

# Getting Started with PSoC 62 + 43xxx

This document provides instructions for getting started with the following kits.

- [PSoC 6 Wi-Fi BT Prototyping Kit \(CY8CPROTO-062-4343W\)](#)
- [PSoC 6 WiFi-BT Pioneer Kit \(CY8CKIT-062-WIFI-BT\)](#)
- [PSoC 62S2 WiFi-BT Pioneer Kit \(CY8CKIT-062S2-43012\)](#)

You can also purchase a kit through the above links.

## Before You Begin

1. Clone Amazon FreeRTOS from Cypress' GitHub [repository](#) and checkout the latest release using its [tag](#). Following command also clones all the submodules.

```
git clone --recurse-submodules
https://github.com/cypresssemiconductorco/amazon-freertos.git --
branch
<release-tag>
```

Example release tag: **201910-MTBAFR1951**

2. Configure AWS IoT and your Amazon FreeRTOS download to connect your device to the AWS Cloud. See [First Steps](#) for instructions. You must use only the version of Amazon FreeRTOS from Cypress' GitHub [repository](#) to work with Cypress kits and ignore the instructions given at Amazon FreeRTOS [user guide](#) for downloading it from elsewhere.
3. Once you have configured AWS IoT, you need to configure the following in Amazon FreeRTOS. These files are located under `<amazon-freertos>/demos/include` directory.
  - a. Wi-Fi settings (SSID, password) and MQTT endpoint in `aws_clientcredential.h`
  - b. Client certificate and private key in `aws_clientcredential_keys.h`

### Note:

1. In this tutorial, the path to the Amazon FreeRTOS download directory is referred to as `<amazon-freertos>`.
2. The maximum length of a file path on Microsoft Windows is 260 characters. Lengthy Amazon FreeRTOS download directory paths can cause build failures. Make sure that the path to the `<amazon-freertos>` directory is fewer than 98 characters.

## Setting Up Your Hardware

First, ensure your kit is programmed with the latest KitProg3 firmware. Follow these steps to update the KitProg3 firmware. See the firmware-loader [page](#) for detailed instructions.

1. Download the [latest firmware-loader](#) tool. The firmware-loader tool is also installed with the ModusToolbox IDE at the location `<install_dir>/ModusToolbox/tools_x.x/fw-loader-x.x`. See [Download and Install ModusToolbox](#) for details.
2. Connect the KitProg USB connector on the kit to your host computer.
  - KitProg3 USB (J8) connector for CY8CPROTO-062-4343W kit
  - KitProg3 USB (J10) connector for CY8CKIT-062-WIFI-BT kit
  - KitProg3 USB (J6) connector for CY8CKIT-062S2-43012 kit

3. Run the following command from the directory where the firmware-loader binary is located.  

```
fw-loader --update-kp3
```

**Note:** On Linux OS, you must run the `udev_rules\install_rules.sh` script before the first run of the `fw-loader`.
4. Ensure the kit is in KitProg3 mode to program it from the ModusToolbox IDE. The amber color STATUS LED (LED2) should be steady ON or blinking at 1 Hz rate. If not, press the SW3 MODE SELECT button to switch the mode to KitProg3.

## Setting Up the Development Environment

Amazon FreeRTOS works with Cypress's ModusToolbox IDE to program the kit. The ModusToolbox IDE is compatible with the Windows, OS X, and Linux operating systems. Before you begin, you must download and install the ModusToolbox IDE.

You can also use CMake to generate project build files from Amazon FreeRTOS application source code, build the project using your preferred build tool, and then program the kit using OpenOCD installed with the ModusToolbox IDE. If you prefer to use a GUI tool for programming, download and install Cypress Programmer from this [web page](#). See [Using CMake with Amazon FreeRTOS](#) for details.

### Download and Install ModusToolbox

Download and install the latest ModusToolbox software from the Cypress Community webpage. Downloading requires you to register with the Cypress Developer Community. Follow the instructions in the ModusToolbox Installation Guide also located on the Community webpage to install the software.

