CYW9MCU7X9N364/Iwa Platform

Quick Start Guide

Associated Part Family: CYW20719, CYW43364

Doc. No.: 002-20918 Rev. *C

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

Contents

About This Document

Purpose and Scope

This document provides information about the CYW9MCU7X9N364/Iwa WICED™ platform. It provides details about the hardware; and how to build, flash, and debug applications on the CYW9MCU7X9N364/Iwa platform.

Note: This document applies to WICED SDK 5.2 or higher.

Audience

This document is for software developers and testers who are using the WICED Development System to create/validate applications using the CYW9MCU7X9N364/Iwa platform.

Acronyms and Abbreviations

In most cases, acronyms and abbreviations are defined on first use.

For a comprehensive list of acronyms and other terms used in Cypress documents, go to www.cypress.com/glossary.

1 Platform

1.1 Overview

CYW9MCU7X9N364/Iwa is a discrete combo MCU evaluation board, which has CYW20719 (Bluetooth) and CYW43364 (Wi-Fi) chips. CYW20719 [1] is an Ultra-Low-Power Bluetooth/BLE Connectivity chip compliant with the Bluetooth 5.0 specification. CYW20719 is also used as an MCU driving the Wi-Fi chip, thus avoiding the need for a dedicated MCU. This platform is suitable for low-cost and low-power IoT applications. CYW43364 [2] is a single-band 2.4-GHz IEEE 802.11 b/g/n-compliant Wi-Fi chip.

Figure 1-1 shows the high-level hardware block diagram of the CYW9MCU7X9N364/Iwa platform. More hardware details can be found in the hardware schematics document [3].

Figure 1-2 shows the high-level software block diagram for the CYW9MCU7X9N364/Iwa platform.

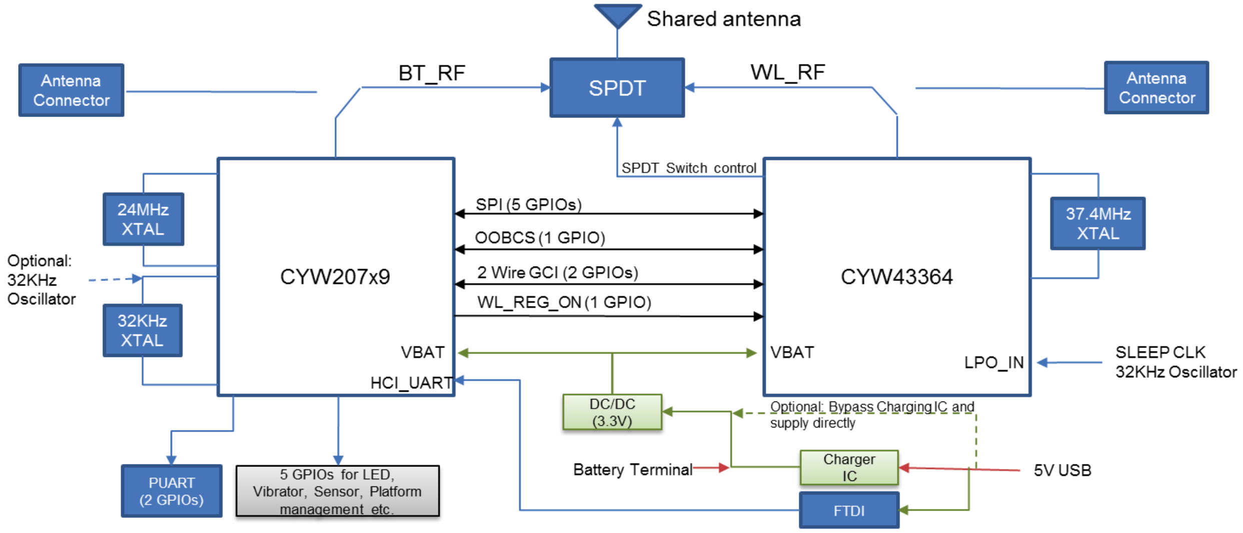


Figure 1-1. High-Level Hardware Block Diagram for CYW9MCU7X9N364/Iwa Platform

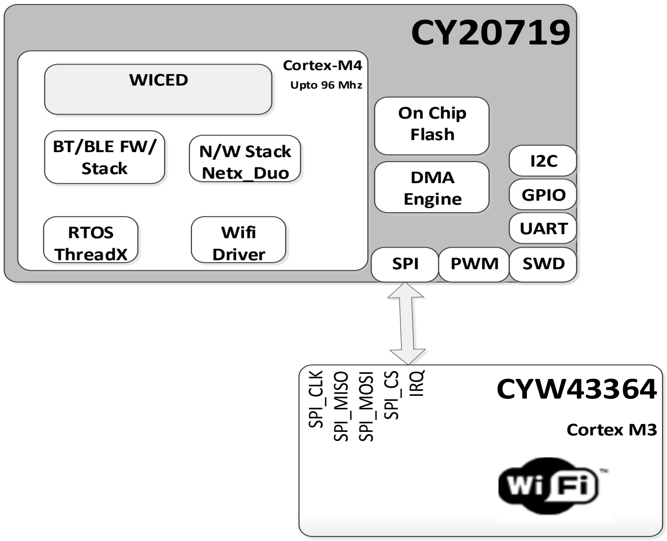


Figure 1-2. High-Level Software Block Diagram for CYW9MCU7X9N364/Iwa Platform

1.2 Features

- MCU: CYW20719
- Bluetooth/BLE: CYW20719
- WLAN: CYW43364
- Single-band 2.4-GHz IEEE 802.11 b/g/n
- Single shared antenna with co-existence (COEX) support
- USB-JTAG/SWD debug interface
- USB-Serial UART interface
- DMA, I^{2}C, SPI, PWM interfaces
- 5 GPIOs available for LED, vibrator, sensor, and platform management
- 2 LEDs (green and red) for user applications
- 1 button available for user applications
- Power supply: external +5 V to +12 V
- Reset button

1.3 CYW9MCU7X9N364/Iwa Board

Figure 1-3 shows the CYW9MCU7X9N364/Iwa board and highlights the components referred later in the document.

