Getting Started with the Andes Corvette-F1 N25

This tutorial provides instructions for getting started with the Andes Corvette-F1 N25 Development Board. If you do not have the Andes Corvette-F1 N25 board, visit the AWS Partner Device Catalog to purchase one from our partner.

Before you begin, you must configure AWS IoT and your downloaded Amazon FreeRTOS to connect your device to the AWS Cloud. See First Steps (p. 59) for instructions. In this tutorial, the path to the downloaded Amazon FreeRTOS package is referred to as <amazon-freertos>.

Overview

This tutorial contains instructions for the following steps:

1. Installing software on the host machine for developing and debugging embedded applications on your Andes Corvette-F1 N25 board.
2. Cross compiling an Amazon FreeRTOS demo application to a binary image.
3. Flashing the application binary image to your board, and then running the application.
4. Monitoring and debugging the application that runs on your board thorugh a serial connection.

Set Up the Andes Corvette-F1 N25 Hardware

- Connect the CON2 (UART) port on the Corvette-F1 N25 Development Board to a USB port on your computer using a micro USB cable.

Set Up Your Development Environment

All tools necessary for developing Amazon FreeRTOS demo applications on the Corvette-F1 N25 Development Board are available at the following GitHub repository. Download the repository and unzip it.

```
https://github.com/andestech/aws_development_tools.git
```

Set Up Cygwin (for Windows only)

On Windows, you are required to use Cygwin to compile, burn, or execute Amazon FreeRTOS applications on the Corvette-F1 N25 Development Board. If you are a Windows user, obtain the compressed Cygwin package from the following path first

and extract it.

```
<path_to_the_aws_development_tools folder>\windows\Cygwin\Cygwin.tar.gz
```

You will find the Cygwin batch file (Cygwin.bat) in the following path.

```
<path_to_the_Cygwin_folder>\Cygwin.bat
```

Execute the batch file to launch the Cygwin terminal which allows you to perform any operation for developing Amazon FreeRTOS applications on Windows.

Set Up the Toolchain

On Linux, you are required to install some necessary programs if your operating system is 64-bit. To install these programs, you just need to execute the shell script in the following path:

```
<path_to_the_aws_development_tools_folder>/linux/toolchain/
install_linux_package.sh
```

The Amazon FreeRTOS port for the Corvette-F1 N25 Development Board is configured to use nds32le-elf-newlib-v5, a custom GNU toolchain for RISC-V, by default. The toolchain is also in the “aws_development_tools” repository you downloaded earlier from GitHub. Find the compressed toolchain from the following path and unzip it. On Windows, you need to unzip toolchain in the way of command line on Cygwin.

```
<path_to_the_aws_development_tools_folder>/[linux|windows]/toolchain/
nds32le-elf-newlib-v5.txz
```

Include the following path to the nds32le-elf-newlib-v5 toolchain binaries in your PATH variable. This is to ensure that the correct toolchain can be invoked for building Amazon FreeRTOS applications later.

```
<path_to_the_nds32le-elf-newlib-v5_toolchain_folder>/bin/
```

With Andes ICE management software “ICEman” that interfaces with the JTAG connector on the Corvette-F1 N25 board, the nds32le-elf-newlib-v5 toolchain can be used with the GNU Debugger (GDB) for debugging.

Set Up ICEman

For debugging Amazon FreeRTOS applications or burning them to a flash, you will

need to execute the Andes ICE management software, ICEman, on the host computer connected with Corvette-F1 N25 Development Board.

