CE224599 – PSoC 4 I²C Slave with Serial Communication Block (SCB)

Objective

This code example demonstrates the basic usage of the PSoC® 4 Serial Communication Block (SCB) Component in I²C Slave mode. Two projects demonstrate different methods for receiving data from an I²C master. One project demonstrates polling for data from the master and the other project demonstrates an interrupt based receive.

Overview

For this example, the SCB in I²C Slave mode accepts command packets to control the color of an RGB LED. The I²C Slave updates its read buffer with a status packet in response to the accepted command. The two projects in this example demonstrate this using a polling method and an interrupt driven callback method.

Requirements

Tool: PSoC Creator™ 4.2

Programming Language: C (Arm® GCC 5.4.1)

Associated Parts: PSoC 4 parts

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

Note: PSoC 4000 series parts do not have enough Timer Counter PWM Components to work with this example.

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.

This example uses the kit's default configuration.

Table 1. Supported Kits and Devices

Development Kit	Series	Device
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	\I2C:scl\	\I2C:sda\	LED_RED	LED_GREEN	LED_BLUE
CY8CKIT-041-40XX CY8CKIT-041-41XX	P3[0]	P3[1]	P3[4]	P2[6]	P3[6]
CY8CKIT-042	P3[0]	P3[1]	P1[6]	P0[2]	P0[3]
CY8CKIT-042-BLE CY8CKIT-042-BLE-A	P3[5]	P3[4]	P2[6]	P3[6]	P3[7]
CY8CKIT-044	P4[0]	P4[1]	P0[6]	P2[6]	P6[5]
CY8CKIT-046	P4[0]	P4[1]	P5[2]	P5[3]	P5[4]
CY8CKIT-048	P4[0]	P4[1]	P1[4]	P2[6]	P1[6]
CY8CKIT-149	P3[0]	P3[1]	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

1. Plug your kit board 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 from **Start** > **All programs** > **Cypress** > **Bridge Control Panel <version>** > **Bridge Control Panel <version>**.
5. Select the KitProg device listed in the **Connected I2C/SPI/RX8 Ports**. Ensure the selected protocol is I^2C (Figure 1).
6. Go to **Tools** > **Protocol Configuration**, and in the **I2C** tab select **I2C Speed** 100kHz (Figure 2).
7. Press the **List** button to ensure that the I^2C Slave device with the address 0x08 (7-bits) is available for communication.

Note: Other I^2C devices can be connected to the I^2C bus. These devices' addresses are shown after the list operation completion. See the development kit documentation for more information about other I^2C devices available on the kit.

8. Begin sending commands.

   In the **Editor** tab of BCP, type the command to be written or data to be read from the I^2C slave device.

   The packet format for writing to a Slave device from the BCP is shown in Table 3.

Table 3. Packet Format for Writing

Start for Write	Slave Address	Start of Packet (SOP)	Red LED TCPWM Compare Value	Green LED TCPWM Compare Value	Blue LED TCPWM Compare Value	End of Packet (EOP)	Stop
w	(0x08)	(0x01)	(0x00) to (0xFF)	(0x00) to (0xFF)	(0x00) to (0xFF)	(0x17)	p

Some of the example commands the I^2C Master can send to the Slave device are shown below.

- **'w 08 01 00 00 00 17 p'** Turns the LED OFF.
- **'w 08 01 FF 00 00 17 p'** Changes the color to red.

- **‘w 08 01 00 FF 00 17 p’** Changes the color to green.
- **‘w 08 01 00 00 FF 17 p’** Changes the color to blue.
- **‘w 08 01 FF FF 00 17 p’** Changes the color to yellow.
- **‘w 08 01 7F 00 7F 17 p’** Changes the color to purple.
- **‘w 08 01 FF FF FF 17 p’** Changes the color to white.

The packet format for reading from the Slave device is shown in Table 4. The ‘x’ symbol denotes which byte to read from the Slave’s read buffer. In this example, the read buffer consists of three bytes: SOP, Status, and EOP.

Table 4. Packet Format for Reading from Slave

Start for Read	Slave Address	SOP	Status	EOP	Stop
r	(0x08)	x	x	x	p

An example read command is **‘r 08 x x x p’** which reads the status of the last write command. In the console output window of the BCP, below the **Editor** tab, the result of this command will be displayed. If the read operation returns **‘r 08 01 00 17 p’**, then the write operation was successful. If the read operation returns **‘r 08 01 FF 17 p’**, then the write operation failed.

