CE222306 – PSoC 4 I^{2}C Communication with Serial Communication Block (SCB)

Objective

This code example demonstrates the basic usage of the PSoC® 4 Serial Communication Block (SCB) Component in I^{2}C Master and I^{2}C Slave modes.

Overview

This code example consists of two projects:

- I2C_SCB_Slave – implements the I^{2}C Slave device to receive and execute commands from I^{2}C Master and controls the RGB LED color. I^{2}C Master can read the command execution result.
- I2C_SCB_Master – implements the I^{2}C Master device to send commands to the I^{2}C Slave device and read the status of the command execution: success or error. The RGB LED shows the result of the command execution: success – green; error – red.

The I2C_SCB_Slave can be controlled by the Bridge Control Panel via KitProg's USB-I^{2}C Bridge or by I2C_SCB_Master.

Requirements

Tool: PSoC Creator™ 4.2

Programming Language: C (Arm® GCC 5.4.1 and Arm MDK 5.22)

Associated Parts: All PSoC 4 parts

Related Hardware: CY8CKIT-040, CY8CKIT-041-40XX, CY8CKIT-041-41XX, CY8CKIT-042, CY8CKIT-042-BLE, CY8CKIT-042-BLE-A, CY8CKIT-044, CY8CKIT-046, CY8CKIT-048, CY8CKIT-149

Hardware Setup

This example project is configured by default to run on the CY8CKIT-042 development kit from Cypress Semiconductor. The project can be migrated to any supported kit by changing the target device with **Device Selector** called from the project's context menu. Table 1 lists the supported kits. You can use different PSoC 4 kits listed in Table 1 for master and slave projects.

This example uses the kit's default configuration with the VDD SELECT jumper set to 5V. Refer to the kit's guide to ensure that the kit is configured correctly.

Table 1. Supported Kits and Devices

Development Kit	Series	Device
CY8CKIT-040	PSoC 4000	CY8C4014LQI-422
CY8CKIT-041-40XX	PSoC 4000S	CY8C4045AZI-S413
CY8CKIT-041-41XX	PSoC 4100S	CY8C4146AZI-S433
CY8CKIT-042	PSoC 4200	CY8C4245AXI-483
CY8CKIT-042-BLE	PSoC 4200 BLE	CY8C4247LQI-BL483
CY8CKIT-042-BLE-A	PSoC 4200 BLE	CY8C4248LQI-BL483
CY8CKIT-044	PSoC 4200M	CY8C4247AZI-M485
CY8CKIT-046	PSoC 4200L	CY8C4248BZI-L489
CY8CKIT-048	PSoC Analog Coprocessor	CY8C4A45LQI-483
CY8CKIT-149	PSoC 4100S Plus	CY8C4147AZI-S475

The pin assignments for the supported kits are provided in Table 2. For these kits, the project includes control files to automatically assign pins with respect to the kit hardware connections during the project build. To change the pin assignments, override the control file selections in the Pin Editor of the Design Wide Resources by selecting the new port or pin number.

Table 2. Pin Assignments

Development Kit	Pin Assignment						
	Common for Both Projects		I2C Master		I2C Slave		
	\I2C:scl\	\I2C:sda\	LED_SUCCESS	LED_ERROR	LED_RED	LED_GREEN	LED_BLUE
CY8CKIT-040	P1[2]	P1[3]	P1[1]	P3[2]	P3[2]	P1[1]	P0[2]
CY8CKIT-041-40XX	P3[0]	P3[1]	P2[6]	P3[4]	P3[4]	P2[6]	P3[6]
CY8CKIT-041-41XX							
CY8CKIT-042	P3[0]	P3[1]	P0[2]	P1[6]	P1[6]	P0[2]	P0[3]
CY8CKIT-042-BLE	P3[5]	P3[4]	P3[6]	P2[6]	P2[6]	P3[6]	P3[7]
CY8CKIT-042-BLE-A							
CY8CKIT-044	P4[0]	P4[1]	P2[6]	P0[6]	P0[6]	P2[6]	P6[5]
CY8CKIT-046	P4[0]	P4[1]	P5[3]	P5[2]	P5[2]	P5[3]	P5[4]
CY8CKIT-048	P4[0]	P4[1]	P2[6]	P1[4]	P1[4]	P2[6]	P1[6]
CY8CKIT-149	P3[0]	P3[1]	P5[5]	P3[4]	P5[2]	P5[5]	P5[7]

Software Setup

This code example requires the Bridge Control Panel software shipped with the PSoC Creator. The configuration of the Bridge Control Panel is described in the "Operation" section.

