Please note that Cypress is an Infineon Technologies Company.

The document following this cover page is marked as "Cypress" document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content

The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers

Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

CYW943907AEVAL1F

Evaluation Kit User Guide

Document Number: 002-18703 Rev. *D

Cypress Semiconductor
An Infineon Technologies Company
198 Champion Court
San Jose, CA95134-1709
www.cypress.com, www.infineon.com

Copyrights

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 shall have no liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. CYPRESS DOES NOT REPRESENT, WARRANT, OR GUARANTEE THAT CYPRESS PRODUCTS, OR SYSTEMS CREATED USING CYPRESS PRODUCTS, WILL BE FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (collectively, “Security Breach”). Cypress disclaims any liability relating to any Security Breach, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any Security Breach. 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. “High-Risk Device” means any device or system whose failure could cause personal injury, death, or property damage. Examples of High-Risk Devices are weapons, nuclear installations, surgical implants, and other medical devices. “Critical Component” means any component of a High-Risk Device whose failure to perform can be reasonably expected to cause, directly or indirectly, the failure of the High-Risk Device, 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 any use of a Cypress product as a Critical Component in a High-Risk Device. You shall indemnify and hold Cypress, its directors, officers, employees, agents, affiliates, distributors, and assigns harmless from and against all claims, costs, damages, and expenses, arising out of any claim, including claims for product liability, personal injury or death, or property damage arising from any use of a Cypress product as a Critical Component in a High-Risk Device. Cypress products are not intended or authorized for use as a Critical Component in any High-Risk Device except to the limited extent that (i) Cypress’s published data sheet for the product explicitly states Cypress has qualified the product for use in a specific High-Risk Device, or (ii) Cypress has given you advance written authorization to use the product as a Critical Component in the specific High-Risk Device and you have signed a separate indemnification agreement.

Contents

Safety Information

The CYW943907AEVAL1F EVK 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 interfere with 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.

	The CYW943907AEVAL1F 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 CYW943907AEVAL1F 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 the user 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

CYW943907AEVAL1F 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 CYW943907AEVAL1F such as changing wires, jumper settings, or measuring voltages can cause stress on the CYW943907AEVAL1F 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 EVK is in operation.

1. Introduction

Thank you for your interest in the CYW943907AEVAL1F Evaluation Kit (EVK). The CYW943907AEVAL1F EVK enables customers to evaluate and develop single-chip Wi-Fi applications using CYW43907 devices.

The CYW943907AEVAL1F EVK uses WICED^{TM} Studio 5.0 (or later) to develop and debug your CYW43907 project. The CYW943907AEVAL1F EVK offers footprint-compatibility with Arduino shields. In addition, the kit features an RJ-45 Ethernet connector, a Micro-SD-card slot, and onboard programmer/debugger and serial bridge chip. The CYW943907AEVAL1F EVK supports only 3.3 V as the operating voltage.

Older revisions of the same kit were named BCM943907AEVAL1F_2 and BCM943907AEVAL1F. CYW43907 and BCM43907 refer to the same device.

WICED Studio 5.0 (or later) supports application development using a WICED development board (CYW943907AEVAL1F). The development system is compatible with the Windows, OS X and Linux operating systems. This document provides instructions for using peripherals, such as I^{2}C or SPI, in WICED sample applications using the WICED Studio Integrated Development Environment (IDE).

Note: This document applies to WICED Studio 5.0 (or later).

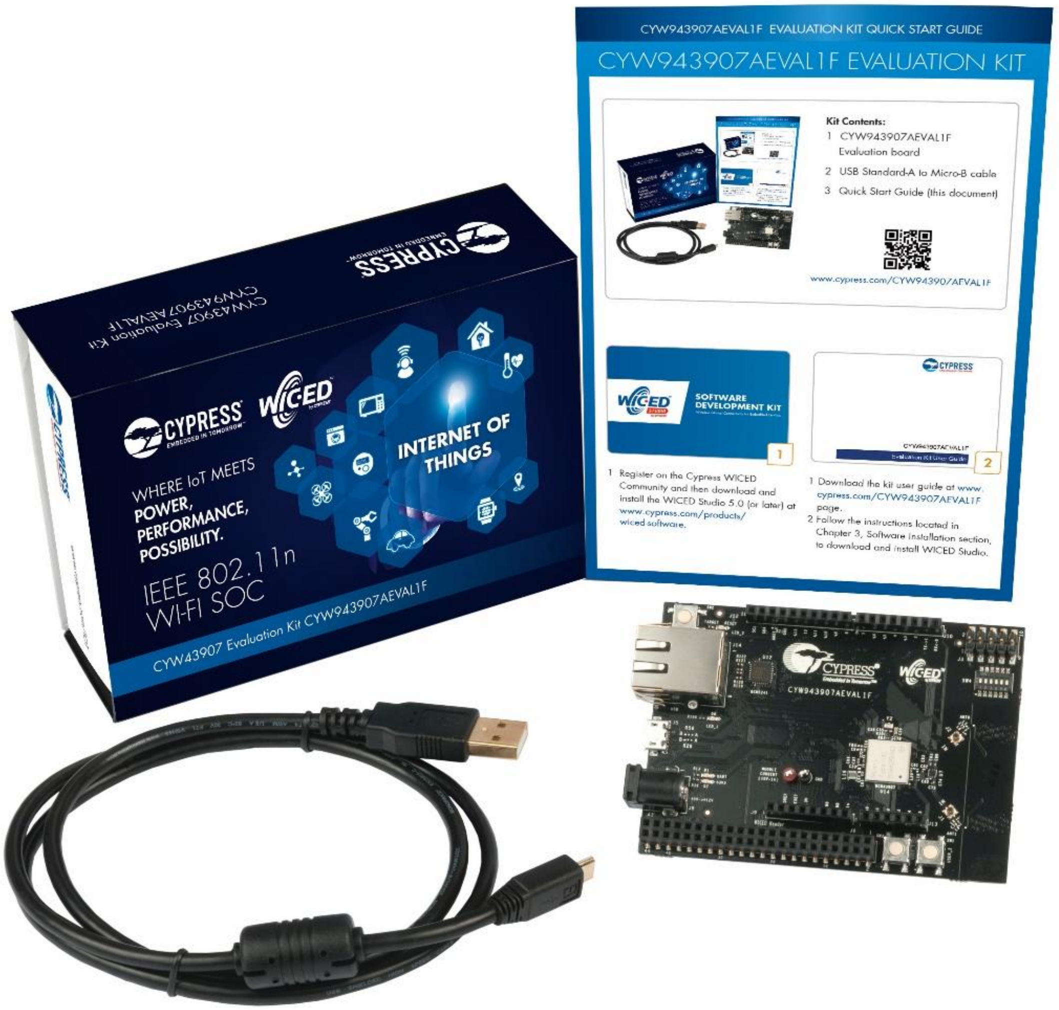
The CYW943907AEVAL1F EVK is available through the Cypress Online Store or through our distributors.

1.1 CYW943907AEVAL1F EVK Contents

The CYW943907AEVAL1F EVK includes the following:

- One CYW943907AEVAL1F Evaluation Board with assembled Arduino headers
- One USB 2.0 Type-A to Micro-B cable

Figure 1-1. CYW943907AEVAL1F Kit Contents


CYW943907AEVAL1F EVALUATION KIT QUICK START GUIDE
CYW943907AEVAL1F EVALUATION KIT
Kit Contents:
1 CYW943907AEVAL1F Evaluation board
2 USB Standard-A to Micro-B cable
3 Quick Start Guide (this document)
www.cypress.com/CYW943907AEVAL1F
WICED SOFTWARE DEVELOPMENT KIT
1 Register on the Cypress WICED Community and then download and install the WICED Studio 5.0 (or later) at www.cypress.com/products/wiced-software.
1 Download the kit user guide at www.cypress.com/CYW943907AEVAL1F page.
2 Follow the instructions located in Chapter 3, Software installation section, to download and install WICED Studio.
CYPRESS EMBEDDED IN TOMORROW
WICED
WHERE IoT MEETS POWER, PERFORMANCE, POSSIBILITY.
INTERNET OF THINGS
IEEE 802.11n WI-FI SOC
CYW43907 Evaluation Kit CYW943907AEVAL1F
CYW943907AEVAL1F

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

Hardware Not Included With the Kit

The CYW943907AEVAL1F EVK does not come with all the hardware needed to perform the demonstrations documented in this guide.

The following hardware is not included with this kit:

- RJ-45 Ethernet cable
- SD-Card
- External power supply
- Dual external antenna
- Potentiometer
- Jumper Wires

1.2 CYW943907AEVAL1F Board Details

The CYW943907AEVAL1F board consists of the blocks shown in Figure 1-2 and Figure 1-3.

1. Reset Switch (SW2)
2. RJ45 Connector (J14)
3. Micro USB (Programming and Debugging) (J5)
4. 5-12 V Power Input (J8)
5. WICED Header (J6)
6. Arduino Header (J13)
7. User Switch 1 (SW3)
8. User Switch 2 (SW1)
9. Arduino Header (J9)
10. PCB Antenna-Main (ANT1)
11. Connector for External Antenna 1 (J1)
12. CYW43907 Type 1GC Module (Murata) (U14)
13. PCB Antenna-Diversity (ANT0)
14. Connector for External Antenna 0 (J2)
15. On-board /External JTAG Switch (SW4)
16. External JTAG Header (J3)
17. Arduino Header (J10)
18. Arduino Header (J12)
19. External PHY chip(U12) – BCM5241
20. External ADC Chip (U3)
21. µSD Connector/slot (J7)

Figure 1-2. CYW943907AEVAL1F Evaluation Board

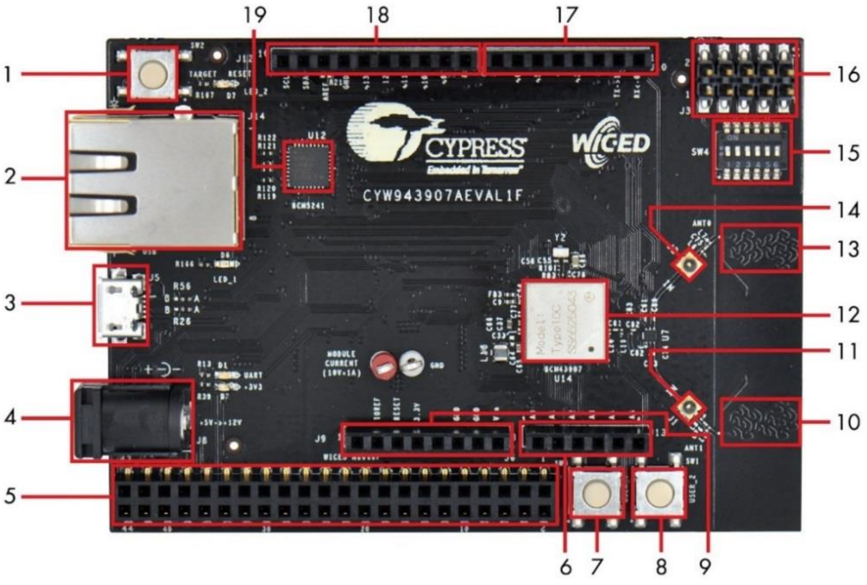
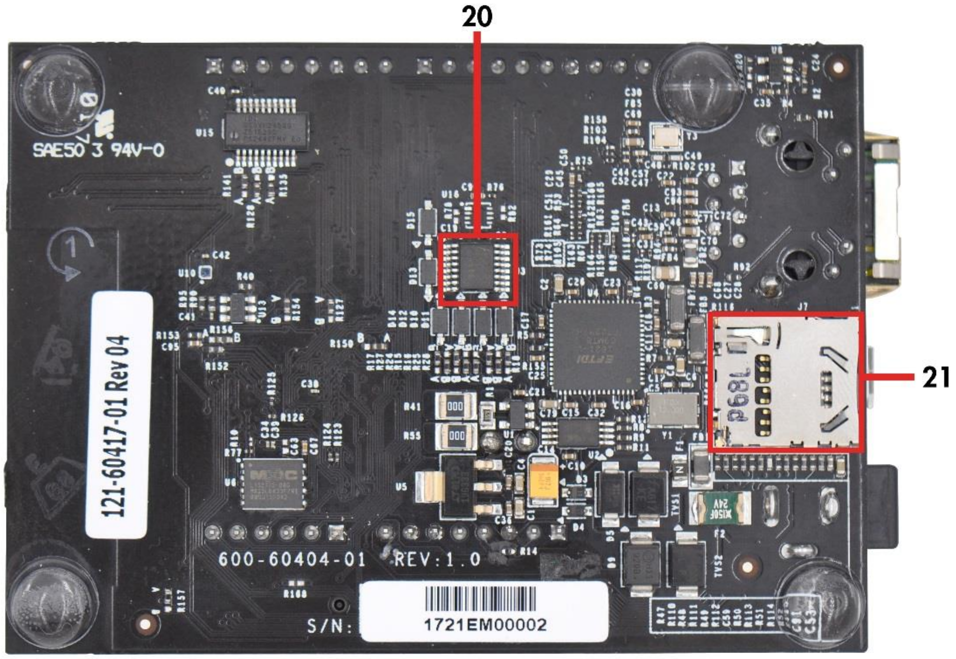


Figure 1-3. CYW943907AEVAL1F Evaluation Board (Back View)



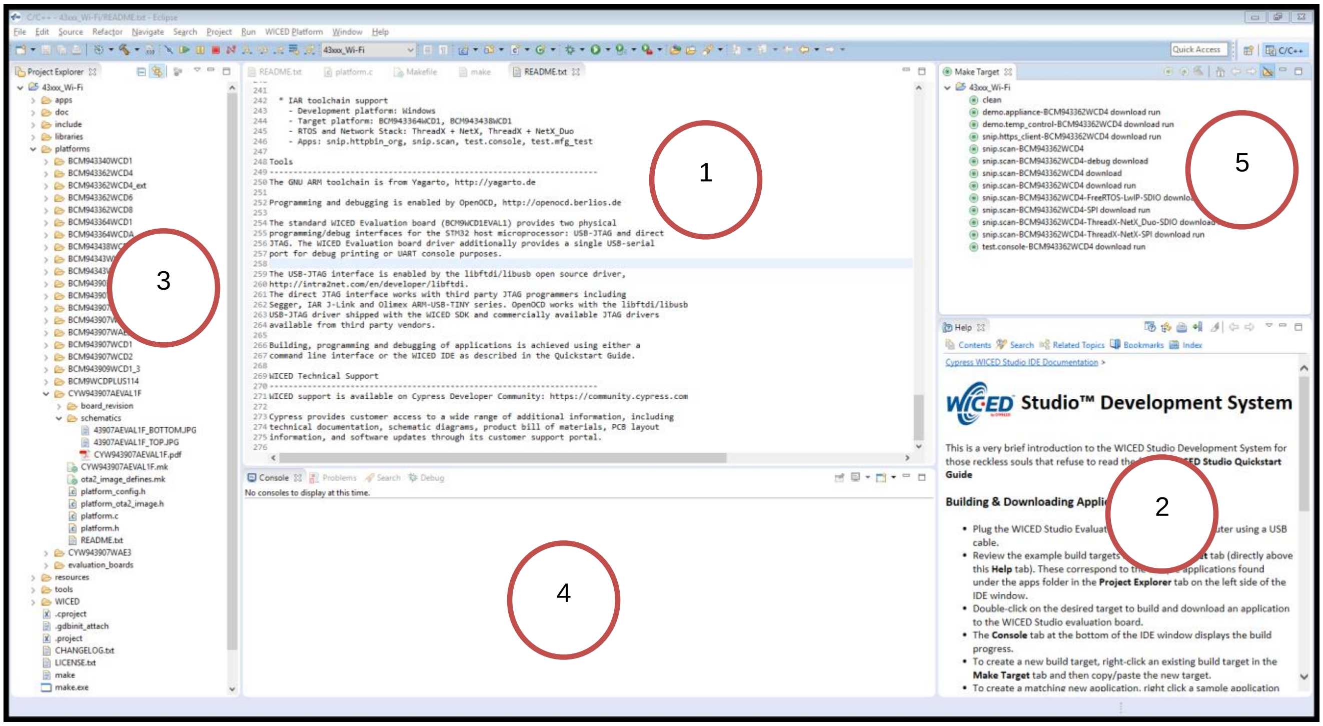
1.3 WICED Studio Development System Overview

WICED Studio 5.0 (or later) supports application development using the WICED Evaluation Board (CYW943907AEVAL1F EVK). Tabs and their location in the WICED IDE are as shown in Figure 1-4.

Figure 1-4 illustrates the following:

1. Edit your application firmware.
2. Help Window that contains instructions on building and downloading applications.
3. Explore existing applications/firmware and library of the Software Development Kit (SDK).
4. View Build messages in the Console window.
5. Create and edit Make Targets for the platform to build your Application/Project.

Figure 1-4. WICED IDE



```
 * IAR toolchain support
   - Development platform: Windows
   - Target platform: BCM943364WCD1, BCM943438WCD1
   - RTOS and Network Stack: ThreadX + NetX, ThreadX + NetX_Duo
   - Apps: snip.httpbin_org, snip.scan, test.console, test.mfg_test

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

Programming and debugging is enabled by OpenOCD, http://openocd.berlios.de

The standard WICED Evaluation board (BCM9WCD1EVAL1) provides two physical
programming/debug interfaces for the STM32 host microprocessor: USB-JTAG and direct
JTAG. The WICED Evaluation board driver additionally provides a single USB-serial
port for debug printing or UART console purposes.

The USB-JTAG interface is enabled by the libftdi/libusb open source driver,
http://intra2net.com/en/developer/libftdi.
The direct JTAG interface works with third party JTAG programmers including
Segger, IAR J-Link and Olimex ARM-USB-TINY series. OpenOCD works with the libftdi/libusb
USB-JTAG driver shipped with the WICED SDK and commercially available JTAG drivers
available from third party vendors.

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

WICED Technical Support
---------------------------------------------------------------------
WICED support is available on Cypress Developer Community: https://community.cypress.com

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 through its customer support portal.
```

No consoles to display at this time.

- clean
- demo.appliance-BCM943362WCD4 download run
- demo.temp_control-BCM943362WCD4 download run
- snip.https_client-BCM943362WCD4 download run
- snip.scan-BCM943362WCD4
- snip.scan-BCM943362WCD4-debug download
- snip.scan-BCM943362WCD4 download
- snip.scan-BCM943362WCD4 download run
- snip.scan-BCM943362WCD4-FreeRTOS-LwIP-SDIO downlo
- snip.scan-BCM943362WCD4-SPI download run
- snip.scan-BCM943362WCD4-ThreadX-NetX_Duo-SDIO download
- snip.scan-BCM943362WCD4-ThreadX-NetX-SPI download run
- test.console-BCM943362WCD4 download run

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 **ED Studio Quickstart Guide**

Building & Downloading Applic

- Plug the WICED Studio Evaluat uter using a USB cable.
- Review the example build targets **t** tab (directly above this **Help** tab). These correspond to th applications found under the apps folder in the **Project Explorer** tab on the left side of the IDE window.
- Double-click on the desired target to build and download an application to the WICED Studio evaluation board.
- The **Console** tab at the bottom of the IDE window displays the build progress.
- To create a new build target, right-click an existing build target in the **Make Target** tab and then copy/paste the new target.
- To create a matching new application, right click a sample application

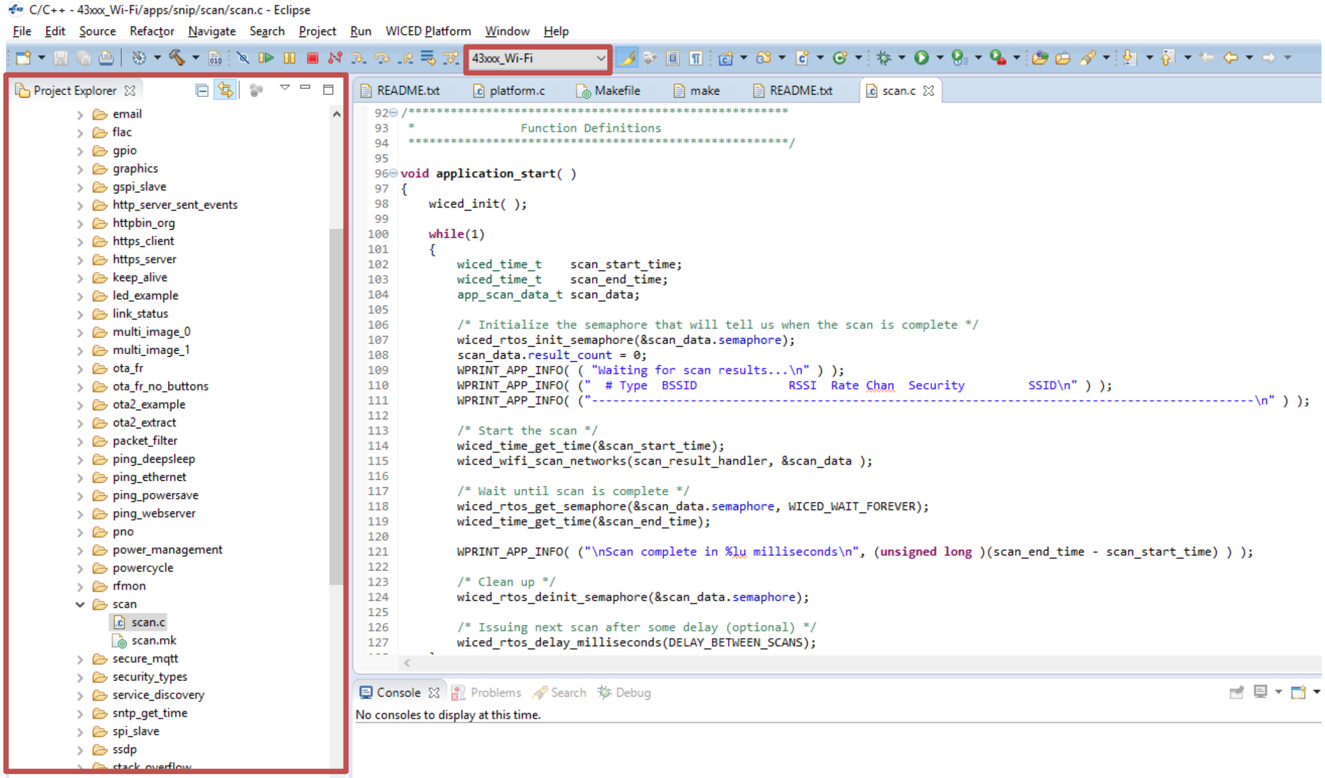
1.4 WICED Studio Code Examples

WICED Studio includes libraries and code examples supporting both Bluetooth and Wi-Fi platforms. Selecting the 43xxx_Wi-fi Filter will show only Wi-Fi platform related files in the project explorer as shown in Figure 1-5.

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 1-6. Code examples under `apps` are further grouped into `demo`, `snip`, `test`, `waf` (WICED Application Framework), and `wwd` (WICED Wi-Fi Driver Application) directories.

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 `test` directory contains applications that are used for simple test and utility. The `waf` directory contains applications that are part of the WICED Application framework, for example, the bootloader. The `wwd` directory contains applications that are developed using the low-level `wwd` API calls and do not rely on higher level WICED APIs. Located within each subdirectory in the apps folder is a `README.txt` that lists and summarizes the applications located within the folder. It should also be noted that not all applications are supported in all platforms. The snip directory contains a `README.txt` with a matrix on what applications are supported in what platforms. For more details on the WICED software stack and APIs, review the application notes and documents available in the doc folder <*WICED SDK installation folder>/WICED-Studio-5.0/43xxx_Wi-Fi/doc*. WICED-QSG204, available in the same path, is a good document to start with.

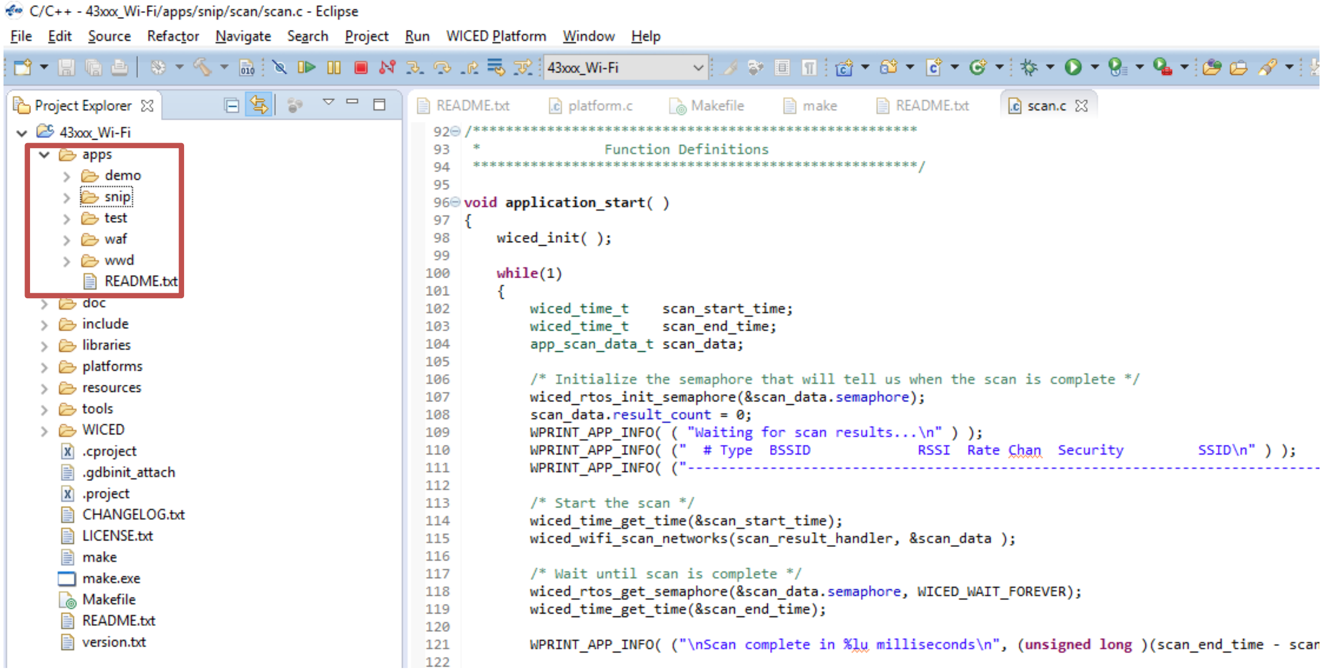
Figure 1-5. Filter for Wi-Fi Code Example in WICED Studio



```
/*****************************************************
 *               Function Definitions
 ******************************************************/

void application_start( )
{
    wiced_init( );

    while(1)
    {
        wiced_time_t    scan_start_time;
        wiced_time_t    scan_end_time;
        app_scan_data_t scan_data;

        /* Initialize the semaphore that will tell us when the scan is complete */
        wiced_rtos_init_semaphore(&scan_data.semaphore);
        scan_data.result_count = 0;
        WPRINT_APP_INFO( ( "Waiting for scan results...\n" ) );
        WPRINT_APP_INFO( ("  # Type  BSSID             RSSI  Rate Chan  Security         SSID\n" ) );
        WPRINT_APP_INFO( ("----------------------------------------------------------------------------------------------\n" ) );

        /* Start the scan */
        wiced_time_get_time(&scan_start_time);
        wiced_wifi_scan_networks(scan_result_handler, &scan_data );

        /* Wait until scan is complete */
        wiced_rtos_get_semaphore(&scan_data.semaphore, WICED_WAIT_FOREVER);
        wiced_time_get_time(&scan_end_time);

        WPRINT_APP_INFO( ("\nScan complete in %lu milliseconds\n", (unsigned long )(scan_end_time - scan_start_time) ) );

        /* Clean up */
        wiced_rtos_deinit_semaphore(&scan_data.semaphore);

        /* Issuing next scan after some delay (optional) */
        wiced_rtos_delay_milliseconds(DELAY_BETWEEN_SCANS);
```

No consoles to display at this time.

Figure 1-6. Code Examples under *apps* Category



```
/*****************************************************
 *               Function Definitions
 ******************************************************/

void application_start( )
{
    wiced_init( );

    while(1)
    {
        wiced_time_t    scan_start_time;
        wiced_time_t    scan_end_time;
        app_scan_data_t scan_data;

        /* Initialize the semaphore that will tell us when the scan is complete */
        wiced_rtos_init_semaphore(&scan_data.semaphore);
        scan_data.result_count = 0;
        WPRINT_APP_INFO( ( "Waiting for scan results...\n" ) );
        WPRINT_APP_INFO( ("  # Type  BSSID             RSSI  Rate Chan  Security         SSID\n" ) );
        WPRINT_APP_INFO( ("------------------------------------------------------------------------

        /* Start the scan */
        wiced_time_get_time(&scan_start_time);
        wiced_wifi_scan_networks(scan_result_handler, &scan_data );

        /* Wait until scan is complete */
        wiced_rtos_get_semaphore(&scan_data.semaphore, WICED_WAIT_FOREVER);
        wiced_time_get_time(&scan_end_time);

        WPRINT_APP_INFO( ("\nScan complete in %lu milliseconds\n", (unsigned long )(scan_end_time - scar
```

1.5 Kit Code Examples

In addition to the examples available in WICED Studio, this EVK includes a few additional code examples, which can be used to quickly evaluate CYW43907 using this kit. These examples are described in the Code Examples chapter.

1.6 Getting Started

To learn quickly about CYW943907AEVAL1F EVK, refer to the CYW943907AEVAL1F Quick Start Guide inside the kit box.

This user guide will help you get acquainted with the CYW943907AEVAL1F EVK:

- The Software Installation chapter describes the installation of the kit software. This includes extracting the required files for WICED Studio 5.0 (or later).
- The Kit Operation chapter describes the major sections of the kit such as the on-board programmer/debugger chip, reset control, headers, programming and debugging of the kit, SD card interface, and Ethernet interface.
- The Hardware chapter describes the CYW943907AEVAL1F EVK hardware and its different blocks.
- The Code Examples chapter describes code examples that will help you understand how to get started with WLAN basic examples.

1.7 IOT Resources and Technical Support

Cypress provides a wealth of data at www.cypress.com/internet-things-iot to help you 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 (https://community.cypress.com).

For assistance, go to: www.cypress.com/support, or contact our customer support at +1(800) 541-4736 Ext. 2 (in the USA), or +1 (408) 943-2600 Ext. 2 (International).

1.8 Additional Learning Resources

Visit CYW943907AEVAL1F EVK and CYW43907 for additional learning resources including datasheets and application notes.

1.9 Document Conventions

Table 1-1. Document Conventions for Guides

Convention	Usage
Italics	Displays file locations, user entered text, and source code: C:\ ...cd\icc\
Italics	Displays file names and reference documentation.
[**Bracketed, Bold**]	Displays keyboard commands in procedures: [**Enter**] or [**Ctrl**] [**C**]
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.

