HID Reference Keyboard User Guide

About this document

Scope and purpose

This document is the user guide for HID Reference Keyboard Design.

Table of contents

1 Introduction

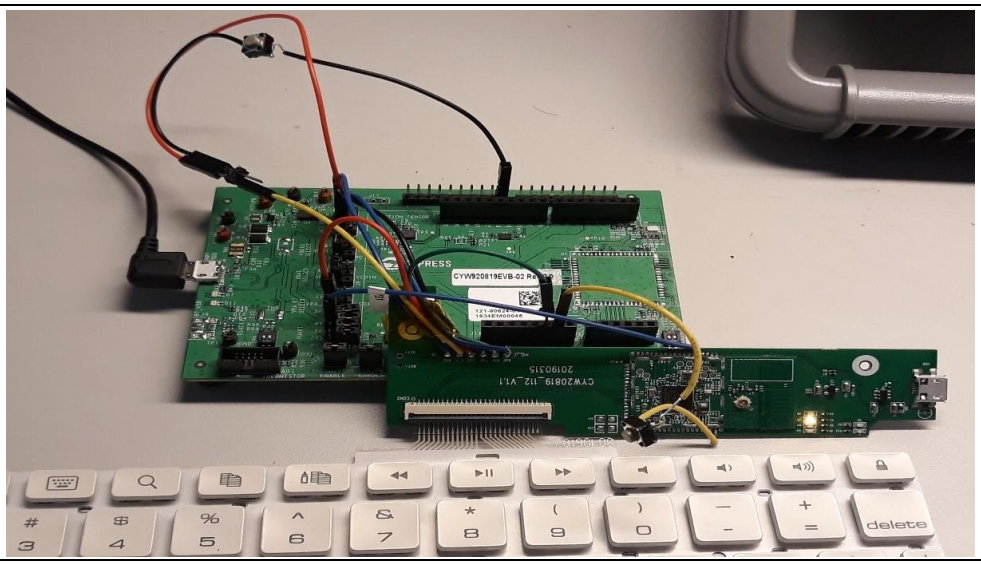
The CYW920819REF-KB-01 platform is designed as a Cypress HID Reference Keyboard, using the CYW920819EVB-02 Evaluation Kit connected to the keyboard hardware as described in this document. It is supported in ModusToolbox® 2.0 with BTSDK 2.1 (or higher). It can be programmed with BR/EDR and/or LE Bluetooth applications to demonstrate a standard Bluetooth keyboard device.

BTSDK 2.1 supplies the “dual_mode_keyboard” Code Example (“HID_20819REF_KB.dual_mode_keyboard” as shown in the ModusToolbox IDE), a sample application that demonstrates both BR/EDR and LE Bluetooth keyboard functionality on the platform. It can be paired with BR/EDR HID hosts or LE Bluetooth HID Over GATT Protocol (HOGP) host devices.

The keyboard is powered by the USB port; however, only power and ground are connected to the keyboard. The port is purely used as a power supply. The keyboard is designed to upgrade firmware over the air. When OTA firmware upgrade is not available, the only way to program firmware is through direct wire connections. This photo shows the keyboard, USB power connection, and necessary programming signals brought out to a connector for wired programming.

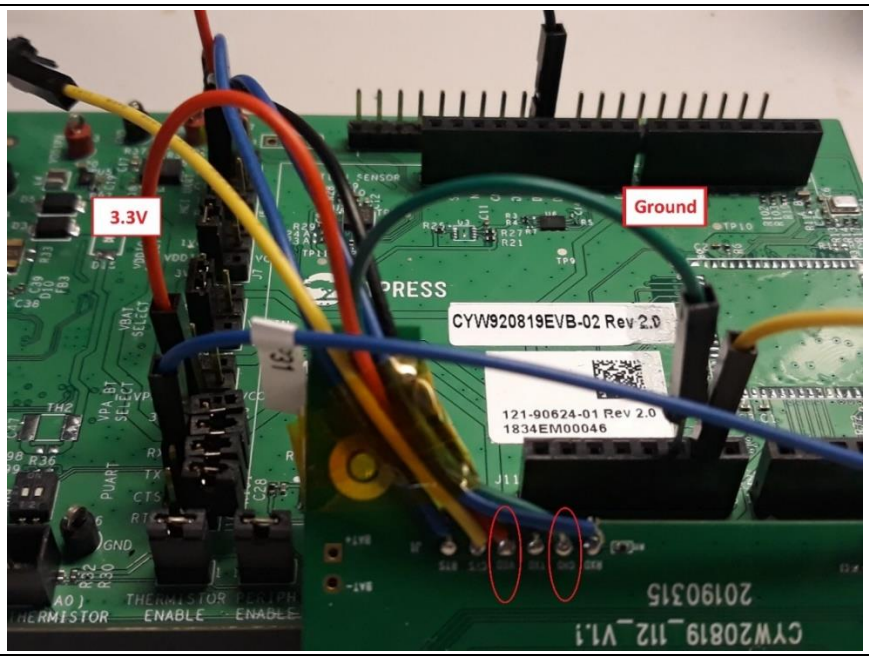
2 Hardware Setup

This chapter describes setting up the hardware using the CYW9208xxEVB-02 Board (EVB) as a UART-USB adapter and power supply.

