Chipsmall
深圳市津利帝科技有限公司
Geelead Technology Limited
采购部
情暖大凉山
公益助学行动

WICED Studio

CYW920719Q40EVB-01 Evaluation Board User Guide

Associated Part Family: CYW20719

Doc. No.: 002-22816 Rev. *B

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

Contents

Safety Information

The CYW920719Q40EVB-01 Evaluation Board (EVB) is intended for use as a development platform for hardware or software in a laboratory environment. The board is an open-system design, which does not include a shielded enclosure. Due to this reason, the board may cause interference to other electrical or electronic devices in close proximity. In a domestic environment, this product may cause radio interference. In such cases, take adequate preventive measures. Also, do not use this board near any medical equipment or RF devices.

Attaching additional wiring to this product or modifying the product operation from the factory default may affect its performance and cause interference with other apparatus in the immediate vicinity. If such interference is detected, suitable mitigating measures must be taken.

CYW920719Q40EVB-01 contains electrostatic discharge (ESD)-sensitive devices. Electrostatic charges readily accumulate on the human body and any equipment, and can discharge without detection. Permanent damage may occur on devices subjected to high-energy discharges. Proper ESD precautions are recommended to avoid performance degradation or loss of functionality. Store unused CYW920719Q40EVB-01 in the protective shipping package.

End-of-Life/Product Recycling

This kit has an end-of-life cycle of five years from the year of manufacturing mentioned on the back of the box. Contact your nearest recycler for discarding the kit.

General Safety Instructions

ESD Protection

ESD can damage boards and associated components. Cypress recommends that you perform procedures only at an ESD workstation. If an ESD workstation is not available, use appropriate ESD protection by wearing an antistatic wrist strap attached to the chassis ground (any unpainted metal surface) on the board when handling parts.

Handling Boards

CYW920719Q40EVB-01 boards are sensitive to ESD. Hold the board only by its edges. After removing the board from its box, place it on a grounded, static-free surface. Use a conductive foam pad if available. Do not slide the board over any surface. Any physical action on CYW920719Q40EVB-01 such as changing wires, jumper settings, or measuring voltages can cause stress on the CYW920719Q40EVB-01 printed circuit board assembly (PCBA). You must ensure that the PCBA has proper support on the bottom side to avoid stress on the PCBA when the EVB is in operation.

1 Introduction

Thank you for your interest in the CYW920719Q40EVB-01 Evaluation Board (EVB). The CYW920719Q40EVB-01 EVB enables customers to evaluate and develop single-chip Bluetooth and Bluetooth Low Energy (BLE) applications using the CYW20719 Bluetooth 5.0 and BLE wireless MCU.

The CYW920719Q40EVB-01 EVB can be used with WICED™ Studio to develop and debug your CYW20719 project. The CYW920719Q40EVB-01 EVB offers footprint-compatibility with Arduino shields. In addition, the kit features an onboard programmer/debugger and serial bridge chip. The CYW920719Q40EVB-01 EVB supports 1.8 V and 3.3 V operation.

WICED Studio supports application development using a WICED development board (CYW920719Q40EVB-01). The development system is compatible with Windows, OS X, and Linux operating systems. This document provides instructions for developing sample applications using WICED Studio.

Note: This document applies to WICED Studio 6.1 (or later).

The CYW920719Q40EVB-01 EVB is available through the Cypress Online Store or through our distributors.

1.1 CYW920719Q40EVB-01 EVB Contents

The CYW920719Q40EVB-01 EVB includes the following:

One CYW920719Q40EVB-01 evaluation board

One USB 2.0 Type-A to Micro-B cable

One Quick Start Guide

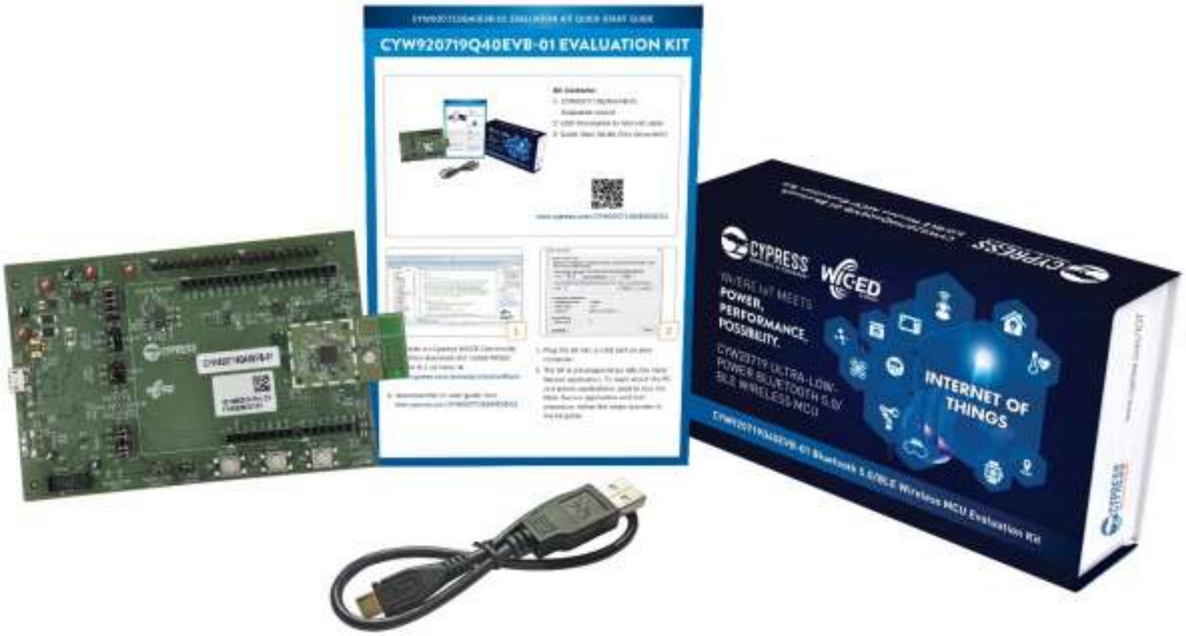

CYW920719Q40EVB-01 EVALUATION KIT
CYPRESS
WICED
POWER, PERFORMANCE, POSSIBILITY.
INTERNET OF THINGS

Figure 1-1. CYW920719Q40EVB-01 Kit Contents

Inspect the kit contents. If you find any part missing, contact your nearest Cypress sales office for assistance: www.cypress.com/support.