1.10 Acronyms

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

Acronym	Definition
SPI	Serial Peripheral Interface
EVK	Evaluation Kit
SDK	Software Development Kit
WICED	Wireless Internet Connectivity for Embedded Devices
JTAG	Joint Test Action Group
I^{2}C	Inter-Integrated Circuit
MQTT	Message Queue Telemetry Transport
POR	Power-on-Reset
PMU	Power Management Unit
VTRIM	Voltage Trimming
LPO	Low Power Oscillator
GPIO	General Purpose Input Output
UART	Universal Asynchronous Receiver/Transmitter
AWS	Amazon Web Services
IDE	Integrated Development Environment
WLAN	Wireless Local Area Network

2. Software Installation

This chapter describes the steps to install the software tools and packages on a PC for using the CYW943907AEVAL1F EVK. This includes the WICED IDE in which the projects will be built and used for programming.

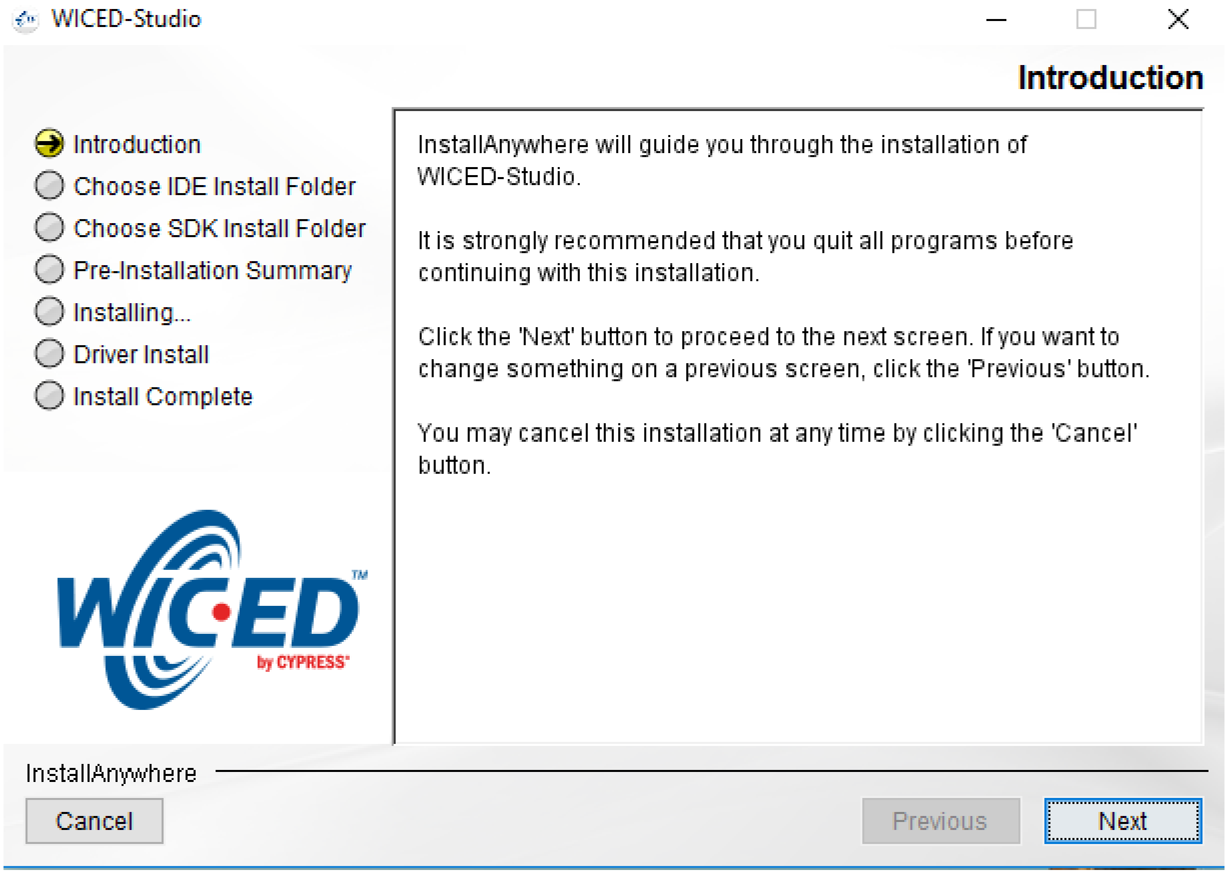
2.1 Before You Begin

All Cypress software installations require administrator privileges. Ensure 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. Ensure you have installed WICED Studio 5.0 (or later).

2.2 Install Software

Follow these steps to install the CYW943907AEVAL1F Evaluation Kit software:

1. Download and install WICED Studio 5.0 (or later) from this web page. Following is a screenshot of the Installer Window when opened.



WICED-Studio

Introduction

➔ Introduction
○ Choose IDE Install Folder
○ Choose SDK Install Folder
○ Pre-Installation Summary
○ Installing...
○ Driver Install
○ Install Complete

InstallAnywhere will guide you through the installation of WICED-Studio.

It is strongly recommended that you quit all programs before continuing with this installation.

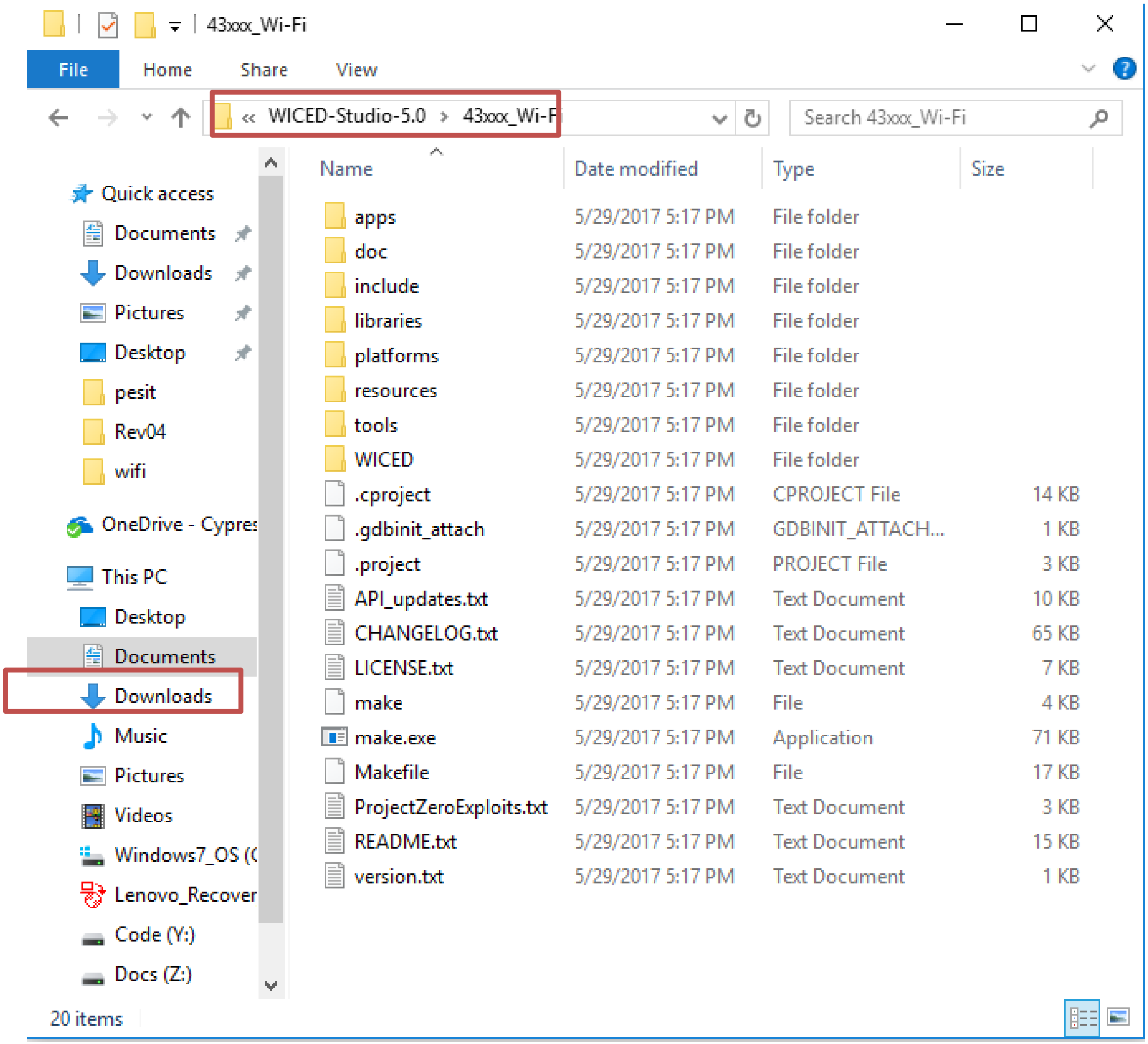
Click the 'Next' button to proceed to the next screen. If you want to change something on a previous screen, click the 'Previous' button.

You may cancel this installation at any time by clicking the 'Cancel' button.

2. Select two folders, one for the IDE and the other for the SDK. The folder for the SDK contains the Framework for developing Wi-Fi applications.

3. As a last step in installation, the installer will ask to select between the Wi-Fi and Bluetooth platforms. Select **43xxx_Wi-Fi** as default.
4. Download the *CY943907AEVAL1F_KitPackage.zip* software from here. The software is available as a zip file.
5. Locate the WICED Wi-Fi-SDK directory in your PC. The default location is *C:\Users\<user name>\Documents\WICED-Studio-5.0\43xxx_Wi-Fi*, as shown in Figure 2-1. However, it may be in a different location depending on the path you choose when installing WICED Studio.

Figure 2-1. WICED SDK Directory

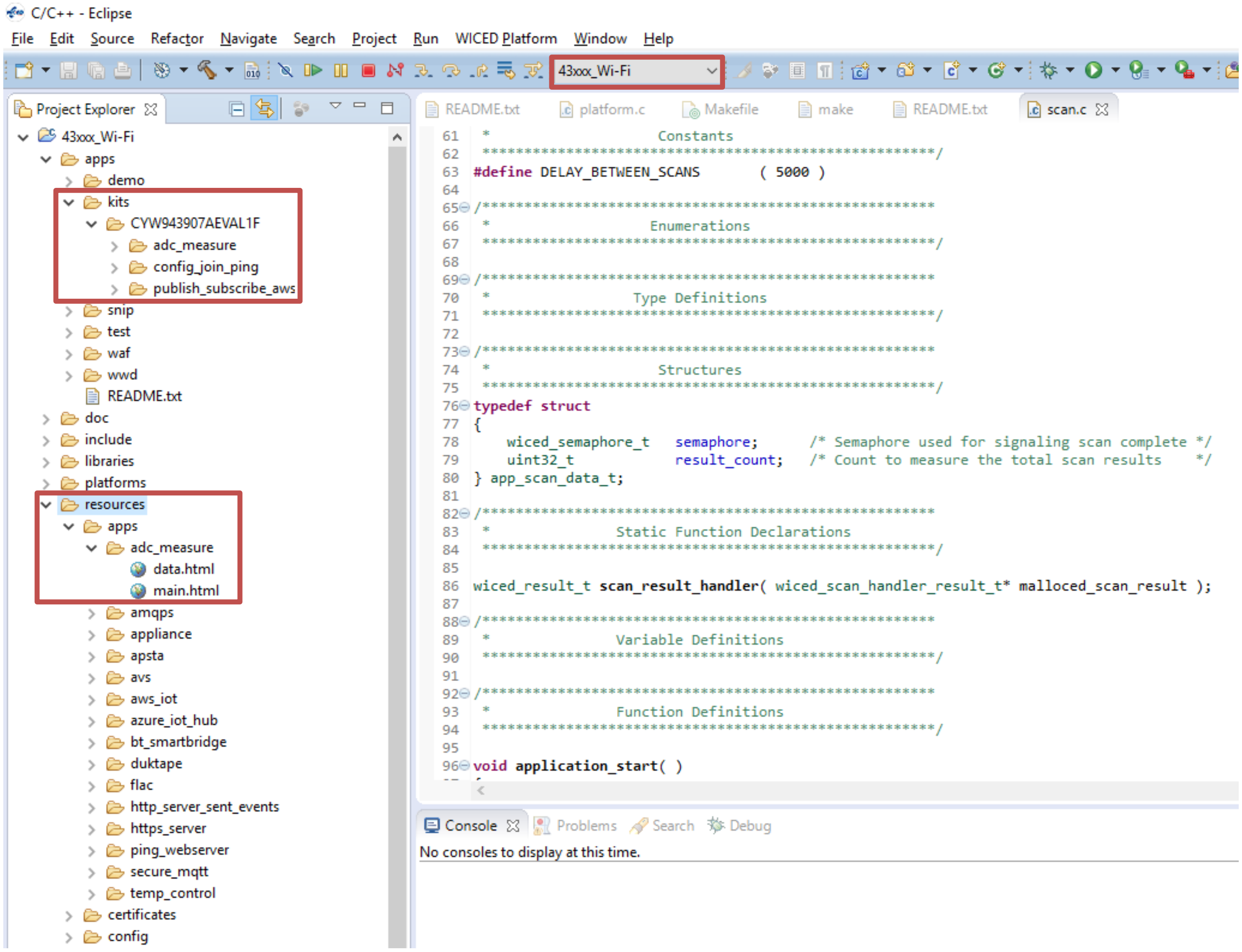


Name	Date modified	Type	Size
apps	5/29/2017 5:17 PM	File folder	
doc	5/29/2017 5:17 PM	File folder	
include	5/29/2017 5:17 PM	File folder	
libraries	5/29/2017 5:17 PM	File folder	
platforms	5/29/2017 5:17 PM	File folder	
resources	5/29/2017 5:17 PM	File folder	
tools	5/29/2017 5:17 PM	File folder	
WICED	5/29/2017 5:17 PM	File folder	
.cproject	5/29/2017 5:17 PM	CPROJECT File	14 KB
.gdbinit_attach	5/29/2017 5:17 PM	GDBINIT_ATTACH...	1 KB
.project	5/29/2017 5:17 PM	PROJECT File	3 KB
API_updates.txt	5/29/2017 5:17 PM	Text Document	10 KB
CHANGELOG.txt	5/29/2017 5:17 PM	Text Document	65 KB
LICENSE.txt	5/29/2017 5:17 PM	Text Document	7 KB
make	5/29/2017 5:17 PM	File	4 KB
make.exe	5/29/2017 5:17 PM	Application	71 KB
Makefile	5/29/2017 5:17 PM	File	17 KB
ProjectZeroExploits.txt	5/29/2017 5:17 PM	Text Document	3 KB
README.txt	5/29/2017 5:17 PM	Text Document	15 KB
version.txt	5/29/2017 5:17 PM	Text Document	1 KB

6. Copy the *CY943907AEVAL1F_KitPackage.zip* file and extract to a temporary location such as "temp". The zip file will extract two directories called "apps" and "resources" inside the *temp/CYW943907AEVAL1F_KitPackage/* directory. Select both of them, Copy (**CTRL+C**) and paste (**CTRL+V**) into *C:\Users\<user name>\Documents\WICED-Studio-5.0\43xxx_Wi-Fi.* Choose the option to merge with existing folders.

   Alternately, copy the *CY943907AEVAL1F_KitPackage.zip* to the location specified above and use the **Extract Here** option if you have 7-Zip or other unzip utility. The zip file should be merged to the existing folders. If WICED Studio 5.0 (or later) is opened with *43xxx_Wi-Fi* as the WICED Filter (Figure 1-5), then the new folders appear as shown in Figure 2-2.

Figure 2-2. Setup Package in WICED Studio 5.0 (or later)



```
43xxx_Wi-Fi
  apps
    demo
    kits
      CYW943907AEVAL1F
        adc_measure
        config_join_ping
        publish_subscribe_aws
    snip
    test
    waf
    wwd
    README.txt
  doc
  include
  libraries
  platforms
  resources
    apps
      adc_measure
        data.html
        main.html
      amqps
      appliance
      apsta
      avs
      aws_iot
      azure_iot_hub
      bt_smartbridge
      duktape
      flac
      http_server_sent_events
      https_server
      ping_webserver
      secure_mqtt
      temp_control
    certificates
    config
```

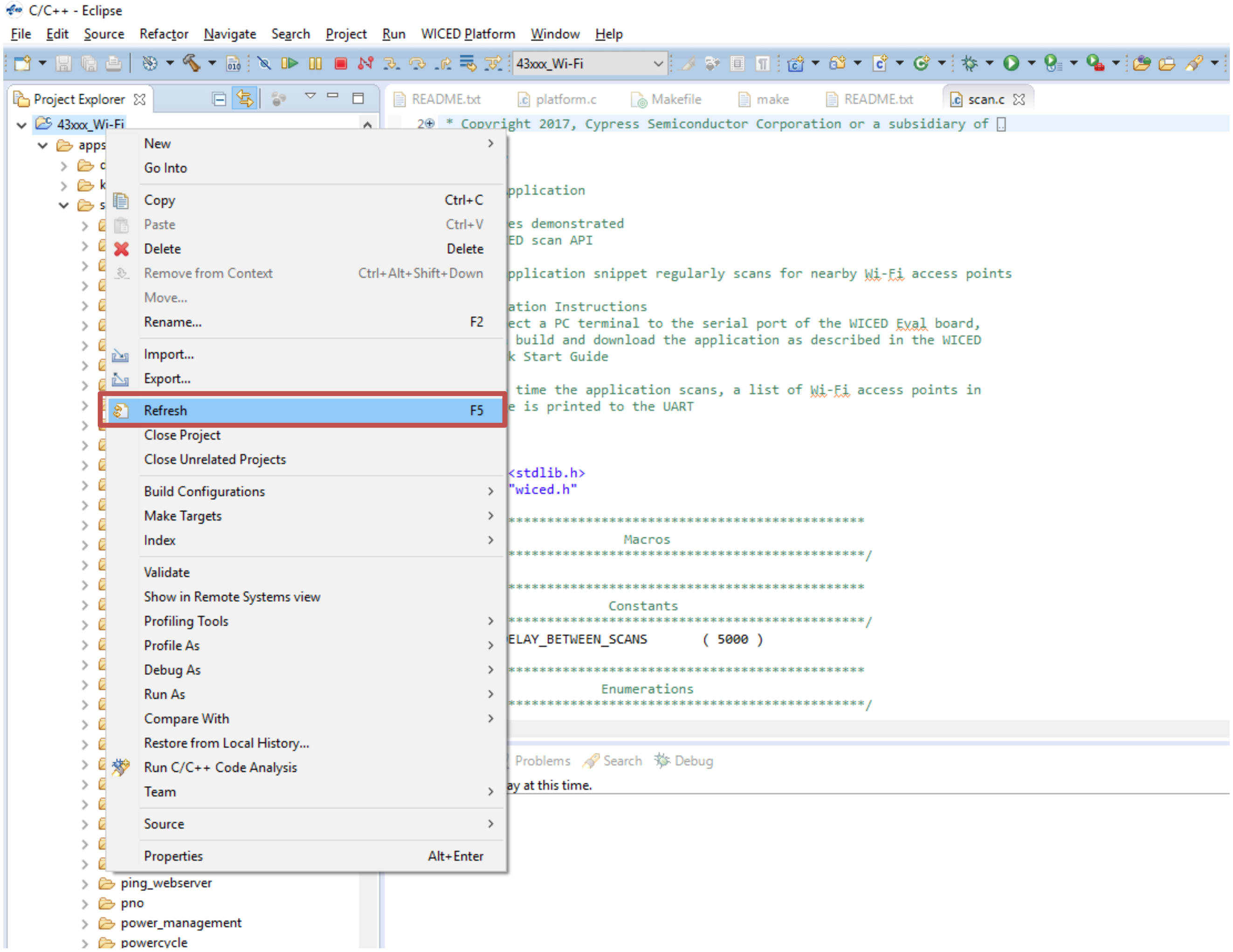
```
 *                    Constants
 ******************************************************/
#define DELAY_BETWEEN_SCANS       ( 5000 )

/******************************************************
 *                   Enumerations
 ******************************************************/

/******************************************************
 *                 Type Definitions
 ******************************************************/

/******************************************************
 *                    Structures
 ******************************************************/
typedef struct
{
    wiced_semaphore_t   semaphore;      /* Semaphore used for signaling scan complete */
    uint32_t            result_count;   /* Count to measure the total scan results    */
} app_scan_data_t;

/******************************************************
 *               Static Function Declarations
 ******************************************************/

wiced_result_t scan_result_handler( wiced_scan_handler_result_t* malloced_scan_result );

/******************************************************
 *               Variable Definitions
 ******************************************************/

/******************************************************
 *               Function Definitions
 ******************************************************/

void application_start( )
```

No consoles to display at this time.

7. The *CY943907AEVAL1F_KitPackage.zip* package contains three code examples, which add to the existing set of examples available in WICED Studio 5.0 or later. Unzipping creates the kits directory under apps, and *adc_measure* in the *resources\apps* directory.

   After unzipping, if the projects are not visible in WICED Studio 5.0 (or later), then right-click the topmost folder (*43xxx_Wi-Fi*) and click **Refresh**, as shown in Figure 2-3.

Figure 2-3. **Refresh** Top Folder



```
 * Copyright 2017, Cypress Semiconductor Corporation or a subsidiary of

pplication

es demonstrated
ED scan API

pplication snippet regularly scans for nearby Wi-Fi access points

ation Instructions
ect a PC terminal to the serial port of the WICED Eval board,
 build and download the application as described in the WICED
k Start Guide

 time the application scans, a list of Wi-Fi access points in
e is printed to the UART

<stdlib.h>
"wiced.h"

*************************************************
               Macros
*************************************************/

*************************************************
             Constants
*************************************************/
ELAY_BETWEEN_SCANS       ( 5000 )

*************************************************
            Enumerations
*************************************************/
```

3. Kit Operation

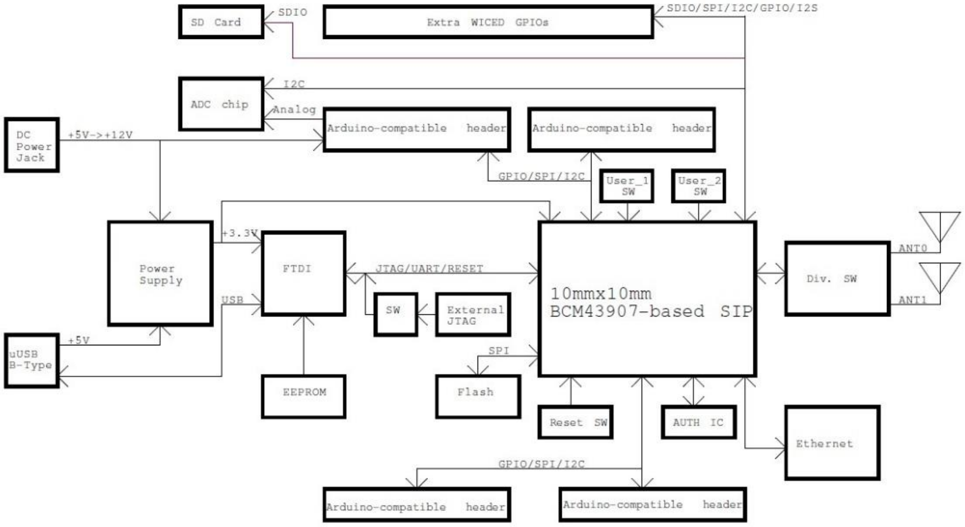
This chapter introduces you to the CYW943907AEVAL1F EVK and the features that will be used as part of the kit operation. Features such as Wi-Fi connection and programming/debugging are discussed in this chapter. The chapter also describes the USB-UART that can be used to communicate with the CYW43907 device on this EVK.

3.1 Theory of Operation

Figure 3-1 illustrates the block diagram of the CYW943907AEVAL1F EVK. This board contains BCM943907/CYW43907 based SiP, which is a Type1GC Wireless module. This module is an embedded network controller solution from Murata. This board also contains a USB-Serial interface, JTAG programmer, and a debugger.

This board features Arduino form-factor compatible headers, which enable Arduino shields to be plugged on top, extending its capabilities. This board also features two user switches, two user LEDs, an RJ-45 connector for Ethernet, and a reset switch for the wireless module.

Figure 3-1. Block Diagram of CYW943907AEVAL1F EVK



3.2 On-board programmer/Debugger and Serial interface Chip

An FT-2232-HQ chip is used for onboard programming, debugging, and USB-Serial functionality. It connects to the computer over a USB interface and connects to the CYW43907-based SiP module over JTAG and UART pins. Alternately, you can use the External JTAG connector (J3) along with switch **SW4** (in all closed positions) in order to use JTAG from connectors such as Olimex.

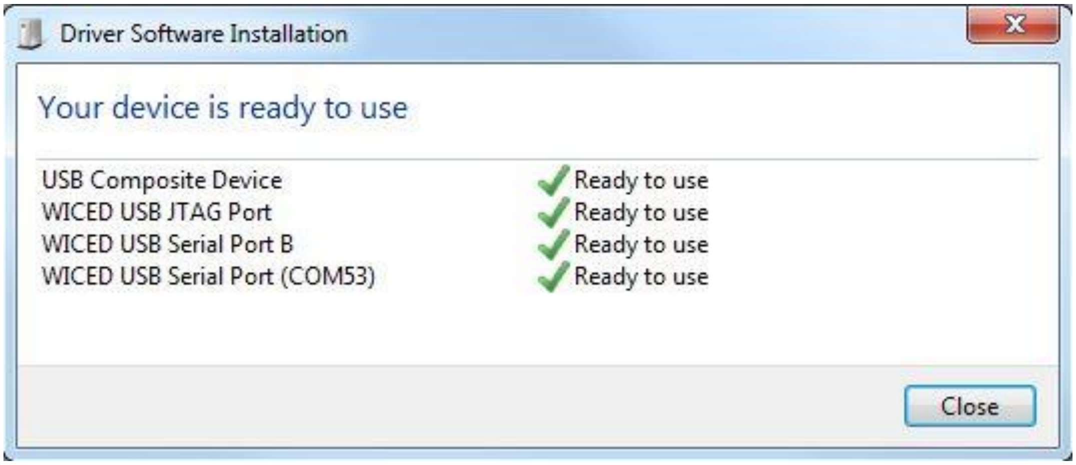
3.3 CYW943907AEVAL1F Kit Connection

The CYW943907AEVAL1F EVK can be powered by the following options:

- External power supply
- USB

When using external power supply, you should use a 5 V–12 V, 2A power supply with 2.1-mm DC Jack (center pin positive). When powered from USB, there are two logical USB devices: a USB-JTAG device and a USB-UART device. Drivers for the CYW943907AEVAL1F EVK are automatically installed during the WICED SDK installation process. When you connect the kit for the first time to your PC, it will initiate the driver search as shown in Figure 3-2.

Figure 3-2. Driver Software Installation



Your device is ready to use

USB Composite Device	✓ Ready to use
WICED USB JTAG Port	✓ Ready to use
WICED USB Serial Port B	✓ Ready to use
WICED USB Serial Port (COM53)	✓ Ready to use

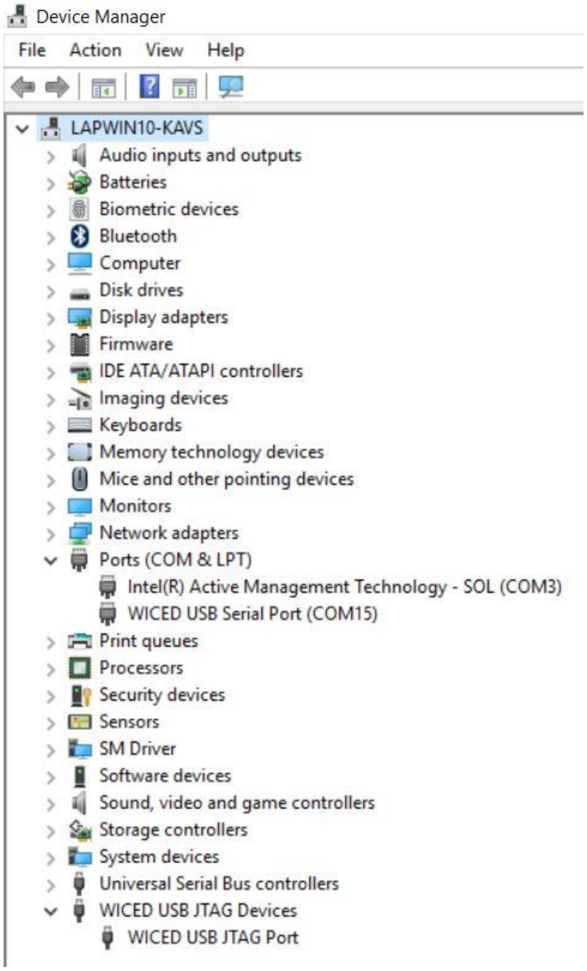
3.3.1 Verifying Driver Installation

To verify the successful completion of driver installation, perform these steps:

1. Right-click **My Computer** > **Properties**.
2. In the System Properties window, select **Device Manager**.
   a. The WICED USB Serial Port is listed under **Ports (COM & LPT)** as shown in Figure 3-3.
   b. The WICED USB JTAG Port is listed under **WICED USB JTAG Devices** as shown in Figure 3-3.

In Figure 3-3, the Device Manager window identifies the WICED USB Serial COM port as COMXX. The assigned port number varies between systems. If the device displays two WICED USB Serial Ports (WICED USB Serial port and WICED USB JTAG Port) instead of one, then follow the link mentioned in this post.

Figure 3-3. Verifying Device Driver Installation



Device Manager

- LAPWIN10-KAVS
  - Audio inputs and outputs
  - Batteries
  - Biometric devices
  - Bluetooth
  - Computer
  - Disk drives
  - Display adapters
  - Firmware
  - IDE ATA/ATAPI controllers
  - Imaging devices
  - Keyboards
  - Memory technology devices
  - Mice and other pointing devices
  - Monitors
  - Network adapters
  - Ports (COM & LPT)
    - Intel(R) Active Management Technology - SOL (COM3)
    - WICED USB Serial Port (COM15)
  - Print queues
  - Processors
  - Security devices
  - Sensors
  - SM Driver
  - Software devices
  - Sound, video and game controllers
  - Storage controllers
  - System devices
  - Universal Serial Bus controllers
  - WICED USB JTAG Devices
    - WICED USB JTAG Port

3.3.2 Troubleshooting

If an error occurred during the automatic driver installation process, the driver may be manually installed from the following directory: *<WICED-SDK>\Drivers\Windows\wiced_uart*.

If the CYW943907AEVAL1F EVK does not appear in the Device Manager, verify that the +3V3 LED is turned ON and check the USB cable.

3.3.3 External Power Supply

The CYW943907AEVAL1F EVK can be supplied using an external power supply (5 V–12 V, 2A), using a 2.5-mm DC Jack with center pin positive. When using external power supply and also connecting a USB cable (for programming/debugging or USB-UART), the voltage on the external power supply should be greater than that of the USB supply. If not, the kit will be actually sourcing its power from USB rather than the external power supply.

