



WICED Studio



WICED CYW20706 Kit Guide

Associated Part Family: CYW2070x

Doc. No.: 002-18191 Rev. **

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
www.cypress.com

Contents

1	Introduction.....	4
1.1	WICED Bluetooth Hardware	4
1.2	WICED Studio Software	4
1.3	IoT Resources and Technical Support	4
2	WICED Studio Development System Overview	5
2.1	WICED Evaluation Board	5
2.2	Software Development Kit	5
2.3	Directory Structure.....	6
2.4	Hardware and Software Requirements.....	6
2.5	Development Process	6
3	Setting up WICED Studio.....	7
3.1	Install WICED Studio	7
3.1.1	Windows	7
3.1.2	Linux	7
3.1.3	Mac OS X	7
3.2	Connect the WICED Evaluation Board	8
3.3	Verify Driver Installation.....	9
3.3.1	Windows	9
3.3.2	Linux	10
3.3.3	Mac OS X	10
4	Using the WICED Studio IDE	11
4.1	Build and Load a Sample Application	11
4.2	Hello Client Peer Application	13
4.3	Testing the Hello Sensor Application.....	14
4.3.1	Hello Sensor Application Structure	14
4.3.2	Hello Sensor Application Test Procedure	14
4.3.2.1	Hello Input Characteristic	14
4.3.2.2	Hello Configuration Characteristic	14
4.4	Viewing Application Trace Messages	15
4.4.1	Routing Trace Messages	15
4.4.2	View Traces Using a Terminal Emulation Program	15
4.4.3	View Traces Using the BTSpy Windows Application	15
4.5	What's Next?	16
Appendix A.	Eclipse IDE Hints & Tips	17
A.1	Hints	17
A.2	Shortcuts	17
Appendix B.	Multiple WICED Evaluation Boards.....	18
Appendix C.	Connecting to Linux Platforms.....	19
Appendix D.	Recovering a Corrupted Board.....	20
References.....		21
Document Revision History		22
Worldwide Sales and Design Support.....		23

Products	23
PSoC® Solutions.....	23
Cypress Developer Community	23
Technical Support.....	23

1 Introduction

This document provides detailed instructions to set up the Cypress Wireless Internet Connectivity for Embedded Devices (WICED; pronounced "wick-ed") CYW92070xV3_EVAL evaluation board for use with the Cypress WICED Studio Development System for Bluetooth Basic Rate (BR) and Low Energy (LE) devices.

1.1 WICED Bluetooth Hardware

The CYW92070xV3_EVAL board features the Cypress WICED Bluetooth CYW2070x device, which can be used as either a CYW20706 (fully embedded version) or as a CYW20707 (standalone BT controller version).

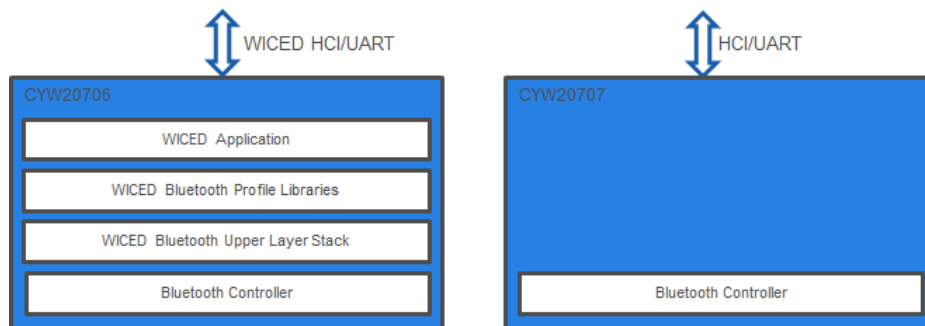


Figure 1-1. CYW20706 and CYW20707 Devices

CYW20706 contains the embedded BT stack and supports running user-developed applications using WICED Studio. CYW20707 operates as a standard BT controller and communicates with an external MCU via the standard HCI UART, but does not support user-developed applications.

For complete hardware information on the CYW92070xV3_EVAL board and CYW20706 and CYW20707 devices, see the *CYW92070xV3_EVAL Evaluation Board Hardware User Guide* [1].

1.2 WICED Studio Software

WICED Studio supports application development using a WICED evaluation board (CYW92070xV3_EVAL). The development system is compatible with Windows, Mac OS X, and Linux operating systems. This document describes the software components included in the WICED Studio Development System and provides instructions for compiling WICED sample applications using the WICED Studio Integrated Development Environment (IDE).

This document applies to **WICED Studio 5** and **WICED Bluetooth CYW20706 modules**. See [1] for information on using the CYW20707 device as a standard HCI BT Controller without WICED Studio.

Instructions in this document must be completed before the WICED evaluation board can be used with WICED Studio.

1.3 IoT Resources and Technical Support

Cypress provides a wealth of data at <http://www.cypress.com/internet-things-iot> to help you to select the right IoT device for your design, and quickly and effectively integrate the device into your design. Cypress provides customer access to a wide range of information, including technical documentation, schematic diagrams, product bill of materials, PCB layout information, and software updates. Customers can acquire technical documentation and software from the Cypress Support Community website (<http://community.cypress.com/>).

2 WICED Studio Development System Overview

The WICED Studio Development System comprises a Software Development Kit (SDK) along with the Eclipse Integrated Development Environment (IDE), enabling development on WICED evaluation boards.

2.1 WICED Evaluation Board

The Cypress WICED evaluation board CYW92070xV3_EVAL incorporates a Cypress CYW20706 device and additional circuitry to enable application programming, debugging, and evaluation.

The CYW92070xV3_EVAL board can be used for feature evaluation, debugging, and developing Bluetooth applications for designs based on the CYW20706 device.

Figure 2-1 shows the front of the CYW92070xV3_EVAL board.