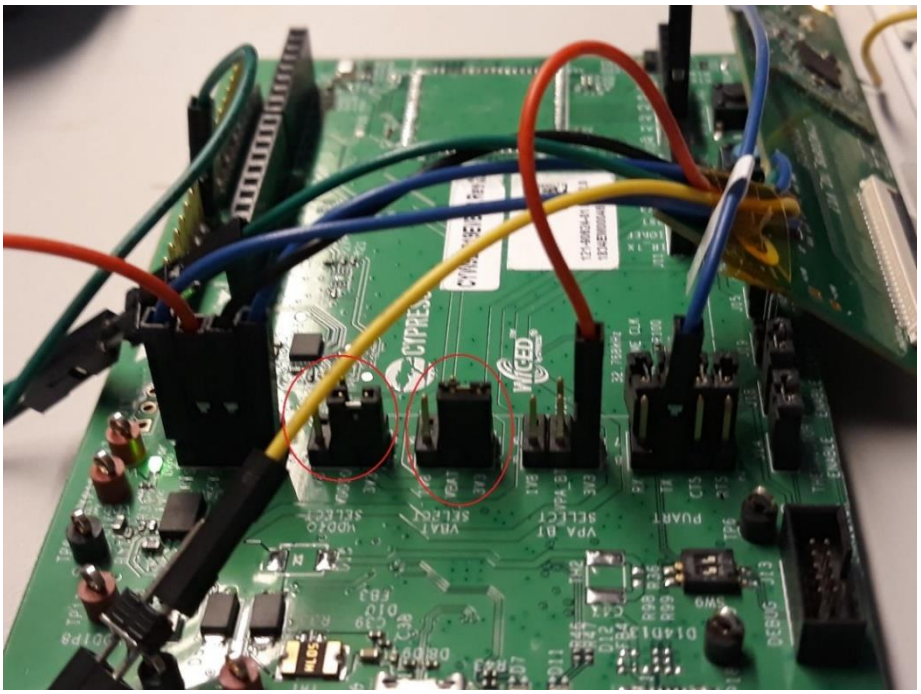


2.1 Power and Ground

1. Connect VDD to EVB J16-P2 where it is labeled as “3V3”.
2. Connect GND to EVB J11-P6, GND.

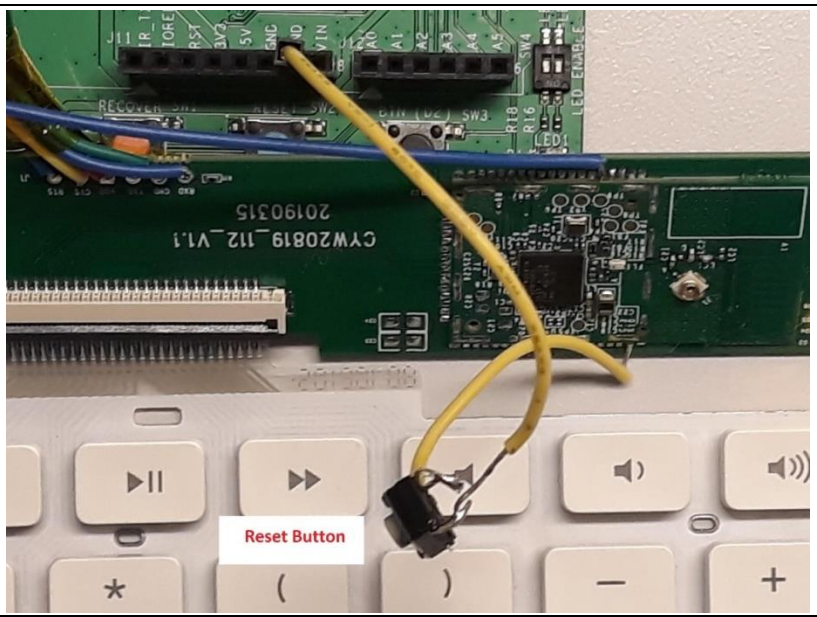


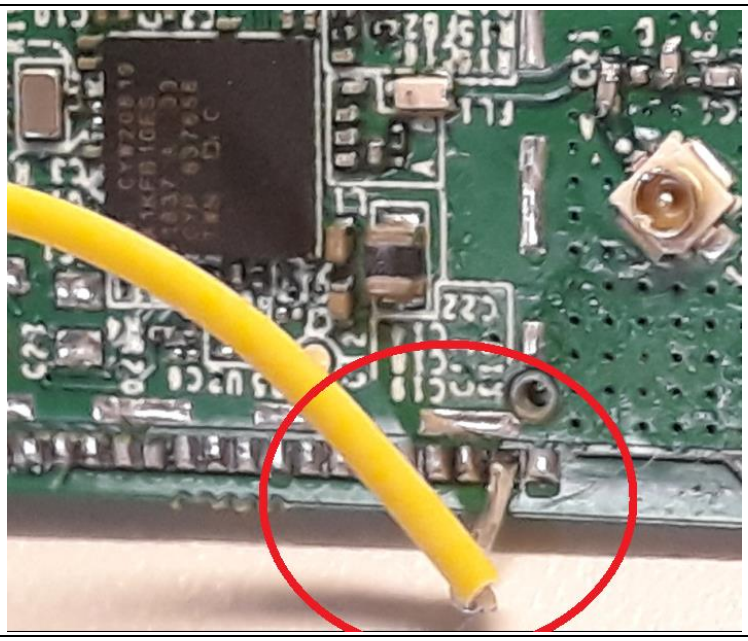
3. To make sure that the board is powered, put jumpers between EVB P2-P4 in J7 and J8 as shown below:



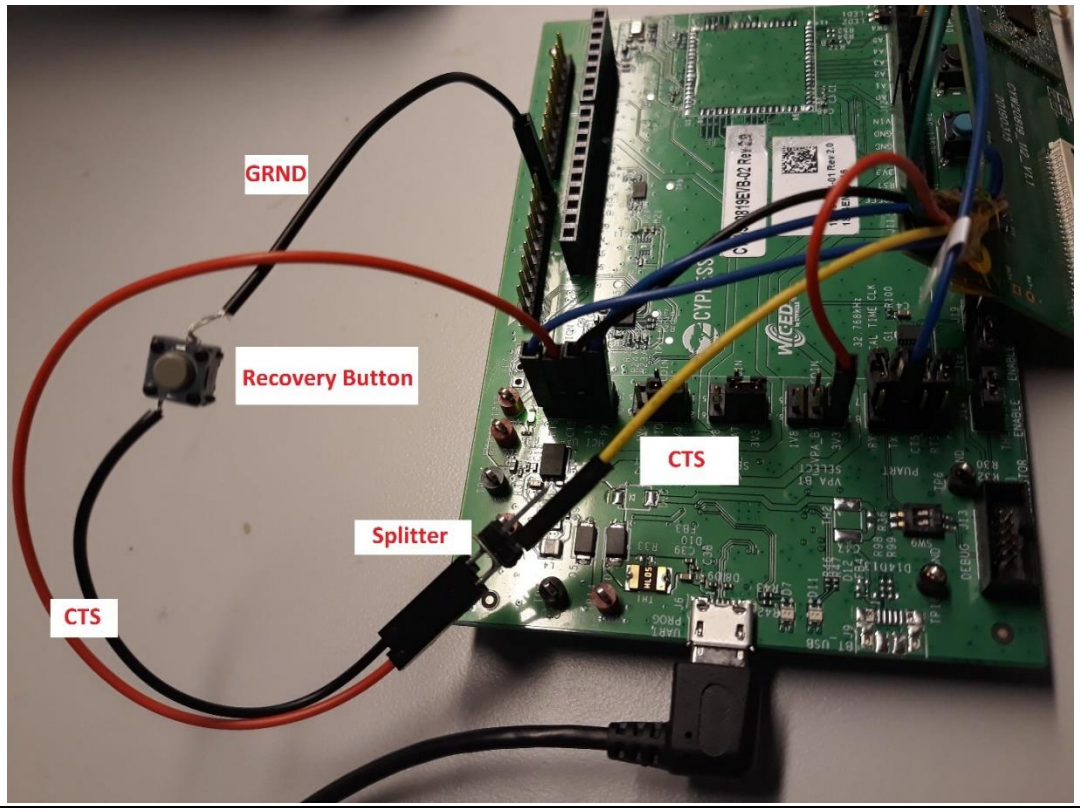
2.2 Reset and Recovery Buttons

Shorting the BT_RST (RST_N) pin in the module to ground will reset CYW20819. Put a push button in between.



