AN85514

Designing a USB-to-RS232 Solution Using Cypress's USB-UART LP Bridge Controller

Author: Madhura Tapse
Associated Project: Yes
Associated Part Family: CY7C65213,CY7C65211,CY7C65215
Associated Software: USB-Serial SDK
Related Resources: Click Here
To get the latest version of this application note, or the associated project file, visit http://www.cypress.com/go/AN85514

Cypress's USB-UART LP Bridge Controller (CY7C65213) brings plug-and-play USB connectivity to legacy UART peripherals. AN85514 describes how to implement a USB-to-RS232 solution using CY7C65213.

Contents

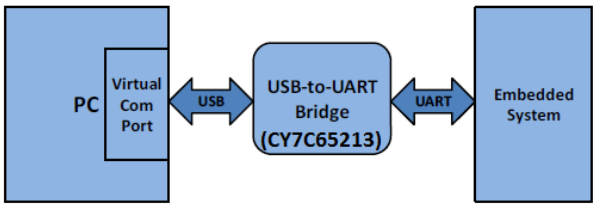
1 Introduction....1
2 Applications....2
3 CY7C65213 USB-UART LP Bridge Controller....2
4 CY7C65213 Interface with MCU....3
5 USB-to-RS232 Cable Solution....5
6 DB9 Connector....6
7 Testing a USB-to-RS232 Cable Solution....7
7.1 Using Cypress Serial Port Test Tool....7
7.2 Using Serial Terminal Application....11
8 Controlling GPIOs of USB-UART LP Bridge Controller....14
A Appendix A: USB-to-RS232 Solution Schematics....15
B Appendix B: Related Resources....17
Document History....18
Worldwide Sales and Design Support....19

1 Introduction

USB technology is ubiquitous and remains the interface of choice between PCs and peripherals/systems. Modern PCs have only USB ports to connect external peripherals/systems. Many systems still use the legacy serial port (UART/RS232) as the interface, which appears as a COM port on the PC. To enable these systems to connect to modern PCs, a “bridge” chip to convert the UART protocol to USB is required.

Cypress’s USB-UART Low Power (LP) Bridge Controller (CY7C65213) enables seamless connectivity between USB-based PCs and systems with a UART interface, as shown in Figure 1. It is a low-power, single-chip, plug-and-play solution that is easy to design. The USB-UART LP Bridge Controller enumerates as a standard COM port on the PC, enabling existing software applications to be reused and accelerating time to market. As you can see in Figure 1, the PC communicates through the bridge as a standard COM port, and the USB-UART LP Bridge Controller seamlessly converts USB signaling to standard UART signals for the peripheral device.

Figure 1. USB-to-RS232–Based System



2 Applications

The USB-UART LP Bridge Controller can be used in the following applications:

- USB-to-RS232 cables
- Bridging applications to enable a connection between USB PCs and legacy RS232 devices
- Point-of-sale (POS) terminals, scanners, and printers
- Industrial and metering devices
- Medical devices
- Set-top boxes
- Servers

Figure 2 illustrates the connectivity between an RS232-enabled barcode scanner and a USB-enabled PC. You can also connect other RS232-enabled devices such as medical devices, magnetic card readers, terminals, or any other legacy device, using an RS232 interface.

Figure 2. USB-to-RS232 Bridge Connectivity to PC



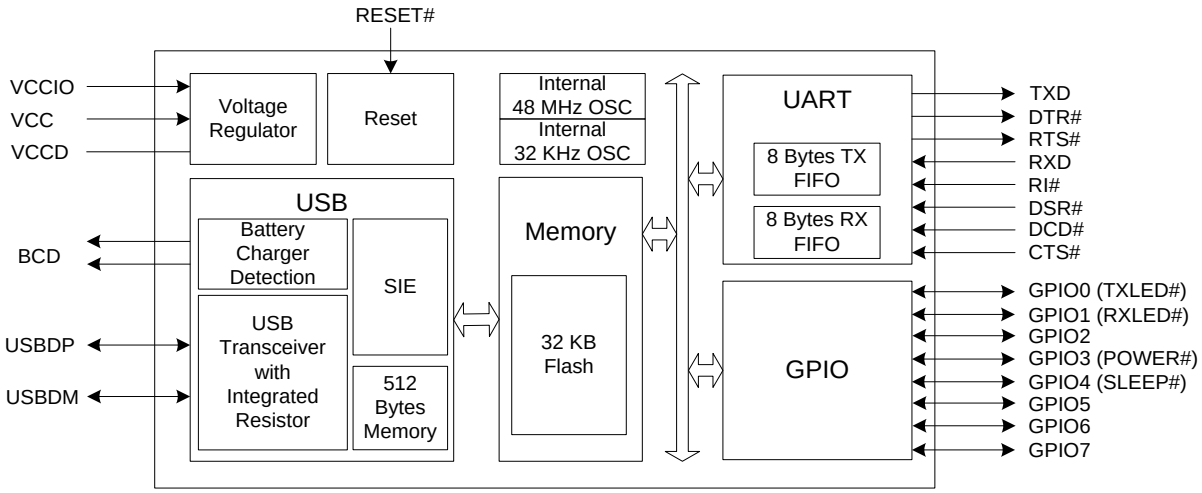
3 CY7C65213 USB-UART LP Bridge Controller

CY7C65213 is a single-chip bridge controller, with an integrated USB Full-Speed device controller, UART controller, internal voltage regulator, clock generator, flash, and USB-IF-compliant battery charger detection in 28-pin SSOP and 32-pin QFN packages. This USB-UART LP Bridge Controller can be designed for bus-powered, mixed-powered and self-powered applications. The Suspend, POWER#, and RI# pins enable an efficient power management design to meet the low-power requirements of the USB specification.

CY7C65213 supports UART data rates from 300 bps to 3 Mbps with 7 or 8 data bits, 1 or 2 stop bits, and even/odd/mark/space or no parity.

Figure 3 shows a block diagram of the Cypress USB-UART LP Bridge Controller solution. For more information, see the CY7C65213: USB-UART LP Bridge Controller datasheet.

Figure 3. CY7C65213 Block Diagram



4 CY7C65213 Interface with MCU

CY7C65213 can serve as an interface between an MCU system with a UART interface (embedded systems) and a PC, as shown in Figure 4. In this bus-powered design, both VCC and VCCIO are powered by VBUS from the USB connector. CY7C65213 has eight UART pins (TxD, RxD, RTS#, CTS#, DTR#, DSR#, DCD#, and RI#), which are connected directly to the MCU. The RxD and TxD pins are used for data transmission. The RTS#, CTS#, DTR#, and DSR# pins are used for hardware flow control. DCD# and RI# pins can be used to support legacy UART applications such as modems.