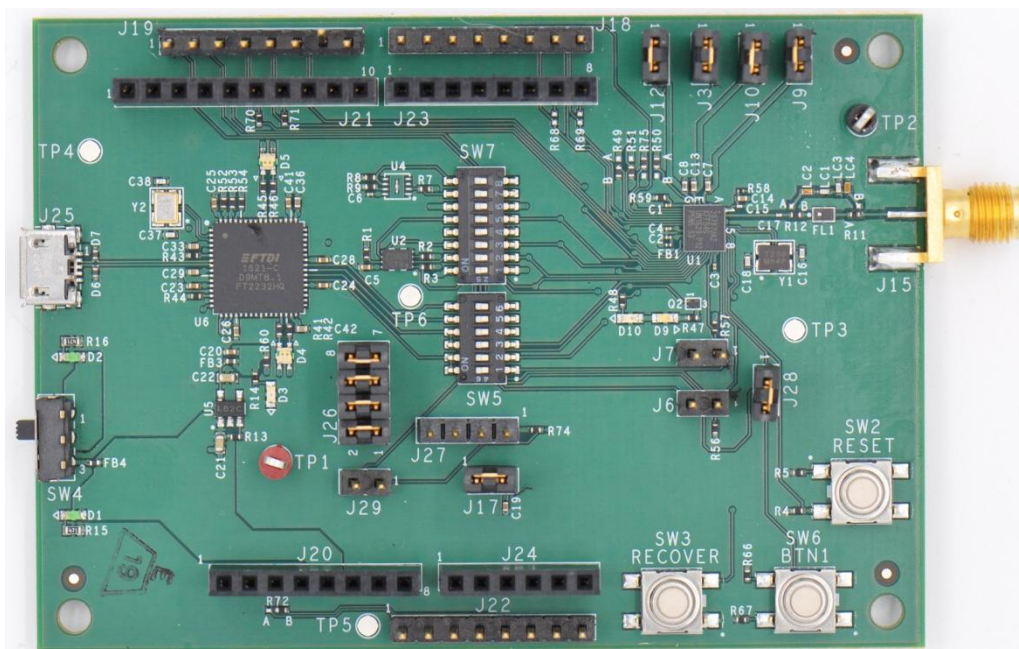


Figure 2-1. CYW92070xV3_EVAL WICED Evaluation Board

2.2 Software Development Kit

WICED Studio includes the following:

- The Bluetooth Basic Rate and Low Energy software stack
- A generic profile-level API (WICED BT API)
- Sample applications that demonstrate the use of the API
- Drivers to access on-chip peripherals (for example, UART, SPI, I²C, ADC, PWM)
- Reference applications for the devices with profiles defined by the Bluetooth SIG
- WICED BT API documentation
- Utilities to support development, testing, and mass production on Windows, Mac OS X, and Linux environments

2.3 Directory Structure

Table 2-1 provides an overview of the directory structure of WICED Studio for the CYW20706 device. WICED Studio may support multiple types of WICED modules depending on installed components; some modules may share components, files, and folders. For example, the ARM_GNU toolchain resides under the 43xxx_Wi-Fi folder structure but is used for the CYW20706 device.

Note: The folder structure presented in the WICED Studio Project Explorer UI window may differ slightly from the directory structure of the files installed on the file system, as some components may be shared between multiple components in common folders on the file system. It is recommended that WICED Studio be used for accessing files and folders rather than directly through the file system. In case of any discrepancies between the structure presented in WICED Studio and the file system, right-click the component in WICED Studio and select Properties to show the location of the component in the file system.

WICED Studio Directory	Directory Contents
Doc	Reference documentation
Doc\20706-A2_Bluetooth	CYW20706 WICED API documentation
Drivers	USB drivers for the evaluation board
wiced_tools	Tools including programming tool, and other utilities and scripts
43xxx_Wi-Fi\tools\ARM_GNU	Toolchain including compiler, linker, and supporting files (libraries and headers)
20706-A2_Bluetooth\apps	Sample applications
20706-A2_Bluetooth\build	Output files of built applications
20706-A2_Bluetooth\include	WICED API function prototypes and definitions
20706-A2_Bluetooth\libraries	Sources for various WICED interface libraries
20706-A2_Bluetooth\platforms	Configuration files and information for supported hardware platforms
20706-A2_Bluetooth\tools	Common utilities used by the IDE build processes
20706-A2_Bluetooth\WICED	WICED core components

Table 2-1. WICED Studio Directory Structure

2.4 Hardware and Software Requirements

WICED Studio runs on 32- and 64-bit versions of Microsoft Windows, Mac OS X, and Linux

The development computer requires a single USB port to connect to the WICED evaluation board

2.5 Development Process

WICED Studio is distributed as executable installers for Windows, Mac OS X, and Linux. To prepare and run an application, do the following:

1. Download and install WICED Studio 4 (see [Install WICED Studio](#)).
2. Connect the WICED evaluation board (see [Connect the WICED Evaluation Board](#)).
3. Create and load an application (see [Build and Load a Sample Application](#)).

3 Setting up WICED Studio

Download WICED Studio from the [Cypress WICED Products website](#) or [Cypress Customer Support Portal](#).

The WICED Studio distribution is provided as a self-installing executable file inside a zip file. Extract the file to a folder on the local hard drive; do not execute the installer from the zip file. Some customized distributions may also provide a configuration file called "config.eml", which should be placed in the same folder.

3.1 Install WICED Studio

Note: "x.x.x.x" used in the filenames denotes the WICED Studio version numbers in actual files.

3.1.1 Windows

1. Unzip the distribution to a local folder, along with the *config.eml* file, if present.
2. Double-click the *WICED-Studio-x.x.x.x-IDE-Installer.exe* file.
3. Follow the prompts to override or accept the default folders for the Eclipse IDE and WICED Studio SDK files.

