SCB_I2cCommSlave Example Project
1.0

Features

- Communication between I^{2}C master and slave
- Simple packet protocol with command and status byte

General Description

This example project demonstrates the basic operation of the I^{2}C slave (SCB mode) component. The I^{2}C slave accepts a packet with a command from the I^{2}C master to control the RGB LED color. The I^{2}C slave updates its buffer with a status packet in response to the accepted command.

Development Kit Configuration

This example project is designed to be executed on CY8CKIT-042 from Cypress Semiconductor. A full description of the kit, along with more example programs and ordering information, can be found at http://www.cypress.com/go/cy8ckit-042.

The project requires configuration settings changes in order to run on CY8CKIT-040 from Cypress Semiconductor. A full description of the kit, along with more example programs and ordering information, can be found at http://www.cypress.com/go/cy8ckit-040.

In order to switch from CY8CKIT-042 to the CY8CKIT-040 following steps should be performed:

1. Change the project's device from CY8C4245AXI-483 to CY8C4014LQI-422 with a Device Selector called from the project's context menu.

2. Change the clock configuration. In the Workspace Explorer window, double-click the project's design-wide resource file and click on the **Edit Clocks…** icons on the **Clocks** tab. Set IMO frequency to **32 MHz**.

3. Change the assignment of the pin components to physical pins. In the Workspace Explorer window, double-click the project's design-wide resource file and assign the pins for I^{2}C slave and RGB LED accordingly to Table 1.

Table 1. Pin assignment of the SCB_I2cCommSlave project

Pin Name	Development Kit	
	CY8CKIT-042	CY8CKIT-040
\I2CS:scl\	P3[0]	P1[2]
\I2CS:sda\	P3[1]	P1[3]
LED_BLUE	P0[3]	P0[2]
LED_GREEN	P0[2]	P1[1]
LED_RED	P1[6]	P3[2]

Project Configuration

The example project consists of the I^{2}C slave (SCB mode) and pin components. The design schematic is shown in Figure 1. The blue annotation components are used to represent the RGB LED installed on the kit. The three pin components are used to control the LED color. The kit provides connection between the I^{2}C slave (PSoC 4A) and I^{2}C master (PSoC 5LP) as well as a pull-up resistor required for the I^{2}C bus operation. The Bridge Control Panel software is provided to control the I^{2}C master.
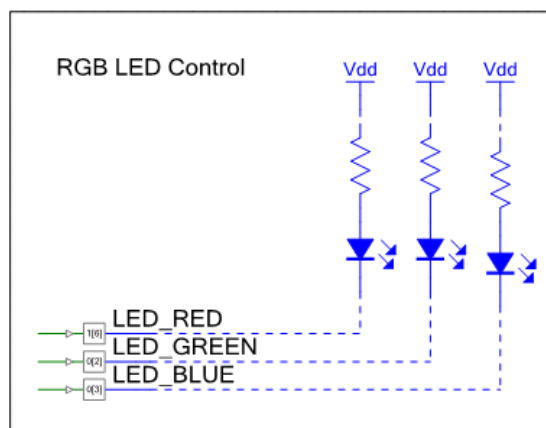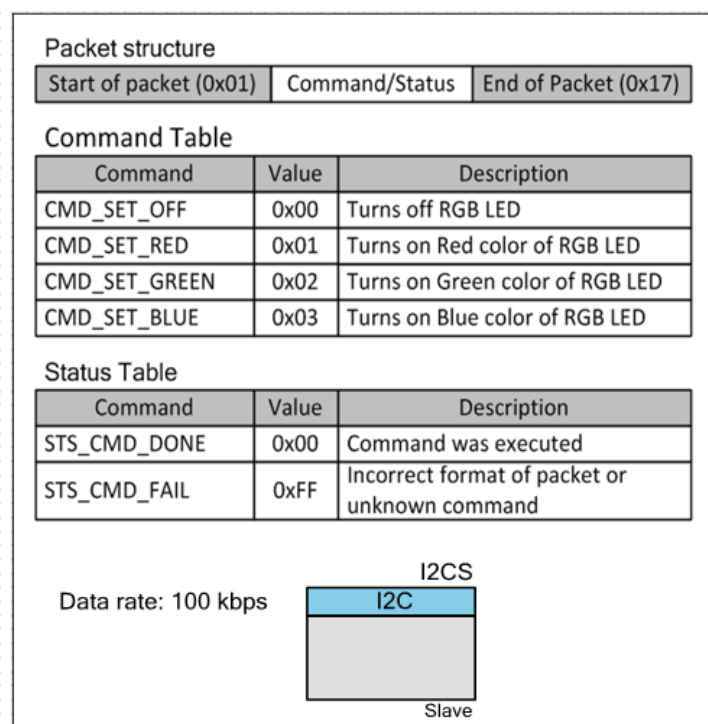
Figure 1. Example project design schematic

