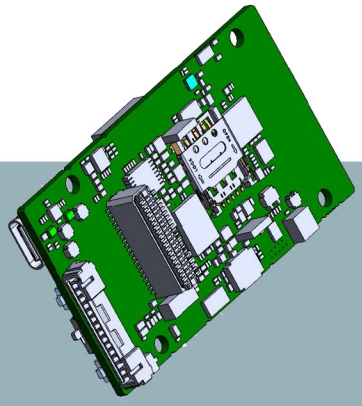
IoT Products



AirPrime SensorHub-AWS

Getting Started Guide for Amazon FreeRTOS

SIERRA WIRELESS®

41114137 Rev. 1 December 2020

Important Notice

Due to the nature of wireless communications, transmission and reception of data can never be guaranteed. Data may be delayed, corrupted (i.e., have errors) or be totally lost. Although significant delays or losses of data are rare when wireless devices such as the Sierra Wireless product are used in a normal manner with a well-constructed network, the Sierra Wireless product should not be used in situations where failure to transmit or receive data could result in damage of any kind to the user or any other party, including but not limited to personal injury, death, or loss of property. Sierra Wireless accepts no responsibility for damages of any kind resulting from delays or errors in data transmitted or received using the Sierra Wireless product, or for failure of the Sierra Wireless product to transmit or receive such data.

Safety and Hazards

Do not operate the Sierra Wireless product in areas where blasting is in progress, where explosive atmospheres may be present, near medical equipment, near life support equipment, or any equipment which may be susceptible to any form of radio interference. In such areas, the Sierra Wireless product **MUST BE POWERED OFF**. The Sierra Wireless product can transmit signals that could interfere with this equipment.

Do not operate the Sierra Wireless product in any aircraft, whether the aircraft is on the ground or in flight. In aircraft, the Sierra Wireless product **MUST BE POWERED OFF**. When operating, the Sierra Wireless product can transmit signals that could interfere with various onboard systems.

Note: Some airlines may permit the use of cellular phones while the aircraft is on the ground and the door is open. Sierra Wireless products may be used at this time.

The driver or operator of any vehicle should not operate the Sierra Wireless product while in control of a vehicle. Doing so will detract from the driver or operator's control and operation of that vehicle. In some states and provinces, operating such communications devices while in control of a vehicle is an offence.

Limitation of Liability

The information in this manual is subject to change without notice and does not represent a commitment on the part of Sierra Wireless. SIERRA WIRELESS AND ITS AFFILIATES SPECIFICALLY DISCLAIM LIABILITY FOR ANY AND ALL DIRECT, INDIRECT, SPECIAL, GENERAL, INCIDENTAL, CONSEQUENTIAL, PUNITIVE OR EXEMPLARY DAMAGES INCLUDING, BUT NOT LIMITED TO, LOSS OF PROFITS OR REVENUE OR ANTICIPATED PROFITS OR REVENUE ARISING OUT OF THE USE OR INABILITY TO USE ANY SIERRA WIRELESS PRODUCT, EVEN IF SIERRA WIRELESS AND/OR ITS AFFILIATES HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES OR THEY ARE FORESEEABLE OR FOR CLAIMS BY ANY THIRD PARTY.

Notwithstanding the foregoing, in no event shall Sierra Wireless and/or its affiliates aggregate liability arising under or in connection with the Sierra Wireless product, regardless of the number of events, occurrences, or claims giving rise to liability, be in excess of the price paid by the purchaser for the Sierra Wireless product.

Patents

This product may contain technology developed by or for Sierra Wireless Inc. This product includes technology licensed from QUALCOMM®. This product is manufactured or sold by Sierra Wireless Inc. or its affiliates under one or more patents licensed from MMP Portfolio Licensing.

Copyright

Trademarks

Sierra Wireless®, AirPrime®, AirLink®, AirVantage® and the Sierra Wireless logo are registered trademarks of Sierra Wireless.

Windows® and Windows Vista® are registered trademarks of Microsoft Corporation.

Macintosh® and Mac OS X® are registered trademarks of Apple Inc., registered in the U.S. and other countries.

QUALCOMM® is a registered trademark of QUALCOMM Incorporated. Used under license.

Other trademarks are the property of their respective owners.