3.1.2 Linux

1. Unzip the distribution to a local folder, along with the *config.eml* file, if present.
2. Open a terminal window and "cd" to the same folder.
3. Apply 'execute' permissions to the installer executable with the following command:

```
chmod +x ./WICED-Studio-x.x.x.x-IDE-Installer.bin
```

4. Launch the installer from the same terminal window:

```
./WICED-Studio-x.x.x.x-IDE-Installer.bin
```

5. Follow the prompts to override or accept the default folders for the Eclipse IDE and WICED Studio SDK files.

If there is a conflict between a pre-existing Java version installed on the system and the Java JRE supplied with the WICED Studio installer, the following error may be encountered during the Linux installation process:

"Installer User Interface Mode Not Supported"

To resolve, use the package manager for your Linux distribution (e.g., dnf, apt-get, yum) to update Java to the latest version. For example:

```
sudo dnf install java
```

3.1.3 Mac OS X

1. Unzip the distribution to a local folder.
2. If there is a config.eml file with the distribution, use Finder to copy it and the *WICED-Studio-x.x.x.x-IDE-Installer* app folder to another folder. This is needed as a workaround for a known OSX 10.12 install issue.
3. Double-click the *WICED-Studio-x.x.x.x-IDE-Installer* app.
4. Follow the prompts to override or accept the default folders for the Eclipse IDE and WICED Studio SDK files.

If the installer fails to execute, you may need to install or update Java to resolve any potential conflict between a pre-existing Java version installed on the system and the Java JRE supplied with the WICED Studio installer:

Open xterm, run the `java -version` command. If it fails to return any results or states that you are running version 1.6, then you need to install the Java SE Development Kit 8 (JDK8), which can be found here: <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>. Once JDK8 is installed, run the `java -version` command again with xterm. If this command returns "1.6", then you will need to fix the symbolic link using the following commands:

```
rm -f /usr/bin/java
```

```
ln -s /System/Library/Frameworks/JavaVM.framework/Versions/Current/Commands/java  
/usr/bin/java
```


3.2 Connect the WICED Evaluation Board

Figure 3-1 shows the CYW92070xV3_EVAL WICED evaluation board.

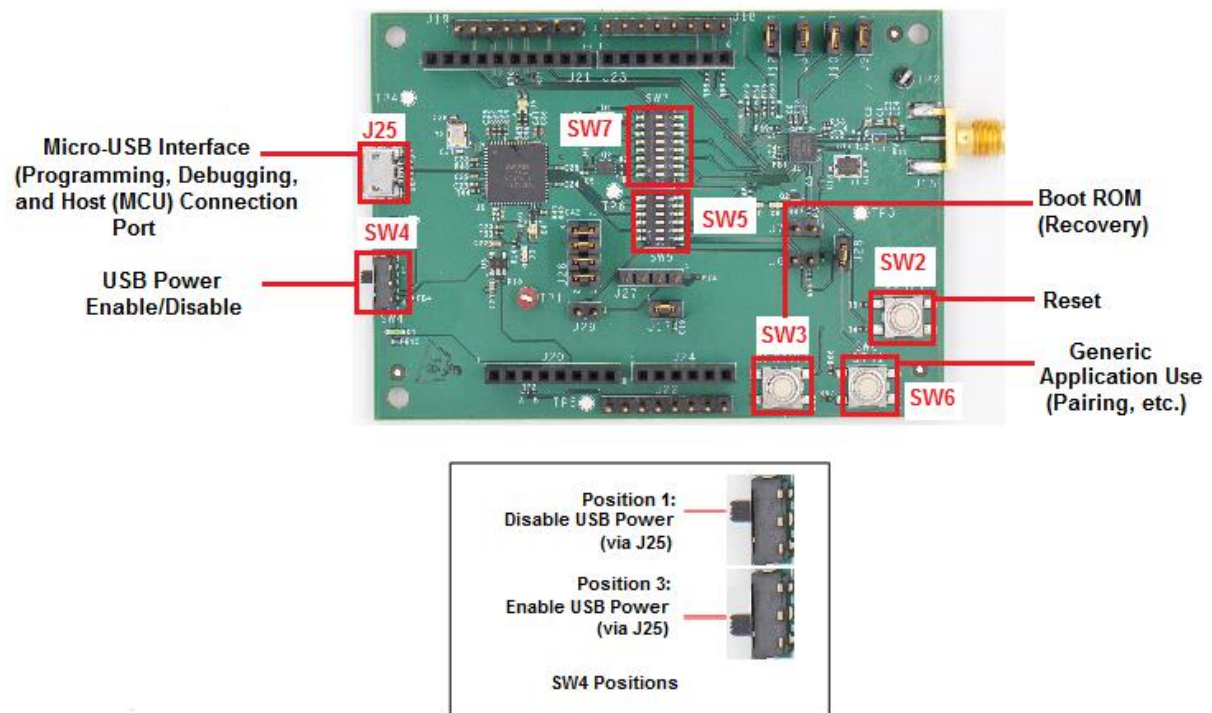


Figure 3-1. CYW92070xV3_EVAL Evaluation Board Settings

The Micro-USB connector (J25) supports UART connections and provides +5 V power to the board if SW4 is in the ON position.

Perform the following steps before connecting the board and verifying the driver installation. See [1] for complete information on DIP switch and jumper settings.

1. Verify that **SW5** DIP switches **1, 3, 5, and 6** are set to the OFF position, and **SW5** switches **2 and 4** are set to the ON position so that the Peripheral UART is selected and able to display embedded application trace messages.
2. Verify that **SW7** DIP switches **1-5** are set to the ON position to enable on-board serial flash memory. **SW7** switches **6-8** are set to the OFF position by default; these may be set to the ON position if an Authentication IC is to be used.
3. Verify that jumpers are in the default positions: **J3, J9, J10, J12, J17, J26, and J28** shorted, and **J29** open.
4. Set **SW4** to the ON position to enable USB power (see Figure 3-1).
5. Connect J25 of the WICED evaluation board to the development PC with a USB cable. The USB UART driver will load automatically.

