

## Objective

This example demonstrates the use of the PSoC® Creator Serial Communication Block (SCB) Component for PSoC 4 in I<sup>2</sup>C master mode. Two projects demonstrate the use of high-level and low-level functions to communicate with an I<sup>2</sup>C slave.

## Overview

This code example consists of two projects:

- I2C\_Master\_High\_Level – implements the I<sup>2</sup>C Master device to send commands to the I<sup>2</sup>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 API that is used transfers entire buffers to the slave device, eliminating the need for byte-wise communication.
- I2C\_Master\_Low\_Level – Implements the same functionality as the High\_Level project but uses lower-level firmware which performs writes and reads on a byte-by-byte basis. The firmware also keeps track of write and read completion.

## Requirements

**Tool:** PSoC Creator™ 4.2

**Programming Language:** C (Arm® GCC 5.4.1)

**Associated Parts:** All PSoC 4 Parts

**Related Hardware:** CY8CKIT-042 PSoC 4 Pioneer Kit

## Hardware Setup

This example uses the CY8CKIT-042 kit to demonstrate the I<sup>2</sup>C master device. However, other PSoC 4 kits with I<sup>2</sup>C support can be used. To get the most out of the example, it is recommended that a second PSoC 4 kit programmed with the EZI2C slave code example [CE195362](#) is connected to the CY8CKIT-042.

If you use two kits, connect a ground pin (**GND**) and the I<sup>2</sup>C pins (**SCL** and **SDA**) together between the two kits. [Table 1](#) below shows the I<sup>2</sup>C pins on each kit. Note that I<sup>2</sup>C bus lines are open drain and must be pulled up to V<sub>DD</sub> using resistors as [Figure 1](#) shows. Resistor values can be calculated from Chapter 7 of the NXP I<sup>2</sup>C bus specification, UM10204, available [here](#).

Figure 1. I2C Pull-up Resistor Schematic

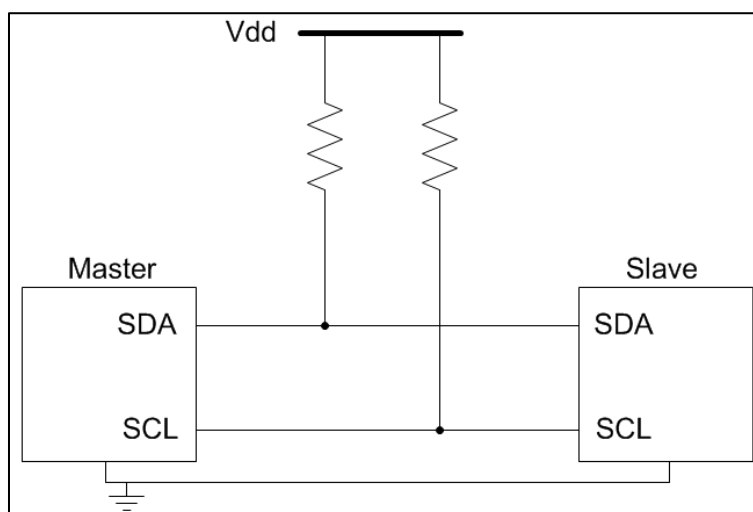


Table 1. Pin Assignments

Development Kit	I <sup>2</sup> C:scI\	I <sup>2</sup> C:sda\
CY8CKIT-041-40XX	P3[0]	P3[1]
CY8CKIT-041-41XX		
CY8CKIT-042	P4[0]	P4[1]
CY8CKIT-042-BLE	P3[5]	P3[4]
CY8CKIT-042-BLE-A		
CY8CKIT-044	P4[0]	P4[1]
CY8CKIT-046	P4[0]	P4[1]
CY8CKIT-048	P4[0]	P4[1]
CY8CKIT-149	P3[0]	P3[1]

## Operation

1. Plug the CY8CKIT-042 kit into your computer's USB port.
2. Build and program either of the PSoC4 I2C Master projects into the CY8CKIT-042. Choose **Debug > Program**. For more information on device programming, see PSoC Creator Help.
3. If a second PSoC 4 Kit is available, program [CE195362](#) into the second kit.
  - a. Connect the two kits following the instructions in [Hardware Setup](#).
  - b. Press the reset button on each kit and observe the LED on the slave kit changing colors.
4. If a second PSoC 4 kit is not available, use an oscilloscope to observe the data and clock on the I<sup>2</sup>C bus lines. Connect oscilloscope probes to each of the I<sup>2</sup>C pins and ground each probe to a pin labeled GND.