### Set Up a Serial Connection

1. Connect the kit to your host computer.
2. The USB Serial port is enumerated for the kit on the host computer. Identify the port number. In Windows, you can identify it using the Device Manager under Ports (COM & LPT).
3. Start a serial terminal and open a connection with the following settings:
  - Baud rate: 115200
  - Data: 8 bit
  - Parity: None
  - Stop bits: 1
  - Flow control: None

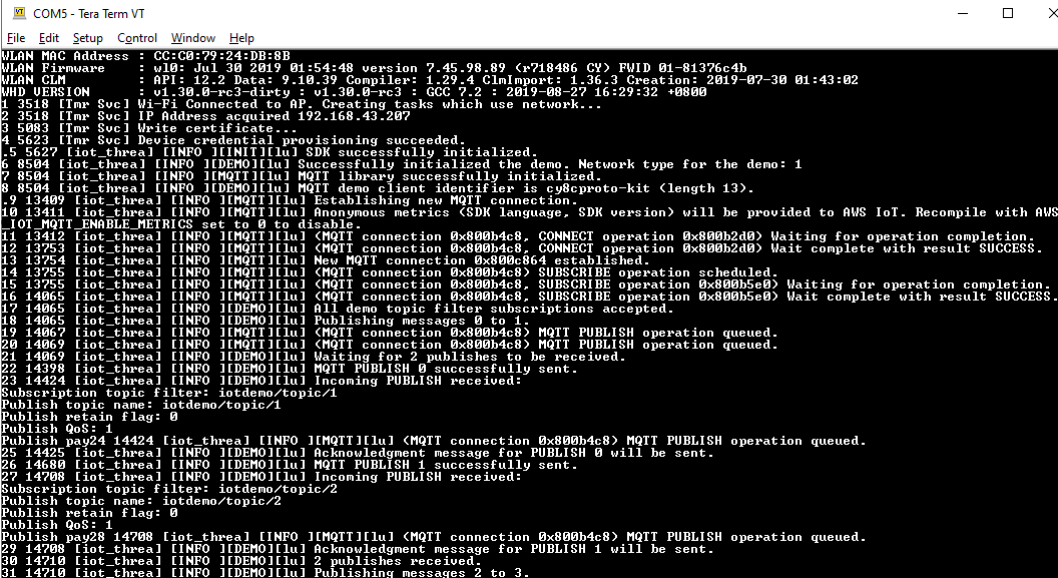
# Build and Run the Amazon FreeRTOS Demo Project

## Build the Amazon FreeRTOS Demo

1. Open the Eclipse IDE for ModusToolbox and choose or create a workspace.
2. From the **File** menu, choose **Import**. Expand **General**, choose **Existing Projects into Workspace**, and then choose **Next**.
3. In **Select Root Directory**, enter:  
`<amazon-freertos>/projects/cypress/<kit-name>/mtb/aws_demos.`
4. The project `aws_demos` should be selected by default.
5. Choose **Finish** to import the project into your workspace.
6. From the Quick Panel, click **Build aws\_demos Application** (Or, use **Project > Build All**). Confirm the project compiles without any errors.