1.2 CYW920719Q40EVB-01 Board Details

Figure 1-2 shows CYW920719Q40EVB-01 with the following features:

1. CYW20719 module with onboard antenna
2. Expansion headers that are compatible with Arduino shields
3. Support for 1.8 V and 3.3 V operation of the CYW20719 device
4. Two user-controlled LEDs, one push button, one recovery button, and one reset button
5. Onboard micro-USB connector for programming and debug purposes

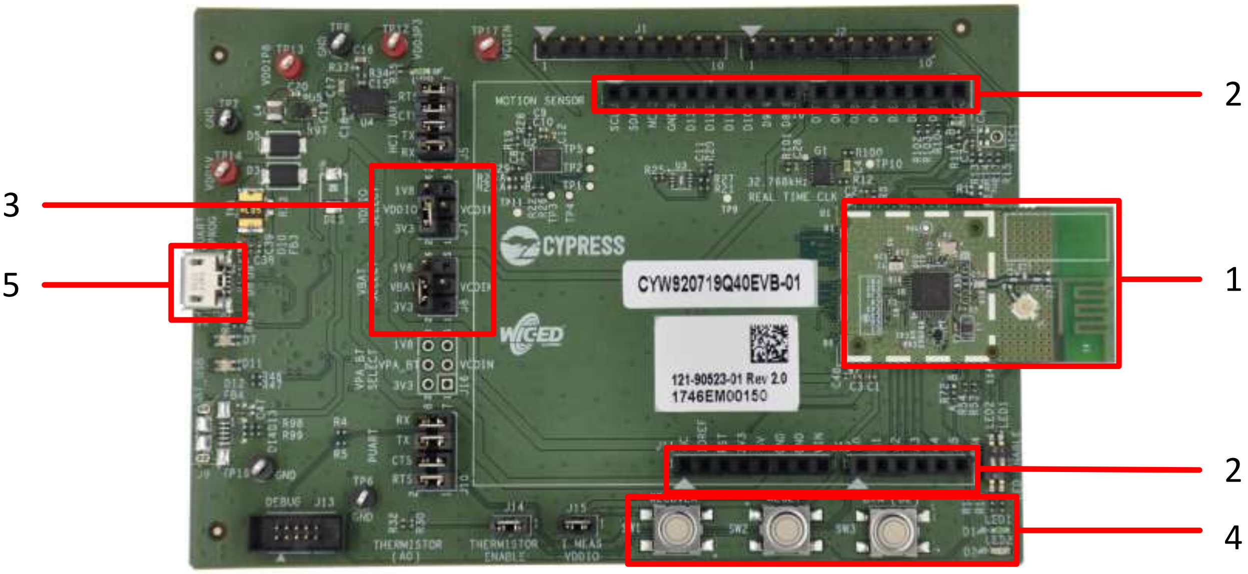


Figure 1-2. CYW920719Q40EVB-01 Evaluation Board

1.3 WICED Studio Development System

The WICED Studio Development System comprises a software development kit (SDK) along with the Eclipse integrated development environment (IDE) to enable development of projects with WICED evaluation boards.

The CYW920719Q40EVB-01 board and WICED Studio can be used for feature evaluation, debugging, and developing Bluetooth applications based on the CYW20719 device.

WICED Studio includes libraries and code examples which can speed up the design and development of user applications.

For detailed information on WICED Studio installation and usage, see Setting up WICED Studio.

1.4 Getting Started

This user guide will provide additional details of the CYW920719Q40EVB-01 EVB:

The WICED Studio chapter describes the installation and usage of the kit software. This includes WICED Studio to develop and debug the applications, Hello Client Peer application to test the hello_sensor application and BTSpy to view trace messages.

The Kit Operation chapter describes the operation of the kit and how to use its various features.

The Code Examples chapter describes code examples that will help you understand how to use an example with the kit.

The Hardware chapter describes the design details of the CYW920719Q40EVB-01 EVB hardware blocks.

1.5 IOT Resources and Technical Support

Cypress provides a wealth of wireless product documentation at www.cypress.com/products/wireless-connectivity to help you to select the right IoT device for your design. In addition, a professional community at community.cypress.com/community/wireless supplies developers the latest software and tools to solve common evaluation and integration problems while interacting directly with both Cypress applications engineers and experienced peers.

1.6 Additional Learning Resources

Visit the CYW920719Q40EVB-01 EVB webpage for additional learning resources including datasheets and application notes.

1.7 Document Conventions

Convention	Usage
`Courier New`	Displays source code examples.
`Consolas`	API and function names (when mentioned within body text) The `WICED_BT_TRACE()` macro can be used to generate printf-style messages from the application code.
Italics	Displays file names, file locations, and reference documentation: *C:\ ...cd\icc\*
File > Open	Represents menu paths: **File** > **Open** > **New Project**
Bold	Displays commands, menu paths, and icon names in procedures: Click the **File** icon and then click **Open**.
Times New Roman	Displays an equation: 2 + 2 = 4
Text in gray boxes	Describes Cautions or unique functionality of the product.

Table 1-1. Document Conventions for Guides

1.8 Acronyms

Acronym	Definition
ADC	Analog to Digital Converter
API	Application Programming Interface
BR	Basic Rate
BT / BLE	Bluetooth / Bluetooth Low Energy
EDR	Enhanced Data Rate
EEPROM	Electrically Erasable Programmable Read-Only Memory
EM	Electro-magnetic
ESS	Environment Sensing Service
EVB	Evaluation Board
GAP	Generic Access Profile
GATT	Generic Attribute Profile
GPIO	General Purpose Input Output
HAL	Hardware Abstraction Layer
HCI	Host Controller Interface
I^2C	Inter-Integrated Circuit
IDE	Integrated Development Environment
JRE	Java Runtime Environment
JTAG	Joint Test Action Group
LE	Low Energy
LED	Light Emitting Diode
LHL	Lean High Land
LPO	Low Power Oscillator
MEMS	Micro Electro-Mechanical System
NTC	Negative Temperature Coefficient
PCB	Printed Circuit Board
PUART	Peripheral UART
PWM	Pulse Width Modulation
RF	Radio Frequency
SDK	Software Development Kit
SIG	Special Interest Group
SoC	System-On-Chip
SPI	Serial Peripheral Interface
SWD	Serial Wire Debug
UART	Universal Asynchronous Receiver/Transmitter

Acronym	Definition
USB	Universal Serial Bus
VDD	Voltage Drain Drain
WICED	Wireless Internet Connectivity for Embedded Devices
XTAL	Crystal Oscillator

Table 1-2. List of Acronyms used in this Document

2 WICED Studio

This section provides detailed instructions to set up the Cypress Wireless Internet Connectivity for Embedded Devices (WICED; pronounced "wick-ed") CYW920719Q40EVB-01 evaluation board for use with the Cypress WICED Studio Development System for Bluetooth Classic (aka BR/EDR - Basic Rate/Enhanced Data Rate) and Low Energy (LE) devices.

WICED Studio supports application development using a WICED device such as the CYW20719 provided on the CYW920719Q40EVB-01 kit. The development system is compatible with Windows, Mac OS X, and Linux operating systems. This section describes the software components included in WICED Studio and provides instructions for compiling WICED sample applications using WICED Studio.

This document applies to WICED Studio 6.1 and WICED Bluetooth CYW20719 devices.

2.1 Before You Begin

All Cypress software installations require administrator privileges. Make sure that you have the required privileges on the system for successful installation. Before you install the kit software, close any other Cypress software that is currently running.

2.2 WICED Studio Overview

WICED Studio SDK includes the following:

Generic profile-level and BT stack level APIs (WICED BT API)

Sample applications that demonstrate the use of the API

Drivers to access on-chip peripherals using WICED HAL APIs (for example, UART, SPI, I2C, ADC, PWM)

WICED BT API documentation

Utilities to support development, testing, and mass production on Windows, Mac OS X, and Linux environments

2.3 Hardware and Software Requirements

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

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

2.4 Development Process

WICED Studio is distributed as executable installers for Windows, Mac OS X, and Linux. Follow these steps to prepare and run an application:

Download and install WICED Studio (see *Install WICED Studio*).