Contact Information

Sales information and technical support, including warranty and returns	Web: sierrawireless.com/company/contact-us/ Global toll-free number: 1-877-687-7795 6:00 am to 5:00 pm PST
Corporate and product information	Web: sierrawireless.com

Revision History

Revision number	Release date	Changes
1	02 Dec 2020	Created

Contents

1: Overview

This document describes how to set up Linux systems (Ubuntu and Debian) to begin using AirPrime® SensorHub with Amazon FreeRTOS.

SensorHub is a reusable horizontal platform designed to anticipate and address a wide range of possible IoT use cases.

Key aspects of the SensorHub are:

- Single hardware design—A comprehensive suite of built-in features, including multiple sensors, and wired and wireless connectivity, enables customers to build for different market segments.
- Modular software architecture—A robust, versatile architecture enables developers to branch software for multiple product variants and SKUs.

2: Hardware Description

This chapter indicates requirements for using SensorHub with Amazon FreeRTOS, and describes the SensorHub's key components.

2.1 Hardware Requirements to Run FreeRTOS Demo

2.1.1 Standard Kit Contents

The AirPrime SensorHub kit includes:

- SensorHub with HL7802 cellular module and BX3105 Wi-Fi/BT module installed
- 2× micro-USB cables
- 1× USB-C cable
- 1× Wide-band flex antenna (U.FL)

2.2 Product Photos and Key Components

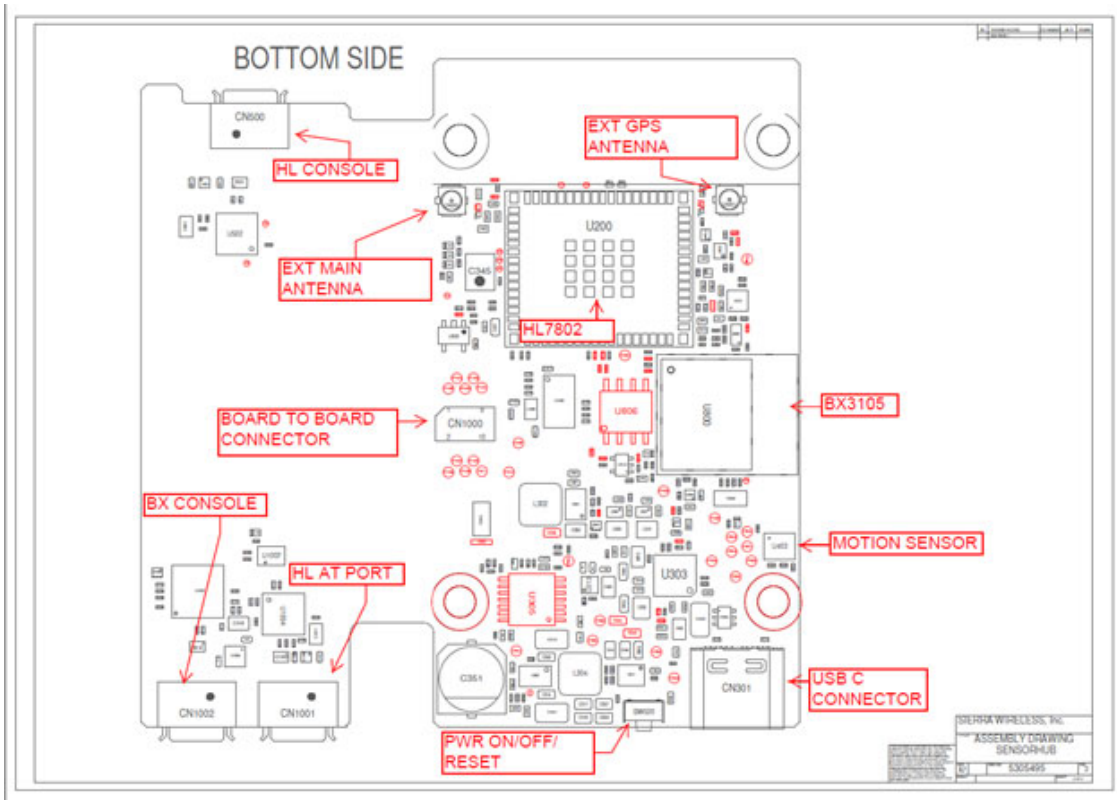


Figure 2-1: SensorHub—Assembly Drawing (Top Side)