The I^{2}C slave is configured to operate with the data rate of 100 kpbs and responds to address 0x08 (7-bits). The component configuration window is shown below.
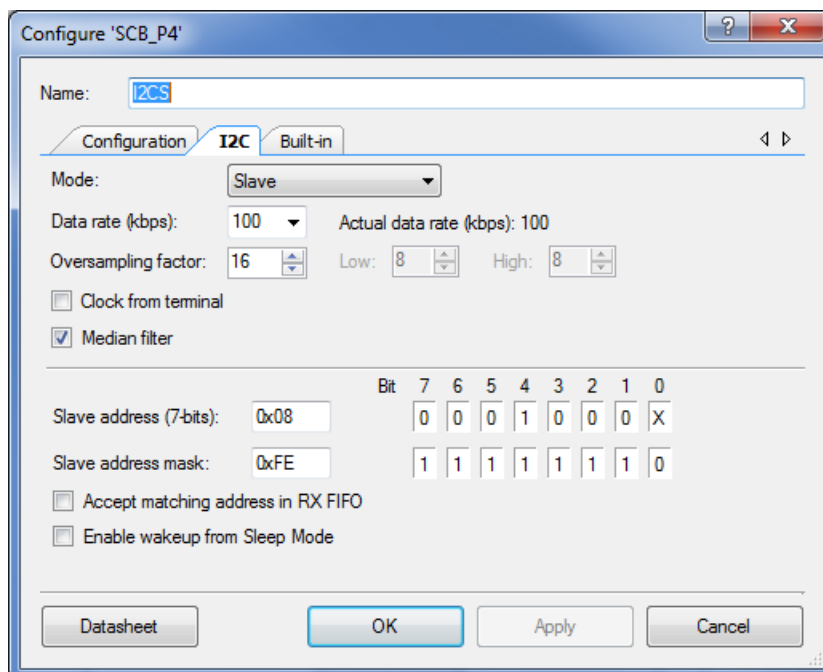
Figure 2. I2C slave (SCB mode) component configuration

Project Description

In the main firmware routine, the I^{2}C slave read and write buffers are configured when a component is started. The write buffer is exposed to the master to write a packet with a command and reads back the packet with a status. Interrupts are enabled to the CPU core as required by the I2C slave component for operation.

The I^{2}C slave waits for communication from the I^{2}C master. The main loop polls I2CS_I2CSlaveStatus() API continuously for a write or read completion event.

When a write completion event is reported, the write buffer content is checked against the valid packet. The packet structure and table with commands are shown below. The basic checks of the packet structure are done: the length of the packet, the start and end of the packet byte. If packet considered as valid the command is retrieved and passed to be executed. The result of a command execution is a change of the LED color. Initially the LED is turned off. The status is updated with a successful command execution or failure in the case when a command is unknown or the packet is considered as invalid. The table with return statuses is shown below. The packet with a status is exposed to the master in the slave read buffer.

When a read completion event is reported, the slave just exposes the read buffer to the master again. The master may not read the packet with a status at all and then it just sends a next packet with a command.

Packet structure

Start of packet (0x01)	Command/Status	End of Packet (0x17)

Table 2. Command constants

Command	Value	Description
CMD_SET_OFF	0	Turns off RGB LED
CMD_SET_RED	1	Turns on Red color of RGB LED
CMD_SET_GREEN	2	Turns on Green color of RGB LED
CMD_SET_BLUE	3	Turns on Blue color of RGB LED

Table 3. Status constants

Status	Value	Description
STS_CMD_DONE	0x00	Command was executed
STS_CMD_FAIL	0xFF	Incorrect format of packet or unknown command

The packets with a command and status are converted into the following I^{2}C master transfers. The packet with a command has a write direction set in the address byte and the packet with a status has a read direction set appropriately.

Packet with command

S	ADDR = 0x08	W	A	SOP = 0x01	A	Command	A	EOP = 0x17	A	P

Packet with status

S	ADDR = 0x08	R	A	SOP = 0x01	A	Status	A	EOP = 0x17	Ā	P

☐ - Master drives the bus   ▨ - Slave drives the bus

Expected Results

Build and program an example project with the I^{2}C slave.