Figure 1. Bridge Control Panel

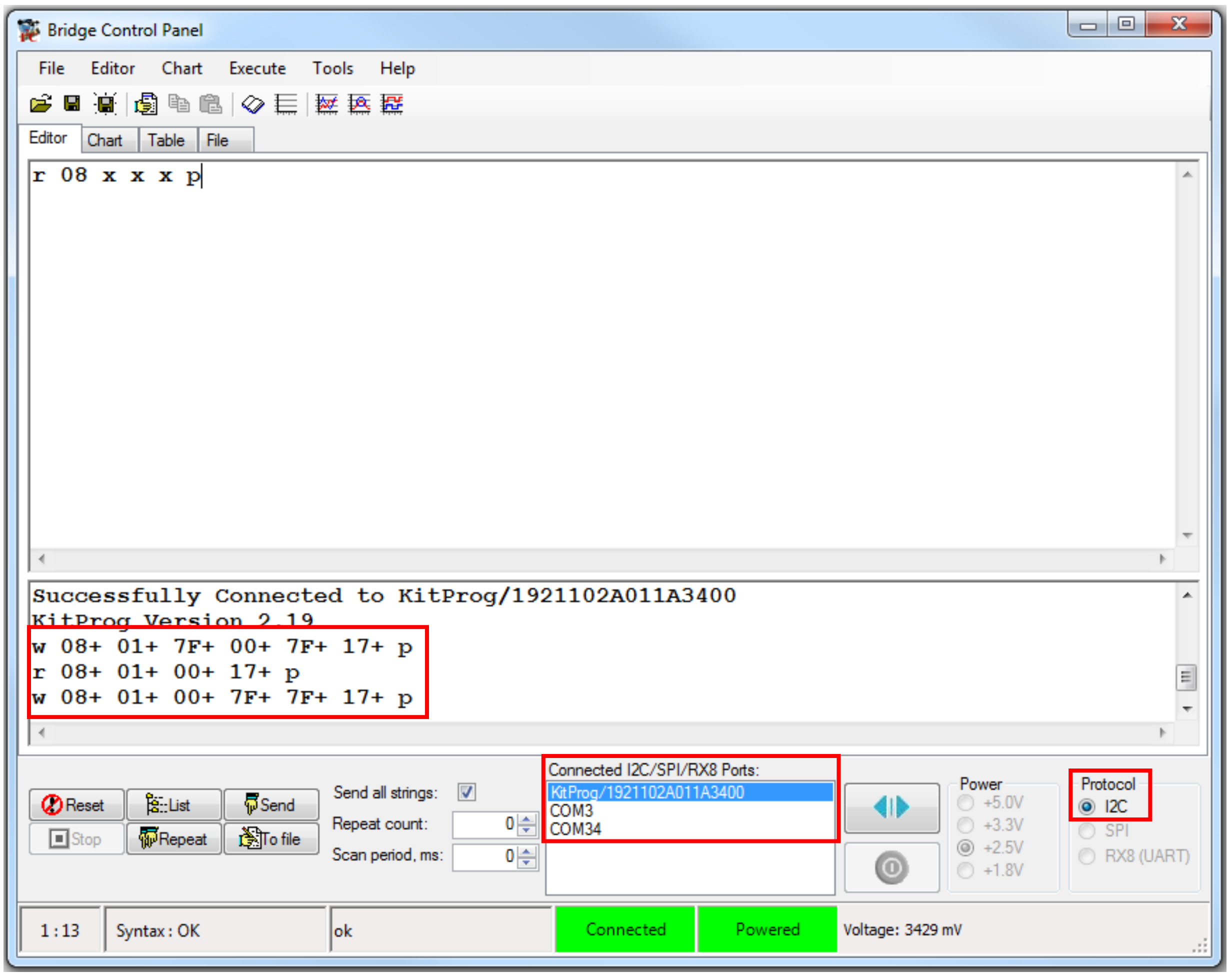
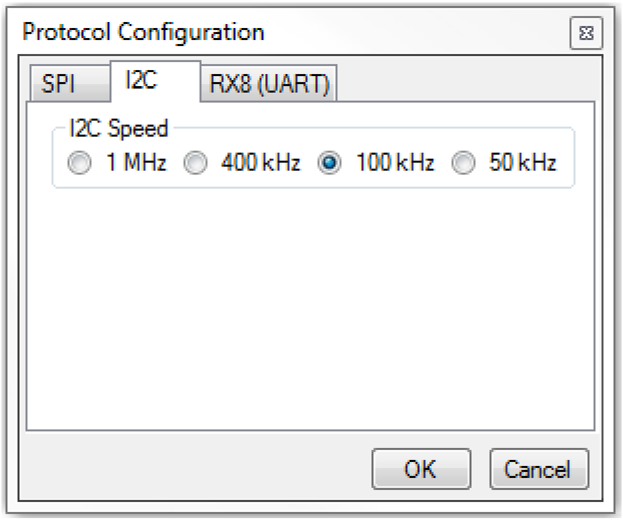