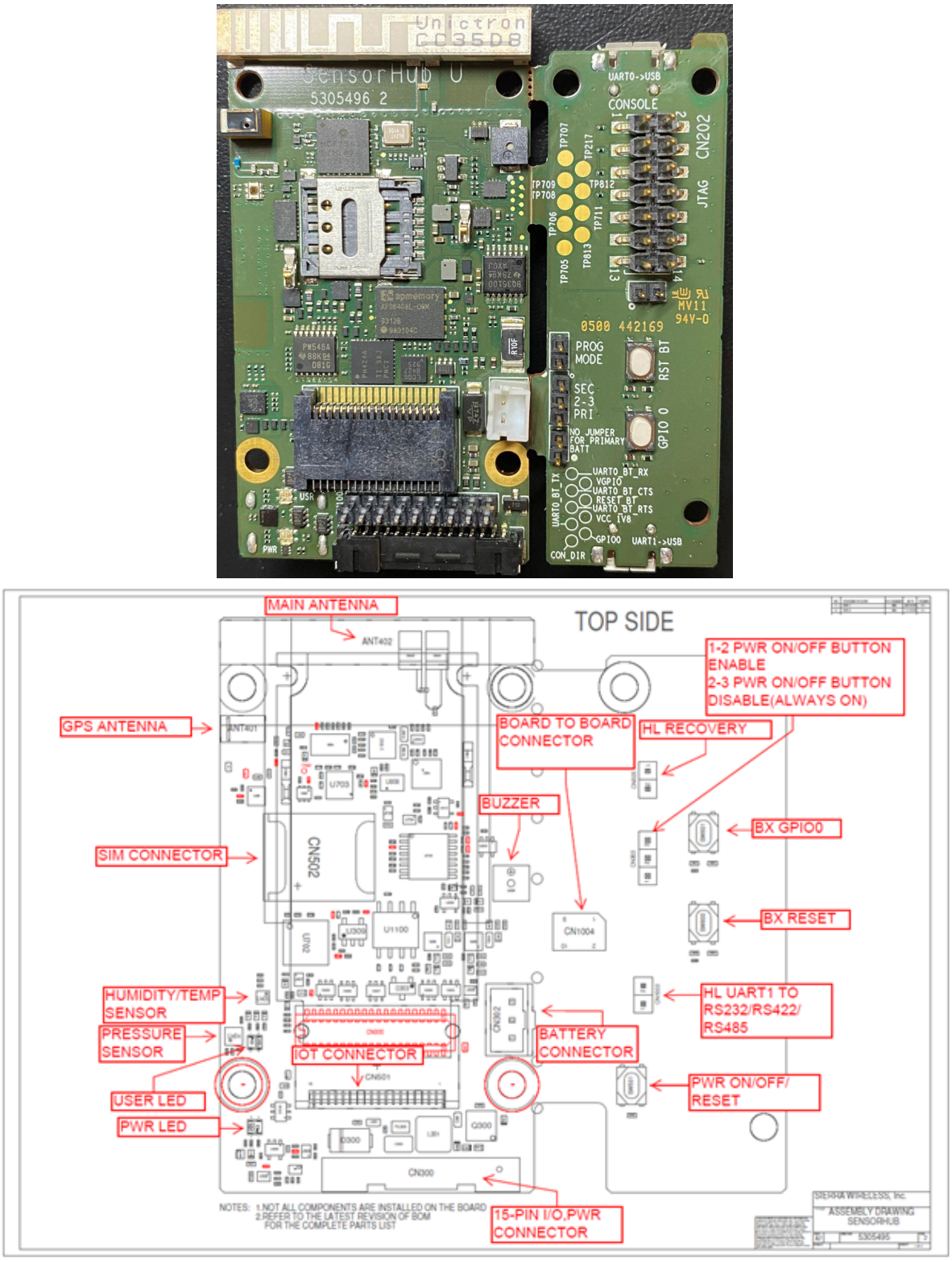


Figure 2-2: SensorHub—Assembly Drawing (Bottom Side)

Table 2-1: SensorHub Integrated Components

SKU	
Description	SensorHub Amazon SKU
Battery	Rechargeable (18650, 3400mah)
Size—PCB	35 mm × 64 mm
Modules	
Radio Module	HL7802
Wi-Fi/BT	BX3105
RF	
Cellular technology	LTE-M1, 2G
GPS—Built into cellular module	Yes
Wi-Fi technology	Yes
Bluetooth LE (5.0)	Yes
Radios/Antennas	
Cellular	Internal
GPS	Internal
Bluetooth LE/Wi-Fi	Internal
Storage	
On-board	Yes (128 MB)
SD card	IoT card
Sensors	
Accelerometer	Yes
Gyroscope	Yes
Pressure	Yes
Temperature	Yes
Humidity	Yes
Light	Yes
Location	GNSS
External Host Interface (not including B2B or IoT card)	
USB	Yes
CAN	Yes

