



CYW943907AEVAL1F

Evaluation Kit User Guide

Doc. No. 002-18703 Rev. *A

Cypress Semiconductor
198 Champion Court
San Jose, CA95134-1709
www.cypress.com

Copyrights

© Cypress Semiconductor Corporation, 2018. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress’s patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

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

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

Contents



CYW943907AEVAL1F	1
Evaluation Kit User Guide	1
Contents	3
Safety Information.....	5
1. Introduction.....	6
1.1 CYW943907AEVAL1F EVK Contents	6
1.2 CYW943907AEVAL1F Board Details	8
1.3 Modus Toolbox Overview	9
1.4 Getting Started	10
1.5 IOT Resources and Technical Support.....	10
1.6 Additional Learning Resources.....	11
1.7 Document Conventions	11
1.8 Acronyms	11
2. Software Installation	12
2.1 System Requirements	12
2.2 Installation Steps	12
3. Kit Operation.....	15
3.1 Theory of Operation.....	15
3.2 Onboard Programmer/Debugger and Serial Interface Chip.....	15
3.3 CYW943907AEVAL1F Kit Connection	16
3.4 Building, Programming, and Debugging CYW943907AEVAL1F EVK	18
4. Hardware	28
4.1 User Switches.....	29
4.2 LED	29
4.3 Reset Control.....	29
4.4 Ethernet.....	30
4.5 Micro SD Connector/Slot	32
4.6 JTAG Connector.....	33
4.7 Connectors	33
4.8 UART Port Configuration on CYW943907AEVAL1F Kit.....	35
4.9 External ADC.....	36
4.10 PWM.....	37
5. Code Examples.....	39

5.1	AWSIOTPubSub Project	39
5.2	WiFiScanner	49
Revision History		50

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. For 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.

	<p>The CYW943907AEVAL1F EVK 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.</p>
--	---

	<p>End-of-Life/Product Recycling</p> <p>This kit has an end-of-life cycle of five years from the year of manufacturing on the back of the box. Contact your nearest recycler for discarding the kit.</p>
--	---

General Safety Instructions

ESD Protection

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

Handling Boards

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 can be used either with WICED® Studio™ IDE (version 5.0 or later), or ModusToolbox™ IDE (version 1.0 or later) to develop and debug your CYW43907 project. This document covers only the development flow based on the ModusToolbox IDE. Refer to the equivalent kit user guide at <http://www.cypress.com/documentation/development-kitsboards/cyw943907aeval1f-evaluation-kit> for the WICED Studio-based development flow. 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 this kit were named BCM943907AEVAL1F_2 and BCM943907AEVAL1F. CYW43907 and BCM43907 refer to the same device.

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



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. RJ-45 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. Onboard /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. μ SD Connector/slot (J7)

Figure 1-2. CYW943907AEVAL1F Evaluation Board

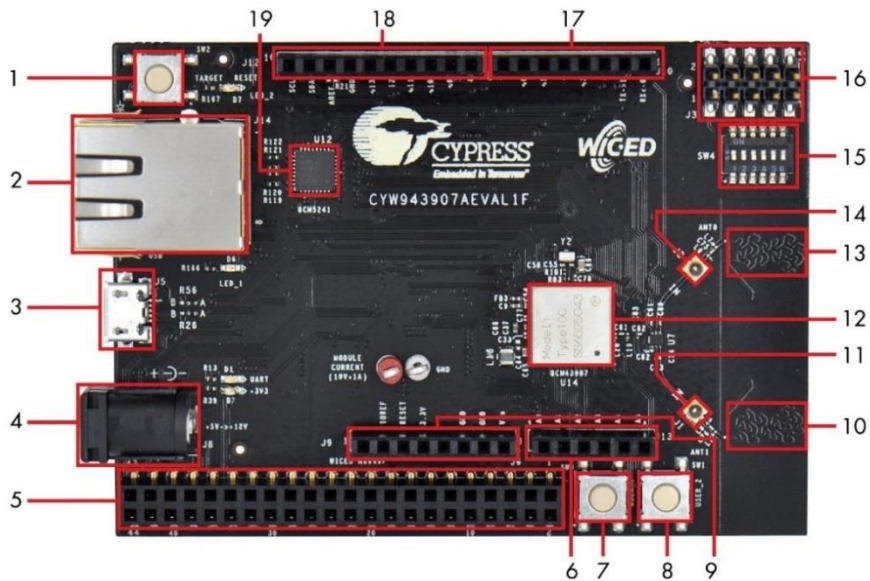


Figure 1-3. CYW943907AEVAL1F Evaluation Board (Back View)

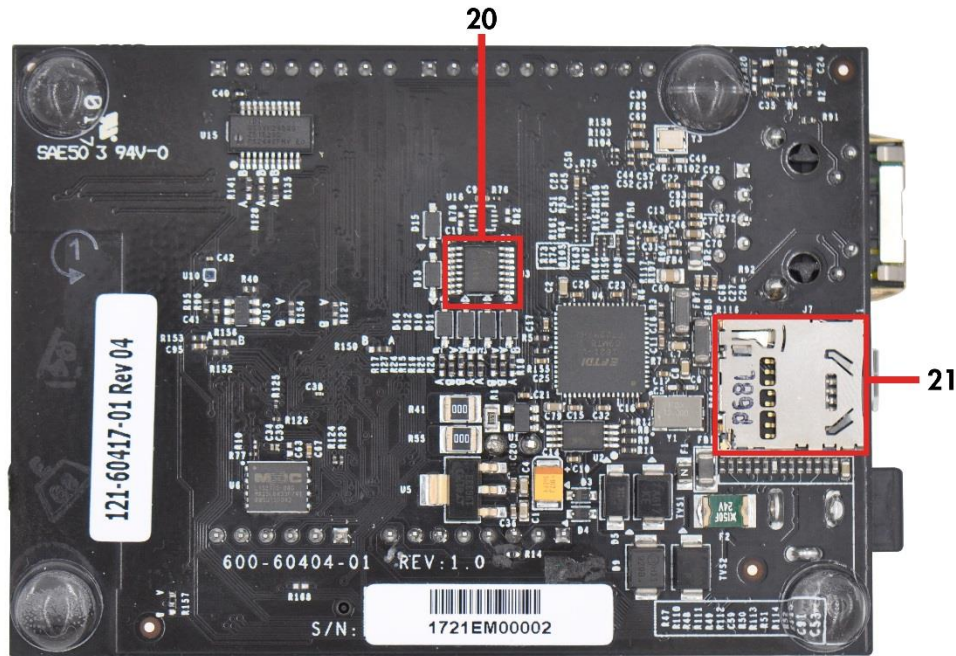


Figure 1-4. WICED IDE

1.3 Modus Toolbox Overview

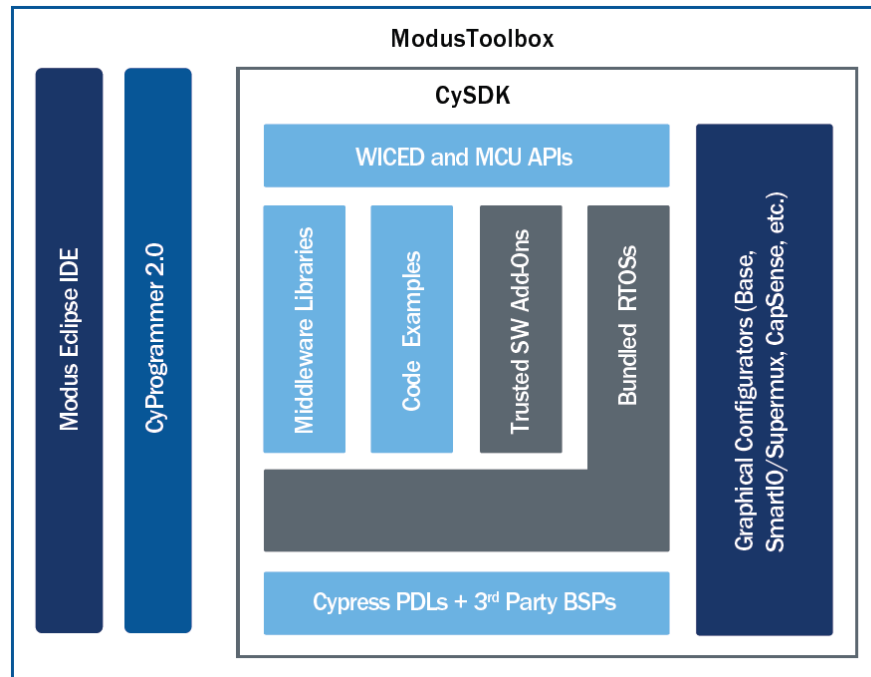
ModusToolbox is a set of tools that enable you to integrate Cypress devices into your existing development methodology. One of the tools is a multi-platform, Eclipse-based Integrated Development Environment (IDE) that supports configuration and application development, called Modus IDE.

Modus IDE is an installable IDE built on top of the Eclipse framework. It allows you to create and modify designs. The user interface integrates into Eclipse. As little of the code as possible interfaces with Eclipse to remain as modular as possible. The IDE provides hooks for launching various tools provided by the Cypress Software Development Kit (SDK).

The Cypress SDK provides the central core of the ModusToolbox IDE. It contains configuration tools, drivers, libraries, middleware, as well as various utilities, make files, and scripts. You may use one or a few of these tools in any environment you prefer.

The following shows a high-level view of the tools included in ModusToolbox.

Figure 1-5. ModusToolbox Architecture



See the following documents for more information:

- ModusToolbox Installation Guide
- Modus IDE Quick Start Guide
- Modus IDE Help
- Configurator Documentation (open from a particular Configurator)
- WICED Power Logger User Guide
- Eclipse Documentation

1.4 Getting Started

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

This 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 Modus Toolbox.
- The [Kit Operation](#) chapter describes the major sections of the kit such as the onboard 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.

1.5 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.

1.6 Additional Learning Resources

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

1.7 Document Conventions

Table 1-1. Document Conventions for Guides

Convention	Usage
<i>Italics</i>	Displays file locations, user entered text, and source code: <code>C:\...cd\iccl</code>
<i>Italics</i>	Displays file names and reference documentation.
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.

1.8 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 provides instructions for installing the ModusToolbox software – a set of tools that enables you to integrate Cypress devices into your existing development methodology. This is a Beta release of ModusToolbox, and it is only available through the [ModusToolbox Early Access Program \(EAP\) website](#).

2.1 System Requirements

Cypress recommends the following minimum system configuration:

- CPU with a PassMark score > 2000 (cpubenchmark.net)
- 4 GB RAM
- 10% of total disk space with at least 10 GB free

ModusToolbox is **not** supported on 32-bit operating systems. It has been tested on the following:

- Windows 7 64-bit / Windows 10 64-bit
- macOS 10.13
- Ubuntu Linux 16.04 LTS

Notes:

- For Windows, you must have Microsoft Visual C++ 2017 Redistributable (x64) - 14.14.26429 installed. This is installed by default with the ModusToolbox Windows installer.
- For Linux and macOS, ModusToolbox requires the following Unix packages to work properly:
 - make (v3.81)
 - mktemp (v8.25)
 - perl (v5.18.2)
 - cmp (v2.8.1)

Some versions of Ubuntu Linux do not include 'make' by default. Use the command `sudo apt-get install make` to install it.

2.2 Installation Steps

2.2.1 Step 1: Uninstall Previous Release

Uninstall any previous version already installed before installing this release.

- **Windows:** The current release installer will prompt you to uninstall a previous release. You can also use the Windows Control Panel.
- **Linux:** Go to the directory where you extracted the *tar.gz* installer. Delete the *ModusToolbox_<version>* directory and its contents.
- **macOS:** Go to the Applications directory to which you moved the *ModusToolbox_<version>* directory. Delete the *ModusToolbox_<version>* directory and its contents.

2.2.2 Step 2: Download the Software

Go to the ModusToolbox EAP [website](#) and download the appropriate software for your platform:

- **Windows:** *ModusToolbox setup 1.0.0.<build>.x86_64.exe*
- **Linux:** *ModusToolbox 1.0.0.<build>.x86_64.tar.gz*
- **macOS:** *ModusToolbox 1.0.0.<build>.dmg*

2.2.3 Step 3: Install ModusToolbox

Note: Do not use spaces in the installation directory name. Various tools such as make do not support spaces. Also, do not use common illegal characters, such as:

* . " / \ [] : ; | = ,

- **Windows:** Run the *ModusToolbox setup 1.0.0.<build>.x86_64.exe* installer program. By default, it is installed here:
C:\Users\<user_name>\ModusToolbox_1.0
Note: If you have not installed ModusToolbox previously, you may be prompted to restart your computer because the installer installs the Microsoft Visual C++ 2017 redistributable.
- **Linux:** Extract the *ModusToolbox 1.0.0.<build>.x86_64.tar.gz* file to your desired location. The extraction process will create a *ModusToolbox_<version>* directory.
- **macOS:**
 1. Double-click the downloaded *ModusToolbox 1.0.0.<build>.dmg* file. This file will be mounted as a disk image on your machine. A window for the mounted image appears.
 2. On the mounted image window, drag and drop the *ModusToolbox_<version>* folder on the left into the *Applications* folder on the right.
 3. Eject the mounted ModusToolbox image in the Finder by right-clicking the *ModusToolbox_<version>* icon and clicking **Eject**.
Note: You may be prompted to install macOS command line developer tools that give you the option to get the XCode/native make application.

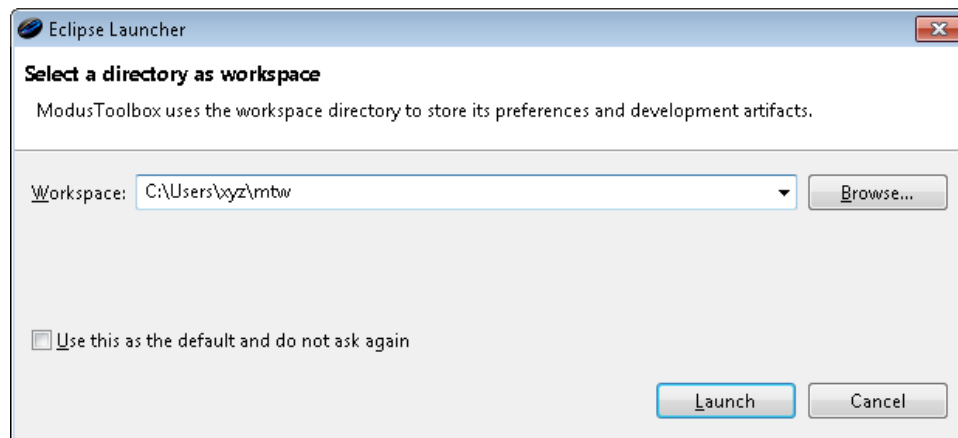
2.2.4 Step 4: Run the Modus IDE

Note: For Linux and macOS to properly initialize modus-shell-1.0 tools, run the Modus IDE at least once after installation. Otherwise, projects will not build. After the IDE loads, modus-shell-1.0 tools will be ready for use.