Operation

This code example may be used in two configurations:

- One kit with the I2C Slave firmware controlled by Bridge Control Panel via KitProg's USB-I2C Bridge (Figure 1).
- Two kits with the I2C Master and I2C Slave firmware (Figure 2). You can use different PSoC 4 kits from Table 1. Supported Kits and Devices.

Figure 1. One Kit Operation

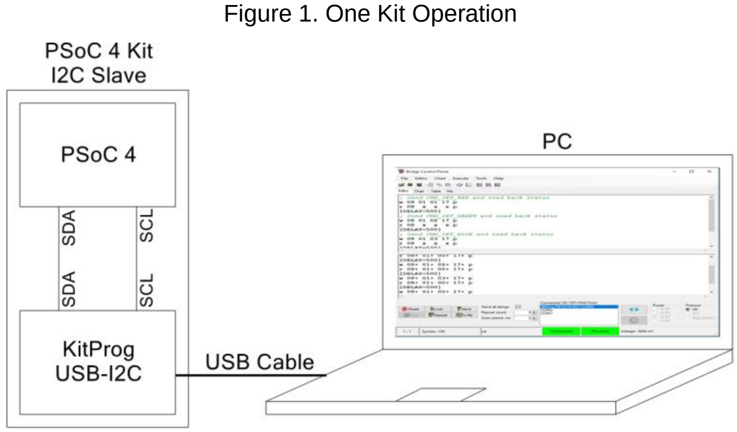
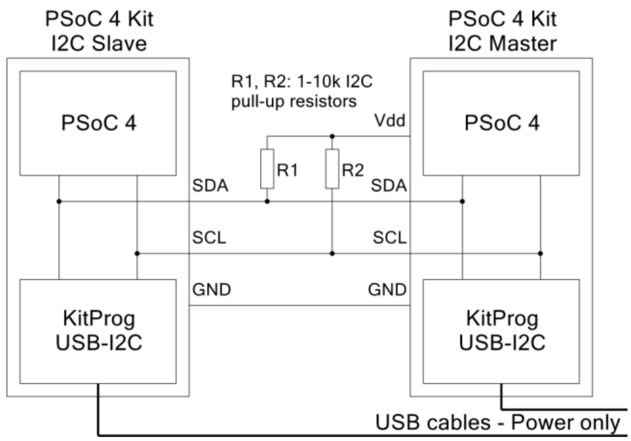


Figure 2. Two Kits Operation



I²C Slave with Bridge Control Panel Operation

1. Plug your kit board in into your computer's USB port.
2. Build the project and program it into the PSoC 4 devices. Choose **Debug** > **Program**. For more information on device programming, see the PSoC Creator Help.
3. Observe the green LED turns on to indicate successful program operation.
4. Open the Bridge Control Panel and select the KitProg of the kit in the **Connected I2C/SPI/RX8 Ports**. Ensure the selected protocol is I²C (Figure 3).
5. Go to **Tools** > **Protocol Configuration** and in the **I2C** tab select **I2C Speed** 100kHz (Figure 4).
6. Press the **List** button to ensure that the I²C Slave device with the address 0x08 (7-bits) is available for communication.

Note: Other I²C devices can be connected to the I²C bus. These devices' addresses are shown after the list operation completion. Refer to the development kit documentation for more information about other I²C devices available on the kit.

7. Go to **File** > **Open file** and open the *BCP_Master_I2cCmd.iic* file located in the *I2C_SCB_Slave* project folder. This file contains the I²C Master commands for communication with the I²C Slave. The commands appear in the Edit window.
8. Select the **Send all strings** checkbox, press the **Repeat** button and observe the RGB LED color changes: red > green > blue > OFF and then repeats. The Bridge Control Panel will show the execution status of the sent command (Figure 5).

Note: To send commands one-by-one, disable the **Send all strings** checkbox, set the cursor to the line with the needed command and press the **Send** button.