As Figure 4 shows, an RS232 level translator is not required because there is interchip connectivity, reducing the BOM cost and complexity.

Figure 5 shows a mixed-powered CY7C65213-based USB-UART design. In this design, the VCC pin is powered by VBUS from the USB connector, and VCCIO is powered by the system power supply.

Figure 6 shows a self-powered USB-UART design based on the CY7C65213 device powered at 3.3 V. In systems where USB-UART LP Bridge Controllers are embedded (connected to a USB Host or a hub within a PCB or system), CY7C65213 can be powered with a 3.3-V power supply connected to the VCC and VCCIO pins, as shown in Figure 6. This reduces the power consumed by CY7C65213 by 10 percent in suspend mode. The 3.3-V internal regulator on VCC/VBUS should be disabled using the Cypress USB-Serial Configuration Utility (see the USB-Serial Configuration Utility User Guide or CYUSBS232 User Guide).

Figure 4. Cypress's USB-UART Bridge Interface with MCU (Bus-Powered Design)

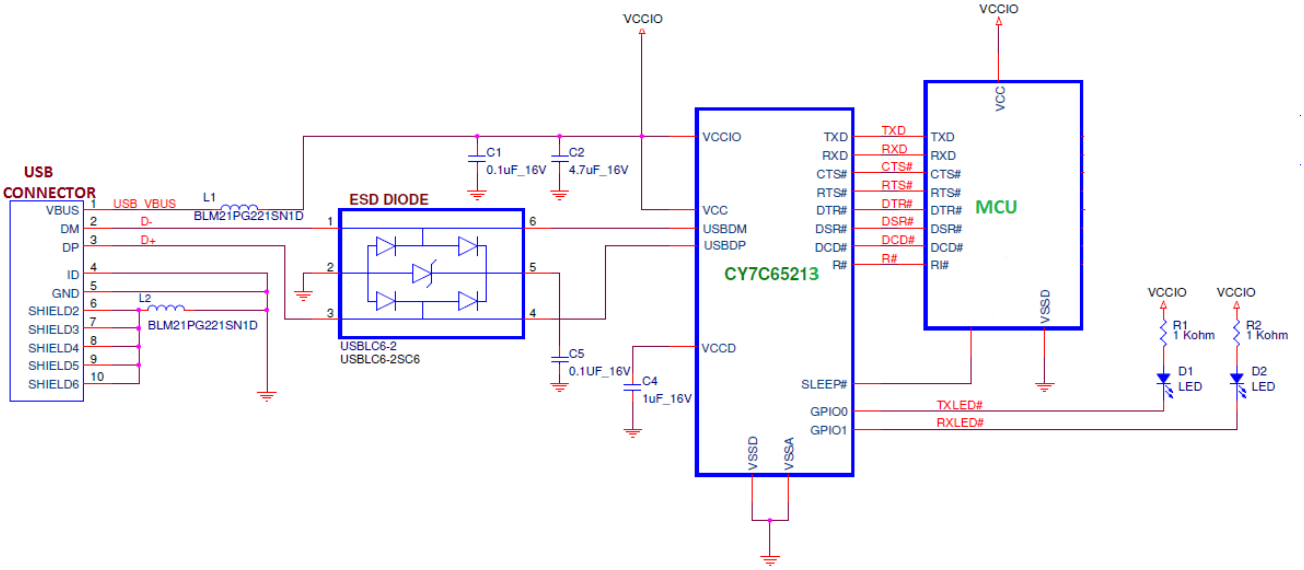


Figure 5. Cypress's USB-UART Bridge Interface with MCU (Mixed-Powered Design)

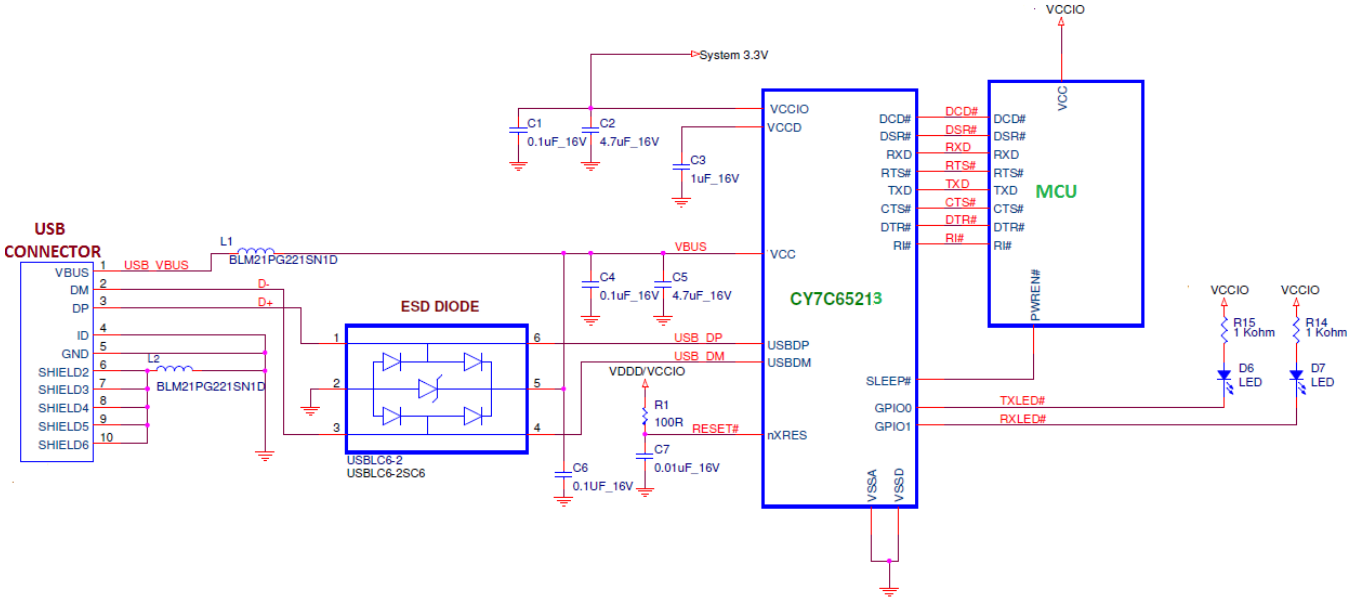
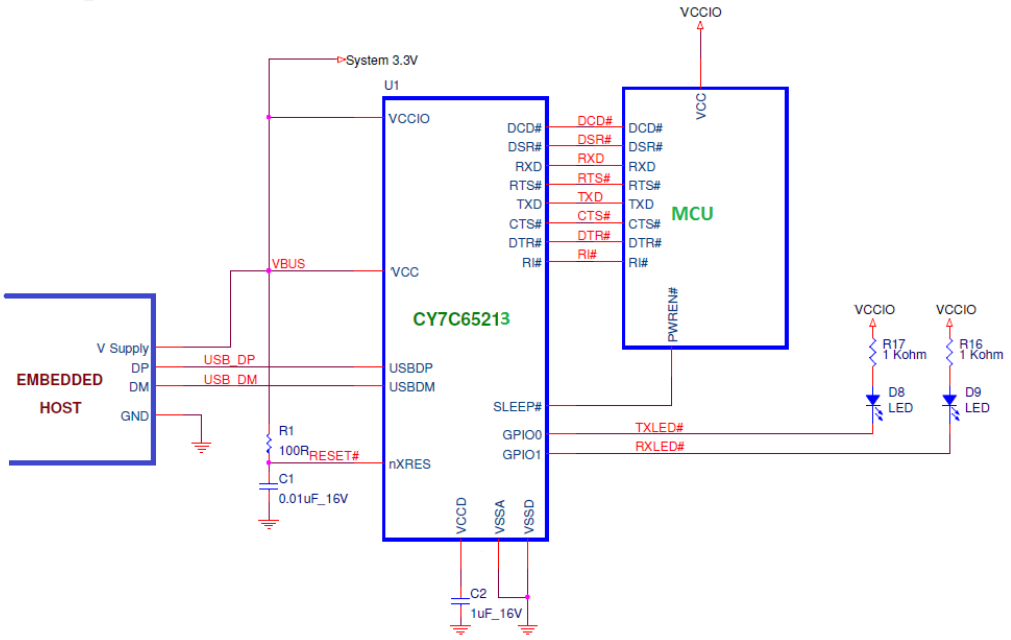