Figure 2. Bridge Control Panel I^{2}C Configuration



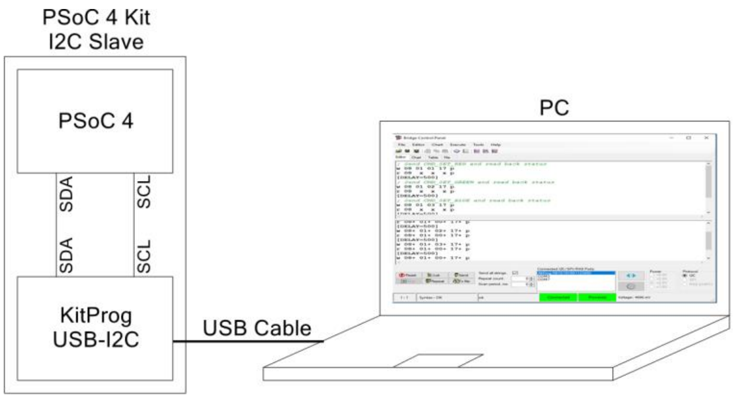
Design

Figure 3 shows the high-level implementation of this example. A PSoC 4 kit is set as an I^{2}C Slave device by using the SCB Component. It communicates with the I^{2}C Master PC through a USB connection to the o-board KitProg device. The Slave is configured with a 5-byte write buffer, which the Master can access and write commands to. In addition, the Slave is configured with a 3-byte read buffer, which the Master can access to read the status of the previous transmission.

The write buffer packet contains 5-bytes of instructions. The first byte is the SOP, followed by the three bytes for red, green, and blue color control, and the final byte is the EOP. The three-color control bytes are used by the Slave to update the compare values for three Timer Counter Pulse Width Modulator (TCPWM) Components. The read buffer contains 3 bytes. The first byte is SOP, the second byte is the Status of the write command from the Master, and the final byte is the EOP. The status byte is either 0x00, indicating a successful write, or 0xFF, indicating a failed write.

The color and brightness of the RGB LED are controlled by the TCPWM components. The period of the TCPWM is set at 255 which results in a 47-kHz refresh rate on the LED given that the input clock is 12 MHz. The duty cycle of the TCPWM is set by the value of the color control bytes written to the Slave. Changing the value of the color control bytes changes the duty cycle of the TCPWM components which, in turn, changes the color and brightness of the RGB LED.