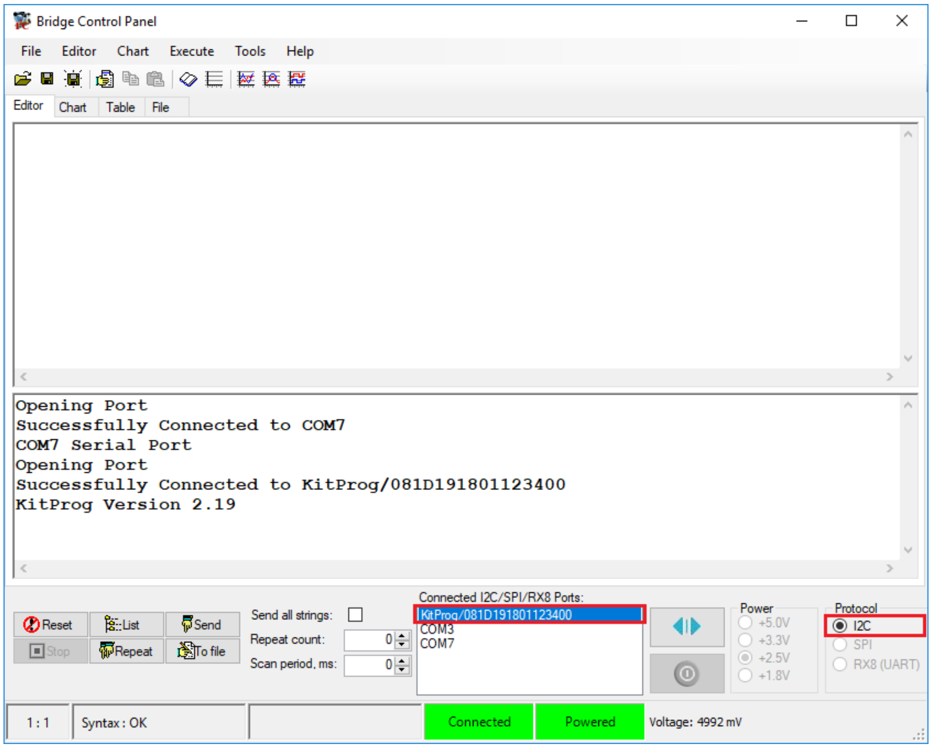


Figure 3. Bridge Control Panel Setup

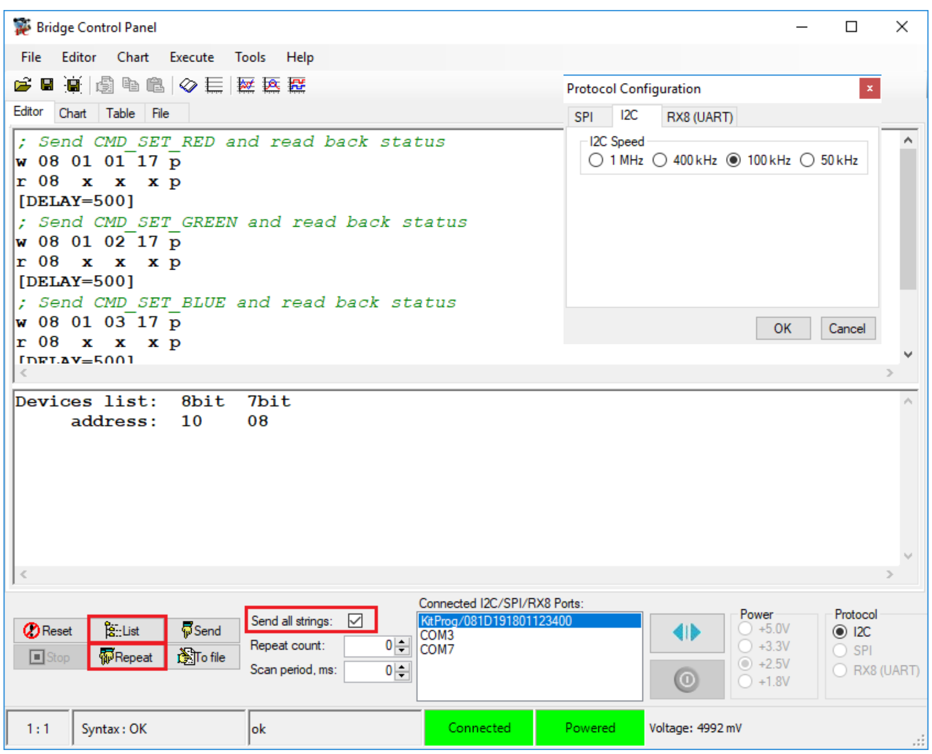
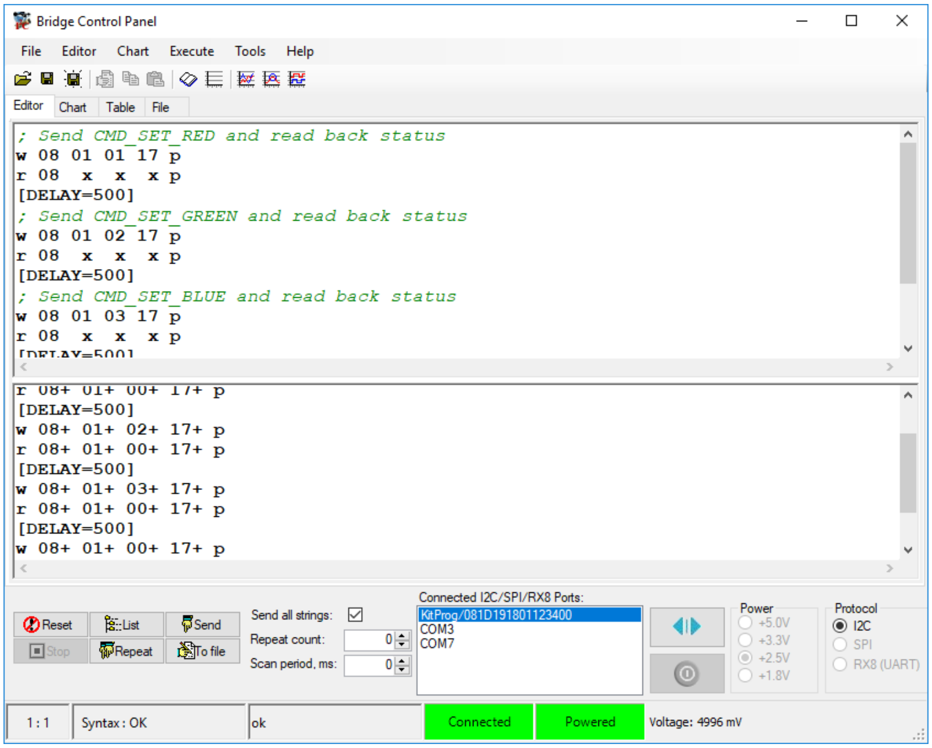