3.4 Building, Programming, and Debugging CYW943907AEVAL1F EVK

3.4.1 Building and Programming a Project for CYW943907AEVAL1F in WICED Studio IDE

To build and program a project for CYW943907AEVAL1F EVK, perform the following steps:

1. To open the WICED IDE on Windows PC, go to **Start** > **All Programs** > **Cypress** > **WICED-Studio**.
2. Select **43xxx_Wi-Fi** in the WICED Target selector drop-down box as shown in Figure 1-5. Building a project requires a corresponding make target, located in the Make Target window. All applications should go under the **apps** directory. The make target path will contain the directory hierarchy starting from **apps** with directory names separated by a period. The project name is followed by a hyphen and then the platform name. Finally, the actions to be performed after the build are specified such as download and run. For example, to build, download, and run the application scan which exists in *apps\snip\scan*, create the following make target:

   ```
   snip.scan-CYW943907AEVAL1F download run
   ```

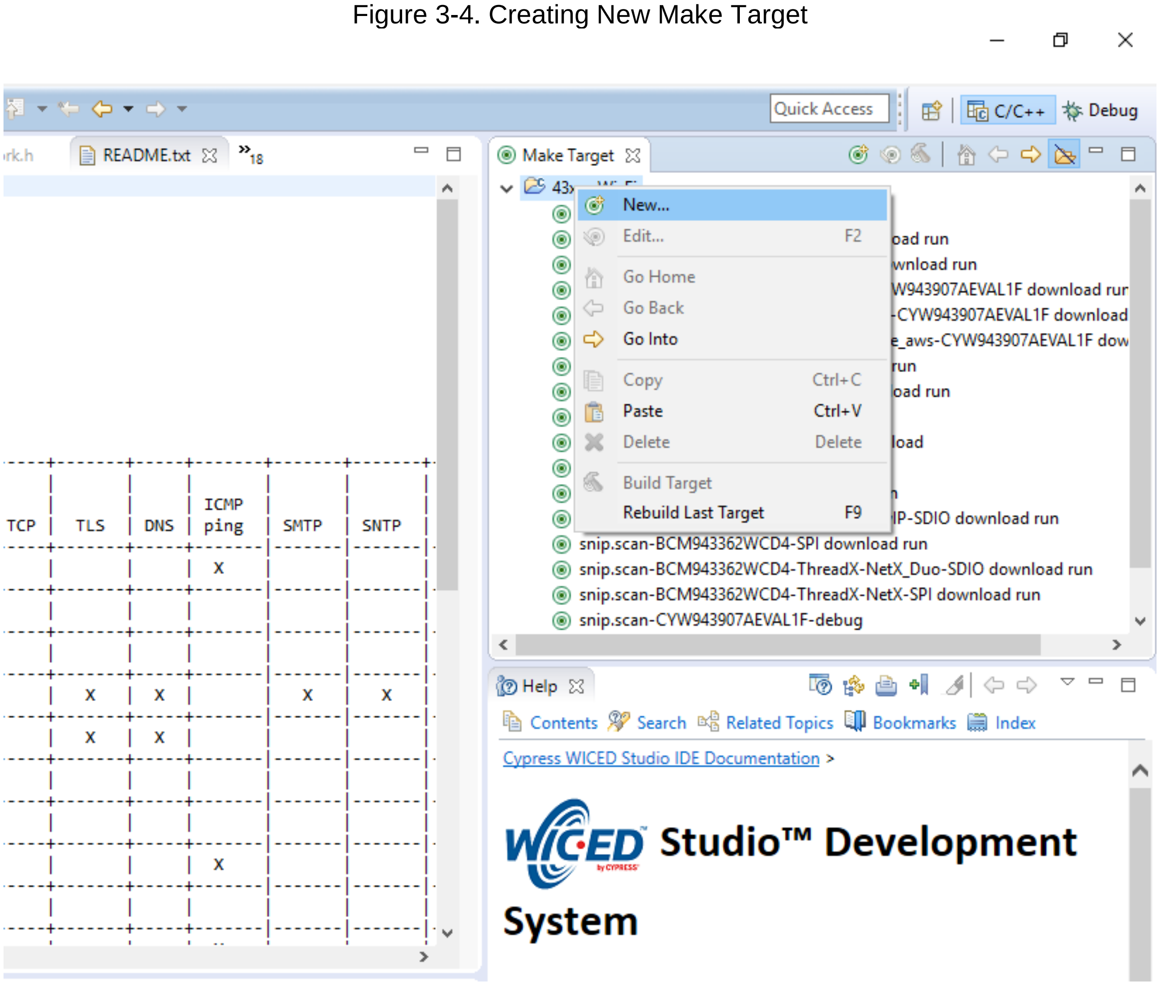
   This project will periodically scan for Wi-Fi access points and will list them using the serial to USB connection on the kit.

Note: By default, kit comes pre-programmed with the same `snip.scan` example.

Perform these steps to create the make target, build, program, and test application scan:

3. Right-click **43xxx_Wi-Fi** in Make Target window as shown in Figure 3-4 and click **New…**

Figure 3-4. Creating New Make Target


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

4. Enter `snip.scan-CYW943907AEVAL1F download run` in the **Target name** field and click the **OK** button

Note: The list of all commands that can be provided in the Make target is listed in *<WICED-SDK installation directory>/ 43xxx_Wi-Fi/Makefile*.

   `snip.scan-CYW943907AEVAL1F download run` indicates the following:

   `snip` = directory inside **apps** folder

   `scan` = Sub-directory and name of the application to be built. For example, to build the console application under *test* directory in *apps*, then use `test.console` instead of `snip.scan`.

   `CYW943907AEVAL1F` = Board/platform name

   `download` = Indicates download to target

   `run` = Resets the target and starts execution

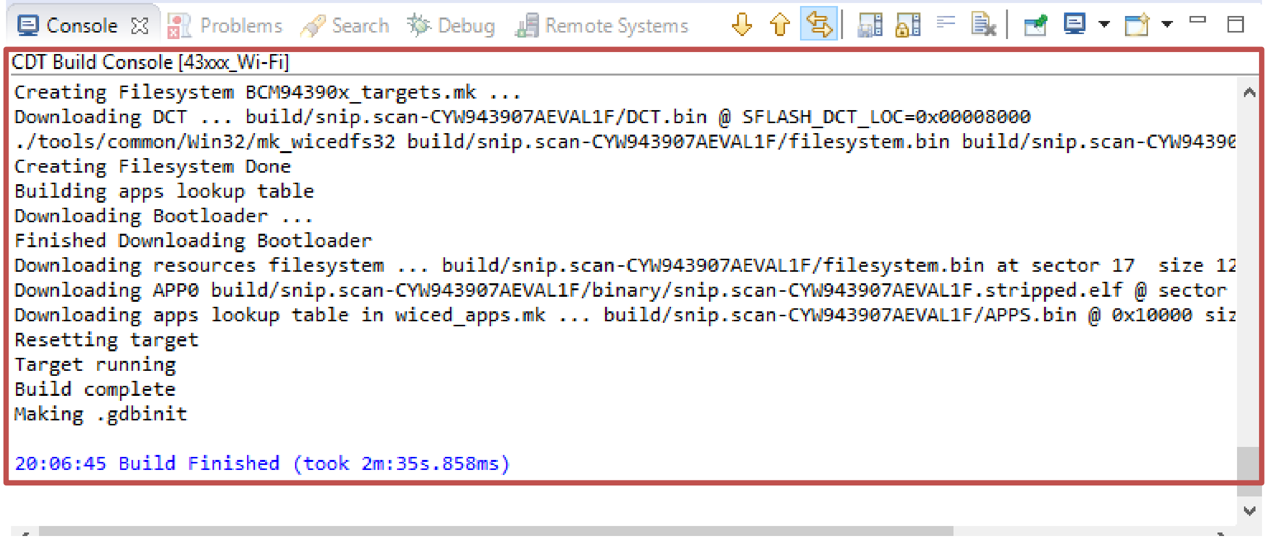
5. Double-click (alternately, right-click and select **Build Target**) the `Clean` Make Target to remove any output from the previous build. It is recommended to Make Clean when any new files are added or removed to the corresponding Target.

Note: Ensure that you have connected CYW943907AEVAL1F EVK to the same PC via USB prior to executing the build target.

6. Double-click (alternatively right-click and select **Build Target**) the `snip.scan-`CYW943907AEVAL1F `download run` Make Target to build and download it to the CYW943907AEVAL1F EVK.

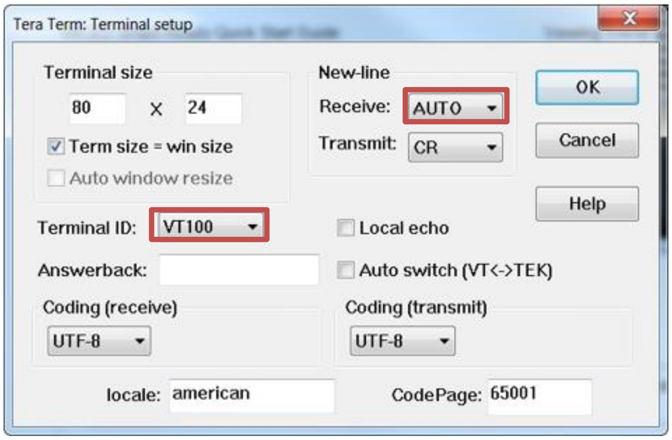
   The project is built and programmed into the CYW943907AEVAL1F EVK, as shown in Figure 3-5.

Figure 3-5. Successful Build and Program



```
Creating Filesystem BCM94390x_targets.mk ...
Downloading DCT ... build/snip.scan-CYW943907AEVAL1F/DCT.bin @ SFLASH_DCT_LOC=0x00008000
./tools/common/Win32/mk_wicedfs32 build/snip.scan-CYW943907AEVAL1F/filesystem.bin build/snip.scan-CYW9439
Creating Filesystem Done
Building apps lookup table
Downloading Bootloader ...
Finished Downloading Bootloader
Downloading resources filesystem ... build/snip.scan-CYW943907AEVAL1F/filesystem.bin at sector 17  size 12
Downloading APP0 build/snip.scan-CYW943907AEVAL1F/binary/snip.scan-CYW943907AEVAL1F.stripped.elf @ sector
Downloading apps lookup table in wiced_apps.mk ... build/snip.scan-CYW943907AEVAL1F/APPS.bin @ 0x10000 siz
Resetting target
Target running
Build complete
Making .gdbinit

20:06:45 Build Finished (took 2m:35s.858ms)
```

7. To view output messages with a terminal emulation program (such as Tera Term), follow these steps:
   a. Start the terminal emulation program.
   b. Set the Terminal ID to **VT100** and New-Line **Receive** to **AUTO**. Other settings should be left at the default settings.



Tera Term: Terminal setup

Terminal size
80 x 24
☑ Term size = win size
□ Auto window resize

New-line
Receive: AUTO
Transmit: CR

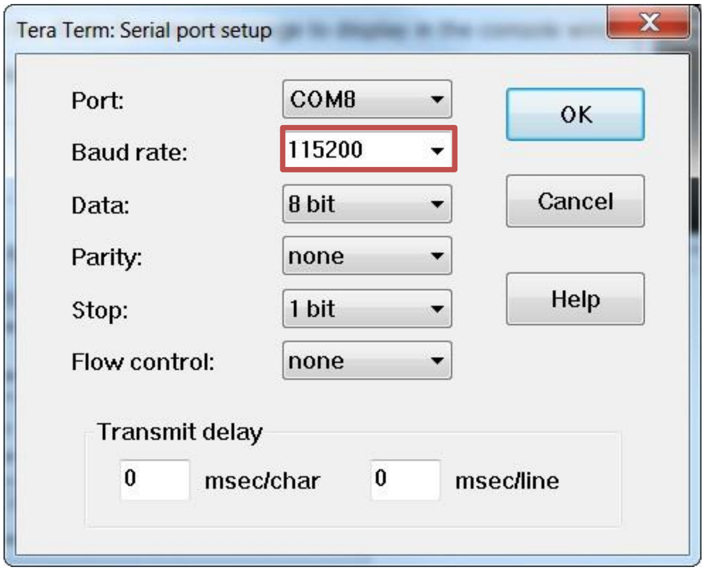
Terminal ID: VT100
Answerback:
□ Local echo
□ Auto switch (VT<->TEK)

Coding (receive): UTF-8
Coding (transmit): UTF-8

locale: american
CodePage: 65001

   c. In the Terminal Emulator, initiate a connection with the Serial port number from the Device Manager on the PC as shown in Figure 3-3.

Note: Exact Port number will vary with the corresponding PC port.



Tera Term: Serial port setup

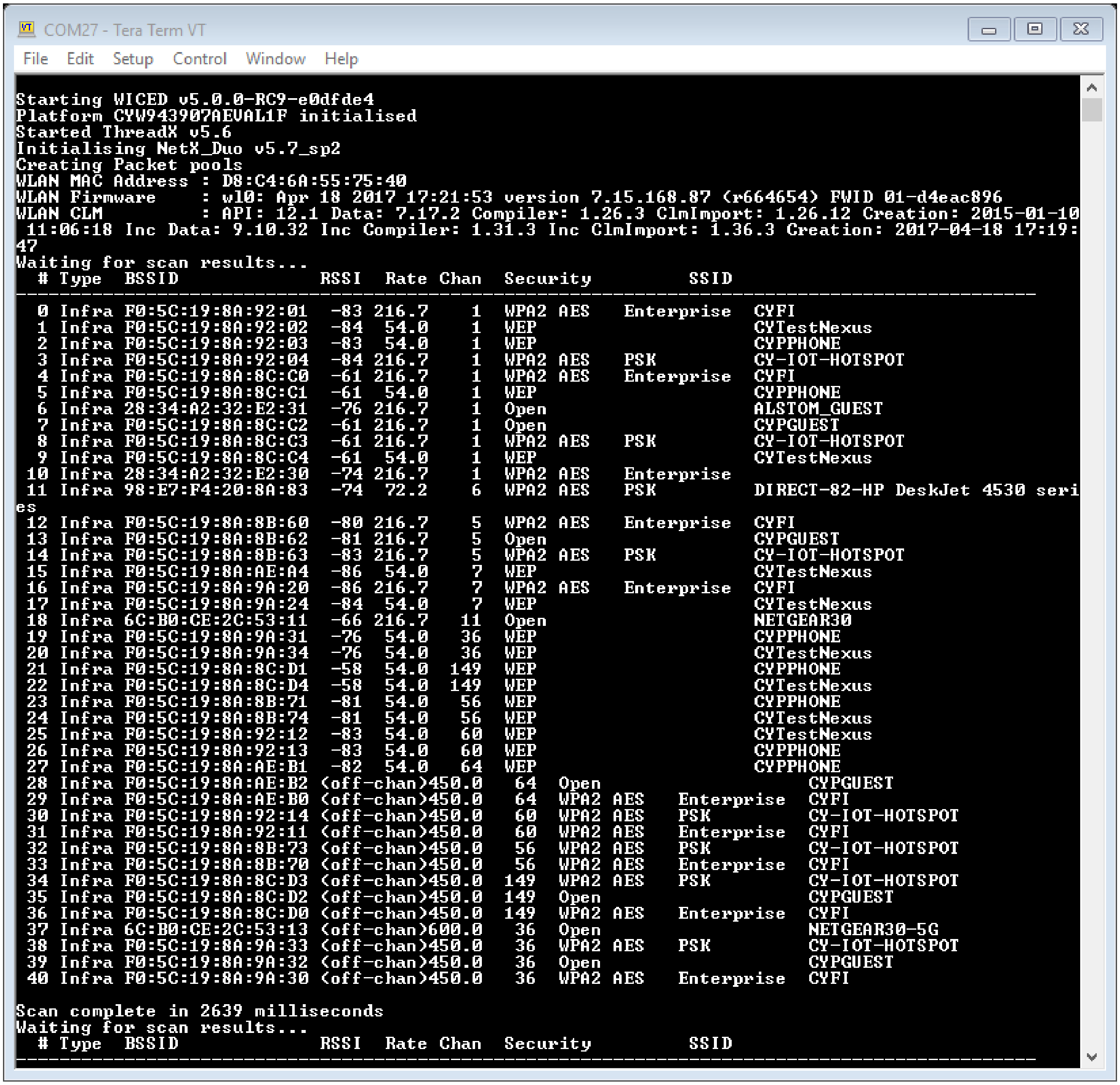
Port: COM8
Baud rate: 115200
Data: 8 bit
Parity: none
Stop: 1 bit
Flow control: none

Transmit delay
0 msec/char
0 msec/line

   d. Press the **Reset** button (see Figure 1-1) on the CYW943907AEVAL1F EVK to view the application start-up messages.

8. The output of the Terminal Emulation program should be similar to what is shown in Figure 3-6.

Figure 3-6. Console Output



```
Starting WICED v5.0.0-RC9-e0dfde4
Platform CYW943907AEVAL1F initialised
Started ThreadX v5.6
Initialising NetX_Duo v5.7_sp2
Creating Packet pools
WLAN MAC Address : D8:C4:6A:55:75:40
WLAN Firmware    : wl0: Apr 18 2017 17:21:53 version 7.15.168.87 (r664654) FWID 01-d4eac896
WLAN CLM         : API: 12.1 Data: 7.17.2 Compiler: 1.26.3 ClmImport: 1.26.12 Creation: 2015-01-10
 11:06:18 Inc Data: 9.10.32 Inc Compiler: 1.31.3 Inc ClmImport: 1.36.3 Creation: 2017-04-18 17:19:
47
Waiting for scan results...
  # Type  BSSID             RSSI  Rate Chan  Security         SSID
----------------------------------------------------------------------------------------------
  0 Infra F0:5C:19:8A:92:01  -83 216.7    1  WPA2 AES   Enterprise  CYFI
  1 Infra F0:5C:19:8A:92:02  -84  54.0    1  WEP                    CYTestNexus
  2 Infra F0:5C:19:8A:92:03  -83  54.0    1  WEP                    CYPPHONE
  3 Infra F0:5C:19:8A:92:04  -84 216.7    1  WPA2 AES   PSK         CY-IOT-HOTSPOT
  4 Infra F0:5C:19:8A:8C:C0  -61 216.7    1  WPA2 AES   Enterprise  CYFI
  5 Infra F0:5C:19:8A:8C:C1  -61  54.0    1  WEP                    CYPPHONE
  6 Infra 28:34:A2:32:E2:31  -76 216.7    1  Open                   ALSTOM_GUEST
  7 Infra F0:5C:19:8A:8C:C2  -61 216.7    1  Open                   CYPGUEST
  8 Infra F0:5C:19:8A:8C:C3  -61 216.7    1  WPA2 AES   PSK         CY-IOT-HOTSPOT
  9 Infra F0:5C:19:8A:8C:C4  -61  54.0    1  WEP                    CYTestNexus
 10 Infra 28:34:A2:32:E2:30  -74 216.7    1  WPA2 AES   Enterprise
 11 Infra 98:E7:F4:20:8A:83  -74  72.2    6  WPA2 AES   PSK         DIRECT-82-HP DeskJet 4530 seri
es
 12 Infra F0:5C:19:8A:8B:60  -80 216.7    5  WPA2 AES   Enterprise  CYFI
 13 Infra F0:5C:19:8A:8B:62  -81 216.7    5  Open                   CYPGUEST
 14 Infra F0:5C:19:8A:8B:63  -83 216.7    5  WPA2 AES   PSK         CY-IOT-HOTSPOT
 15 Infra F0:5C:19:8A:AE:A4  -86  54.0    7  WEP                    CYTestNexus
 16 Infra F0:5C:19:8A:9A:20  -86 216.7    7  WPA2 AES   Enterprise  CYFI
 17 Infra F0:5C:19:8A:9A:24  -84  54.0    7  WEP                    CYTestNexus
 18 Infra 6C:B0:CE:2C:53:11  -66 216.7   11  Open                   NETGEAR30
 19 Infra F0:5C:19:8A:9A:31  -76  54.0   36  WEP                    CYPPHONE
 20 Infra F0:5C:19:8A:9A:34  -76  54.0   36  WEP                    CYTestNexus
 21 Infra F0:5C:19:8A:8C:D1  -58  54.0  149  WEP                    CYPPHONE
 22 Infra F0:5C:19:8A:8C:D4  -58  54.0  149  WEP                    CYTestNexus
 23 Infra F0:5C:19:8A:8B:71  -81  54.0   56  WEP                    CYPPHONE
 24 Infra F0:5C:19:8A:8B:74  -81  54.0   56  WEP                    CYTestNexus
 25 Infra F0:5C:19:8A:92:12  -83  54.0   60  WEP                    CYTestNexus
 26 Infra F0:5C:19:8A:92:13  -83  54.0   60  WEP                    CYPPHONE
 27 Infra F0:5C:19:8A:AE:B1  -82  54.0   64  WEP                    CYPPHONE
 28 Infra F0:5C:19:8A:AE:B2 (off-chan)450.0   64  Open                   CYPGUEST
 29 Infra F0:5C:19:8A:AE:B0 (off-chan)450.0   64  WPA2 AES   Enterprise  CYFI
 30 Infra F0:5C:19:8A:92:14 (off-chan)450.0   60  WPA2 AES   PSK         CY-IOT-HOTSPOT
 31 Infra F0:5C:19:8A:92:11 (off-chan)450.0   60  WPA2 AES   Enterprise  CYFI
 32 Infra F0:5C:19:8A:8B:73 (off-chan)450.0   56  WPA2 AES   PSK         CY-IOT-HOTSPOT
 33 Infra F0:5C:19:8A:8B:70 (off-chan)450.0   56  WPA2 AES   Enterprise  CYFI
 34 Infra F0:5C:19:8A:8C:D3 (off-chan)450.0  149  WPA2 AES   PSK         CY-IOT-HOTSPOT
 35 Infra F0:5C:19:8A:8C:D2 (off-chan)450.0  149  Open                   CYPGUEST
 36 Infra F0:5C:19:8A:8C:D0 (off-chan)450.0  149  WPA2 AES   Enterprise  CYFI
 37 Infra 6C:B0:CE:2C:53:13 (off-chan)600.0   36  Open                   NETGEAR30-5G
 38 Infra F0:5C:19:8A:9A:33 (off-chan)450.0   36  WPA2 AES   PSK         CY-IOT-HOTSPOT
 39 Infra F0:5C:19:8A:9A:32 (off-chan)450.0   36  Open                   CYPGUEST
 40 Infra F0:5C:19:8A:9A:30 (off-chan)450.0   36  WPA2 AES   Enterprise  CYFI

Scan complete in 2639 milliseconds
Waiting for scan results...
  # Type  BSSID             RSSI  Rate Chan  Security         SSID
----------------------------------------------------------------------------------------------
```

3.4.2 Troubleshooting

If a `download_dct` error message is displayed despite connecting the board, follow the steps outlined in this post.

3.4.3 Debugging a Project using Breakpoints

After programming a project, it is possible to debug it in CYW943907AEVAL1F EVK using the built-in debugger.

Note that the scan example used in Building and Programming a Project for CYW943907AEVAL1F in WICED Studio IDE section is also used here. The steps outlined in Building and Programming a Project for CYW943907AEVAL1F in WICED Studio IDE should be first followed with a slight change (adding `-debug` to the Make Target command and removing run). Instead of using the make command, `snip.scan-CYW943907AEVAL1F download run`, use the following:

```
snip.scan-CYW943907AEVAL1F-debug download
```

If `-debug` is not added, then it will be built for release. The important difference between the debug and release configurations is optimization. Debug is built with no optimization and release is built with optimization. It is possible to debug without using `-debug` as well, but with many variables and lines optimized away, many breakpoints might not get hit.

Note that Breakpoints must be placed after starting a debug session in WICED Studio 5.0 or later. If any breakpoints were created prior to the start of the debug session, their properties must be changed to be enabled for all Threads.

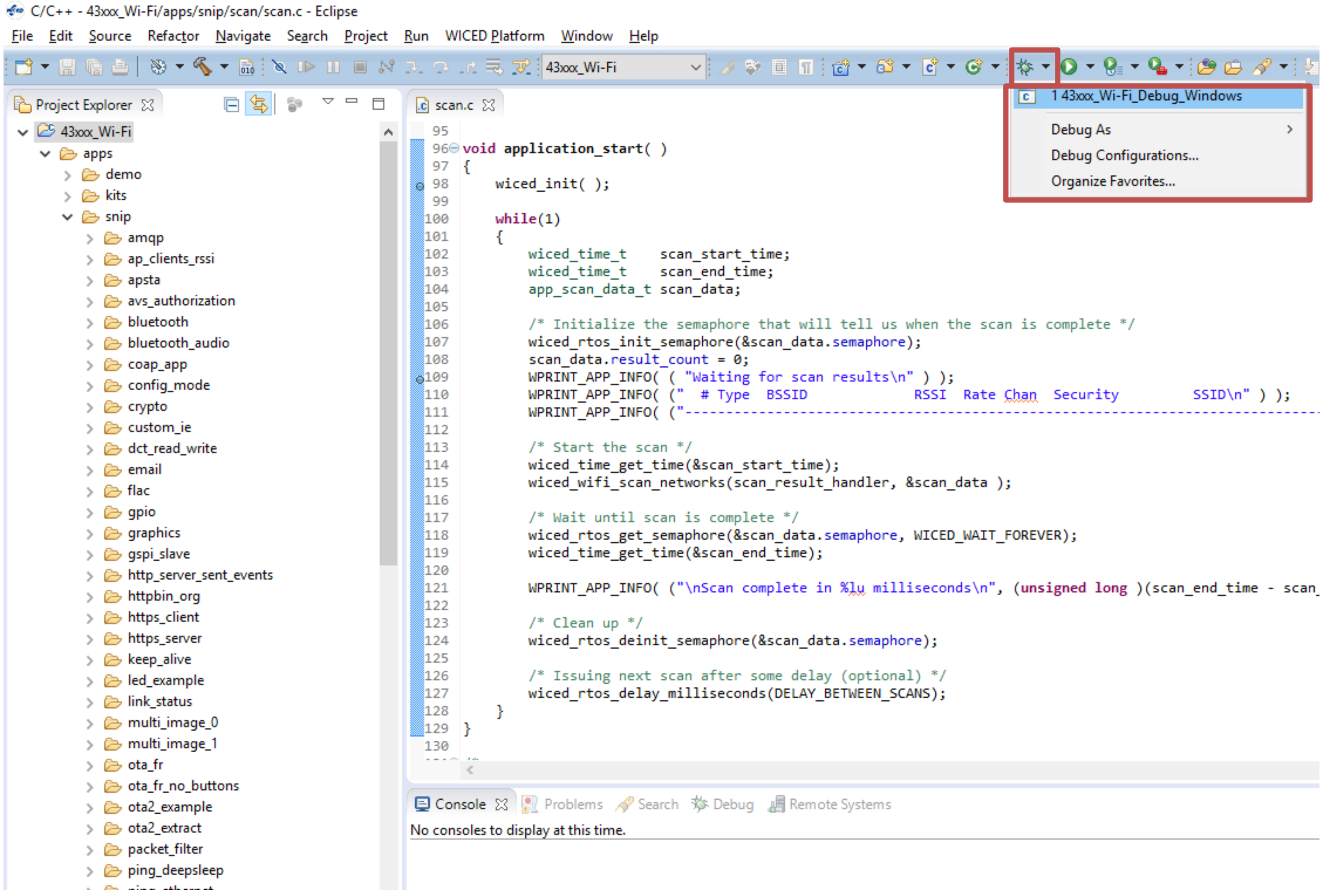
Perform these steps to debug the project:

1. Execute the Make Target described above to download the project to the device.
2. Click the arrow next to the **Debug** icon as shown in Figure 3-7 and select *43xxx-Wi-Fi_Debug_Windows*. The Confirm Perspective Switch dialog appears; click the **Yes** button. The Debug session starts and halts in the *start_GCC.s* file.

Note:

- The Confirm Perspective Switch dialog is not displayed if you have previously selected the **Remember my decision checkbox** in the Confirm Perspective Switch dialog.
- If any MakeFile/Build error occurs, then clean (using the *Clean* make target), re-build, and download to the CYW943907AEVAL1F EVK again. The Debug session starts and halts in the `start_GCC.s` file.
- In the Debug Perspective, the Project Explorer window goes away by default. To view the source files, switch back to the “C/C++” perspective.
- To switch between perspectives use the “C/C++” or “Debug” icon at the top right corner of screen.

Figure 3-7. Debugging Project



```
void application_start( )
{
    wiced_init( );

    while(1)
    {
        wiced_time_t    scan_start_time;
        wiced_time_t    scan_end_time;
        app_scan_data_t scan_data;

        /* Initialize the semaphore that will tell us when the scan is complete */
        wiced_rtos_init_semaphore(&scan_data.semaphore);
        scan_data.result_count = 0;
        WPRINT_APP_INFO( ( "Waiting for scan results\n" ) );
        WPRINT_APP_INFO( ("  # Type  BSSID             RSSI  Rate Chan  Security         SSID\n" ) );
        WPRINT_APP_INFO( ("-------------------------------------------------------------------------------

        /* Start the scan */
        wiced_time_get_time(&scan_start_time);
        wiced_wifi_scan_networks(scan_result_handler, &scan_data );

        /* Wait until scan is complete */
        wiced_rtos_get_semaphore(&scan_data.semaphore, WICED_WAIT_FOREVER);
        wiced_time_get_time(&scan_end_time);

        WPRINT_APP_INFO( ("\nScan complete in %lu milliseconds\n", (unsigned long )(scan_end_time - scan_

        /* Clean up */
        wiced_rtos_deinit_semaphore(&scan_data.semaphore);

        /* Issuing next scan after some delay (optional) */
        wiced_rtos_delay_milliseconds(DELAY_BETWEEN_SCANS);
    }
}
```

No consoles to display at this time.

3. Open the *scan.c* file from the Project Explorer window. Click on the line with `WPRINT_APP_INFO( ( "Waiting for scan results...\n" ) );` and press the **[Ctrl +Shift+B]** keys on your keyboard. A blue hollow circle along with a check mark appears next to the line number, as shown in Figure 3-8.