- **Windows:** *Navigate to <installed_location>/ModusToolbox/eclipse and run ModusToolbox.exe.*
Note: You may be asked to grant access by the firewall manager.
- **Linux:** *Navigate to <extracted_location>/ModusToolbox/eclipse and run ModusToolbox.*
- **macOS:** *Navigate to the directory where you moved the ModusToolbox.app file (e.g., /Applications/ModusToolbox) and run ModusToolbox.app.*

When the Modus IDE runs for the first time, a dialog opens to specify the Workspace location. The default location for the workspace is: *[user_home]\mtw*.

Figure 2-1. Selecting Workspace for Modus IDE



Note: Select the **Use this as the default ...** check box to hide this dialog on future startups.

Note: Do not use spaces or illegal characters anywhere in the path of the workspace name or location.

2.2.5 Next Steps

Refer to the *Quick Start Guide* for brief instructions to create, build, and program projects. Refer also to the *Modus IDE User Guide* for more detailed information. These documents are available from the ModusToolbox EAP website.

3. Kit Operation



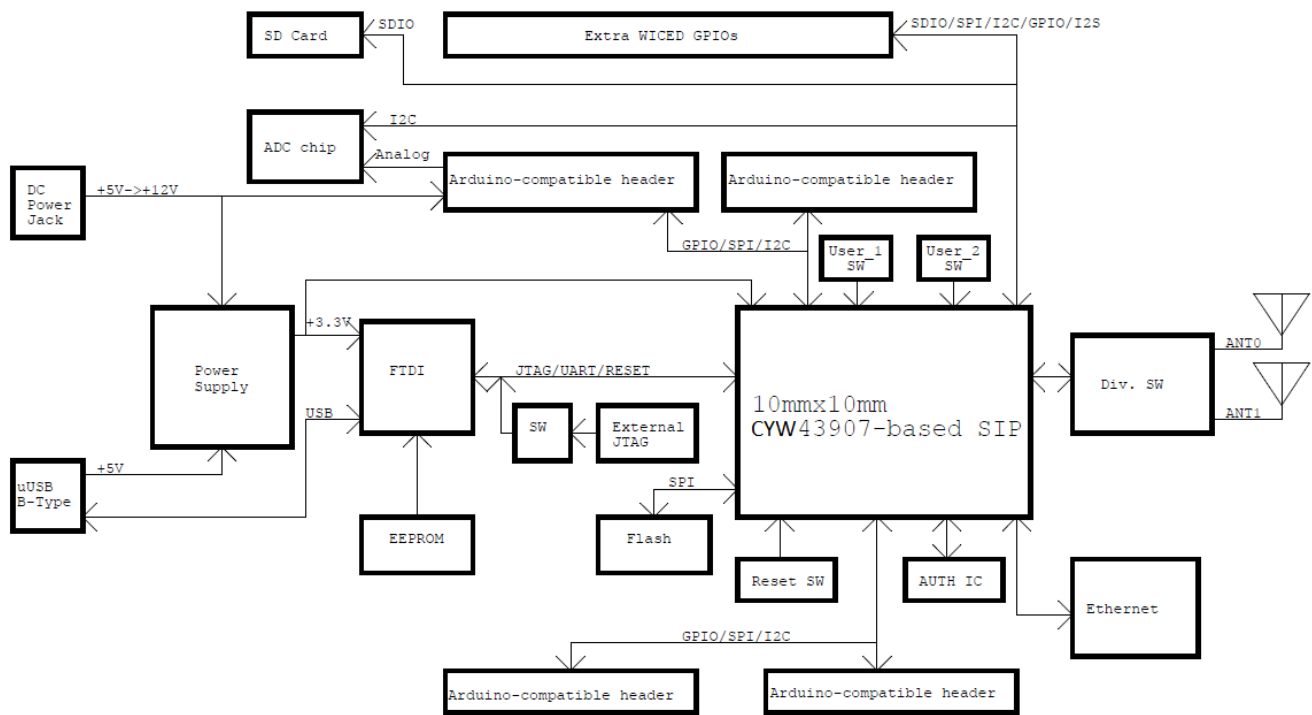
This chapter introduces you to the CYW943907AEVAL1F EVK and the features that will be used as part of the kit operation. Features such as Wi-Fi connection and programming/debugging are discussed in this chapter. The chapter also describes the USB-UART bridge that can be used to communicate with the CYW43907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

3.1 Theory of Operation

Figure 3-1 shows the block diagram of the CYW943907AEVAL1F EVK. This board contains a BCM943907/CYW43907-based System in package (SiP), which is a Murata 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 that 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 Onboard Programmer/Debugger and Serial Interface Chip

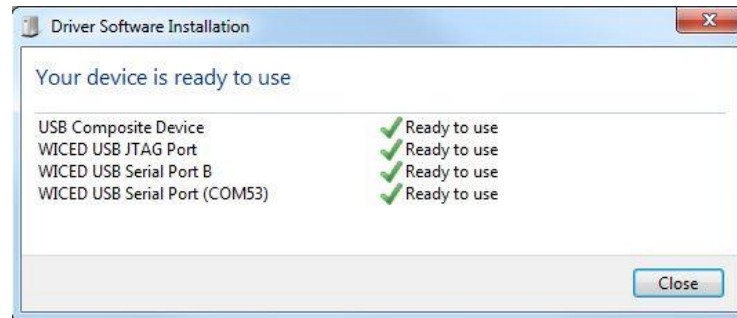
An FT-2232-HQ chip is used for onboard programming, debugging, and USB-Serial functionality. It connects to the computer over a USB interface and connects to the CYW43907-based SiP module over JTAG and UART pins.

3.3 CYW943907AEVAL1F Kit Connection

The CYW943907AEVAL1F EVK can be powered by either an external power supply or through USB.

When using an external power supply, use a 5 V–12 V, 2 A power supply with a 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. \

Figure 3-2. Driver Software Installation



3.3.1 Verifying Driver Installation

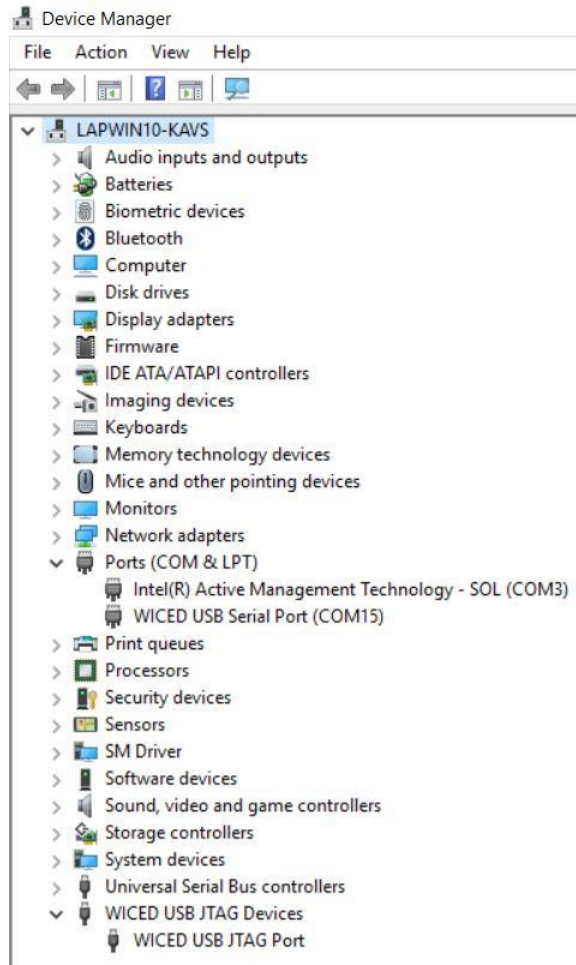
Drivers for the CYW943907AEVAL1F EVK are automatically installed during the ModusToolbox IDE installation process. When you connect the kit for the first time to your PC, it will search for the appropriate driver as shown in [Figure 3-2](#).

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**. Device Manager lists the two logical USB devices as follows (see [Figure 3-3](#)):
 - WICED USB Serial Port under **Ports (COM & LPT)**
 - WICED USB JTAG Port under **WICED USB JTAG Devices**

As [Figure 3-3](#) shows, Device Manager 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, follow the link in this [Cypress Developer Community post](#).

Figure 3-3. Verifying Device Driver Installation



3.3.2 Troubleshooting

If an error occurs 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 Device Manager, verify that the +3V3 LED (D2) is turned ON, and check the USB cable.

3.3.3 External Power Supply

The CYW943907AEVAL1F EVK can be powered using an external power supply (5 V–12 V, 2A), using a 2.5-mm DC Jack with the center pin positive. When using an external power supply and also connecting a USB cable (for programming/debugging or USB-UART), the voltage on the external power supply must 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.

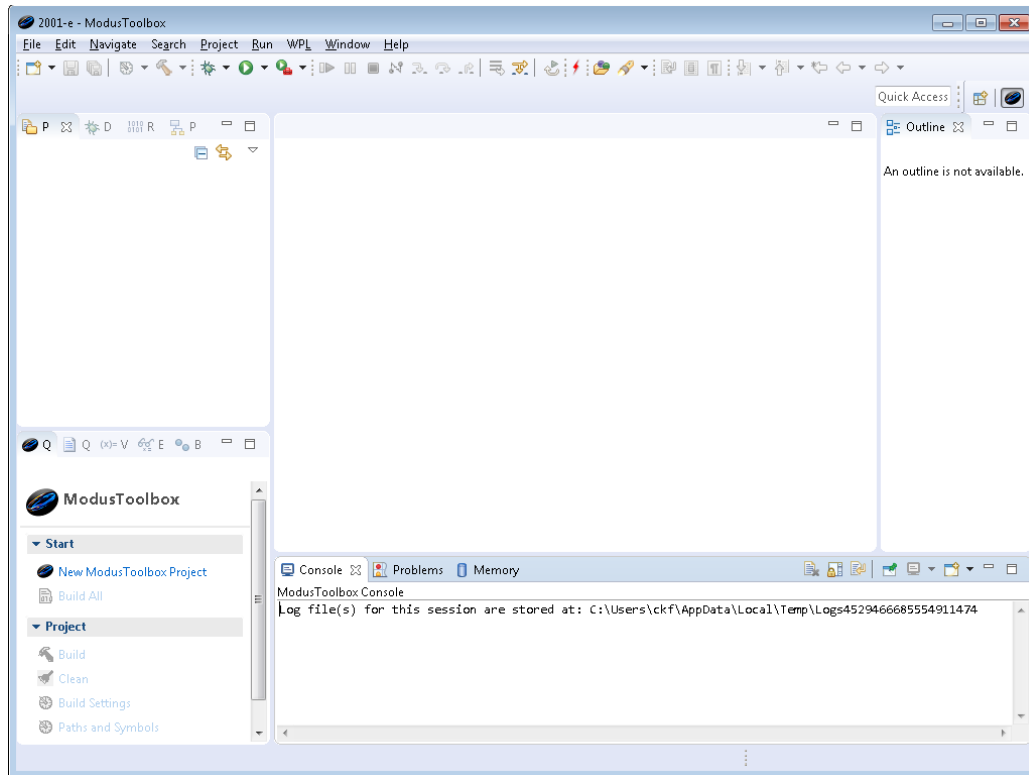
3.4 Building, Programming, and Debugging CYW943907AEVAL1F EVK

3.4.1 Building and Programming a Project for CYW943907AEVAL1F in ModusToolbox

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

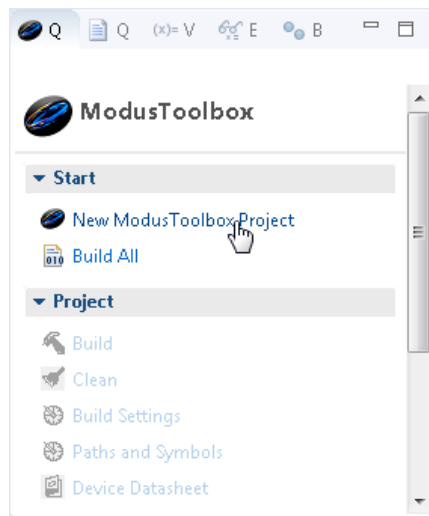
1. Launch Modus IDE. Run the “ModusToolbox” executable file to launch the IDE as applicable for your operating system. The Modus IDE is installed in the <user_home>ModusToolbox_1.0\ eclipse directory by default.

Figure 3-4. Modus IDE



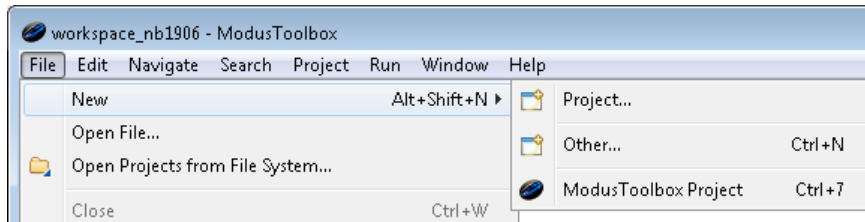
2. Click **New ModusToolbox Project** in the ModusToolbox IDE’s Quick Panel.

Figure 3-5. Starting a New Project



Alternatively, select **File > New > ModusToolbox Project** or press **Ctrl+7**.