Run the Bridge Control Panel software which is shipped with the PSoC Creator. It is used to control the I^{2}C master implemented on the PSoC 5LP which is available on the kit. Follow the steps below to setup communication between the master and slave:

1. Select the KitProg device into the list of the Connected Ports.

2. Make sure that the selected Protocol is I^{2}C.

3. Go to Tools->Protocol Configuration and select I2C Speed 100kHz.

4. Press the List button ⟦ 🏷::List ⟧ to make sure that the I^{2}C slave device with address 0x08 (7-bits) is available for communication[1].

---

[1] The device with 7-bits address 0x50 appears on I^{2}C when list command is executed for CY8CKIT-040. This I^{2}C device is Serial F-RAM.
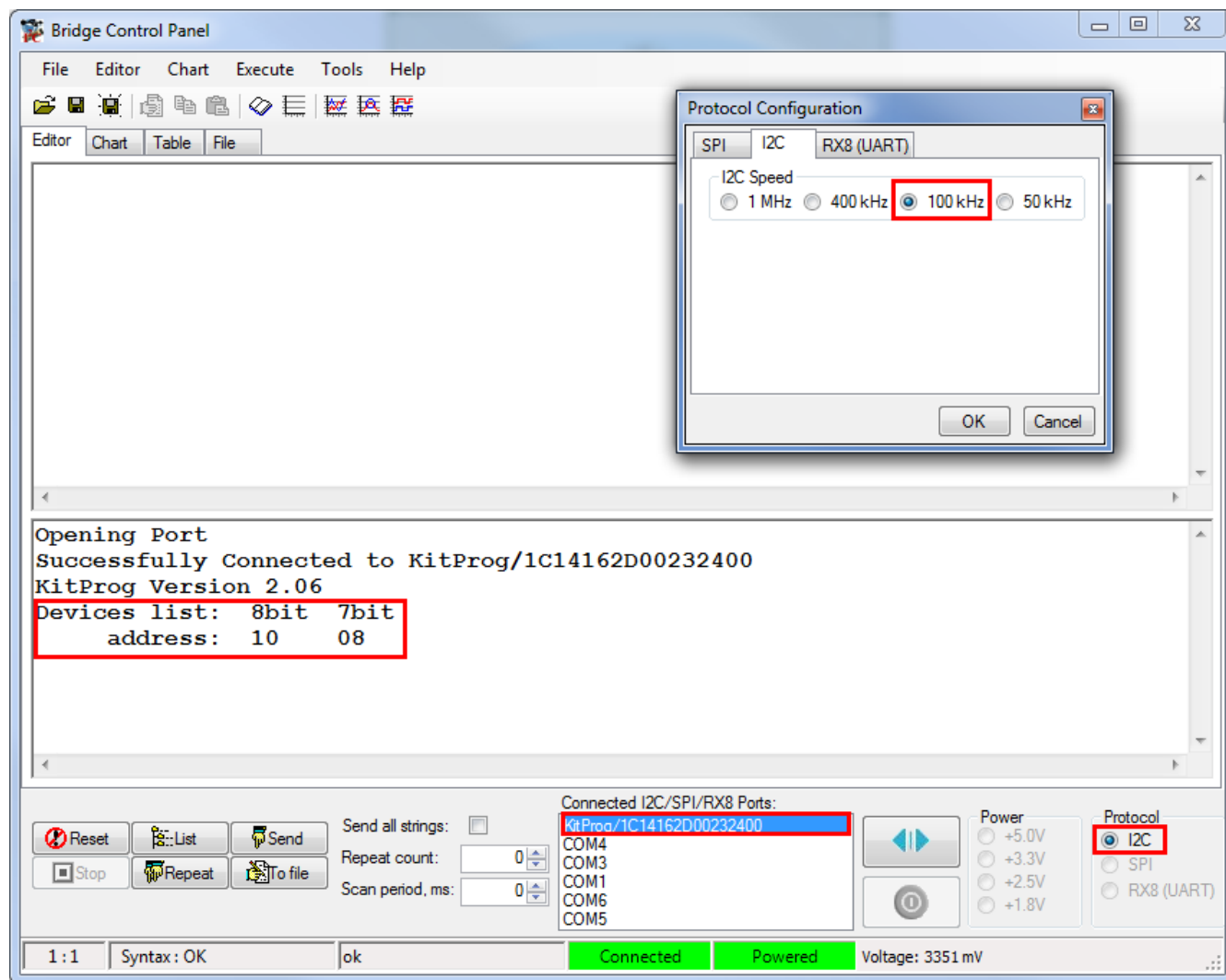
Figure 3. Bridge Control Panel I2C master setup

1. To load master commands for communication with the I^{2}C slave use Open icon 📂 . Navigate to BCP_Master_I2cCmd.iic file which is attached to the workspace and open it. The commands should appear in the Edit window.