Figure 6. Cypress's USB-UART Bridge Controller in an Embedded System (Self-Powered Design)

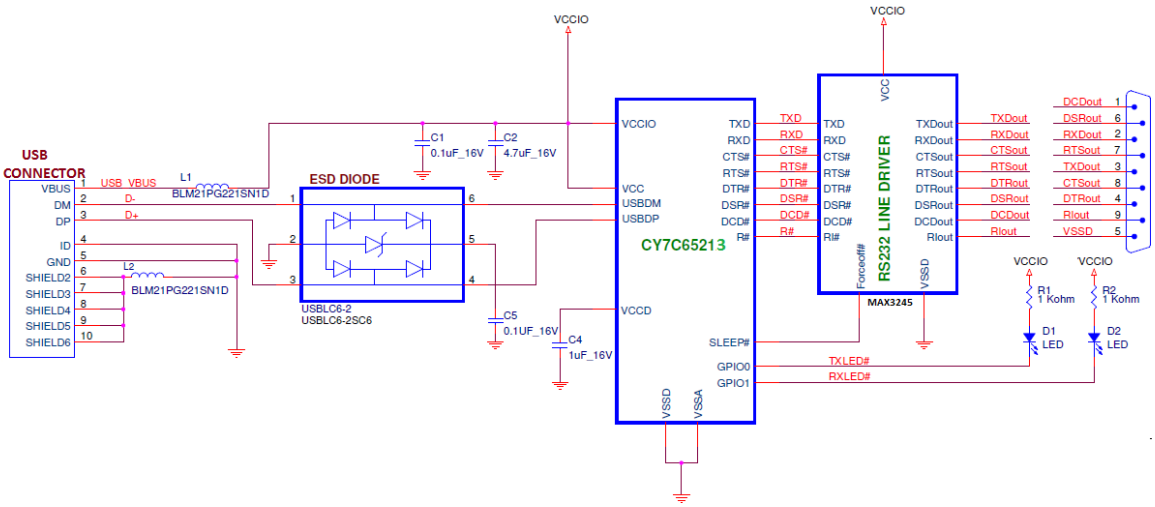


5 USB-to-RS232 Cable Solution

The USB-to-RS232 cable solution is a general-purpose application that provides PC connectivity to peripherals with an RS232 interface. This implementation has a USB connector on one end and a DB9 male connector (RS232 serial port) on the other end. See the DB9 Connector section for more details.

The RS232 serial port (DB9 connector) operates at RS232 voltage levels. The UART interface of the USB-UART LP Bridge Controller operates at TTL level. To get the USB-UART LP Bridge Controller's UART signals at the RS232 voltage level, the UART interface of the bridge controller needs to be connected to an RS232 line driver, which provides the level conversion for RS232 signaling, as shown in Figure 7.

Figure 7. USB-to-RS232 Solution Using CY7C65213



The RS232 protocol follows bipolar signaling, where the output signal toggles between negative and positive voltages. A logic '1' is called a "mark," with a voltage between –3 V and –15 V; a logic '0' is called a "space," with a voltage between +3 V and +15 V. For noise immunity, the minimum transmit levels are –5 V for a mark and +5 V for a space. The RS232 line driver performs the voltage level translation between the CY7C65213 UART interface and the RS232 device.

The UART input lines of CY7C65213 (DSR#, CTS#, and RxD) should be connected to the logic outputs of the RS232 line driver chip. The UART output lines of CY7C65213 (DTR#, RTS#, and TxD) should be connected to the logic inputs of the RS232 line driver chip. CY7C65213 has been tested with Maxim's MAX3245 device.

Appendix A provides the schematics for a USB-to-RS232 solution.

6 DB9 Connector

DB9 connectors are used for RS232 serial communication that enables asynchronous data transmission. Figure 8 shows a DB9 male connector, and Table 1 describes the pinout of the DB9 male connector.