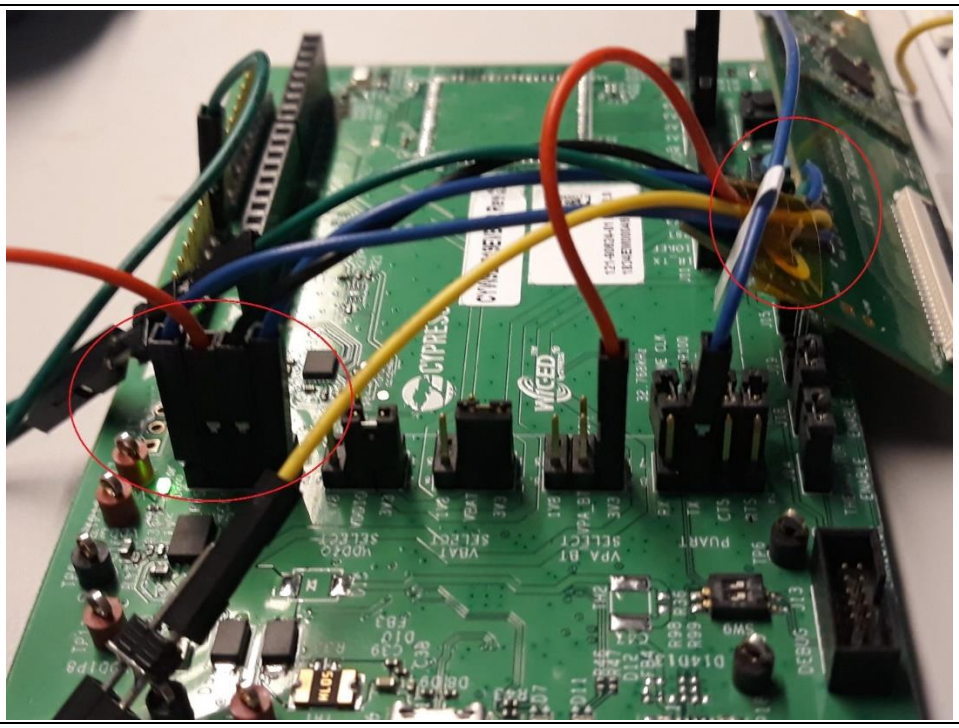


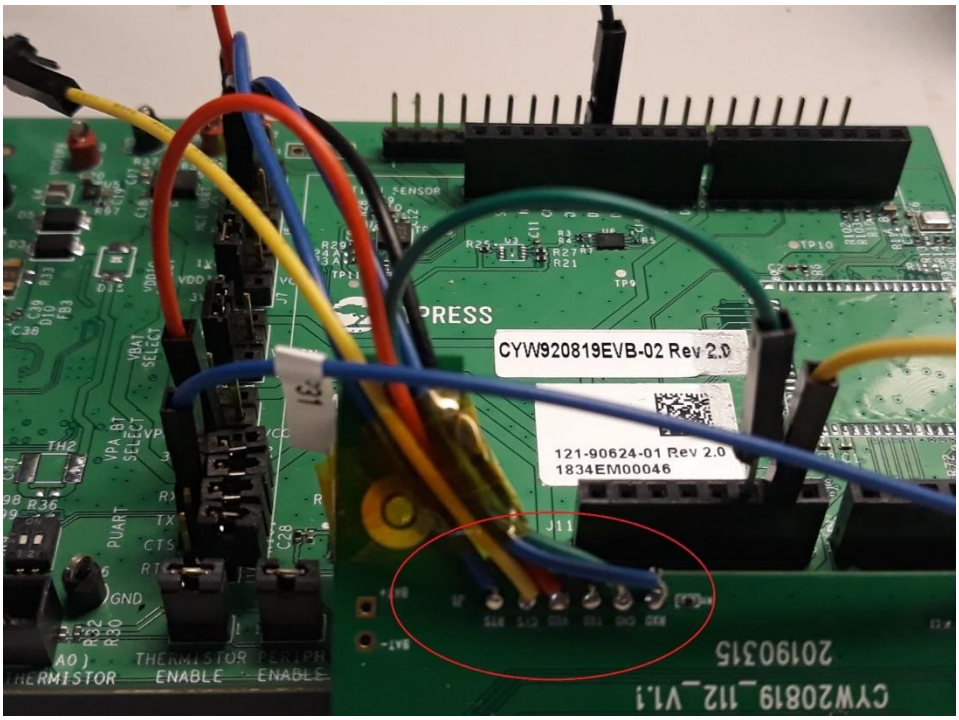
While shorting CTS to ground, resetting the device will allow the device to bypass Flash boot and enter Auto Baud Recovery mode. Because CTS needs to be connected to two places, use a splitter to connect it to a push button and EVB J5-P6.



2.3 Connecting HCI UART

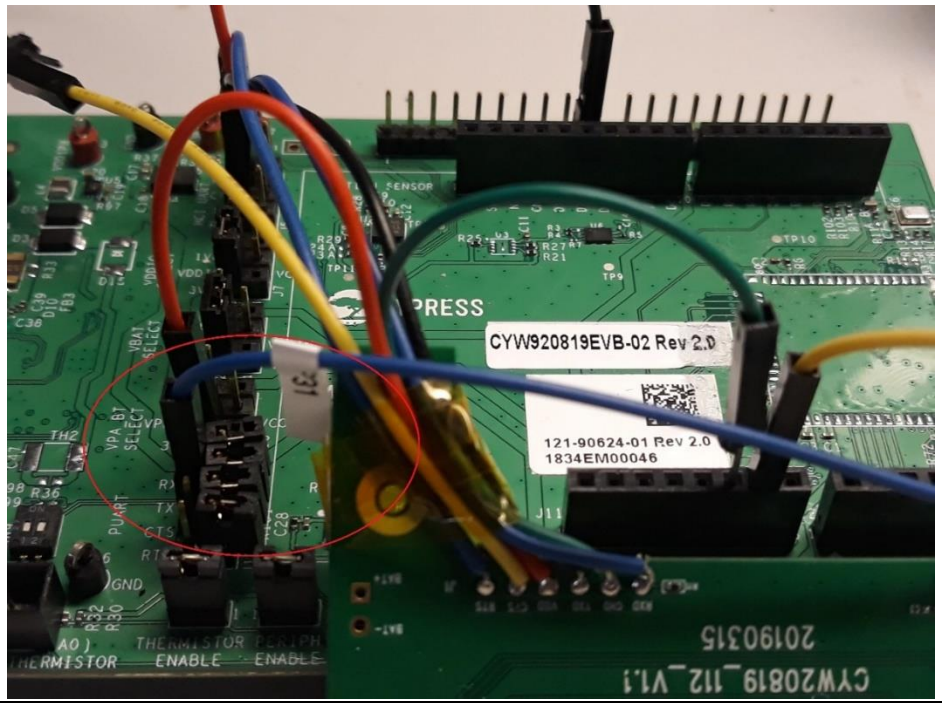
Remove all jumpers from EVB J5 and connect TX, RX, CTS, and RTS signals to the TX, RX, CTS, and RTS pins of the reference keyboard.

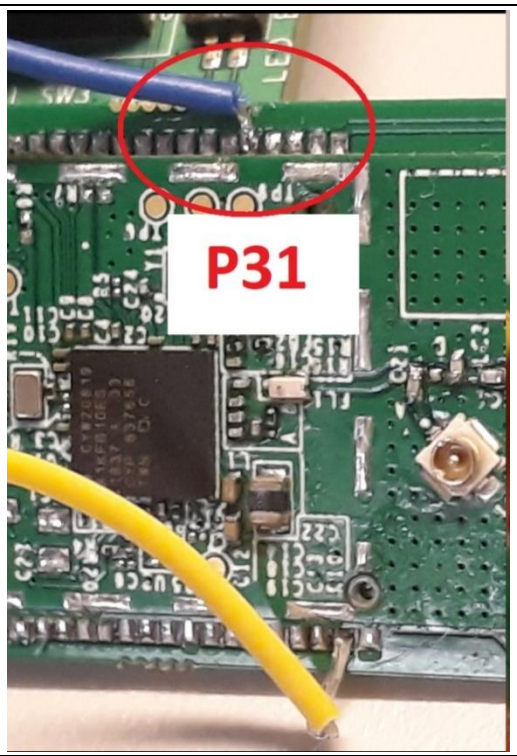




2.4 Connecting PUART

Remove all jumpers from EVB J10 and connect EVB P31 J10-P6 (TX).