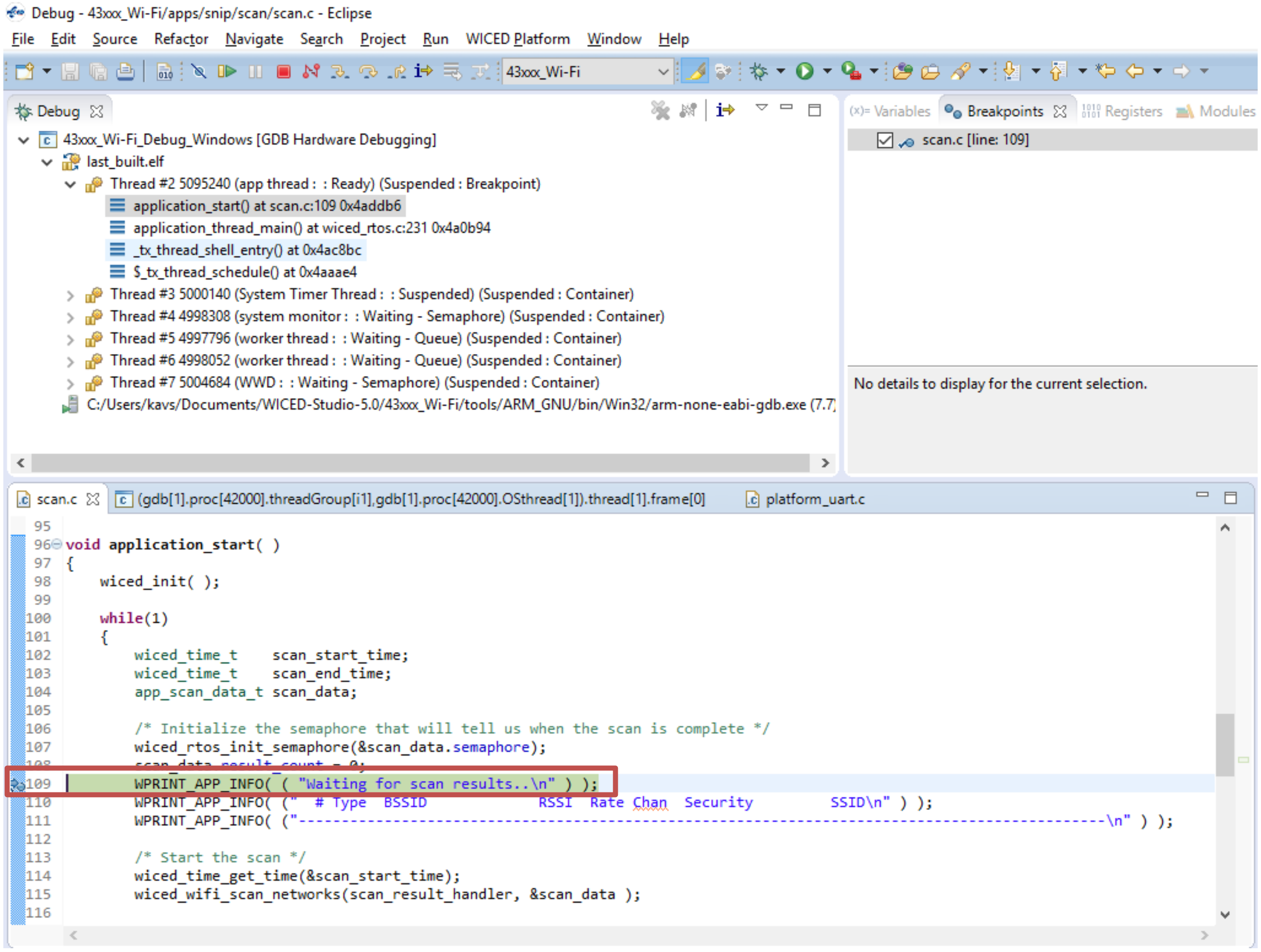
4. From the main menu, click **Run** > **Resume**. Execution will stop at the breakpoint that you added. To continue after hitting the breakpoint, click **Resume** again.

5. To disable the breakpoint, press the **[Ctrl+Shift+B]** keys again on the same line, or deselect the corresponding checkbox in the Breakpoints window.

Note: If the Breakpoint window does not appear, then choose **Window** > **Show View** > **Breakpoints**.

6. To terminate the Debugging session, click **Run** > **Terminate**, or click on the red Square icon. Once you terminate the session, click on “**C/C++**” in the upper right corner to return to the C/C++ perspective.

Figure 3-8. Placing Breakpoint in Code


Debug - 43xxx_Wi-Fi/apps/snip/scan/scan.c - Eclipse
File Edit Source Refactor Navigate Search Project Run WICED Platform Window Help
43xxx_Wi-Fi
Debug
(x)= Variables
Breakpoints
Registers
Modules
scan.c
(gdb[1].proc[42000].threadGroup[i1],gdb[1].proc[42000].OSthread[1]).thread[1].frame[0]
platform_uart.c

```
43xxx_Wi-Fi_Debug_Windows [GDB Hardware Debugging]
    last_built.elf
        Thread #2 5095240 (app thread : : Ready) (Suspended : Breakpoint)
            application_start() at scan.c:109 0x4addb6
            application_thread_main() at wiced_rtos.c:231 0x4a0b94
            _tx_thread_shell_entry() at 0x4ac8bc
            $_tx_thread_schedule() at 0x4aaae4
        Thread #3 5000140 (System Timer Thread : : Suspended) (Suspended : Container)
        Thread #4 4998308 (system monitor : : Waiting - Semaphore) (Suspended : Container)
        Thread #5 4997796 (worker thread : : Waiting - Queue) (Suspended : Container)
        Thread #6 4998052 (worker thread : : Waiting - Queue) (Suspended : Container)
        Thread #7 5004684 (WWD : : Waiting - Semaphore) (Suspended : Container)
    C:/Users/kavs/Documents/WICED-Studio-5.0/43xxx_Wi-Fi/tools/ARM_GNU/bin/Win32/arm-none-eabi-gdb.exe (7.7)
```

☑ scan.c [line: 109]

No details to display for the current selection.

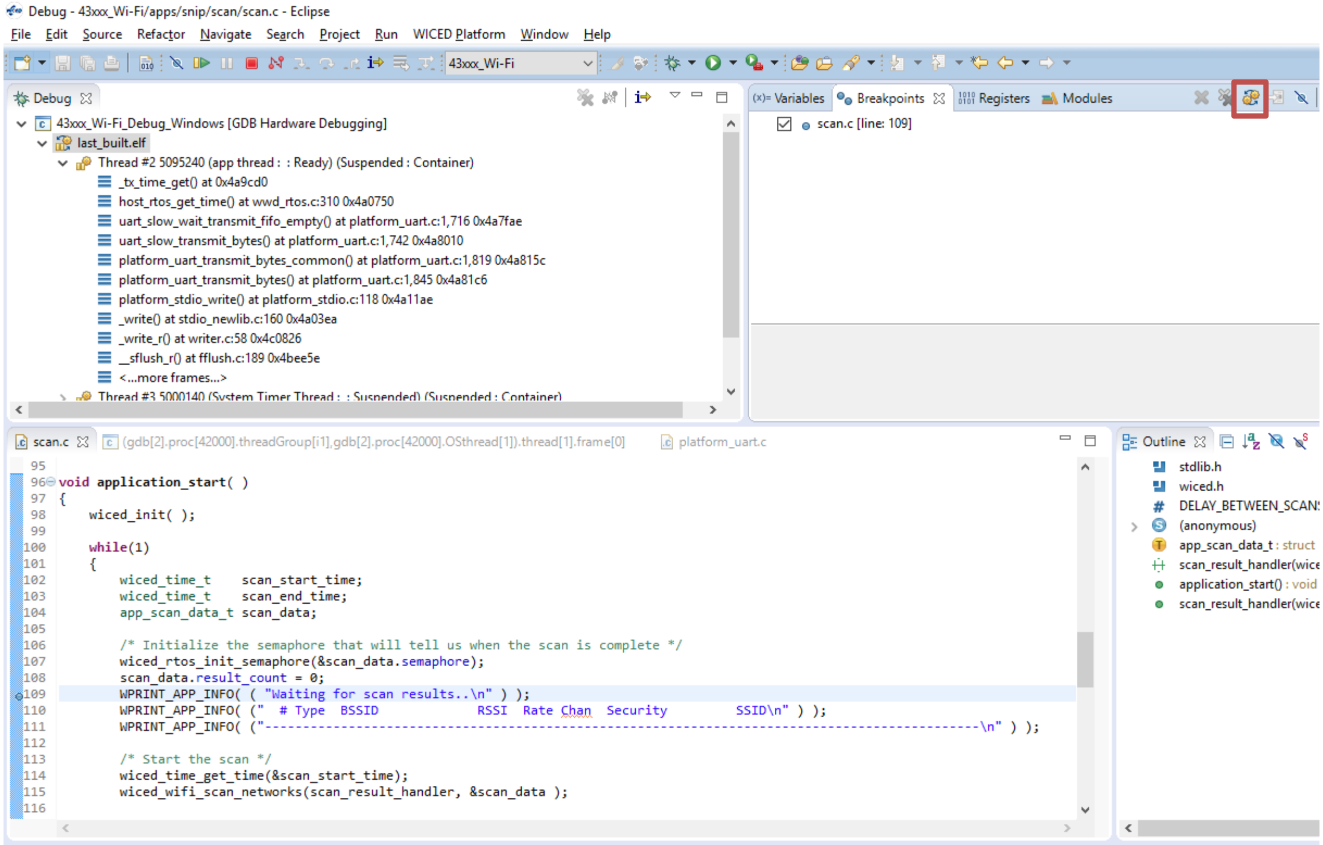
```

void application_start( )
{
    wiced_init( );

    while(1)
    {
        wiced_time_t     scan_start_time;
        wiced_time_t     scan_end_time;
        app_scan_data_t scan_data;

        /* Initialize the semaphore that will tell us when the scan is complete */
        wiced_rtos_init_semaphore(&scan_data.semaphore);
        scan_data.result_count = 0;
        WPRINT_APP_INFO( ( "Waiting for scan results..\n" ) );
        WPRINT_APP_INFO( ("  # Type  BSSID             RSSI  Rate Chan  Security         SSID\n" ) );
        WPRINT_APP_INFO( ("----------------------------------------------------------------------------------------------\n" ) );

        /* Start the scan */
        wiced_time_get_time(&scan_start_time);
        wiced_wifi_scan_networks(scan_result_handler, &scan_data );

```

7. If Breakpoints are created prior to starting the current Debug session, they will not be associated with the current thread and will be indicated with a blue circle without a check mark. To enable the Breakpoints in the current thread, associate the properties from the Breakpoints window with the current thread.

Note: If you do not see any breakpoints in the Breakpoints window, click the **Show Breakpoints Supported by Selected Target** icon as shown in Figure 3-9. The breakpoints are displayed.

8. Right-click the desired breakpoint checkbox and click **Breakpoint Properties…**. Click the **last_built.elf** check box, as shown in Figure 3-10. The check mark appears before the actual breakpoint indicating its association with the current execution.

Figure 3-9. Show Breakpoints Icon


Debug - 43xxx_Wi-Fi/apps/snip/scan/scan.c - Eclipse
File Edit Source Refactor Navigate Search Project Run WICED Platform Window Help
43xxx_Wi-Fi
Debug
(x)= Variables
Breakpoints
Registers
Modules
scan.c
platform_uart.c
Outline

43xxx_Wi-Fi_Debug_Windows [GDB Hardware Debugging]
last_built.elf
Thread #2 5095240 (app thread : : Ready) (Suspended : Container)
_tx_time_get() at 0x4a9cd0
host_rtos_get_time() at wwd_rtos.c:310 0x4a0750
uart_slow_wait_transmit_fifo_empty() at platform_uart.c:1,716 0x4a7fae
uart_slow_transmit_bytes() at platform_uart.c:1,742 0x4a8010
platform_uart_transmit_bytes_common() at platform_uart.c:1,819 0x4a815c
platform_uart_transmit_bytes() at platform_uart.c:1,845 0x4a81c6
platform_stdio_write() at platform_stdio.c:118 0x4a11ae
_write() at stdio_newlib.c:160 0x4a03ea
_write_r() at writer.c:58 0x4c0826
__sflush_r() at fflush.c:189 0x4bee5e
<...more frames...>
Thread #3 5000140 (System Timer Thread : : Suspended) (Suspended : Container)

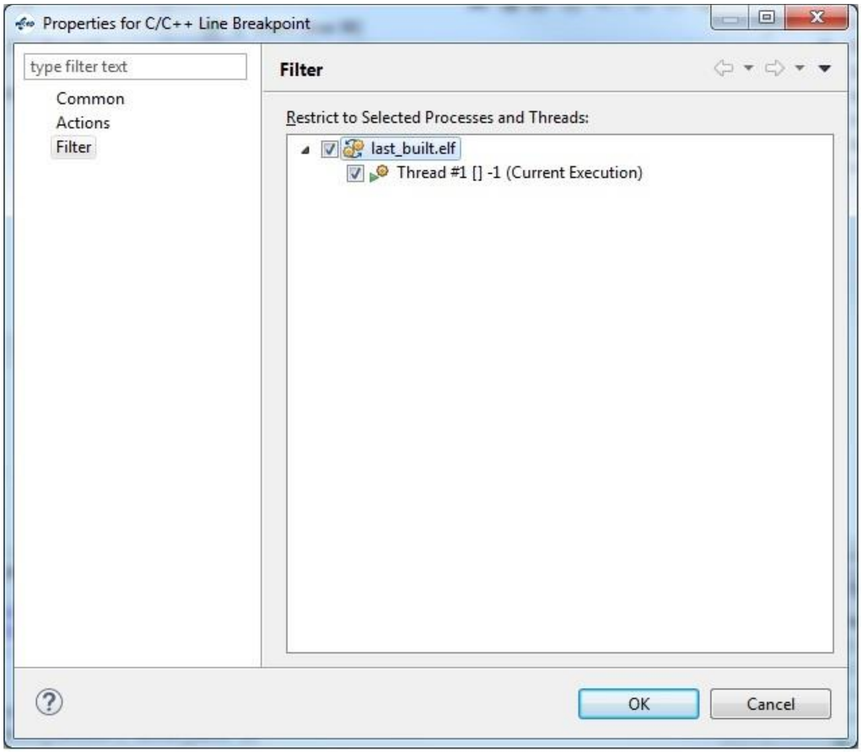
☑ scan.c [line: 109]

(gdb[2].proc[42000].threadGroup[i1].gdb[2].proc[42000].OSthread[1]).thread[1].frame[0]

```
void application_start( )
{
    wiced_init( );

    while(1)
    {
        wiced_time_t    scan_start_time;
        wiced_time_t    scan_end_time;
        app_scan_data_t scan_data;

        /* Initialize the semaphore that will tell us when the scan is complete */
        wiced_rtos_init_semaphore(&scan_data.semaphore);
        scan_data.result_count = 0;
        WPRINT_APP_INFO( ( "Waiting for scan results..\n" ) );
        WPRINT_APP_INFO( ("  # Type  BSSID             RSSI  Rate Chan  Security         SSID\n" ) );
        WPRINT_APP_INFO( ("----------------------------------------------------------------------------------------------\n" ) );

        /* Start the scan */
        wiced_time_get_time(&scan_start_time);
        wiced_wifi_scan_networks(scan_result_handler, &scan_data );
```

stdlib.h
wiced.h
DELAY_BETWEEN_SCAN!
(anonymous)
app_scan_data_t : struct
scan_result_handler(wice
application_start() : void
scan_result_handler(wice

Figure 3-10. Enabling Breakpoint for Current Execution



Filter

Restrict to Selected Processes and Threads:

☑ last_built.elf
☑ Thread #1 [] -1 (Current Execution)

4. Hardware

This chapter describes the CYW943907AEVAL1F EVK hardware and its different blocks, such as Bootstrap, reset control, Arduino-compatible headers, and module connectors.

The schematic is available at the following location after installing the software from Software Installation.

`<WICED_SDK_Directory>\43xx_Wifi\platforms\CYW943907AEVAL1F\schematics`.

4.1 Bootstrap and Control Pins

Bootstrap options available in the CYW943907AEVAL1F EVK are shown in Table 4-1. The pins are sampled at power-on reset (POR) to determine various operating modes. Sampling occurs a few milliseconds after an internal POR or de-assertion of the external POR. After the POR, each pin assumes the GPIO or alternative function specified in the CYW43907 Alternate GPIO function table in the CYW43907 Datasheet.

Ensure the gSPI mode and SDIO Host are not turned on at the same time since they share the same set of lines. For more information regarding bootstrap options, refer to the CYW43907 Datasheet.

Bootstrap options other than GPIO_7 and GPIO_13, are not available for the user to modify in this board.

To change Bootstrap options for GPIO_7 and GPIO_13, refer to the “Bootstraps, Flash” page of schematics.

Table 4-1. Bootstrap Options Available in CYW943907AEVAL1F EVK

Pin	Strap Function	Strap Pull	
		Chip Default	Board Default
GPIO_1	gSPI Mode 0 = Enable gSPI Mode 1 = Disable gSPI Mode	0	0
GPIO_7	WCPU Boot Mode: 0 = TCROM Boot 1 = TCMSRAM Boot	0	1 R135=10K to WLAN_VDDIO
GPIO_11	ACPU Boot Mode: 0 = SOCROM Boot 1 = SOCSRAM Boot	0	0
GPIO_13	SDIO Mode: 0 = SDIO Device 1 = SDIO Host	0	1 R141=10K to WLAN_VDDIO
GPIO_15	PMU VTRIM_enable 0 = VTRIM disable 1 = PMU VTRIM enabled **Note:** GPIO_15 is not a strap option for the B0 silicon revision of the device.	0	0

Pin	Strap Function	Strap Pull	
		Chip Default	Board Default
RF_SW_CTRL_5	Host DAP Clock Sel 1 = Enable XTAL clock for DAP subsystem 0 = Disable Use Test clock TCK for DAP subsystem	0	0
RF_SW_CTRL_7	PMU resource initialization mode selection 1 = Mode 1 0 = Mode 2	0	0
RF_SW_CTRL_9	LPO(Low Power oscillator) Selection: 0 = LPO from HIB (Hibernation Block) 1 = Internal 32KHz LPO	0	0

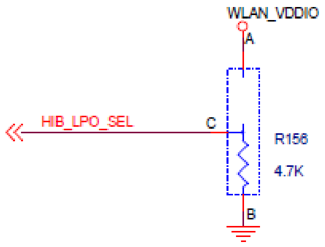
Note: There is no bootstrapping provision for GPIO_1, GPIO_11, GPIO_15, RF_SW_CTRL_5, RF_SW_CTRL_7, and RF_SW_CTRL_9 on the CYW943907AEVAL1F EVK board.

1. HIB_LPO_SEL and RF_SW_CTRL_9_HIB_LPO_SEL_UART2_TX

HIB_LPO_SEL

By default, it is automatically selected between the external crystal and internal oscillator. If only an internal oscillator is used, strapping of HIB_LPO_SEL is irrelevant.

Figure 4-1. HIB_LPO_SEL strapping option



RF_SW_CTRL_9 can also be used for LPO Selection. However, it should be pulled LOW to use the recommended external 32.768-kHz HIB crystal.

0 = LPO from HIB

1 = Internal 32-kHz LPO

Note:

- It is strongly recommended that you use the external 32.768-kHz crystal for a more accurate clock. Accuracy of the clock is critical when using power save mode.
- If an internal LPO is used, the board may become unresponsive at higher temperatures (>80°C).

2. HIB_REG_ON_IN

Used by the Hibernation (HIB) block to power up internal CYW43907 regulators. If the HIB_REG_ON_IN pin is LOW, regulators are disabled. For the HIB_REG_ON_IN pin to work as designed, HIB_REG_ON_OUT must be connected to REG_ON.

The CYW43907/BCM43907 datasheet states that HIB_REG_ON_IN needs to be delayed by at least two cycles of the 32.768-kHz clock after VBAT and VDDIO have reached 90% of their final values. To ensure a proper bootup, the RC delay circuit for HIB_REG_ON_IN is essential as shown in the following figure:

Figure 4-2. HIB_REG_ON_IN Delay Circuit

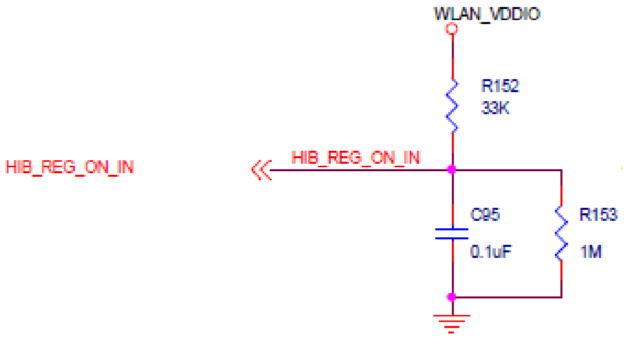
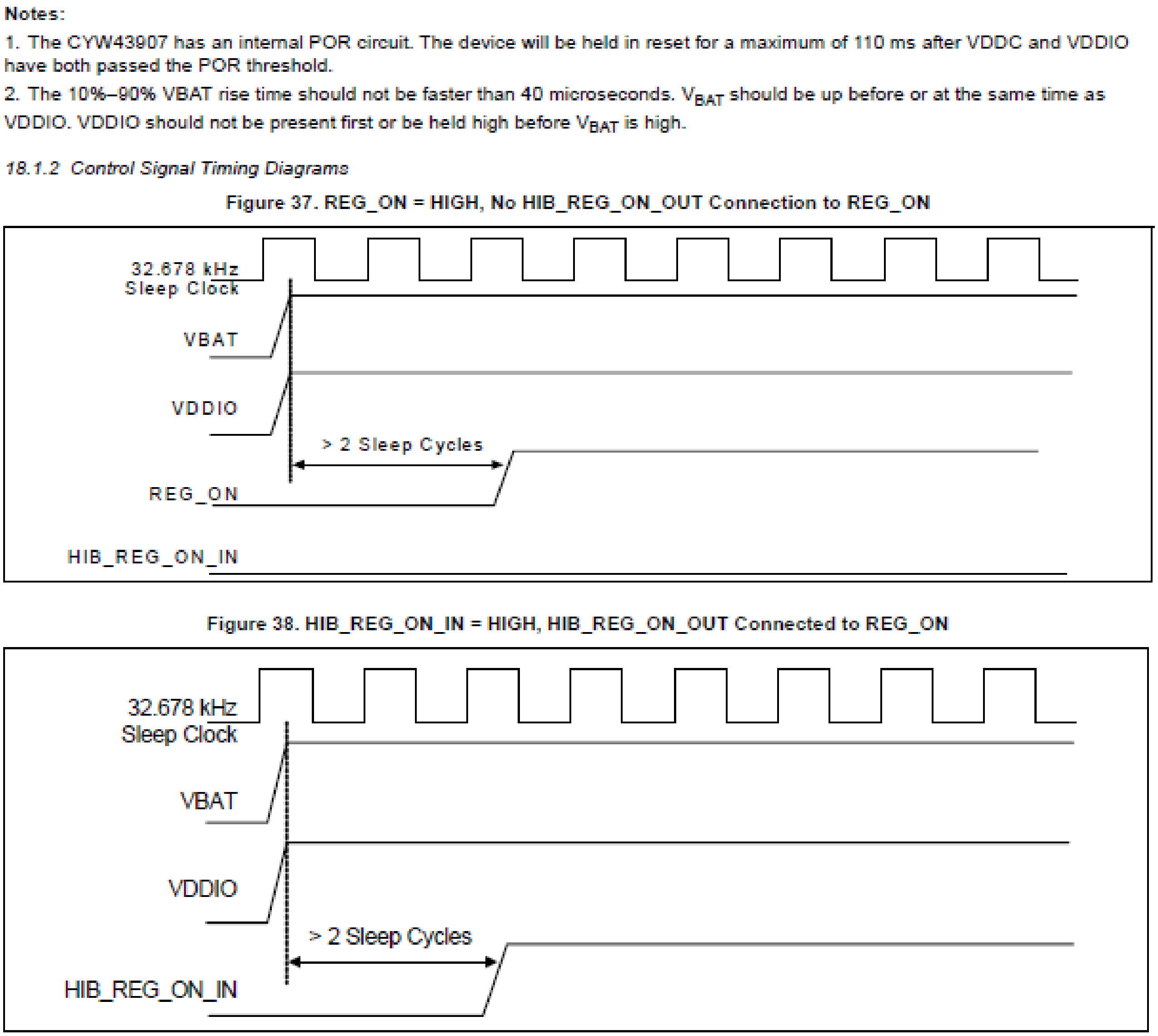


Figure 4-3. An Excerpt from CYW43907 Datasheet (Section 18)



Notes:

1. The CYW43907 has an internal POR circuit. The device will be held in reset for a maximum of 110 ms after VDDC and VDDIO have both passed the POR threshold.

2. The 10%–90% VBAT rise time should not be faster than 40 microseconds. V_{BAT} should be up before or at the same time as VDDIO. VDDIO should not be present first or be held high before V_{BAT} is high.

18.1.2 Control Signal Timing Diagrams

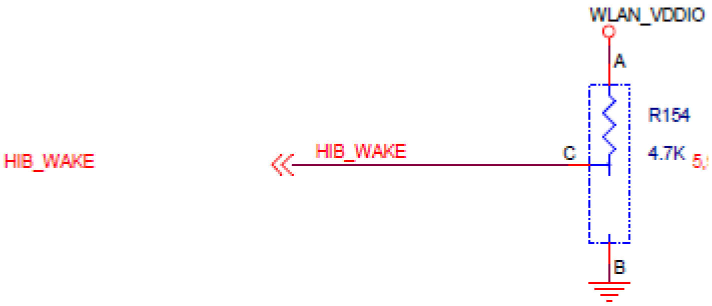
Figure 37. REG_ON = HIGH, No HIB_REG_ON_OUT Connection to REG_ON

Figure 38. HIB_REG_ON_IN = HIGH, HIB_REG_ON_OUT Connected to REG_ON

3. HIB_WAKE

Used to wakeup chip from Hibernation mode. This pin should be pulled HIGH.

Figure 4-4. HIB_WAKE Strapping Option



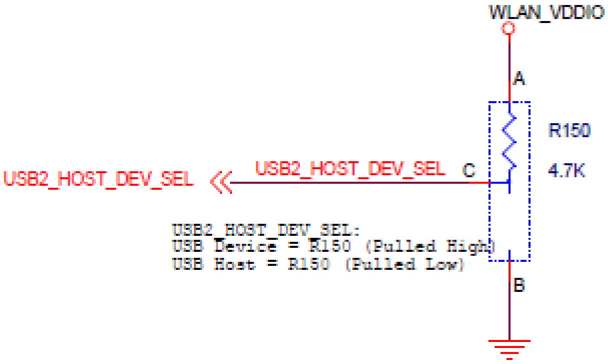
4. USB2_HOST_DEV_SEL

Used to select the USB mode; it is set in USB DEVICE mode by default.

0 = USB HOST mode

1 = USB DEVICE mode

Figure 4-5. USB2_HOST_DEV_SEL Strapping Option



5. JTAG_SEL and GPIO_8_TAP_SEL States for Test and Debug Function Selection

JTAG_SEL

Set JTAG_SEL:

0 = JTAG interface disabled

1 = JTAG debug mode enabled

GPIO_8_TAP_SEL

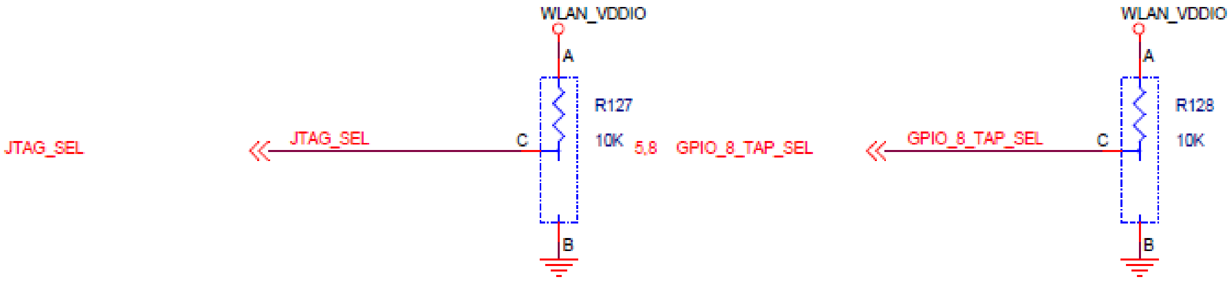
Set GPIO_8 (TAP_SEL):

0 = WLAN JTAG

1 = APPS JTAG

Note: Default setup is JTAG_SEL = HIGH / TAP_SEL = HIGH.

Figure 4-6. JTAG_SEL and GPIO_8_TAP_SEL Strapping Option



6. GPIO_1_GSPI_MODE

GPIO_1 is used for gSPI mode. By default, CYW43907 enables gSPI.

0=gSPI engine enabled

1=gSPI engine disabled

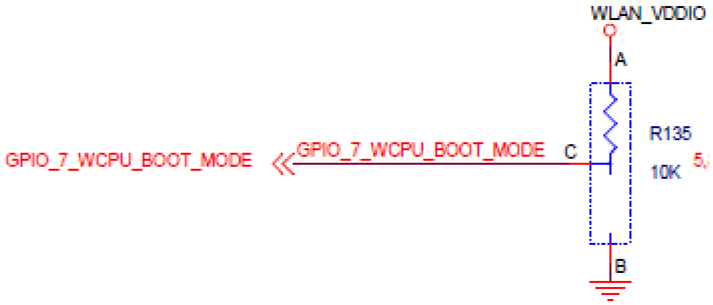
7. GPIO_7_WCPU_BOOT_MODE

GPIO_7 is used for WCPU Boot mode, and pulled HIGH for recommended TCMSRAM Boot mode.

0 = TCROM Boot

1 = TCMSRAM Boot

Figure 4-7. GPIO_7_WCPU_BOOT_MODE Strapping Option



8. GPIO_9_USB_SEL

GPIO_9 is used for USB selection. CYW43907 uses USB mode only; therefore, pull this pin HIGH.

0= HSIC Sel

1 =USB PHY

9. GPIO_11_ACPU_Boot Mode

GPIO_11 is used for ACPU Boot mode; by default, CYW43907 sets it to the recommended SOCROM Boot.

0 = SOCROM Boot

1 = SOCSRAM Boot

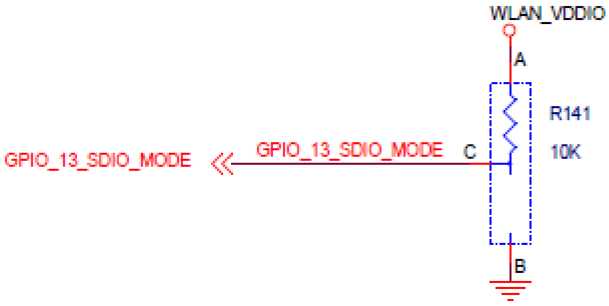
10. GPIO_13_SDIO_MODE

GPIO_13 is used to select the SDIO mode; it is set in SDIO HOST mode by default.

0 = SDIO Device

1 = SDIO Host

Figure 4-8. GPIO_13_SDIO_MODE Strapping Option



11. RF_SW_CTRL_7_RSRC_INIT_MODE_UART1_TX_OUT

This pin should be pulled HIGH.

Highly Recommended to pull up RF_SW_CTRL_7 via a 10-kΩ resistor to WLAN_VDDIO during bootup. Do not leave it floating. If left floating, the first-time programming of the SFLASH fails, while second-time programming passes.

12. CYW43907 in Deep Sleep Mode

To wake up CYW43907 from Deep Sleep mode, any GPIOs from GPIO_0 to GPIO_15 (except GPIO_13) can be used. The selected GPIO should not be pin-muxed; instead it should be a dedicated one for Deep Sleep wake up of CYW43907.

13. CYW43907 Power Signals for Unused Interfaces

Connect the below power signals to GND when the associated interface is not used.