Figure 8. DB9 Male Connector

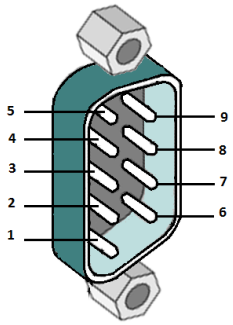


Table 1. DB9 Male Connector Pinout

DB9	Signal	Description	Direction
1	DCD#	Data Carrier Detect	IN
2	RxD	Receive Data	IN
3	TxD	Transmit Data	OUT
4	DTR#	Data Terminal Ready: Used by the data terminal to signal to the data set that it is ready for operation, active LOW	OUT
5	GND	Signal Ground	
6	DSR#	Data Set Ready: Used by the data set to signal to the data terminal that it is ready for operation and ready to receive data, active LOW	IN
7	RTS#	Request To Send: Handshaking, active LOW	OUT
8	CTS#	Clear To Send: Handshaking, active LOW	IN
9	RI#	Ring Indicator: Used by the data set to indicate to the data terminal that a ringing condition has been detected	IN

7 Testing a USB-to-RS232 Cable Solution

A USB-to-RS232 cable solution can be tested by the following two methods:

1. Using the Cypress Serial Port Test Tool
2. Using any serial terminal application such as Tera Term

7.1 Using Cypress Serial Port Test Tool

This test tool can be used to test the USB-to-RS232 cable in three different configurations as follows:

- **Loopback test:** In this test, only one USB-to-RS232 cable is used. RxD and TxD, RTS# and CTS#, and DTR#, and DSR# of the same cable are connected to form a loopback connection.
- **Back-to-back test:** In this test, two USB-to-RS232 cables are used. RxD, CTS#, and DSR# signals of one cable are connected to the TxD, RTS#, and DTR# signals of a second cable and vice versa to form a cross-cable connection.
- **Y cable test:** This test checks the DCD# and RI# signals of the USB-to-RS232 cable along with the TxD, RxD, RTS#, CTS#, DTR# and DSR# signals.

7.1.1 Loopback Test

The loopback test sends data from the PC to the USB-UART cable and returns it (looped back) to the PC to determine if the USB-UART LP Bridge Controller works or not. As shown in Figure 7, RxD and TxD, RTS# and CTS#, and DTR# and DSR# signals of the same cable are connected to form a loopback connection This test checks the RxD, TxD, RTS#, CTS#, DTR#, and DSR# signals of the USB-UART cable. If the data sent from the PC matches the received data, the loopback test is passed.

Hardware requirements:

- One CY7C65213-based USB-to-RS232 cable
- A PC with one USB port (running Windows XP or later)

Software and driver requirements:

Download the Cypress Serial Port Test Tool from the Cypress website.

The Microsoft Windows driver for the USB-UART LP Bridge Controller is installed from the Windows website automatically when the USB-UART LP Bridge Controller is plugged into the USB Host controller and enumerated. Alternatively, you can download the UART Communication Device Class (CDC) and vendor driver from the Cypress website. After this driver installation, both Microsoft CDC APIs as well as Vendor APIs work with USB-UART LP Bridge Controller. For Linux and Mac, no driver installation is required. The native CDC class driver supports the USB-UART LP Bridge Controller.

Test procedure for loopback test:

1. Complete the hardware connections as shown in Figure 9.
2. In the test tool, select the **Enable Loopback** option, as shown in Figure 10; then, select the Cypress USB Serial COM port from the **Device 1 (DUT)** drop-down menu.
3. Click **Run Test**.
4. If the tests are completed successfully, the utility displays “All tests completed successfully” and a green check mark appears.

Figure 9. Hardware Connections for Loopback Test

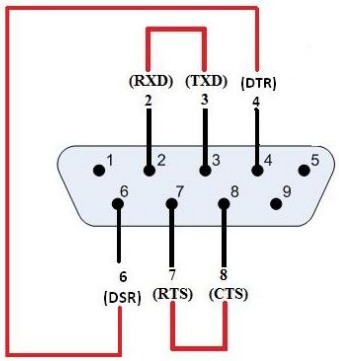
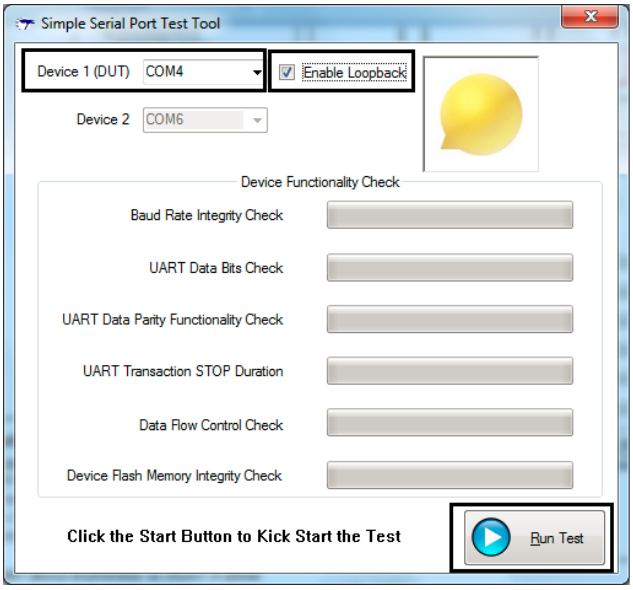


Figure 10. Running Cypress Serial Port Test Tool for Loopback Test



7.1.2 Back-to-Back Test

The back-to-back test transfers data from one USB-UART cable to another that is connected to a USB port on a PC. As shown in Figure 11, RxD, CTS#, and DSR# signals of one cable are connected to the TxD, RTS#, and DTR# signals of the second cable and vice versa to form a cross-cable connection. This test checks the RxD, TxD, RTS#, CTS#, DTR#, and DSR# signals of the USB-UART cable. If the data sent by the PC matches the received data, the back-to-back test is passed.

Hardware requirements:

- Two CY7C65213-based USB-to-RS232 cables
- A PC with two USB ports (running Windows XP or later)

Software and driver requirements:

Download the Cypress Serial Port Test Tool from the Cypress website.

The Microsoft Windows driver for the USB-UART LP Bridge Controller is installed from the Windows website automatically when the USB-UART LP Bridge Controller is plugged into the USB Host controller and enumerated. Alternatively, you can download the UART CDC and vendor driver from the Cypress website. After this driver installation, both Microsoft CDC APIs as well as Vendor APIs work with USB-UART LP Bridge Controller. For Linux and Mac, no driver installation is required. The native CDC class driver supports the USB-UART LP Bridge Controller.

Test procedure for back-to-back test:

1. Complete the hardware connections as shown in Figure 11.
2. In the test tool, deselect the **Enable Loopback** option, as shown in Figure 12.
3. Select the Cypress USB Serial COM port in two drop-down menus: **Device1 (DUT)** and **Device 2**.
4. Click **Run Test**.
5. If the tests are completed successfully, the utility displays “All tests completed successfully,” and a green check mark appears.

