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.

CE220541 - PSoC 6 MCU SCB EZI2C

Objective

This code example demonstrates the implementation of an EZI2C Slave using the SCB Component on a PSoC® 6 MCU. It also demonstrates how to control the color and intensity of an RGB LED using TCPWM Components.

Overview

This code example implements an I²C slave using an SCB Component (configured as EZI2C), which receives the data required to control an RGB LED from an I²C master. In this example, a host PC running the Cypress' Bridge Control Panel (BCP) software is used as an I²C master. The RGB LED control is implemented using three TCPWM Components (configured as PWM). The color and intensity of the RGB LED is controlled by changing the duty cycle of the PWM signals.

Requirements

Tool: PSoC Creator™ 4.2

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

Associated Parts: All PSoC 6 MCU parts

Related Hardware: CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit

Hardware Setup

The code example works with the default settings on the CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit. If the settings are different from the default values, see the "Selection Switches" table in the kit guide to reset to the default settings.

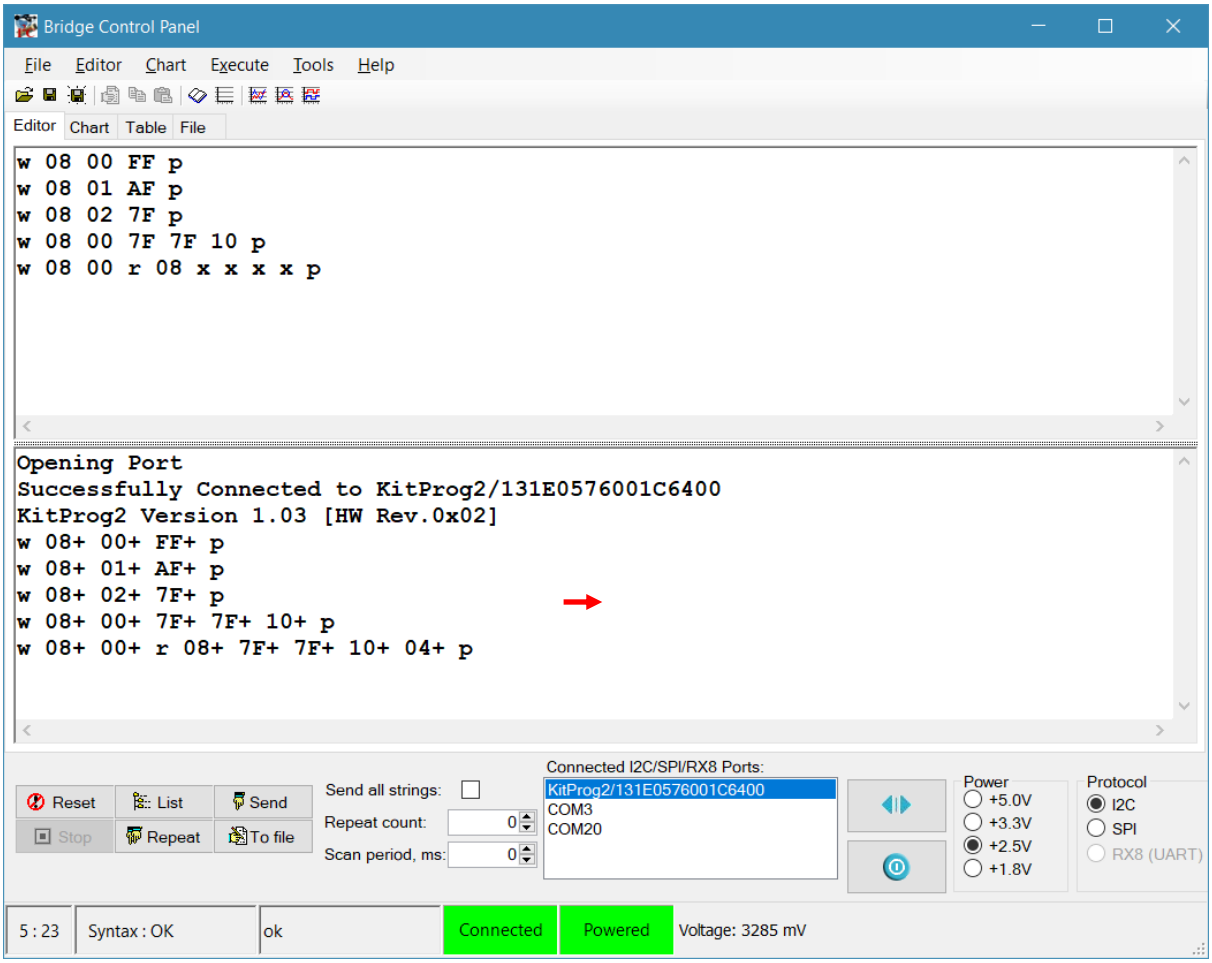
Software Setup

This section describes how to set up the BCP software for viewing sensor data sent over I²C.

