Objective
CE60246 demonstrates the use of a custom developed USBUART component with PSoC® 3 / PSoC 5.

Overview
The USBUART component uses a USB interface to emulate a COM port. This custom component is based on a standard USB component, modified to enumerate and act as a CDC USB device. High-level communication functions are available on the PSoC device side and the PC communicates through a standard terminal program. This example project demonstrates a PSoC project using a USBUART component to echo any data sent to it from a PC terminal.

Component List

<table>
<thead>
<tr>
<th>Instance Name</th>
<th>Component Name</th>
<th>Version</th>
<th>Component Category</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>USBUART_1</td>
<td>USBUART</td>
<td>1.0</td>
<td>Custom</td>
<td>VID, PID, Device Release, Mfr String and Product String are set</td>
</tr>
</tbody>
</table>

The USBUART component is the only component used in this example. To use this component in a project, a dependency must be set pointing to the library project containing the USBUART component. The instructions for adding this component to your own project are mentioned in the following section.

Importing a Custom Component in a Project
Follow these steps to add the USBUART component to a new project:

- Right click on the project name in the Workspace Explorer and select Dependencies.
Add a new User Dependency. Navigate to and select the USBUART_Component_Library.cyprj file provided with the example project.

Ensure **Components** is selected and the **Code** check box is cleared.

Click **OK** and the USBUART component now appears in a new tab in the **Component Catalog**. It is now ready to be placed on your schematic.

**Top Design**

![USBUART component in schematic](image)
Component Configuration

USBUART

Note Each USB product must have a unique combination of Vendor ID (VID) and Product ID (PID). The default values provided here are used as an example, but a unique VID (assigned by the USB Implementers Forum) and PID must be used in a final product.

Design Wide Resources

Changes must be made in the Clocks tab in the design wide resources file (.cydwr) to use USB on the PSoC 3 device. The clocks can be configured by clicking Edit Clock in the Clocks tab. You must make the following changes.

- **IMO**: Select Osc 24.000 MHz.
- **PLL**: Select Input from IMO and desired frequency as 48MHz.
- **ILO**: Select 100 kHz.
- **USB**: Enable and select IMOx2 – 48.000 MHz.
- **Master Clock**: Select PLL_OUT– 48.000 MHz.
Operation
This project echoes any data sent to it through the PC COM terminal. During initialization, global interrupts are enabled and the USBUART component is started and enumerated. In the main loop, the receive buffer is checked for data. If data exists, the data is read out of the buffer and written back to the PC. After writing data to the PC, the PSoC must wait for the transmission to complete before continuing; otherwise, the PSoC may send new data to the PC before it is ready. This code is shown in Code 1.

Code 1 USBUART Example Code
```c
#include <device.h>

uint8 Count;
uint8 Buffer[128];

void main()
{
    /* Initialization Code: */
    CYGlobalIntEnable;
    USBUART_1_Start(0, USBUART_1_3V_OPERATION);//!!NOTE!! Make sure this matches your board voltage!
    while(!USBUART_1_bGetConfiguration());
    USBUART_1_CDC_Init();

    /* Main Loop: */
    for(;;)
    {
        Count = USBUART_1_bGetRxCount();
        if(Count != 0) /* Check for input data from PC */
        {
            USBUART_1_ReadAll(Buffer); /* Echo data back to PC */
            USBUART_1_Write(Buffer, Count);
            while(!USBUART_1_bTxIsReady()){} /* Wait for Tx to finish */
        }
    }
}
```

Apart from the standard USBFS APIs, the following APIs are implemented for this component.

- **uint8 USBUART_bGetRxCount(void)**
  - Description: Get the size of valid data in the receive buffer
  - Parameters: None
  - Return Value: uint8 : Returns the number of valid data bytes in the receive buffer

- **void USBUART_ReadAll (uint8 *pData)**
  - Description: Reads all data from the receive buffer into pData.
  - Parameters: (uint8*) pData , the pointer to the array to which data from receive buffer will be loaded into.
  - Return Value: None

- **void USBUART_Write(uint *pData, uint8 bLength)**
  - Description: Writes the data to the Tx buffer
  - Parameters:
    - uint* pData: the pointer to the array from which data will be loaded to the Tx buffer
    - uint8 bLength: The number of bytes that are written from array pData to the Tx buffer.
  - Return Value: None

