

## Objective

This example demonstrates interfacing with an external NOR flash memory in Quad-SPI mode 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 BLE Pioneer Kit](#), [PSoC 6 Wi-Fi-BT Pioneer Kit](#), [PSoC 6 Wi-Fi BT Prototyping Kit](#)

## Overview

Demonstrates interfacing with an external NOR flash memory using the Serial Memory Interface (SMIF) block in Quad Serial Peripheral Interface (QSPI) mode. This example also checks the integrity of the read data against written data.

## Hardware Setup

This example uses the PSoC 6 Wi-Fi-BT Pioneer Kit's default configuration. Refer to the kit guide to ensure the kit is configured correctly. You can also use PSoC 6 BLE Pioneer Kit or PSoC 6 Wi-Fi BT Prototyping Kit by importing the application for that kit.

**Note:** The PSoC 6 BLE Pioneer kit and the PSoC 6 Wi-Fi-BT Pioneer kit ship with KitProg2. ModusToolbox only works with KitProg3. Before using this code example, make sure that the kit is upgraded to KitProg3. 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 If you do not upgrade, you will see an error like "unable to find CMSIS-DAP device" or "KitProg firmware is out of date".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 example uses a serial terminal program. Install one on your PC if you don't have one. The instructions use [Tera Term](#).