Figure 4. Bridge Control Panel: I²C Master Configuration

Figure 5. Bridge Control Panel: Command Execution Results



Two Kits Operation

1. Follow first three steps from I2C Slave with Bridge Control Panel Operation.
2. Plug your second kit board into your computer's USB port. This kit will be the I²C Master.
3. Set the I2C_SCB_Master project as active: right-click on the project under the **Source** tab in the **Workspace Explorer**, and select **Set as Active Project**.
4. Build the project and program it into the PSoC 4 device. Choose **Debug** > **Program**. For more information on device programming, see the PSoC Creator Help.

Note: To program the correct kit, disconnect other kits from the USB port of your computer during the programming operation.

5. Observe the red color of the RGB LED on the Master kit, indicating an error in the I²C communication - I²C Slave is not connected yet.
6. Connect the kits with the I²C bus: connect I2C:scl, I2C:sda, and GND pins of the Slave kit to the corresponding pins of the kit with the Master firmware and connect the I²C pull-up resistors. You can use resistors in range of 1-10 kΩ. Alternatively, you can enable it with Bridge Control Panel. Refer to Table 2 for pin numbers.

Note: By default, I²C pull-up resistors are disabled. To enable them, you must activate the USB-I²C bridge: open Bridge Control Panel and select the Master or Slave kit KitProg in the **Connected I2C/SPI/RX8 Ports**. Ensure the selected protocol is I²C (Figure 3).

7. Observe the green color of the RGB LED on the I2C Master kit and the RGB LED color changes on the I²C Slave kit: red > green > blue > OFF and then repeats.

Note: The CY8CKIT-149 kit has only green and blue LEDs. With the Master firmware, the green LED (LED11) indicates success in the I²C communication, and the blue LED (LED1) indicates I2C errors. With the Slave firmware, three green LEDs are used: LED11, LED12, and LED13. With the Master control, these LEDs will be enabled in the cycle: LED11 > LED12 > LED13 > all OFF and then repeats.