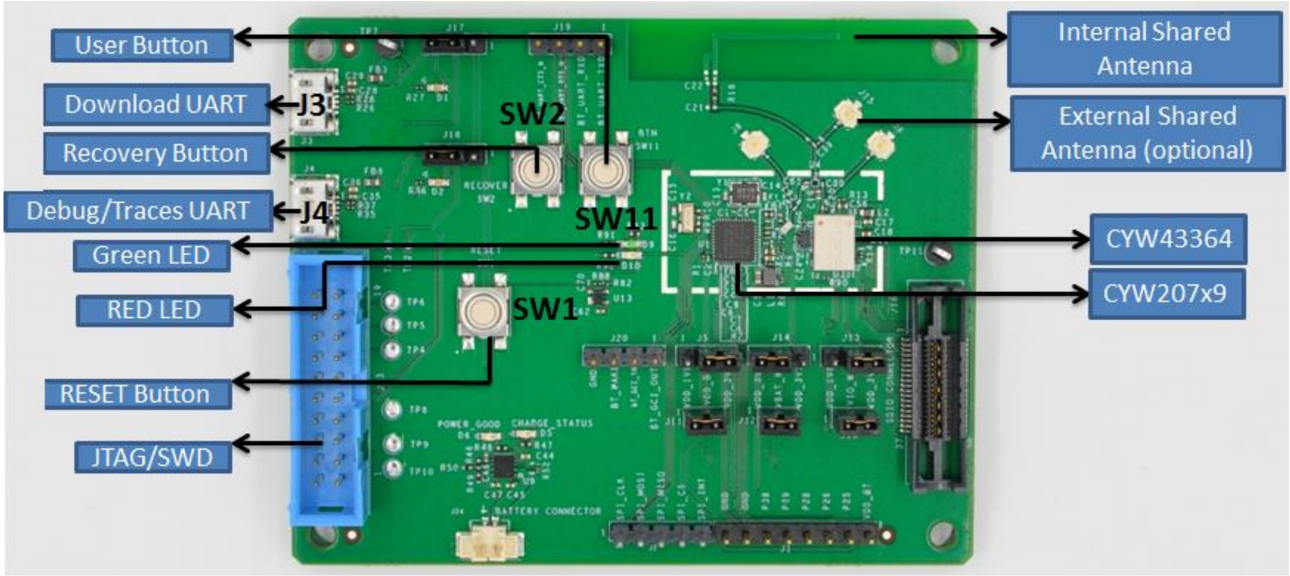


Figure 1-3. CYW9MCU7X9N364/Iwa Board Details

1.4 UART Ports

Two UART ports mounted on the board:

- **J3**: For downloading the software using BT_HOST_UART pins from the MCU
  - This port is enumerated as “WICED HCI UART” on the PC
- **J4**: For collecting logs/traces using PUART pins from the MCU
  - This port is enumerated as “WICED Peripheral UART” on the PC

Detailed instructions on how to download the image and collect traces are explained in the subsequent sections.

1.5 User Button

Application developers have an option to use one button (SW11) that can be programmed for user-defined functionality. To enable this button, make sure that SW10 Pin 1 is in the ON position on the board.

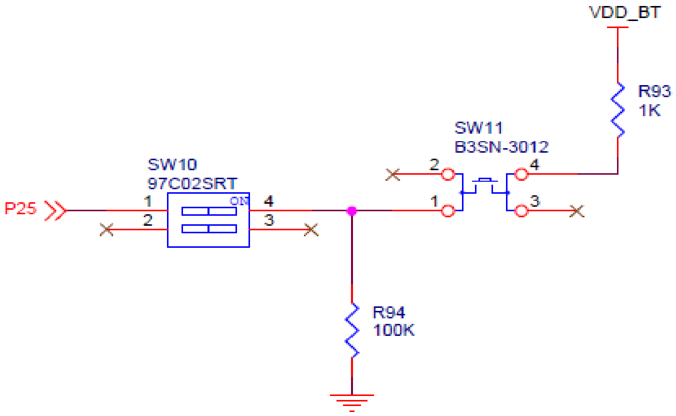


Figure 1-4. User Button and SW10 Settings

1.6 Reset Button

SW1 on the board acts as the reset button to reset the target board connected to the MCU.

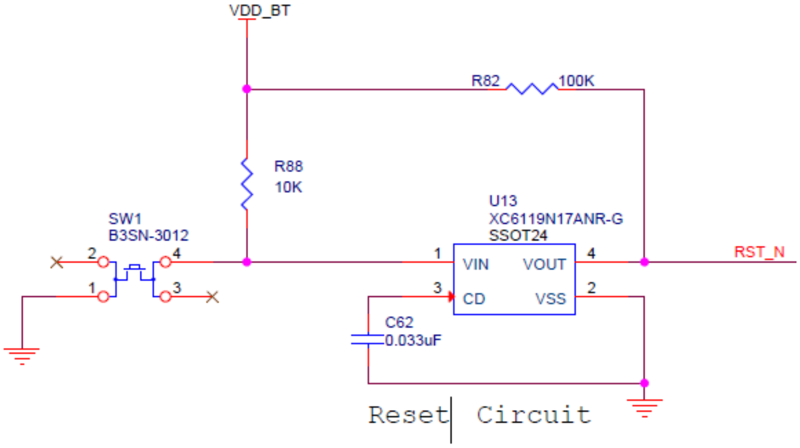


Figure 1-5. Reset Button Circuit

1.7 LED

Two LEDs are mounted on the board. To use these LEDs, make sure that SW9 pins 1 and 2 are in the ON position.