Connectors	
USB Connectors—USB-C	1
USB Connectors—micro-USB	3
IoT Connector (8 mm tall module)	Populated for future use
Molex 15-pin Nano Fit	Partial
U.FL—on board	Population option
Battery (JST 3-pin)	Yes
SIM 4FF	1
Jumpers	
Two-pin	2
Three-pin	1
UX interfaces	
LEDs (Tri-color)	1
System reset/Power Switch	2
BX reset button	1
BX programming button	1
Power	
Battery	3400 maH secondary
DC power	7V–36V
Charging	Yes
Current Monitoring (Gauge)	Yes
Operating States/Environmental	
Operating voltage	DC: 7V–36V
	Secondary: 3.4V–4.3V
Operating temperature	Class B—-30°C to +75°C
Operating humidity	95% relative humidity over temperature range +20°C to +60°C
Vibration and Shock	Vibration spec: MIL-STD-810G, Method 514.6C
	Mechanical shock spec: MIL-STD-810G, Method 516.6
	Procedure I (Functional Shock)

2.3 Datasheets

Datasheets are available for the SensorHub's cellular and Wi-Fi/BT modules:

- HL7802 Datasheet—https://source.sierrawireless.com/devices/hl-series/hl7802/
- BX3105 Datasheet—https://source.sierrawireless.com/devices/wifi-bluetooth-modules/bx3105

2.4 Additional Hardware References

Additional hardware and related reference materials are available at source.sierrawireless.com/devices/iot-products/sensorhub.

3: Set Up Your Development Environment

3.1 Supported IDEs

The CMake build system is required to build the FreeRTOS demo and test applications for this device. FreeRTOS supports CMake versions 3.13 and later.

You can download the latest version of CMake from CMake.org. Both source and binary distributions are available.

1. Check your current version:

```
$ cmake --version
```

2. If your version is older than 3.13 (i.e. 3.12 or less), uninstall cmake:

```
$ sudo apt remove cmake
```

3. Download the latest cmake package:

```
$ wget https://github.com/Kitware/CMake/releases/download/v3.18.4/cmake-3.18.4-Linux-x86_64.sh
$ sudo cp ./cmake-3.18.4-Linux-x86_64.sh /opt
```

Note: v3.18.4 was the latest version at time of publication. If a newer version is available, modify the commands above accordingly.

4. Make the package executable:
5. Install cmake:

```
$ sudo bash /opt/cmake-3.18.4-Linux-x86_64.sh
```

The script installs to /opt/cmake-3.18.4-Linux-x86_64.

6. Create a symbolic link to the script to get the cmake command:

```
$ sudo ln -s /opt/cmake-3.*your_version*/bin/* /usr/local/bin
```

7. Confirm that cmake is available:

```
$ cmake --version
```

3.2 Toolchains

ESP-IDF uses CMake to build software.

Several prerequisite packages must be installed and the ESP32 toolchain for Linux must be added to your system:

1. For Linux (Ubuntu and Debian) systems, install the prerequisite packages for ESP-IDF:

```
$ sudo apt-get install git wget libncurses-dev flex bison \
  gperf python python-pip python-setuptools python-serial \
  python-cryptography python-future python-pyparsing \

$ cmake ninja-build ccache
```

2. Download the ESP2 toolchain for Linux from the Espressif website:
   - For 64-bit Linux:

     https://dl.espressif.com/dl/xtensa-esp32-elf-linux64-1.22.0-80-g6c4433a-5.2.0.tar.gz
   - For 32-bit Linux:

     https://dl.espressif.com/dl/xtensa-esp32-elf-linux32-1.22.0-80-g6c4433a-5.2.0.tar.gz
3. Extract the downloaded toolchain file to the ~/esp:
   - For 64-bit Linux:

```
$ mkdir -p ~/esp
$ cd ~/esp
$ tar -xzf ~/Downloads/xtensa-esp32-elf-linux64-1.22.0-80-g6c4433a-5.2.0.tar.gz
```