Connect the WICED evaluation board (see *Connect the WICED Evaluation Board*).

Create and load an application (see *Build and Load a Sample Application*).

2.5 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 prior to running the installer.

2.5.1 Install WICED Studio

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

2.5.1.1 Windows

1. Unzip the distribution to a local folder, along with the *config.enl* 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.

2.5.1.2 Linux

1. Unzip the distribution to a local folder, along with the *config.enl* 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 (for example, dnf, apt-get, yum) to update Java to the latest version. For example:

```
sudo dnf install java
```

2.5.1.3 Mac OS X

1. Unzip the distribution to a local folder.
2. If there is a *config.enl* file with the distribution, use Finder to copy the file 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. 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
```

2.5.2 Connect the WICED Evaluation Board

Figure 2-1 shows the CYW920719Q40EVB-01 WICED evaluation board.

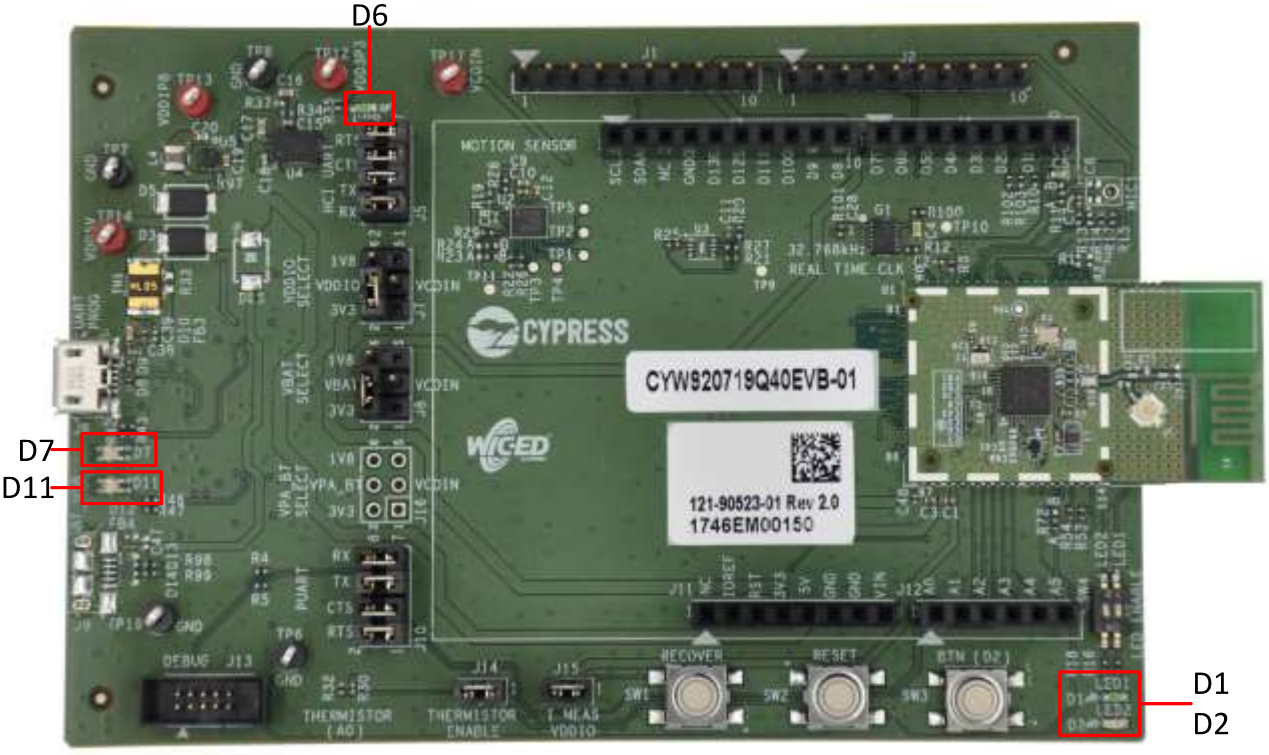


Figure 2-1. CYW920719Q40EVB-01 Evaluation Board

The Micro-USB connector (J6) supports UART connections and provides +5 V power to the board.

Follow these steps before connecting the board and verifying the driver installation:

1. Verify that all the jumpers are in default configuration as shown in Table 3-1 to Table 3-5, so that the Peripheral UART is selected and can display embedded application trace messages.
2. Connect J6 of the WICED evaluation board to the development PC with a USB cable. The USB UART driver should load automatically.

See Jumpers and Buttons and Switches for complete information on DIP switches and jumper settings.

The LEDs labelled on the board serve the following purposes:

D6 (Green) indicates that 3.3 V power is ON

D7 (variable color) indicates HCI UART activity

D11 (variable color) indicates peripheral UART activity

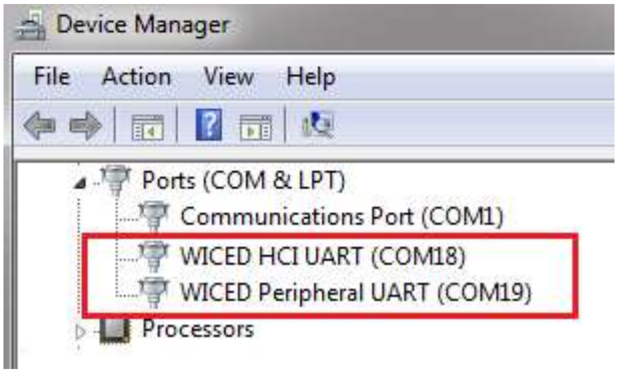
D1 (green) and D2 (red) are generic user LEDs controlled by GPIOs

2.5.3 Verify Driver Installation

2.5.3.1 Windows

1. Open the Device Manager (right-click **MyComputer**, 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 2-2, the 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.



Ports (COM & LPT)
Communications Port (COM1)
WICED HCI UART (COM18)
WICED Peripheral UART (COM19)
Processors

Figure 2-2. 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 new COM ports for the WICED board do not appear in the Device Manager after the drivers are installed via Windows Update, manually install the drivers from the *Drivers\Windows\wiced_uart* folder of the WICED Studio installation. Double-click the installer *DPInst.exe for 32-bit* Windows and *DPInst_x64.exe* for 64-bit Windows. Alternately, you can run the driver installers from inside WICED Studio; right-click the driver file name and select **Open With** > **System Editor**.

If the error persists, check all jumper settings (see Jumpers) on the board and replace the USB cable.

2.5.3.2 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. On common Linux distributions, the serial UART ports (such as */dev/ttySx* or */dev/ttyUSBx* devices, including the */dev/ttyWICED_xxx* 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. You can do this using the following command:

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

Note: For this command to take effect, you must log out and log in again.

2.5.3.3 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. Follow the instructions here and update to a specific version.

On OS X versions 10.11 and later, use the Apple version of the FTDI driver. Remove previous instance of the FTDI version of the driver, if any, 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.

2.6 Using the WICED Studio IDE

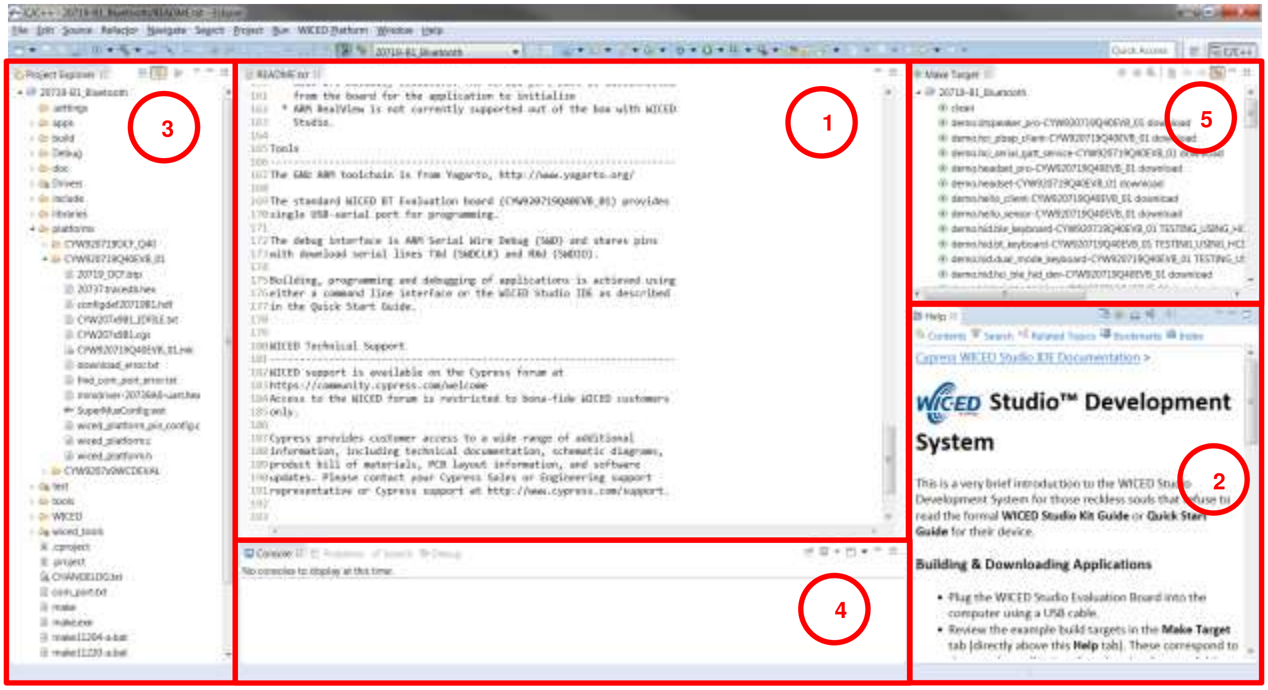
This section describes how to use the WICED Studio IDE to create application build targets for the WICED evaluation board and how to download applications to the board.

2.6.1 WICED Studio IDE UI

Double-click the WICED Studio icon on the desktop to start the IDE. Some WICED Studio packages may offer support for multiple devices, and on first-time execution of the IDE, you might be prompted to select the default platform. If prompted, select **20719-B1_Bluetooth**. You can change the default platform later from the **WICED Target Selector** drop-down list in the IDE.

Figure 2-3 shows the WICED Studio UI, illustrating the following functionality available through the IDE:

1. Edit your application firmware.
2. Access Help that contains instructions on building and downloading applications.
3. Explore existing applications/firmware and libraries of the SDK.
4. View Build messages in the Console window.
5. Create and edit Make Targets for the platform to build your application or project.