## Design and Implementation

In each project, CY8CKIT-042 acts as an I<sup>2</sup>C master device sending commands to a PSoC 4 device programmed as an I<sup>2</sup>C slave using code example CE195362.

The application uses two I<sup>2</sup>C buffers to communicate between the master and slave device. The write buffer carries the commands to be written and the status buffer contains the status of the previous transaction. In each buffer, the first and last bytes contain specific values which are used to check for correct packet format. The remaining positions are reserved for commands and status values.

The command packet is organized as shown below.

Start of Packet (SOP)	Red Compare Value	Green Compare Value	Blue Compare Value	End of Packet (EOP)
(0x01)	(0x00 – 0xFF)	(0x00 – 0xFF)	(0x00 – 0xFF)	(0x17)

After a command packet is written to the slave device, the slave updates the contents of the status buffer indicating either success or failure of the previous write operation. The status packet is organized as shown below.

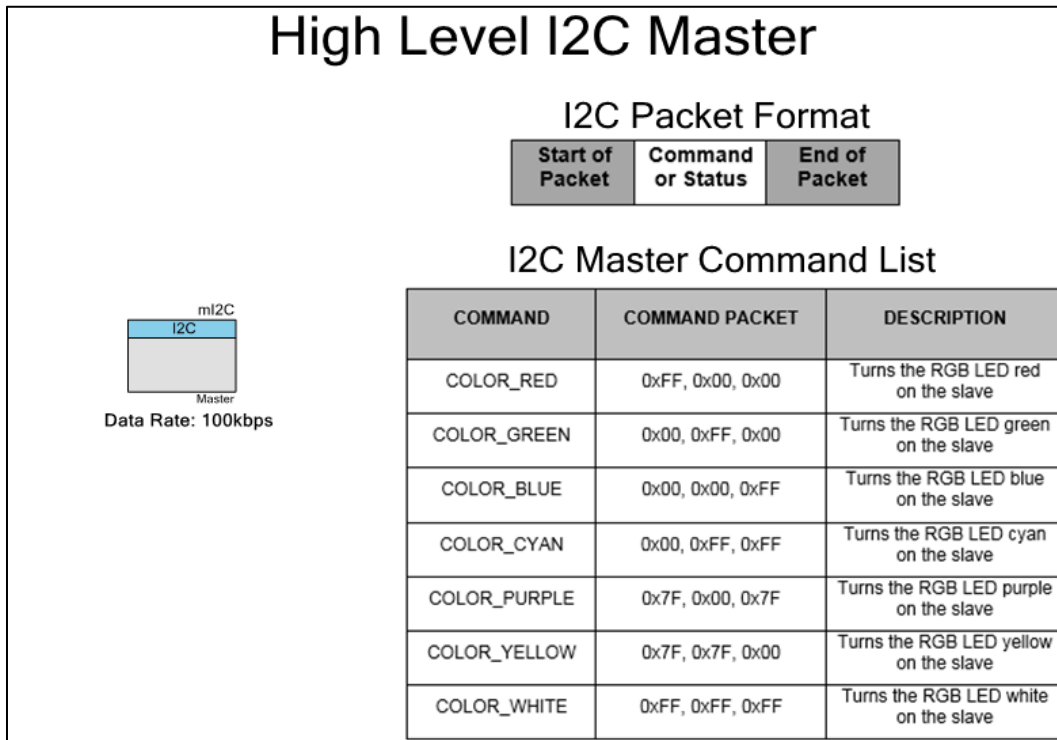
SOP	Status	EOP
(0x01)	(0x00 or 0xFF)	(0x17)

The firmware for both projects begins by initializing buffers for reads and writes. Firmware then writes the red compare value to the write buffer and transfers the buffer to the slave. After a successful write, the master reads status data from the slave into the read buffer. If this read is successful, the firmware waits for a half-second before cycling to the next LED color. The process repeats indefinitely.