Figure 11**.** Hardware Connections for Back-to-Back Test

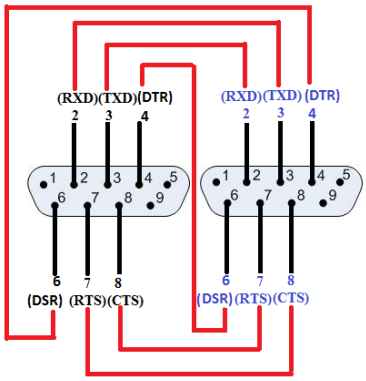
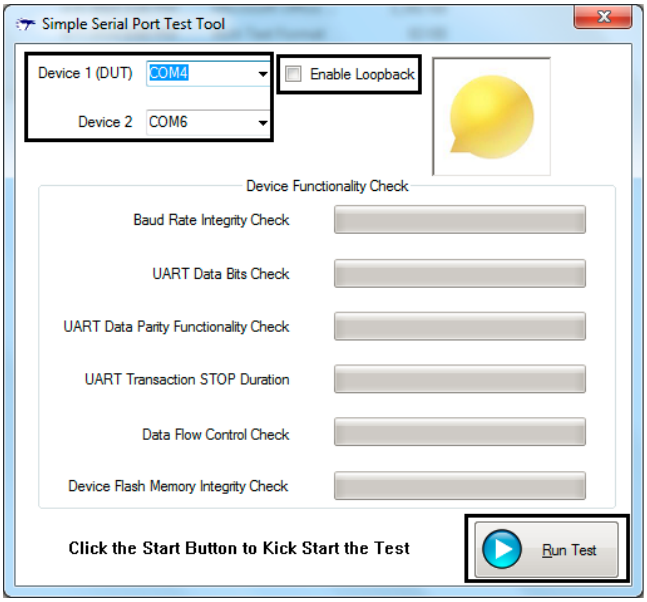


Figure 12. Running Cypress Serial Port Test Tool for Back-to-Back Test



7.1.3 Y Cable Test

A Y cable test is a test in which the DCD# and RI# signals of a USB-UART cable are verified along with the TxD, RxD, RTS#, CTS#, DTR#, and DSR# signals.

In this test, data is sent from a PC to a USB-UART cable, which is the device under test (DUT). This USB-UART cable (DUT) is connected to two other USB-UART cables (device 2 and Y cable device), as shown in Figure 13, which may or may not be connected to the same PC. The TxD, RxD, RTS#, CTS#, DTR#, and DSR# signals of DUT are connected to the RxD, TxD, CTS#, RTS#, DSR#, and DTR# signals of device 2 respectively. The DCD# and RI# signals of DUT are connected to the DTR# and RTS# signals of the Y cable device.

If the data sent by the PC matches the data received by the Y cable, the test is passed.

Hardware requirements:

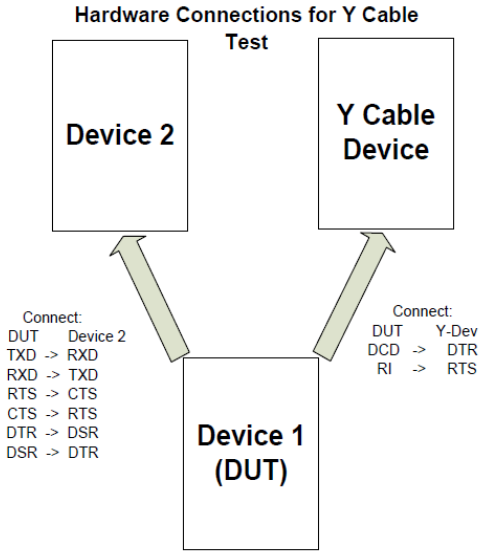
- Three CY7C65213-based USB-to-RS232 cables
- A PC with three USB ports (running Windows XP or later)

Software and driver requirements:

Download the Cypress Serial Port Test Tool from the Cypress website.

The Microsoft Windows driver for the USB-UART LP Bridge Controller is installed from the Windows website automatically when the USB-UART LP Bridge Controller is plugged into the USB Host controller and enumerated. Alternatively, you can download the UART CDC and vendor driver from the Cypress website. After this driver installation, both Microsoft CDC APIs as well as Vendor APIs work with USB-UART LP Bridge Controller. For Linux and Mac, no driver installation is required. The native CDC class driver supports the USB-UART LP Bridge Controller.

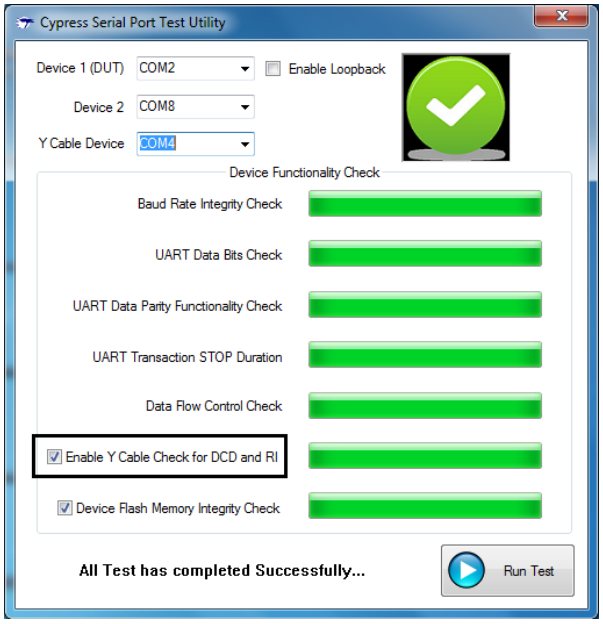
Figure 13. Hardware Connections for Y Cable Test



Test procedure for Y cable test:

1. Complete the hardware connections as shown in Figure 11. Make note of the COM port numbers of **Device 1 (DUT)**, **Device 2**, and **Y Cable Device.**
2. Select the **Enable Y Cable Check for DCD and RI** option, as shown in Figure 14.
3. Select the Cypress USB Serial COM ports in the three drop-down menus: **Device 1 (DUT)**, Device 2, and **Y Cable Device.**
4. Click **Run Test**.
5. If the tests are completed successfully, then the utility displays “All tests completed successfully,” and a green check mark appears.