```
     from the board for the application to initialize
   * ARM RealView is not currently supported out of the box with WICED
     Studio.

Tools
--------------------------------------------------------------------------
The GNU ARM toolchain is from Yagarto, http://www.yagarto.org/

The standard WICED BT Evaluation board (CYW920719Q40EVB_01) provides
single USB-serial port for programming.

The debug interface is ARM Serial Wire Debug (SWD) and shares pins
with download serial lines TXd (SWDCLK) and RXd (SWDIO).

Building, programming and debugging of applications is achieved using
either a command line interface or the WICED Studio IDE as described
in the Quick Start Guide.


WICED Technical Support
--------------------------------------------------------------------------
WICED support is available on the Cypress forum at
https://community.cypress.com/welcome
Access to the WICED forum is restricted to bona-fide WICED customers
only.

Cypress provides customer access to a wide range of additional
information, including technical documentation, schematic diagrams,
product bill of materials, PCB layout information, and software
updates. Please contact your Cypress Sales or Engineering support
representative or Cypress support at http://www.cypress.com/support.
```

No consoles to display at this time.

Cypress WICED Studio IDE Documentation >

WICED Studio™ Development System

This is a very brief introduction to the WICED Studio Development System for those reckless souls that refuse to read the formal **WICED Studio Kit Guide** or **Quick Start Guide** for their device.

Building & Downloading Applications

- Plug the WICED Studio Evaluation Board into the computer using a USB cable.
- Review the example build targets in the **Make Target** tab (directly above this **Help** tab). These correspond to

Figure 2-3. WICED Studio IDE

2.6.2 WICED Studio SDK Directory Structure

Table 2-1 lists the directory structure of WICED Studio for the CYW20719 device. WICED Studio may support multiple types of WICED modules depending on the installed components; some modules may share components, files, and folders.

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 view the location of the component in the file system. For example, the ARM_GNU toolchain resides under the *43xxx_Wi-Fi* folder in the filesystem structure, but as used for the CYW20719 device, the *ARM_GNU* folder appears in the *wiced_tools* folder in the WICED Studio Project Explorer UI folder tree.

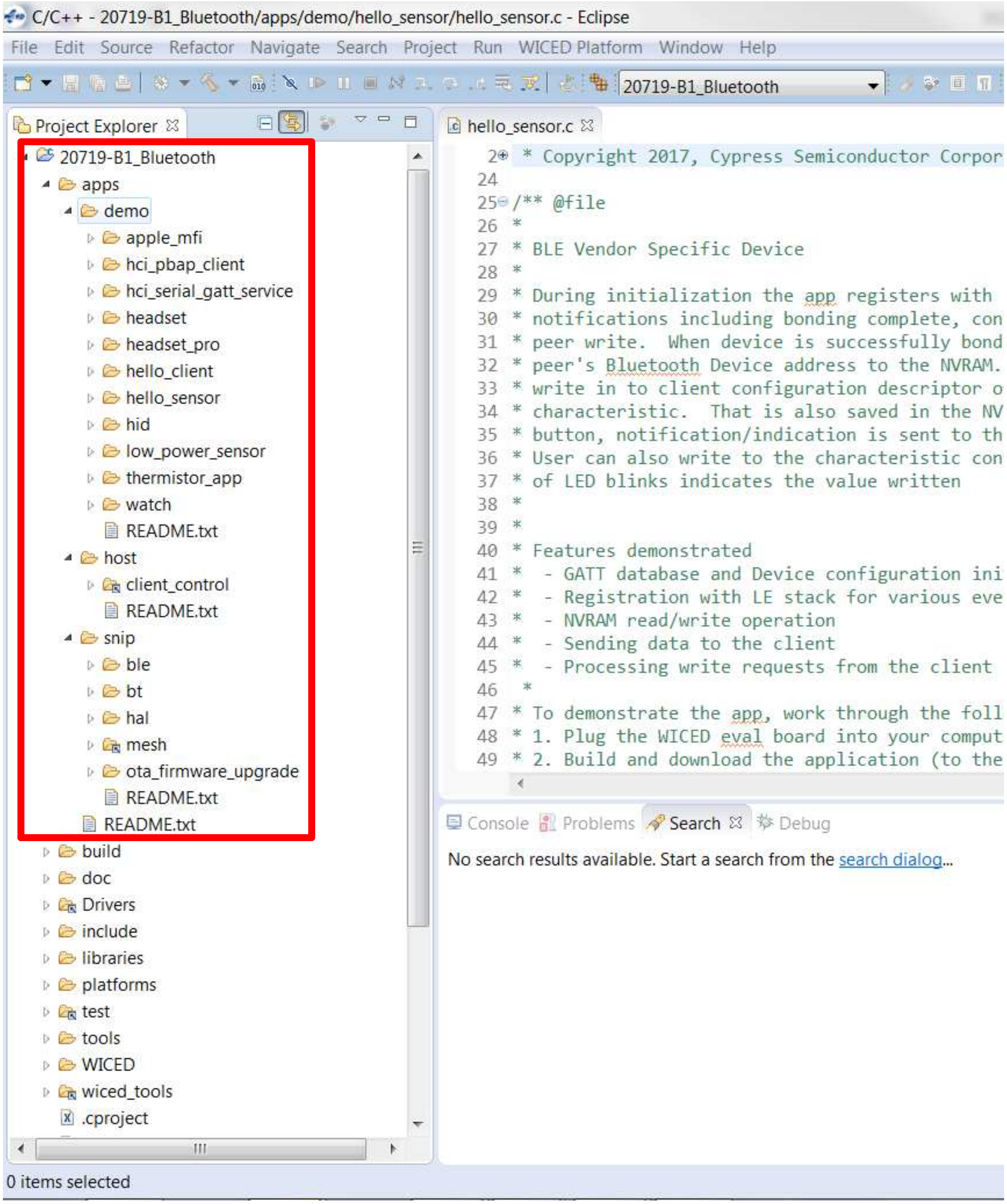
WICED Studio Project Explorer Directory	Directory Contents
20719-B1_Bluetooth\doc	Reference documentation
20719-B1_Bluetooth\Drivers	USB drivers for the evaluation board
20719-B1_Bluetooth\wiced_tools	Tools including programming tool, and other utilities and scripts
20719-B1_Bluetooth\wiced_tools\ARM_GNU	Toolchain including compiler, linker, and supporting files (libraries and headers)
20719-B1_Bluetooth\apps	Sample applications
20719-B1_Bluetooth\build	Output files of built applications
20719-B1_Bluetooth\include	WICED API function prototypes and definitions
20719-B1_Bluetooth\libraries	Sources for various WICED interface libraries
20719-B1_Bluetooth\platforms	Configuration files and information for supported hardware platforms
20719-B1_Bluetooth\test	Tools provided for automation testing
20719-B1_Bluetooth\tools	Common utilities used by the IDE build processes
20719-B1_Bluetooth\WICED	WICED core components

Table 2-1. WICED Studio SDK Directory Structure

2.6.3 WICED Studio Code Examples

Application examples can speed up the design process by serving as templates for development. Code examples are located under the apps category (in the Project Explorer window), as shown in Figure 2-4.

The *demo* directory contains applications that combine various WICED features into a single application. The *snip* directory contains application snippets that demonstrate how to use various WICED libraries and API functions. The *host* directory contains applications that demonstrate WICED BT API usage for host MCU apps. Located within each subdirectory in the *apps* folder is a *README.txt* that lists and summarizes the applications located within the folder:



```
 * Copyright 2017, Cypress Semiconductor Corpor