   - For 32-bit Linux:

```
$ mkdir -p ~/esp
$ cd ~/esp
$ tar -xzf ~/Downloads/xtensa-esp32-elf-linux32-1.22.0-80-g6c4433a-5.2.0.tar.gz
```

   The toolchain extracts into the ~/esp/xtensa-esp32-elf/ directory.
4. To use the toolchain, you must update your PATH environment variable in the ~/.profile file. To make xtensa-esp32-elf available for all terminal sessions, use either of the following methods:
   - Add the following line to your ~/.profile file:

```
$ export PATH="$HOME/esp/xtensa-esp32-elf/bin:$PATH"
```

   - Alternatively, create an alias for the above command. This way you can get the toolchain only when you need it. To do this, add the following line to your ~/.profile file:

```
$ alias get_esp32='export \
   PATH="$HOME/esp/xtensa-esp32-elf/bin:$PATH"'
```

     Then when you need the toolchain you can type `get_esp32` on the command line and the toolchain will be added to your PATH.

Note: If you have /bin/bash set as the login shell, and both .bash_profile and .profile exist, then update .bash_profile instead of .profile.

5. Log off and log in back to make the .profile changes effective.
6. Verify that PATH is correctly set:
   a. Run the following command:

```
$ printenv PATH
```

   b. Make sure the beginning of the displayed response contains the correct path:

```
$ printenv PATH

/home/user-name/esp/xtensa-esp32-elf/bin:/home/user-name/bin:/ho
me/user-name/.local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin
:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
```

     Instead of /home/user-name, there should be a home path specific to your installation.

3.3 Establishing a Serial Connection

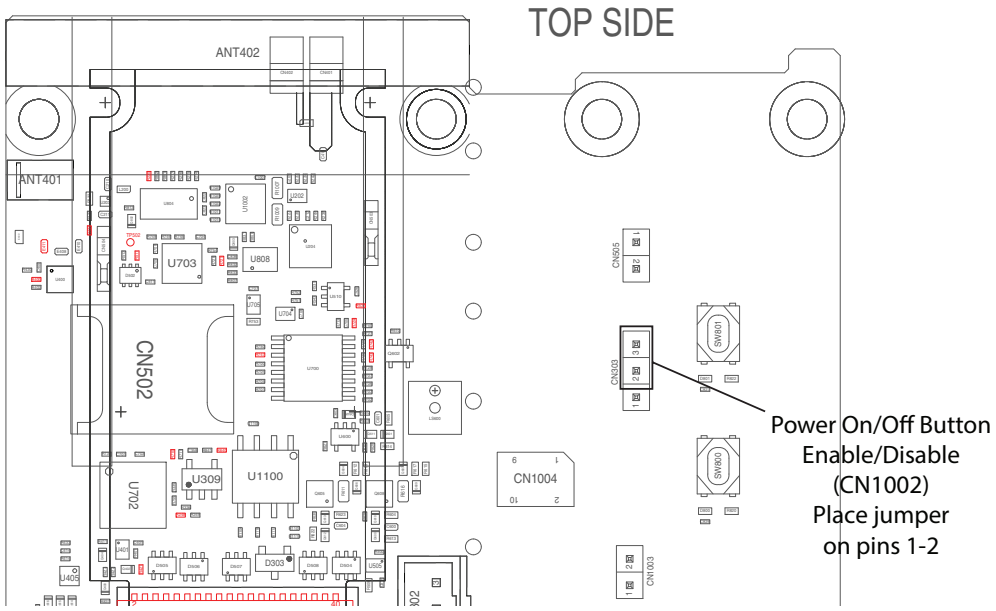
For instructions for establishing a serial connection, refer to:

https://docs.espressif.com/projects/esp-idf/en/v4.1/get-started/establish-serial-connection.html