The LEDs called out in Figure 3-2 serve the following purposes:

- D1 (green) indicates that SW4 is in position 3 (that is, USB power is enabled).
- D2 (green) indicates that an external PC is connected to the Micro-USB interface (J25).
- D3 (green) indicates that 3.3 V power is ON.
- D4 (variable color, but orange during a download) indicates HCI UART activity.
- D5 (variable color) indicates peripheral UART activity.
- D9 (blue) and D10 (red) are generic LEDs controlled by GPIOs.

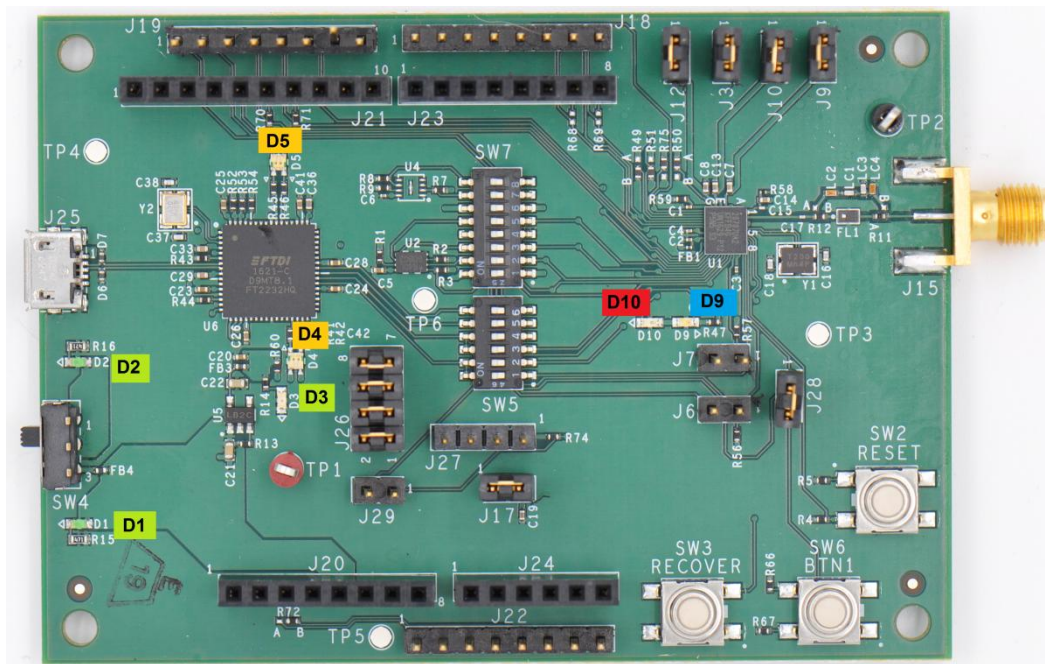


Figure 3-2. LED Indicators

3.3 Verify Driver Installation

3.3.1 Windows

1. Open **Device Manager** (right-click **My Computer**, select **Properties**, and then select **Device Manager**).
2. In the **Device Manager** window, verify that two new USB serial COM ports are listed under **Ports (COM & LPT)**.

Note: In [Figure 3-3](#), Device Manager identifies the new WICED evaluation board USB serial COM ports as 'WICED HCI UART' and 'WICED Peripheral UART'. Assigned port numbers vary among systems.

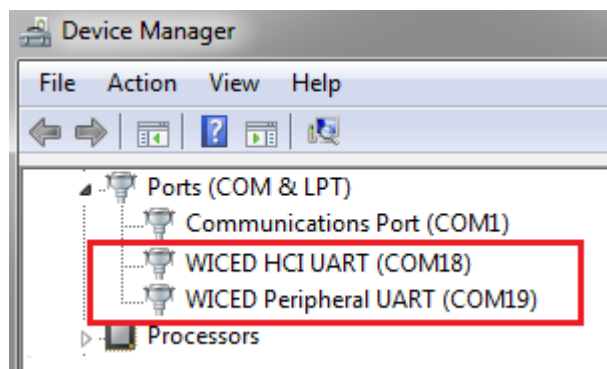


Figure 3-3. Device Manager COM Ports

Note: If an error occurs during driver installation, download new drivers from Windows Update. Verify that you have an Internet connection, disconnect and reconnect the board, and wait for the drivers to automatically install.

If the new WICED evaluation board COM ports do not appear in Device Manager after reinstalling drivers via Windows Update, then the drivers can be manually installed from the 'Drivers' folder of the WICED Studio installation.

If the error persists, check all switch settings (see [Connect the WICED Evaluation Board](#)) on the board and replace the USB cable.

3.3.2 Linux

- Open a terminal window and verify that two UART ports are listed, usually `/dev/ttyWICED_HCI_UART0` and `/dev/ttyWICED_PUART1`, although numbers may vary.

Note: An additional step may be required when connecting a WICED board to a computer running Linux; see [Connecting to Linux Platforms](#).

3.3.3 Mac OS X

- Open an xterm window and verify that two UART ports are listed, usually `/dev/tty.usbserial-144A` and `/dev/tty.usbserial-144B`, although numbers may vary. The lower-numbered port is WICED HCI UART, the higher-numbered port is WICED Peripheral UART.

Note: On OS X versions 10.10 or earlier, the Apple version of the FTDI driver that ships with the OS has known issues and must be updated to a specific version by following the instructions here:

<https://learn.sparkfun.com/tutorials/how-to-install-ftdi-drivers/mac>

On OS X versions 10.11 and later, the Apple version of the FTDI driver must be used. If there was a previous instance of the FTDI version of the driver, it must be removed using the following commands in an xterm window:

```
sudo rm -rf /Library/Extensions/FTDIUSBSerialDriver.kext
```

```
sudo rm -rf /System/Library/Extensions/FTDIUSBSerialDriver.kext
```

Reboot the system after performing the rm commands.

