SCB_I2cCommSlave Example Project
1.20

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.

The I^{2}C slave in this project is designed to communicate with an I^{2}C master in either of the following configurations:

- The I^{2}C master already present on the development kit. The master can be controlled using the provided Bridge Control Panel software.
- The SCB_I2cCommMaster Example project, programmed onto another device to serve as the master.

Development Kit Configuration

This example project is designed to run on the CY8CKIT-042 kit from Cypress Semiconductor. A 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 to run on other kits from Cypress Semiconductor. Table 1 is the list of the supported kits. To switch from CY8CKIT-042 to any other kit, change the project's device with the help of Device Selector called from the project's context menu.

Table 1. Development Kits vs Parts

Development Kit	Device
CY8CKIT-040	CY8C4014LQI-422
CY8CKIT-041	CY8C4045AZI-S413 / CY8C4146AZI-S433
CY8CKIT-042	CY8C4245AXI-483
CY8CKIT-042-BLE	CY8C4247LQI-BL483
CY8CKIT-044	CY8C4247AZI-M485

CY8CKIT-046	CY8C4248BZI-L489
CY8CKIT-048	CY8C4A45LQI-483

The pin assignments for the supported kits are in Table 2.

Table 2. Pin Assignment

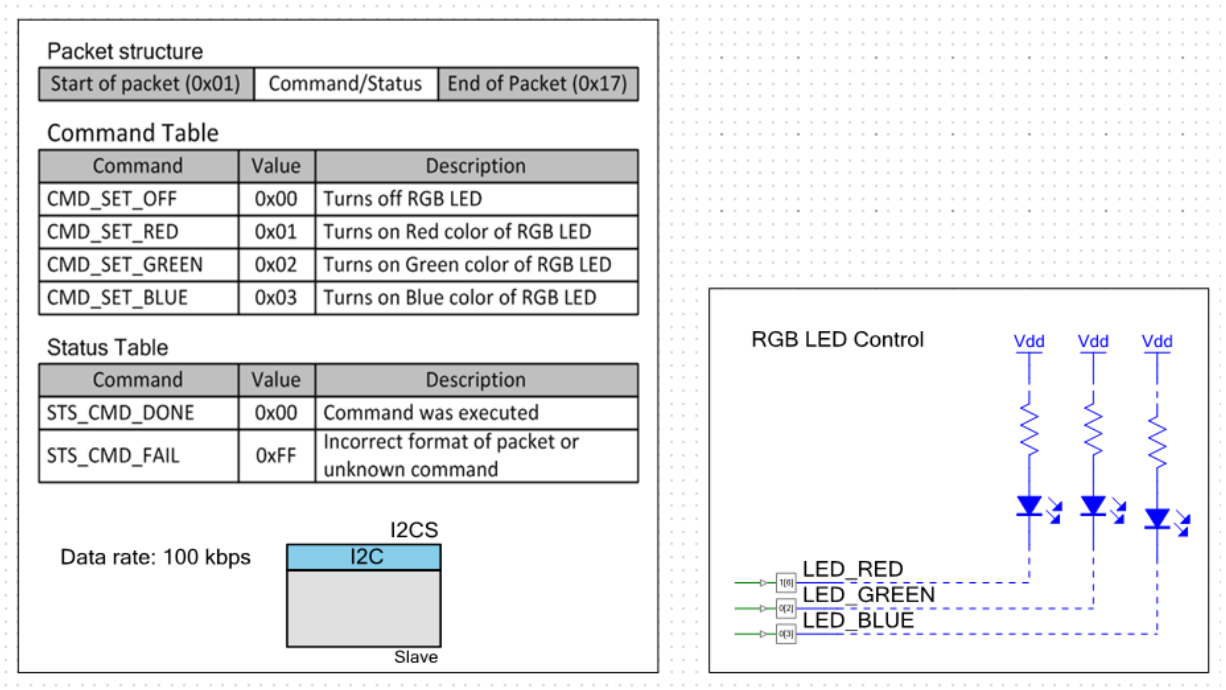
Development Kit	Pin Name				
	\I2CS:scl\\	**\I2CS:sda\**	**LED_RED**	**LED_GREEN**	**LED_BLUE**
CY8CKIT-040	P1[2]	P1[3]	P3[2]	P1[1]	P0[2]
CY8CKIT-041	P3[0]	P3[1]	P3[4]	P2[6]	P3[6]
CY8CKIT-042	P3[0]	P3[1]	P1[6]	P0[2]	P0[3]
CY8CKIT-042 BLE	P3[5]	P3[4]	P2[6]	P3[6]	P3[7]
CY8CKIT-044	P4[0]	P4[1]	P0[6]	P2[6]	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]

Note The project control files handle the pins placement automatically according to a selected PSoC.