- GREEN LED connected to GPIO P29
- RED LED connected to GPIO P28

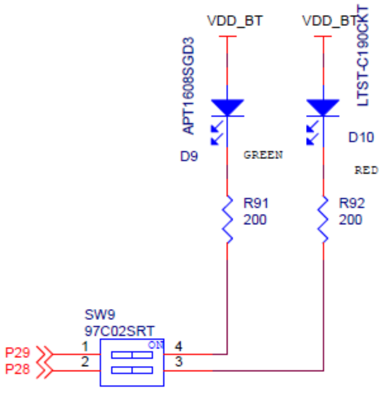
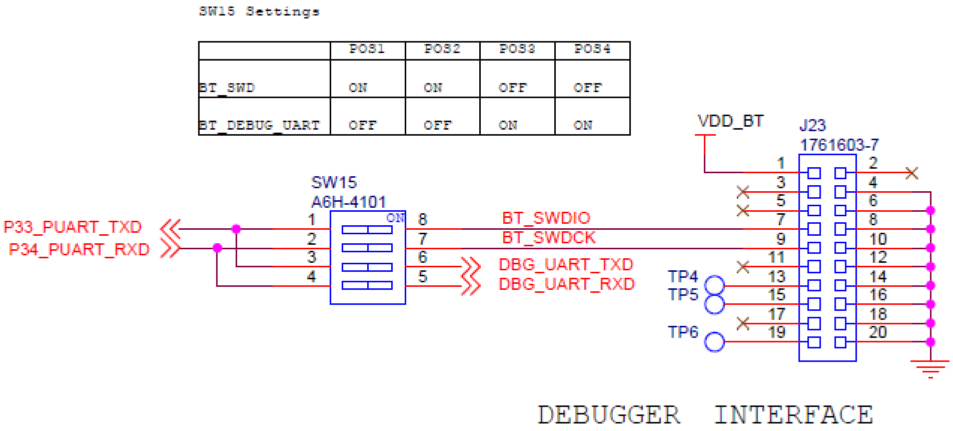


Figure 1-6. LEDs and SW9 Settings

1.8 JTAG Connection

On this board, UART lines used for collecting traces and debugger (SWD) lines are muxed. You can use only one functionality at a time. To enable SWD, put pins 1 and 2 to ON, and pins 3 and 4 to OFF position.


SW15 Settings

	POS1	POS2	POS3	POS4
BT_SWD	ON	ON	OFF	OFF
BT_DEBUG_UART	OFF	OFF	ON	ON

Figure 1-7. SW15 Settings for JTAG Connection

1.9 PWM

Four PWM channels are available on board, connected to GPIO pins 26, 27, 28, and 29. The PWM functionality can be demonstrated through the *temp_control* WICED demo application with varying duty cycles applied on the GREEN and RED LEDs mounted on the board.

1.10 Antenna Configuration

By default, the board comes with an internal shared antenna. For debug purposes, you can connect a single shared external antenna as well as separate dual antennas for BT and Wi-Fi.

Make the following changes for a dual-antenna configuration:

- Change **C20** Capacitor from C to B (Wi-Fi external antenna)
- Change **C67** Capacitor from C to A (BT external antenna)

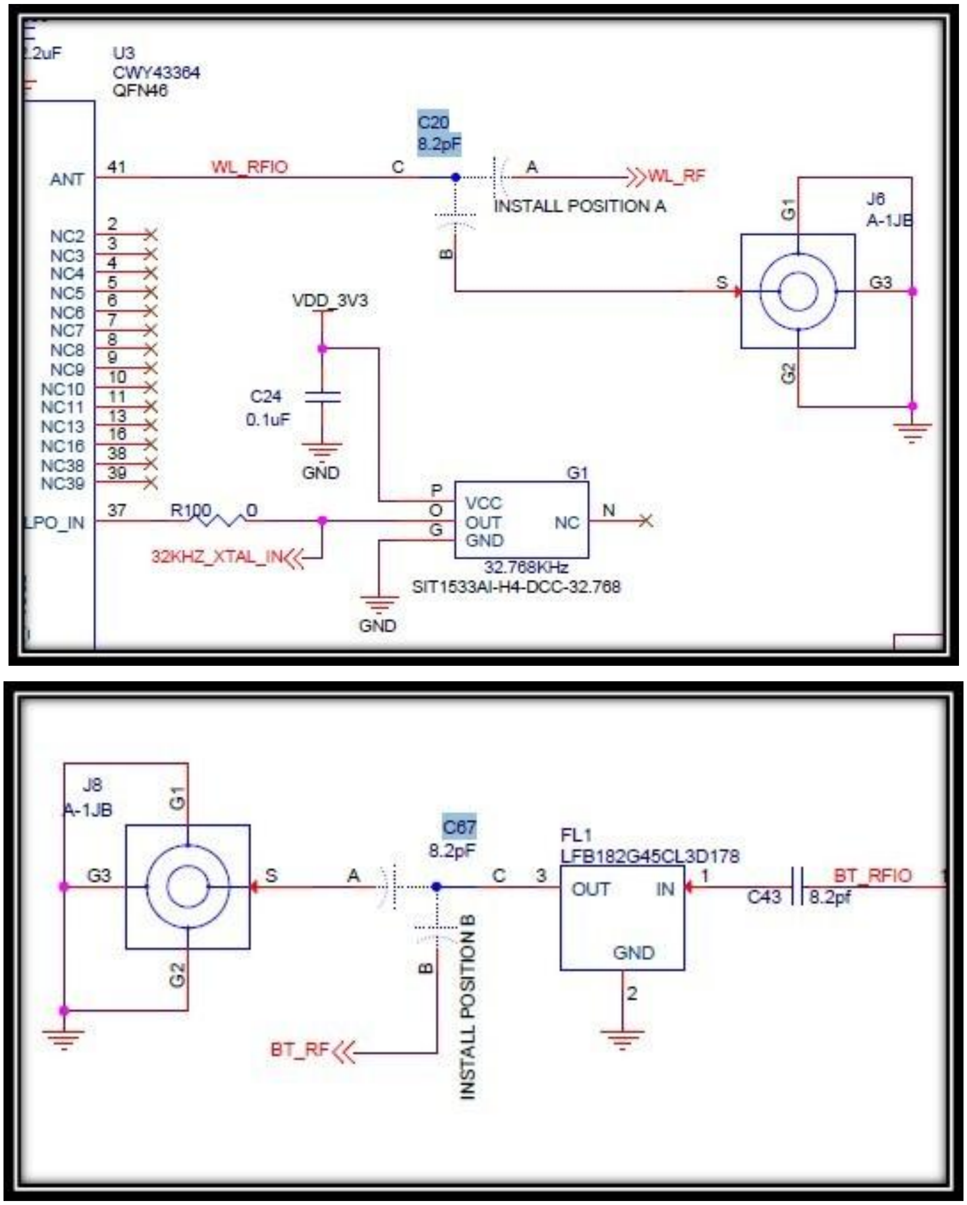


Figure 1-8. Dual Antenna Configuration Changes

Make the following change for a single shared external antenna:

- Change the **C59** Capacitor from C to B.