VDDIO_SD

VDDIO_AUDIO

USB_VDD_3V3

VDDIO_RMII

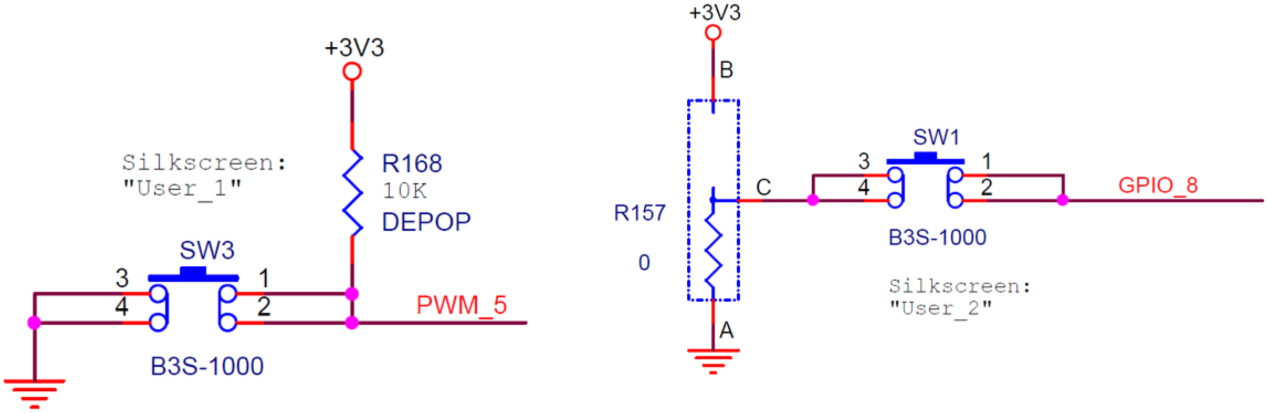
4.2 User Switches

There are two user switches available on the board named USER_1 and USER_2. Table 4-2 shows the Pin names and Enumeration used in WICED for the switches.

Table 4-2. User Switch available on the board

Switch	CYW43907 Pin Name	WICED_ENUM_ID	Alternate Enumeration in WICED
USER_1 (SW3)	PWM_5	WICED_GPIO_18	WICED_BUTTON1
USER_2 (SW1)	GPIO_8	WICED_GPIO_4	WICED_BUTTON2

Figure 4-9. User Switch Circuit Diagram



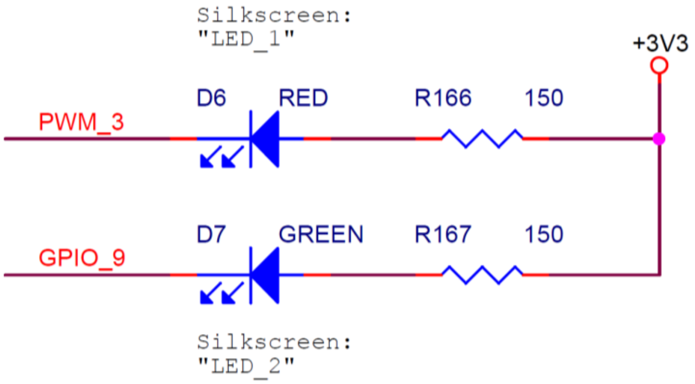
4.3 LED

There are two user LEDs available named LED_1 and LED_2. Table 4-3 shows the Pin name and Enumeration used in WICED for these LEDs.

Table 4-3. User LED Available on the Board

Switch	CYW43907 Pin Name	WICED_ENUM_ID	Alternate Enumeration in WICED
LED_1	PWM_3	WICED_GPIO_16	WICED_LED1
LED_2	GPIO_9	WICED_GPIO_5	WICED_LED2

Figure 4-10. User LED Circuit Diagram



4.4 Reset Control

The CYW43907 device can be reset using the “Target Reset” switch **SW2** or a `reset` command from the on-board programmer/debugger and serial interface chip, as shown in Figure 4-1111. The CYW43907/BCM43907 datasheet states that HIB_REG_ON_IN needs to be delayed by at least 2 cycles of the 32.768-kHz clock after VBAT and VDDIO have reached 90% of their final values. To ensure proper boot-up, the RC delay circuit for HIB_REG_ON_IN is essential as shown in Figure 4-12. See Section 4.1 for details on RC Delay Circuit.

Figure 4-11. Reset Circuit Diagram

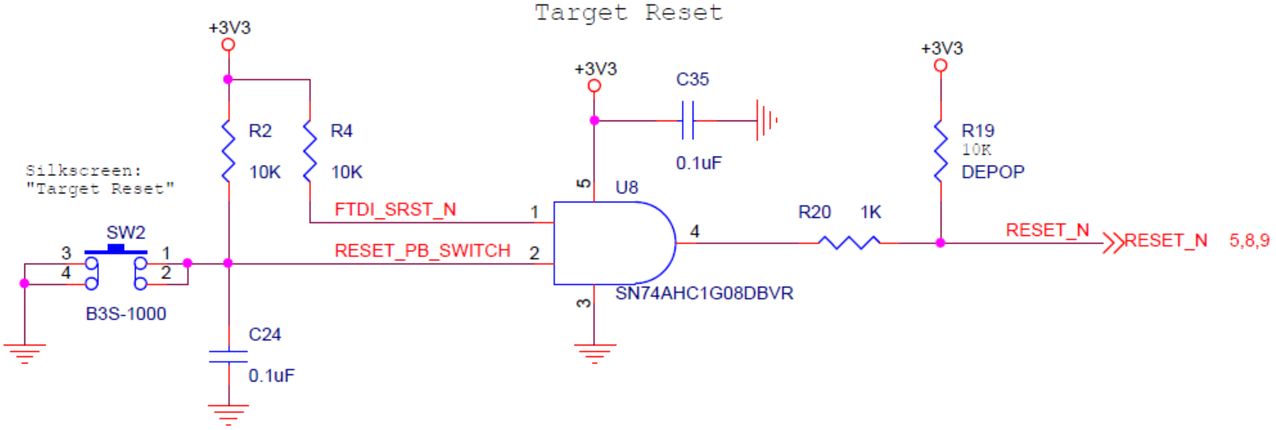
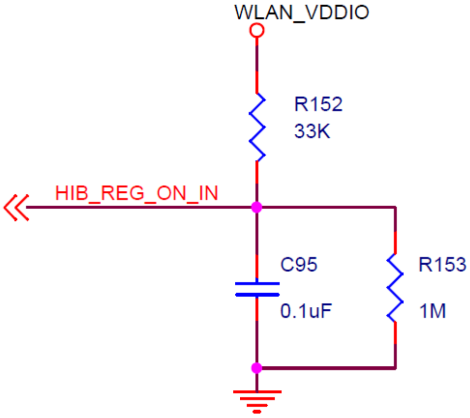


Figure 4-12. HIB_REG_ON_IN RC Delay Circuit



4.5 Ethernet

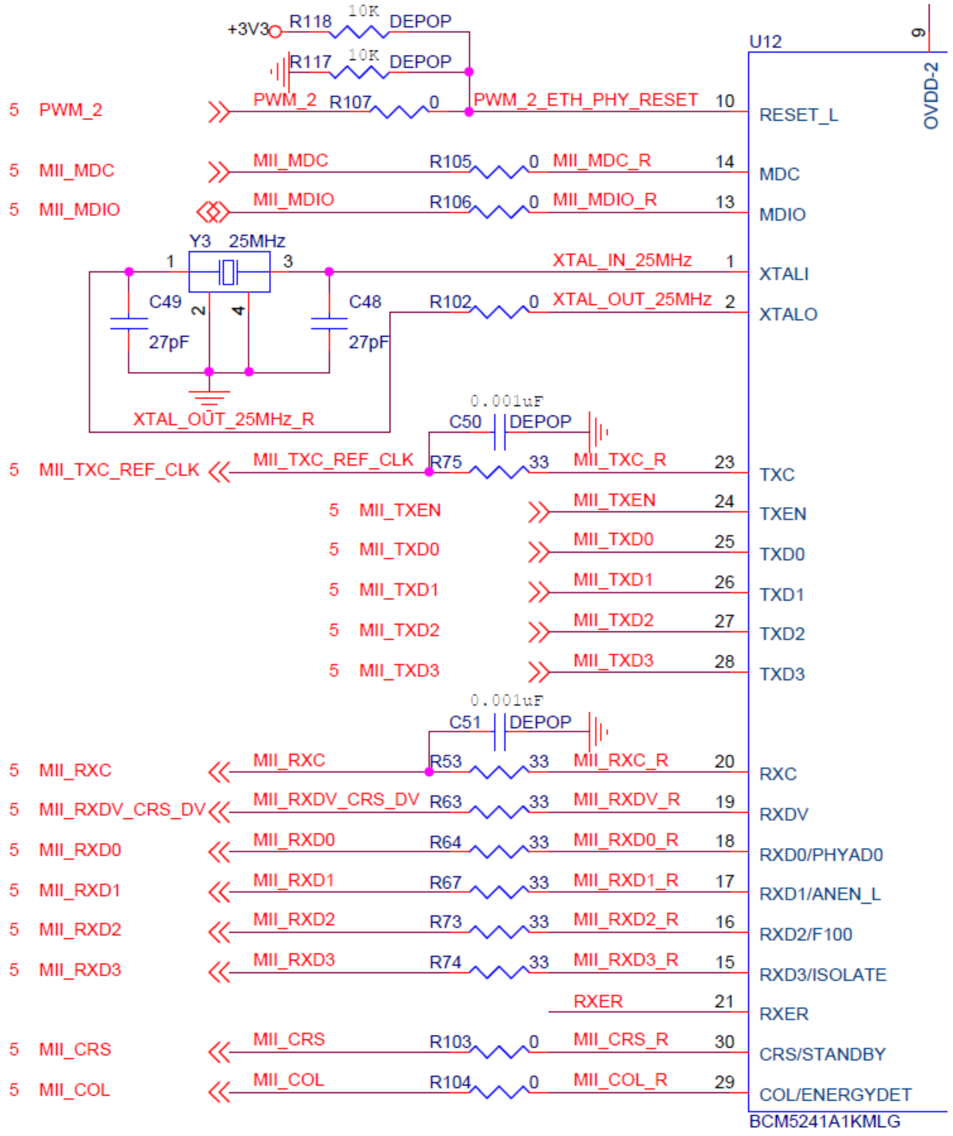
The ethernet MAC Controller in CYW43907 interfaces with an external PHY chip, BCM5241, using the Media Independent Interface (MII) as shown in Figure 4-13. The same signals are also listed in Table 4-4. CYW43907 also supports Reduced Media Independent Interface (RMII). The controller can transmit and receive data at 10 Mbps and 100 Mbps.

Table 4-4. CYW43907 EMAC to PHY Chip Connection

SL. NO	CYW43907 Pin Name	Net Name in Schematic	BCM5241 Pin Name
1	RMII_G_RXC	MII_RXC	RXC
2	RMII_G_COL	MII_COL	COL/ENERGYDET
3	RMII_G_CRS	MII_CRS	CRS/STANDBY
4	RMII_G_TXC	MII_TXC_RMII_REF_CLK	TXC
5	RMII_G_TXD0	MII_TXD0	TXD0
6	RMII_G_TXD1	MII_TXD1	TXD1
7	RMII_G_TXD2	MII_TXD2	TXD2
8	RMII_G_TXD3	MII_TXD3	TXD3

SL. NO	CYW43907 Pin Name	Net Name in Schematic	BCM5241 Pin Name
9	RMII_G_RXD0	MII_RXD0	RXD0/PHYAD0
10	RMII_G_RXD1	MII_RXD1	RXD1/ANEN_L
11	RMII_G_RXD2	MII_RXD2	RXD2/F100
12	RMII_G_RXD3	MII_RXD3	RXD3/ISOLATE
13	RMII_MDIO	MII_MDIO	MDIO
14	RMII_MDC	MII_MDC	MDC
15	RMII_G_TXEN	MII_TXEN	TXEN
16	RMII_G_RXDV	MII_RXDV_CRS_DV	RXDV
17	PWM_2	PWM_2	RESET_L

Figure 4-13. Ethernet MAC Controller to External PHY Connection



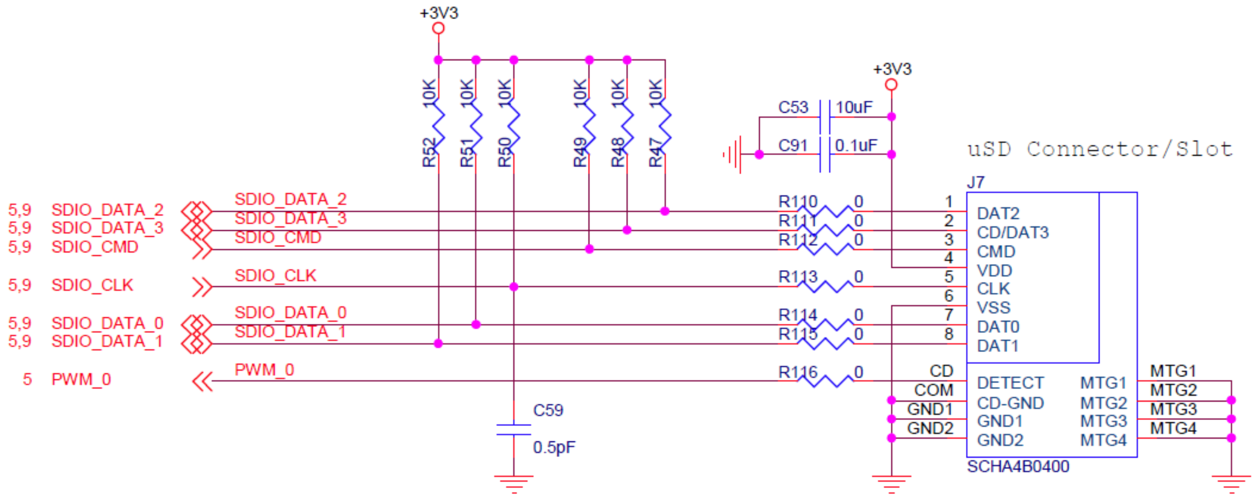
4.6 Micro SD Connector/Slot

A Micro SD connector is connected to the SDIO Interface of CYW43907. The CYW43907 device supports both SDIO 3.0 Host and device modes. Figure 4-14 shows the interface between the Micro SD connector and CYW43907. The same signals are also listed in Table 4-5.

Table 4-5. Micro SD Connector signals

SL. NO	CYW43907 Based SIP Pin Name	Micro SD Connector/Slot Name
1	SDIO_DATA_0	DAT0
2	SDIO_DATA_1	DAT1
3	SDIO_DATA_2	DAT2
4	SDIO_DATA_3	CD/DAT3
5	SDIO_CMD	CMD
6	SDIO_CLK	CLK
7	PWM_0	DETECT

Figure 4-14. Micro SD Connector Circuit Diagram



4.7 JTAG Connector

4.7.1 On-board Programmer/Debugger and Serial Interface Chip

The on-board programmer/debugger chip uses JTAG to program/debug CYW43907 based SiP module.

Table 4-6 shows the connection between CYW43907 and On-board Programmer/Debugger chip. In addition to the connections listed in the table, JTAG_SEL and GPIO_8_TAP_SEL lines have been pulled high to make sure programming/debugging is enabled through JTAG in in CYW43907.

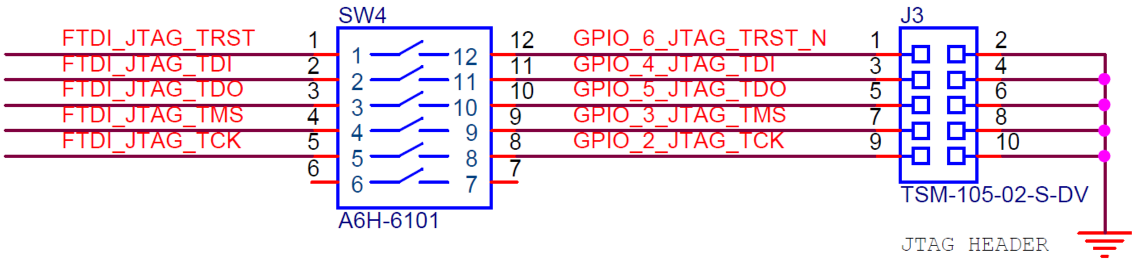
Table 4-6. Connection between CYW43907 and On-board Programmer/Debugger

SL. No	CYW43907 Based SIP Pin Name	On-board Programmer/Debugger Connection
1	GPIO_2_JTAG_TCK	FTDI_JTAG_TCK
2	GPIO_3_JTAG_TMS	FTDI_JTAG_TMS
3	GPIO_4_JTAG_TDI	FTDI_JTAG_TDI
4	GPIO_5_JTAG_TDO	FTDI_JTAG_TDO
5	GPIO_6_JTAG_TRST_L	FTDI_JTAG_TRST

4.7.2 External JTAG

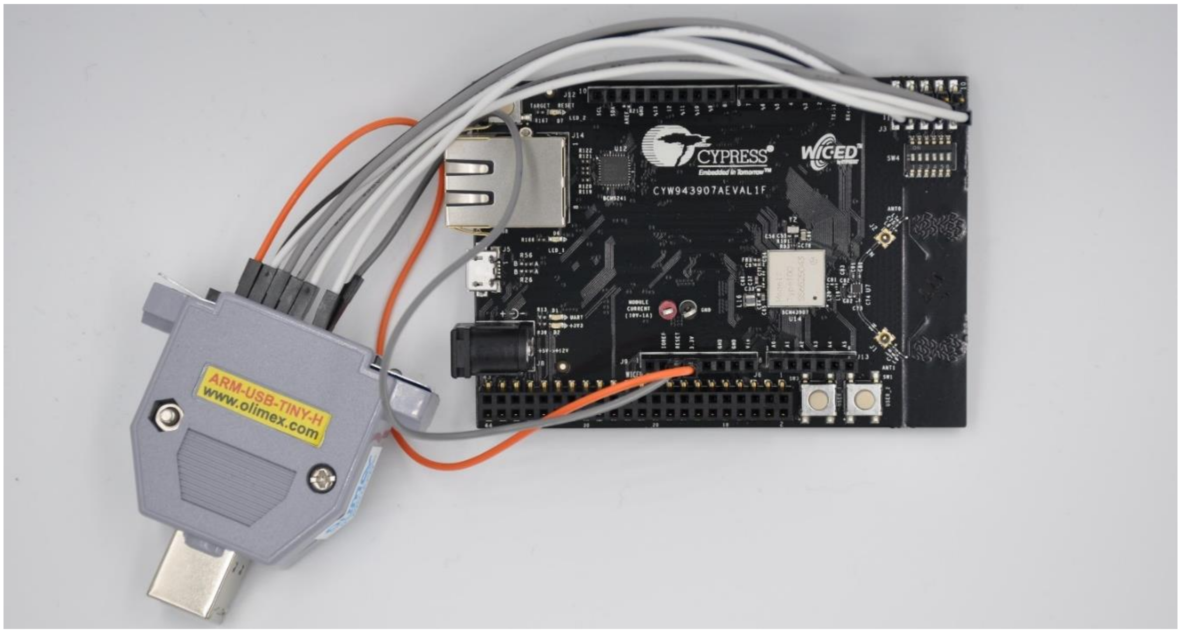
To use the external JTAG connector (J3), set all positions in switch **SW4** to closed and connect your external JTAG debugger. Ensure the drivers for the debugger hardware are installed in the same PC where WICED Studio is installed. When using Olimex connectors, such as Olimex_ARM-USB-TINY-H, add "JTAG=Olimex_ARM-USB-TINY-H" in your make target to debug. Figure 4-15 shows the relevant part of the schematic for connecting an External JTAG device. Figure 4-16 shows the connection between Olimex and the CYW943907AEVAL1F EVK.

Figure 4-15. External JTAG Connector Circuit Diagram



```
For JTAG from U4 -> SW4: All open
For External JTAG from J3 -> SW4: All Close
```

Figure 4-16. JTAG to Olimex Connection



4.8 Connectors

4.8.1 WICED Header

J6 is the WICED header available on the CYW943907AEVAL1F EVK. This is a 44-pin header containing I²C, SDIO, UART, SPI, PWM lines, and I/Os. Note that some signals are shared with the Arduino header (UART0 Tx/Rx) and On-board Programmer/debugger chip (UART1). Table 4-7 illustrates the J6 pinout.

Table 4-7. WICED Header Pinout

Eval Board Header	CYW43907 Pin Name	WICED Enumeration	Alternate Enumeration
J6.1	PWM_4	WICED_GPIO_17	WICED_PWM_5
J6.2	PWM_5	WICED_GPIO_18	WICED_BUTTON1
J6.3	I2S0_MCK	WICED_GPIO_28	WICED_I2S_1
J6.4	I2S0_SD_OUT	WICED_GPIO_32	WICED_I2S_1
J6.5	I2S0_SCK_BCLK	WICED_GPIO_29	WICED_I2S_1
J6.6	I2S0_WS_LRCLK	WICED_GPIO_30	WICED_I2S_1
J6.7	PWM_3	WICED_GPIO_16	WICED_LED1
J6.8	GND	N/A	N/A
J6.9	SPI_1_CLK	WICED_GPIO_38	WICED_SPI_2
J6.10	I2S1_SD_OUT	WICED_GPIO_37	WICED_I2S_3
J6.11	SPI_1_MISO	WICED_GPIO_39	WICED_SPI_2
J6.12	SPI_0_CLK	WICED_GPIO_20	WICED_SPI_1
J6.13	SPI_1_MOSI	WICED_GPIO_40	WICED_SPI_2
J6.14	SPI_0_MOSI	WICED_GPIO_21	WICED_SPI_1

Eval Board Header	CYW43907 Pin Name	WICED Enumeration	Alternate Enumeration
J6.15	SPI_1_CS	WICED_GPIO_41	WICED_SPI_2
J6.16	SPI_0_CS	WICED_GPIO_22	WICED_SPI_1
J6.17	SPI_0_MISO	WICED_GPIO_19	WICED_SPI_1
J6.18	UART0_RXD_IN	WICED_PERIPHERAL_PIN_3	WICED_UART_2
J6.19	GND	N/A	N/A
J6.20	UART0_TXD_OUT	WICED_PERIPHERAL_PIN_4	WICED_UART_2
J6.21	USB2_HOST_DEV_SEL	N/A	N/A
J6.22	UART0_CTS_IN	WICED_PERIPHERAL_PIN_5	WICED_UART_2
J6.23	I2C_0_SCL	WICED_GPIO_49	WICED_I2C_1
J6.24	UART0_RTS_OUT	WICED_PERIPHERAL_PIN_6	WICED_UART_2
J6.25	I2C_0_SDA	WICED_GPIO_48	WICED_I2C_1
J6.26	I2S1_MCK	WICED_GPIO_33	WICED_I2S_3
J6.27	I2S1_WS_LRCLK	WICED_GPIO_35	WICED_I2S_3
J6.28	GND	N/A	N/A
J6.29	I2S1_SCK_BCLK	WICED_GPIO_34	WICED_I2S_3
J6.30	SDIO_DATA_1	WICED_GPIO_45	N/A
J6.31	SDIO_DATA_0	WICED_GPIO_44	N/A
J6.32	SDIO_CLK	WICED_GPIO_42	N/A
J6.33	SDIO_CMD	WICED_GPIO_43	N/A
J6.34	SDIO_DATA_3	WICED_GPIO_47	N/A
J6.35	SDIO_DATA_2	WICED_GPIO_46	N/A
J6.36	RF_SW_CTRL_6_UART1_RXD	WICED_PERIPHERAL_PIN_1	WICED_UART_1
J6.37	UART1_TXD	WICED_PERIPHERAL_PIN_2	WICED_UART_1
J6.38	RF_SW_CTRL_8_UART2_RXD	WICED_PERIPHERAL_PIN_7	WICED_UART_3
J6.39	UART2_TXD	WICED_PERIPHERAL_PIN_8	WICED_UART_3
J6.40	HIB_WAKE	N/A	N/A
J6.41	HIB_LPO_SEL	N/A	N/A
J6.42	HIB_REG_ON_IN	N/A	N/A
J6.43	USB2_DN	N/A	N/A
J6.44	USB2_DP	N/A	N/A

4.8.2 Arduino-Compatible Headers

J9, J13, J12, and J10 are the Arduino headers available in the CYW943907AEVAL1F EVK. Table 4-8 shows the pinout of the Arduino Header. Note the following points while connecting an Arduino shield to the board:

- 5-V pin of Header (J9) is not connected to the board.
- The maximum current that an Arduino shield can sink from the board depends on the application that is running. In general, 100 mA is the worst-case scenario.
- The Arduino Analog reference is connected to the 3V3 (3.3 V) power supply through R21, which is not populated by default. In other words, the analog reference is not driven by default.
- An external ADC attached to CYW43907 helps to achieve analog functionality on the Arduino headers.

Table 4-8. Arduino Header Pinout

Eval Board Header	CYW43907 Pin Name/ Kit Signal Name	ARDUINO Header Name	WICED Enumeration	Alternate Enumeration
J10.1	GPIO_0	D0	WICED_GPIO_1	N/A
J10.2	GPIO_1	D1	WICED_GPIO_2	N/A
J10.3	GPIO_13	D2	WICED_GPIO_9	N/A
J10.4	GPIO_7	D3	WICED_GPIO_3	WICED_PWM_6
J10.5	GPIO_14	D4	WICED_GPIO_10	N/A
J10.6	GPIO_16	D5	WICED_GPIO_12	WICED_PWM_3
J10.7	GPIO_15	D6	WICED_GPIO_11	WICED_PWM_4
J10.8	I2S0_SD_IN	D7	WICED_GPIO_31	WICED_I2S_1
J12.1	I2S1_SD_IN	D8	WICED_GPIO_36	WICED_I2S_3
J12.2	PWM_4	D9	WICED_GPIO_17	WICED_PWM_5
J12.3	GPIO_11	D10	WICED_GPIO_7	WICED_PWM_2
J12.4	GPIO_10	D11	WICED_GPIO_6	WICED_PWM_1
J12.5	GPIO_12	D12	WICED_GPIO_8	N/A
J12.6	GPIO_9	D13	WICED_GPIO_5	WICED_LED2
J12.7	GND	GND	N/A	N/A
J12.8	ARD_AREF	AREF	N/A	N/A
J12.9	I2C_1_SDA	SDA	WICED_GPIO_50	WICED_I2C_2
J12.10	I2C_1_SCL	SCL	WICED_GPIO_51	WICED_I2C_2
J13.1	ARD_AD0	A0	N/A	N/A
J13.2	ARD_AD1	A1	N/A	N/A
J13.3	ARD_AD2	A2	N/A	N/A
J13.4	ARD_AD3	A3	N/A	N/A
J13.5	ARD_AD4_SDA	A4	N/A	N/A
J13.6	ARD_AD5_SCL	A5	N/A	N/A
J9.1	NC	NC	N/A	N/A
J9.2	ARD_IOREF	IOREF	N/A	N/A
J9.3	ARD_RESET	RESET	N/A	N/A
J9.4	3V3	3.3V	N/A	N/A
J9.5	NC	5V	N/A	N/A

Eval Board Header	CYW43907 Pin Name/ Kit Signal Name	ARDUINO Header Name	WICED Enumeration	Alternate Enumeration
J9.6	GND	GND	N/A	N/A
J9.7	GND	GND	N/A	N/A
J9.8	VIN_EXT	VIN	N/A	N/A

4.9 UART Port Configuration on CYW943907AEVAL1F Kit

The CYW43907 has three UART ports: slow UART, fast UART, and GCI UART. Slow UART and GCI UART are 2-wire interfaces while fast UART is a 4-wire interface that can support up to a 3 Mbps baud rate. Slow UART is routed to the On-board Programmer/debugger chip for UART to USB communication. The UART peripherals are defined in *platforms/CYW943907AEVAL1F/platform.c*. The following table (also available in *platforms/CYW943907AEVAL1F/platform.h*) shows the UART pins available on the Kit.

WICED Peripheral Enumeration ID	Pin Name on CYW43907	MURATA Module Pin Name	Header Pin Number	WICED Enumeration
WICED_PERIPHERAL_PIN_1	RF_SW_CTRL_6	RF_SW_CTRL_6_UART1_RXD	J6:36	WICED_UART_1
WICED_PERIPHERAL_PIN_2	RF_SW_CTRL_7	RF_SW_CTRL_7_UART1_TXD	J6:37	WICED_UART_1
WICED_PERIPHERAL_PIN_3	UART0_RXD	UART0_RXD_IN	J6:18	WICED_UART_2
WICED_PERIPHERAL_PIN_4	UART0_TXD	UART0_TXD_OUT	J6:20	WICED_UART_2
WICED_PERIPHERAL_PIN_5	UART0_CTS	UART0_CTS_IN	J6:22	WICED_UART_2
WICED_PERIPHERAL_PIN_6	UART0_RTS	UART0_RTS_OUT	J6:24	WICED_UART_2
WICED_PERIPHERAL_PIN_7	RF_SW_CTRL_8	RF_SW_CTRL_8_UART2_RXD	J6:38	WICED_UART_3
WICED_PERIPHERAL_PIN_8	RF_SW_CTRL_9	RF_SW_CTRL_9_UART2_TXD	J6:39	WICED_UART_3

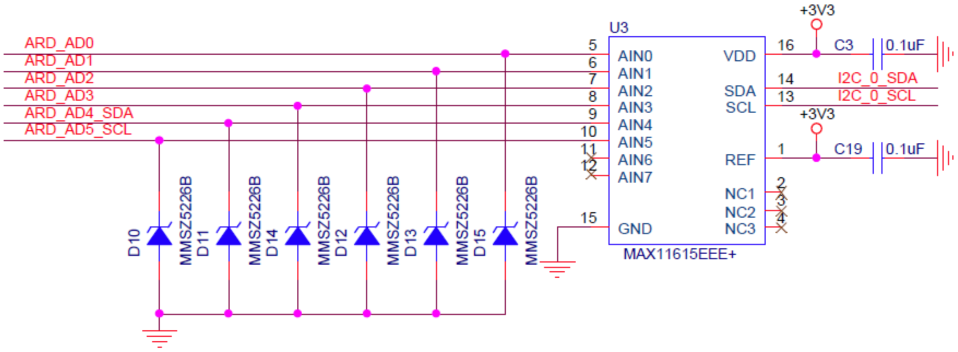
4.10 External ADC

CYW43907 does not have any in-built ADC block. Analog measurements from the Arduino header analog pins is achieved using an external ADC chip (MAX11615) connected to CYW43907 through an I2C interface (I2C_0 module-Slave Address 0x33). Table 4-9 lists the connections between CYW43907 and the external ADC Circuit diagram is shown in Figure 4-17.

Table 4-9. External ADC Connection

I2C Line	CYW43907 Pin Name	WICED Enumeration	Alternate Enumeration
SDA	I2C_0_SDA	WICED_GPIO_48	WICED_I2C_1
SCL	I2C_0_SCL	WICED_GPIO_49	WICED_I2C_1

Figure 4-17. External ADC Circuit Diagram



4.11 PWM

There are six dedicated PWM outputs available on CYW43907. These PWMs can be multiplexed onto different pins. You can find their definitions in *platforms/CYW943907AEVAL1F/platform.c* inside WICED Studio. The PWMs can be re-assigned to other pins by changing the first argument of the `platform_pwm_t platform_pwm_peripherals` structure in *platform.c*. Table 4-10 through Table 4-15 show the possible combinations and their Arduino header locations.