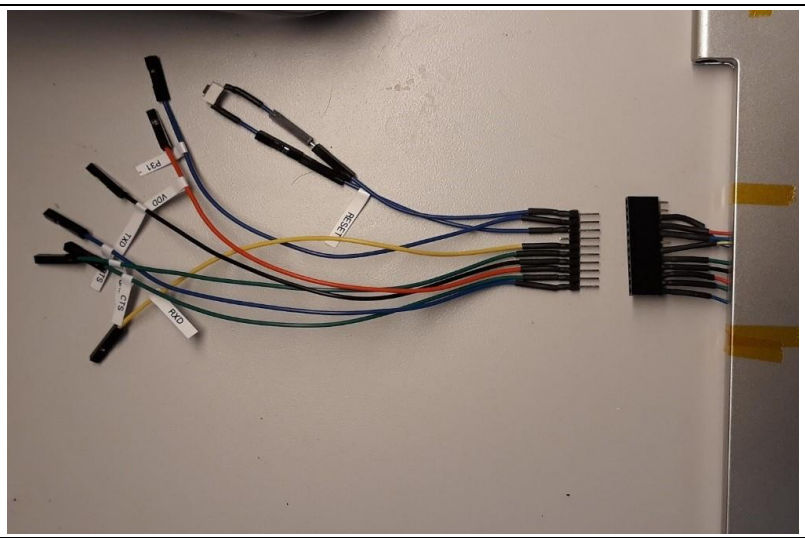


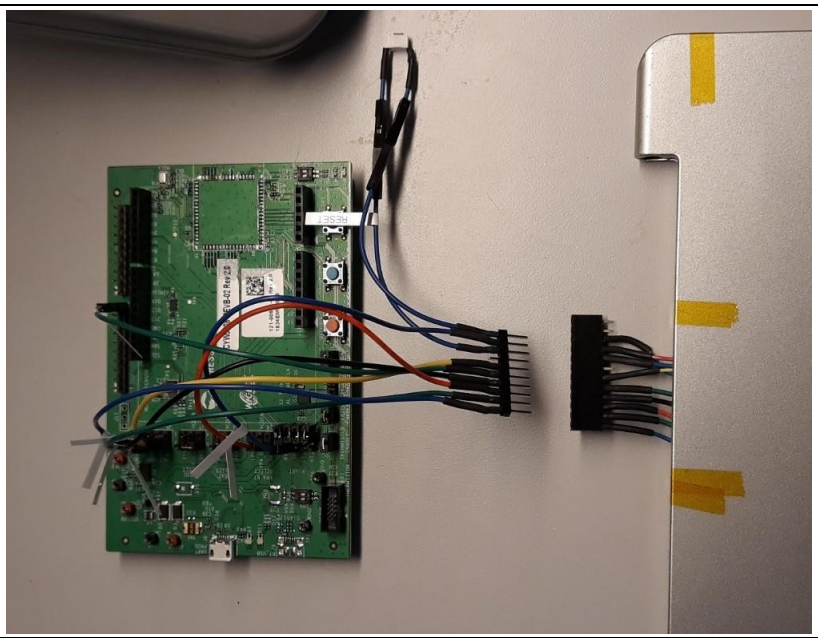


When the EVB board is connected to a USB port on a host PC, the system will enumerate two serial ports: one for HCI UART and one for PUART. The HCI UART port is used for programing from ModusToolbox. Use a serial port terminal application to open the PUART port so that the firmware debug output can be shown. Use 115200 Baud, 8 bits, no parity. Use 'LF' as the 'return' character for both TX and RX lines.

Hardware Setup

The wiring can be arranged as shown:





With this arrangement, after programming, the device can be detached easily and used as a stand-alone device.

3 Programming

3.1 Auto Baud Recovery Mode for Programming

When upgrading firmware through the HCI UART, the device must be put into Auto Baud Recovery Mode for programming. Do the following to put the device into this mode:

1. Press and hold the **Recovery** button.
2. While the **Recovery** button is held down, press and release the Reset button.
3. Release the **Recovery** button.

If it is done properly, the PUART port should not show any output after the **Reset** button is pressed in Step 2.