4 Using the WICED Studio IDE

This section describes how to:

- Use the WICED Studio IDE to create application build targets for the WICED evaluation board.
- Download applications to the board.
- Verify that the application running on the board is working correctly using a Windows 10 PC with Bluetooth capability.

4.1 Build and Load a Sample Application

Start the IDE by double-clicking the **WICED-Studio** icon on the desktop. Some WICED Studio packages may offer support for multiple devices, and on first-time execution of the IDE, a dialog screen may be seen prompting to select the default platform. If shown, select “20706-A2_Bluetooth”. The default platform may always be changed later from the drop down menu selector control in the IDE.

The WICED Studio IDE looks similar to the screenshot shown in [Figure 4-1](#).

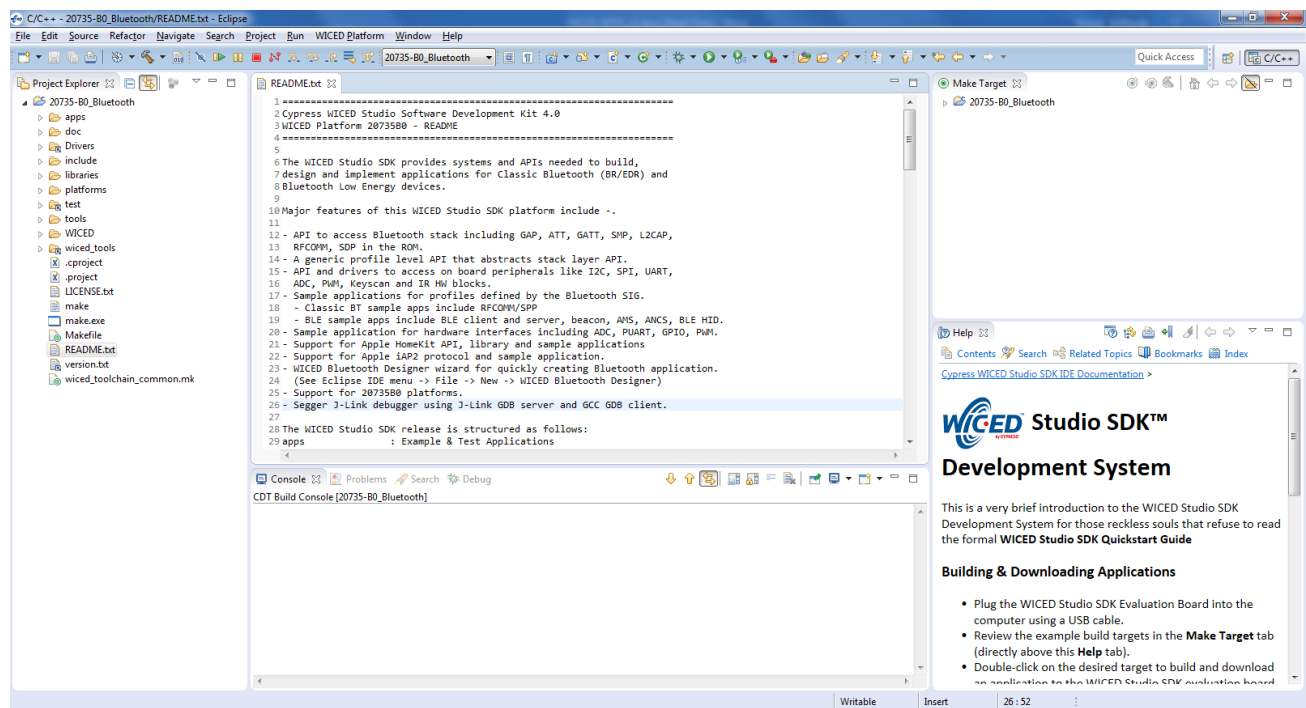


Figure 4-1. WICED Studio IDE

The **Help** pane in the lower-right corner of the IDE (see [Figure 4-1](#)) describes how to build and download the sample applications shown in the **Make Target** pane, which is located above the Help pane, and describes how to create new applications and associated make targets based on the samples. The Make Target pane contains make targets that are preconfigured for sample applications that run on CYW92070xV3_EVAL evaluation boards.

The example below shows how to build and run the "hello_sensor" sample application on the CYW20706, which can then demonstrate basic LE sensor functionality by connecting to a Windows 10 host PC or Android or iOS device running the Hello Client sample host application, provided as part of WICED Studio under peer_apps in the application folder (See [The Hello Client Peer Application](#)). The source code for the hello_sensor sample and "Hello Client" host peer application can be found in WICED Studio in the Project Explorer window under 20706-A2_Bluetooth\Apps\hello_sensor.

1. Connect the evaluation board to the PC.
2. Verify that the UART port is present after the WICED evaluation board is connected to the PC (see [Verify Driver Installation](#))

3. Double-click the **hello_sensor-BCM920706_P49 download** target start the build. The IDE console pane (bottom center of the IDE window) will display the build and download progress.

The build output looks similar to the following:

```
22:24:45 **** Build of configuration Release for project 20706-A2_Bluetooth ****

"C:\Users\username\Documents\WICED\WICED-Studio-4.0\20706-A2_Bluetooth\make.exe" hello_sensor-BCM920706_P49 download

Compiling spar_setup.c
Compiling hello_sensor.c
Compiling wiced_bt_cfg.c
Compiling lib_installer.c
Linking target ELF
OK, made elf.

..\..\..\43xxx_Wi-Fi\tools\ARM_GNU\bin\Win32\arm-none-eabi-objdump: section '.aon'
mentioned in a -j option, but not found in any input file

Call to hello_sensor_spar_crt_setup @ 0021844d

OK, made C:/Users/username/Documents/WICED/WICED-Studio-4.0/20706-A2_Bluetooth/Wiced-BT/spar/../../build/hello_sensor-BCM920706_P49-rom-ram-Wiced-release/A_20703A2-hello_sensor-rom-ram-spar.cgs. MD5 sum is:

806c5323f8b67625baf701f62d71e7c9 *../../build/hello_sensor-BCM920706_P49-rom-ram-Wiced-release/A_20703A2-hello_sensor-rom-ram-spar.cgs

-----

Patch code starts at          0x00270400 (RAM address)
Patch code ends at            0x00271C2C (RAM address)
Patch RW/ZI size              2140 bytes
Application starts at         0x002173FC (RAM address)
Application ends at           0x00218449 (RAM address)

Patch code size                6188 bytes
Application RAM footprint      4173 bytes
-----

Total RAM footprint            6313 bytes (6.2kiB)
-----

Converting CGS to HEX...
Conversion complete

Creating OTA images...
Conversion complete
OTA image footprint in NV is 21935 bytes

Detecting device...
Device found

Downloading application...
Download complete

Application running

09:11:40 Build Finished (took 1m:46s.379ms)
```