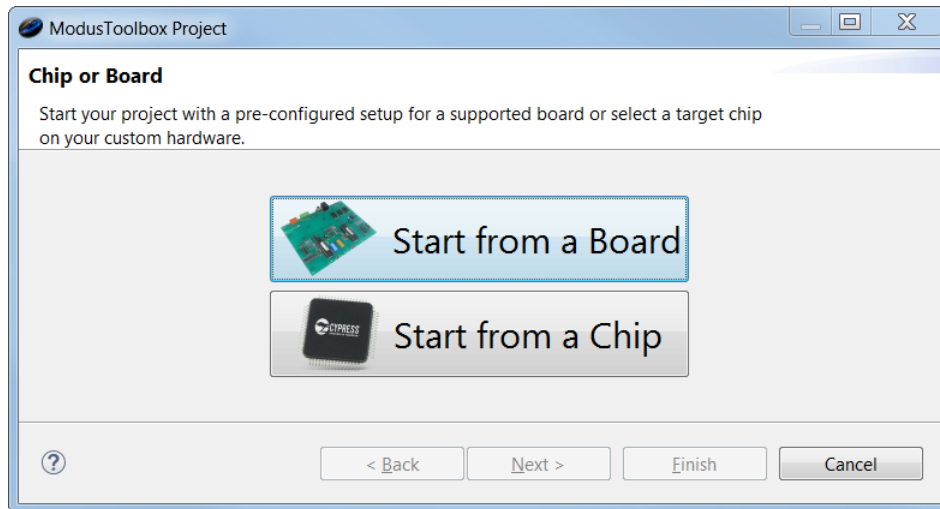
Figure 3-6. Create New Project



3. Select CYW943907AEVAL1F Board and Starter Project.

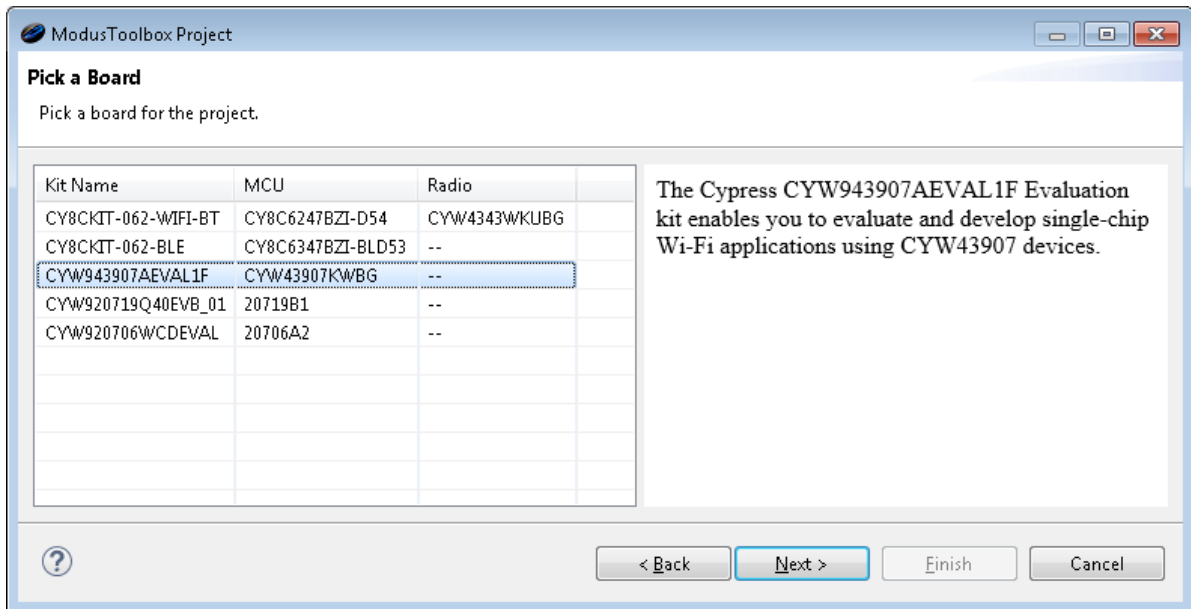
On the ModusToolbox Project dialog, chose the option to start your project from a board (see Figure 3-7). You can start a development project based on the CYW43907 device only through the “Start from a Board” option and not through the “Start from a Chip” option.

Figure 3-7. Select Start from a Board



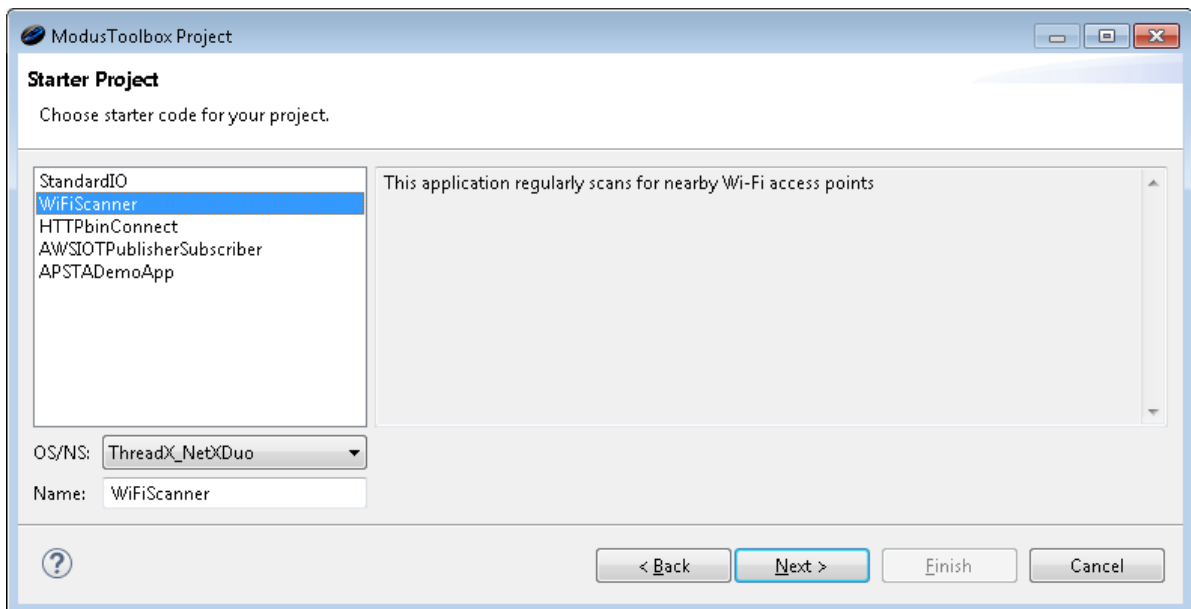
The dialog displays a list of boards, showing the Kit Name, MCU, and Radio (if applicable), and shows a description of the board that you select. For this example, select the CYW943907AEVAL1F board.

Figure 3-8. Selecting the Board



4. Click **Next** to open the Starter Project page. This page lists various starter projects available for the selected kit. See Figure 3-9.

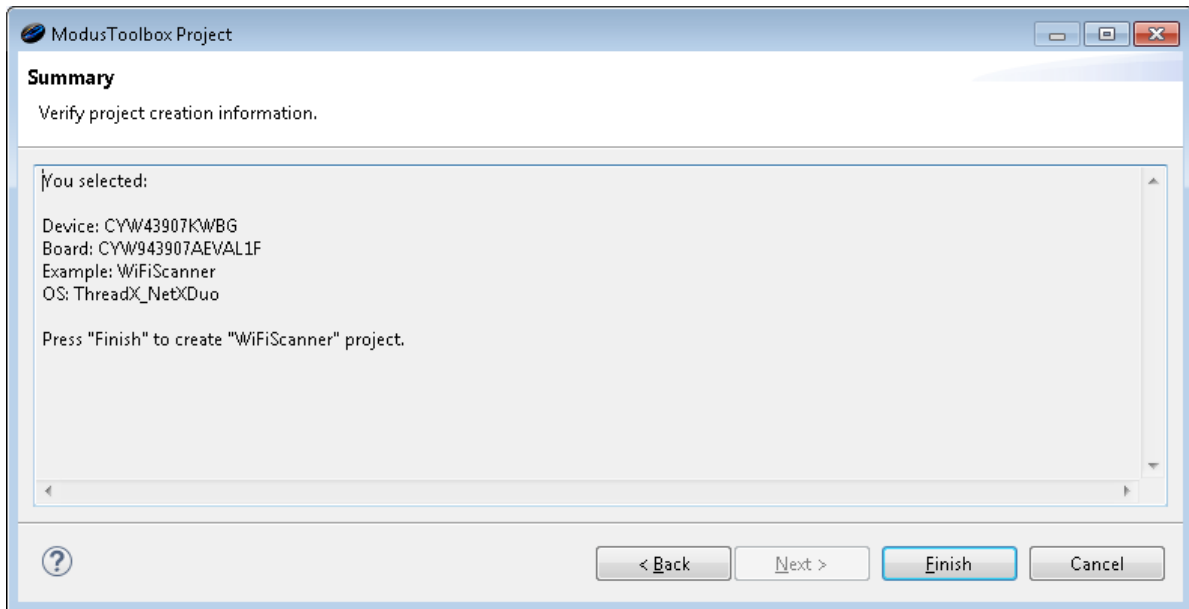
Figure 3-9. Select the Starter Project



For this example, do the following:

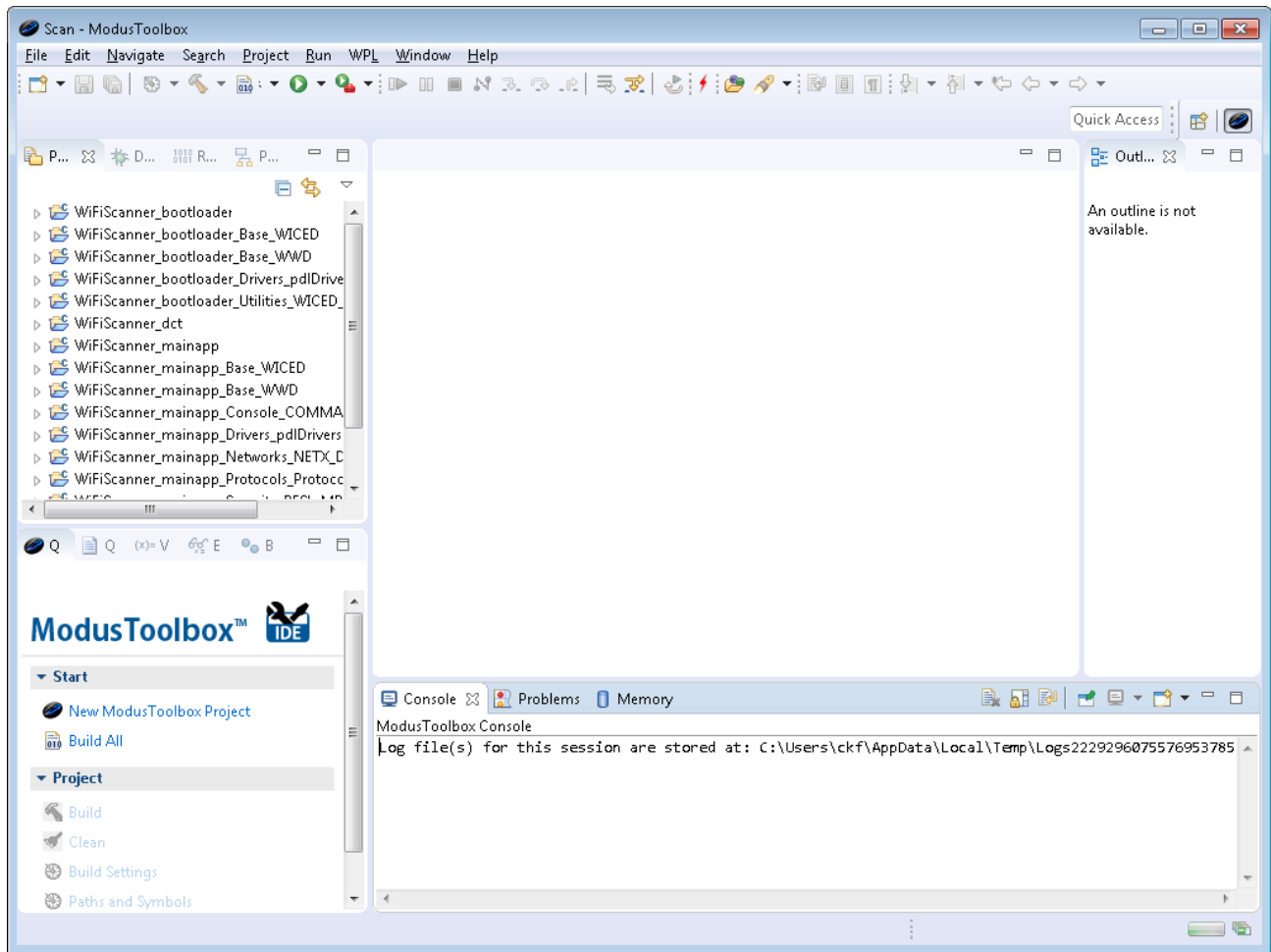
- Select the "WiFiScanner" project from the list.
 - Select the appropriate OS/NS. In this case, "ThreadX_NetXDuo" is the only option available. Other starter projects also have "ThreadX_NetX" as an option.
 - Type a name for your project. Do not use spaces, underscores or any other special characters in the project name. In this case, "WiFiScanner" is the default name.
5. Click **Next** to open the Summary page. This page shows the options chosen for this project. Review them to ensure that they are correct.

Figure 3-10. Summary of Project Options



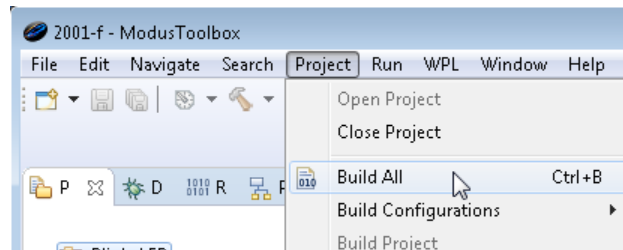
6. Click **Finish** to create the project. After a few moments, the Modus IDE displays the progress information. When complete, project folders appear in the Project Explorer pane as Figure 3-11 shows.

Figure 3-11. Created Project



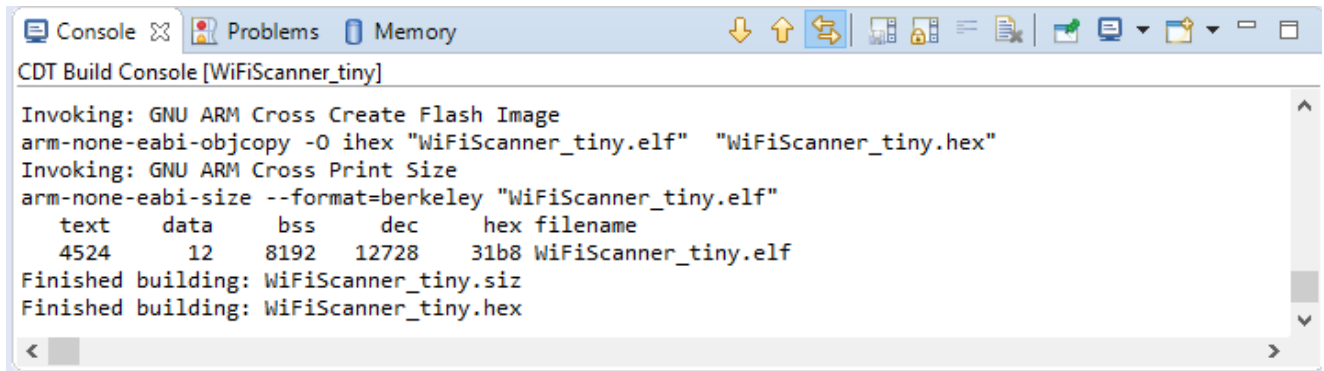
7. Build Starter Project. After loading a project, select **Project > Build All** (or press **Ctrl+B**).