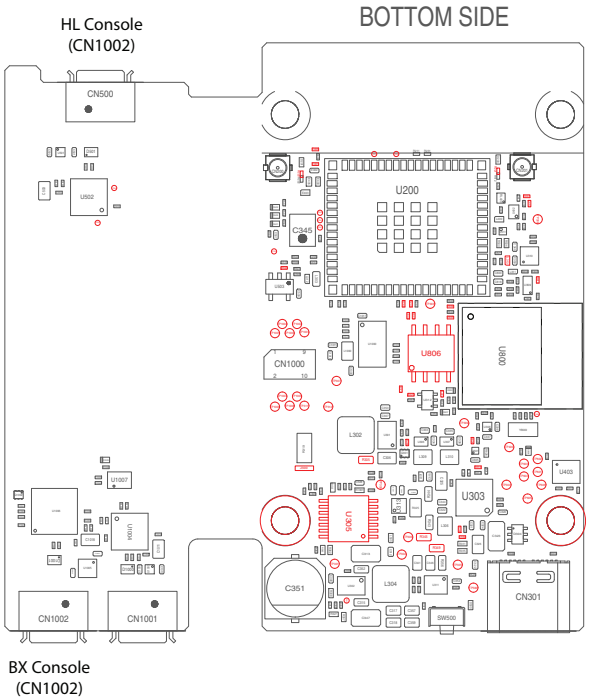
4: Set Up Your Hardware

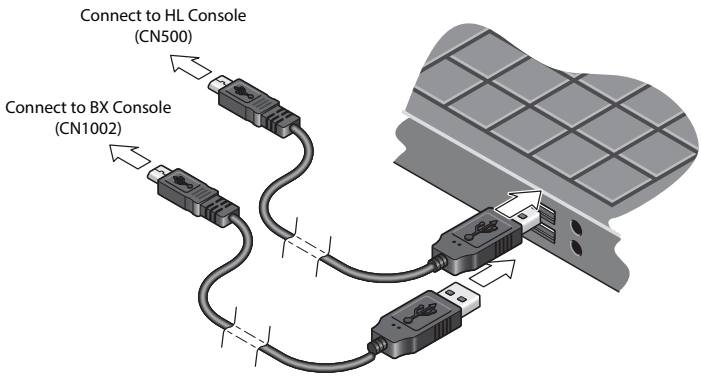
Set up your SensorHub as follows:

1. Enable the Power On/Off Button—On the top side of the device, place a jumper across pins 1–2 of CN303 (a 3-pin header).



2. Connect the device's HL (cellular module) console (CN1001) and BX (Wi-Fi/BT module) console (CN1002) micro-USB connectors to your computer.





5: Set Up Your AWS Account and Permissions

An AWS account is required to receive data from the SensorHub in the cloud.

To create an AWS account:

1. See Create and Activate an AWS Account.

To add an IAM user to your AWS account:

1. Use the instructions at https://docs.aws.amazon.com/iot/latest/developer-guide/setting-up.html. The relevant section is “Create a user and grant permissions”.

To grant your IAM user account access to AWS IoT and FreeRTOS, attach the following IAM policies to your IAM user account—AmazonFreeRTOSFullAccess and AWSIoTFullAccess:

To attach the AmazonFreeRTOSFullAccess policy to your IAM user:

1. Attach the first policy (AmazonFreeRTOSFullAccess) to your account:
   a. Browse to the IAM console, and from the navigation pane, choose **Users**.
   b. Enter your user name in the search text box, and then choose it from the list.
   c. Choose **Add permissions**.
   d. Choose **Attach existing policies directly**.
   e. In the search box, enter AmazonFreeRTOSFullAccess, choose it from the list, and then choose **Next: Review**.
   f. Choose **Add permissions**.
2. Attach the second policy (AWSIoTFullAccess) to your account:
   a. Browse to the IAM console, and from the navigation pane, choose **Users**.
   b. Enter your user name in the search text box, and then choose it from the list.
   c. Choose **Add permissions**.
   d. Choose **Attach existing policies directly**.
   e. In the search box, enter AWSIoTFullAccess, choose it from the list, and then choose **Next: Review**.
   f. Choose **Add permissions**.

For more information about IAM and user accounts, see the IAM User Guide.

For more information about policies, see IAM Permissions and Policies.

6: Provision SensorHub with AWS IoT

Your SensorHub must be registered with AWS IoT to communicate with the AWS Cloud. To register your device with AWS IoT, you need the following:

- An AWS IoT policy

  The AWS IoT policy grants your device permissions to access AWS IoT resources. It is stored on the AWS Cloud.
- An AWS IoT thing

  An AWS IoT thing allows you to manage your devices in AWS IoT. It is stored on the AWS Cloud.
- A private key and X.509 certificate

  The private key and certificate allow your device to authenticate with AWS IoT.

To register your device, follow the procedures below.

6.1 Create an AWS IoT Policy

To create an AWS IoT policy:

1. Get your AWS account ID:
   a. Browse to the AWS Management Console.
   b. Locate and expand the menu beneath your account name in the upper-right corner, and choose **My Account**.