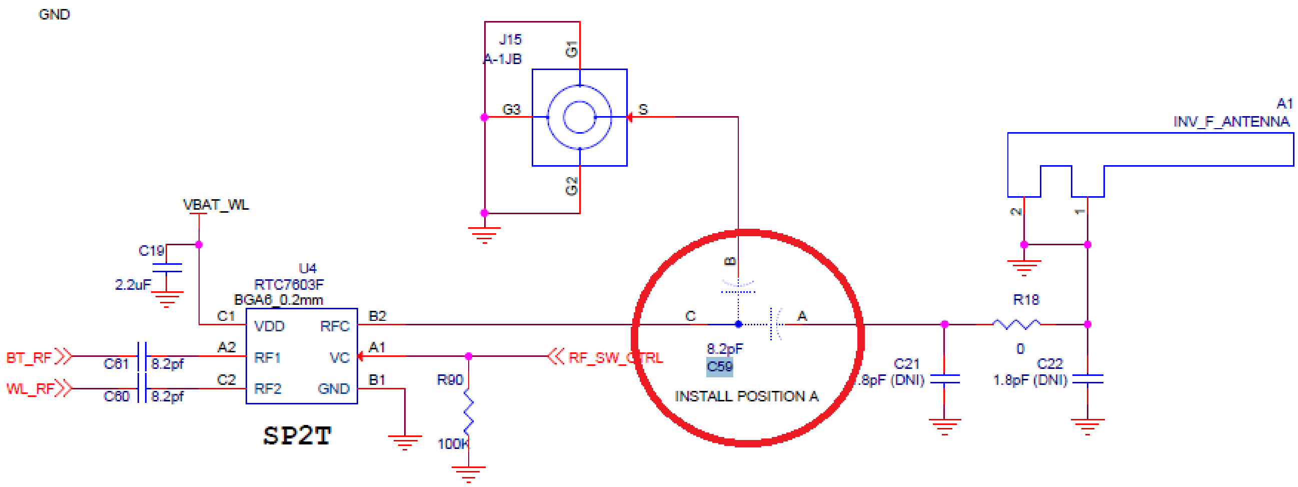


Figure 1-9. Single Shared External Antenna Configuration Changes

1.11 DIP Switch Settings

Table 1-1 indicates switch settings used for normal operation on Iwa board.

Switch	#	State	Signal Name	Description
SW5	1	ON	SPI_CLK	BT - Wi-Fi SPI interface
	2	ON	SPI_MOSI	
	3	ON	SPI_MISO	
	4	ON	SPI_CS	
	5	ON	SPI_INT	
	6	X	Unused	
SW14	1	ON	BT_GCI_IN	BT- Wi-Fi COEX lines
	2	ON	BT_GCI_OUT	
	3	ON	BT_WAKE	WLAN Host Wake
	4	X	Unused	
SW10	1	ON	P25	Button (SW 11)
	2	X	Unused	
SW9	1	ON	P29	Green LED
	2	ON	P28	Red LED
SW8	1	ON	BT_HOST_UART_RXD	BT Host UART lines for flashing
	2	ON	BT_HOST_UART_TXD	
	3	ON	BT_HOST_UART_CTS_N	
	4	ON	BT_HOST_UART_RTS_N	
SW13	1	ON	DBG_UART_RXD	BT Trace UART line for console
	2	ON	DBG_UART_TXD	
SW15	1	OFF	BT_SWDIO	JTAG line connection
	2	OFF	BT_SWDCK	
	3	ON	DBG_UART_TXD	Debug Traces Connection
	4	ON	DBG_UART_RXD	

Table 1-1. Default DIP Switch Settings on the Board/Iwa Platform

1.12 Jumper Settings

Figure 1-10 highlights the jumper settings required for normal operation on the Iwa board.

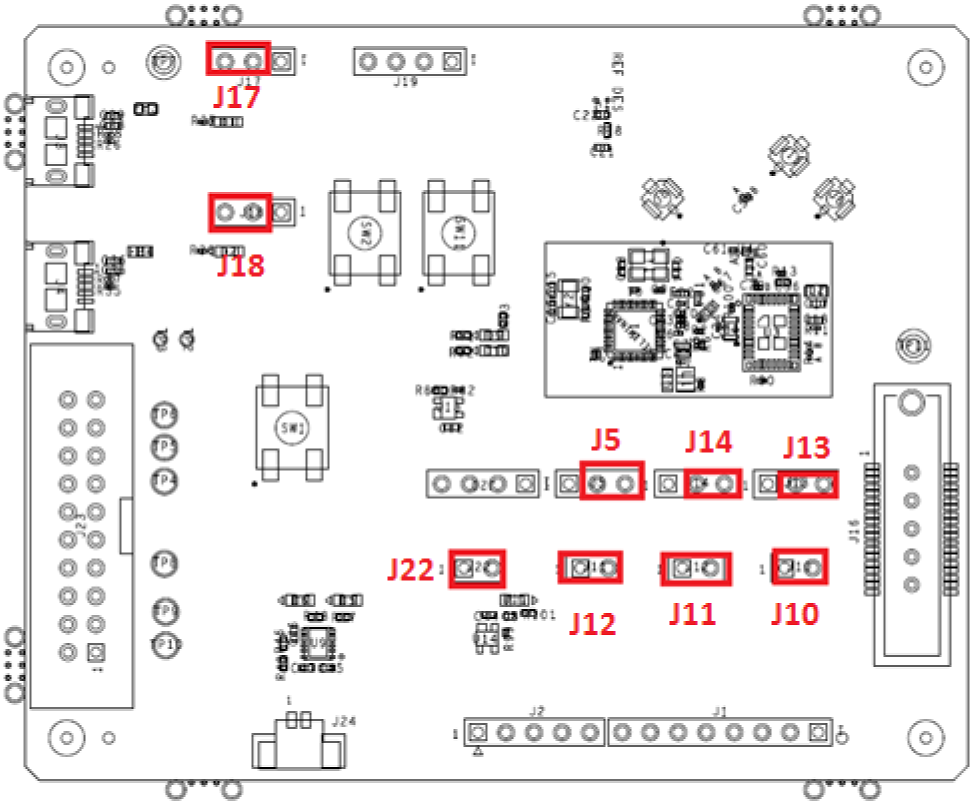


Figure 1-10. Board Layout Highlighting Jumper Settings with RED Color

2 Build and Flash Instructions

To build and flash an application for the CYW9MCU7X9N364 target, perform the following steps:

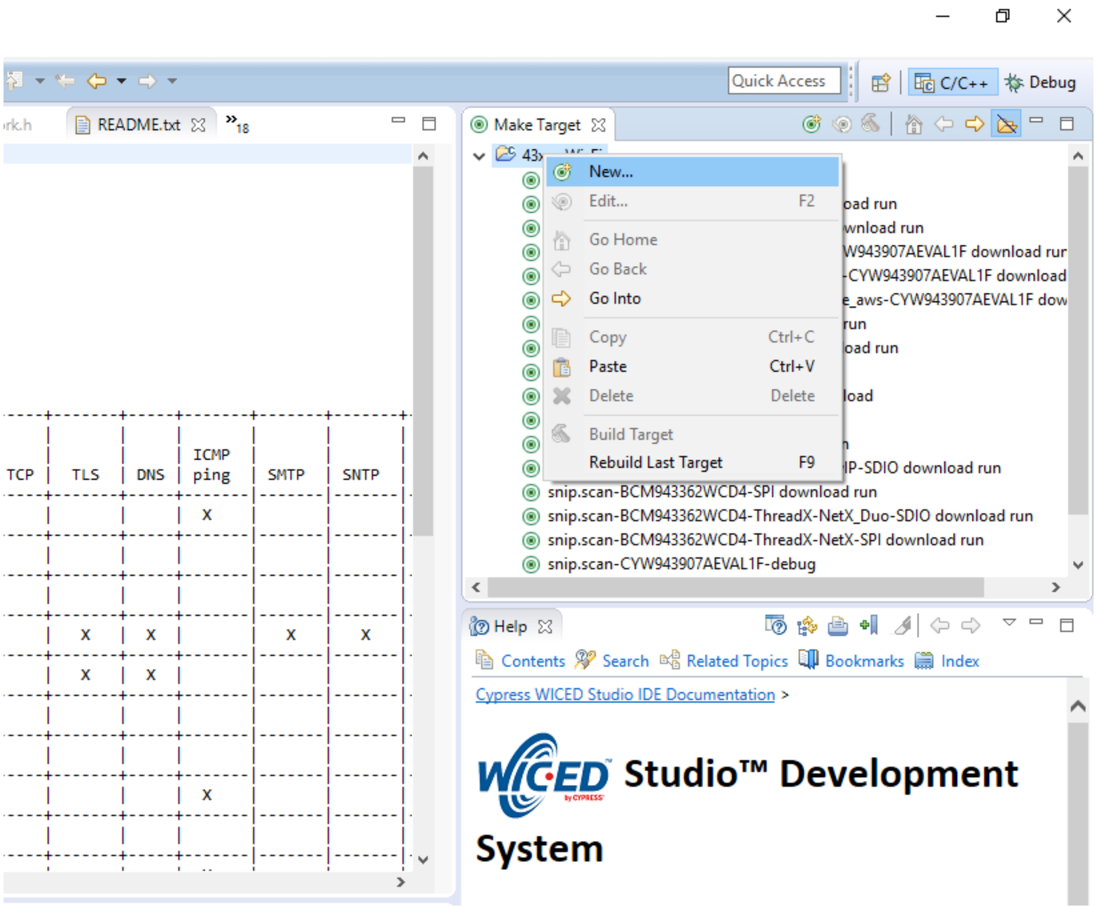

Quick Access
C/C++
Debug
README.txt
Make Target
New...
Edit... F2
Go Home
Go Back
Go Into
Copy Ctrl+C
Paste Ctrl+V
Delete Delete
Build Target
Rebuild Last Target F9
TCP
TLS
DNS
ICMP ping
SMTP
SNTP
snip.scan-BCM943362WCD4-SPI download run
snip.scan-BCM943362WCD4-ThreadX-NetX_Duo-SDIO download run
snip.scan-BCM943362WCD4-ThreadX-NetX-SPI download run
snip.scan-CYW943907AEVAL1F-debug
Help
Contents
Search
Related Topics
Bookmarks
Index
Cypress WICED Studio IDE Documentation >
WICED Studio™ Development System

Figure 2-1. WICED Studio Build

1. Connect the USB cable with the PC and device at the J3 USB port first. Note the COM port number enumerated through Device Manager as “WICED HCI UART”.
2. For collecting the traces, connect one more USB cable to J4 on the hardware and note the COM port number enumerated through Device Manager as “WICED Peripheral UART”.
3. Click **New Make Target** on the *43xxx_Wifi* project.
4. Enter `test.console-CYW9MCU7X9N364-xip BT_DEVICE_ADDRESS=101010101010 UART=COM<COMJ3> download` in the **Target name** field and click **OK**.

   This indicates the following:

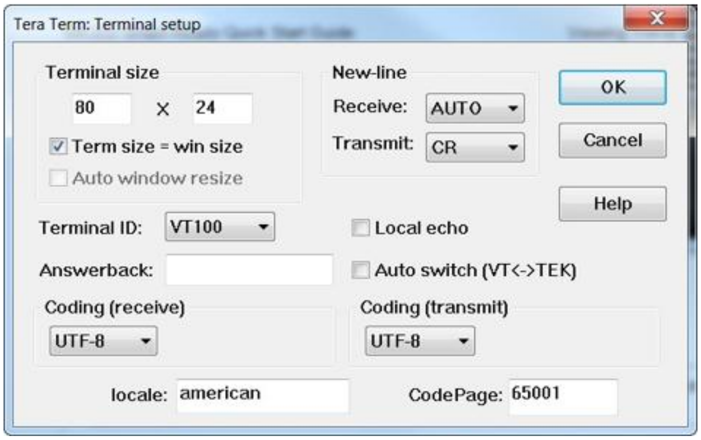
   - `test` = directory inside **apps** folder
   - `console` = subdirectory and name of the application to be built. For example, to build the *iot_gateway* application under the *demo* directory in *apps*, use `demo.iot_gateway` instead of `test.console`.
   - `CYW9MCU7X9N364`= board/platform name
   - `xip` = execute from flash (OCF)
   - `download` = indicates download to target