Figure 3-12. Build Project



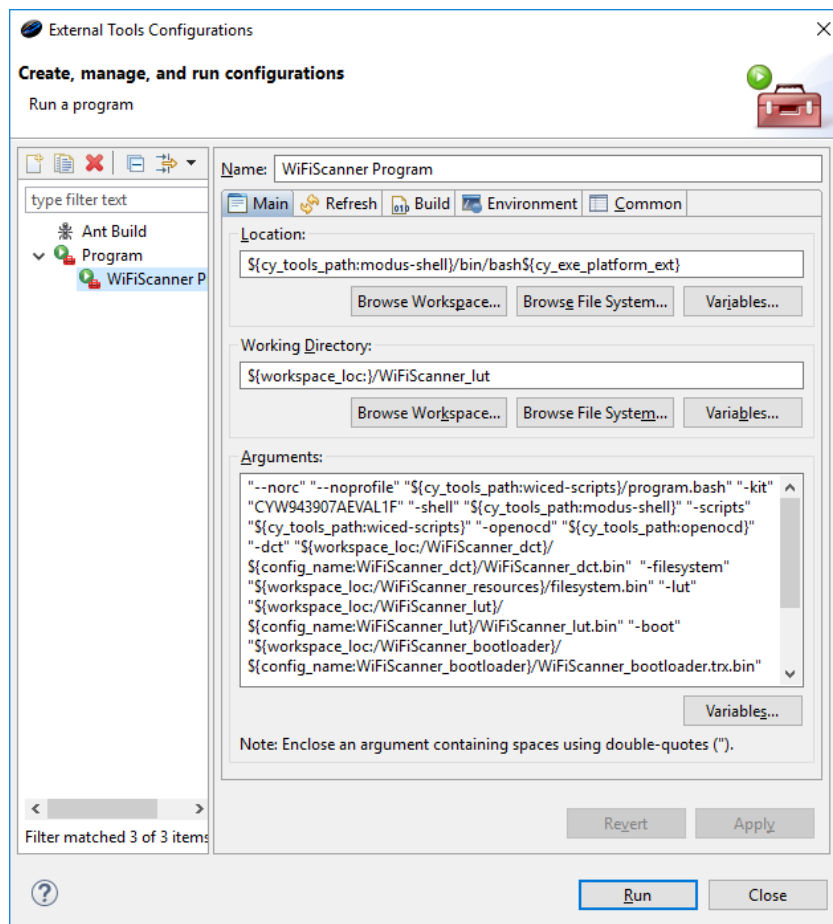
Messages appear in the console, indicating whether the build was successful (see Figure 3-13).

Figure 3-13. Build Console Messages



8. Program the Starter Project. To program the device, click the **External Tools** button. The first time you do this, the **External Tools Configurations** dialog opens. On the left side, expand **Program** and select the project, and click **Run**.

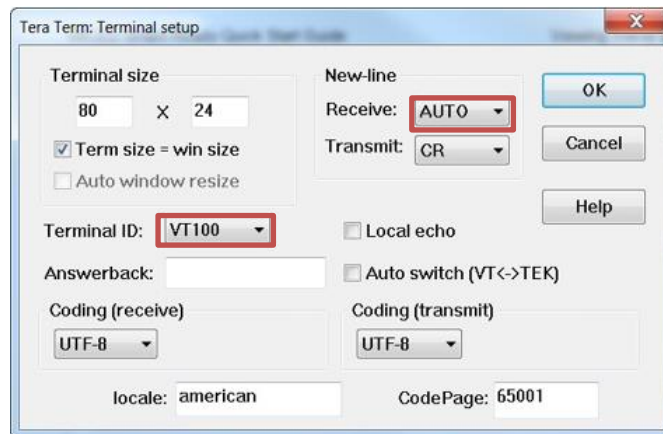
Figure 3-14. External Tool Configuration for Programming the Project



If needed, the IDE builds the project and displays messages in the console. If the build is successful, device programming starts immediately. If there are build errors, error messages appear in the console.

9. Do the following to view output messages with a terminal emulation program (such as Tera Term):
 - a. Start the terminal emulation program.
 - b. Set Terminal ID to **VT100** and New-Line **Receive** to **AUTO**. Leave other settings at their default values.

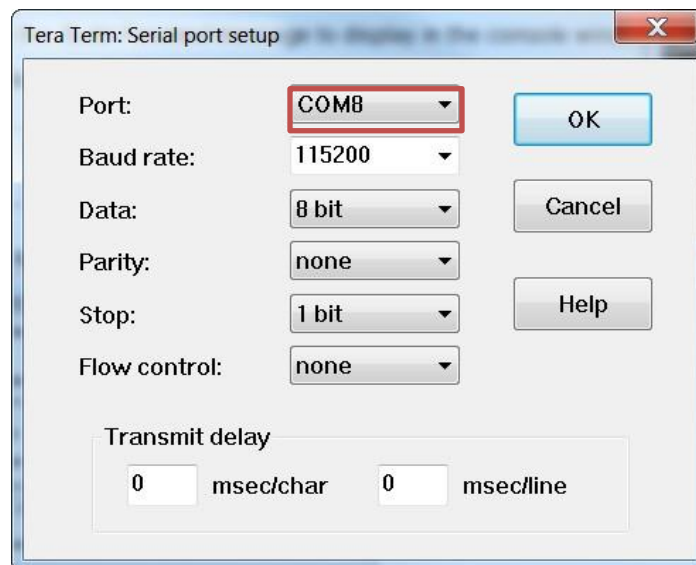
Figure 3-15. Tera Term Configuration



- c. In the terminal emulator, initiate a connection with the serial port number listed on Device Manager as shown in Figure 3-3.

Note: The exact port number will vary with the corresponding PC port.

Figure 3-16. Serial Port Setup



- d. Press the **Reset** button (see Figure 1-1) on the CYW943907AEVAL1F EVK to view application startup messages. The terminal emulation program should look similar to Figure 3-17.

Figure 3-17. Console Output

```

COM33 - Tera Term VT
File Edit Setup Control Window Help

Starting WICED Wiced_ModusToolbox_001.000.000
Platform CYW943907AEVAL1F initialised
Started ThreadX ver5.8
WICED_core Initialized
Initialising NetX_Duo ver5.10_sp3
Creating Packet pools
WLAN MAC Address : A4:08:EA:D9:C0:B2
WLAN Firmware : w10: Oct 23 2017 03:40:42 version 7.15.168.101 (r674438) FWID 01-13cae12
WLAN CLM : API: 12.2 Data: 9.10.74 Compiler: 1.31.3 Clmimport: 1.36.3 Creation: 2017-10-23 03:36:41
Waiting for scan results...
# Type BSSID RSSI Rate Chan Security SSID CCode Flag
-----
0 Infra C0:FF:D4:93:3A:AE -62 216.7 36 Open NETGEAR22-5G
1 Infra F0:5C:19:8A:8B:70 -90 450.0 36 WPA2 AES Enterprise CYFI IN PROBE
2 Infra F0:5C:19:8A:9A:30 -80 450.0 44 WPA2 AES Enterprise CYFI IN PROBE
3 Infra F0:5C:19:8A:9A:31 -80 54.0 44 WEP CYPHONE IN PROBE
4 Infra F0:5C:19:8A:9A:32 -80 450.0 44 Open CYFI_GUEST IN PROBE
5 Infra F0:5C:19:8A:9A:33 -80 450.0 44 WPA2 AES PSK CY-IOT-HOTSPOT IN PROBE
6 Infra F0:5C:19:8A:9A:34 -80 54.0 44 WEP CYTestNexus IN PROBE
7 Infra F0:5C:19:8A:8C:D0 -78 450.0 149 WPA2 AES Enterprise CYFI IN PROBE
8 Infra F0:5C:19:8A:9B:90 -87 450.0 149 WPA2 AES Enterprise CYFI IN BEACON
9 Infra F0:5C:19:8A:8C:D1 -77 54.0 149 WEP CYPHONE IN PROBE
10 Infra F0:5C:19:8A:8C:D2 -78 450.0 149 Open CYFI_GUEST IN PROBE
11 Infra F0:5C:19:8A:8C:D3 -77 450.0 149 WPA2 AES PSK CY-IOT-HOTSPOT IN PROBE
12 Infra F0:5C:19:8A:9B:96 -88 450.0 149 WPA2 AES PSK IN BEACON
13 Infra F0:5C:19:8A:8C:D4 -77 54.0 149 WEP CYTestNexus IN BEACON
14 Infra F0:5C:19:8A:8C:D5 -77 450.0 149 WPA2 AES PSK IN BEACON
15 Infra F0:5C:19:8A:9B:94 -89 54.0 149 WEP CYTestNexus IN PROBE
16 Infra F0:5C:19:8A:AE:A0 -66 216.7 1 WPA2 AES Enterprise CYFI PROBE
17 Infra F0:5C:19:8A:AE:A2 -67 216.7 1 Open CYFI_GUEST PROBE
18 Infra F0:5C:19:8A:8C:A2 -75 216.7 1 Open CYFI_GUEST PROBE
19 Infra F0:5C:19:8A:AE:A3 -70 216.7 1 WPA2 AES PSK CY-IOT-HOTSPOT PROBE
20 Infra F0:5C:19:8A:AE:A4 -69 54.0 1 WEP CYTestNexus PROBE
21 Infra F0:5C:19:8A:9B:84 -85 54.0 1 WEP CYTestNexus PROBE
22 Infra F0:5C:19:8A:9B:80 -85 216.7 1 WPA2 AES Enterprise CYFI PROBE
23 Infra F0:5C:19:8A:AE:A1 -68 54.0 1 WEP CYPHONE PROBE
24 Infra F0:5C:19:8B:67:00 -79 216.7 1 Open CYFI_GUEST BEACON
25 Infra F0:5C:19:8A:9B:82 -76 216.7 1 Open CYFI_GUEST BEACON
26 Infra F0:5C:19:8A:9B:83 -82 216.7 1 WPA2 AES PSK CY-IOT-HOTSPOT BEACON
27 Infra F0:5C:19:8A:9B:85 -82 216.7 1 WPA2 AES PSK BEACON
28 Infra C0:FF:D4:93:3A:AF OFF 216.7 5 Open NETGEAR22 PROBE
29 Infra F0:5C:19:8A:9A:20 -78 216.7 5 WPA2 AES Enterprise CYFI BEACON
30 Infra F0:5C:19:8A:9A:21 -82 54.0 5 WEP CYPHONE BEACON
31 Infra F0:5C:19:8A:9A:22 -77 216.7 5 Open CYFI_GUEST BEACON
32 Infra F0:5C:19:8A:9A:23 -81 216.7 5 WPA2 AES PSK CY-IOT-HOTSPOT BEACON
33 Infra F0:5C:19:8A:9A:25 -82 216.7 5 WPA2 AES PSK BEACON

```

3.4.2 Debugging a Project for CYW943907AEVAL1F in ModusToolbox


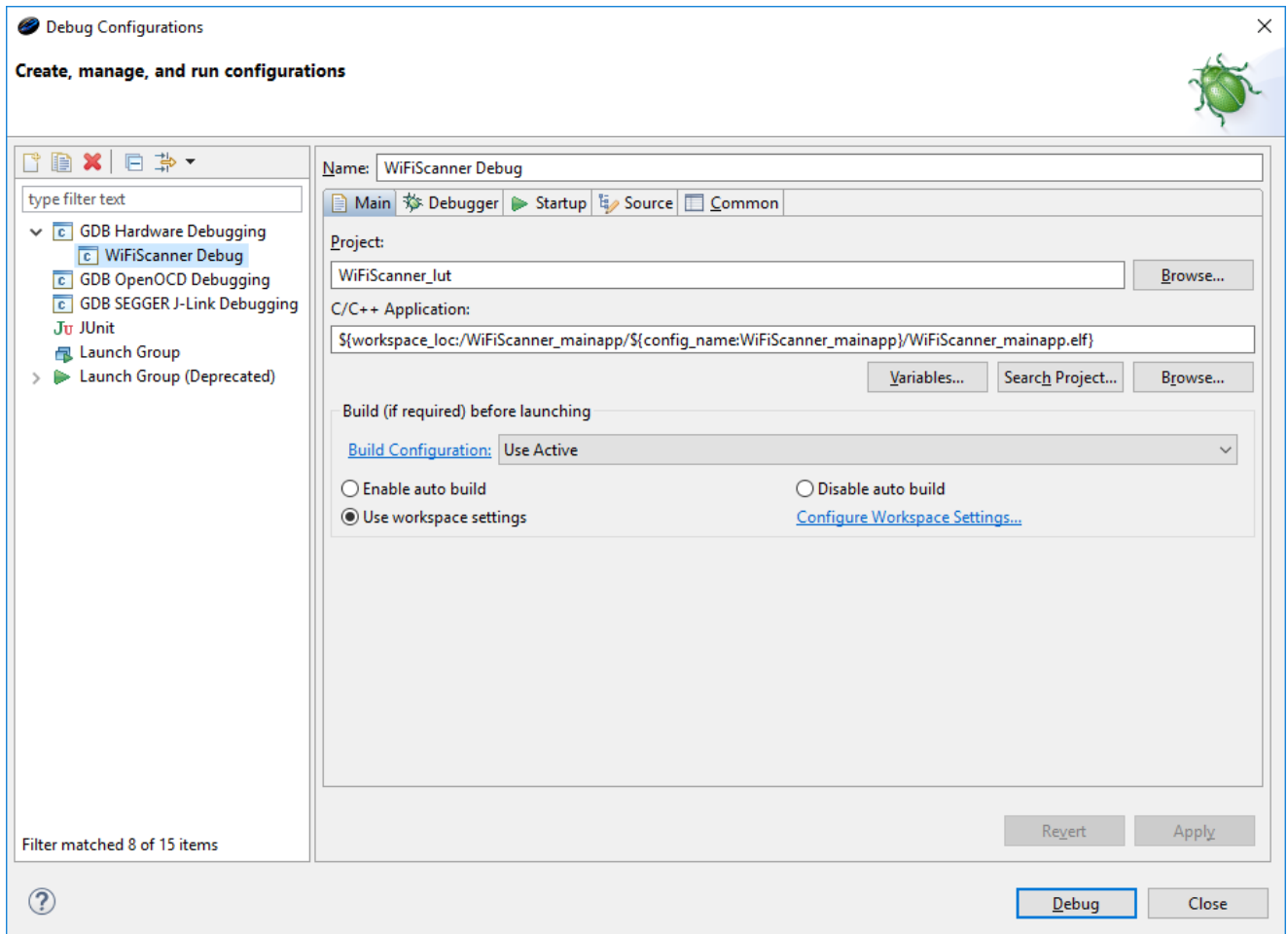
1. Click the **Debug**  button. The first time you do this, the Debug Configurations dialog appears. You can select the appropriate configuration to use for debugging.

