

# SCB\_UartComm Example Project

## 1.10

## Features

- Simple communication with the PSoC 4 from the serial terminal software

## General Description

This example project demonstrates the basic operation of the SCB component in the UART mode. The polling method is used to wait for a received character. As soon as the character is received it is transmitted back. The serial terminal can be used on the PC to send characters and get them back. The example will echo every received character.

## 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-042	CY8C4245AXI-483
CY8CKIT-042-BLE	CY8C4247LQI-BL483
CY8CKIT-044	CY8C4247AZI-M485

Make sure that the IMO frequency is **24 MHz** after device was changed.

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

Table 2. Pin Assignment

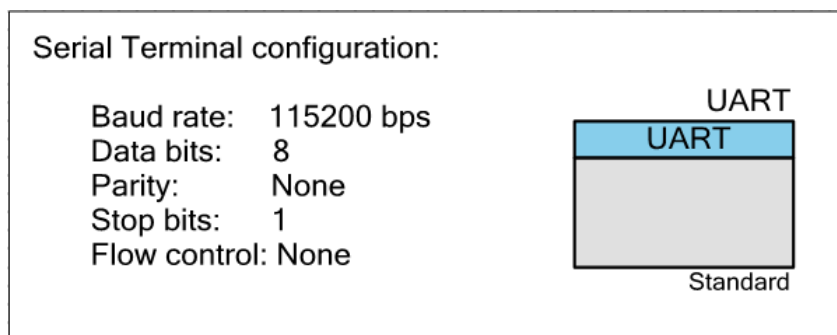
Pin Name	Development Kit		
	CY8CKIT-042	CY8CKIT-042 BLE	CY8CKIT-044
\UART:rx\	P0[4]*	P1[4]	P7[0]
\UART:tx\	P0[5]*	P1[5]	P7[1]

\* Connect P0[4] (\UART:rx\ ) and P0[5] (\UART:tx\ ) to the Rx and Tx terminals (Pin P12[7] and P12[6]) on header J8.

## Project Configuration

The example project consists of the single UART (SCB mode) component. The design schematic is shown in Figure 1. The configuration for serial terminal connection is shown near the component.

Figure 1. Example Project Design Schematic



The component configuration Tabs (Basic and Advanced) are shown below.

Figure 2. UART (SCB mode) Basic Tab Configuration

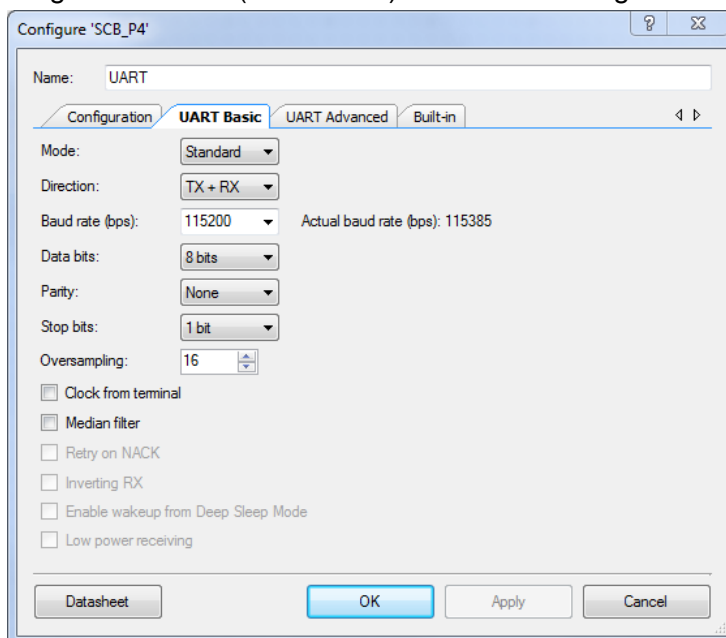


Figure 3. UART (SCB mode) Advanced Tab Configuration

Configure 'SCB\_P4'

Name: UART

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

Buffers size  
 RX buffer size: 8  
 TX buffer size: 8  
☐ Byte mode

Interrupt  
☒ None  
☐ Internal  
☐ External

DMA  
☐ RX output  
☐ TX output

Interrupt sources  
☐ UART done  
☐ TX FIFO not full  
☐ TX FIFO empty  
☐ TX FIFO overflow  
☐ TX FIFO underflow  
☐ TX lost arbitration  
☐ TX NACK  
☐ TX FIFO level  
☐ RX FIFO not empty  
☐ RX FIFO full  
☐ RX FIFO overflow  
☐ RX FIFO underflow  
☐ RX frame error  
☐ RX parity error  
☐ RX FIFO level

FIFO levels  
 TX FIFO: 0  
 RX FIFO: 7

☐ Multiprocessor mode  
 Address (hex): 2  
 Mask (hex): FF  
☐ Accept matching address in RX FIFO

RX FIFO drop  
☐ On parity error  
☐ On frame error

Flow control  
☐ RTS Polarity: Active Low  
☐ CTS Polarity: Active Low  
 RTS FIFO level: 4

Datasheet OK Apply Cancel

## Project Description

In the main firmware routine, the UART component is configured for operation and started. The constant text is transmitted to identify the example project and correct configuration of the serial terminal software. If this text is readable the serial terminal is configured properly and the following communication will return expected results.

After a text has been sent, the code enters the locked loop and polls UART for a received character. As soon as the character is received it is transmitted back. As a result every transmitted character will be re-transmitted back.

## Expected Results

To communicate with PSoC 4 from the serial terminal software, follow this procedure:



1. Connect USB Mini B to appropriate header of the kit. The kit enumerates as a **KitProg USB-UART** and is available under the Device Manager, Ports (COM & LPT). A communication port is assigned to the **KitProg USB-UART**. Note the **COMX** port number.
2. Run the available serial terminal software.
3. Select **COMX** (where X – is the specific communication port that is assigned to **KitProg USB-UART**).
4. Configure serial terminal connection with the parameters values listed in Figure 1: Baud rate – 115200, Data bits – 8, Parity – None, Stop bits – 1, and Flow control – None.
5. Build and program the SCB\_UartComm example project into the device.
6. Observe the text displayed in the serial terminal output window.

\*\*\*\*\*

This is SCB\_UartComm datasheet example project.

If you are able to read this text the terminal connection is configured correctly.

Start transmitting the characters to see an echo in the terminal.

7. Start communicating with the device as explained in the [Project Description](#).

© Cypress Semiconductor Corporation, 2015. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.