Note: The warning ‘section ‘.aon’ mentioned in a -j option, but not found in any input file’ above is not critical; this is only an indication that the application did not use any data in retention RAM.

Note: If the download fails, see [Appendix D: Recovering a Corrupted Board](#) for instructions on how to reset the board to boot from ROM in autobaud mode.

Note: Because the sample target includes the **download** option, the tool will download the firmware to the evaluation board automatically when the build is complete, where it is stored in the serial flash. The board will reset after the download is successful. When the board boots and finds a valid application image in serial flash, it will execute the image.

4.2 Hello Client Peer Application

The Hello Client (hello_client) peer sample application is provided with WICED Studio to complement the Hello Sensor (hello_sensor) application running on the CYW20706 device. It is located in the WICED Studio Project Explorer here::

20706-A2_Bluetooth\Apps\hello_sensor\peer_apps

The application is provided as full source code for Windows, iOS, and Android, along with an executable binary that runs on 32-bit and 64-bit Windows 10 machines. The application is also compatible with Windows versions 8 and 8.1, but not lower versions. This section demonstrates the use of the Windows version of the Hello Client application, but the process is similar for all host operating systems.

1. On the Windows host PC, open the **Settings** screen (usually from the **Start** button) and select **Devices** and then **Bluetooth**.
2. Click **Add a device** and wait while the PC searches for devices in range.
3. Select the **Hello Sensor** device (this is the embedded application running on the CYW20706).
4. Click **Pair**, and wait for the device connection to complete.
5. Run *HelloClient.exe* on the Windows host PC.

If multiple devices have been paired, a Hello Client Select Device window similar to that shown in [Figure 4-2](#) is displayed to select the correct device.

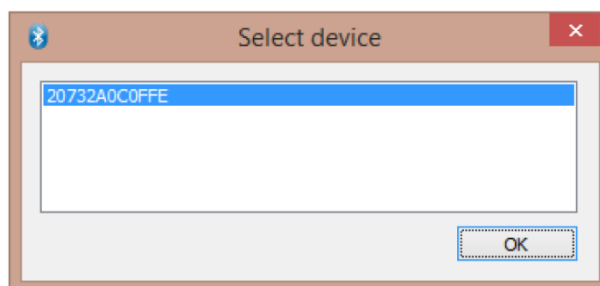


Figure 4-2. Hello Client Select Device Window

The window shows a list of Bluetooth device addresses for any Hello Sensor applications running on evaluation boards that have been paired to the PC.

6. Select the correct device (if it is not already selected) and click **OK** to initiate a connection to the evaluation board. The connection process may take 5–10 seconds.

If only one device has been paired, the selection screen is skipped.

Note: The Bluetooth device address programmed in the evaluation board is located in WICED Studio here:

20702-A2_Bluetooth\platforms\BCM920706_P49\20706_SFLASH.btp

To change the Bluetooth device address, open the file and modify the **DLConfigBD_ADDRBase** variable. This value can alternately be overwritten in the make target of the application by setting **BT_DEVICE_ADDRESS=xxxxxxxxxxx** where xxxxxxxxxxxx is a 12-digit hexadecimal value representing the Bluetooth device address. Digits may be substituted with asterisk characters, in which case WICED Studio will generate random hexadecimal values for the asterisks, using the host PC MAC address so that the random digits will always be the same from a single PC, but different for other PCs.

4.3 Testing the Hello Sensor Application

4.3.1 Hello Sensor Application Structure

The Hello Sensor application provides paired devices with the following information:

- A Hello Service (a proprietary service) with two proprietary characteristics:
 - The value of the Hello Input characteristic (read-only) may be retrieved using one of the following methods:
 - Manually, by using a mouse to click the **Read** button on the HelloClient PC application.
 - Automatically, by pressing the **Application** button **SW6** (see [Figure 3-1](#)) on the WICED evaluation board (the **Allow Notifications** drop-down must be selected to allow automatic notifications).
 - The Hello Configuration read-write characteristic is used to configure the number of times an LED on the evaluation board blinks (see [Figure 3-2](#)) when the **Application** button is pressed.
- A Device Information Service that provides information including:
 - Manufacturer Name
 - Model Number
 - System ID
- A Battery Service that provides a battery-level indication

4.3.2 Hello Sensor Application Test Procedure

To test the application with a WICED evaluation board, follow the instructions provided for each of the Hello Service characteristics (see [Figure 4-3](#)).

4.3.2.1 Hello Input Characteristic

1. Select **Allow Notifications** in the combo box.
2. Press the **Application** button **SW6** on the WICED board (see [Figure 3-1](#)). The Hello X message is displayed in the **Value** field. Each time the button is pressed, the message number increments.

4.3.2.2 Hello Configuration Characteristic

1. Change the **Value** field for the Hello Configuration to '5' and then click **Write**.
2. Press the **Application** button **SW6** on the WICED board (see [Figure 3-1](#)). The LED on the board blinks five times.