- **uint8 USBUART_bTxIsReady(void)**
  - Description: Checks to see if the device is ready to send data
  - Parameters: None
  - Return Value: uint8, If the device is not ready to transmit data, 0 is returned. Else, the Tx buffer size is returned.
uint8 USBUART_CDC_Init(void)

- **Description:** Initializes the component
- **Parameters:** None
- **Return Value:** uint8, this function returns 1.
- **Note:** This function was called USBUART_Init in previous versions of the component (distributed with code example Rev *D* and earlier).

**Hardware Connections**

- You can use the default jumper settings of CY8CKIT-001 available in PSoC Development Kit Board Guide. This project should be adaptable to work on almost any board with USB capabilities. Note this does not include the CY8CKIT-003 FirstTouch Starter Kit, which does not have a USB interface.
- Jumper 8 may be changed to VBUS location, if you wish to power the board through USB.

**Output**

- Build the project and program the chip.
  - **Note:** The default device selection is PSoC 3 (CY8C3866AXI-040). To use this project with PSoC 5 family, do the following:
  - Go to **Project → Device Selector →** Select **PSOC 5** device (CY8C5588AXI-060), build the project again and program the PSoC 5 device as follows:

    - Plug the USB cable into the DVK and reset the board (SW4). When Windows prompts you with Found New Hardware Wizard, direct the wizard to the \USBUART_Example_Project\USBUART_Example_Project.cysn\Generated_Source\PSoC 3 directory, where the USBUART_CDC.inf file is located. This allows the device to enumerate correctly.
- Check to see which COM port the device is mapped to in the **Device Manager**.

![Device Manager](image)

- Open a COM terminal (such as HyperTerminal) and select the appropriate port. The baud rate selected does not matter. Ensure that the local echo is turned OFF.

- When a connection is established, the PSoC returns any information typed in the terminal.

**Related Application Notes and Example Projects**

- **USB Peripheral Basics** - AN57294
- **PSoC® 3 / PSoC 5 USB HID Fundamentals** - AN57473
- **PSoC® 3 / PSoC 5 USB HID Intermediate** - AN58726
- **USB Bulk Loopback With PSoC® 3** - AN56718
- **USB Vendor Commands with PSoC® 3** - AN56377
- **Interrupt Loopback** - PSoC® 3 / PSoC 5
- **Isochronous Transfers in PSoC® 3 / PSoC 5**
### Document History

**Document Title:** USBUART in PSoC® 3 / PSoC 5  
**Document Number:** 001-60246

<table>
<thead>
<tr>
<th>Revision</th>
<th>ECN</th>
<th>Orig. of Change</th>
<th>Submission Date</th>
<th>Description of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>2889259</td>
<td>DRSW</td>
<td>03/16/2010</td>
<td>New example project</td>
</tr>
</tbody>
</table>
| *A       | 2930450 | DRSW | 05/10/2010 | 1. Fixed mistake in clock setting instructions and updated related screen shot  
2. Updated project to fix a warning message and mistake in component datasheet.  
3. Shortened some folder names and library project name. |
| *B       | 2959649 | DRSW | 06/23/2010 | Updated project and document to support Beta 4.1 and PSoC 5. |
| *C       | 3013689 | DRSW | 08/23/2010 | Updated software version. |
| *D       | 3119539 | DRSW | 01/21/2011 | GCC compiler warnings are fixed and project rebuilt in FCS RC9.  
Two copies of the project attached - one with ES2 as the default device,  
one with ES3 as the default device. Otherwise, the projects are identical. |
| *E       | 3287514 | DRSW | 06/20/2011 | Updated component to fix enumeration bug with 5 V operation that occurred on a small number of USB hubs.  
Updated project and example code to use new USBUART_CDC_Init API  
Specified this example project does not work on the CY8CKIT-003 FirstTouch Starter Kit, which does not have a USB interface. |

**PSoc** is a registered trademark of Cypress Semiconductor Corp. **PSoC Creator** is a trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

---

© Cypress Semiconductor Corporation, 2010-2011. 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.

This 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.