Figure 14. Running Cypress Serial Port Test Tool for Y Cable Test



7.2 Using Serial Terminal Application

7.2.1 Hardware Requirements

- Two CY7C65213-based USB-to-RS232 cables
- A 9-pin female-to-female crossover RS232 serial cable, with connections for the “A” and “B” ends of the cable:

DB9 A	DB9 B
2	3
3	2

- A PC with two USB ports (running Windows XP or later)

7.2.2 Software and Driver Requirements

Tera Term, HyperTerminal, or a similar terminal application must be installed on the PC.

The Microsoft Windows driver for the USB-UART LP Bridge Controller is installed from the Windows website automatically when the USB-UART LP Bridge Controller is plugged into the USB Host controller and enumerated. Alternatively, you can download the UART CDC and vendor driver from the Cypress website. After this driver installation, both Microsoft CDC APIs as well as Vendor APIs work with USB-UART LP Bridge Controller. For Linux and Mac, no driver installation is required. The native CDC class driver supports the USB-UART LP Bridge Controller.

7.2.3 Test Procedure

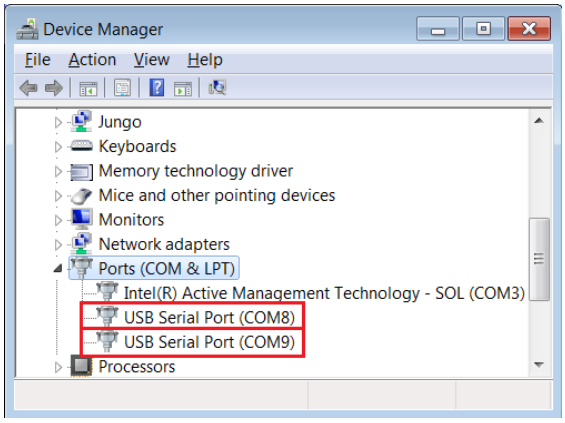
1. Connect two CY7C65213-based USB-to-RS232 cables to two different USB ports of a PC. Connect these devices using a female-to-female serial cable, as shown in Figure 15.

Figure 15. Back-to-Back Setup



2. Open the **Device Manager** on the PC and check for the enumerated Cypress USB-UART LP Bridge Controller and the corresponding COM ports, as shown in Figure 16.

Figure 16. Cypress Serial Ports Listed in Device Manager



3. Open Tera Term (or any other serial terminal application) and select the COM ports for communication in the Setup/Options menu, as shown in Figure 17 and Figure 18.

Figure 17. Access One COM Port in Tera Term

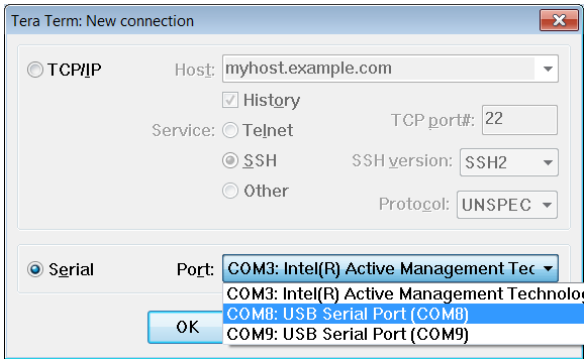
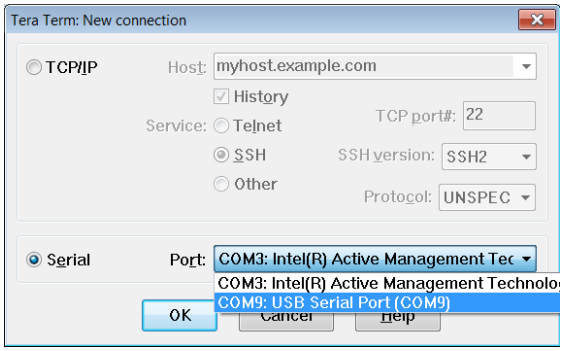
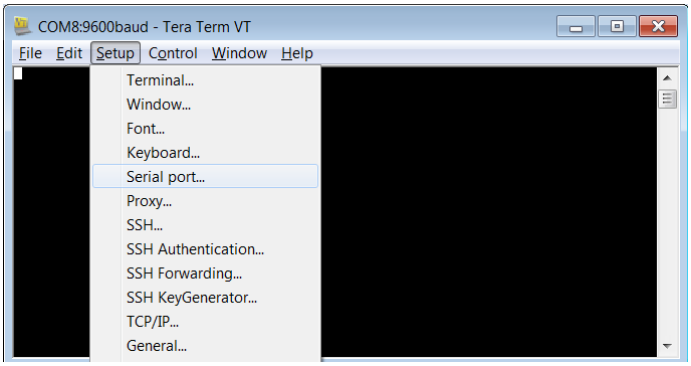


Figure 18. Access Another COM Port in Tera Term



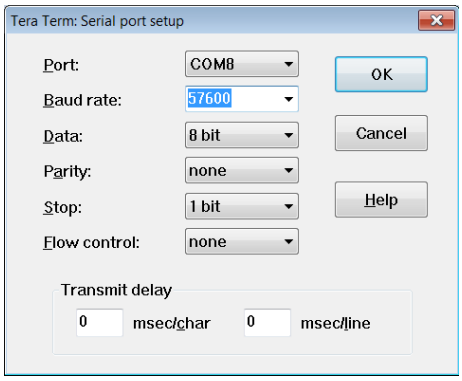
4. You can use the terminal application's **Setup** option to modify the parameters of the COM port (data bits, parity, baud rate, and flow control), as shown in Figure 19 and Figure 20.

Figure 19. Serial Port Configuration



5. Select any **Baud rate** (for example, 57600, as shown in Figure 20) and click **OK.**

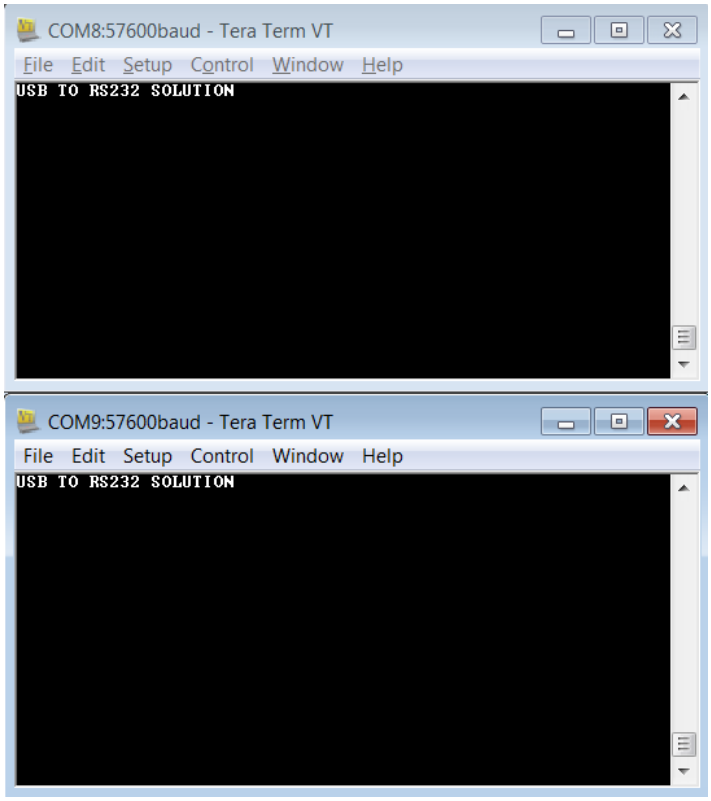
Figure 20. Tera Term Serial Port Setup Window