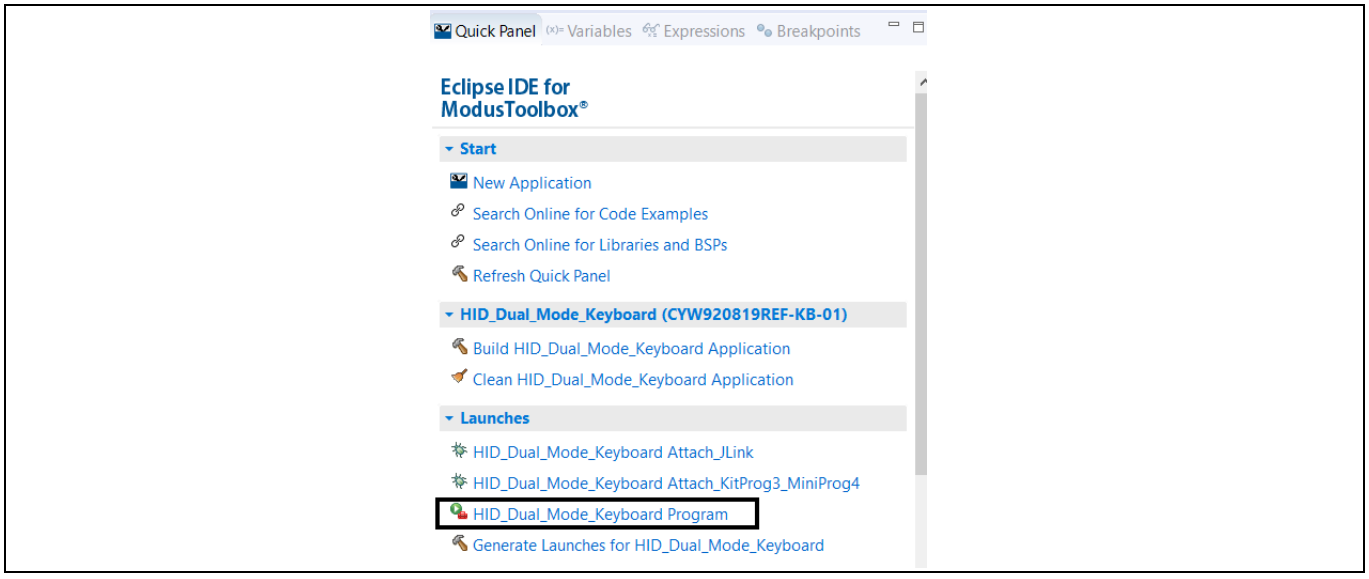
If the Recovery button is not available, do the following to put the device into Auto Baud Recovery mode.

1. Use the Client Control application supplied with BTSDK, or any other serial terminal application to open the HCI UART port.
   When the port is opened, it is equivalent to the Recovery button being pressed because the CTS line is driven low for hardware flow control.
2. Press the **Reset** button while the port is open, and then close the port to be ready for programming.
   This will be the same effect as the Recovery button steps above. Again, after the Reset button is pressed, if the device is in Auto Baud Recovery Mode, it should not have any output on the PUART port.

With this method, you must ensure that no character or command is sent to the port after reset because when sending any data, the CTS line can be toggled for flow control and the device will exit Auto Baud Recovery Mode.

3.2 Building and Downloading Firmware

1. In Eclipse IDE for ModusToolbox, click on the **New Application** link in the Quick Panel to create a new project.
2. Select the CYW920819REF-KB-01 platform and then select the HID Dual Mode Keyboard application. This installs the HID_Dual_Mode_Keyboard application.
3. Use the **Program Launch** link in the Quick Panel to build the application and program it to the board.



Programming

Alternatively, after the application has been created in the ModusToolbox IDE, the firmware can be built and downloaded using the command line. Do the following:

Open a command prompt (CMD, xterm, etc.) and change directory to the ModusToolbox workspace folder (the *mtw* folder in the user home directory by default), and execute the following commands:

```
$ cd HID_Dual_Mode_Keyboard
$ make clean
$ make program
```

After download, the application executes; the PUART output should show as follows:

```
<<CY DUAL MODE KB start>>
OTA_FW_UPGRADE
SKIP_PARAM_UPDATE
…
```

The application uses the following default values:

HCI_UART (TESTING_USING_HCI=0) transport is disabled

PUART=P31 (Baud 115200, 8 bits, no parity)

Device will sleep (power off for HIDOFF), need to use Recovery for reprogramming.

The key matrix is enabled.