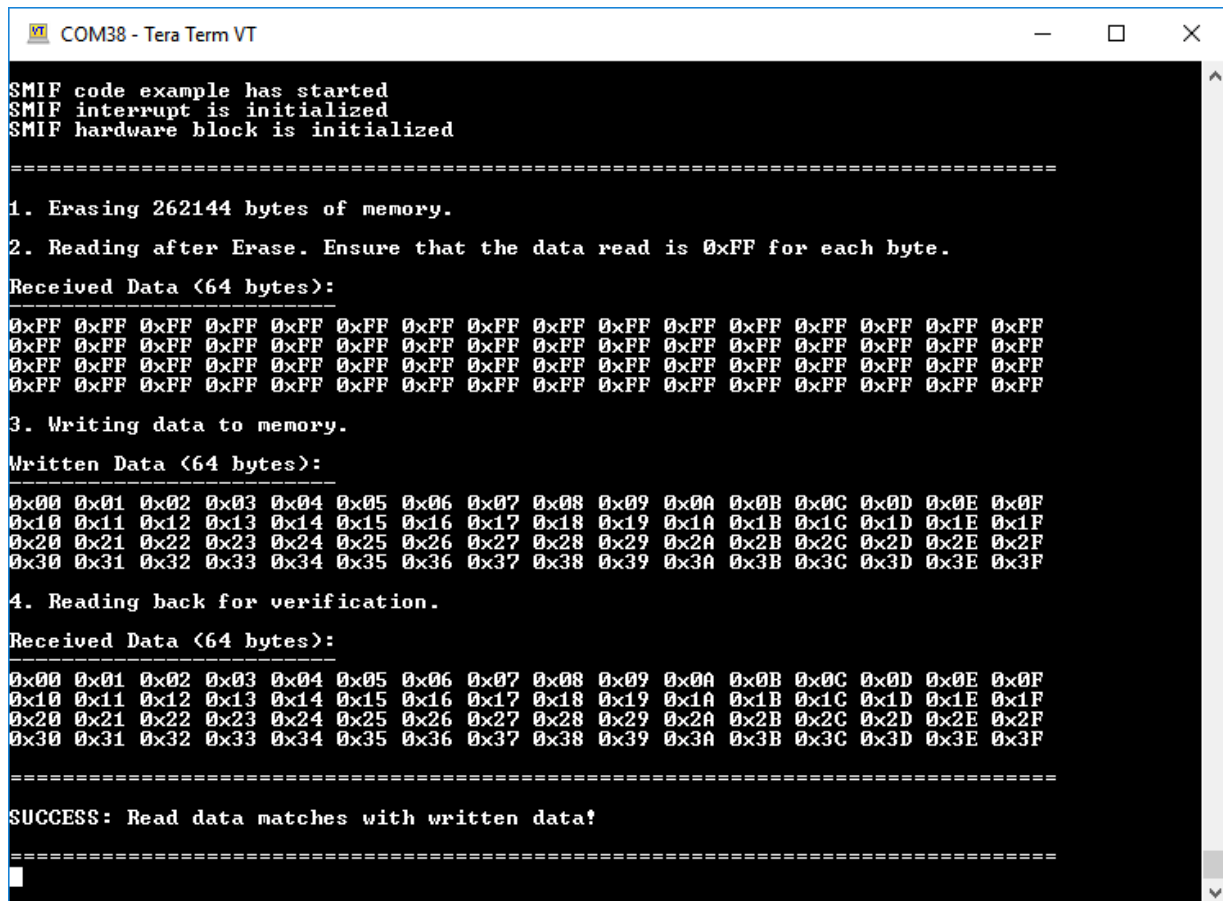
## Operation

1. Connect the Pioneer board to your PC using the provided USB cable through the USB connector.
2. Open a terminal program and select the KitProg COM port. Set the other serial port parameters as follows:
  - a) Baud Rate: 115200
  - b) Data: 8 bits
  - c) Parity: None
  - d) Stop: 1 bit
  - e) Flow Control: None
3. Import the application into a new workspace. See [KBA225201](#).
4. Build the application. Choose **Project > Build All**.
5. Program the PSoC 6 MCU device. Select the **mainapp** project. In the **QuickPanel**, scroll down and click the **Program Kitprog3** item.

6. Press and release the USER button on the kit when prompted on the terminal window. This is required only the first time the example is run on a kit to enable Quad mode in the memory. Enabling Quad mode is required for communicating in QSPI mode.
7. Observe the KIT\_LED1 to determine the status of the SMIF operation.
  - a) LED is blinking: Successful operation
  - b) LED is on : Failed operation

Make sure that debug messages displayed in the terminal window are as expected. [Figure 1](#) is a snapshot of the serial terminal output.

Figure 1. Serial Terminal Output



```

SMIF code example has started
SMIF interrupt is initialized
SMIF hardware block is initialized

=====

1. Erasing 262144 bytes of memory.

2. Reading after Erase. Ensure that the data read is 0xFF for each byte.
Received Data <64 bytes>:
0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF
0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF
0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF
0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF

3. Writing data to memory.
Written Data <64 bytes>:
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

4. Reading back for verification.
Received Data <64 bytes>:
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

=====

SUCCESS: Read data matches with written data!

=====
  
```

## 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 QSPI resource implements a SPI-based communication for interfacing external memory devices with PSoC. QSPI resource is configured with four data lines and single slave select line. This example writes 64 bytes of data to external memory in Quad SPI mode. The written data is read back to check its integrity. The UART resource outputs debug information to a terminal window. A user LED indicates the status of read and write operation.

The firmware uses source code (*cycfg\_qspi\_memslot.c* and *cycfg\_qspi\_memslot.h* files) generated from the QSPI Configurator. This source code defines data structures that hold the memory configuration.

## Resources

Table 1 lists the resources used in this example, and how they are used in the design.

Table 1. ModusToolbox Resources

Resource	Alias	Purpose	Non-Default Settings
Quad Serial Memory Interface (QSPI) 0	KIT_QSPI	Communicate with QSPI NOR Flash	Figure 2, Figure 3
Serial Communication Block (SCB) 5	KIT_UART	UART to transmit debug data	Figure 4
General Purpose Input / Output (GPIO)	KIT_BTN1	User button for enabling Quad mode	Figure 5
General Purpose Input / Output (GPIO)	KIT_LED1	LED to indicate SMIF transfer status	Figure 6

## Parameter Settings

Non-default settings for each resource is outlined in red in the following figures.

Figure 2 shows the KIT\_QSPI resource parameter settings.

ModusToolbox supports a stand-alone application called QSPI Configurator (star-marked section in Figure 2), which enables a user to configure the memory commands through a Graphical User Interface (GUI). This application is invoked from the *Parameters* pane of the QSPI resource under *Peripherals* tab. Configure the device as shown in the below image. Save the file in the *GeneratedSource* folder.