/** @file
 *
 * BLE Vendor Specific Device
 *
 * During initialization the app registers with
 * notifications including bonding complete, con
 * peer write.  When device is successfully bond
 * peer's Bluetooth Device address to the NVRAM.
 * write in to client configuration descriptor o
 * characteristic.  That is also saved in the NV
 * button, notification/indication is sent to th
 * User can also write to the characteristic con
 * of LED blinks indicates the value written
 *
 *
 * Features demonstrated
 *   - GATT database and Device configuration ini
 *   - Registration with LE stack for various eve
 *   - NVRAM read/write operation
 *   - Sending data to the client
 *   - Processing write requests from the client
  *
 * To demonstrate the app, work through the foll
 * 1. Plug the WICED eval board into your comput
 * 2. Build and download the application (to the
```

Figure 2-4. Code Examples under `apps` *Category*

For more details on the WICED software stack and APIs, see the documents available in the *doc* folder in the Project Explorer pane of WICED Studio. The *API.html* file contains API documentation, and it is recommended that you open the file using the default system web browser. Right-click *API.html* and select **Open With** > **System Editor**.

2.6.4 Build and Load a Sample Application

The **Help** pane in the lower-right corner of the IDE (see Figure 2-3) describes how to build and download the sample applications shown in the **Make Target** pane, which is located above the **Help** pane. The **Help** pane also 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 CYW920719Q40EVB-01 evaluation boards.

The hello_sensor sample demonstrates basic LE sensor functionality. The source file *hello_sensor.c* contains information on how to exercise the application with the Hello Client host peer application. You can find the source code for the hello_sensor sample and Hello Client host peer application in WICED Studio in the **Project Explorer** pane under *20719-B1_Bluetooth\apps\demo\hello_sensor*. This example shows how to build and run the hello_sensor sample application on the CYW20719:

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. In the WICED Studio **Make Target** pane, find the make target *demo.hello_sensor-CYW920719Q40EVB_01 download* in the list. Double-click the Make Target to execute it. The IDE console pane (bottom center of the IDE window) will display the build and download progress.

   The build output should look similar to the following:

```
14:06:29 **** Build of configuration Release for project 20719-B1_Bluetooth ****
"C:\\Users\\<USER>\\Documents\\WICED-Studio-X.X\\20719-B1_Bluetooth\\make.exe"
demo.hello_sensor-CYW920719Q40EVB_01 download
Linking target ELF
..\..\..\43xxx_Wi-Fi\tools\ARM_GNU\bin\Win32\arm-none-eabi-ld: Warning: type of symbol
`' changed from 2 to 1 in WICED\chips\20719B1\patches/patch.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 @ 00216385
OK, made C:/Users/<USER>/Documents/WICED-Studio-X.X/20719-
B1_Bluetooth/WICED/wpan/../../build/hello_sensor-CYW920719Q40EVB_01-rom-ram-Wiced-
release/A_20719B1-hello_sensor-rom-ram-spar.cgs. MD5 sum is:
77a9eb1634d7b9716de18582d787103a *../../build/hello_sensor-CYW920719Q40EVB_01-rom-ram-
Wiced-release/A_20719B1-hello_sensor-rom-ram-spar.cgs
-------------------------------------------------------------------------------
Patch code starts at              0x00270400 (RAM address)
Patch code ends at                0x00273B9C (RAM address)
Patch RW/ZI size                  1880 bytes
Application starts at             0x00215348 (RAM address)
Application ends at               0x00216381 (RAM address)

Patch code size                        14236 bytes
Application RAM footprint               4153 bytes
                                      ------
Total RAM footprint                     6033 bytes (5.9kiB)
-------------------------------------------------------------------------------
Converting CGS to HEX...
Conversion complete

Creating OTA images...
Conversion complete
OTA image footprint in NV is 23296 bytes

Downloading application...
Download complete

Application running. Press Reset button on the 20719-B1 board before opening the WICED
HCI port from the Client Control app.