Table 4-10. WICED_PWM_1 Combinations

Pin MUX Selection	Header Pin	Header Name
PIN_GPIO_10 (DEFAULT)	J12.4	Arduino D11 (MOSI)
PIN_GPIO_0	J10.1	Arduino D0
PIN_GPIO_8	_	–
PIN_GPIO_12	J12.5	Arduino D12 (MISO)
PIN_GPIO_14	–	–
PIN_GPIO_16	J10.6	Arduino D5
PIN_PWM_0	–	–

Table 4-11. WICED_PWM_2 Combinations

Pin MUX Selection	Header Pin	Header Name
PIN_GPIO_11 (DEFAULT)	J12.3	Arduino D10
PIN_GPIO_1	J10.1	Arduino D0
PIN_GPIO_7	J10.4	Arduino D3
PIN_GPIO_9	J12.9	Arduino SCK
PIN_GPIO_13	J10.3	Arduino D2
PIN_GPIO_15	J10.7	Arduino D6
PIN_PWM_1	–	–

Table 4-12. WICED_PWM_3 Combinations

Pin MUX Selection	Header Pin	Header Name
PIN_GPIO_16 (DEFAULT)	J10.6	Arduino D5
PIN_GPIO_8	–	–
PIN_GPIO_0	J10.1	Arduino D0
PIN_GPIO_10	J12.4	Arduino D11 (MOSI)
PIN_GPIO_12	J12.5	Arduino D12 (MISO)
PIN_GPIO_14	–	–
PIN_PWM_2	–	–

Table 4-13. WICED_PWM_4 Combinations

Pin MUX Selection	Header Pin	Header Name
PIN_GPIO_15 (DEFAULT)	J10.7	Arduino D6
PIN_GPIO_1	J10.1	Arduino D0
PIN_GPIO_7	J10.4	Arduino D3
PIN_GPIO_9	J12.9	Arduino SCK
PIN_GPIO_11	J12.3	Arduino D10
PIN_GPIO_13	J10.3	Arduino D2
PIN_PWM_3	–	–

Table 4-14. WICED_PWM_5 Combinations

Pin MUX Selection	Header Pin	Header Name
PIN_PWM_4 (DEFAULT)	J6.1	Arduino A1
PIN_GPIO_0	J10.1	Arduino D0
PIN_GPIO_8	–	–
PIN_GPIO_10	J12.4	Arduino D11 (MOSI)
PIN_GPIO_12	J12.5	Arduino D12 (MISO)
PIN_GPIO_14	–	–
PIN_GPIO_16	J10.6	Arduino D5

Table 4-15. WICED_PWM_6 Combinations

Pin MUX Selection	Header Pin	Header Name
PIN_GPIO_7 (DEFAULT)	J10.4.4	Arduino D3
PIN_GPIO_1	J10.1	Arduino D0
PIN_GPIO_9	J12.9	Arduino SCK
PIN_GPIO_11	J12.3	Arduino D10
PIN_GPIO_13	J10.3	Arduino D2
PIN_GPIO_15	J10.7	Arduino D6
PIN_PWM_5	–	–

5. Code Examples

This chapter demonstrates the functionality of CYW43907 devices using the CYW943907AEVAL1F EVK code examples. Download and extract the zip file from the CYW943907AEVAL1F EVK web page as specified in the Install Software section. The code examples once un-zipped can be viewed in WICED Studio 5.0 (or later). In addition to the added examples there are already many apps (snip.gpio, test.console, and so on) that are available in WICED Studio 5.0.

5.1 Using Code Examples

Code examples already added can be compiled after creating Make Targets. See Building and Programming a Project for CYW943907AEVAL1F in WICED Studio IDE for the process of creating targets.

Create the following four Make Targets in WICED Studio 5.0 (or later):

- "`snip.gpio-CYW943907AEVAL1F download run`" for the gpio example, which is already present in WICED Studio.
- "`kits.CYW943907AEVAL1F.config_join_ping-CYW943907AEVAL1F download run`" for the config_join_ping project.
- "`kits.CYW943907AEVAL1F.publish_subscribe_aws-CYW943907AEVAL1F download run`" for the aws publish and subscribe project.
- "`kits.CYW943907AEVAL1F.adc_measure-CYW943907AEVAL1F download run`" for the adc_measure project.

5.2 GPIO

5.2.1 Project Description

The `gpio` project demonstrates toggling of LEDs and turning them off when one of the User switches is pressed.

The `gpio` project consists of the following files:

- *gpio.c:* This file contains the main application function **`application_start()`**, which is the entry point and execution of the firmware application.
- *gpio.mk:* This is the makefile which adds the source of the application.

5.2.2 Hardware Connections

No specific hardware connections are required for this project because all connections are hardwired on the CYW943907AEVAL1F EVK.

5.2.3 Verify Output

Perform these steps to verify the output:

Create and run a Make Target for the `gpio` project using the description specified in Building and Programming a Project for CYW943907AEVAL1F in WICED Studio IDE.

After initialization of the platform, LEDs will keep flashing (toggling). When the user switch is pressed, the corresponding LED turns off.

The example also prints a message to the debug UART at startup. Open a Terminal Emulation program and connect to the WICED serial port as detailed in Step 8 in the section UART Port Configuration on CYW943907AEVAL1F Kit to see the message printed at startup.

5.3 Config_join_ping

5.3.1 Project Description

The config_join_ping project demonstrates connectivity between CYW943907AEVAL1F EVK and a Wi-Fi access point. This example is based on existing examples available in the WICED Studio 5.0 (or later) SDK namely, *apps/snip/scan, apps/snip/dct_read_write* and *test/console*. On startup, this application shows a console through which the user can enter commands to scan, configure, join, and ping Wi-Fi access points.

The *config_join_ping* project consists of the following files:

- *config_join_ping.c*: This file contains the main application function `application_start()`, which is the entry point and execution of the firmware application. It also contains the function definitions for joining, pinging, printing Wi-Fi configuration, scanning Wi-Fi, and the scan result handler.
- *config_join_ping.mk*: This is the makefile, which adds the sources, components (in this application, console and ping are used), and the name of the application. Note that the name of the makefile must match the name of the project folder for the make process to work properly. Also, the “NAME” string in the makefile must be unique among all projects in the *apps* folder.

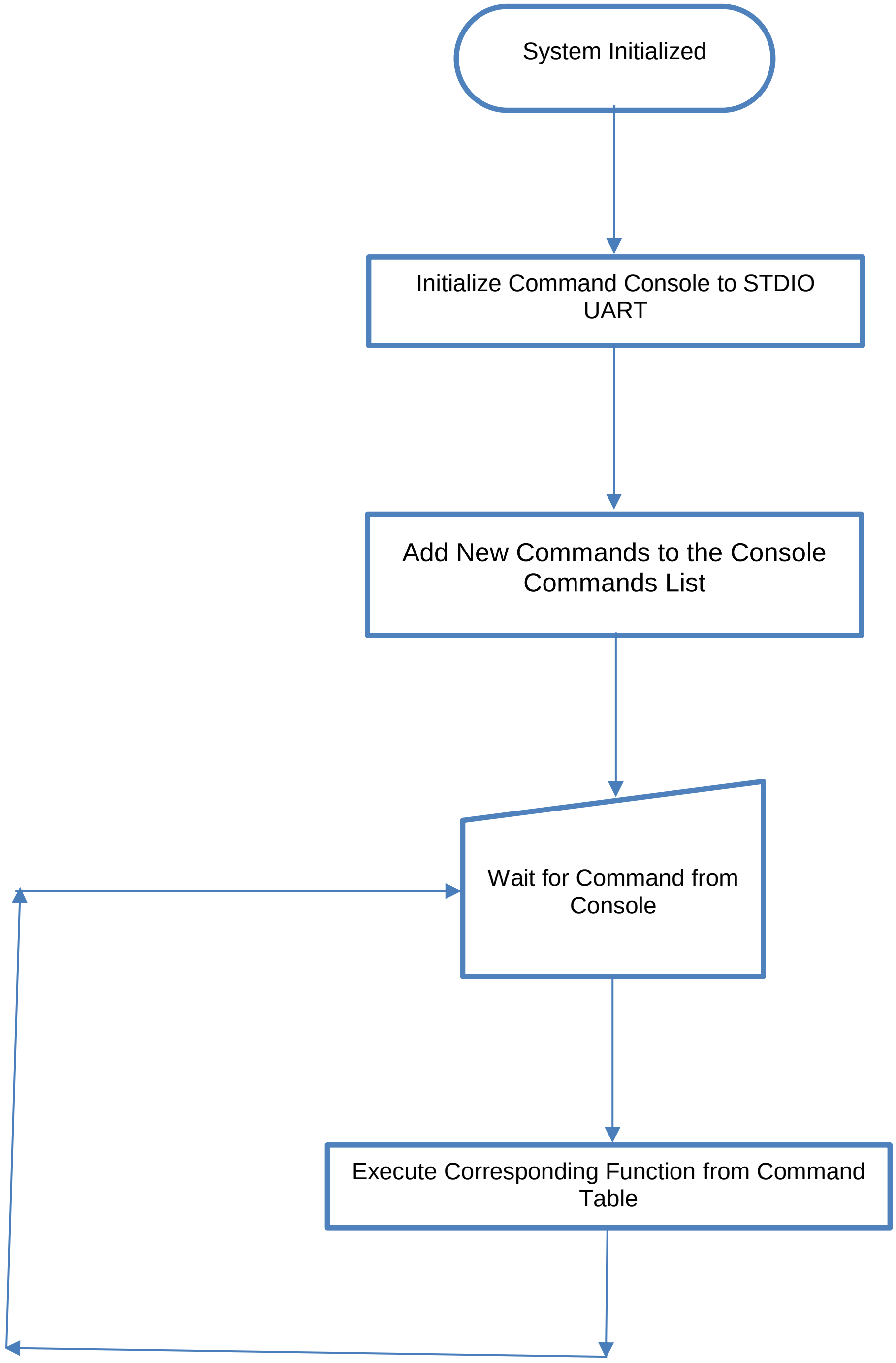
5.3.2 Hardware Connections

No specific hardware connections are required for this project because all connections are hardwired on the CYW943907AEVAL1F EVK.

5.3.3 Flowchart

Figure 5-1 illustrates the *config_join_ping* flowchart.

Figure 5-1. config_join_ping Flowchart


System Initialized
Initialize Command Console to STDIO UART
Add New Commands to the Console Commands List
Wait for Command from Console
Execute Corresponding Function from Command Table

5.3.4 Verify Output

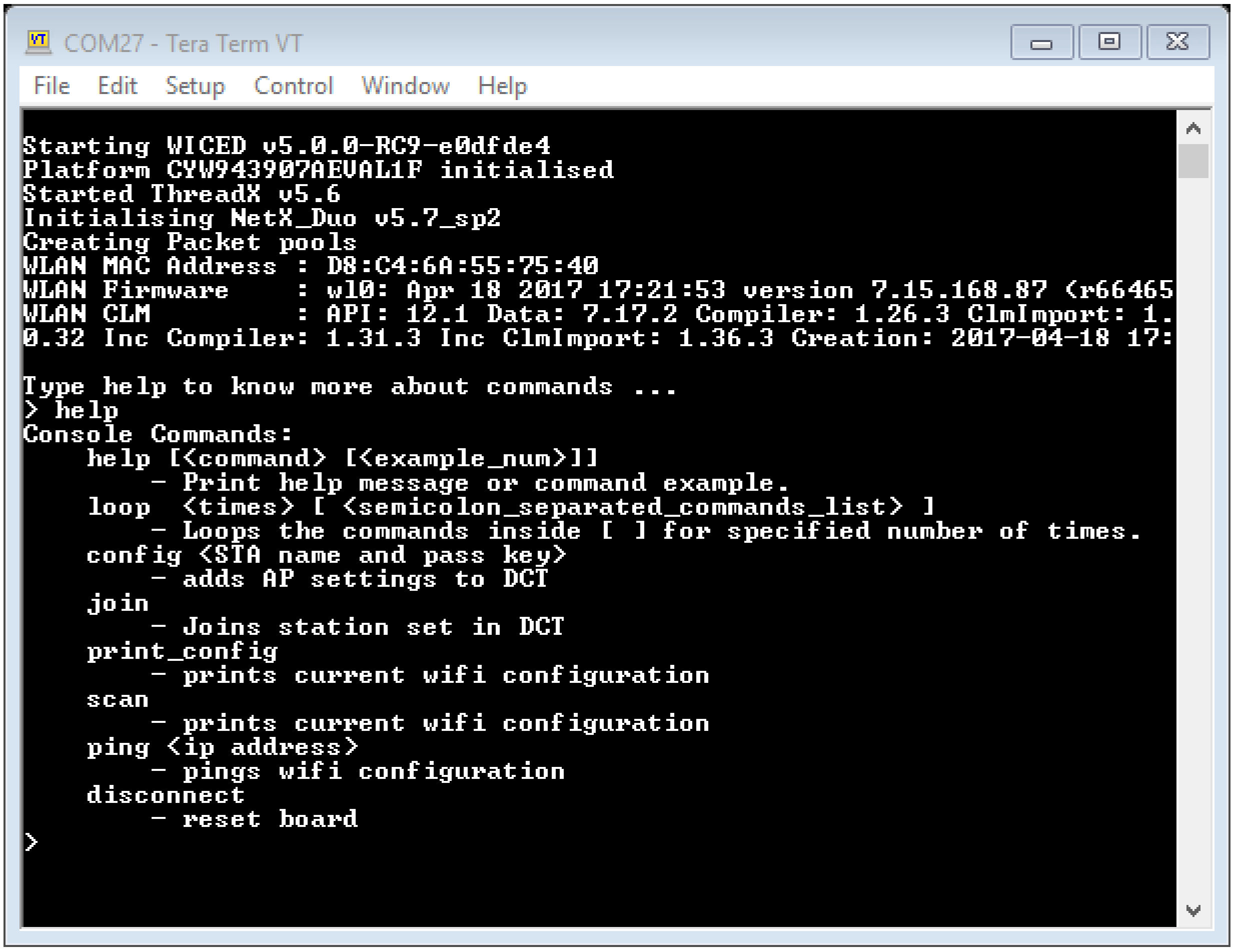
Perform these steps to verify the output:

Create and run a Make Target for the *config_join_ping* project using the description specified in Building and Programming a Project for CYW943907AEVAL1F in WICED Studio IDE. Open a Terminal Emulation program and connect to the WICED serial port as detailed in Step 8 in the section UART Port Configuration on CYW943907AEVAL1F Kit.

After initialization of the platform, Wi-Fi, and other components, the cursor will stop and wait for you to enter commands.

Type the `help` command to see the list of available commands as shown in Figure 5-2.

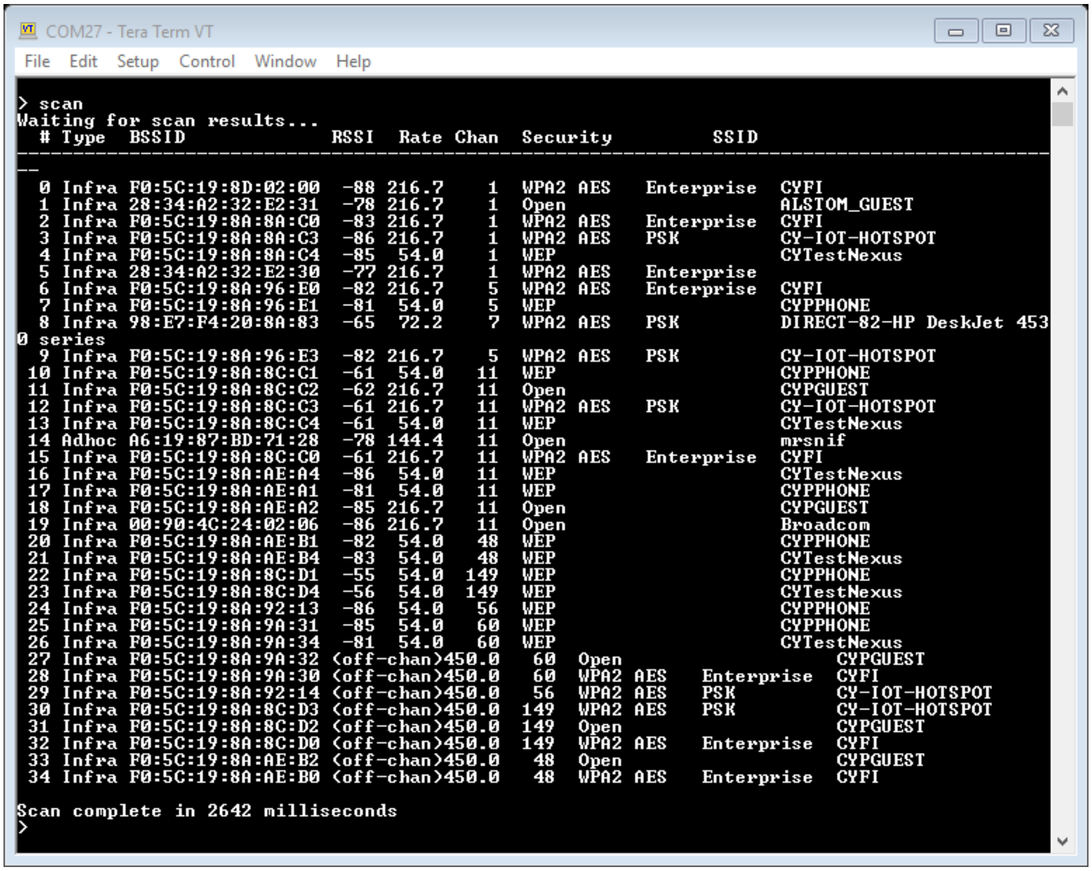
Figure 5-2. Initial Console Output



```
Starting WICED v5.0.0-RC9-e0dfde4
Platform CYW943907AEVAL1F initialised
Started ThreadX v5.6
Initialising NetX_Duo v5.7_sp2
Creating Packet pools
WLAN MAC Address : D8:C4:6A:55:75:40
WLAN Firmware    : wl0: Apr 18 2017 17:21:53 version 7.15.168.87 (r66465
WLAN CLM         : API: 12.1 Data: 7.17.2 Compiler: 1.26.3 ClmImport: 1.
0.32 Inc Compiler: 1.31.3 Inc ClmImport: 1.36.3 Creation: 2017-04-18 17:

Type help to know more about commands ...
> help
Console Commands:
    help [<command> [<example_num>]]
        - Print help message or command example.
    loop  <times> [ <semicolon_separated_commands_list> ]
        - Loops the commands inside [ ] for specified number of times.
    config <STA name and pass key>
        - adds AP settings to DCT
    join
        - Joins station set in DCT
    print_config
        - prints current wifi configuration
    scan
        - prints current wifi configuration
    ping <ip address>
        - pings wifi configuration
    disconnect
        - reset board
>
```

Type the `scan` command to find the list of available Wi-Fi access points as shown in Figure 5-3.

Figure 5-3. Scan Output



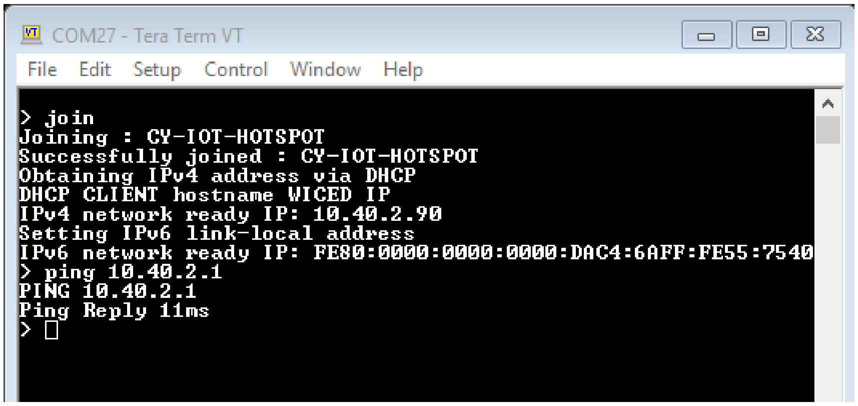
```
> scan
Waiting for scan results...
  # Type  BSSID             RSSI  Rate Chan  Security         SSID
--------------------------------------------------------------------------------------------
--
  0 Infra F0:5C:19:8D:02:00  -88 216.7    1  WPA2 AES   Enterprise  CYFI
  1 Infra 28:34:A2:32:E2:31  -78 216.7    1  Open                   ALSTOM_GUEST
  2 Infra F0:5C:19:8A:8A:C0  -83 216.7    1  WPA2 AES   Enterprise  CYFI
  3 Infra F0:5C:19:8A:8A:C3  -86 216.7    1  WPA2 AES   PSK         CY-IOT-HOTSPOT
  4 Infra F0:5C:19:8A:8A:C4  -85  54.0    1  WEP                    CYTestNexus
  5 Infra 28:34:A2:32:E2:30  -77 216.7    1  WPA2 AES   Enterprise
  6 Infra F0:5C:19:8A:96:E0  -82 216.7    5  WPA2 AES   Enterprise  CYFI
  7 Infra F0:5C:19:8A:96:E1  -81  54.0    5  WEP                    CYPPHONE
  8 Infra 98:E7:F4:20:8A:83  -65  72.2    7  WPA2 AES   PSK         DIRECT-82-HP DeskJet 453
0 series
  9 Infra F0:5C:19:8A:96:E3  -82 216.7    5  WPA2 AES   PSK         CY-IOT-HOTSPOT
 10 Infra F0:5C:19:8A:8C:C1  -61  54.0   11  WEP                    CYPPHONE
 11 Infra F0:5C:19:8A:8C:C2  -62 216.7   11  Open                   CYPGUEST
 12 Infra F0:5C:19:8A:8C:C3  -61 216.7   11  WPA2 AES   PSK         CY-IOT-HOTSPOT
 13 Infra F0:5C:19:8A:8C:C4  -61  54.0   11  WEP                    CYTestNexus
 14 Adhoc A6:19:87:BD:71:28  -78 144.4   11  Open                   mrsnif
 15 Infra F0:5C:19:8A:8C:C0  -61 216.7   11  WPA2 AES   Enterprise  CYFI
 16 Infra F0:5C:19:8A:AE:A4  -86  54.0   11  WEP                    CYTestNexus
 17 Infra F0:5C:19:8A:AE:A1  -81  54.0   11  WEP                    CYPPHONE
 18 Infra F0:5C:19:8A:AE:A2  -85 216.7   11  Open                   CYPGUEST
 19 Infra 00:90:4C:24:02:06  -86 216.7   11  Open                   Broadcom
 20 Infra F0:5C:19:8A:AE:B1  -82  54.0   48  WEP                    CYPPHONE
 21 Infra F0:5C:19:8A:AE:B4  -83  54.0   48  WEP                    CYTestNexus
 22 Infra F0:5C:19:8A:8C:D1  -55  54.0  149  WEP                    CYPPHONE
 23 Infra F0:5C:19:8A:8C:D4  -56  54.0  149  WEP                    CYTestNexus
 24 Infra F0:5C:19:8A:92:13  -86  54.0   56  WEP                    CYPPHONE
 25 Infra F0:5C:19:8A:9A:31  -85  54.0   60  WEP                    CYPPHONE
 26 Infra F0:5C:19:8A:9A:34  -81  54.0   60  WEP                    CYTestNexus
 27 Infra F0:5C:19:8A:9A:32 <off-chan>450.0   60  Open                   CYPGUEST
 28 Infra F0:5C:19:8A:9A:30 <off-chan>450.0   60  WPA2 AES   Enterprise  CYFI
 29 Infra F0:5C:19:8A:92:14 <off-chan>450.0   56  WPA2 AES   PSK         CY-IOT-HOTSPOT
 30 Infra F0:5C:19:8A:8C:D3 <off-chan>450.0  149  WPA2 AES   PSK         CY-IOT-HOTSPOT
 31 Infra F0:5C:19:8A:8C:D2 <off-chan>450.0  149  Open                   CYPGUEST
 32 Infra F0:5C:19:8A:8C:D0 <off-chan>450.0  149  WPA2 AES   Enterprise  CYFI
 33 Infra F0:5C:19:8A:AE:B2 <off-chan>450.0   48  Open                   CYPGUEST
 34 Infra F0:5C:19:8A:AE:B0 <off-chan>450.0   48  WPA2 AES   Enterprise  CYFI

Scan complete in 2642 milliseconds
>
```

Type the `config <SSID> <password>` command. This command writes the given configuration in the Device Configuration Table (DCT). These values are stored in flash memory on the board.

Type the `print_config` command to validate if the SSID and password match and are appropriately written in the DCT.

Type the `join` command. The `join` command joins the network specified by the SSID and password from the DCT. Ping the Access Point (usually 192.168.1.1) or 8.8.8.8 (IP address of Google, if your AP is connected to internet) and check if the network is up and responding. The message “`Ping Reply 11ms`” is displayed as shown in Figure 5-4.

Figure 5-4. Join and Ping



```
> join
Joining : CY-IOT-HOTSPOT
Successfully joined : CY-IOT-HOTSPOT
Obtaining IPv4 address via DHCP
DHCP CLIENT hostname WICED IP
IPv4 network ready IP: 10.40.2.90
Setting IPv6 link-local address
IPv6 network ready IP: FE80:0000:0000:0000:DAC4:6AFF:FE55:7540
> ping 10.40.2.1
PING 10.40.2.1
Ping Reply 11ms
> 
```

To disconnect from the access point, use the command `disconnect`.

The console component maintains a history of commands typed, which can be accessed using the **Up/Down** arrow keys.

5.4 Adc_measure

5.4.1 Project Description

This project demonstrates measuring values from the external ADC chip on the board and posting the values to a web page accessible from the WLAN network. This code example is based on an existing code example (*apps/demo/temp_control*) available in WICED Studio 5.0 (or later). On startup, the adc_measure code example joins the Wi-Fi Access Point specified in the *wifi_config_dct.h* file and starts a web page where the ADC count is reported.

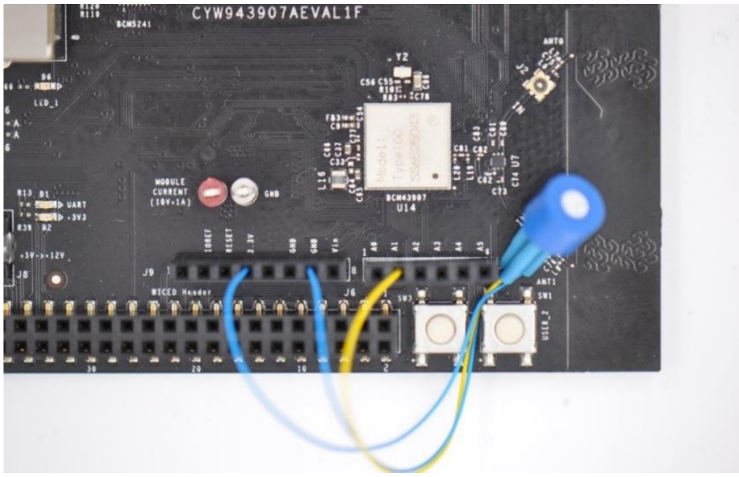
The project consists of the following files:

- *adc_measure.c*: This file contains the main application function `application_start(),` which is the entry point and execution of the firmware application. It also contains the function definitions for initializing, conducting ADC measurement, starting the web page, and processing an ADC update.
- *adc_measure.m*k: This is the makefile which adds the sources, components (in this application, component HTTP_server, device_configuration, Xively, SNTP and Gedday are used), and the name of the application. It also adds the required resources for the web page, which is available in the *resources/apps* directory.
- *i2c.c*: This file contains the required function definitions for initializing and taking ADC samples from the external ADC (MAX 11615) available in the CY9W43907AEVAL1F EVK.
- *wifi_config_dct.h*: This file contains the Wi-Fi Access Point credentials (SSID and pass phrase key) and soft AP credentials. Enter the client access point name and password credentials prior to building the application. These are specified as CLIENT_AP_SSID and CLIENT_AP_PASSPHRASE. Note that the security type may also have to be changed if the access point does not use WPA2 security. The Wi-Fi access point must be connected to the internet to get the current time using Network Time protocol (NTP). If the Wi-Fi access point is not connected to the internet, then it will assume 00:00:00 UTC time and will start the web page.

5.4.2 Hardware Connections

Connect a potentiometer (10 kΩ) between VCC and GND with the center terminal connected to channel 1 of the ADC (pin A1 in the Arduino header) as shown in Figure 5-5. If you do not have a potentiometer to test, then you can connect a wire between VCC and ADC channel 2 (to simulate full scale) or a wire between GND and ADC channel 1 (to simulate zero scale). Alternately, you can connect an adjustable DC power supply to ADC channel 1.

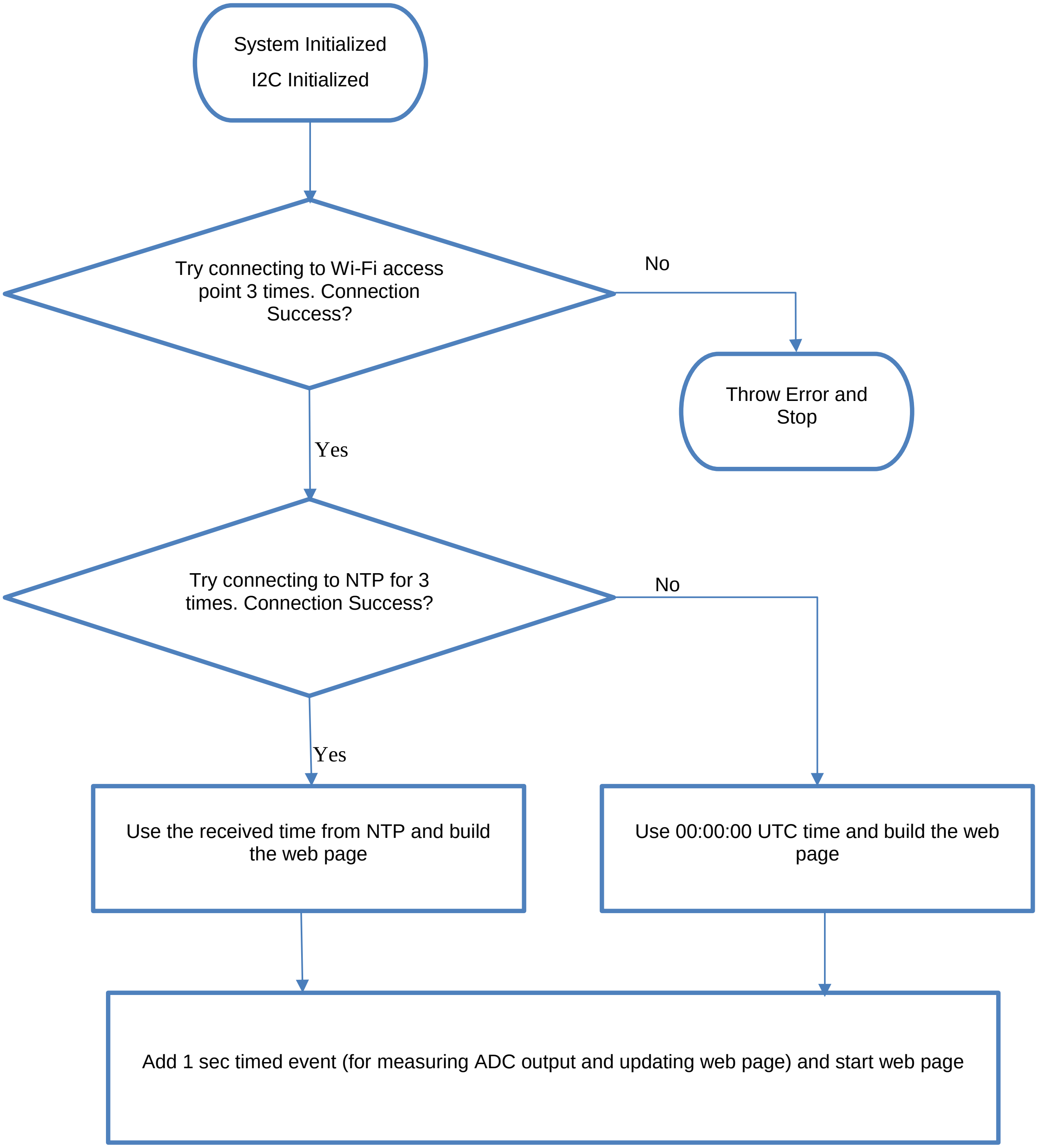
Figure 5-5. Potentiometer Connection



5.4.3 Flowchart

Figure 5-6 illustrates the adc_measure flowchart.