Figure 2. QSPI Resource Parameter Settings

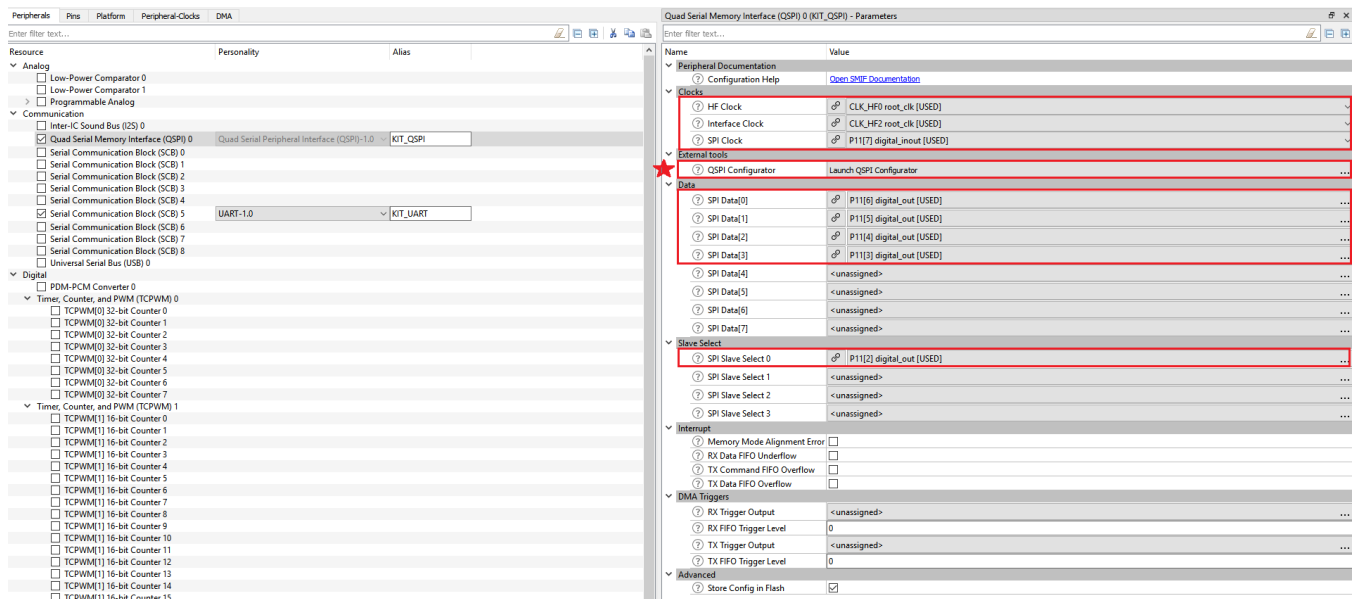


Figure 3 shows the QSPI Configurator.

Figure 3. QSPI Configurator

File Options Help

PSoC 6 ▾

Slave Slot	Memory Part Number	Data Select	Memory Mapped	Pair With Slot	Start Address	Size	End Address	Write Enable	Config Data In Flash	Encrypt
0	S25FL512S ▾	Quad SPI-Data[0:3] ▾	<input checked="" type="checkbox"/>	None ▾	0x18000000	0x10000	0x1800FFFF	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
1	Not used ▾	SPI-MOSI:MISO Data[0:1] ▾	<input type="checkbox"/>	None ▾	0x18010000	0x10000	0x1801FFFF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	Not used ▾	SPI-MOSI:MISO Data[0:1] ▾	<input type="checkbox"/>	None ▾	0x18020000	0x10000	0x1802FFFF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	Not used ▾	SPI-MOSI:MISO Data[0:1] ▾	<input type="checkbox"/>	None ▾	0x18030000	0x10000	0x1803FFFF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Location: C:/Users/yejt/ModusToolbox\_1.0/tools/qspi-configurator-1.0/data/memory/S25FL512S.cymem