The ICEman program can be executed directly in a Linux system. If you are using the Windows operating system, make sure to install the ICEman driver from the following path first:

```
<path_to_the_aws_development_tools_folder>/windows/ICEman driver/
zadig_20171002_win7/zadig_20171002_win7.exe
```

For details about the installation procedure, refer to the Driver installation guide in the following path:

```
<path_to_the_aws_development_tools_folder>/windows/ICEman driver/
Driver_install_guide.pptx
```

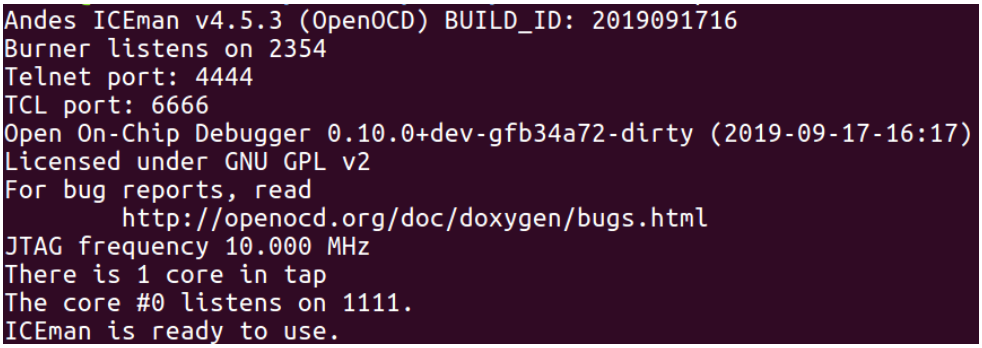
To execute the ICEman program, change the current working directory to the following folder:

```
<path_to_the_aws_development_tools_folder>/[linux|windows]/ICEman
```

And use the following command to execute ICEman:

```
$ sudo ./ICEman -Z v5
```

You will see the following message when ICEman is launched:



Note:

Be sure to keep ICEman running when you are debugging or burning a flash.

Install CMake

The CMake build system is required to build the Amazon FreeRTOS demo and test applications for the Corvette-F1 N25 Development Board. Amazon FreeRTOS supports CMake versions 3.13 and later.

If you are using a Linux system, download the latest version of CMake, including its

source and binaries, from CMake.org.

If you are using the Windows operating system, CMake is ready-to-use in the Cygwin you downloaded and set up.

For more details about using CMake with Amazon FreeRTOS, see Using CMake with Amazon FreeRTOS (p. 65).

Establish a Serial Connection between your host computer and your board

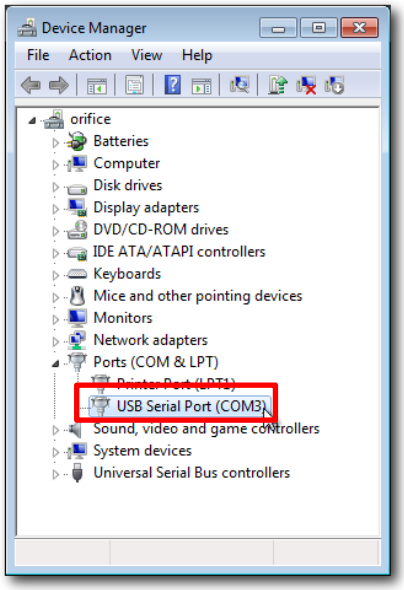
1. Attach one end of a USB cable to your host computer, and the other end to your Corvette-F1 N25 board. Your host computer should be able to detect the board right away.

   On Linux, issue the dmesg command in a command line console and verify a successful connection from a message like below:

```
[59524.253265] usb 1-2: New USB device found, idVendor=0403, idProduct=6010
[59524.253269] usb 1-2: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[59524.253271] usb 1-2: Product: Dual RS232-HS
[59524.253273] usb 1-2: Manufacturer: FTDI
[59524.261430] ftdi_sio 1-2:1.0: FTDI USB Serial Device converter detected
[59524.261541] usb 1-2: Detected FT2232H
[59524.269350] usb 1-2: FTDI USB Serial Device converter now attached to ttyUSB0
[59524.273393] ftdi_sio 1-2:1.1: FTDI USB Serial Device converter detected
[59524.273496] usb 1-2: Detected FT2232H
[59524.281321] usb 1-2: FTDI USB Serial Device converter now attached to ttyUSB1
```

   Note:
   The Corvette-F1 N25 board exposes two USB interfaces to the host computer. One (ttyUSB0) is to the MCU's JTAG functionality and the other (ttyUSB1) is to the MCU's physical UARTx port.