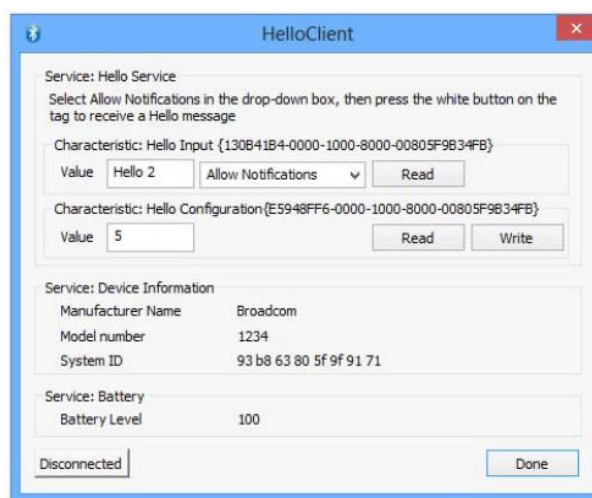


Figure 4-3. Hello Client Services Window

4.4 Viewing Application Trace Messages

Trace messages from the application can be viewed if the application is compiled with the `BT_TRACE_ENABLE` compile flag defined. Compile flags can be defined in the application makefile. For example, the `BT_TRACE_ENABLE` flag is enabled by default (see *makefile.mk* in the hello sensor application source folder in WICED Studio).

With the `BT_TRACE_ENABLE` compile flag defined, the `WICED_BT_TRACE()` macro can then be used to generate printf-style messages from the application code.

4.4.1 Routing Trace Messages

Trace messages can be routed to any of the following destinations using the `wiced_set_debug_uart()` API with the appropriate parameter:

- HCI UART serial port – `wiced_set_debug_uart(WICED_ROUTE_DEBUG_TO_HCI_UART)`
- PUART serial port – `wiced_set_debug_uart(WICED_ROUTE_DEBUG_TO_PUART)`
- BTSPy Windows application – `wiced_set_debug_uart(WICED_ROUTE_DEBUG_TO_WICED_UART)`

The BTSPy option also allows Bluetooth protocol trace messages to be viewed alongside application traces. See [View Traces Using the BTSPy Windows Application](#) for information on viewing traces with the BTSPy option.

The quickest and simplest way to view application traces is to use the HCI UART or PUART options with a terminal emulation program connected to the appropriate UART port. See [View Traces Using a Terminal Emulation Program](#) for information on terminal settings for the UART route options.

Note: If using the HCI UART serial port, note that downloads from WICED Studio will also use the same port, so do not open the HCI UART port with the terminal emulation program until after the application download has completed in WICED Studio. The PUART serial port has no such inherent contentions, so the Hello_Sensor sample application routes to PUART by default.

4.4.2 View Traces Using a Terminal Emulation Program

To view traces with a terminal emulation program (such as Tera Term on Windows, PuTTY on Linux, or SerialTools on Mac OS X), start the terminal emulation program and open a connection to the desired UART port (either HCI or Peripheral, as specified in the call to the `wiced_set_debug_uart()` API) using the following settings:

Baud Rate: 115200*

Data: 8 bit

Parity: None

Stop: 1 bit

Flow Control: None

Terminal ID: VT100

New-Line Receive: Auto

Press the **Reset** button (see [Figure 3-1](#)) on the WICED evaluation board to view the application startup messages.

***Note:** The Hello Sensor application uses 115200 as the default baud rate. Other applications in WICED Studio may use different baud rates, which may be found in each sample application's initialization of the `wiced_transport_cfg_t` structure, used in the call to the `wiced_transport_init()` API.

4.4.3 View Traces Using the BTSPy Windows Application

By routing application traces with the `wiced_set_debug_uart(WICED_ROUTE_DEBUG_TO_WICED_UART)` API call, traces can be viewed using the separate BTSPy and ClientControl utilities. These Windows applications are supplied with WICED Studio under the `wiced_tools` and `Apps\client_control` folders respectively.

The ClientControl utility connects to the HCI UART port and receives trace message packets. These message packets are then sent through a socket connection to the BTSPy utility, where they can be displayed and logged.

The BTSPy trace option provides an advantage over the UART options: applications may also configure the stack to generate encoded HCI trace message packets showing all activity between the stack and the controller over the virtual HCI interface embedded in the CYW2070x device, which can be decoded and displayed by the BTSPy utility. Application trace messages may then be viewed side by side with Bluetooth protocol trace messages.

To configure the stack to generate HCI trace messages, applications must define a callback function and register it with the stack using the `wiced_bt_dev_register_hci_trace()` API. The callback function implementation should call the `wiced_transport_send_hci_trace()` API with the data received in the callback. See the Hello Sensor sample implementation of `hello_sensor_hci_trace_cback()` in the `hello_sensor.c` file, controlled by the `ENABLE_HCI_TRACE` compile flag.

To route application trace messages to BTSPy, along with stack traces, the `ENABLE_HCI_TRACE` flag must be defined in the application makefile, and the application source code must call the `wiced_set_debug_uart()` API to route traces to `WICED_ROUTE_DEBUG_TO_WICED_UART`. As noted, the hello sensor application routes traces to PUART by default, and therefore the `ENABLE_HCI_TRACE` flag is not defined by default in the hello sensor makefile.

To view traces in BTSPy:

1. Define `ENABLE_HCI_TRACE` in the application makefile.
2. Ensure that the application calls `wiced_set_debug_uart(WICED_ROUTE_DEBUG_TO_WICED_UART)`.
Note: hello sensor calls this API in `hello_sensor.c`, routing to PUART by default.
3. Compile and download the application to the evaluation board.
4. Run the ClientControl utility.
5. In the ClientControl utility, set the baud rate to the baud rate used by the application (115200 for Hello Sensor; for other applications, see the initialization of the `wiced_transport_cfg_t` structure used in the call to the `wiced_transport_init()` API).
6. Run the BTSPy utility.
7. In the ClientControl utility, open the HCI UART COM port.
8. Press the **Reset** button (see [Figure 3-1](#)) on the WICED evaluation board to view trace messages in the BTSPy window.