User part number: S25FL512S Erase time: 520 ms ▾

Status register busy mask: 0x01 Chip erase time: 134 s ▾

Status register quad enable mask: 0x02 Program time (μs): 340

Size of memory: 0x04000000 Description: 64Mbytes 3V serial Flash memory

Program page size: 0x00000200

Erase block size (bytes): 0x00040000

Number of address bytes for SPI transactions: 0x03

SPI modes supported: ☐ Single ☐ Dual ☒ Quad ☐ Octal

Single SPI Commands								Dual SPI Commands								Quad SPI Commands								Octal SPI Commands							
Description	Number	Command Width	Address Width	Mode	Mode Width	Dummy Cycles	Data Width																								
Read command format	0xEB	Single ▾	Quad ▾	0x01	Quad ▾	4	Quad ▾																								
Write enable command format	0x06	Single ▾	Single ▾	NA	Single ▾	NA	Single ▾																								
Write disable command format	0x04	Single ▾	Single ▾	NA	Single ▾	NA	Single ▾																								
Erase command format	0xD8	Single ▾	Single ▾	NA	Single ▾	NA	Single ▾																								
Chip erase command format	0x60	Single ▾	Single ▾	NA	Single ▾	NA	Single ▾																								
Program command format	0x38	Single ▾	Single ▾	NA	Quad ▾	NA	Quad ▾																								
Read status register command (containing QE bit)	0x35	Single ▾	Single ▾	NA	Single ▾	NA	Single ▾																								
Read status register command (containing WIP bit)	0x05	Single ▾	Single ▾	NA	Single ▾	NA	Single ▾																								
Write status register command (containing QE bit)	0x01	Single ▾	Single ▾	NA	Single ▾	NA	Single ▾																								