3.3 TESTING_USING_HCI Option

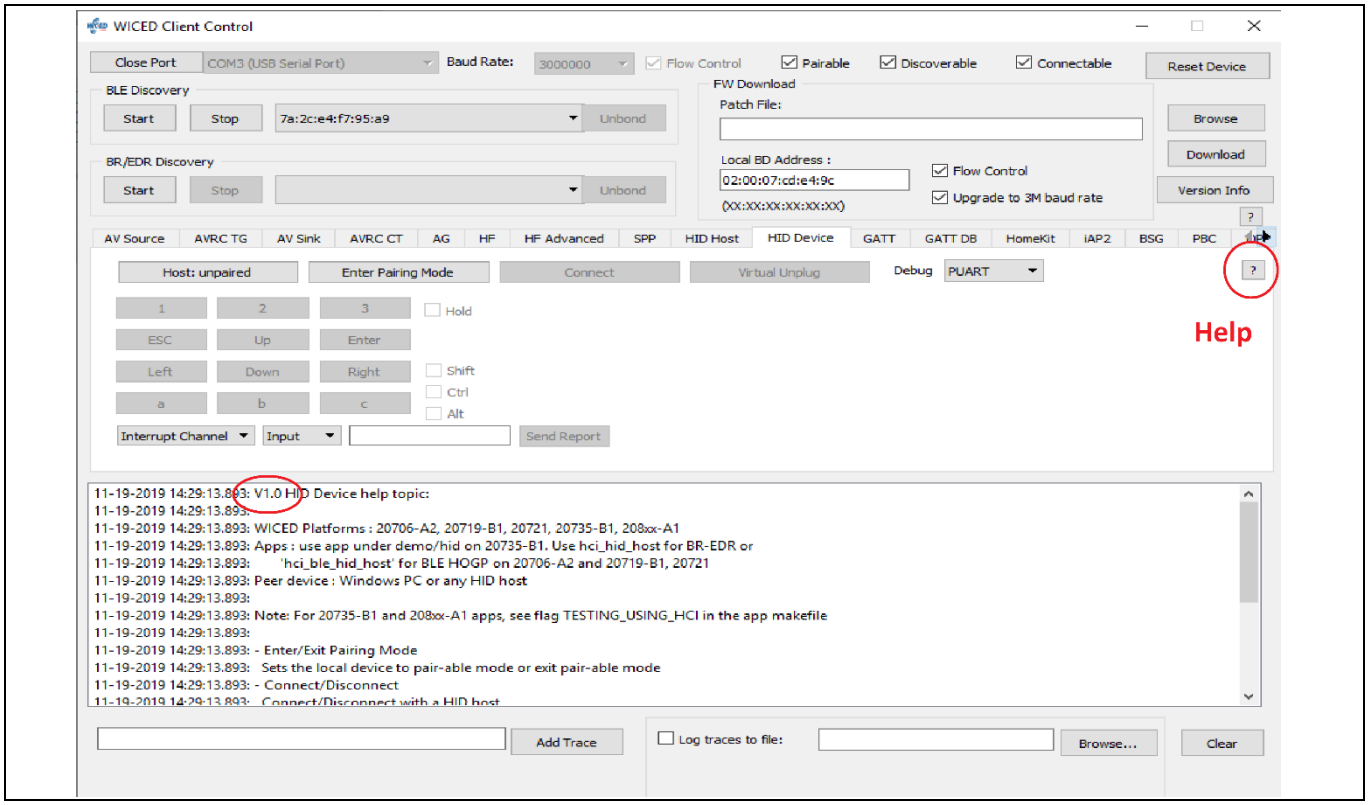
With the reference keyboard platform, HCI transport is disabled by default. To use BTSDK host utilities (Client Control or BtSpy), the TESTING_USING_HCI option must be set to '1'. This can be configured in the application makefile, or can be supplied as a command line override as follows:

```
$ make clean
$ make program TESTING_USING_HCI=1
```

PUART output:

```
<<CY DUAL MODE KB start>>
TESTING_USING_HCI
SLEEP_ALLOWED=2
OTA_FW_UPGRADE
SKIP_PARAM_UPDATE
…
```

Use ModusToolbox 2.0 with BTSDK 2.1 (or higher) to ensure the correct version of Client Control which supports BR/EDR HID. Click **Help**; the output should display a version of "V1.0 HID" or higher as shown in the following:



3.4 SLEEP_ALLOWED Option

There are four levels of Sleep options, which are configurable in the makefile or via command line override.

In makefile, change SLEEP_ALLOWED_DEFAULT=n

In Cygwin command line, use SLEEP_ALLOWED=n

Where n is:

0 No sleep allowed

1 Sleep is allowed without Deep Sleep:

Radio powered down and digital core is mostly powered down except for RAM, registers, and some core logic. The device can wake up either after a programmed period or upon receiving an external event. Processor is paused during Sleep and does not require boot upon wake up. The device consumes approximately 50 uA.

2 Deep Sleep (shutdown Sleep) is allowed of type ePDS (Extended Power Down Sleep):

Only the main RAM and ePDS control circuitry retains power. All other components are powered OFF. The device can wake up either after a programmed period or upon receiving an external event. The processor is paused during Sleep and does not require boot upon wake up. The device consumes about 8 uA floor current.

3 Deep Sleep (shutdown Sleep) is allowed of type HIDOFF:

The device is powered OFF with minimum wake up control circuit. The device can wake up either after a programmed period or upon receiving an external event. In this Deep Sleep mode, the firmware always requires full-boot when waking up from HIDOFF. The device consumes about 1 uA.

Programming

By default, the application is set to SLEEP_ALLOWED=2. If TESTING_USING_HCI is enabled and the HCI UART port is opened by Client Control, Deep Sleep will be disabled.

PUART output:

```
<<CY DUAL MODE KB start>>
SLEEP_ALLOWED=2
OTA_FW_UPGRADE
SKIP_PARAM_UPDATE
…
```

3.5 LED Options

The LED functionality is enabled by default. It can be disabled by changing the makefile to use LED_SUPPORT_DEFAULT=0, or via a command line override using the LED=0 parameter. The LED function should be turned off for power measurement.

There are four LEDs available on the keyboard reference board: WHITE, BLUE, YELLOW, and RED.

- WHITE: Used for the Caps Lock indicator. This LED is set or cleared by the host after a Bluetooth HID link is connected. When you press the **Caps Lock** key, the host receives the key state, and it changes the Caps Lock state at the host, which will send the report back to the reference board to set or clear this LED accordingly.
- BLUE: Used for LE link status. This LED blinks while in the pairing state, is solid on when the LE link is up, and off when the link is down.
- YELLOW: Used for BR/EDR link status. This LED blinks while in the pairing state, is solid on the BR/EDR link is up, and off when the link is down.
- RED: Error indicator. Not used.