The BCP software is installed automatically as part of the PSoC Creator installation. Follow these steps to configure the BCP:

1. Open the BCP from **Start** > **All Programs** > **Cypress** > **Bridge Control Panel <version>** > **Bridge Control Panel <version>.**
2. Select **KitProg2/<serial_number>** under **Connected I2C/SPI/RX8 Ports** (see Figure 1). Note that the PSoC 6 BLE Pioneer Kit must be connected to the USB port of your computer.

Figure 1. Bridge Control Panel



3. Select **Tools** > **Protocol Configuration**, navigate to the **I2C** tab, and set the **I2C speed** to '100 kHz'. Click **OK**.

BCP is now ready for reading and displaying the sensor data. For the testing procedure, see Operation.

Operation

1. Open the *CE220541_SCB_EZI2C* code example in PSoC Creator.
2. Build the project (**Build** > **Build CE220541_SCB_EZI2C**).
3. Connect the CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit to your computer using the USB cable provided.
4. Program the PSoC 6 MCU (**Debug** > **Program**). See the kit guide for details on programming the kit.
5. Configure the BCP software as described in the section Software Setup.
6. In the **Editor** tab of BCP, type the command to send RGB LED control data and then click **Send**. Observe that the RGB LED turns ON with the specified color and intensity.

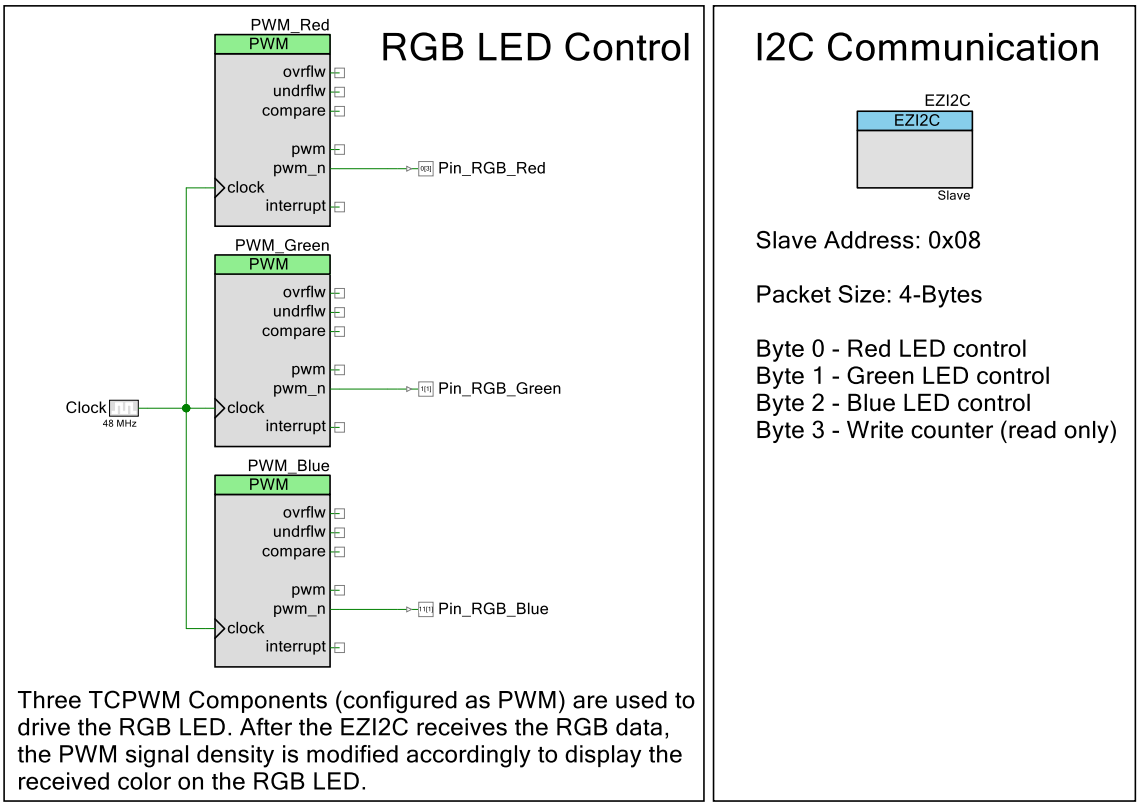
   For example, sending command '`w 08 00 FF FF FF p`' will turn the RGB LED ON with white color and full intensity and sending command '`w 08 00 00 00 00 p`' will turn the RGB LED OFF.