Figure 4. KIT\_UART Configuration

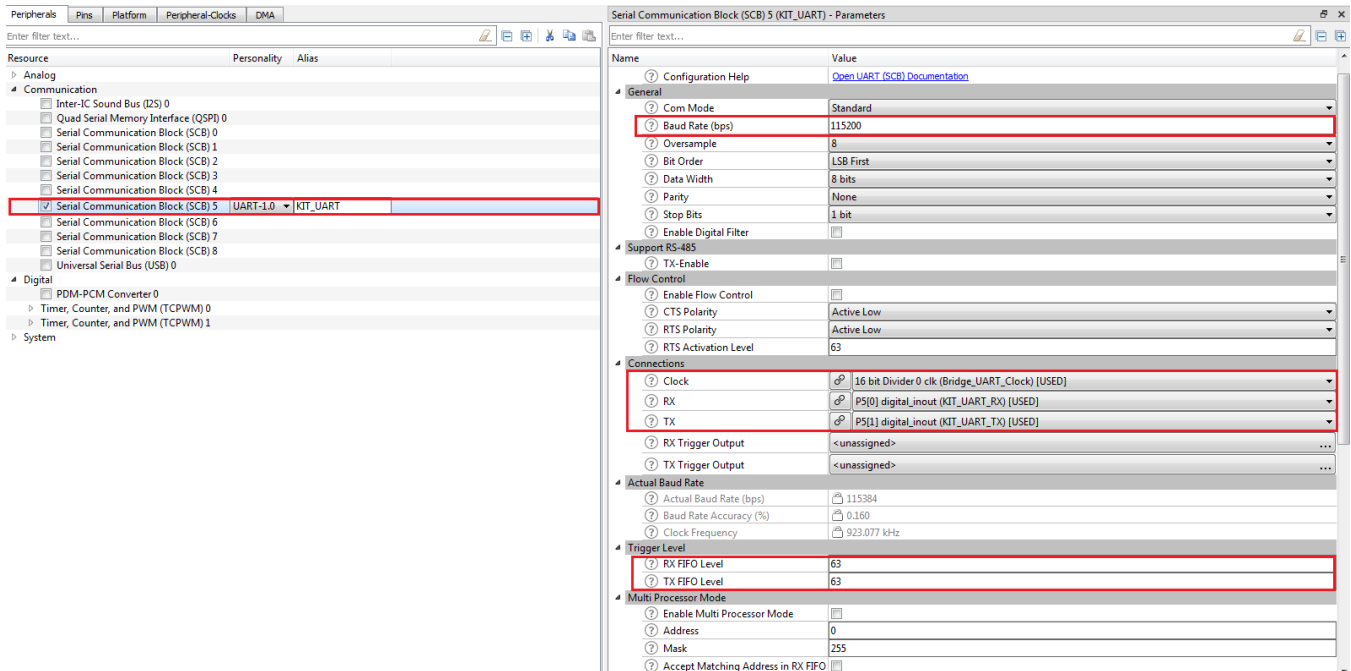


Figure 5. KIT\_BTN1 Configuration

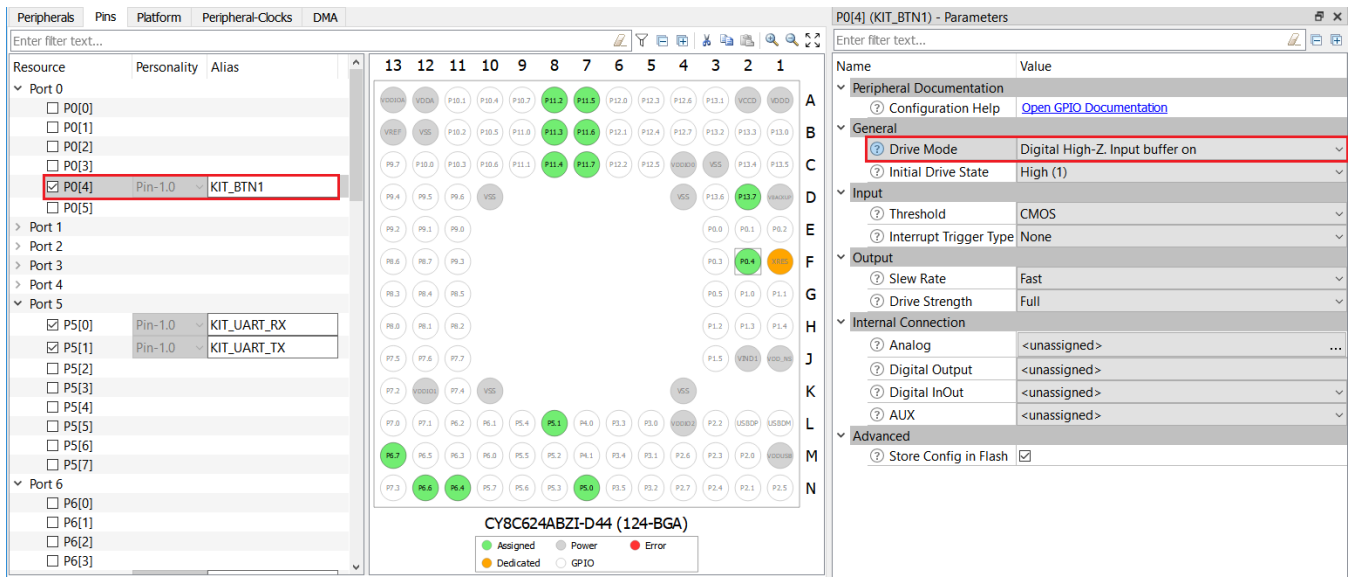
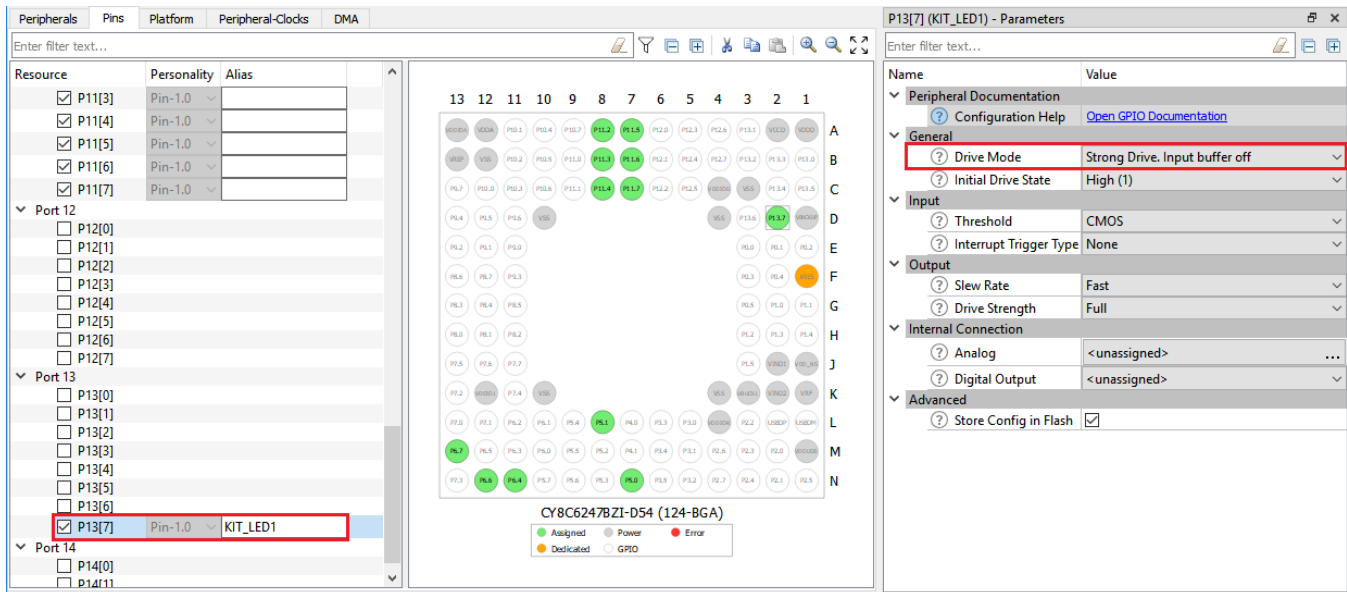