Figure 3-18. Debug Configuration

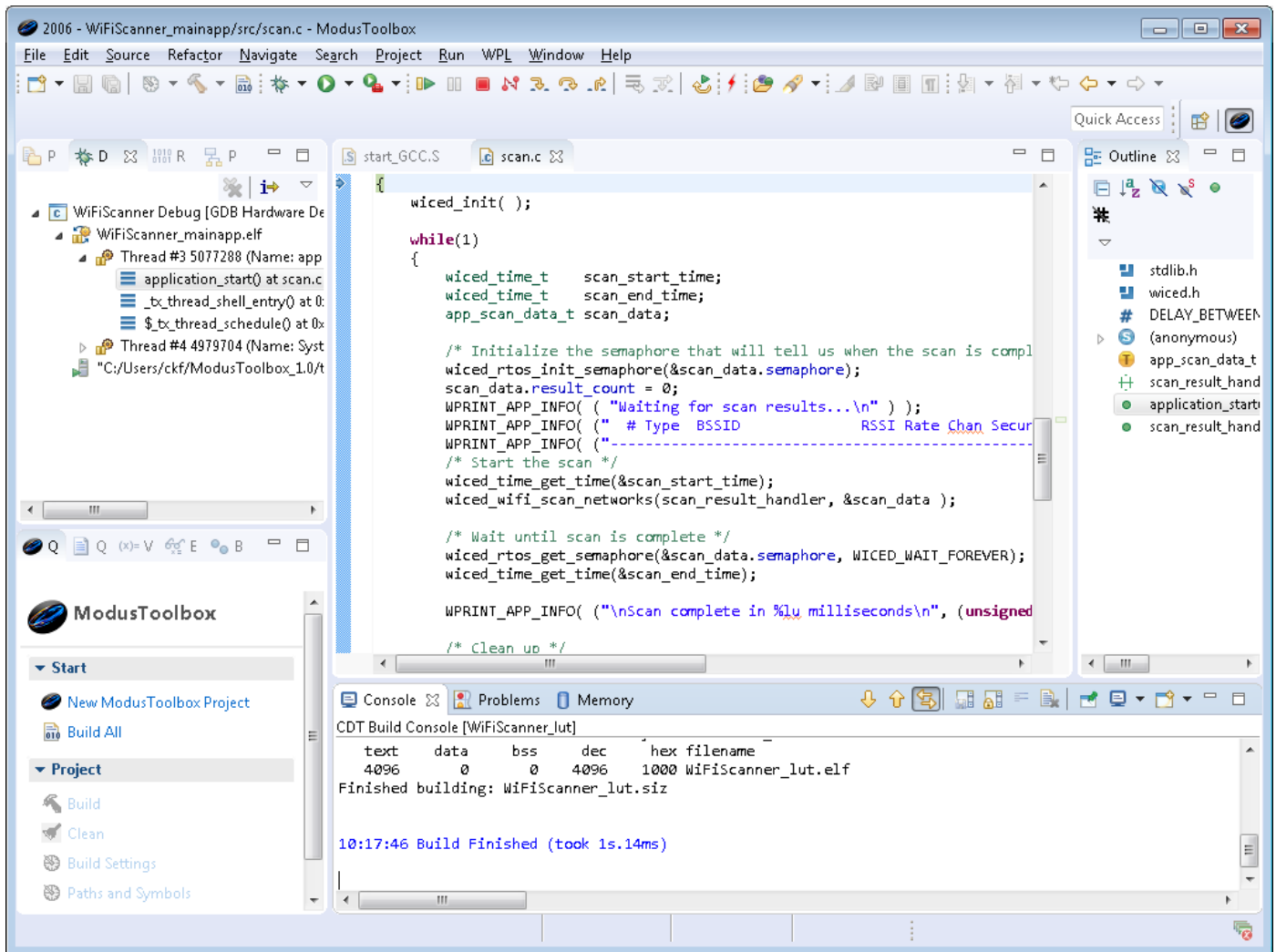


2. On the Debug Configuration dialog, select the WiFiScanner Debug configuration and click **Debug**.

If required, the IDE builds the project and messages display in the console. If the build is successful, the IDE switches to debug mode automatically. If there are build errors, error messages appear.

When a successful build is completed, the Modus IDE switches to debug mode as shown in Figure 3-19.

Figure 3-19 Modus IDE Debug Window



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 as part of the *CYW943907AEVAL1F Hardware Files.zip* file at <http://www.cypress.com/documentation/development-kitsboards/cyw943907aeval1f-evaluation-kit>.

Bootstrap options available in the CYW943907AEVAL1F EVK are shown in [Table 4-1](#). Pins are sampled at power-on reset (POR) to determine various operating modes. Sampling occurs a few milliseconds after an internal POR or deassertion 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 because 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 = SOCRAM 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
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 32 kHz LPO	0	0

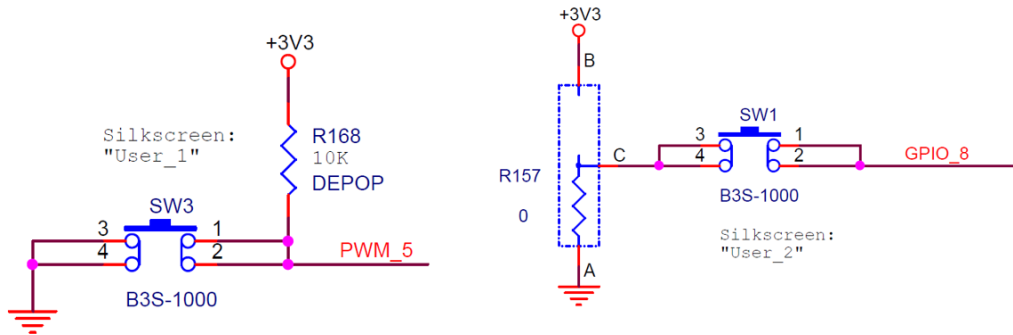
4.1 User Switches

There are two user switches available on the board. These are named USER_1 and USER_2. Table 4-2 shows the pin names and enumeration used in the SDK for the switches.

Table 4-2. User Switch available on the board

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

Figure 4-1. User Switch Circuit Diagram



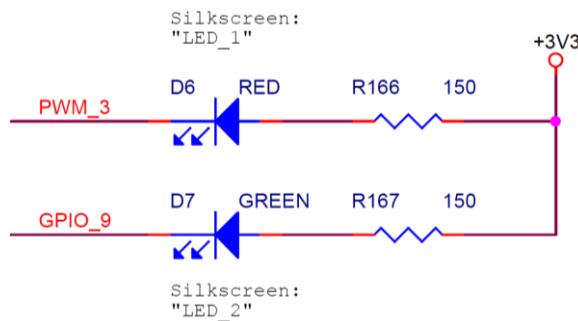
4.2 LED

There are two user LEDs available. These are named LED_1 and LED_2. Table 4-3 shows the pin name and enumeration used in SDK for these LEDs.

Table 4-3. User LED Available on the Board

SWITCH	CYW43907 PIN NAME	WICED_ENUM_ID	ALTERNATE ENUMERATION IN SDK
LED_1	PWM_3	WICED_GPIO_16	WICED_LED1
LED_2	GPIO_9	WICED_GPIO_5	WICED_LED2

Figure 4-2. User LED Circuit Diagram



4.3 Reset Control

The CYW43907 device can be reset using the “Target Reset” switch SW2 or a reset command from the onboard programmer/debugger and serial interface chip, as shown in Figure 4-3. 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 proper bootup, the RC delay circuit for HIB_REG_ON_IN is essential as shown in Figure 4-4.

Figure 4-3. Reset Circuit Diagram

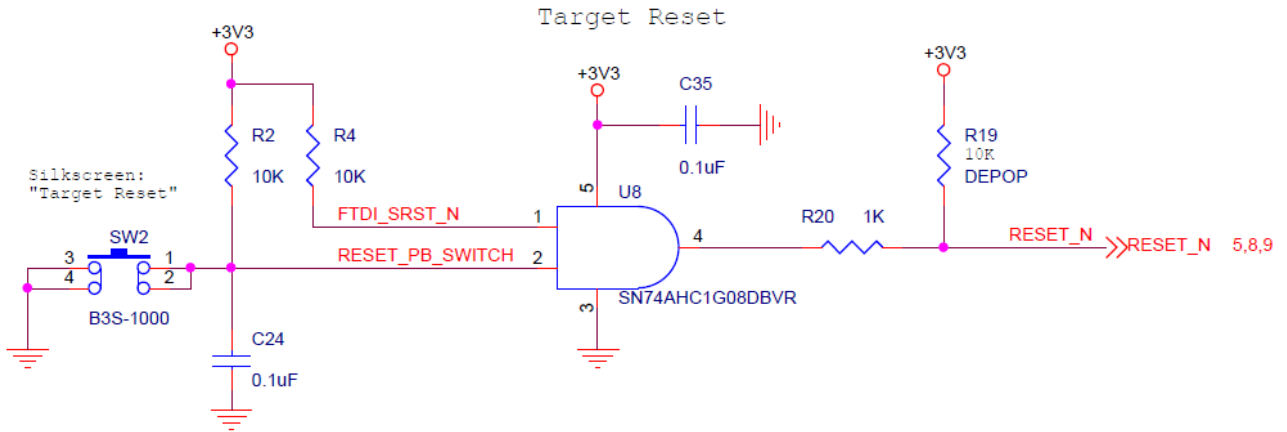
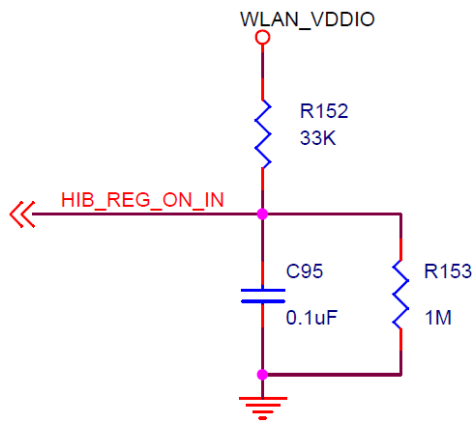


Figure 4-4. HIB_REG_ON_IN RC Delay Circuit



4.4 Ethernet

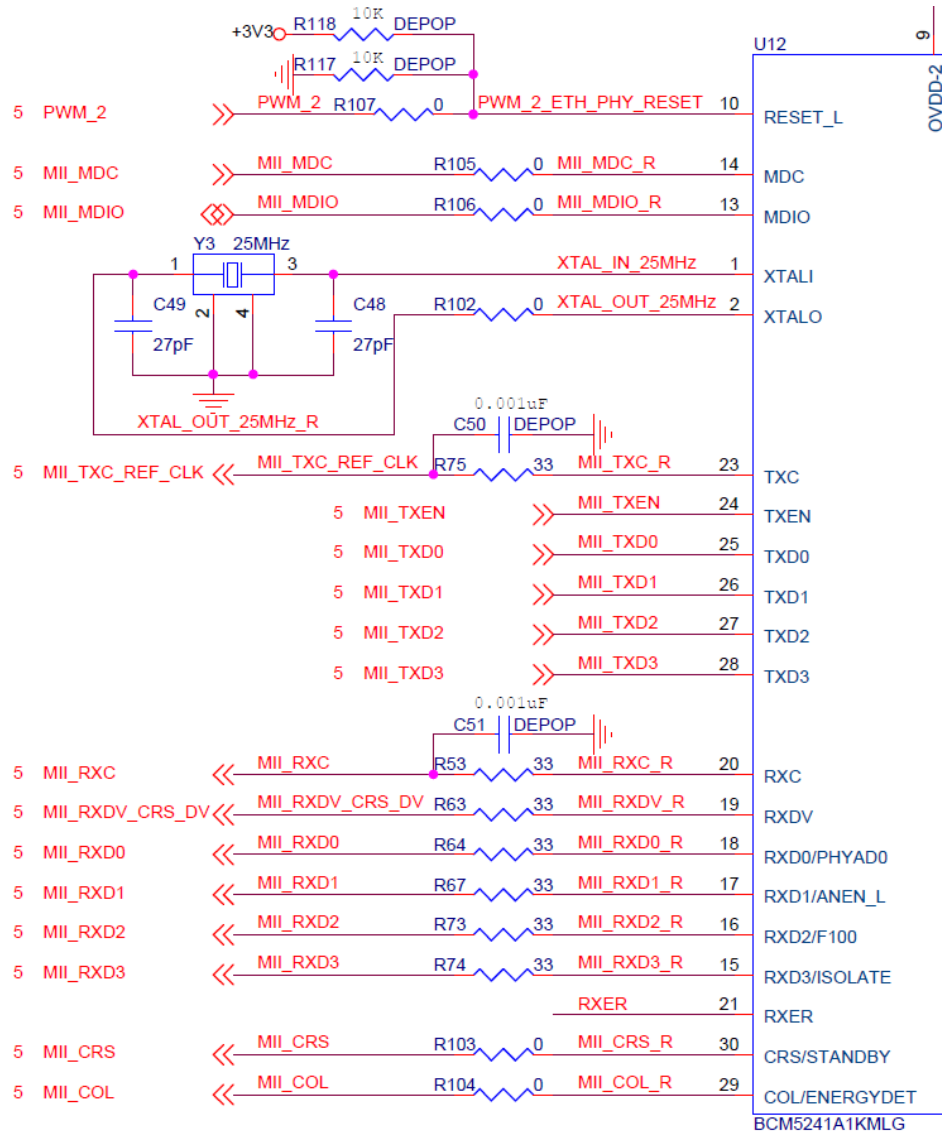
The Ethernet MAC Controller in CYW43907 interfaces with an external PHY chip, BCM5241, using the Media Independent Interface (MII) as shown in Figure 4-5. The same signals are also listed in Table 4-4. CYW43907 also supports Reduced Media Independent Interface (RMII). The controller can transmit and receive data at 10 Mbps and 100 Mbps.