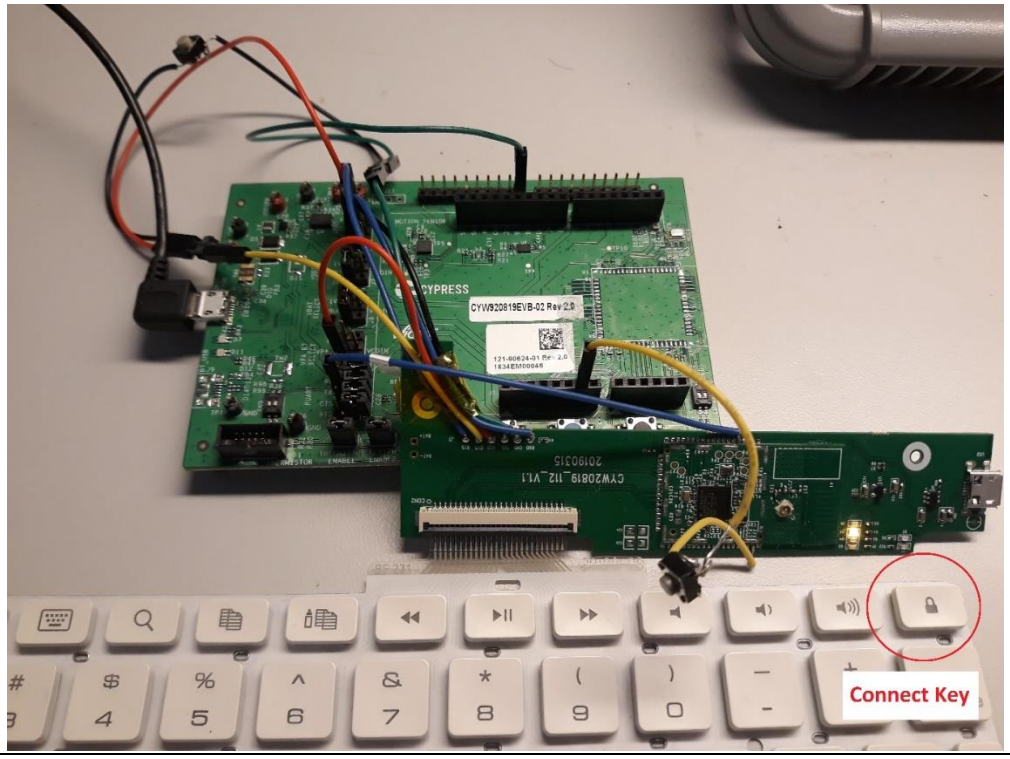
4 Testing the Reference Keyboard CYW920819REF-KB-01 Platform

Do the following:

1. Connect the CYW920819EVB-02 USB port to the host PC.
2. Connect a terminal emulation program to the PUART serial port to allow the application debug output to be shown.
3. Build and program the image to device.
4. Press the Reset button to reset the device. The Blue LED should blink five times to indicate that the firmware is running.

Because there is no “Connect” button designed for the reference keyboard to initiate Bluetooth connections, the right-most top ‘Lock’ key is used as the **Connect** button. Pressing this key once will start BR/EDR pairing. While the EVB is in BR/EDR pairing, pressing this key one more time will switch to the EVB to LE pairing. Pressing it one more time while the EVB is in LE pairing will stop the pairing process.

Pressing the **Connect** key will perform a ‘virtual cable unplug’ and erase any previously paired host information. Thus, in that state, pairing would need to be performed again to reconnect back to any previously paired host. When the pairing information is erased in the reference platform, the host also must ‘unpair’, ‘forget’, or otherwise remove the paired device from the host’s Bluetooth configuration so that it may be paired again to reconnect.



4.1 BR/EDR Link Test

1. Press the **Connect** key once for BR/EDR pairing mode. The YELLOW LED should blink.
2. Pair to a BR/EDR host.
3. After successful pairing, the YELLOW LED should glow to indicate that the link is connected.
4. From the host, open a text editor or notepad application, and then start typing on the reference keyboard to make sure that keystrokes are sent to the host.
5. Press the **Caps Lock** key several times to make sure that the WHITE LED is toggled to show the Caps Lock status. When it is locked, pressing any letter key should result in upper-case letters appearing at the host.
6. Disconnect the link from the host side (if the host is capable) and press a key from the reference keyboard to make sure that it can reconnect back to the host. After reconnecting, verify that the keystrokes can still be received by the host.
7. Press the **Reset** button to disconnect the keyboard. Press a key on the keyboard to reconnect. After reconnecting, verify that keystrokes can still be received by the host.
8. Power cycle the keyboard and press a key from the keyboard to reconnect. After reconnecting, verify that keystrokes can still be received by the host.

4.2 LE Link Test

1. Press the **Connect** button twice for LE pairing mode. The BLUE LED should blink.
2. Pair to an LE host. Note that it is better to use a different host than the one used for BR/EDR testing to avoid connection problems. This is because the keyboard has the same Bluetooth address for both BR/EDR and LE, the host may not be able to differentiate between the two Bluetooth links from the same device address.
3. After successful pairing, the BLUE LED should be solid to indicate that the link is connected.
4. Repeat steps 4 through 8 from the BR/EDR link test.

Revision history

Date	Version	Description
2019-12-09	**	Initial release
2020-02-26	*A	Updated SLEEP_ALLOWED Option.
2020-11-18	*B	Updated project creation information for ModusToolbox 2.2 + BTSDK 2.8.

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2020-11-19
Published by
Infineon Technologies AG
81726 Munich, Germany

Do you have a question about this document?
Go to www.cypress.com/support

Document reference
002-29203 Rev. *B

IMPORTANT NOTICE
The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics (“Beschaffenheitsgarantie”) .

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer’s compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer’s products and any use of the product of Infineon Technologies in customer’s applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer’s technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

WARNINGS
Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies’ products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.