## Run the Amazon FreeRTOS Demo Project

1. Click the project `aws_demos` in the workspace
2. From the Quick Panel, click `aws_demos Program (KitProg3)`. This step will program the board and the demo application starts running after the programming is finished.
3. You can view the status of the running application in the serial terminal. Following figure shows a part of the terminal output.



```
COM5 - Tera Term VT
File Edit Setup Control Window Help
WLAN MAC Address : CC:08:79:24:D8:8B
WLAN Firmware   : w10: Jul 30 2019 01:54:48 version 7.45.98.89 (r719486 CY) FWID 01-81376c4b
WLAN CIM        : API: 12.2 Data: 9.10.39 Compiler: 1.29.4 ClmImport: 1.36.3 Creation: 2019-07-30 01:43:02
WHD VERSION     : v1.30.0-rc3-dirty: v1.30.0-rc3 : GCC 7.2 : 2019-08-27 16:29:32 +0800
1 3518 [Time Svc] Wi-Fi Connected to AP. Creating tasks which use network...
2 3518 [Time Svc] IP Address acquired 192.168.43.207
3 5083 [Time Svc] Write certificate...
4 5623 [Time Svc] Device credential provisioning succeeded.
5 5627 [iot_thermal [INFO [IINIT][lu] SDK successfully initialized.
6 8504 [iot_thermal [INFO [IDEMO][lu] Successfully initialized the demo. Network type for the demo: 1
7 8504 [iot_thermal [INFO [IMQTT][lu] MQTT library successfully initialized.
8 8504 [iot_thermal [INFO [IDEMO][lu] MQTT demo client identifier is cy8cproto-kit (length 13).
9 13409 [iot_thermal [INFO [IMQTT][lu] Establishing new MQTT connection.
10 13411 [iot_thermal [INFO [IMQTT][lu] Anonymous metrics (SDK language, SDK version) will be provided to AWS IoT. Recompile with AWS_IOT_MQTT_ENABLE_METRICS set to 0 to disable.
11 13412 [iot_thermal [INFO [IMQTT][lu] <MQTT connection 0x800b4c8, CONNECT operation 0x800b2d0> Waiting for operation completion.
12 13753 [iot_thermal [INFO [IMQTT][lu] <MQTT connection 0x800b4c8, CONNECT operation 0x800b2d0> Wait complete with result SUCCESS.
13 13754 [iot_thermal [INFO [IMQTT][lu] New MQTT connection 0x800b864 established.
14 13755 [iot_thermal [INFO [IMQTT][lu] <MQTT connection 0x800b4c8> SUBSCRIBE operation scheduled.
15 13755 [iot_thermal [INFO [IMQTT][lu] <MQTT connection 0x800b4c8, SUBSCRIBE operation 0x800b5e0> Waiting for operation completion.
16 14065 [iot_thermal [INFO [IMQTT][lu] <MQTT connection 0x800b4c8, SUBSCRIBE operation 0x800b5e0> Wait complete with result SUCCESS.
17 14065 [iot_thermal [INFO [IDEMO][lu] All demo topic filter subscriptions accepted.
18 14065 [iot_thermal [INFO [IDEMO][lu] Publishing messages 0 to 1.
19 14067 [iot_thermal [INFO [IMQTT][lu] <MQTT connection 0x800b4c8> MQTT PUBLISH operation queued.
20 14069 [iot_thermal [INFO [IDEMO][lu] <MQTT connection 0x800b4c8> MQTT PUBLISH operation queued.
21 14069 [iot_thermal [INFO [IDEMO][lu] Waiting for 2 publishes to be received.
22 14398 [iot_thermal [INFO [IDEMO][lu] MQTT PUBLISH 0 successfully sent.
23 14424 [iot_thermal [INFO [IDEMO][lu] Incoming PUBLISH received:
Subscription topic filter: iotdemo/topic/1
Publish topic name: iotdemo/topic/1
Publish retain flag: 0
Publish qos: 1
Publish msg#0 14424 [iot_thermal [INFO [IMQTT][lu] <MQTT connection 0x800b4c8> MQTT PUBLISH operation queued.
25 14425 [iot_thermal [INFO [IDEMO][lu] Acknowledgment message for PUBLISH 0 will be sent.
26 14680 [iot_thermal [INFO [IDEMO][lu] MQTT PUBLISH 1 successfully sent.
27 14708 [iot_thermal [INFO [IDEMO][lu] Incoming PUBLISH received:
Subscription topic filter: iotdemo/topic/2
Publish topic name: iotdemo/topic/2
Publish retain flag: 0
Publish qos: 1
Publish msg#0 14708 [iot_thermal [INFO [IMQTT][lu] <MQTT connection 0x800b4c8> MQTT PUBLISH operation queued.
29 14708 [iot_thermal [INFO [IDEMO][lu] Acknowledgment message for PUBLISH 1 will be sent.
30 14710 [iot_thermal [INFO [IDEMO][lu] 2 publishes received.
31 14710 [iot_thermal [INFO [IDEMO][lu] Publishing messages 2 to 3.
```

The MQTT demo publishes messages on four different topics (`iotdemo/topic/n`, where  $n=1$  to  $4$ ) and subscribes to all those topics to receive the same messages back. When a message is received, the demo publishes acknowledgement message on the topic `iotdemo/acknowledgements`.

The description of the debug messages printed on the terminal is given below with reference to the serial number of the debug messages. The WICED Host Driver (WHD) driver details are printed first without serial numbering.

- #1 to #4 – Device connects to the configured Access Point (AP) and gets provisioned by connecting to the AWS server using the configured end point and certificates.

- #5 to #13 – MQTT library is initialized and device establishes MQTT connection.
- #14 to #17 – Device subscribes to all the topics to receive the published messages back
- #18 to #30 – Device publishes two messages and waits to receive them back. When each message is received, the device sends acknowledgement message for that received message.

The same cycle of publishing, receiving, and acknowledging continues until all the messages are published with two messages per cycle and until the number of cycles as configured are completed.

## Using CMake with Amazon FreeRTOS

You can alternatively use CMake to build and run the demo application. See [CMake Prerequisites](#) to setup CMake and a native build system.