- `UART=COM<COMJ3>` = indicates the “WICED HCI UART” detected on the PC
- `BT_DEVICE_ADDRESS=101010101010` = Assigns a fixed BT Device address to the device (Optional)

Extra optional argument:

- `run` = Optional as device reboots automatically after download

5. Double-click (alternately, right-click and select **Build Target**) **Clean Make Target** to remove any output from the previous build. It is recommended to do Make clean when any new files are added or removed to the corresponding target.
6. Do the following to set the board in download mode before downloading the software onto the target platform:
   - Approach 1
     a. Power OFF the board.
     b. Hold down SW2.
     c. Power up the board.
     d. Release SW2.
   - Approach 2
     a. Open the “WICED HCI UART” using Tera Term/PuTTY.
     b. Reset the target using the SW1 button.
     c. Disconnect the “WICED HCI UART.”
7. Double-click (alternatively, right-click and select **Build Target**) the `test.console-CYW9MCU7X9N364 BT_DEVICE_ADDRESS=101010101010 UART=COM<COMJ3> download` make target to build and download it to the CYW9MCU7X9N364 board.
8. To view output messages with a terminal emulation program (such as Tera Term), follow these steps:
   a. Open “WICED Peripheral UART” i.e., <COMJ4>.
   b. Set baud rate to 115200.
   c. In Terminal setup, set Receive to AUTO.



Tera Term: Terminal setup

Terminal size: 80 X 24

☑ Term size = win size

□ Auto window resize

New-line

Receive: AUTO

Transmit: CR

Terminal ID: VT100

□ Local echo

Answerback:

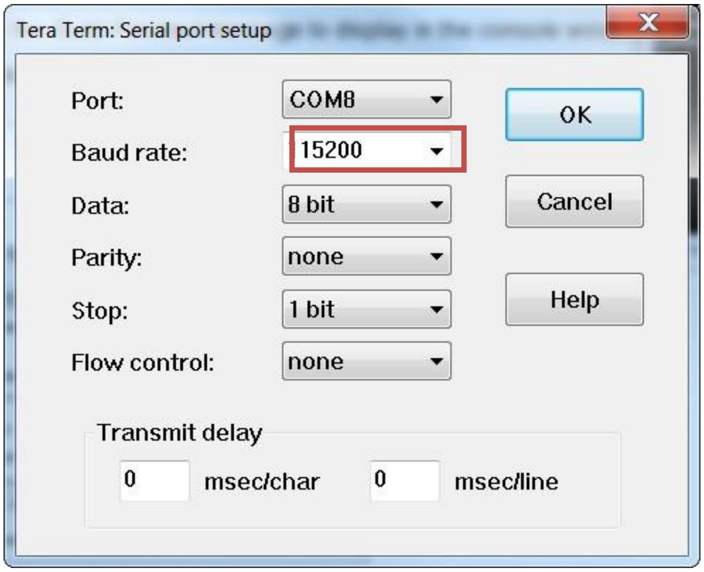
□ Auto switch (VT<->TEK)

Coding (receive): UTF-8

Coding (transmit): UTF-8

locale: american

CodePage: 65001



Tera Term: Serial port setup

Port: COM8

Baud rate: 15200

Data: 8 bit

Parity: none

Stop: 1 bit

Flow control: none

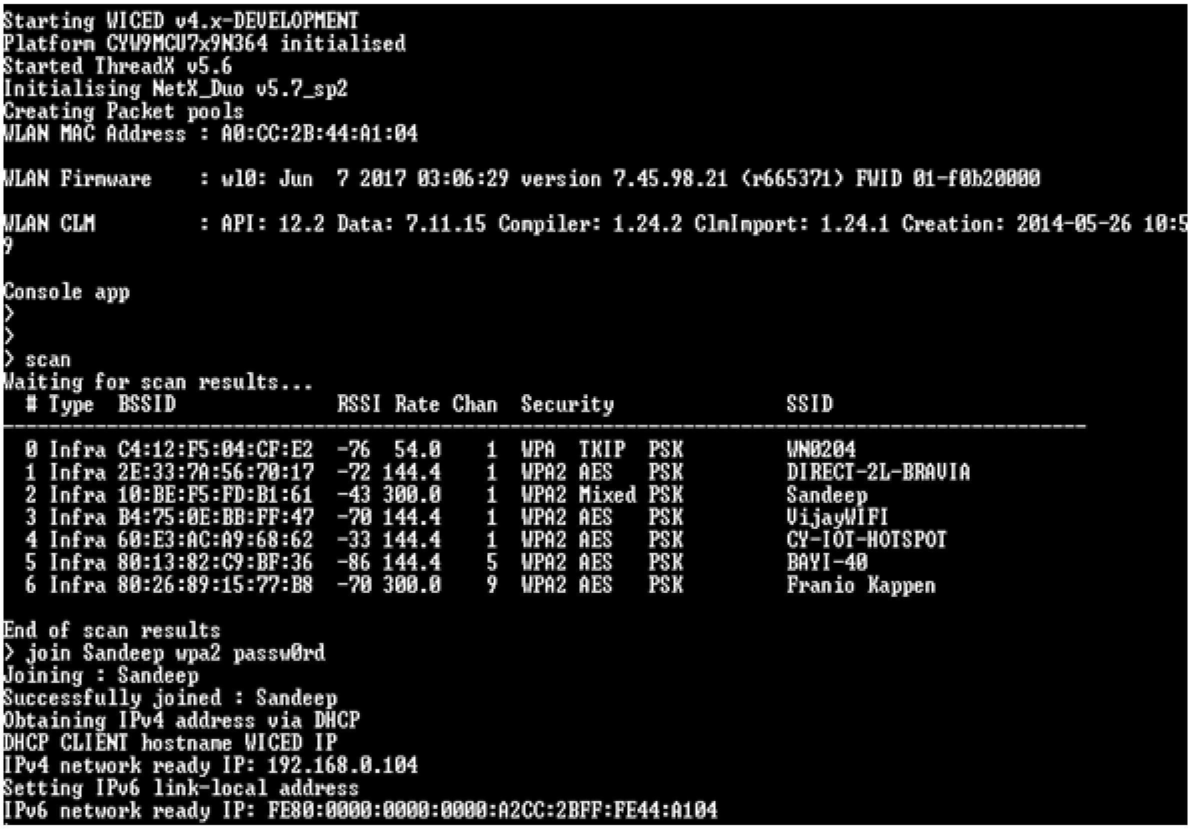
Transmit delay

0 msec/char 0 msec/line

Figure 2-2. Tera Term Settings for Logs

d. Press the **Reset** button on CYW9MCU7X9N364 to view the application start-up messages.