      Your account ID is displayed under **Account Settings**.
2. Get your AWS region:
   a. Browse to the AWS IoT console.
   b. In the navigation pane, choose **Settings**.

      Your AWS IoT endpoint is displayed under Custom Endpoint. It should look like 1234567890123-ats.iot.us-east-1.amazonaws.com.

      In this example, your region would be us-east-1.
3. In the navigation pane, choose **Secure**, choose **Policies**, and then choose **Create**.
4. Enter a name to identify your policy.

5. In the Add statements section, choose **Advanced mode**. Copy and paste the following JSON into the policy editor window. Replace `aws-region` and `account-id` with your AWS Region and account ID.

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Action": "iot:Connect",
        "Resource":"arn:aws:iot:aws-region:account-id:*"
    },
    {
        "Effect": "Allow",
        "Action": "iot:Publish",
        "Resource": "arn:aws:iot:aws-region:account-id:*"
    },
    {
         "Effect": "Allow",
         "Action": "iot:Subscribe",
         "Resource": "arn:aws:iot:aws-region:account-id:*"
    },
    {
         "Effect": "Allow",
         "Action": "iot:Receive",
         "Resource": "arn:aws:iot:aws-region:account-id:*"
    }
    ]
}
```

This policy grants the following permissions:

- iot:Connect

  Grants your device the permission to connect to the AWS IoT message broker with any client ID.
- iot:Publish

  Grants your device the permission to publish an MQTT message on any MQTT topic.
- iot:Subscribe

  Grants your device the permission to subscribe to any MQTT topic filter.
- iot:Receive

  Grants your device the permission to receive messages from the AWS IoT message broker on any MQTT topic.

Note: All devices in your fleet must have credentials with privileges that authorize intended actions only, which include (but are not limited to) AWS IoT MQTT actions such as publishing messages or subscribing to topics with specific scope and context. The specific permission policies can vary for your use cases. Identify the permission policies that best meet your business and security requirements.

For sample policies, refer to https://docs.aws.amazon.com/iot/latest/developer-guide/example-iot-policies.html. Also refer to https://docs.aws.amazon.com/iot/latest/developerguide/security-best-practices.html.

6. Choose **Create**.

6.2 Create an IoT Thing, Private Key, and Certificate

To create an IoT thing, private key, and certificate for your device:

1. Browse to the AWS IoT console.
2. In the navigation pane, choose **Manage**, and then choose **Things**.
3. If you do not have any IoT things registered in your account, the "You don't have any things yet" page is displayed. If you see this page, choose **Register a thing**. Otherwise, choose **Create**.
4. On the "Creating AWS IoT things" page, choose **Create a single thing**.
5. On the "Add your device to the thing registry" page, enter a name for your thing, and then choose **Next**.
6. On the "Add a certificate for your thing" page, under One-click certificate creation, choose **Create certificate**.
7. Download your private key and certificate by choosing the **Download** links for each.
8. Choose **Activate** to activate your certificate. Certificates must be activated prior to use.
9. Choose **Attach a policy** to attach a policy to your certificate that grants your device access to AWS IoT operations.
10. Choose the policy you just created and choose **Register thing**.

7: Download FreeRTOS

The amazon-freertos repository must be downloaded with the feature/cellular branch. Use the following command:

```
$ git clone https://github.com/aws/amazon-freertos.git \
   --recurse-submodules -b feature/cellular
```

Please note that the feature/cellular branch (like all feature/*xxx* branches) is a temporary branch for development, which will be merged to the master branch.

After downloading amazon-freertos, Python dependencies must be installed. Use the following command and replace `/DOWNLOAD_PATH` with the path to the cloned repository in your filesystem:

```
$ python -m pip install --user -r /DOWNLOAD_PATH/amazon-freertos/vendors/espressif/esp-idf/requirements.txt
```

8: Configure Free RTOS

Some configuration files in your FreeRTOS directory must be edited before you can compile and run any demos on your SensorHub.

8.1 Configure Your AWS IoT Endpoint

You must provide FreeRTOS with your AWS IoT endpoint so the application running on your device can send requests to the correct endpoint:

1. Browse to the AWS IoT console.
2. In the navigation pane, choose **Settings**.

   Your AWS IoT endpoint is displayed in Endpoint. It should look like 1234567890123-ats.iot.us-east-1.amazonaws.com. Make a note of this endpoint.
3. In the navigation pane, choose **Manage**, and then choose **Things**.

   Your device should have an AWS IoT thing name. Make a note of this name.
4. Open FREERTOS_BASEDIR/demos/include/aws_clientcredential.h.
5. Specify values for the following constants:

```
#define clientcredentialMQTT_BROKER_ENDPOINT "Your AWS IoT endpoint";
#define clientcredentialIOT_THING_NAME "Your device's AWS IoT thing name"
```

8.2 Format Your AWS IoT Credentials

FreeRTOS needs the AWS IoT certificate and private keys associated with your registered thing and its permissions policies to successfully communicate with AWS IoT on behalf of your device.

Note: To configure your AWS IoT credentials, you need the private key and certificate that you downloaded from the AWS IoT console when you registered your device. After you have registered your device as an AWS IoT thing, you can retrieve device certificates from the AWS IoT console, but you cannot retrieve private keys.

FreeRTOS is a C language project, and the certificate and private key must be specially formatted to be added to the project.

1. In a browser window, open

   FREERTOS_BASEDIR/tools/certificate_configuration/CertificateConfigurator.html
2. Under Certificate PEM file, choose the ID-certificate.pem.crt that you downloaded from the AWS IoT console.
3. Under Private Key PEM file, choose the ID-private.pem.key that you downloaded from the AWS IoT console.

4. Choose **Generate** and save aws_clientcredential_keys.h, and then save the file in FREERTOS_BASEDIR/demos/include. This overwrites the existing file in the directory.

Note: The certificate and private key are hard-coded for demonstration purposes only. Production-level applications should store these files in a secure location.

8.3 Set Up for Cellular Communication

The SensorHub comes supplied with a Sierra Wireless SIM.

To set up the SensorHub's cellular connectivity:

1. Open amazon-freertos\demos\include\aws_cellular_demo.h.
2. In the following constants, enter the Sierra Wireless SIM's API (na.lp.swir) and leave the DNS server blank. (The DNS server is obtained automatically by the Sierra SIM AT connection time.)

```
#define configCELLULAR_APN "na.lp.swir";
#define configCELLULAR_DNS_SERVER ""
```

9: Build the FreeRTOS demo

To build the FreeRTOS demo:

1. Go to the amazon-freertos folder.
2. Set the CMake configuration for the demo. Use the following command:

```
$ cmake -DVENDOR=sierra -DBOARD=sensorhub -DCOMPILER=xtensa-esp32 \
 -DSECURE_SOCKETS_CELLULAR=1 -DBOARD_HAS_CELLULAR=1 -GNinja -S . \
 -B build-demos
```

3. Build the demo using CMake. Use the following command:

```
$ cmake --build build-demos --clean-first
```

10: Run the FreeRTOS Demo

10.1 Run FreeRTOS

To run the FreeRTOS demo:

1. Go to the amazon-freertos folder.
2. Flash the demo app to the Host MCU (BX3105)—Use the following commands (replace ttyUSBx with the BX console's port):

```
$ python ./vendors/espressif/esp-idf/tools/idf.py erase_flash \
 -p /dev/ttyUSBx -B build-demos
$ python ./vendors/espressif/esp-idf/tools/idf.py flash -p \
 /dev/ttyUSBx -B build-demos
$ python ./vendors/espressif/esp-idf/tools/idf.py monitor -p \
 /dev/ttyUSBx  -B build-demos
```

   The last command starts the ESP-IDF monitor.

3. The demo app will start after a system reset—press the system reset button to run the demo.

   The demo app will send MQTT messages to AWS IOT cloud. To verify that this is working, see Monitor MQTT Messages on the Cloud, below.

10.2 Monitor 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/#`, and then choose **Subscribe to topic**.

   The console will display `"Hello World…"`, indicating the system is working.

11: Debugging

The FreeRTOS demo app will show its log messages on the BX console port.

To enable debug logs:

1. Open amazon-freertos\vendors\aws\modules\3gpp\ports\cellular\include\private\cellular_internal.h
2. Set LIBRARY_LOG_LEVEL to IOT_LOG_DEBUG:

```
#define LIBRARY_LOG_LEVEL IOT_LOG_DEBUG
```

3. To enable logging:
   a. Open amazon-freertos\vendors\sierra\boards\sensorhub\ports\comm_if\comm_if_sierra.c
   b. Set COMM_IF_DEBUG to 1:

```
#define COMM_IF_DEBUG 1
```