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 4) 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



Packet structure

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

Command Table

Command	Value	Description
CMD_SET_OFF	0x00	Turns off RGB LED
CMD_SET_RED	0x01	Turns on Red color of RGB LED
CMD_SET_GREEN	0x02	Turns on Green color of RGB LED
CMD_SET_BLUE	0x03	Turns on Blue color of RGB LED

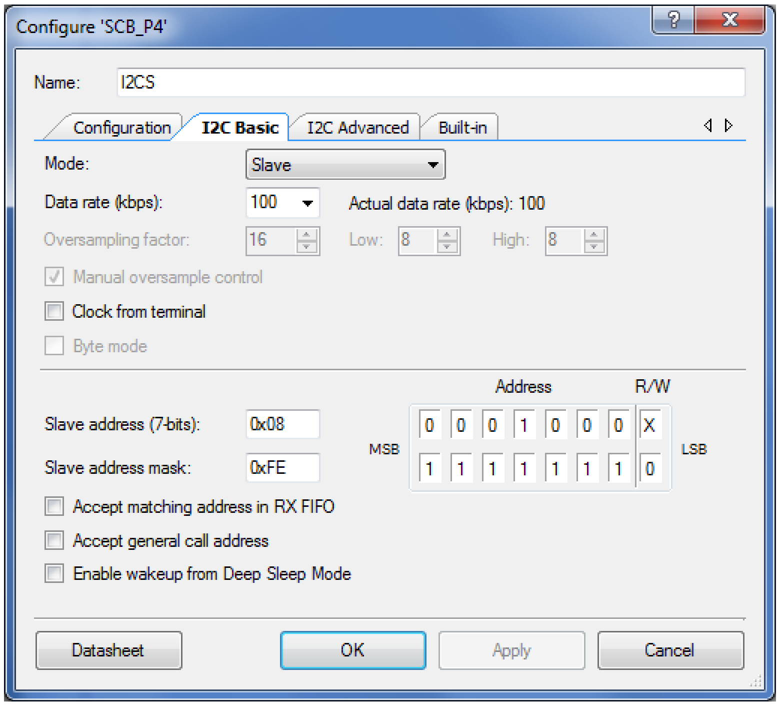
Status Table

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

Data rate: 100 kbps

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

Figure 2. I2C Slave (SCB mode) Component Configuration



Configure 'SCB_P4'

Name: I2CS

Configuration | **I2C Basic** | I2C Advanced | Built-in

Mode: Slave

Data rate (kbps): 100 Actual data rate (kbps): 100

Oversampling factor: 16 Low: 8 High: 8

[✓] Manual oversample control

[ ] Clock from terminal

[ ] Byte mode

	Address							R/W	
MSB	0	0	0	1	0	0	0	X	LSB
	1	1	1	1	1	1	1	0	

Slave address (7-bits): 0x08

Slave address mask: 0xFE

[ ] Accept matching address in RX FIFO

[ ] Accept general call address

[ ] Enable wakeup from Deep Sleep Mode

Datasheet OK Apply Cancel

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)

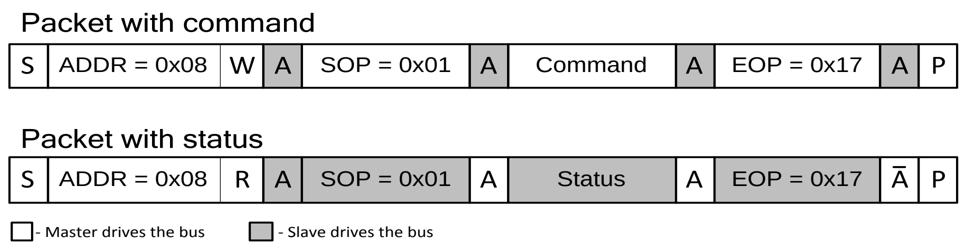
Table 3. Command constants

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 4. 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.



