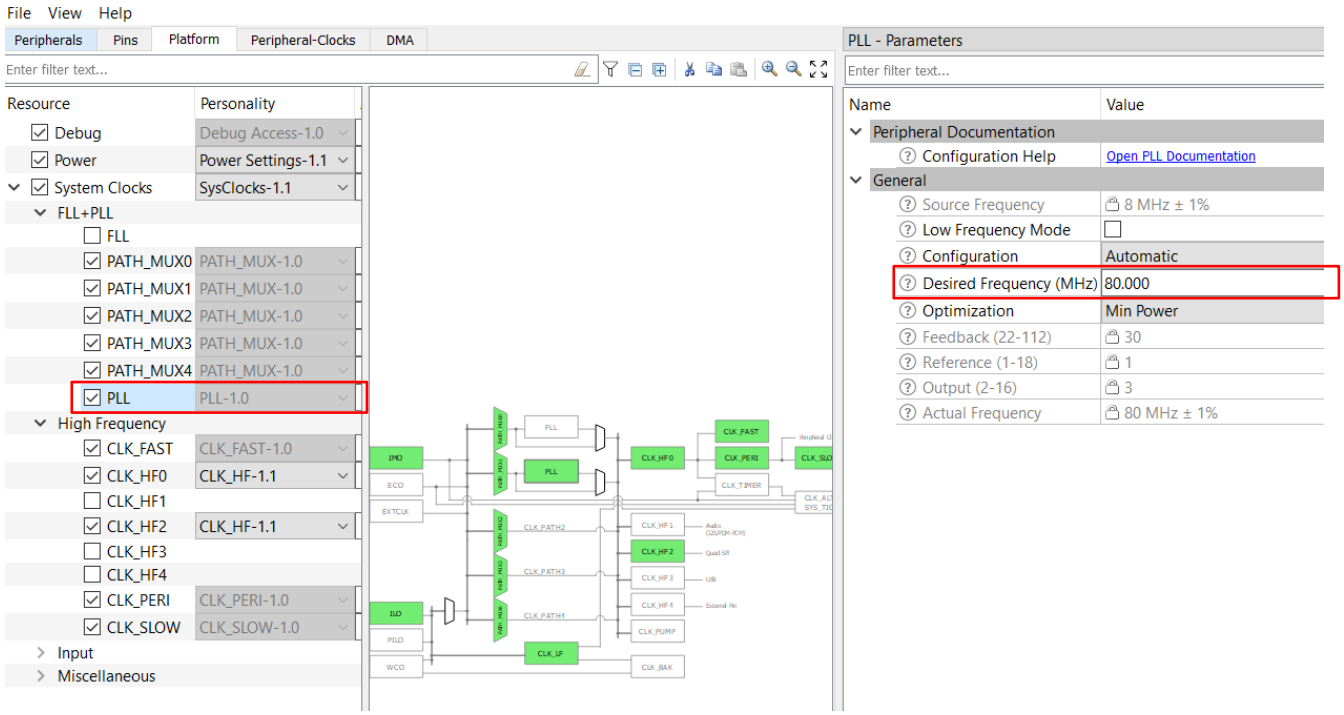
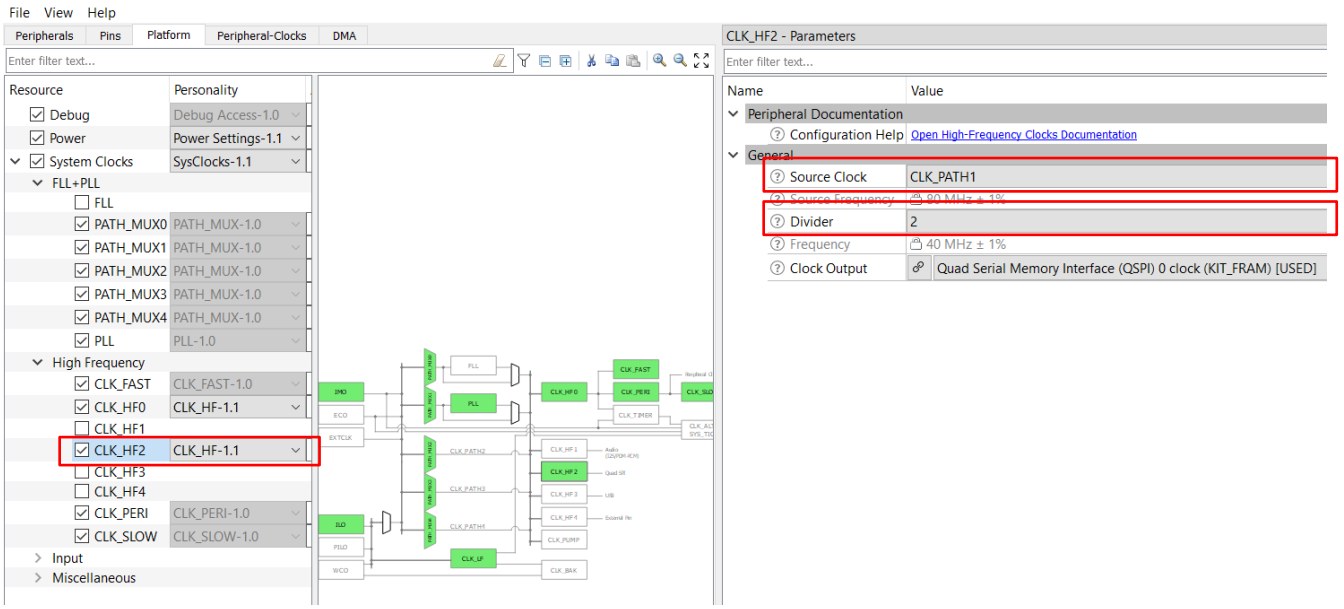