6. Repeat step 4 for the second COM port.
7. Type characters and numbers into one Tera Term terminal application and observe the second terminal application window.

Figure 21 shows the data transfer between two CY7C65213-based USB-UART cables.

Figure 21. Data Transfer over Terminals



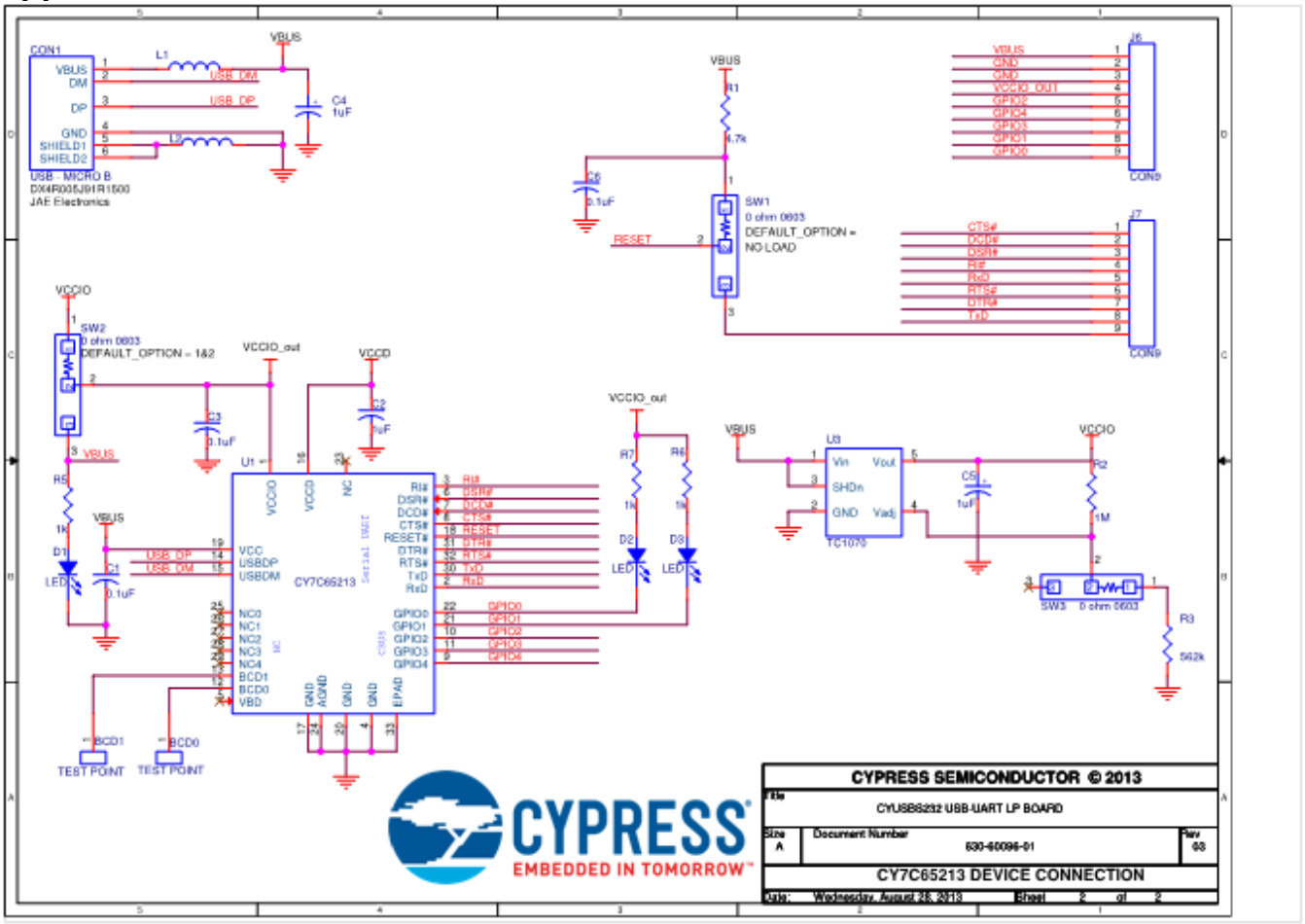
8 Controlling GPIOs of USB-UART LP Bridge Controller