Expected Results

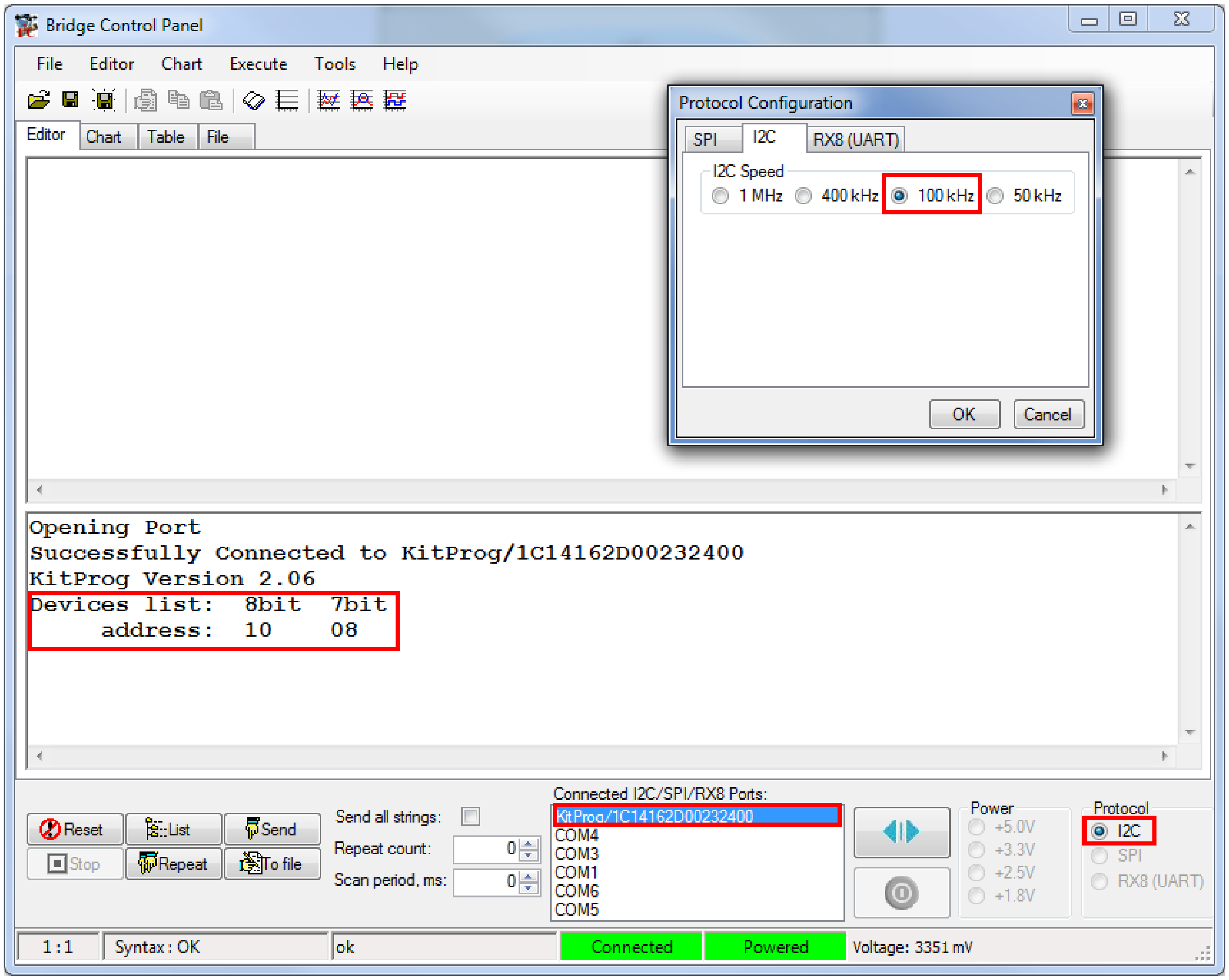
Build example project and program into the device.

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] Other I^{2}C devices can be connected to the I^{2}C bus. The addresses of these devices are shown after list operation completion. Refer to the development kit documentation for more information about other I^{2}C devices available on the kit.

Figure 3. Bridge Control Panel I2C Master Setup



I2C Speed: ○ 1 MHz ○ 400 kHz [✓ 100 kHz] ○ 50 kHz

```
Opening Port
Successfully Connected to KitProg/1C14162D00232400
KitProg Version 2.06
Devices list:  8bit   7bit
     address:  10     08
```