Figure 3. PSoC 4 Kit I2C connection to PC



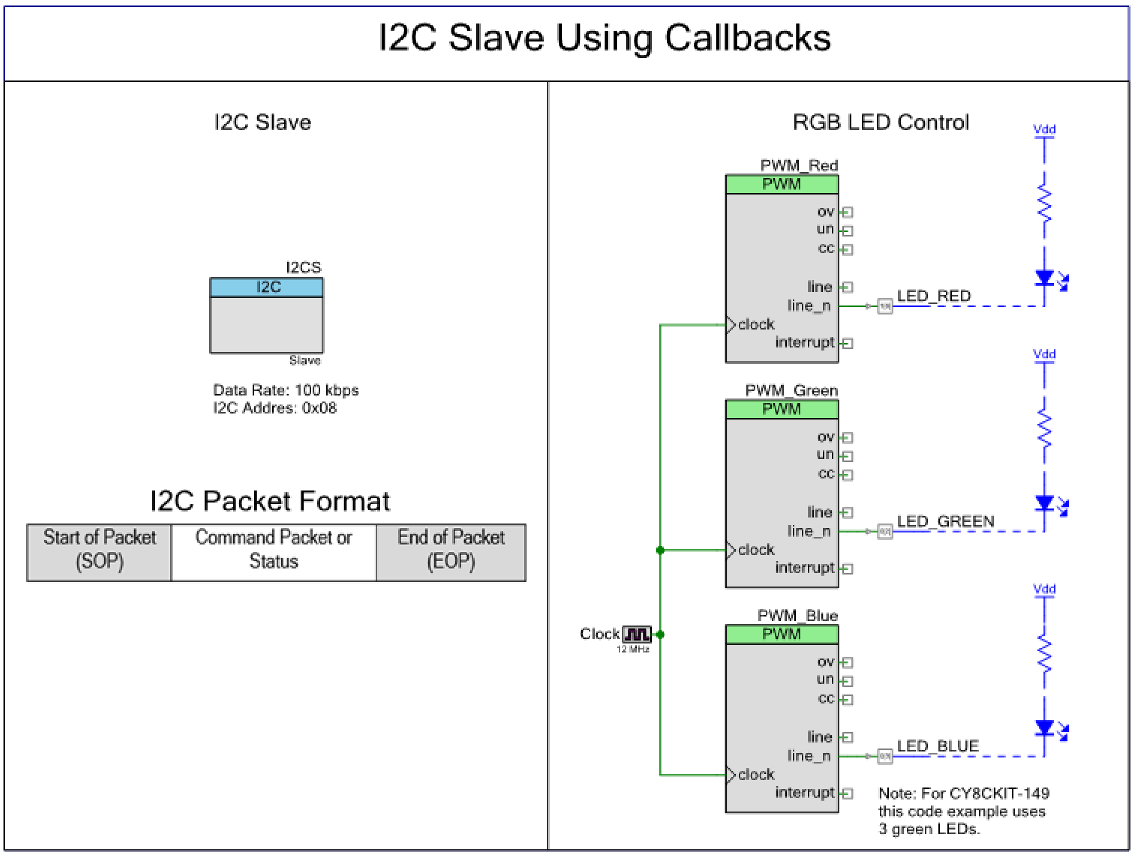
I²C Slave Using Callback Method

Figure 4 shows the top schematic in PSoC Creator for the I²C Slave using the callback method. The callback method uses an interrupt to set a flag in firmware. Firmware polls the flag and writes and reads command from the Master if the flag is set. The firmware updates the TCPWM components and resets the write buffer for the next commands.

The *cyapicallbacks.h* header file has been changed to include a macro definition, which activates the callback function in a generated source file for the SCB. When the project is built, the source files are created and the *I2CS_I2C_INT.c* file activates the callback function.

Note: The schematic and Component configuration is the same in the polling method.

Figure 4. I²C Slave Schematic for Callback Method



I2C Slave Using Polling Method

In the polling method, the PSoC 4 Slave device continually checks to see if a command has been sent from the Master PC. If a command has completed, the Slave checks the packet for correct size, SOP, command, and EOP. After verifying the packet, the Slave utilizes the packet data to update the TCPWM compare values. It also updates the status value in the read buffer.

Components and Settings

Table 5 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 5. PSoC Creator Components

Component	Instance Name	Purpose	Non-default Settings
I^2C Slave (SCB Mode)	I2CS	To enable I^2C communication between the Master and Slave device.	Default settings only
Digital Output Pin	LED_RED	Shows results of commands execution	Open drain, drives low
	LED_GREEN		
	LED_BLUE		
PWM (TCPWM Mode)	PWM_Red	Control the brightness of the RGB LED	**Period**: 255
	PWM_Green		
	PWM_Blue		
Clock	Clock	Drives PWM Components	**Frequency**: 12MHz

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

Reusing This Example

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

Related Documents

Code Examples	
CE222306 – PSoC 4 I2C Master with Serial Communication Block (SCB)	Demonstrates how to use the Serial Communication Block in I^2C Master mode
CE195362 – PSoC 4 EZI2C Slave with Serial Communication Block (SCB)	Demonstrates the basic usage of the EZI2C Slave implemented with the Serial Communication Block (SCB)
Application Notes	
AN79953 – Getting Started with PSoC 4	Introduces the PSoC 4 architecture and development tools.
PSoC Creator Component Datasheets	
Serial Communication Block (SCB)	Supports serial communication usage
Pins	Supports connection of hardware resources to physical pins
Timer Counter (TCPWM)	Supports fixed-function Timer/Counter implementation
Clock	Supports local clock generation
Device Documentation	
PSoC 4 Datasheets	PSoC 4 Technical Reference Manuals
Development Kit (DVK) Documentation	
PSoC 4 Kits	

Document History

Document Title: CE224599 – PSoC 4 I^{2}C Slave with Serial Communication Block (SCB)

Document Number: 002-24599

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6276467	BFMC	07/13/2018	Updated to new template Changed to use PWM components to control the LED color and intensity Updated to match the PSoC 6 I^{2}C Slave 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

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