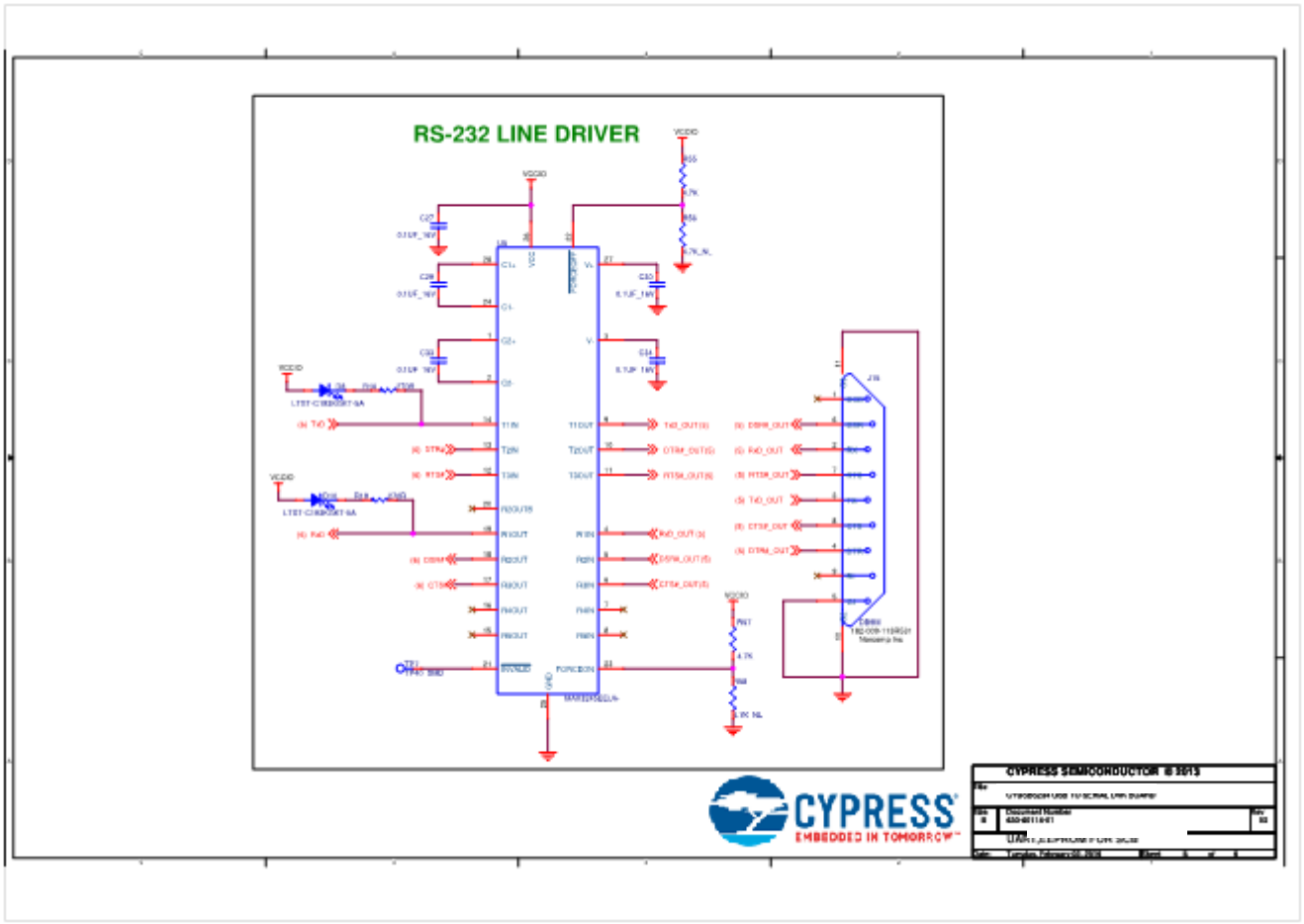
The USB-UART LP Bridge Controller (CY7C65213) as well as USB-Serial Bridge Controllers (CY7C65211 and CY7C65215) support I/O control using APIs in Windows, Linux, Android, and Mac OS systems. These controllers support four I/O modes — Input, Tristate, Drive1, and Drive 0. All these I/O lines are tri-stated by default in firmware and can be configured by using the Cypress USB-Serial Configuration Utility. The status of the input GPIO can be retrieved by using the CyGetGPIOValue() API. The status of the output GPIO can be set to Drive 1 or Drive 0 by using the CySetGPIOValue() API. Note that Input and Tristate I/O modes can be set only by using the Cypress USB-Serial Configuration Utility. See the knowledge base article KBA92641 for an example code, which demonstrates the usage of the GPIO control APIs, CyGetGPIOValue() and CySetGPIOValue().

USB-Serial Bridge Controllers (CY7C65211 and CY7C65215) support configurable I^2C and SPI interfaces. Cypress provides I^2C and SPI control APIs in Windows, Linux, Android and Mac OS systems.

See the Cypress USB-Serial API Guide for more information on these APIs. See the USB-Serial Software Development Kit for complete documentation and example codes.

A Appendix A: USB-to-RS232 Solution Schematics




RS-232 LINE DRIVER
CYPRESS SEMICONDUCTOR © 2013

B Appendix B: Related Resources

Cypress USB-UART and USB-Serial Bridge Controller design resources include datasheets, development kits, reference designs, firmware, and software tools. The resources are summarized in Table 2.

Table 2: USB-UART LP/USB-Serial Bridge Controller: Related Resources

Design	Available Resources	Where To Find Resources
Datasheets	CY7C65213 USB-UART LP Bridge Controller Datasheet	Datasheet Link
	CY7C65211 USB-Serial Single Channel Bridge Controller Datasheet	Datasheet Link
	CY7C65215 USB-Serial Dual Channel Bridge Controller Datasheet	Datasheet Link
Hardware	Development Board – Schematic, Board files, and documentation	Development Kit (DVK) Schematic Board files available with ■ CYUSBS232 USB-UART LP RDK (For CY7C65213) ■ CYUSBS234 USB-Serial DVK (For CY7C65211) ■ CYUSBS236 USB-Serial DVK (For CY7C65215)
	IBIS models	IBIS model files
Code Examples	USB-Serial SDK includes the following code examples: ■ Example to communicate with other UART devices when USB-UART LP or USB-Serial Bridge Controller is configured as UART in both CDC and vendor modes ■ Example to communicate with an I^2C slave device when USB-Serial Bridge Controller is configured as the I^2C master in both CDC and vendor modes ■ Example to communicate with an SPI slave device when USB-Serial Bridge Controller is configured as the SPI master in both CDC and vendor modes ■ Example to access USB-UART LP or USB-Serial Bridge Controller and control GPIOs	USB-Serial Software Development Kit
Host PC Software	GUI-based Windows application to configure USB-UART LP and USB-Serial Bridge controllers. This application is located in USB-Serial Software Development Kit.	USB-Serial Configuration Utility User Guide
Drivers	USB-Serial Driver Installer for Windows installs all required UART CDC and UART/I^2C/SPI vendor class drivers.	USB-Serial Driver Installer - Windows

Document History

Document Title: AN85514 - Designing a USB-to-RS232 Solution Using Cypress's USB-UART LP Bridge Controller

Document Number: 001-85514

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	3853976	DTNK/SETU	12/30/2012	New application note
*A	4505587	RSKV	09/18/2014	Modified abstract. Removed Figure 3. Updated steps in Test Procedure section.
*B	4682506	MVTA/DBIR	03/16/2015	Updated application note with Gen2 USB-UART controller CY7C65213. Updated template
*C	5105459	SMSN/MVTA	02/12/2016	Updated application note with related resources section
*D	5698250	AESATP12	04/26/2017	Updated logo and copyright

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

Cypress Developer Community

Forums | WICED IOT 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, 2012-2017. 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. 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.