## I2C\_Master\_High\_Level Design

The I2C\_Master\_High\_Level project uses the SCB I2C Component in master mode with a data rate of 100 kbps. It uses high-level functions from the SCB API to write the command buffer and read the status from the slave. The high-level functions handle start and stop conditions and abstract away from the byte-wise transactions that lower-level functions use. The schematic in Figure 2 shows the available commands.

Figure 2. Schematic for the I2C\_Master\_High\_Level



**Note:** The Schematic for the I2C\_Low\_Level project is the same.

## I2C\_Master\_Low\_Level Design

The SCB Component is configured as an I<sup>2</sup>C master with a data rate of 100 kbps. The firmware uses low-level functions that send start and stop conditions before and after each write or read operation. Firmware transmits the write buffer by iterating through each byte in the buffer. Firmware reads the read buffer in a similar manner. The low-level API may be preferred if the user needs a higher level of control in the timing of transactions.

## Components and Settings

Table 2 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 2. PSoC Creator Components

Component	Instance Name	Purpose	Non-default Settings
I2C (SCB mode)	mI2C	Enables I <sup>2</sup> C communication as master	<b>Mode:</b> Master

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

## Reusing This Example

This example is designed for use with the CY8CKIT-042 pioneer kit. To port the design to a different PSoC 4 device and/or kit, change the target device using the Device Selector (**Project > Device Selector**) and update the pin assignments in the Design Wide Resources Pins settings as needed.

For this example, ensure that the I<sup>2</sup>C pins **SCL** and **SDA** as well as a **GND** pin are connected properly between the master and slave devices. Pull the I<sup>2</sup>C bus lines up to V<sub>DD</sub> using resistors as shown in [Figure 1](#). Resistor values can be calculated from Chapter 7 of the NXP I<sup>2</sup>C bus specification, UM10204, available [here](#).

In some cases, a resource used by a code example (for example, the CapSense hardware) 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 what a particular device supports.

## Related Documents

Application Notes	
<a href="#">AN86526</a> – PSoC 4 and PSoC Analog Coprocessor I2C Bootloader	Describes an I <sup>2</sup> C-based bootloader and demonstrates how to create an I <sup>2</sup> C-based bootloadable project.
<a href="#">AN86439</a> – PSoC 4 – Using GPIO Pins	Describes how to use PSoC 4 GPIO pins with various use case examples
Code Examples	
<a href="#">CE195362</a> – PSoC 4 EZI2C Slave with Serial Communication Block (SCB)	Demonstrates the basic usage of the EZI2C Slave Component with the Serial Communication Block
<a href="#">CE224599</a> – I2C Slave using a Serial Communication Block (SCB) with PSoC 4	Demonstrates the basic operation of the I2C slave (SCB mode) Component.
PSoC Creator Component Datasheets	
<a href="#">Serial Communication Block (SCB)</a>	Supports serial communication usage
Device Documentation	
<a href="#">PSoC 4 Datasheets</a>	<a href="#">PSoC 4 Technical Reference Manuals</a>
Development Kit Documentation	
<a href="#">PSoC 4 Kits</a>	

## Document History

Document Title: CE222306 – PSoC 4 I2C Master

Document Number: 002-22306

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5985995	MYKZTMP1	01/12/2018	New code example
*A	6469732	BFMC	02/07/2019	Converted to Master Only Updated I <sup>2</sup> C command format Updated pin table Moved SCB to top of component list

## 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 Forums](#) | [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.