Table 4-4. CYW43907 EMAC to PHY Chip Connection

SI.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

Sl.No.	CYW43907 Pin Name	Net Name in Schematic	BCM5241 Pin Name
6	RMII_G_TXD1	MII_TXD1	TXD1
7	RMII_G_TXD2	MII_TXD2	TXD2
8	RMII_G_TXD3	MII_TXD3	TXD3
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_CRSDV	RXDV
17	PWM_2	PWM_2	RESET_L

Figure 4-5. Ethernet MAC Controller to External PHY Connection



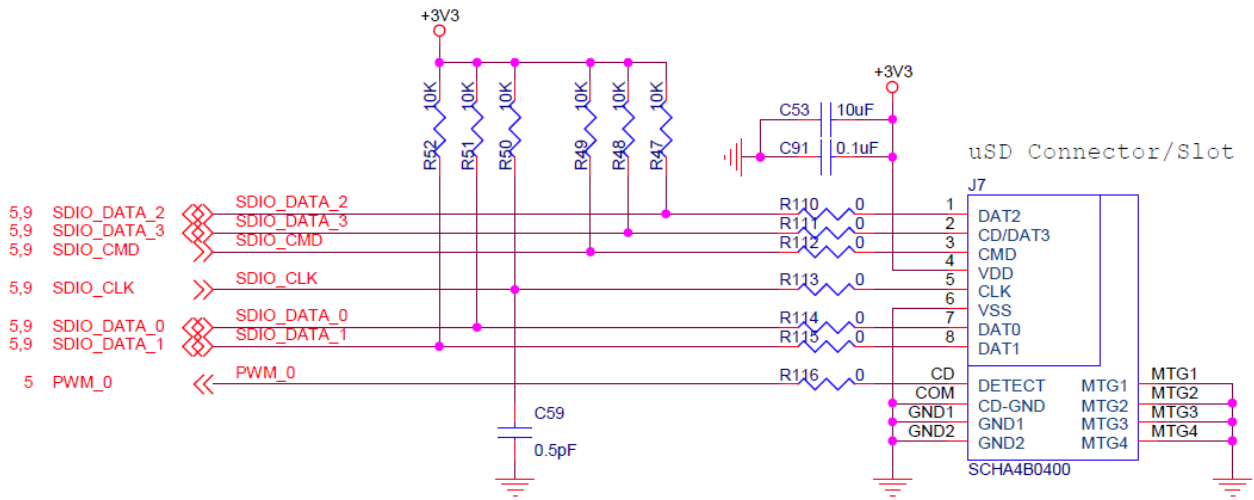
4.5 Micro SD Connector/Slot

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 4-6 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

S 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 4-6. Micro SD Connector Circuit Diagram



4.6 JTAG Connector

4.6.1 Onboard Programmer/Debugger and Serial Interface Chip

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

Table 4-6 shows the connection between CYW43907 and the onboard 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 CYW43907.

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

Sl. No.	CYW43907-Based SiP Pin Name	Onboard 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 Connectors

4.7.1 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 Onboard Programmer/debugger chip (UART1). Table 4-7 illustrates the J6 pinout.

Table 4-7. WICED Header Pinout

EVAL BOARD HEADER	CYW43907 PIN NAME	SDK 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

EVAL BOARD HEADER	CYW43907 PIN NAME	SDK ENUMERATION	ALTERNATE ENUMERATION
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
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

4.7.2 Arduino-Compatible Headers

J9, J13, J12, and J10 are 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:

- The 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	SDK 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
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

4.8 UART Port Configuration on CYW943907AEVAL1F Kit

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 onboard programmer/debugger chip for UART-to-USB communication. UART peripherals are defined in the following file:

<installed_location>/ModusToolbox_1.0/libraries/wiced_base-1.0/components/WIFI-SDK/platforms/CYW943907AEVAL1F/platform.c.

The following table (also available in *platforms/CYW943907AEVAL1F/platform.h*) shows the UART pins available on the kit.

Table 4-9.

SDK PERIPHERAL ENUMERATION ID	PIN NAME ON CYW43907	MURATA MODULE PIN NAME	HEADER PIN NUMBER	SDK 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

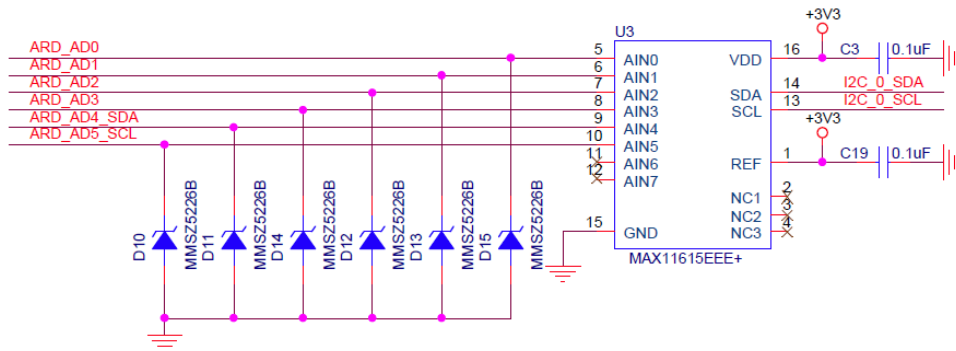
4.9 External ADC

CYW43907 does not have any built-in 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 4-10 lists the connections between CYW43907. The external ADC circuit diagram is shown in Figure 4-7.

Table 4-10. External ADC Connection

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

Figure 4-7. External ADC Circuit Diagram



4.10 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*. These PWMs can be reassigned to other pins by changing the first argument of the `platform_pwm_t platform_pwm_peripherals` structure in *platform.c*. Table 4-11 through Table 4-16 show the possible combinations and their Arduino header locations.

Table 4-11. 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 4-12. 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	<input type="checkbox"/>	<input type="checkbox"/>

Table 4-13. WICED_PWM_3 Combinations

Pin MUX Selection	Header Pin	Header Name
PIN_GPIO_16 (DEFAULT)	J10.6	Arduino D5
PIN_GPIO_8	<input type="checkbox"/>	<input type="checkbox"/>
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	<input type="checkbox"/>	<input type="checkbox"/>
PIN_PWM_2	<input type="checkbox"/>	<input type="checkbox"/>

Table 4-14. 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 MUX SELECTION	HEADER PIN	HEADER NAME
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 4-15. 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 4-16. 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 CYW943907AEVAL1F EVK code examples. The code examples are already bundled with the SDK, which can be found in `<InstallationFolder>ModusToolbox_1.0\libraries\wiced_base-1.0\examples\apps\snips`.

Code examples can be compiled after creating the ModusToolbox project. Refer to [Building and Programming a Project for CYW943907AEVAL1F](#) in for details on creating, building, and downloading projects.

5.1 AWSIOTPubSub Project

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](#).

On startup, the `AWSIOTPubSub` code example joins a Wi-Fi access point specified in the `/AWSIOTPubSub_mainapp/src/wifi_config_dct` file, connects to AWS, subscribes to the specified topic, and then alternately tries to publish and subscribe messages.

The project consists of the following files:

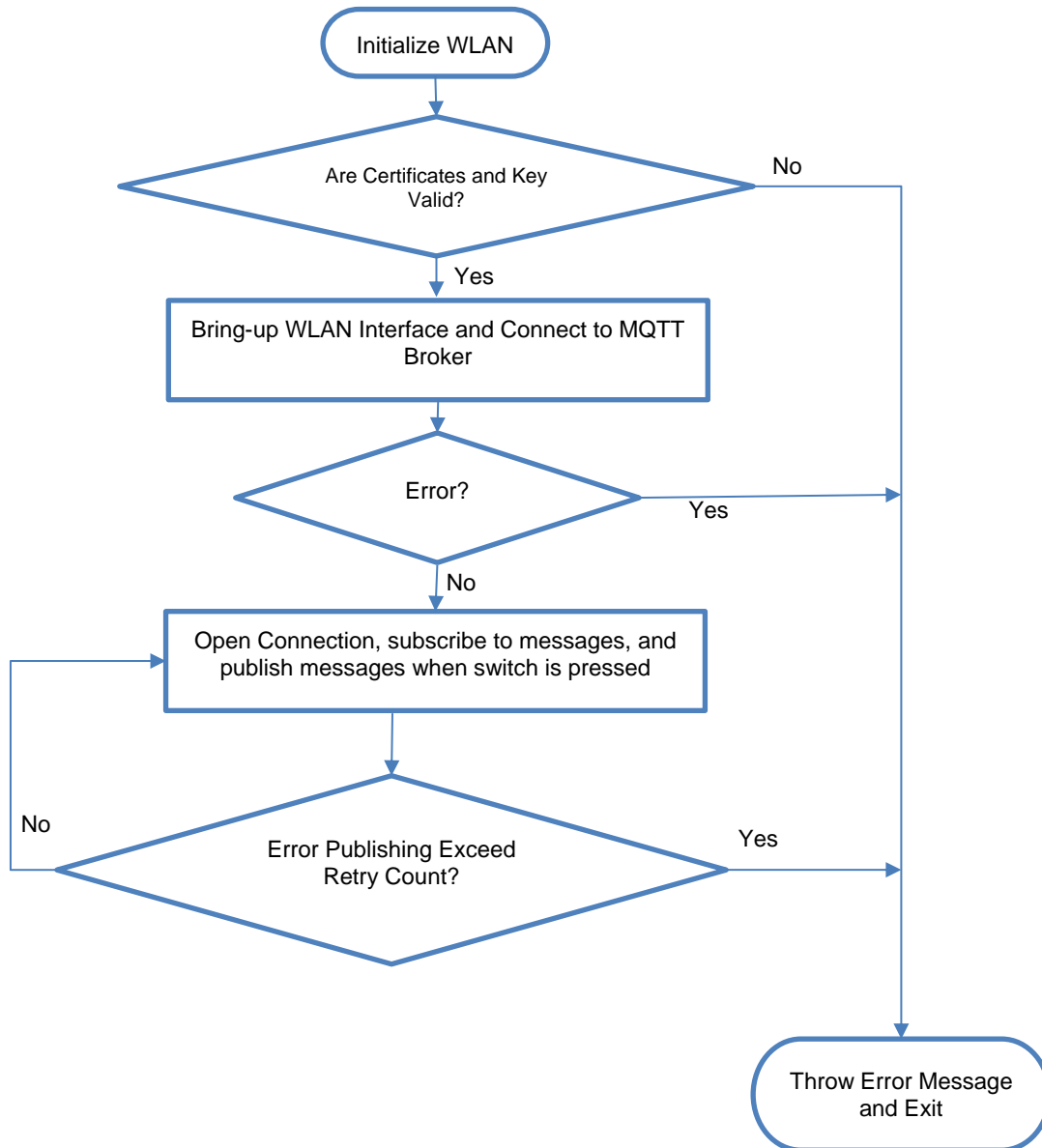
- `aws_app.c`: This file contains the main application function `application_start()` which is the entry point and execution of the firmware application. It also contains function definitions for initializing, publishing, and subscribing to `AWSaws_config.h`: This file contains all the necessary macro definitions such as `WICED_TOPIC` and `WICED_MESSAGESTR`.
- `wifi_config_dct.h`: This file contains the Wi-Fi access point credentials (SSID and pass phrase key) and soft AP credentials. You should enter the client access point name and password credentials before 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.

5.1.1 Hardware Connections

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

5.1.2 Flowchart

Figure 5-1.



5.1.3 Verify Output

5.1.3.1 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](#).

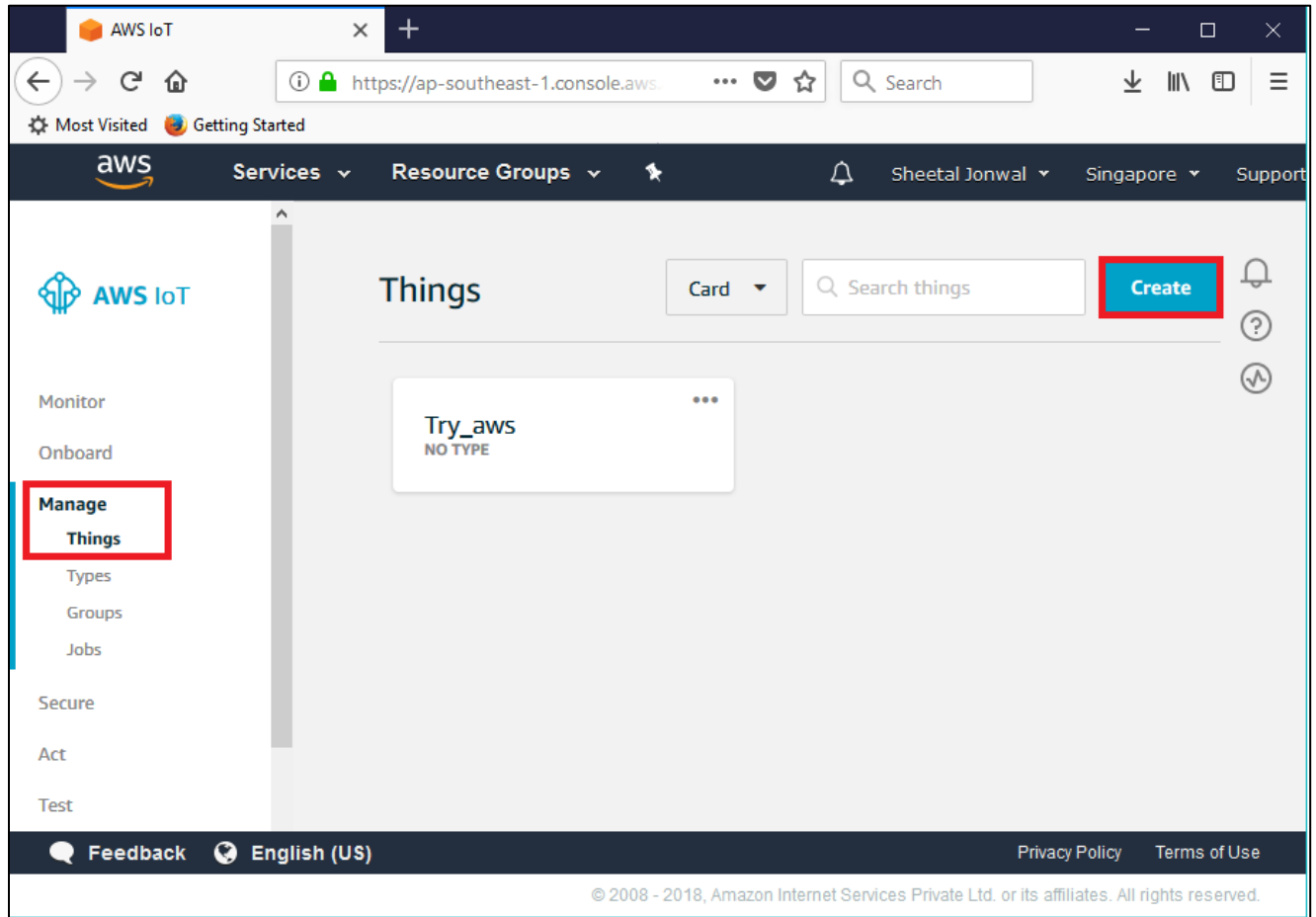
1. Open <https://aws.amazon.com> and choose **Create an AWS Account**.
2. Follow the online instructions. Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

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.

5.1.3.2 Create a Thing

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 5-2.

Figure 5-2. 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 that you create a Thing.