4.5 What's Next?

Now that you have a basic understanding of how to compile and download a WICED Studio application, we recommend building and running the example applications provided in the WICED Studio applications directory.

The header of the main source file of every application provides additional information on the features demonstrated by the application and the usage model.

[“IDE Hints & Tips”](#) contains hints and tips about navigating the WICED Studio code base.

We hope you enjoy using the WICED Studio Development System!

-- The WICED Development Team

Appendix A. Eclipse IDE Hints & Tips

A.1 Hints

1. The **Help** tab (and any other tab) may be click-dragged to any window pane to customize the IDE layout.
2. To revert to the C/C++ perspective (rather than the Debug perspective for example), click the C/C++ icon in the top-right corner of the window.

A.2 Shortcuts

A useful cheat-sheet outlining shortcuts for the WICED Studio IDE (Eclipse) is included online at:

<http://www.cheat-sheets.org/saved-copy/eclipseCDT8.0-cheatsheet.pdf>

Particularly useful keystrokes are listed below:

- General search: to search the WICED-Studio tree for a variable:
 1. **Click** the root WICED-Studio folder in the Project Explorer pane.
 2. Press **CTRL-H** (on Windows).
 3. In the File Search tab, enter the variable name (regular expressions work too).
 4. Click **Search**.
- Search for a C source element (variable, function, enum, etc.).
 1. Open a C source file, for example: 20706-A2_Bluetooth\apps\hello_sensor\hello_sensor.c.
 2. Press **CTRL-SHIFT-T**.
 3. Start typing an element, for example, `BTM_BLE_ADVERT_`.
 4. Suggestions appear in the pop-up window.
- Press **ALT-Left** (arrow) and **ALT-Right** (arrow) to navigate between open files.

Appendix B. Multiple WICED Evaluation Boards

Multiple boards can be programmed from a single computer to run the same or different applications. To use the feature, edit the make targets for the required applications to add the `UART=<serial_port_name>` parameters, using the names of the appropriate HCI UART serial ports.

Figure B-1 shows two WICED evaluation boards connected to a PC and appropriate targets to build and download the hello sensor application.

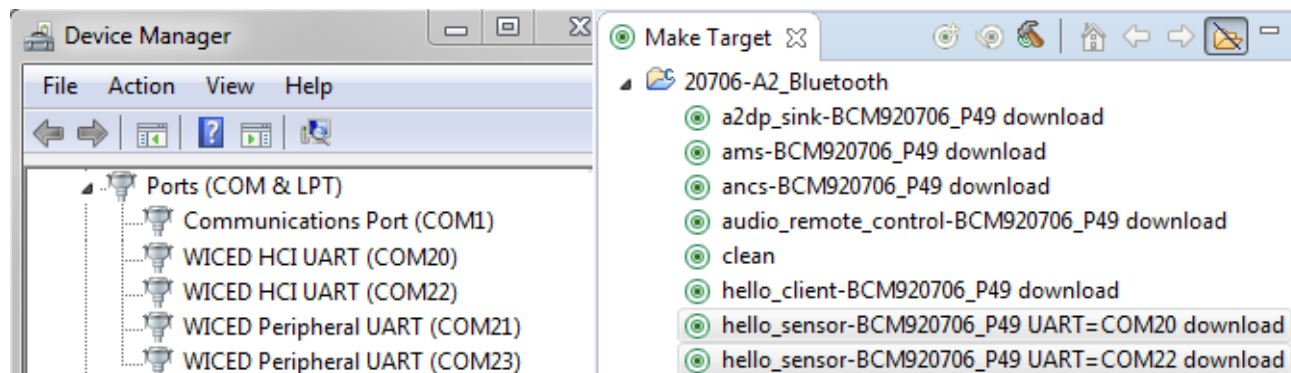


Figure B-1. Configuration for Two WICED Evaluation Boards

Appendix C. Connecting to Linux Platforms

An additional step is required when connecting a WICED board to a computer running Linux. On common Linux distributions, the serial UART ports (such as `/dev/ttySx` or `/dev/ttyUSBx` devices, including the `/dev/ttyWICED*` devices installed by WICED Studio) belong to the root user and to the dialout group. Standard users are not allowed to access these devices.

An easy way to allow the current user access to Linux's serial ports is by adding the user to the dialout group. This can be done using the following command:

```
$sudo usermod -a -G dialout $USER
```

Note: For this command to take effect, the user must log out and then log back in.

Appendix D. Recovering a Corrupted Board

The following steps describe how to reset a WICED evaluation board to factory defaults if it has become unresponsive to WICED Studio download attempts. This will erase any user application previously running or downloaded into memory and reset the board to reboot to the default state, ready to accept a new download. This can be necessary if a previous application had set the UART to an unrecognized baud rate (115200 and 3000000 are the default baud rates attempted by the download process), or if the serial flash had been corrupted.

Note: See [Figure 3-1](#) for port and switch references associated with the recovery steps provided in this section.

First, verify the following:

1. J25 of the WICED evaluation board is connected to the development PC.
2. The USB-UART driver is correctly installed (see [Verify Driver Installation](#)).
3. The SWx mini-switches are in their correct positions (see [Connect the WICED Evaluation Board](#)),

Next, use the following method to recover the board:

1. Press and hold the **Recovery** button (SW3).
2. Press and hold the **Reset** (SW2) button for 1 second.
3. Release SW2.
4. Release SW3.
5. Re-program the board as normal.

References

Document (or Item) Name	Number	Source
[1] CYW92070xV3_EVAL Evaluation Board Hardware User Guide	002-16535	community.cypress.com

Document Revision History

Document Title: WICED CYW20706 Kit Guide

Document Number: 002-18191

Revision	ECN	Issue Date	Description of Change
**	5576545	04/10/2017	Initial release

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



Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spanion 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, Spanion, the Spanion 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.