The output of the terminal emulation program should be similar to what is shown below:



```
Starting WICED v4.x-DEVELOPMENT
Platform CYW9MCU7x9N364 initialised
Started ThreadX v5.6
Initialising NetX_Duo v5.7_sp2
Creating Packet pools
WLAN MAC Address : A0:CC:2B:44:A1:04

WLAN Firmware    : wl0: Jun  7 2017 03:06:29 version 7.45.98.21 (r665371) FWID 01-f0b20000

WLAN CLM         : API: 12.2 Data: 7.11.15 Compiler: 1.24.2 ClmImport: 1.24.1 Creation: 2014-05-26 10:5
9

Console app
>
>
> scan
Waiting for scan results...
  # Type  BSSID             RSSI Rate Chan  Security               SSID
----------------------------------------------------------------------------------------------
  0 Infra C4:12:F5:04:CF:E2  -76  54.0    1  WPA  TKIP  PSK         WN0204
  1 Infra 2E:33:7A:56:70:17  -72 144.4    1  WPA2 AES   PSK         DIRECT-2L-BRAVIA
  2 Infra 10:BE:F5:FD:B1:61  -43 300.0    1  WPA2 Mixed PSK         Sandeep
  3 Infra B4:75:0E:BB:FF:47  -70 144.4    1  WPA2 AES   PSK         VijayWIFI
  4 Infra 60:E3:AC:A9:68:62  -33 144.4    1  WPA2 AES   PSK         CY-IOT-HOTSPOT
  5 Infra 80:13:82:C9:BF:36  -86 144.4    5  WPA2 AES   PSK         BAYI-40
  6 Infra 80:26:89:15:77:B8  -70 300.0    9  WPA2 AES   PSK         Franio Kappen

End of scan results
> join Sandeep wpa2 passw0rd
Joining : Sandeep
Successfully joined : Sandeep
Obtaining IPv4 address via DHCP
DHCP CLIENT hostname WICED IP
IPv4 network ready IP: 192.168.0.104
Setting IPv6 link-local address
IPv6 network ready IP: FE80:0000:0000:0000:A2CC:2BFF:FE44:A104
```

Figure 2-3. Logs

3 Supported WICED Demo Applications

Table 3-1 provides the list of WICED demo applications supported on the CYW9MCU7X9N364/Iwa platform.

WICED App	Description
IoT Gateway	End-to-end demo gateway app using HTTP Proxy Service (HPS) over BLE for communication with the device and HTTP/ Constrained Application Protocol (CoAP) over Wi-Fi with Cloud service.
Hps_server (BT Internet Gateway)	Demo Gateway using HTTP Proxy service for communication between device and gateway.
Restful_smart_server (BT Internet Gateway)	Demo Gateway using BLE RESTful APIs for communication between Gateway and Cloud service. The gateway and device communicate using standard GATT services.
Ble_wifi_introducer	APP to configure Wi-Fi credentials over BLE.
Coap_exosite	WICED CoAP client with Exosite cloud. Sends/receives data to/from Exosite cloud using CoAP.
Temp_control	Measures the temperature of the WICED evaluation board and displays the temperature and set point status on a local webpage
Pubsub (Aws_iot)	Demonstrates get/set states of a thing using Amazon Web Services (AWS) thing shadow over Message Queue Telemetry Transport (MQTT).
Appliance	Demonstrates how a simple web page can be used to send information to a UART terminal when a button on the webpage is clicked. The app mimics a very basic UI to control a home appliance such as a washing machine.

Table 3-1. List of WICED Demo Apps Supported on Iwa Platform

4 Execute In Place (XIP)

Iwa platform supports executing code from flash (OCF). This feature allows more free RAM for applications. It is a compile-time option that you can enable with make target options "-xip".

For example:

```
test.console-CYW9MCU7X9N364-xip BT_DEVICE_ADDRESS=101010101010 UART=COM<COMJ3> download
```

By using this option, by default, the application source will execute from flash. The code executing from flash is expected to run slower compared to RAM execution and therefore frequently executing code may need to be run from RAM for better performance. This can be configured through the linker script available at *WICED\platform\MCU\BCM920739\GCC\B1\20739B1_ocf_ram.ld*

5 Over The Air (OTA) Update

Iwa platform supports OTA updates using OCF over Wi-Fi and BLE. To understand the terminology/steps below, see basic *WICED-OTA2.pdf* [4].

5.1 Creating OTA images

Iwa platform generates three OTA images.

- **OTA application image** (*OTA2_image_file.bin*), which contains an upgradable application, dct, lut, and ota2_extract.
- **OTA file system image** (*OTA2_fs_image_file.bin*), which contains Wi-Fi firmware and other resources.
- **OTA bootloader image** (*waf.ota2_btmcu_bootloader-CYW9MCU7X9N364.ota.bin*), which contains bootloader image and Bluetooth firmware.

Do the following to generate OTA images.

Step 1: Build the ota2_extractor application using the make command:

```
snip.ota2_extract-CYW9MCU7X9N364
```

This generates *build/ota2_extract/binary/ota2_extract-CYW9MCU7X9N364.ota.bin.*

Step 2: Update the OTA target application makefile (for example, snip.scan) with the following entry:

```
OTA_APPLICATION := snip.ota2_extract-$(PLATFORM)

OTA_APP := build/$(OTA_APPLICATION)/binary/$(OTA_APPLICATION).ota.bin
```

Step 3: Create OTA images using the make command:

```
snip.scan-CYW9MCU7X9N364 ota2_image
```

This generates *OTA2_image_file.bin* and *OTA2_fs_image_file.bin* images at the *build/snip.scan-CYW9MCU7X9N364* folder.