Figure 5-3. Create a Thing

CREATE A THING
STEP 1/3

Add your device to the thing registry

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 ▾

Create a type

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

Figure 5-4. Add a Certificate to Your Thing

CREATE A THING
STEP 2/3

Add a certificate for your thing

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 certificate

Create with CSR

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

📄 Create with CSR

Use my certificate

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

Get started

Skip certificate and create thing

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

Create thing without certificate

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

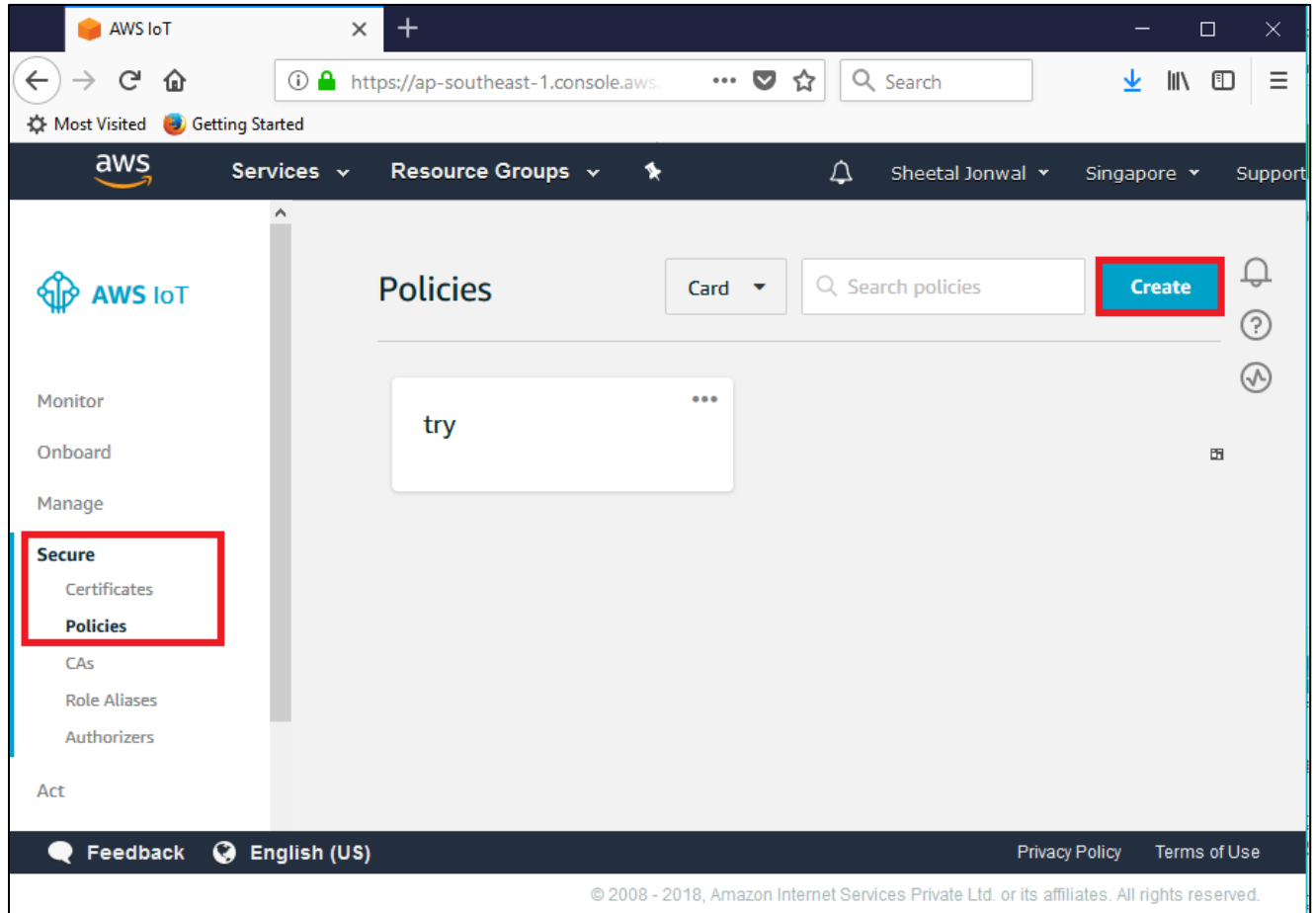
CYW943907AEVAL1F Evaluation Kit User Guide, Doc. No. 002-18703 Rev. *B

42

5.1.3.3 Create a policy

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

Figure 5-5. 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:***.
4. Assign an Amazon Resource Name (ARN) in the **Resource ARN** field. To use a wildcard, change the last part of the Resource ARN as follows:

```
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”.

5. Select the check box **Allow Effect** as shown in [Figure 5-6](#).

Figure 5-61. 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

Add statements
 Policy statements define the types of actions that can be performed by a resource. **Advanced mode**

Action

Resource ARN

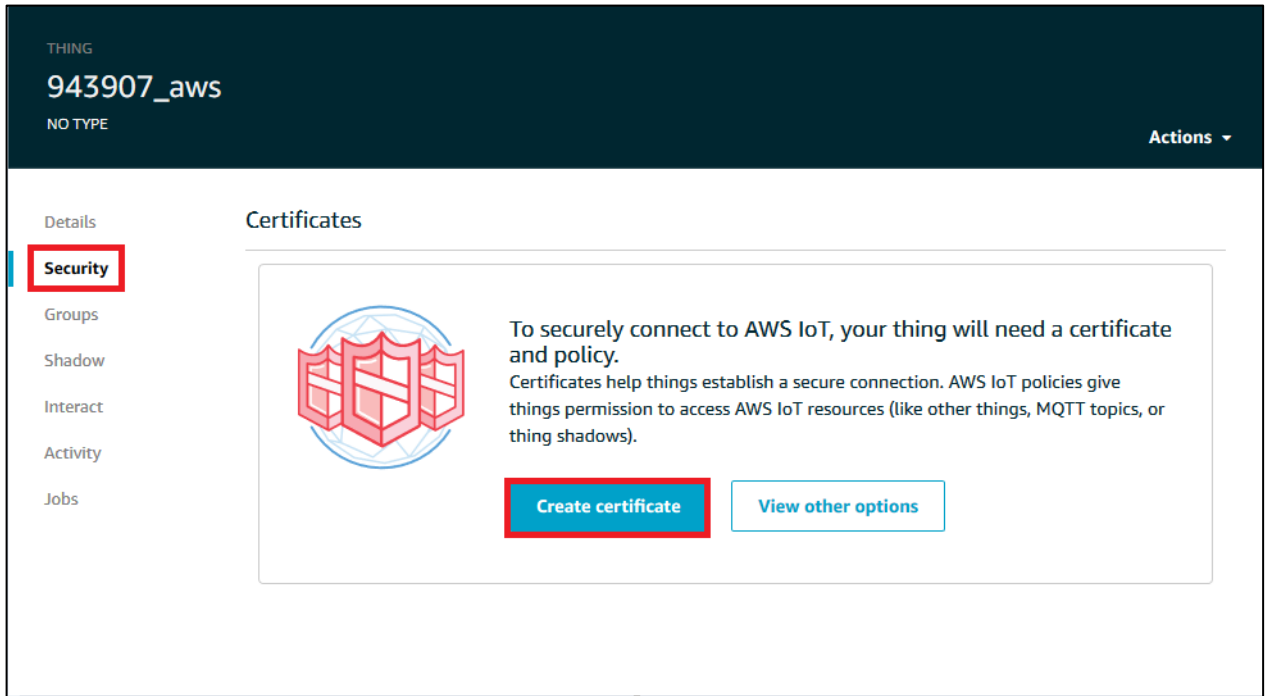
Effect
 Allow Deny

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

5.1.3.4 Create a Certificate for a Thing

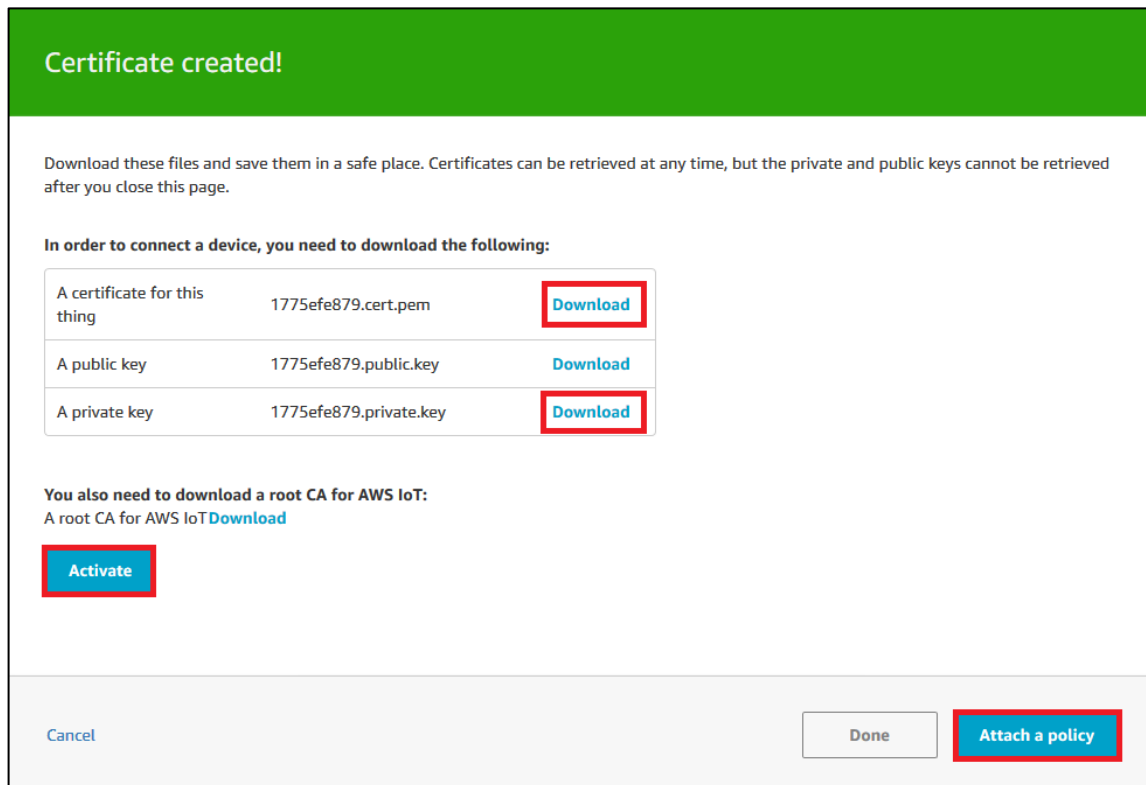
1. In the AWS IoT Console window, go to **Manage > Things** and then click on the created thing (for example, 943907_aws). The created Thing window appears.
2. In the left navigation pane, click on **Security**, then **Create certificate** as shown in [Figure 5-7](#).

Figure 5-7 Create Certificates



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 5-8](#).

Figure 5-8. Download and Activate Certificates

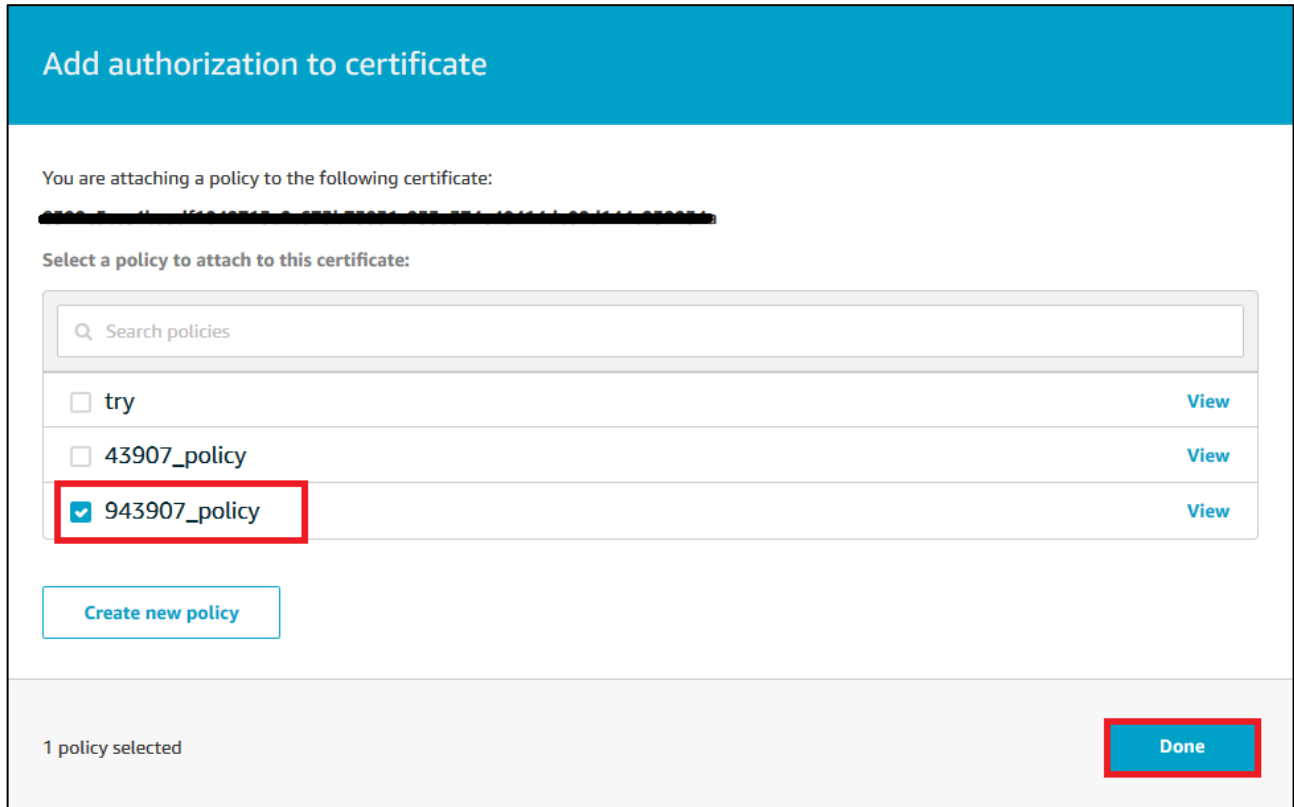


Notes:

- The certificate and private key cannot be revisited later for download and must be saved while creating the certificate.

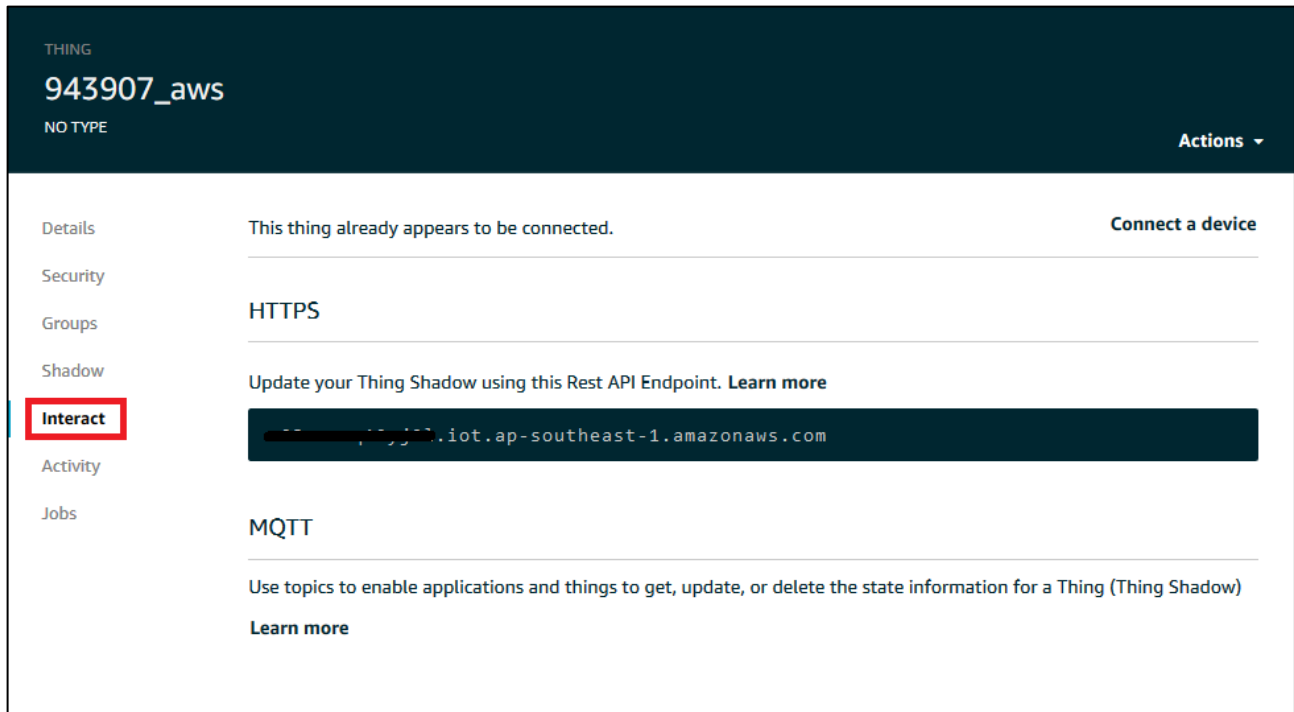
- Copy the content of the downloaded Thing certificate and paste in the client.cer in the location `C:\Users\<User_name>\ModusToolbox_1.0\libraries\wiced_base-1.0\examples\apps\snips\laws_iot_app\client.cer`
Copy the content of downloaded Private key and paste in `privkey.cer` in the following location `C:\Users\<User_name>\ModusToolbox_1.0\libraries\wiced_base-1.0\examples\apps\snips\laws_iot_app\privkey.cer`.
5. Select the check box next to the policy you want to choose, and click **Done**, as shown in [Figure 5-9](#).

Figure 5-9. Register a Thing



6. 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 whether the correct policy is linked.
9. Click **Things** in the left-hand panel to validate whether 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 5-10](#).

Figure 5-10. Endpoint



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.

5.1.3.5 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 *AWSIOTPubSub_bootloader_Base_WICED/wiced_base-1.0/components/WIFI-SDK/WICED/WWD/include/wwd_constants.h*

5.1.3.6 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](#).

After it has been 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 alternately as shown in [Figure 5-11](#). 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** to see the number of messages exchanged as shown in [Figure 5-12](#).

Figure 5-11. Publish_Subscribe_aws Output

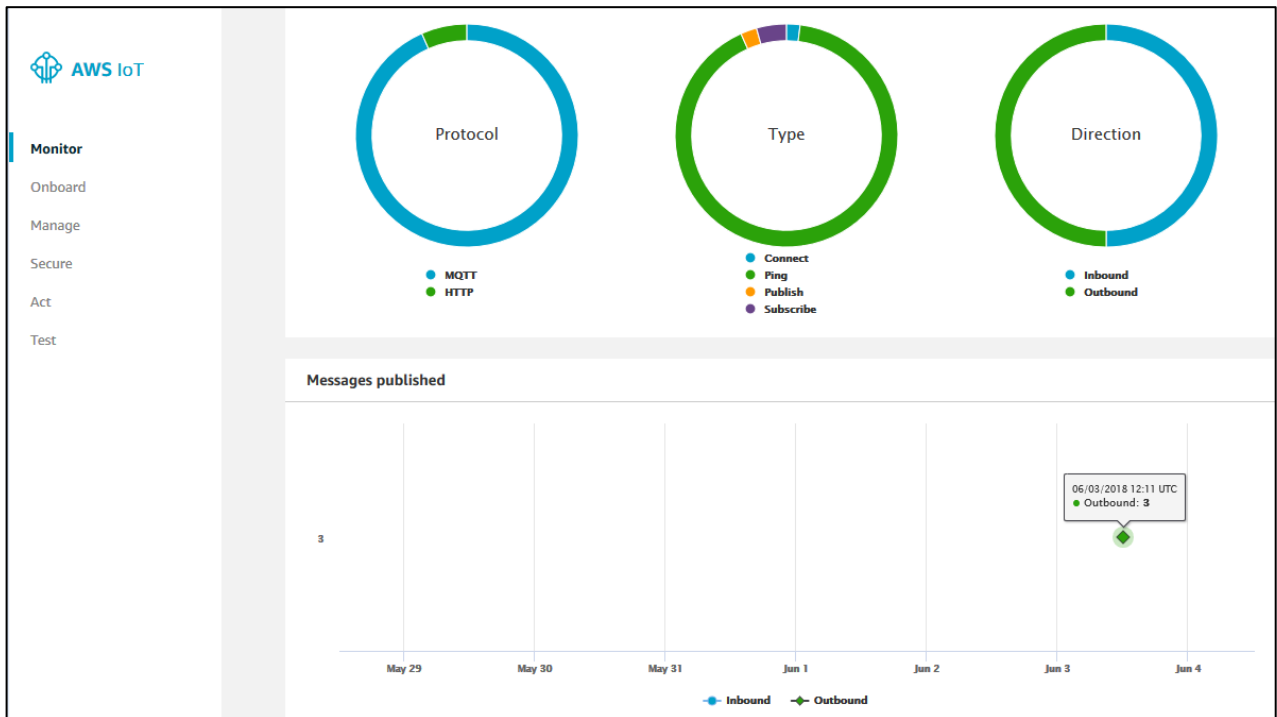
```