Send all strings: ☐
Repeat count: 0
Scan period, ms: 0

Connected I2C/SPI/RX8 Ports:
KitProg/1C14162D00232400
COM4
COM3
COM1
COM6
COM5

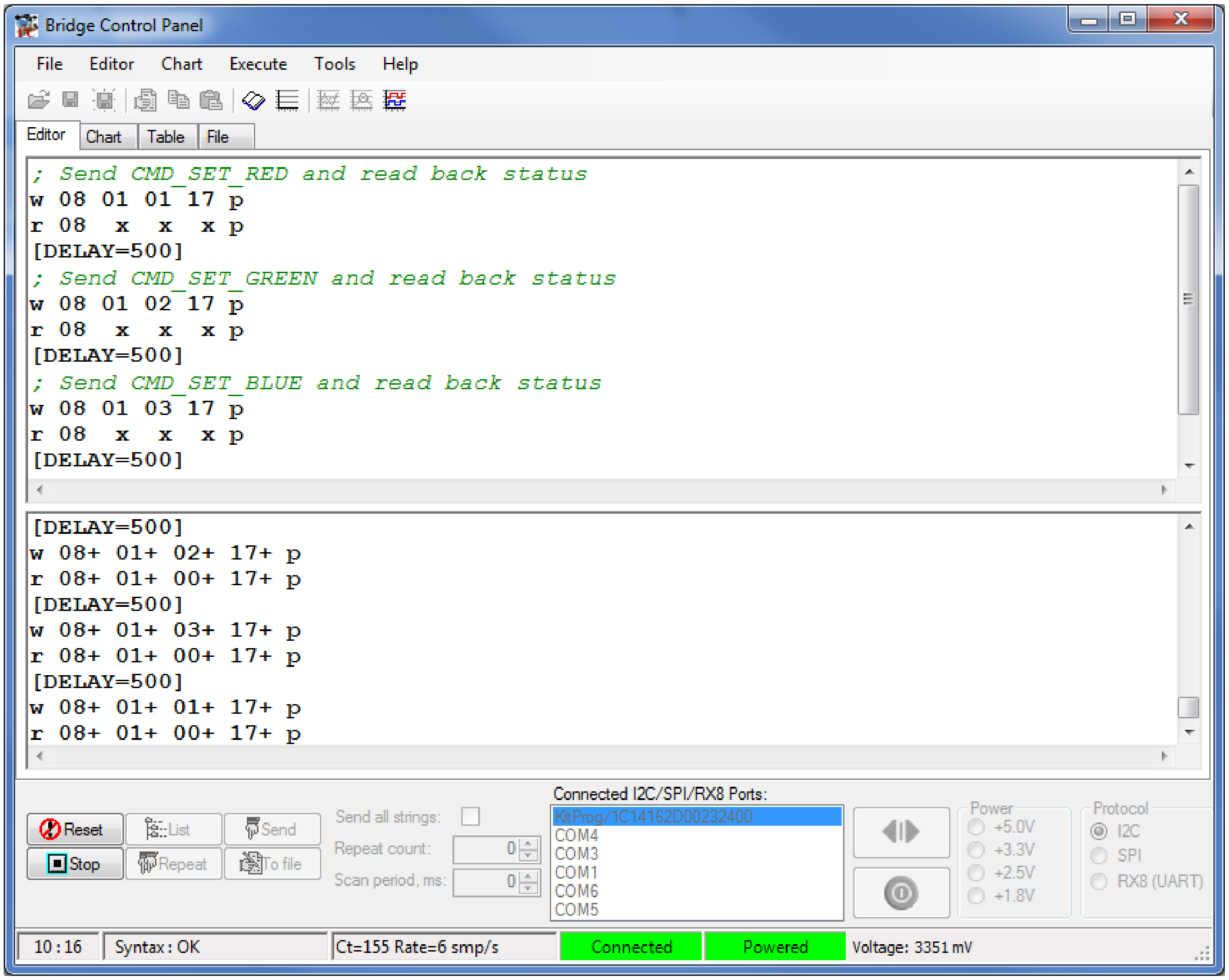
Protocol: [✓ I2C] ○ SPI ○ RX8 (UART)

1:1 | Syntax : OK | ok | Connected | Powered | Voltage: 3351 mV

5. 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.
6. 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



```
; Send CMD_SET_RED and read back status
w 08 01 01 17 p
r 08  x  x  x p
[DELAY=500]
; Send CMD_SET_GREEN and read back status
w 08 01 02 17 p
r 08  x  x  x p
[DELAY=500]
; Send CMD_SET_BLUE and read back status
w 08 01 03 17 p
r 08  x  x  x p
[DELAY=500]
```

```
[DELAY=500]
w 08+ 01+ 02+ 17+ p
r 08+ 01+ 00+ 17+ p
[DELAY=500]
w 08+ 01+ 03+ 17+ p
r 08+ 01+ 00+ 17+ p
[DELAY=500]
w 08+ 01+ 01+ 17+ p
r 08+ 01+ 00+ 17+ p
```

Send all strings: ☐
Repeat count: 0
Scan period, ms: 0

Connected I2C/SPI/RX8 Ports:
KitProg/1C14162D00232400
COM4
COM3
COM1
COM6
COM5

Power: +5.0V, +3.3V, +2.5V, +1.8V
Protocol: [✓ I2C] SPI RX8 (UART)

10 : 16 | Syntax : OK | Ct=155 Rate=6 smp/s | Connected | Powered | Voltage: 3351 mV