   You can also control individual LEDs by writing the value to the specific memory location. For example, sending command '`w 08 02 7F p`' will turn the RGB LED ON with blue color and medium intensity.
7. Type the command '`w 08 03 r 08 x p`' to read the number of Write operations performed after device program/reset.

Design and Implementation

Figure 2 shows the PSoC Creator schematics of this code example. This code example uses EZI2C, three TCPWMs, three Pins, and Clock Components.

Figure 2. TopDesign Schematic



In this example, the EZI2C is configured with a 4-bytes buffer (memory), which can be accessed by the I^2C master. The first three bytes are writable and hold the red, green, and blue LED intensity and the fourth byte, which is read-only, holds the number of Write operations performed after the device reset.

EZI2C allows an I^2C master to either access the individual byte from the slave memory (by specifying the memory address in the Write command) or all memory bytes at once. For the testing procedure, see Operation.

To control the color and intensity of an RGB LED, three PWMs with period value of 255 (~195 kHz) are used. The duty cycle of the PWMs are controlled in the firmware and specified by the I^2C master. Changing the duty cycle of the PWM signal will result in change in the LED intensity. By changing the intensity of individual LEDs, various colors can be produced on the RGB LED as a color mixing solution. Intensity range in this example is 0x01 to 0xFF, 0x00 turns the LED OFF.

Components and Settings

Table 1 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 1. List of PSoC Creator Components

Component	Instance Name	Purpose	Non-default Settings
EZI2C (SCB)	EZI2C	I2C Slave operation	-
PWM (TCPWM)	PWM_Red	Generate square wave and bring out the signal to GPIO	**Period 0**: 255u **Compare 0**: 0u
	PWM_Green		**Period 0**: 255u **Compare 0**: 0u
	PWM_Blue		**Period 0**: 255u **Compare 0**: 0u
Digital Output Pin	Pin_RGB_Red	Drive the PWM signal to LED	-
	Pin_RGB_Green		-
	Pin_RGB_Blue		-
Clock	Clock	Drive the PWM at 48MHz	**Frequency**: 48 MHz

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

Table 2 shows the pin assignment for the project done through the **Pins** tab in the **Design Wide Resources** window.

Table 2. Pin Names and Location

Pin Name	Location
EZI2C:scl	P6[0]
EZI2C:sda	P6[1]
Pin_RGB_Red	P0[3]
Pin_RGB_Green	P1[1]
Pin_RGB_Blue	P11[1]

Reusing This Example

This code example is designed to run on CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit. To port the design to a different PSoC 6 MCU device and/or kit, change the target device in Device Selector, and update the pin assignments in the Design Wide Resources Pins settings as needed. For single-core PSoC 6 MCU devices, port the code from *main_cm4.c* to *main.c* file.

Related Documents

Application Notes	
AN210781 –Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes PSoC 63 with Bluetooth Low Energy (BLE) Connectivity and how to build your first PSoC Creator project
PSoC Creator Component Datasheets	
EZI2C	Supports simplified I^2C slave implementation
PWM	Supports fixed-function PWM implementation
Pins	Supports connection of hardware resources to physical pins
Clock	Supports local clock generation
Device Documentation	
PSoC 6 MCU: PSoC 63 with BLE Datasheet	PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
Development Kit (DVK) Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	

Document History

Document Title: CE220541 - PSoC 6 MCU SCB EZI2C

Document Number: 002-20541

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5845468	SRDS	08/17/2017	Initial public release
*A	5991548	SRDS	12/21/2017	Updated template and minor text changes. Updated project to PSoC Creator 4.2 Beta.

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

© Cypress Semiconductor Corporation, 2017. 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. Other names and brands may be claimed as property of their respective owners.