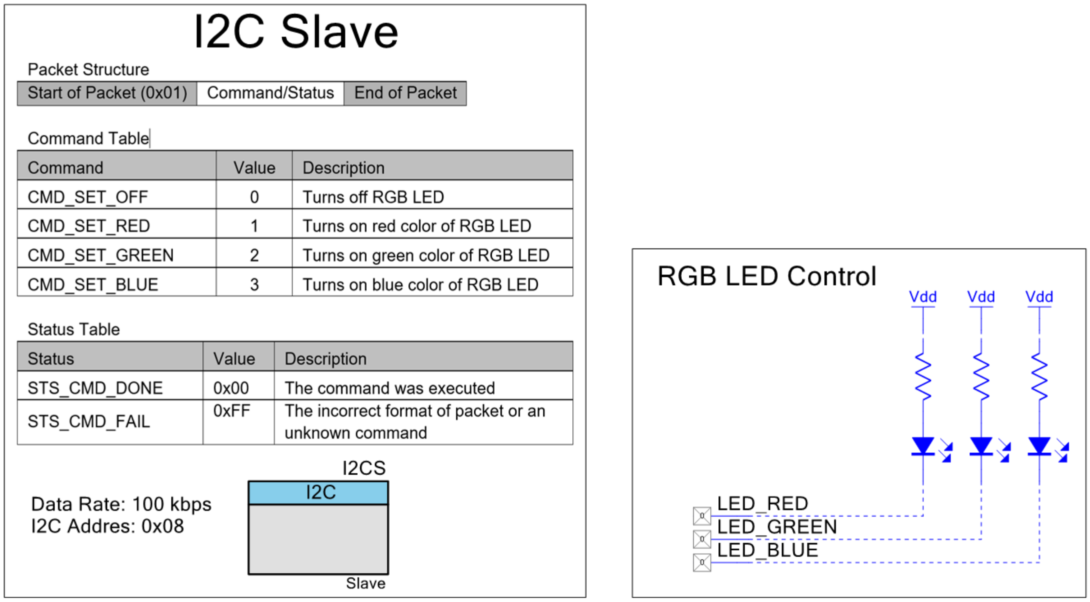
Design and Implementation

This example has two projects: I²C Slave and I²C Master.

I2C_SCB_Slave

The I2C_SCB_Slave design schematic is shown in Figure 6.

Figure 6. I2C_SCB_Slave Top Design Schematic



The design schematic consists of the SCB Component in I²C Slave mode and three Digital Output Pins to control the RGB LED color. The I²C Component has access to the two buffers: Write and Read. The Write buffer is exposed to the master to write a packet with a command, and the Read buffer contains the packet with the command execution status.

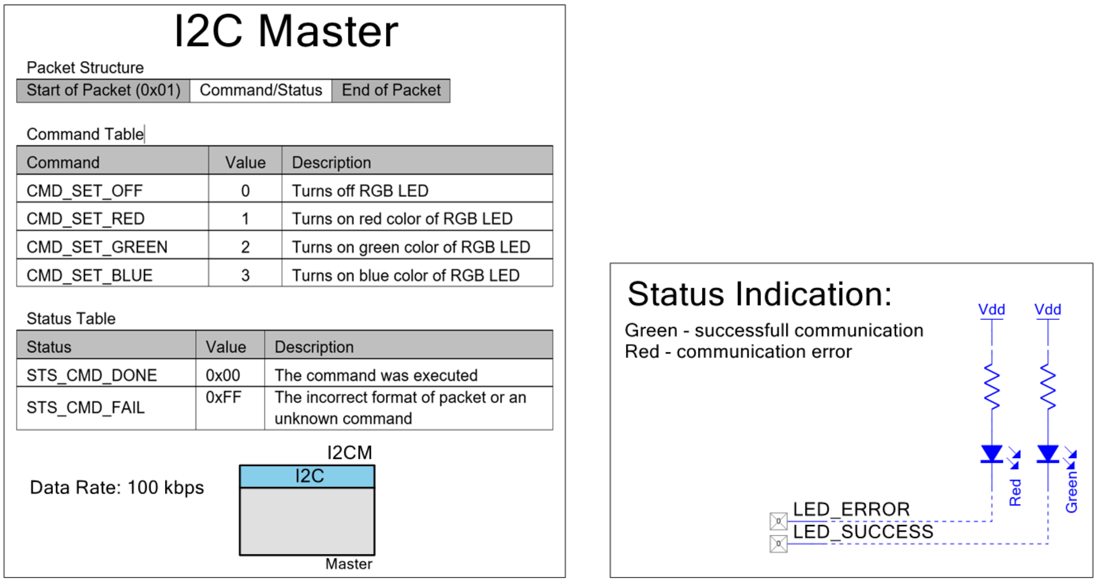
The firmware:

- Initializes the I²C Slave Read and Write buffers, and the I2C Component.
- Waits for communication from the I²C Master: the main loop polls the `I2CS_I2CSlaveStatus()` API continuously to detect the completion of a Write or Read operation.
- After the completion of the Write operation, the Write buffer content is checked. If the packet is valid, the command executes. The result of the command execution is changing the LED color. After the command execution, the Read buffer updates with the status: success or error.
- After the completion of the Read operation, the Slave exposes the Read buffer to the master again. The Master may not read the packet with a status.