   On Windows, open "Device Manager" and verify a successful connection from the listing of a new device along with a COM port number under the node "Ports (COM and LPT)". The figure below shows that a new entry "USB Serial Port (COM3)" is listed in the Device Manager after the USB cable is plugged, indicating a device has been attached to COM3 of the host computer.



2. Use the following settings to start a serial connection to either USB interface (i.e., ttyUSBx or COMx for Windows):

Terminal Setting	Value
BAUD rate	115200
Data	8 bit
Parity	None
Stop	1 bit
Flow control	None

   For example, proceed with the following steps if you are using PuTTY:

   a. On Linux, issue the following command to launch the PuTTY program:

      ```
      $ sudo ./putty
      ```

      On Windows, double-click the desktop shortcut  to start PuTTY.

   b. Set the **Connection type** to “Serial”.

   c. Set the **Serial line** to “/dev/ttyUSBx” for Linux or “COMx” for Windows.

   d. Set the **Speed** to 115200.

   e. Click on the **Open** button.

For more information about installing a terminal emulator to set up a serial connection, see Installing a Terminal Emulator (p. 65).

Build, Flash, and Run a Amazon FreeRTOS Demo Project

Set Wi-Fi and device information

Edit the following file to specify your Wi-Fi SSID and password for your Amazon

FreeRTOS demo project. This is to enable the project to connect to the internet and use MQTT to connect to the AWS cloud.

<amazon-freertos>/demos/include/aws_clientcredential.h

Next, edit the following file to provide your demo project with the device information, which includes the certificate, MQTT endpoint and device name.

<amazon-freertos>/demos/include/aws_clientcredential_keys.h

In the case of building the Amazon FreeRTOS test project, edit the following two files to configure the Wi-Fi settings and device information.

<amazon-freertos>/tests/include/aws_clientcredential.h

<amazon-freertos>/tests/include/aws_clientcredential_keys.h

Generate Build Files for Your Demo Project and Build the Demo

From the root directory of your downloaded Amazon FreeRTOS package, issue the CMake command as follows to generate build files for your demo project:

```
$ cmake -DVENDOR=andes -DBOARD=corvette_f1_n25 -DAFR_IS_TESTING=0 -DCOMPILER=riscv32-elf -B
build
```

The output message will be as the following:

```
====================Configuration for Amazon FreeRTOS====================
  Version:                  201908.00
  Git version:              github

Target microcontroller:
  vendor:                   Andes
  board:                    Corvette-F1 N25
  description:              Development kit for Corvette-F1 N25
  family:                   AE250
  data ram size:            200KB
  program memory size:      1MB

Host platform:
  OS:                       Linux-4.15.0-65-generic
  Toolchain:                riscv32-elf
  Toolchain path:           /home/kevin/nds32le-elf-newlib-v5
  CMake generator:          Unix Makefiles

Amazon FreeRTOS modules:
  Modules to build:         common, crypto, defender, dev_mode_key_provisioning,
                            freertos_plus_tcp, greengrass, https, kernel, mqtt, pkcs11,
                            pkcs11_implementation, platform, secure_sockets, serializer,
                            shadow, tls, wifi
  Enabled by user:          defender, greengrass, https, mqtt, pkcs11, pkcs11_
                            implementation, platform, secure_sockets, shadow, wifi
  Enabled by dependency:    common, crypto, demo_base, dev_mode_key_provisioning,
                            freertos, freertos_plus_tcp, kernel, pkcs11_mbedtls,
                            secure_sockets_freertos_plus_tcp, serializer, tls, utils
  3rdparty dependencies:    http-parser, jsmn, mbedtls, pkcs11, tinycbor
  Available demos:          demo_defender, demo_greengrass_connectivity, demo_https,
                            demo_mqtt, demo_shadow, demo_tcp
  Available tests:
========================================================================
```

Issue the following commands to build the binary image of your demo application:

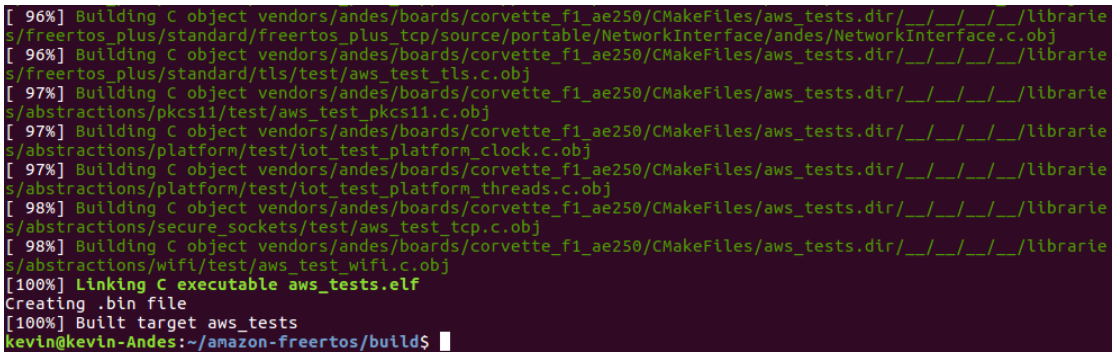
```
$ cd build
```

```
$ make
```

The binary will be generated in the following path:

```
<amazon-freertos>/vendors/andes/boards/corvette_f1_n25/aws_demos/aws_demos.bin
```

The output messages for the build will look like the following:



In the case of the Amazon FreeRTOS test project, use the following commands to generate build files and build the application:

```
$ cmake -DVENDOR=andes -DBOARD=corvette_f1_n25 -DAFR_IS_TESTING=1 -DCOMPILER=riscv32-elf -B build
```

```
$ cd build
```

```
$ make
```

The binary image of the test project will be generated in the following path:

```
<amazon-freertos>/vendors/andes/boards/corvette_f1_n25/aws_tests/aws_tests.bin
```

Note:
Be sure to use the CMake command to generate build files whenever you switch between the aws_demos project and the aws_tests project.

Burn the Application to Flash and Run

Find the script files (i.e., target_burn_linux.sh for Linux and target_burn_windows.sh for Windows) for programming Amazon FreeRTOS demo application to the flash memory on the Corvette-F1 N25 board in the following path:

```
<amazon-freertos>/vendors/andes/tools
```

On Linux, issue the following command to flash your demo application to the board:

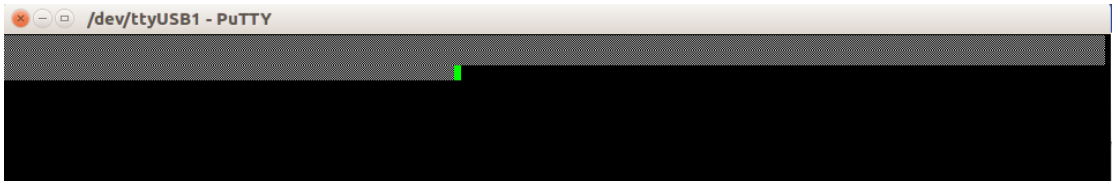
```
bash target_burn_linux.sh <path_to_the_program_binary >
```

On Windows, issue the following command in the Cygwin terminal to program your demo to the flash:

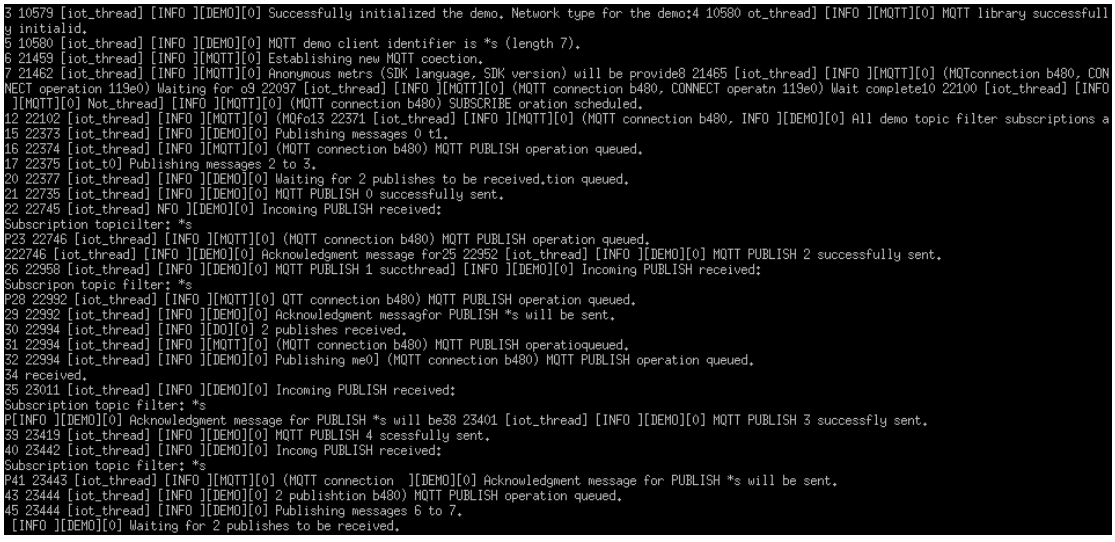
```
bash target_burn_windows.sh <path_to_the_program_binary>
```

Note:
Make sure the ICEman program is launched before you perform flash programming.

You will see the following message in the serial console during the programming process.



The demo application will be executed automatically right after it is burned to the board. You will see output messages like below in your serial console:



Debug the Application

1. Execute the ICEman program.
2. Open a console and issue the command **riscv32-elf-gdb**.

3. In the GDB command line console, issue the following command to load the program executable (.elf file).

```
file <amazon-freertos>/build/[aws_demos.elf|aws_tests.elf]
```

4. Issue the command as follows to build the connection between your host computer and your board:

```
Target remote <your IP>:1111
```

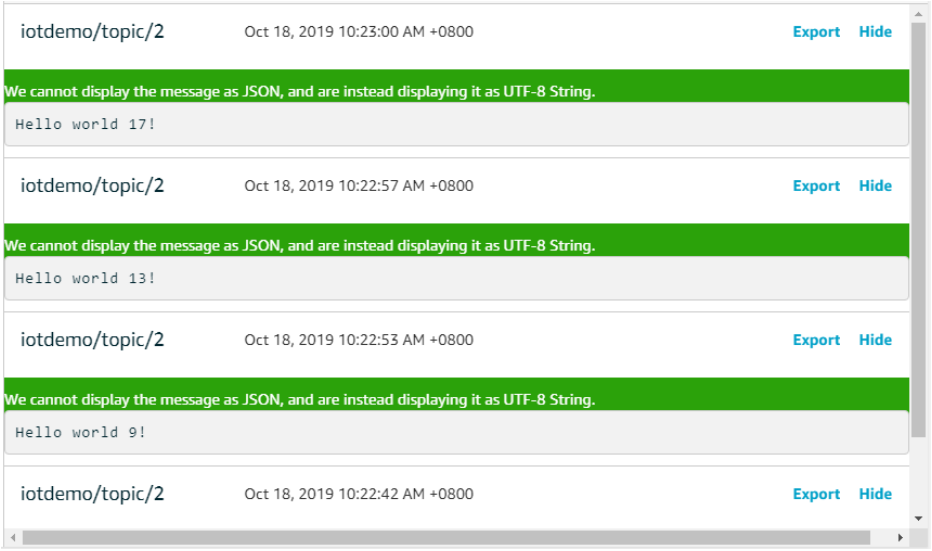
5. Use GDB commands to debug the FreeRTOS demo or test project.

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/topic/#**, and then choose **Subscribe to topic**.
4. Your **AWS IoT console** will display the topics that you subscribed.



Troubleshooting

For troubleshooting information about Getting Started with Amazon FreeRTOS, see Troubleshooting Getting Started (p. 64).