1. Use the following command to generate build files. Specify the target board through -DBOARD option.

```
cmake -DVENDOR=cypress -DBOARD=CY8CPROTO_062_4343W -
DCOMPILER=arm-gcc -S
<amazon-freertos> -B <build_dir>
```

If you are using Windows, you must specify the native build system using -G option because CMake uses Visual Studio by default. For example:

```
cmake -DVENDOR=cypress -DBOARD=CY8CPROTO_062_4343W -DCOMPILER=arm-
gcc -S
<amazon-freertos> -B <build_dir> -G Ninja
```

If arm-none-eabi-gcc is not in your shell path, you also need to set the AFR\_TOOLCHAIN\_PATH CMake variable. For example:

```
-DAFR_TOOLCHAIN_PATH=/home/user/opt/gcc-arm-none-eabi/bin
```

2. Use the following command to build the project using CMake.

```
CMake --build <build_dir>
```

3. Finally, program the aws\_demos.elf file generated under <build\_dir> using Cypress Programmer.

## Monitoring MQTT Messages on the Cloud

You can use the MQTT client in the AWS IoT console to monitor the messages that your device sends to the AWS Cloud.

To subscribe to the MQTT topic with the AWS IoT MQTT client

1. Sign in to the [AWS IoT console](#).
2. In the navigation pane, choose **Test** to open the MQTT client.
3. In **Subscription topic** enter `iotdemo/#`, select "**Display Payloads as Strings**" and then click **Subscribe to topic**.

A '#' symbol at the end of a topic name acts as wildcard. i.e. Subscription to the topic "iotdemo/#" receives the messages published to any topic that starts with "iotdemo/". You can change the topic name by modifying the macro `IOT_DEMO_MQTT_TOPIC_PREFIX` in

the file `<amazon-freertos>/demos/mqtt/iot_demo_mqtt.c`.

4. Reset the kit to get it to publish MQTT messages which will then be seen on the console test client.

## Known Issues

Problem	Workaround
The BLE Stack contains a defect that can cause memory corruption at application run time.	We have limited the number of BLE links available on our devices to 1. This will be fixed in the next release.

## Running Other Demos

The following demo applications have been tested and verified to work with this release. You can find these demos under `<amazon-freertos>/demos` directory. See the Amazon FreeRTOS [user guide](#) for running these demos.

- Bluetooth Low Energy Demo
- Over-the-Air Updates Demo (only for CY8CPROTO-062-4343W & CY8CKIT-062S2-43012 kits)
- Secure Sockets Echo Client Demo
- AWS IoT Device Shadow Demo

## Debugging

The KitProg on the kit supports debugging over SWD protocol. To debug the Amazon FreeRTOS application, click the `aws_demos` project in the workspace and then click **aws\_demos Debug (KitProg3)** from the **Quick Panel**.

## OTA Updates

PSoC® 62 devices have passed all the required FreeRTOS qualification tests. However, the optional OTA feature implemented in the PSoC 62 firmware library is still pending evaluation. The OTA feature as-implemented currently passes all of the OTA qualification tests except “OTA\_incorrect\_wifi\_password” ([https://github.com/aws/amazon-freertos/blob/master/tools/ota\\_e2e\\_tests/aws\\_ota\\_test/aws\\_ota\\_test\\_case\\_incorrect\\_wifi\\_password.py](https://github.com/aws/amazon-freertos/blob/master/tools/ota_e2e_tests/aws_ota_test/aws_ota_test_case_incorrect_wifi_password.py)).

When a successfully validated OTA image is applied to a device using the PSoC 62 MCU and the device is unable to communicate with AWS IoT Core, the device will not be able to automatically rollback to the original known good image. This may result in the device being unreachable from AWS IoT Core for any further updates. This functionality is still under development by the Cypress team.

For the `CONFIG_OTA_UPDATE_DEMO_ENABLED` `aws_demos` build, the define `OTA_SUPPORT` must be set when building.

- For cmake builds: “`-DOTA_SUPPORT=1`”.
- For MTB builds: edit the Makefile and set `OTA_SUPPORT=1`.

The code signer certificate must also be set in `demos/include/aws_ota_codesigner_certificate.h`

OTA Updates also requires the use of mcuboot. Please see the `vendors/cypress/documents/`

OTA\_README.md file for more information.

If you have further questions or need technical support, please contact the [Cypress Developer Community](#).