14:06:42 Build Finished (took 13s.108ms)
```

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, it is possible the memory on the board has been corrupted by a previously loaded application, or the application used a custom baud rate that the download process does not detect. In that case it may be necessary to put the board in to recovery mode. To do this, first, press and hold the **Recovery** button (SW1), then press the **Reset** button (SW2), release the **Reset** button (SW2), and then release the **Recovery** button (SW1).

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 on-chip flash. The board will reset after the download is successful. When the board boots and finds a valid application image in on-chip flash, it will execute the image.

2.6.5 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 CYW20719 device. You can find the sample application in *20719-B1_Bluetooth\apps\demo\hello_sensor\peer_apps* in the WICED Studio Project Explorer.

The entire source code of the application is provided for Windows, iOS, and Android, along with an executable binary that runs on 32-bit and 64-bit Windows 10 systems. The application is also compatible with Windows 8 and 8.1, but not with lower versions. These steps demonstrate the use of the Windows-based Hello Client application; however, the process is similar for all host operating systems:

1. On the Windows host PC, open the **Settings** screen (usually from **Start**), select **Devices**, and then **Bluetooth**.
2. Click **Add a device** and wait while the PC searches for the devices in range.
3. Select the **Hello Sensor** device, which will appear as "Hello" (this is the embedded application running on CYW20719).
4. Click **Pair** and wait for the device connection to complete.
5. Run *HelloClient.exe* on the Windows host PC. The executable is in the WICED Studio Project Explorer under *apps\demo\hello_sensor\peer_apps\Windows\HelloClient\Release\<x64|x86>.*

   If multiple devices have been paired, a Hello Client Select Device window similar to that shown in Figure 2-5. appears. 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. You can select the correct device from the list displayed in this window.

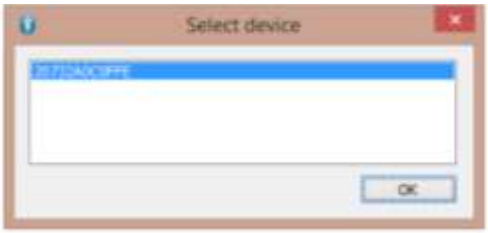


Figure 2-5. Hello Client Select Device Window

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 to 10 seconds.

   If only one device has been paired, the selection screen will not appear

   Note: By default, 20719-B1 LE applications use LE random addressing. If desired, this can be changed to use a specific address, or you may specify a portion of the address to be specific and a portion to be generated randomly. To specify all or part of the Bluetooth device address, edit the file *20719-B1_Bluetooth\apps\demo\hello_sensor\wiced_bt_cfg.c* in WICED Studio Project Explorer and disable LE random addressing by changing the setting to "WICED_BT_CFG_DEFAULT_RANDOM_ADDRESS_NEVER_CHANGE" in the following line:

   ```
   .rpa_refresh_timeout = WICED_BT_CFG_DEFAULT_RANDOM_ADDRESS_CHANGE_TIMEOUT,
   ```

   When LE random addressing is disabled, the Bluetooth device address programmed in the evaluation board comes from the file *20719-B1_Bluetooth\platforms\CY920719Q40EVB_01\20719_OCF.btp* in the WICED Studio Project Explorer.

   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=xxxxxxxxxxxx`, where xxxxxxxxxxxx is a 12-digit hexadecimal value representing the Bluetooth device address. You can substitute the digits 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.

2.6.6 Testing the Hello Sensor Application

2.6.6.1 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:

- o Manually, by clicking **Read** on the HelloClient PC application.
- o Automatically, by pressing button **SW 3** (see Figure 3-3) on the WICED evaluation board (**Allow Notifications** must be selected from the **Value** drop-down list 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 3-3) when the value is changed and whenever button **SW 3** 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

The application sends dummy battery value and increments for every GATT read operation of the battery level. In order to get the actual values from the battery, please refer to *include/hal/wiced_hal_batmon.h* file for using ADC to measure battery level.

2.6.6.2 Hello Sensor Application Test Procedure

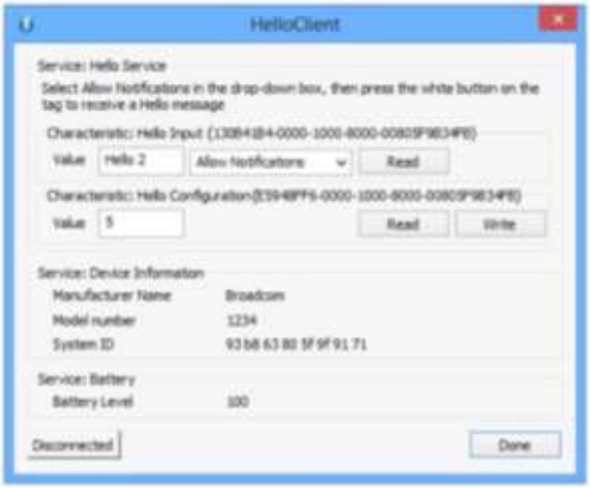
To test the application with a WICED evaluation board, follow the instructions provided for each of the Hello Service characteristics (see Figure 2-6).

2.6.6.2.1 Hello Input Characteristic

1. Select **Allow Notifications** from the **Value** drop-down list for Hello Input.
2. Press button **SW 3** on the WICED board (see Figure 3-3). The Hello X message is displayed in the **Value** field. Each time the button is pressed, the message number increments.

2.6.6.2.2 Hello Configuration Characteristic

1. Change the **Value** field for the Hello Configuration to **5** and then click **Write**. The red LED on the board blinks 5 times
2. Press button **SW 3** on the WICED board (see Figure 3-3). The LED on the board again blinks five times.



Service: Hello Service

Select Allow Notifications in the drop-down box, then press the white button on the tag to receive a Hello message

Characteristic: Hello Input (1B0B41B4-0000-1000-8000-00805F9B34FB)

Value: Hello 2 — Allow Notifications

Characteristic: Hello Configuration(E3948FF6-0000-1000-8000-00805F9B34FB)

Value: 5

Service: Device Information

Manufacturer Name	Broadcom
Model number	1234
System ID	93 b8 63 80 5f 9f 91 71

Service: Battery

Battery Level	100

Figure 2-6. HelloClient Services Window

2.6.7 Viewing Application Trace Messages

You can view trace messages from the application if the application is compiled with the `WICED_BT_TRACE_ENABLE` compile flag defined. You can define compile flags in the application *makefile*. In the hello_sensor application, the `WICED_BT_TRACE_ENABLE` flag is enabled by default (see *makefile.mk* in the hello sensor application source folder in WICED Studio).

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

2.6.7.1 Routing Trace Messages

You can route trace messages 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 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: Downloads from WICED Studio use the HCI UART serial port, so if you are also using the HCI UART serial port for debug messages, do not open the HCI UART port with the terminal emulation program until after the application download has completed in WICED Studio to avoid contention. The PUART serial port has no such contention, so the hello_sensor sample application uses the PUART by default. The other options (HCI_UART and WICED_UART) are provided in the source file as comments. You can comment out the PUART lines and uncomment either the HCI_UART or WICED_UART line if you want to try one of the other options.

2.6.7.2 View Traces Using a Terminal Emulation Program

To view traces with a terminal emulation program (such as Tera Term or PuTTY 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[a]

Data: 8 bit

Parity: None

Stop: 1 bit

Flow Control: None

Terminal ID: VT100

New-Line Receive: Auto

Press **Reset** (SW2) on the WICED evaluation board to view the application startup messages. For example:

```
Hello Sensor Start
hello_sensor_management_cback: 15
local keys read from NVRAM result: 0
hello_sensor_management_cback: 0
hello_sensor_application_init
wiced_bt_gatt_register: 0
wiced_bt_gatt_db_init 0
hello_sensor_load_keys_for_address_resolution 00 00 00 00 00 00  result:40
hello_sensor_management_cback: 17
Advertisement State Change: 3
wiced_bt_start_advertisements 0
hello_sensor_timeout: 1, ft:959
hello_sensor_timeout: 2, ft:1944
hello_sensor_timeout: 3, ft:2927
```

2.6.7.3 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, you can view traces using the separate BTSpy and ClientControl utilities. These Windows applications are supplied with WICED Studio and are available in the *wiced_tools* and *apps\host\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 configure the stack to generate Bluetooth protocol trace messages showing all activity between the stack and the controller over the virtual HCI interface embedded in the CYW20719 device. The Bluetooth protocol trace messages will be encoded into WICED HCI message packets and sent along with application trace messages (from `WICED_BT_TRACE()`), which can be encoded and displayed by the BTSpy utility. You can then view the application trace messages in sequence with the corresponding Bluetooth protocol trace messages.

[a] The Hello Sensor application uses 115200 as the default baud rate for the PUART, where traces are routed by default in that application. Other applications in WICED Studio may use different trace destinations and 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 for routing traces to the HCI UART port, and in any calls to the `wiced_hal_puart_set_baudrate()` API overriding the default rate for routing traces to the PUART port.

To configure the stack to generate Bluetooth protocol trace messages, applications must route trace messages to the ClientControl/BTSpy receivers by calling wiced_set_debug_uart (WICED_ROUTE_DEBUG_TO_WICED_UART), and 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, for an example.

To view application and Bluetooth protocol traces in BTSpy:

1. Define `WICED_BT_TRACE_ENABLE` in the application *makefile.mk* file. For example, add the line:

   ```
   C_FLAGS += -DWICED_BT_TRACE_ENABLE
   ```

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,* to route to the PUART by default. However, commented out code is provided to change this to the WICED_UART instead.

3. Compile and download the application to the evaluation board.
4. Press **Reset** (SW2) on the WICED evaluation board to reset the HCI UART after downloading, before opening the port with ClientControl.