Figure 13. CLK_HF2 Resource Parameter Settings – Source Clock and Divider



Note: The PLL frequency change from its default (144-MHz) may impact the clock for other peripherals and generate clock constrain errors while applying changes through. In that case, clock constrain should be fixed either by changing the *Connections->Clock* option (*8 bit Divider 0 clk*, *8 bit Divider 1 clk,… 16 bit Divider clk 0*,.. etc.) available under the Clock setting of respective peripheral (through **Peripherals** tab of *design.modus*) or adjusting the selected clock *Divider* value through **Peripheral-Clocks** tab of *design.modus*. Errors related to any clock constrain are issued while saving the Device Configurator settings **(File** > **Save)**.

Reusing This Example

This code example is designed for the CY8CKIT-062-WiFi-BT Pioneer Kit. To use the design on a different PSoC 6 MCU kit, import the application for that kit. If you are unsure how to import an application, see KBA225201. Changing to a different kit may require you to reassign pins. Table 3 lists the target PSoC 6 MCU kits with PSoC 6 MCU manufacturing part no on it.

Table 3. PSoC 6 MCU Kits to PSoC 6 Device Used

PSoC 6 MCU Kit Name	PSoC 6 MCU Device Used
CY8CKIT-062-4343W	CY8C624ABZI-D44
CY8CKIT-062-WiFi-BT	CY8C6247BZI-D54
CY8CKIT-062-BLE	CY8C6347BZI-BLD53
CY8CMOD-062-4343W	CY8C624ABZI-D44
CY8CPROTO-062-4343W	CY8C624ABZI-D44

In some cases, a resource used by a code example (for example, a peripheral) 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 device datasheet for information on specific resource the device supports.

Related Documents

For a comprehensive list of PSoC 6 MCU resources, see KBA223067 in the Cypress community.

Application Notes	
AN218375 - Designing with Cypress Quad SPI (QSPI) F-RAM™	Provides functional details, timing, and example code for SPI F-RAMs.
AN221774 – Getting Started with PSoC 6 MCU	Describes PSoC 6 MCU devices and how to build your first ModusToolbox application and PSoC Creator project.
AN215656 – PSoC 6 MCU: Dual-CPU System Design	Describes the dual-CPU architecture in PSoC 6 MCU and shows how to build a simple dual-CPU design.
AN304 – SPI Guide for F-RAM™	Provides functional details, timing, and example code for SPI F-RAMs.
Code Examples	
CE220823 - PSoC 6 MCU SMIF Memory Write and Read Operation	Demonstrates interfacing with an external NOR flash memory in QSPI mode using the SMIF block in PSoC 6 MCU.
Visit the Cypress GitHub site for a comprehensive collection of code examples using ModusToolbox IDE	
Device Documentation	
PSoC 6 MCU Datasheets	PSoC 61, PSoC 62, PSoC 63 MCU Datasheets
SPI F-RAM (CY15B104QN) Datasheet	1.8-3.6 V (3.3 V typ.), 50 MHz, 4Mb SPI F-RAM datasheet
Development Kit Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	
CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit	
CY8CPROTO-062-4343W PSoC 6 Wi-Fi BT Prototyping Kit	
CY15FRAMKIT-002 Development Kit	
Tool Documentation	
ModusToolbox	The Cypress IDE for IoT designers

Document History

Document Title: CE222460 – SPI F-RAM Access Using PSoC 6 MCU SMIF

Document Number: 002-28033

Revision	ECN	Submission Date	Description of Change
**	6667788	09/05/2019	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