OTA bootloader image is generated while building OTA service Application as mentioned in Section 5.2.

5.2 OTA Service Application

To demonstrate OTA, first you must flash the OTA service application onto Iwa platform.

Step 1: Change the *ota2_btmcu_example\wifi_config_dct.h* file with the appropriate AP information.

Step 2: Build and flash the application using the make command:

```
snip.ota2_btmcu_example-CYW9MCU7X9N364 ota2_apps_download UART=COM<COMJ3>
```

After successful flash, the *ota2_btmcu_example* application starts running.

This step also generates *waf.ota2_btmcu_bootloader-CYW9MCU7X9N364.ota.bin* image at build/waf.ota2_btmcu_bootloader/binary folder for updating boot loader image as mentioned in Section 5.1

5.3 Update Over Wi-Fi

Step 1: Copy the *OTA2_image_file.bin* and *OTA2_fs_image_file.bin* (generated in "Create OTA images" section above) files to the root directory of the remote web server. This step is optional if the images are already available in remote web server.

Step 2: Flash the ota2_btmcu_example application into the Iwa platform.

Step 3: From the command prompt, issue the following command to update the file system image:

```
> get_fs_update <web server ipaddress>/OTA2_fs_image_file.bin
```

Reboot the board.

Step 5: From the command prompt, issue the following command to update the application image:

```
> get_app_update <web server ipaddress>/OTA2_image_file.bin
```

Reboot the board.

The updated application is extracted and launched at the next boot.

5.4 Update/Recovery Over BLE

Update over BLE is required if Wi-Fi functionality is unavailable. Update over BLE is done mainly for the below cases:

- **Case 1:** If the file system image is corrupted (Wi-Fi firmware is part of this image).
- **Case 2:** If the OTA application is corrupted and unable to start the OTA service over internet.
- **Case 3:** If an upgrade to the bootloader application is required (Bluetooth Firmware patch is part of this application).

Iwa platform does not have enough internal flash to contain a separate factory reset image. Therefore, factory reset is supported over BLE OTA service.

Follow below steps to perform OTA over BLE service:

Step 1: Press and hold SW11 on the Iwa platform.

Step 2: Press the reset button. The red LED starts blinking.

Step 3: Keep SW11 pressed until the red LED stops blinking (this takes approximately 10 seconds).

Step 4: Iwa platform starts advertising service named "BTOTA."

Step 5: Connect to “BTOTA” from phone or PC using OTA host application [5].

Step 6: Copy the required images for upgrade to a local folder of phone or PC.

- Application image upgrade: *OTA_image_file.bin*
- File system upgrade (Wi-Fi FW): *OTA_fs_image_file.bin*
- Bootloader/Bluetooth firmware upgrade: *waf.ota2_btmcu_bootloader-CYW9MCU7X9N364.ota.bin*

Step 7: Run the OTA host application [5] on phone/PC and send these images.

Step 8: Press the reset button to reboot the Iwa platform. After reboot, updated images are used.

5.5 OTA with XIP Build

Do the following to demonstrate this feature:

Step 1: Flash the OTA + XIP images into Iwa platform.

```
./make snip.ota2_extract-CYW9MCU7X9N364
```

```
        ./make snip.ota2_btmcu_example-CYW9MCU7X9N364-xip ota2_apps_download UART=COM<COMJ3>
```

Step 2: Create the XIP OTA image and put it into the web server:

```
        ./make snip.scan-CYW9MCU7X9N364-xip ota2_image
```

Step 3: Update images over Wi-Fi or BLE service as described above.

Note: Upgrading OTA images built with XIP over a non-XIP layout (and vice versa) will not work.

6 Platform/Application Customization Options

6.1 Heap Size

You can overwrite the default HEAP size (30 KB) in the application make file as follows:

```
GLOBAL_DEFINES += PLATFORM_HEAP_SIZE=35*1024
```

6.2 Application Stack Size

You can overwrite the default application stack size (6 KB) in the application make file as follows:

```
GLOBAL_DEFINES += APPLICATION_STACK_SIZE=10*1024
```

6.3 Using printf

By default, trace implementation uses `libc printf`, which supports a wide variety of format specifiers. However, there is an option to use a cut down version of printf implementation available from BT ROM code. This version supports limited format specifiers reducing the overall application size. This can be enabled by specifying the following statement in the application make file:

```
USE_LIBC_PRINTF := 0
```

6.4 MCU Power Save

By default, platform support MCU power save feature. Applications can take advantage of this feature by calling the `wiced_platform_mcu_enable_powersave()` API.

In the same way, applications can disable power save by calling the `wiced_platform_mcu_disable_powersave()` API.

7 References

[1] CYW20719 datasheet

[2] CYW43364 datasheet: http://www.cypress.com/documentation/datasheets/cyw43364-single-chip-ieee-80211-bgn-macbasebandradio

[3] CYW9MCU7X9N364 HW Schematics: Can be found in WICED SDK at this location:

Wiced-SDK\platforms\CYW9MCU7x9N364\schematics

[4] WICED-OTA2.pdf: Can be found in WICED SDK at this location:

Wiced-SDK\doc

[5] OTA Host Application: Can be found in WICED SDK at this location:

Wiced-SDK\common\peer_apps\ota_firmware_upgrade

Document Revision History

Document Title: CYW9MCU7X9N364/Iwa Platform Quick Start Guide

Document Number: 002-20918

Revision	ECN	Issue Date	Description of Change
**	5865524	08/28/2017	CYW9MCU7X9N364 Quick Start Guide -R 1.0: Initial release
*A	5888640	09/19/2017	Added DIP Switch Settings and Jumper settings on the board Updated printf usage section
*B	5944287	10/27/2017	Board snapshot changed with new revision of board Used the new USB device descriptor names "WICED HCI UART" and "WICED Peripheral UART"
*C	6005981	01/04/2017	Updated with new features supported on the platform i.e., XIP, OTA, Power save

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 MCU

Cypress Developer Community

Community Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

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

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. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress does not assume any liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.