Figure 5-6. adc_measure Flowchart


System Initialized
I2C Initialized
Try connecting to Wi-Fi access point 3 times. Connection Success?
No
Throw Error and Stop
Yes
Try connecting to NTP for 3 times. Connection Success?
No
Yes
Use the received time from NTP and build the web page
Use 00:00:00 UTC time and build the web page
Add 1 sec timed event (for measuring ADC output and updating web page) and start web page

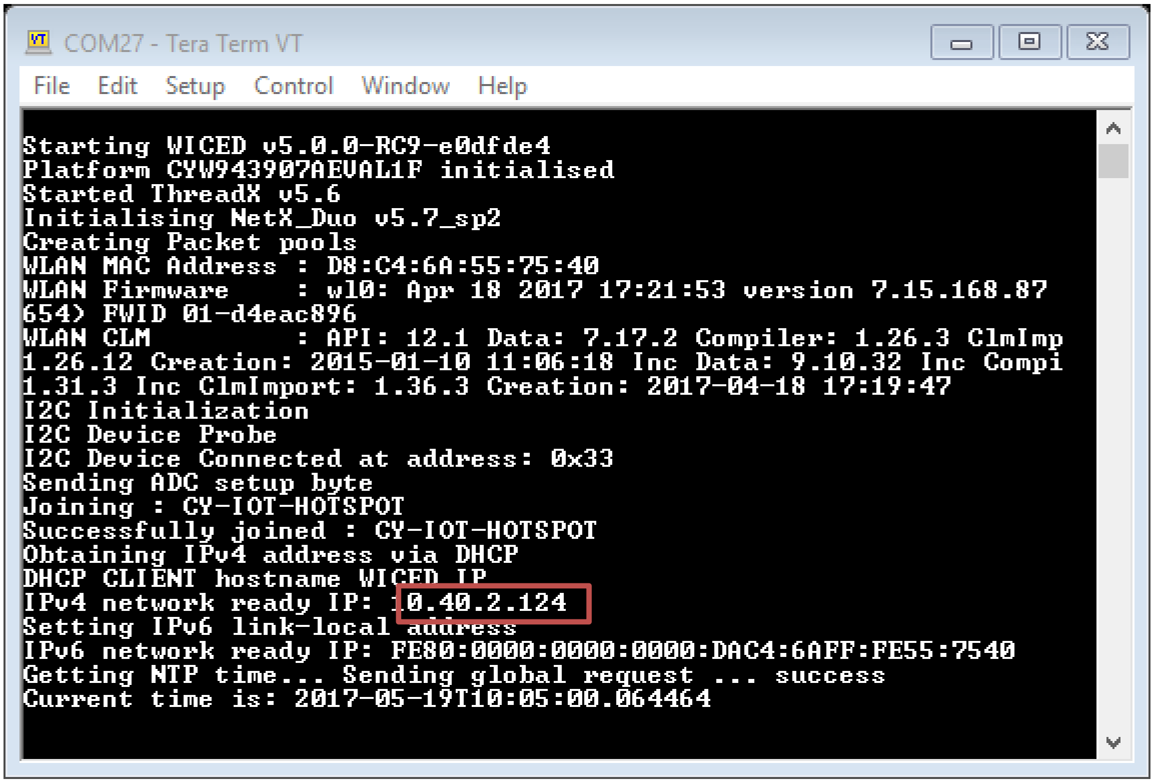
5.4.4 Access Point Credentials

Enter your credentials (SSID and pass phrase key) in the *wifi_config_dct.h* file. The `CLIENT_AP_SSID` macro should be updated with your access point's SSID, the `CLIENT_AP_PASSPHRASE` macro should be updated with your access point's pass phrase key, and the `CLIENT_AP_SECURITY` macro should be updated with the security type of your access point. This is `WICED_SECURITY_WPA2_MIXED_PSK` if the access point uses WPA2-PSK. If the AP uses different security, then choose the correct one defined in enum `wiced_security_t` from *43xxx_Wi-Fi\WICED\WWD\include\wwd_constants.h*.

5.4.5 Verify Output

Create and run a Make Target for the *adc_measure* project, similar to the procedure provided in Building, Programming, and Debugging CYW943907AEVAL1F EVK. If connection to the Wi-Fi access point is successful, then wait for NTP time request to complete. Output of the terminal program should be similar to the screenshot in Figure 5-7.

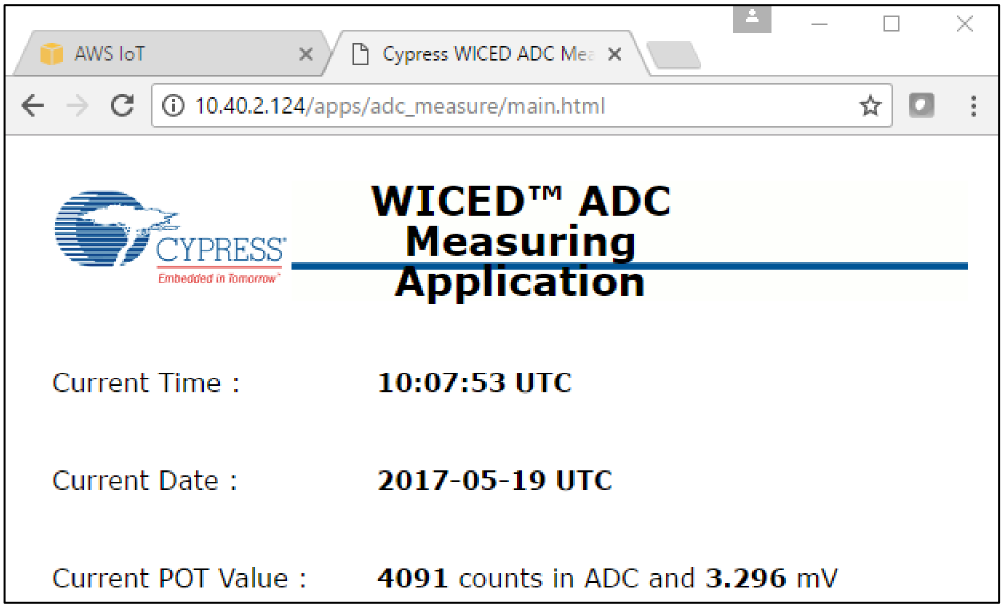
Figure 5-7. NTP Success



```
Starting WICED v5.0.0-RC9-e0dfde4
Platform CYW943907AEVAL1F initialised
Started ThreadX v5.6
Initialising NetX_Duo v5.7_sp2
Creating Packet pools
WLAN MAC Address : D8:C4:6A:55:75:40
WLAN Firmware    : wl0: Apr 18 2017 17:21:53 version 7.15.168.87
654) FWID 01-d4eac896
WLAN CLM         : API: 12.1 Data: 7.17.2 Compiler: 1.26.3 ClmImp
1.26.12 Creation: 2015-01-10 11:06:18 Inc Data: 9.10.32 Inc Compi
1.31.3 Inc ClmImport: 1.36.3 Creation: 2017-04-18 17:19:47
I2C Initialization
I2C Device Probe
I2C Device Connected at address: 0x33
Sending ADC setup byte
Joining : CY-IOT-HOTSPOT
Successfully joined : CY-IOT-HOTSPOT
Obtaining IPv4 address via DHCP
DHCP CLIENT hostname WICED IP
IPv4 network ready IP: 10.40.2.124
Setting IPv6 link-local address
IPv6 network ready IP: FE80:0000:0000:0000:DAC4:6AFF:FE55:7540
Getting NTP time... Sending global request ... success
Current time is: 2017-05-19T10:05:00.064464
```

Enter the IP address as the URL in your web browser, as shown in the terminal output in Figure 5-7 (for example, 10.40.2.124). The browser will show the output as shown in Figure 5-8. Note that the PC and CYW943907AEVAL1F EVK should be connected to the same access point. Rotate the potentiometer and verify that the value shown on the web page changes accordingly. One easy way to validate the correct functioning is to rotate the potentiometer to one of the extremes and observe if the full scale value appears. In case you do not have access to a potentiometer, then you can use an adjustable power supply or wires to connect 3.3 V and GND to the ADC input alternatively.

Figure 5-8. Webpage



WICED™ ADC Measuring Application

Current Time : 10:07:53 UTC

Current Date : 2017-05-19 UTC

Current POT Value : **4091** counts in ADC and **3.296** mV

5.5 Publish_subscribe_aws

5.5.1 Project Description

This project demonstrates publishing a message to a *Thing* in the Amazon Web Services (AWS) cloud and subscribing to the same messages. A *Thing* is a representation of a specific device or logical entity. For more information, refer to the AWS Documentation. This example is based on an existing code example (*apps/demo/aws_iot/pub_sub/publisher* and *apps/demo/aws_iot/pub_sub/subscriber*) available from WICED Studio 5.0 (till WICED Studio 6.1). On startup, the publish_subscribe_aws code example joins a Wi-Fi access point specified in the *wifi_config_dct.h* file, connects to AWS, subscribes to the specified topic, and then alternately tries to publish LIGHT ON and LIGHT OFF messages.

Note: In WICED Studio 6.2 and later, the above projects are moved to the following location: *apps/demo/aws/iot/pub_sub/publisher* and *apps/demo/aws/iot/pub_sub/subscriber*.

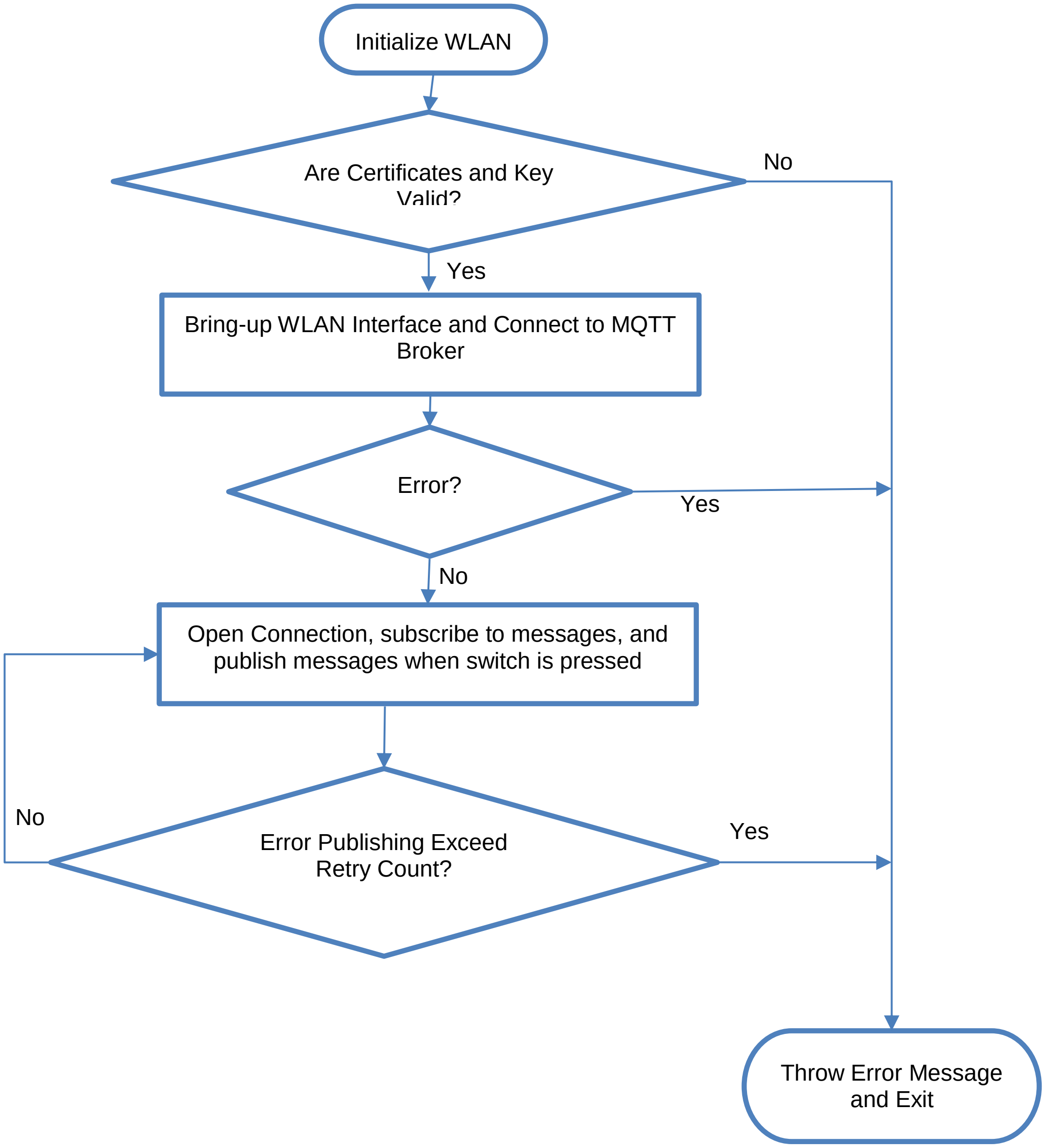
The project consists of the following files:

- *publish_subscribe.c*: This file contains the main application function `application_start()`, which is the entry point and execution of the firmware application. It also contains the function definitions for initializing, publishing, and subscribing to AWS.
- *publish_subscribe_aws.mk*: This is the makefile which adds the sources, protocols, components (in this application, the MQTT component is used), and the name of the application. It also adds the required resources for the web page, which are available in the *resources/apps* directory. Note that this project uses certificates from *apps/aws_iot* directory.
- *wifi_config_dct.h*: This file contains the Wi-Fi access point credentials (SSID and pass phrase key) and soft AP credentials. The user should enter the client access point name and password credentials prior to building the application. These are specified as CLIENT_AP_SSID and CLIENT_AP_PASSPHRASE. Note that the security type may also have to be changed if the access point does not use WPA2 security. The Wi-Fi access point must have access to the internet to connect with AWS.

5.5.2 Hardware Connections

No specific hardware connections are required for this project because all connections are hardwired on the CYW943907AEVAL1F EVK.

5.5.3 Flowchart


Initialize WLAN
Are Certificates and Key Valid?
No
Yes
Bring-up WLAN Interface and Connect to MQTT Broker
Error?
Yes
No
Open Connection, subscribe to messages, and publish messages when switch is pressed
No
Error Publishing Exceed Retry Count?
Yes
Throw Error Message and Exit

5.5.4 Verify Output

5.5.4.1 Set up an AWS Account and Create a Thing, Policy, and Certificate

An AWS account allows you to view AWS account activity, view usage reports, and manage AWS Security Credentials. When you sign up for AWS, your AWS account is automatically signed up for all services in AWS, including AWS IoT. You are charged only for the services that you use.

For more information about AWS IOT, see the help pages of AWS here.

To set up a new account, perform these steps:

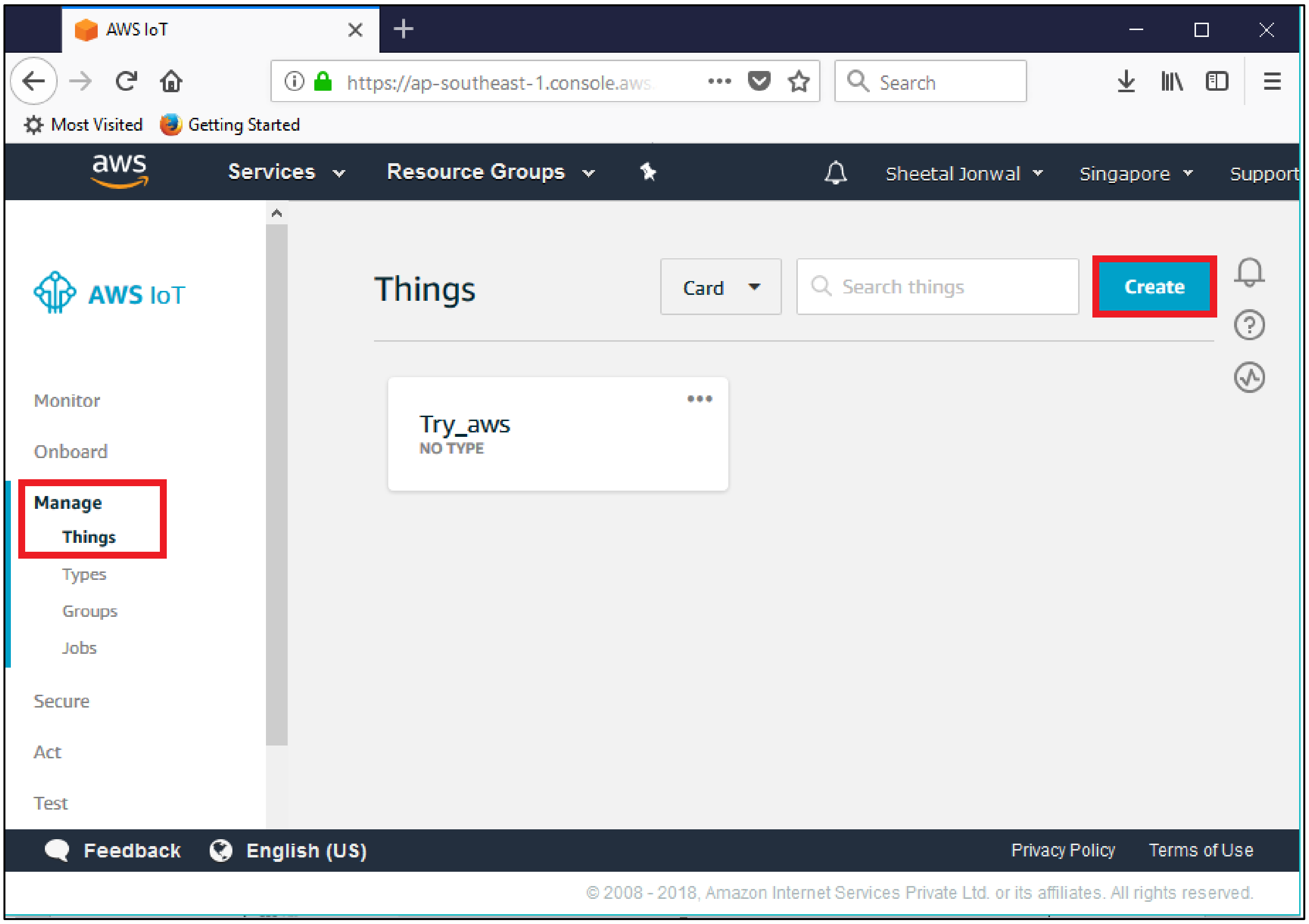
1. Open https://aws.amazon.com and choose **Create an AWS Account**.
2. Follow the online instructions. Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.
3. In the Console Home page, select your AWS Region (in this example Asia pacific (Singapore) is used), and choose the **AWS IoT** service. The AWS IoT Console window appears.

Create a Thing

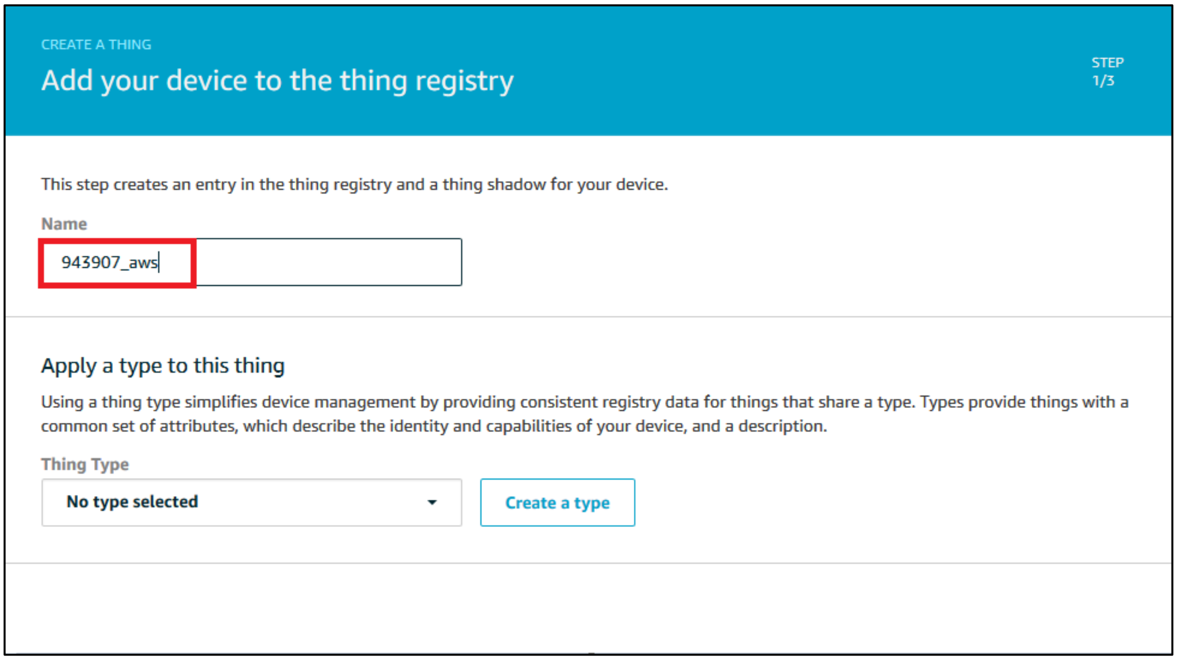
To create a `Thing`, perform these steps:

1. In the AWS IoT Console window, choose **Manage** > **Things** on the left-hand panel, and then click the **Create** button as shown in Figure 5-9.

Figure 5-9. Create Thing



2. Select **Create a single thing.** Each Thing is uniquely identified by its name. Assign a name in the **Name** field (for example, 943907_aws), and click the **Next** button.
   Note: It is possible to exchange messages without a need to create a Thing (by having a certificate with an attached policy), but AWS recommends to create the same.



CREATE A THING

Add your device to the thing registry

STEP 1/3

This step creates an entry in the thing registry and a thing shadow for your device.

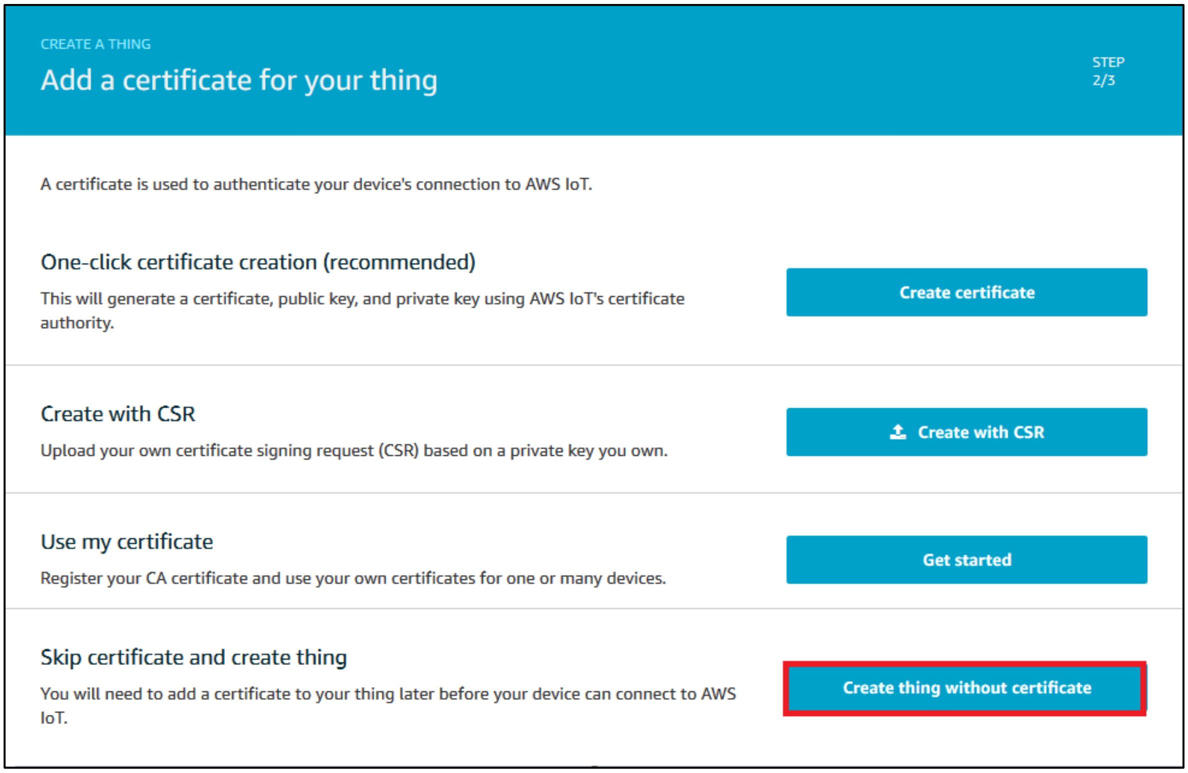
Name: 943907_aws

Apply a type to this thing

Using a thing type simplifies device management by providing consistent registry data for things that share a type. Types provide things with a common set of attributes, which describe the identity and capabilities of your device, and a description.

Thing Type: No type selected

3. In **Add a certificate for your thing** window, select **Create thing without certificate**.



CREATE A THING

Add a certificate for your thing

STEP 2/3

A certificate is used to authenticate your device's connection to AWS IoT.

One-click certificate creation (recommended)

This will generate a certificate, public key, and private key using AWS IoT's certificate authority.

Create with CSR

Upload your own certificate signing request (CSR) based on a private key you own.

Use my certificate

Register your CA certificate and use your own certificates for one or many devices.

Skip certificate and create thing

You will need to add a certificate to your thing later before your device can connect to AWS IoT.

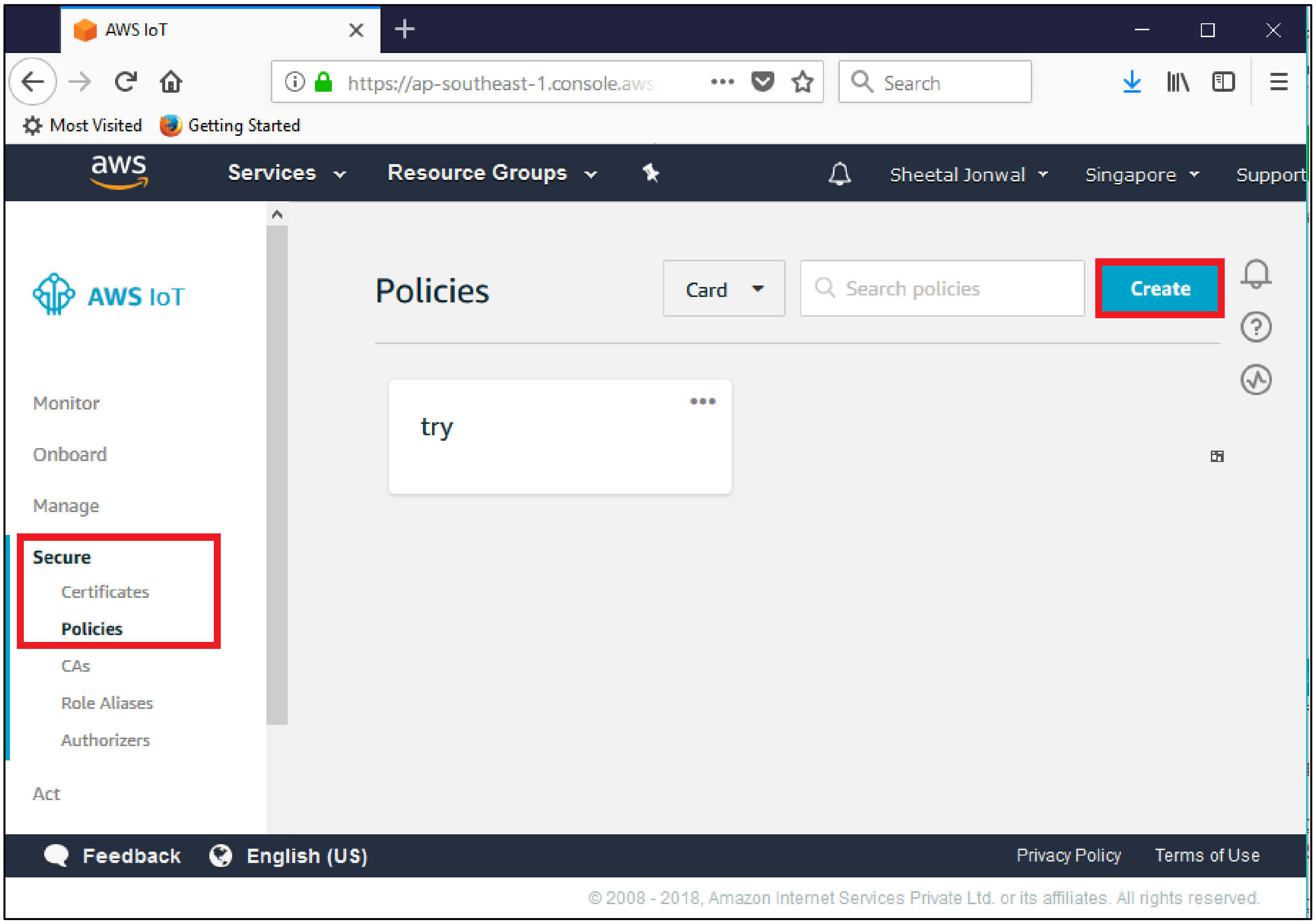
4. You will see your Thing successfully created on **Things** window.