I2C_SCB_Master

The I2C_SCB_Master design schematic is shown in Figure 7.

Figure 7. I2C_SCB_Master Top Design Schematic



The design schematic consists of the SCB Component in I²C Master mode and two Digital Output Pins to show the I²C communication status with the RGB LED color. The green color indicates a communication completion without errors, the red color – some error occurred in the communication session.

The firmware:

- Initializes the I²C Master Component.
- Writes to the I²C Slave packet with the LED color command.
- Reads from the I²C Slave status of command execution.
- Checks the I²C transfer errors: if the transfer completed without errors, the RGB LED shows green, otherwise the LED shows red.

Components and Settings

Table 3 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.

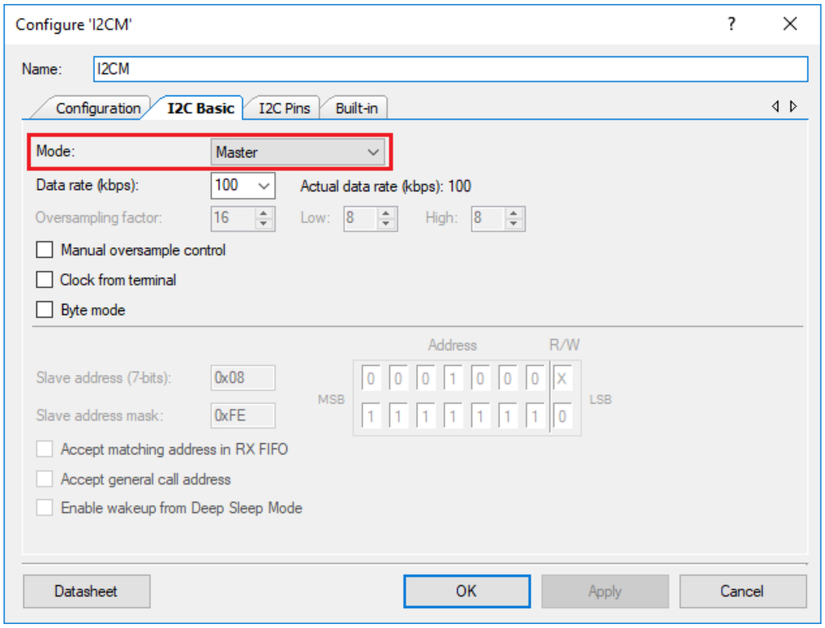
Table 3. PSoC Creator Components

Component	Instance Name	Purpose	Non-default Settings
I²C Slave (SCB Mode)	I2CS	To handle communication between the I²C Master and Slave devices	None
I²C Master (SCB Mode)	I2CM		See Figure 8
Digital Output Pin	LED_RED	Shows results of commands execution	HW connection: OFF
	LED_GREEN		
	LED_BLUE		
	LED_ERROR	Shows the status of the I²C communication	
	LED_SUCCESS		

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

Figure 8 highlights the non-default settings for the SCB Component for I2C_SCB_Master project.

Figure 8. SCB Component Configuration for I2C_SCB_Master Project



Reusing This Example

This example is designed for the kits, listed in Table 1. To port the design to a different PSoC 4 device and/or kit, change the target device using **Device Selector** and update the pin assignments in the Design Wide Resources Pins settings as needed.

Related Documents

Application Notes	
AN79953 – Getting Started with PSoC 4	Introduces the PSoC 4 architecture and development tools.
PSoC Creator Component Datasheets	
Pins	Supports connection of hardware resources to physical pins
Serial Communication Block (SCB)	Supports the hardware SCB block
Device Documentation	
PSoC 4 Datasheets	PSoC 4 Technical Reference Manuals
Development Kit (DVK) Documentation	
PSoC 4 Kits	

Document History

Document Title: CE222306 – PSoC 4 I²C Communication with Serial Communication Block (SCB)

Document Number: 002-22306

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5985995	MYKZTMP1	01/12/2018	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

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