Figure 6. KIT\_LED1 Configuration



## Related Documents

For a comprehensive list of PSoC 6 MCU resources, see [KBA223067](#) in the Cypress community.

Application Notes	
<a href="#">AN210781</a> – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes PSoC 6 MCU with BLE Connectivity devices.
<a href="#">AN221774</a> – Getting Started with PSoC 6 MCU	Describes PSoC 6 MCU devices and how to build your first ModusToolbox application and PSoC Creator project.
<a href="#">AN215656</a> – 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.
Code Examples	
<a href="#">CE218472</a> - PSoC 6 MCU Comparing External Voltages Using a Low-Power Comparator	
Visit the <a href="#">Cypress GitHub site</a> for a comprehensive collection of code examples using ModusToolbox IDE	
Device Documentation	
<a href="#">PSoC 6 MCU: PSoC 63 with BLE Datasheet</a>	<a href="#">PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual</a>
Development Kit Documentation	
<a href="#">CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit</a>	
<a href="#">CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit</a>	
<a href="#">CY8CPROTO-062-4343W PSoC 6 Wi-Fi BT Prototyping Kit</a>	
Tool Documentation	
<a href="#">ModusToolbox</a>	The Cypress IDE for IoT designers

## Cypress Resources

Cypress provides a wealth of data at [www.cypress.com](http://www.cypress.com) to help you to select the right device, and quickly and effectively integrate the device into your design.

For the PSoC 6 MCU devices, see [KBA223067](#) in the Cypress community for a comprehensive list of PSoC 6 MCU resources.

## Document History

Document Title: CE220823 - PSoC 6 MCU SMIF Memory Write and Read Operation

Document Number: 002-25536

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6369719	YEKT	11/2/2018	Initial public release
*A	6484271	YEKT	4/16/2019	Code example updated for ModusToolbox 1.1
*B	6544943	VAIR	4/16/2019	Code example has been revised completely for clarity and correctness



## 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	<a href="http://cypress.com/arm">cypress.com/arm</a>
Automotive	<a href="http://cypress.com/automotive">cypress.com/automotive</a>
Clocks & Buffers	<a href="http://cypress.com/clocks">cypress.com/clocks</a>
Interface	<a href="http://cypress.com/interface">cypress.com/interface</a>
Internet of Things	<a href="http://cypress.com/iot">cypress.com/iot</a>
Memory	<a href="http://cypress.com/memory">cypress.com/memory</a>
Microcontrollers	<a href="http://cypress.com/mcu">cypress.com/mcu</a>
PSoC	<a href="http://cypress.com/psoc">cypress.com/psoc</a>
Power Management ICs	<a href="http://cypress.com/pmic">cypress.com/pmic</a>
Touch Sensing	<a href="http://cypress.com/touch">cypress.com/touch</a>
USB Controllers	<a href="http://cypress.com/usb">cypress.com/usb</a>
Wireless Connectivity	<a href="http://cypress.com/wireless">cypress.com/wireless</a>

## 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](http://cypress.com/support)

All other trademarks or registered trademarks referenced herein are the property of their respective owners.



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

© Cypress Semiconductor Corporation, 2018-2019. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress does not assume any liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit [cypress.com](http://cypress.com). Other names and brands may be claimed as property of their respective owners.