5. Run the ClientControl utility from *apps\host\client_control*. You can open the utility from inside WICED Studio. Right-click the utility and select **Open With** > **System Editor**.
6. In the ClientControl utility, select the HCI UART port in the UI, and set the baud rate to the baud rate used by the application for the HCI UART (3000000 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).
7. Run the BTSpy utility from *wiced_tools*. You can open the utility from inside WICED Studio. Right-click the utility and select **Open With** > **System Editor**.
8. In the ClientControl utility, open the HCI UART COM port by clicking the Open Port button.
9. Press **Reset** (SW2) on the WICED evaluation board to reset the kit and view trace messages in the BTSpy window.

3 Kit Operation

This section provides detailed instructions to set up the Cypress Wireless Internet Connectivity for Embedded Devices (WICED; pronounced "wick-ed") CYW920719Q40EVB-01 evaluation board for use with the Cypress WICED Studio Development System for Bluetooth Classic (aka BR/EDR - Basic Rate/Enhanced Data Rate) and Low Energy (LE) devices.

This chapter introduces you to the CYW920719Q40EVB-01 EVB and the features that will be used as part of kit operation. This chapter also discusses features such as the Bluetooth connection and programming/debugging as well as the USB-UART bridge device that can be used to communicate with the CYW20719-B1 device on this EVB.

3.1 Theory of Operation

The CYW920719Q40EVB-01 is built around the CYW20719-B1 device. Figure 3-1 shows the block diagram of the CYW20719-B1 device. See the CYW20719-B1 datasheet for details of device features.

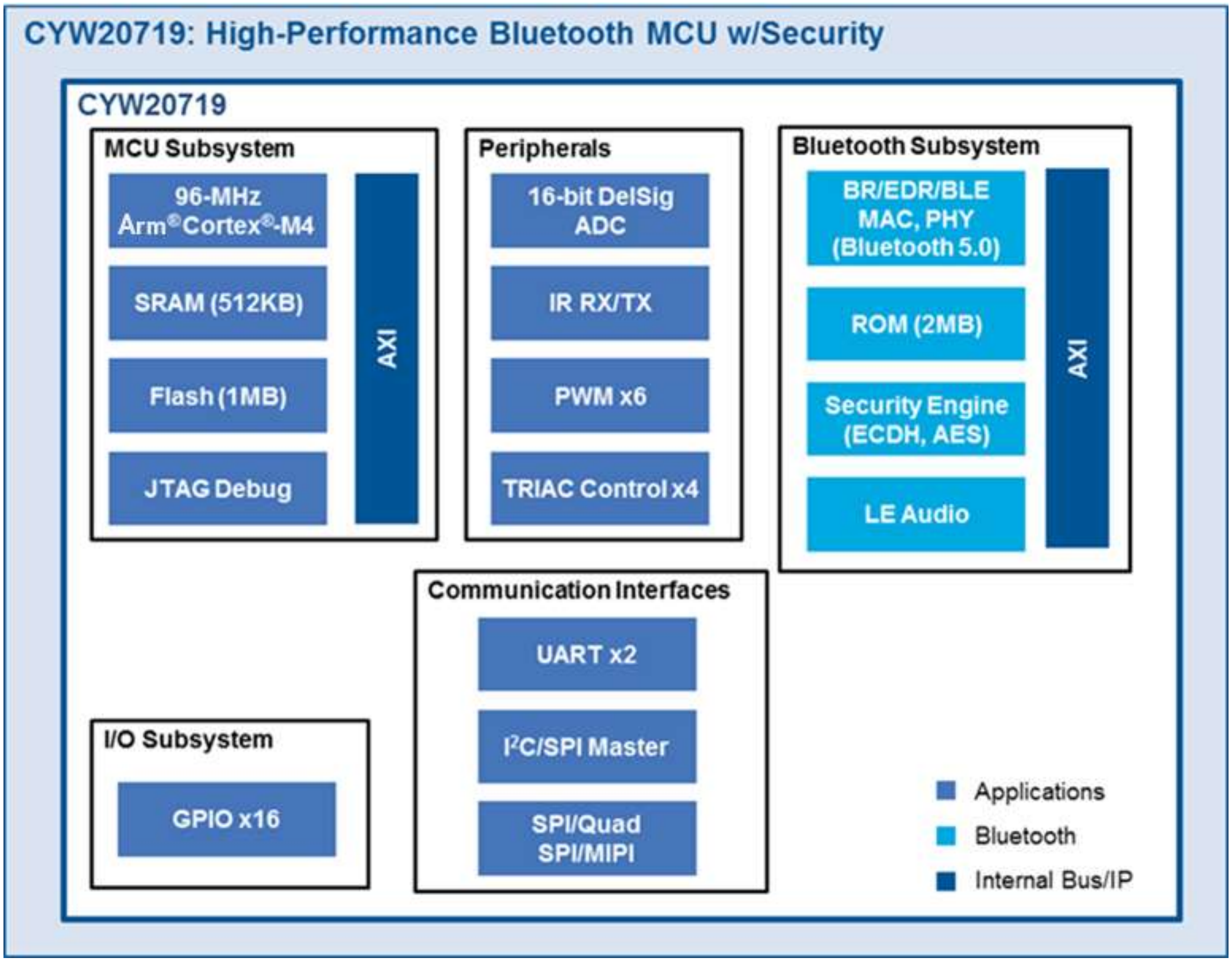


Figure 3-1. Block Diagram of CYW20719-B1 Device

Figure 3-2 illustrates the block diagram of the CYW920719Q40EVB-01 Evaluation Board. This board contains a CYW20719-B1 Bluetooth device and a USB-Serial interface/debugger. The kit features Arduino form-factor compatible headers, which enables Arduino shields to be plugged on top, extending its capabilities. It also features one user switch, one reset switch, one recovery switch, two user LEDs, a thermistor, and a motion sensor.

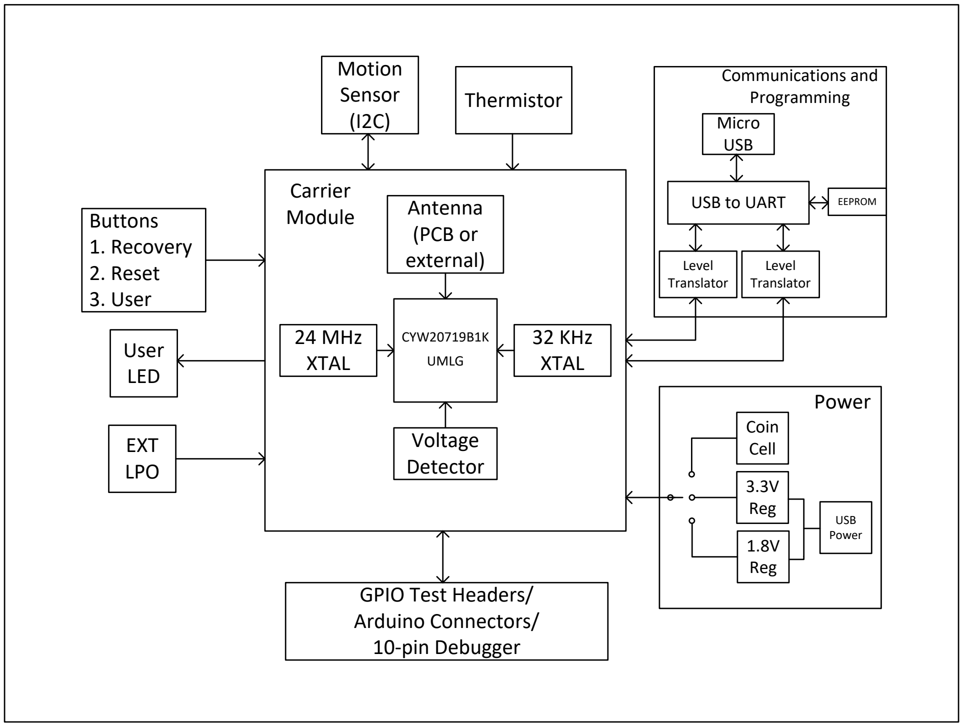


Figure 3-2. Block Diagramof CYW920719Q40EVB-01 EVB

Figure 3-3 and Figure 3-4 show the markup of the CYW920719Q40EVB-01 board. See the list below for a description of the numbered items.

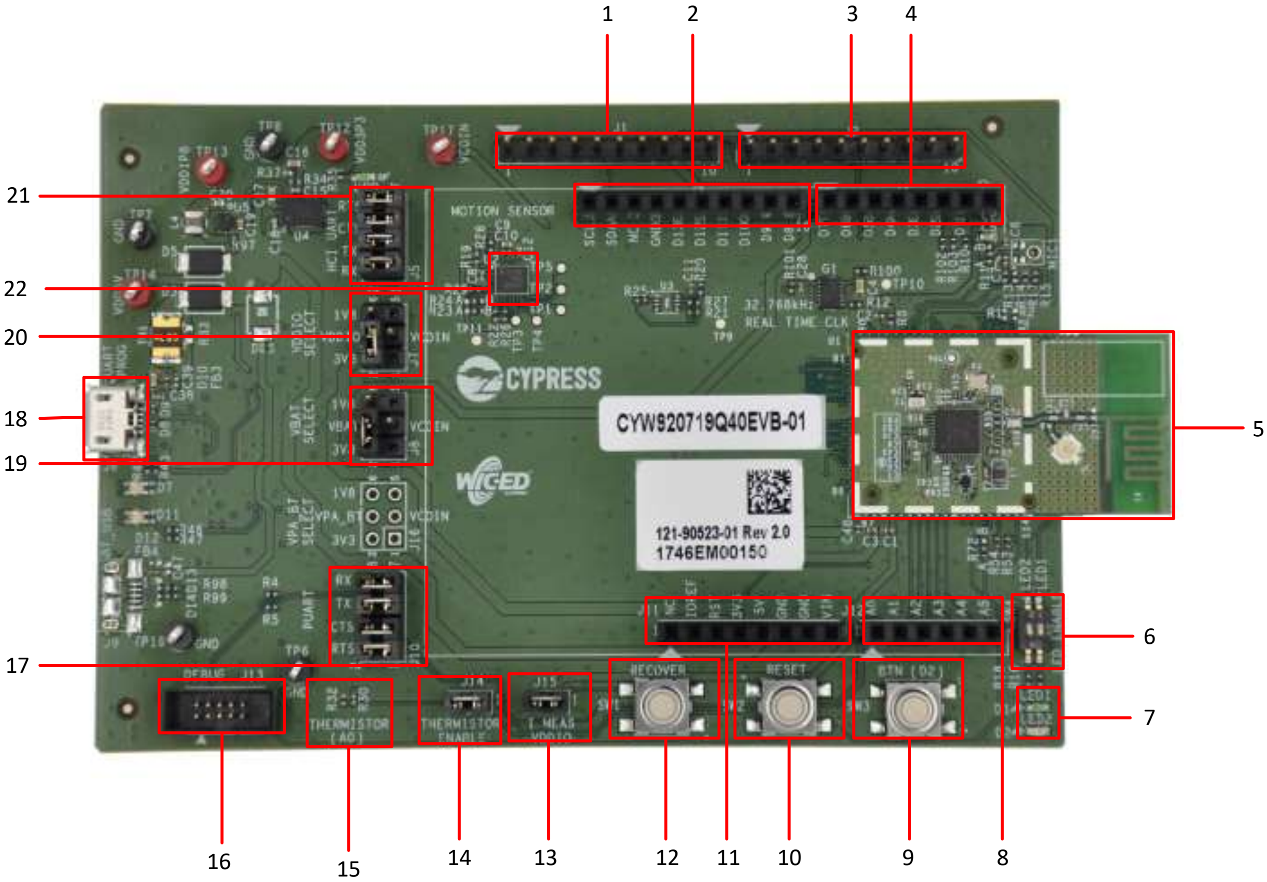


Figure 3-3. CYW920719Q40EVB-01 Evaluation Board