2. There are two options of a master transfer execution.

   - A single command execution: set the cursor to the line with the command into the Edit window and press Enter. The RGB LED should change its color accordingly to the executed command.

   - Repeat the command execution: select a number of commands and press the Repeat button. The RGB LED should change its color accordingly to the executed commands.

   The delays are added between commands to notice a LED color change.
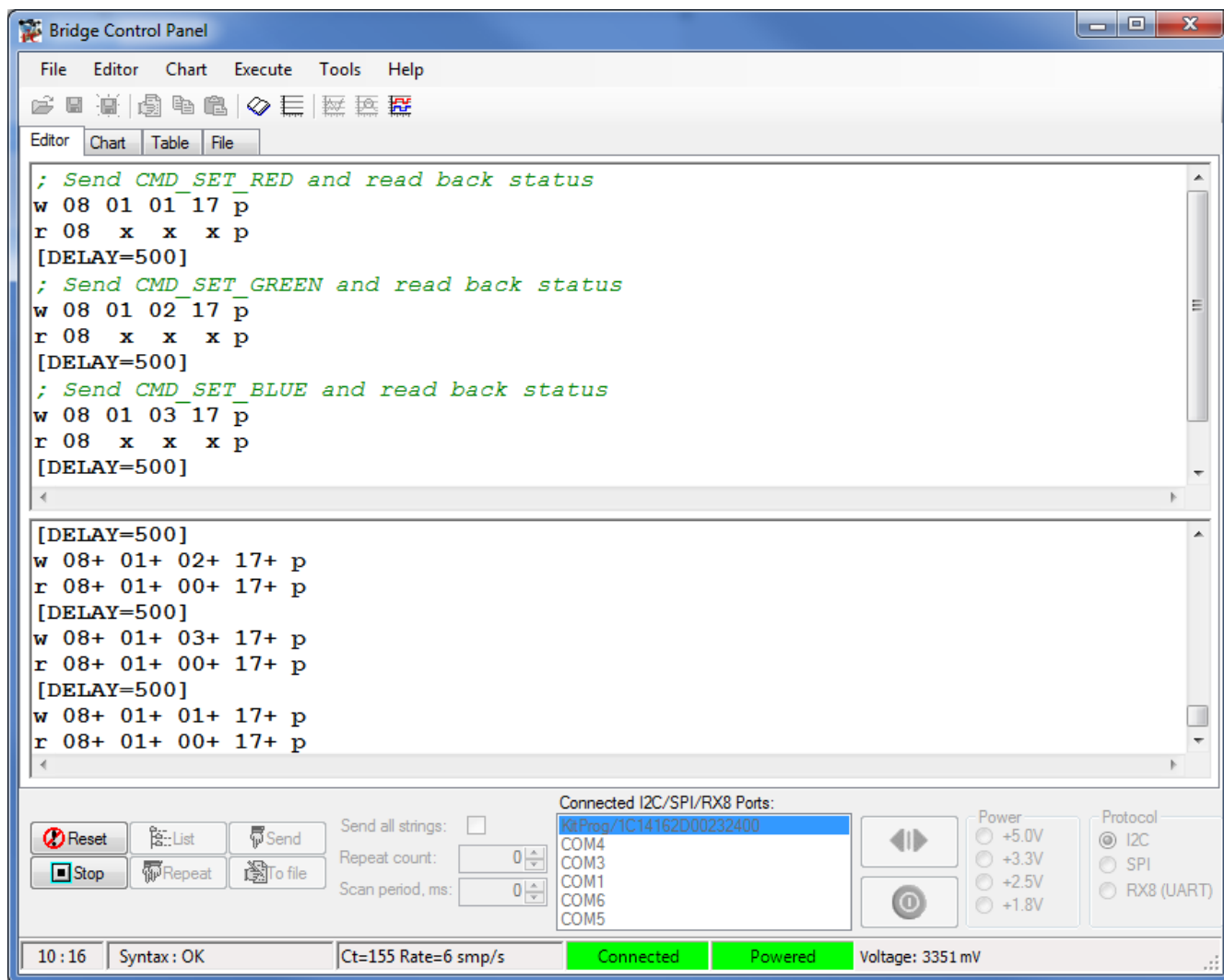
Figure 4. Repeat command execution result