Create a policy

To create a `policy`, perform these steps:

1. In the AWS IoT Console window, go to **Secure** > **Policies**, and then click the **Create** button as shown in Figure 5-10.

Figure 5-10. Create a Policy



2. The **Create a policy** window appears. Assign a policy name in the **Name** field (for example, 943907_policy).
3. In **Add statement**, specify the Action as **iot:***.

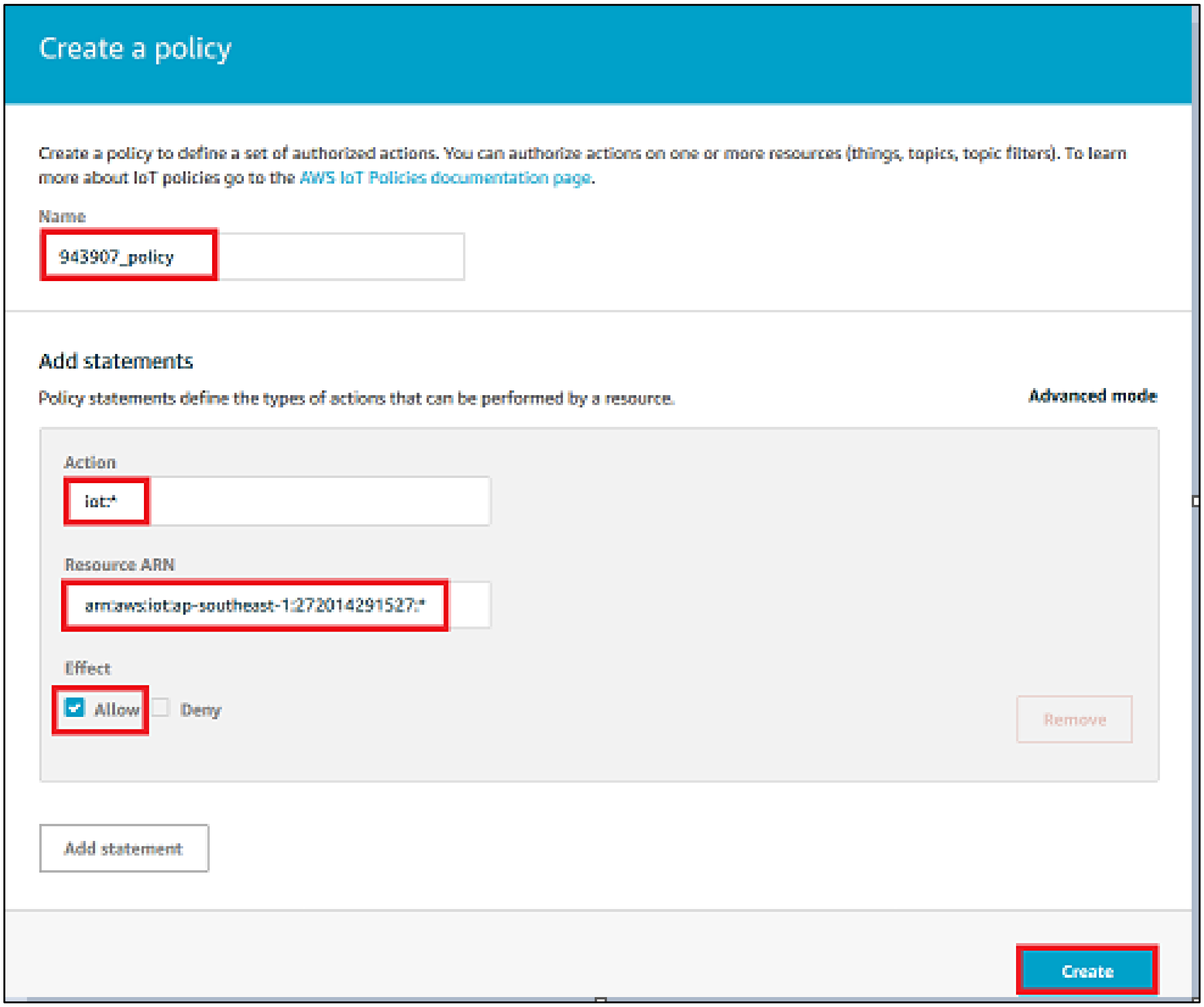
Assign an Amazon Resource Name (ARN) in the **Resource ARN** field. To use a wildcard, change the last part of the Resource ARN from "`arn:aws:iot:us-east-1:xxxxxxxxxxxx:topic/replaceWithATopic`" to "`arn:aws:iot:us-east-1:xxxxxxxxxxxx:*`".

Notes:

- Use the region that you selected when you set up your account.
- Replace xxxxxxxxxxxx with the appropriate value for your ARN.
- In the ARN name, ensure to change "`topic/replaceWithATopic`" to "*", where "*" indicates all topics. If you want to use the certificates only for a specific topic (in our case, "`943907_led_onoff`" is the one defined as `WICED_TOPIC` macro in *publish_subscribe.c*), use the following Resource ARN "`arn:aws:iot:us-east-1:xxxxxxxxxxxx:943907_led_onoff`".

4. Select the check box **Allow Effect** as shown in Figure 5-11.

Figure 5-111. Create a Policy using given details



Create a policy

Create a policy to define a set of authorized actions. You can authorize actions on one or more resources (things, topics, topic filters). To learn more about IoT policies go to the AWS IoT Policies documentation page.

Name: 943907_policy

Add statements

Policy statements define the types of actions that can be performed by a resource.

Action: iot:*

Resource ARN: arn:aws:iot:ap-southeast-1:272014291527:*

Effect: ☑ Allow ☐ Deny

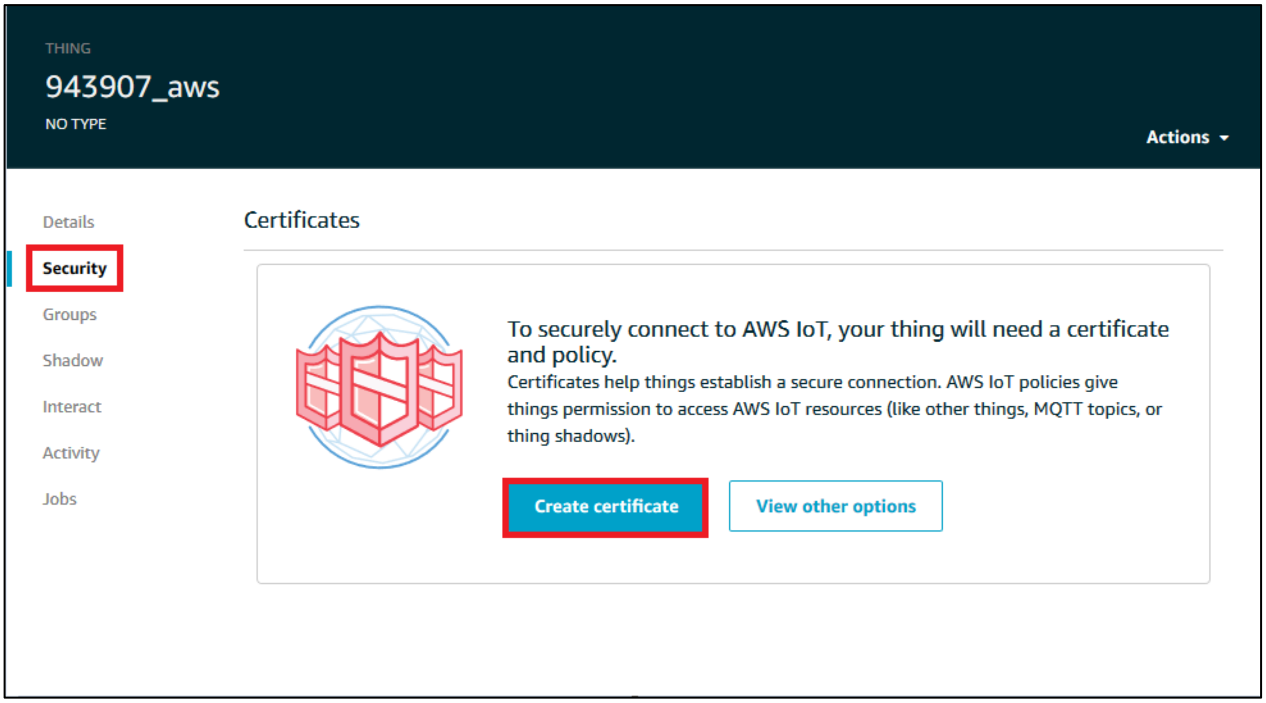
5. Click the **Create** button. You will see your policy created in **Policies** window.

Create a Certificate for a Thing

To create a `certificate`, perform these steps:

1. In the AWS IoT Console window, go to **Manage** > **Things** and then click on the created thing (for example, 943907_aws). The created Thing window appears.
2. In the left navigation pane, click on **Security**, then **Create certificate** as shown in Figure 5-12.

Figure 5-12 Create Certificates

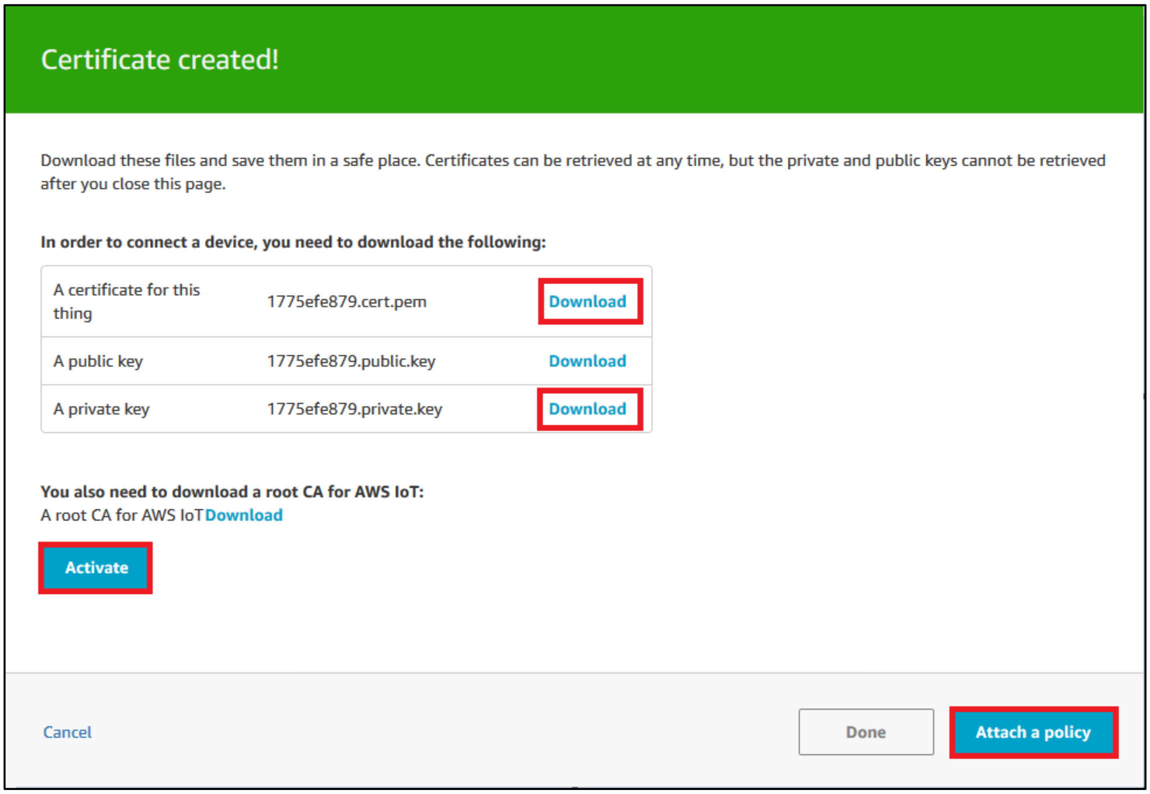


To securely connect to AWS IoT, your thing will need a certificate and policy.

Certificates help things establish a secure connection. AWS IoT policies give things permission to access AWS IoT resources (like other things, MQTT topics, or thing shadows).

3. In the **Certificate created!** Window, download **A certificate for this thing** and **A private key** by clicking on corresponding **Download** tabs. Click on **Activate** tab and **Attach a policy** as shown in Figure 5-13.

Figure 5-13. Download and activate the Certificates



Download these files and save them in a safe place. Certificates can be retrieved at any time, but the private and public keys cannot be retrieved after you close this page.

In order to connect a device, you need to download the following:

A certificate for this thing	1775efe879.cert.pem	Download
A public key	1775efe879.public.key	Download
A private key	1775efe879.private.key	Download

You also need to download a root CA for AWS IoT:
A root CA for AWS IoTDownload

Notes:

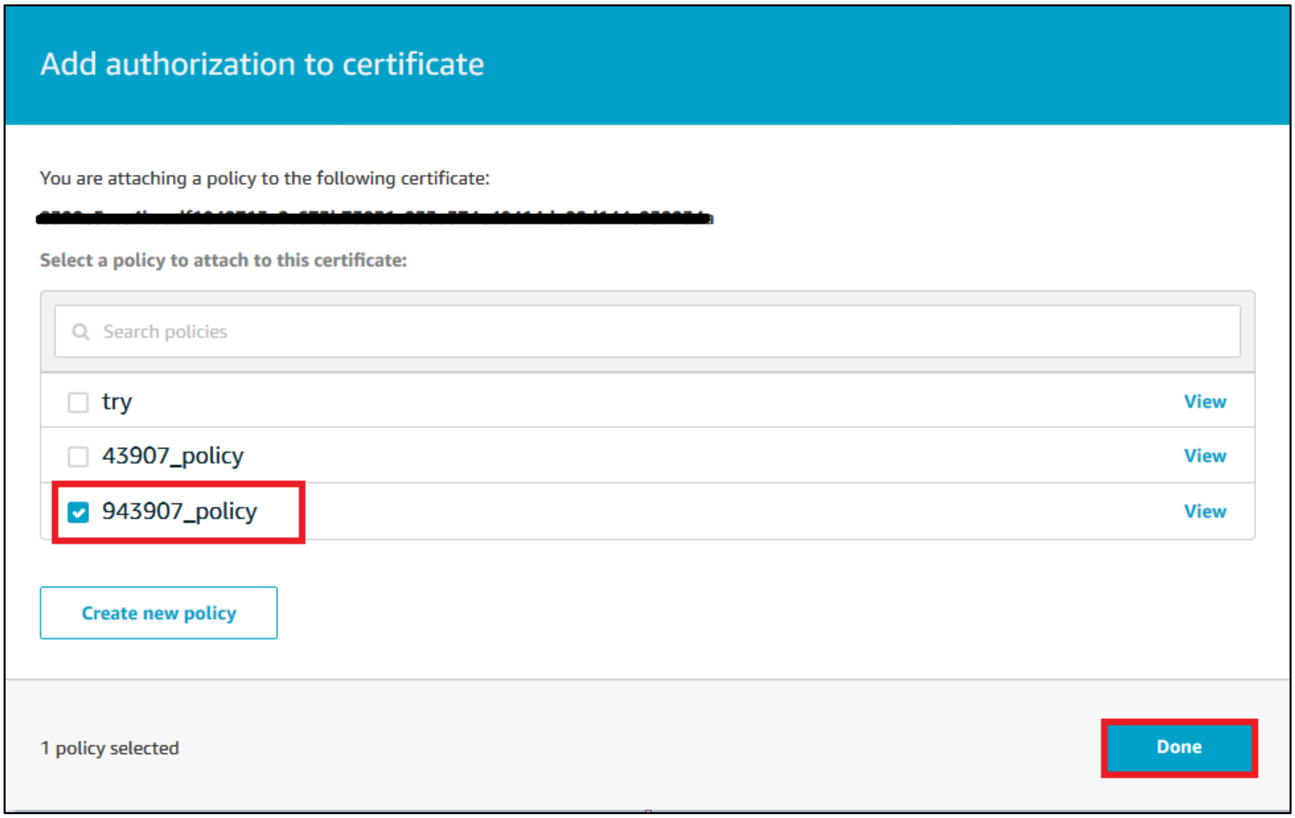
- The certificate and private key cannot be revisited later for download and must be saved while creating the certificate.
- Copy the content of the downloaded Thing certificate and paste in the client.cer at location `<WICED-SDK>\43xxx_Wi-Fi\resources\apps\aws_iot\client.cer`.

  (In WICED Studio 6.2 and later, paste the content at location `<WICED-SDK>\43xxx_Wi-Fi\resources\apps\aws\iot\publisher\client.cer` and `<WICED-SDK>\43xxx_Wi-Fi\resources\apps\aws\iot\subscriber\client.cer`.)
- Copy the content of downloaded Private key and paste in `privkey.cer` at location `<WICED-SDK>\43xxx_Wi-Fi\resources\apps\aws_iot\privkey.cer`.

  (In WICED Studio 6.2 and later, paste the content at location `<WICED-SDK>\43xxx_Wi-Fi\resources\apps\aws\iot\publisher\privkey.cer` and `<WICED-SDK>\43xxx_Wi-Fi\resources\apps\aws\iot\subscriber\privkey.cer`.)

5. Select the check box next to the policy you want to choose and then click **Done**, as shown in Figure 5-14.

Figure 5-14. Register a Thing



Add authorization to certificate

You are attaching a policy to the following certificate:

[redacted]

Select a policy to attach to this certificate:

Search policies

□ try View

□ 43907_policy View

☑ 943907_policy View

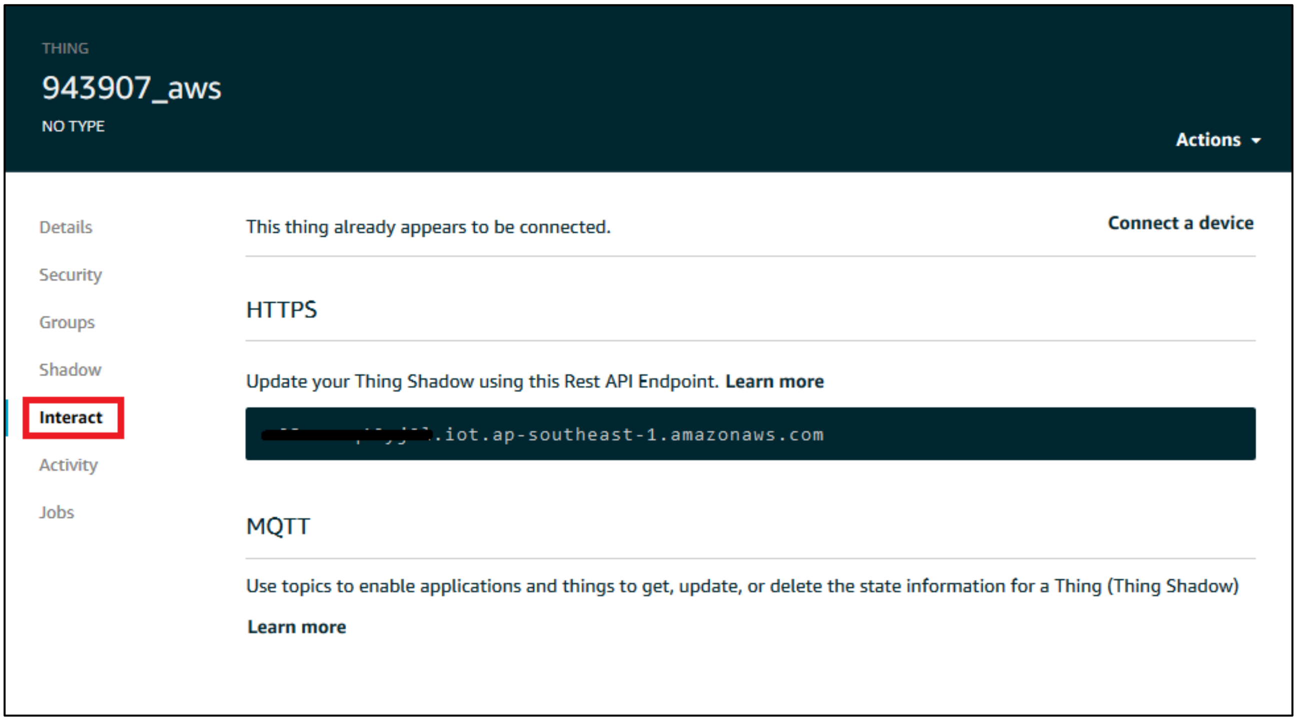
Create new policy

1 policy selected

Done

6. Now, click on the thing you have created. Go to **Security**. The Certificates window appears.
7. Click on the created certificate. The Certificate details appears.
8. Click **Policies** in the left-hand panel to validate if the correct policy is linked.
9. Click **Things** in the left-hand panel to validate if the correct Thing is linked.
10. Click the specific Thing. The Thing ARN window appears.
11. In the left navigation pane, choose **Interact**.
12. Copy the Endpoint from the HTTPS tab as shown in Figure 5-15.

Figure 5-15. Endpoint



THING

943907_aws

NO TYPE

Actions ▾

Details

Security

Groups

Shadow

Interact

Activity

Jobs

This thing already appears to be connected.

Connect a device

HTTPS

Update your Thing Shadow using this Rest API Endpoint. **Learn more**

```
[redacted].iot.ap-southeast-1.amazonaws.com
```

MQTT

Use topics to enable applications and things to get, update, or delete the state information for a Thing (Thing Shadow)

Learn more

13. Navigate to the *publish_subscribe.c* file to update the `MQTT_BROKER_ADDRESS` macro with the endpoint address copied from **HTTPS** tab. Remove the first string before "." in endpoint and replace it with * and copy it to the `REGION` macro. In this case, it is "`*.iot.ap-southeast-1.amazonaws.com`".

14. The created Thing, policy, and certificate are used to interact with the AWS IoT.

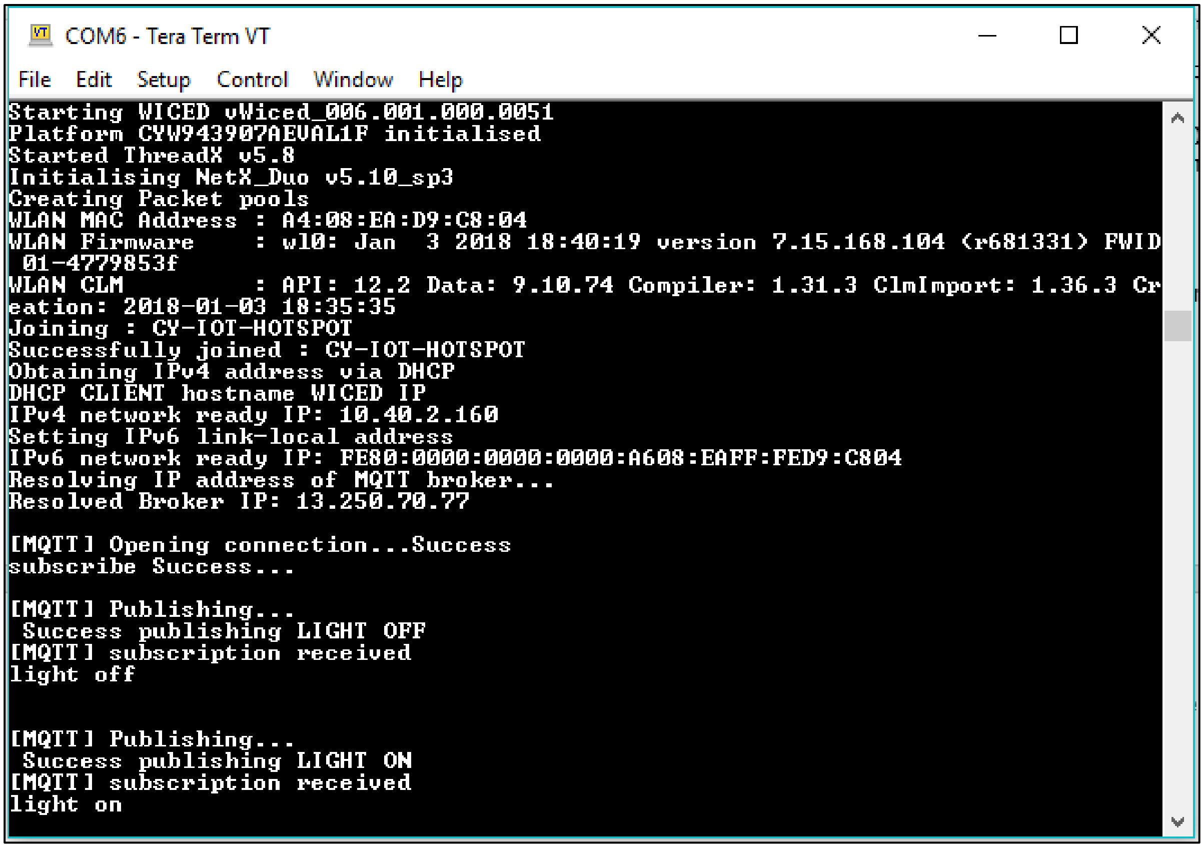
5.5.4.2 *Access Point Credentials*

Enter your credentials (SSID and pass phrase key) in the *wifi_config_dct.h* file. The `CLIENT_AP_SSID` macro should be updated with your access point's SSID, the `CLIENT_AP_PASSPHRASE` macro should be updated with your access point's pass phrase key and the `CLIENT_AP_SECURITY` macro should be updated with the security type of your access point. This is "`WICED_SECURITY_WPA2_MIXED_PSK`" if your access point uses WPA2-PSK. If your AP uses a different security, then choose the correct one defined in `enum wiced_security_t` from *43xxx_Wi-Fi\WICED\WWD\include\wwd_constants.h*.

5.5.4.3 *Build, Program, and Verify*

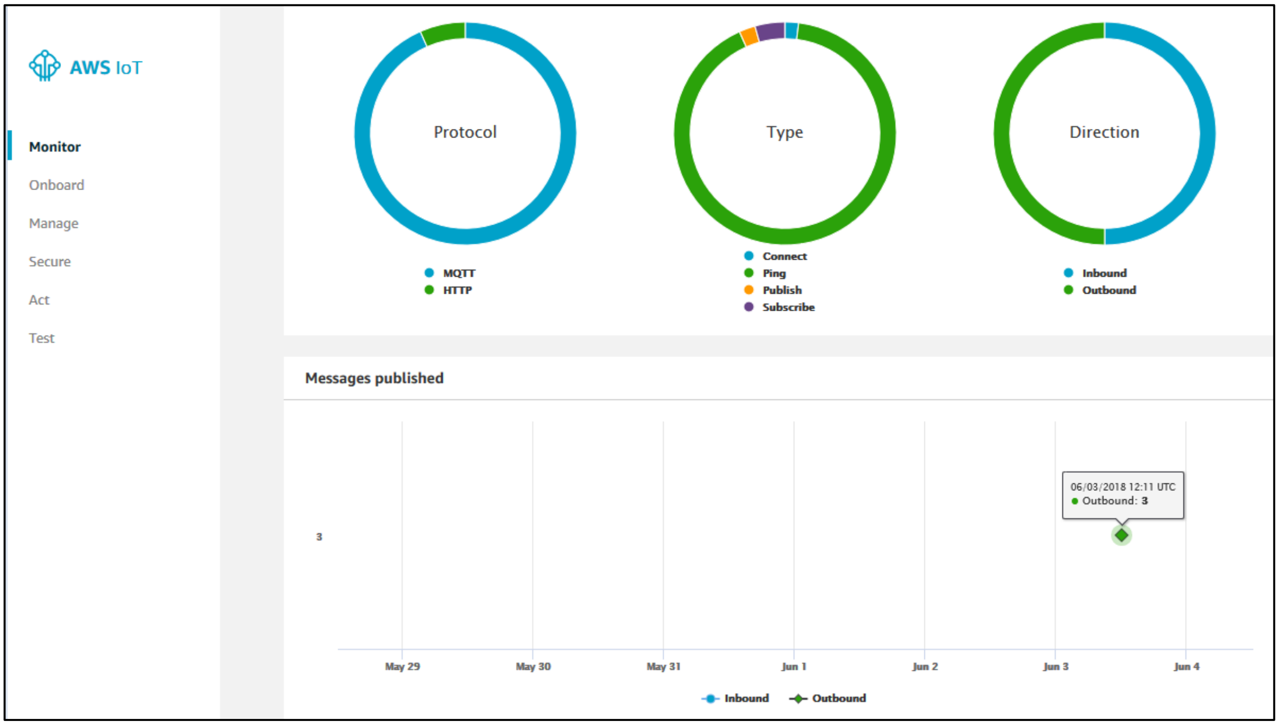
Your Wi-Fi access point must be connected to the internet to verify the example. Build and program the `publish_subscribe_aws` example using a similar procedure to the one provided in Building, Programming, and Debugging CYW943907AEVAL1F EVK. Once programmed, the CYW943907AEVAL1F EVK will try to connect to AWS IoT and subscribe to the specified topic. After that if you press switch USER_1 it will turn LED_1 on and off alternatively as shown in Figure 5-16. Note that this is being done over the cloud. That is, pushing the switch publishes a message to the cloud. The LED turns on in response to a notification from the cloud. You can also observe the messages inside the AWS console. In the AWS IoT console window, go to **Monitor > Messages published** and you will see the number of messages exchanged as shown in Figure 5-17.

Figure 5-16. `Publish_Subscribe_aws` Output



```
Starting WICED vWiced_006.001.000.0051
Platform CYW943907AEVAL1F initialised
Started ThreadX v5.8
Initialising NetX_Duo v5.10_sp3
Creating Packet pools
WLAN MAC Address : A4:08:EA:D9:C8:04
WLAN Firmware    : wl0: Jan  3 2018 18:40:19 version 7.15.168.104 (r681331) FWID
 01-4779853f
WLAN CLM         : API: 12.2 Data: 9.10.74 Compiler: 1.31.3 ClmImport: 1.36.3 Cr
eation: 2018-01-03 18:35:35
Joining : CY-IOT-HOTSPOT
Successfully joined : CY-IOT-HOTSPOT
Obtaining IPv4 address via DHCP
DHCP CLIENT hostname WICED IP
IPv4 network ready IP: 10.40.2.160
Setting IPv6 link-local address
IPv6 network ready IP: FE80:0000:0000:0000:A608:EAFF:FED9:C804
Resolving IP address of MQTT broker...
Resolved Broker IP: 13.250.70.77

[MQTT] Opening connection...Success
subscribe Success...

[MQTT] Publishing...
 Success publishing LIGHT OFF
[MQTT] subscription received
light off


[MQTT] Publishing...
 Success publishing LIGHT ON
[MQTT] subscription received
light on
```

Figure 5-17. Messages Published



Revision History

0

Revision	Date	Description of Change
**	06/22/2017	Initial version of the kit guide.
*A	06/18/2018	Updated ‘Section 5.5: Publish_subscribe_aws’ based on the new user interface of Amazon AWS
*B	02/28/2020	Updated Section Bootstrap
*C	03/13/2020	Updated 6. GPIO_1_GSPI_MODE in Section Bootstrap and Control Pins
*D	06/08/2020	Corrected links to Community Software Forums