COMS - Tera Term VT
File Edit Setup Control Window Help

Starting WICED Wiced ModusToolbox_001.000.000
Platform CYW943907AEVAL1F initialised
Started ThreadX ver5.8
WICED_core Initialized
Initialising NetX_Duo ver5.10_sp3
Creating Packet pools
WLAN MAC Address : A4:08:EA:D9:D4:E6
WLAN Firmware : wl0: Oct 23 2017 03:40:42 version 7.15.168.1
WLAN CLM : API: 12.2 Data: 9.10.74 Compiler: 1.31.3 Clr
Joining : CY-IOT-HOTSPOT
Failed to join : CY-IOT-HOTSPOT
Joining : CY-IOT-HOTSPOT
Successfully joined : CY-IOT-HOTSPOT
Obtaining IPv4 address via DHCP
DHCP CLIENT hostname WICED IP
IPv4 network ready IP: 10.40.2.46
Setting IPv6 link-local address
IPv6 network ready IP: FE80:0000:0000:0000:A608:EAFF:FED9:D4E6
Resolving IP address of AWS cloud...
Resolved Broker IP: 52.74.98.203

[AWS] Opening connection...OK.
[AWS] Subscribing...OK.
[AWS] Publishing...OK.
[AWS] Waiting some time for ping exchange...
[AWS] Received HELLO WICED for TOPIC : MQTT/WICED/TOPIC
    
```

Figure 5-12. Messages Published



5.2 WiFiScanner

5.2.1 Project Description

The WiFiScanner project demonstrates regular scanning of nearby Wi-Fi Access Points (APs). The WiFiScanner mainapp project consists of the following files:

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

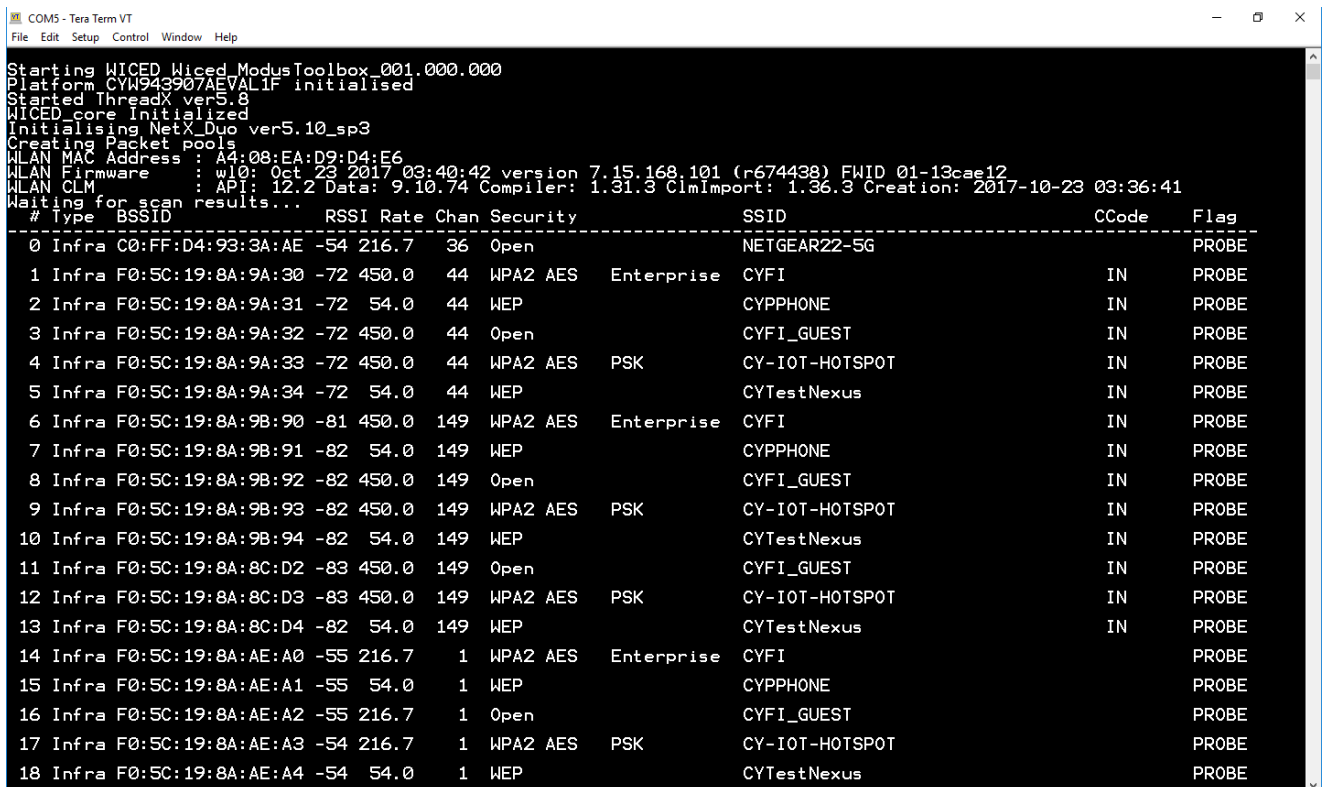
5.2.2 Hardware Connections

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

5.2.3 Verify Output

1. Create and program the project using the description specified in [Building and Programming a Project for CYW943907AEVAL1F](#) in .
2. After initialization of the platform, the example prints relevant information like SSID, country code, security etc of all the nearby Access Points.as shown in Figure 5-13..
3. 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.

Figure 5-13. Console Output for WiFiScanner Application



```

COM5 - Tera Term VT
File Edit Setup Control Window Help
Starting WICED Wiced_ModusToolbox_001.000.000
Platform CYW943907AEVAL1F initialised
Started ThreadX ver5.8
WICED_core Initialized
Initialising NetX_Duo ver5.10_sp3
Creating Packet pools
WLAN MAC Address : A4:08:EA:D9:D4:E6
WLAN Firmware : w10: Oct 23 2017 03:40:42 version 7.15.168.101 (r674438) FWID 01-13cae12
WLAN CLM : API: 12.2 Data: 9.10.74 Compiler: 1.31.3 ClmImport: 1.36.3 Creation: 2017-10-23 03:36:41
Waiting for scan results...
-----
# Type BSSID RSSI Rate Chan Security SSID CCode Flag
-----
0 Infra C0:FF:D4:93:3A:AE -54 216.7 36 Open NETGEAR22-5G IN PROBE
1 Infra F0:5C:19:8A:9A:30 -72 450.0 44 WPA2 AES Enterprise CYFI IN PROBE
2 Infra F0:5C:19:8A:9A:31 -72 54.0 44 WEP CYPHONE IN PROBE
3 Infra F0:5C:19:8A:9A:32 -72 450.0 44 Open CYFI_GUEST IN PROBE
4 Infra F0:5C:19:8A:9A:33 -72 450.0 44 WPA2 AES PSK CY-IOT-HOTSPOT IN PROBE
5 Infra F0:5C:19:8A:9A:34 -72 54.0 44 WEP CYTestNexus IN PROBE
6 Infra F0:5C:19:8A:9B:90 -81 450.0 149 WPA2 AES Enterprise CYFI IN PROBE
7 Infra F0:5C:19:8A:9B:91 -82 54.0 149 WEP CYPHONE IN PROBE
8 Infra F0:5C:19:8A:9B:92 -82 450.0 149 Open CYFI_GUEST IN PROBE
9 Infra F0:5C:19:8A:9B:93 -82 450.0 149 WPA2 AES PSK CY-IOT-HOTSPOT IN PROBE
10 Infra F0:5C:19:8A:9B:94 -82 54.0 149 WEP CYTestNexus IN PROBE
11 Infra F0:5C:19:8A:8C:D2 -83 450.0 149 Open CYFI_GUEST IN PROBE
12 Infra F0:5C:19:8A:8C:D3 -83 450.0 149 WPA2 AES PSK CY-IOT-HOTSPOT IN PROBE
13 Infra F0:5C:19:8A:8C:D4 -82 54.0 149 WEP CYTestNexus IN PROBE
14 Infra F0:5C:19:8A:AE:A0 -55 216.7 1 WPA2 AES Enterprise CYFI IN PROBE
15 Infra F0:5C:19:8A:AE:A1 -55 54.0 1 WEP CYPHONE IN PROBE
16 Infra F0:5C:19:8A:AE:A2 -55 216.7 1 Open CYFI_GUEST IN PROBE
17 Infra F0:5C:19:8A:AE:A3 -54 216.7 1 WPA2 AES PSK CY-IOT-HOTSPOT IN PROBE
18 Infra F0:5C:19:8A:AE:A4 -54 54.0 1 WEP CYTestNexus IN PROBE

```

Revision History



Document Title: CYW943907AEVAL1F Evaluation Kit User Guide

Document Number: 002-18703

Revision	Issue Date	Origin of Change	Description of Change
**	06/22/2017	KAVS	Initial version
*A	06/18/2018	SHJL	Updated 'Section 5.5: Publish_subscribe_aws' based on the new user interface of Amazon AWS
*B	07/31/2018	RROY	Porting of existing WICED Studio document (002-18703 rev*A) to Modus IDE based development flow