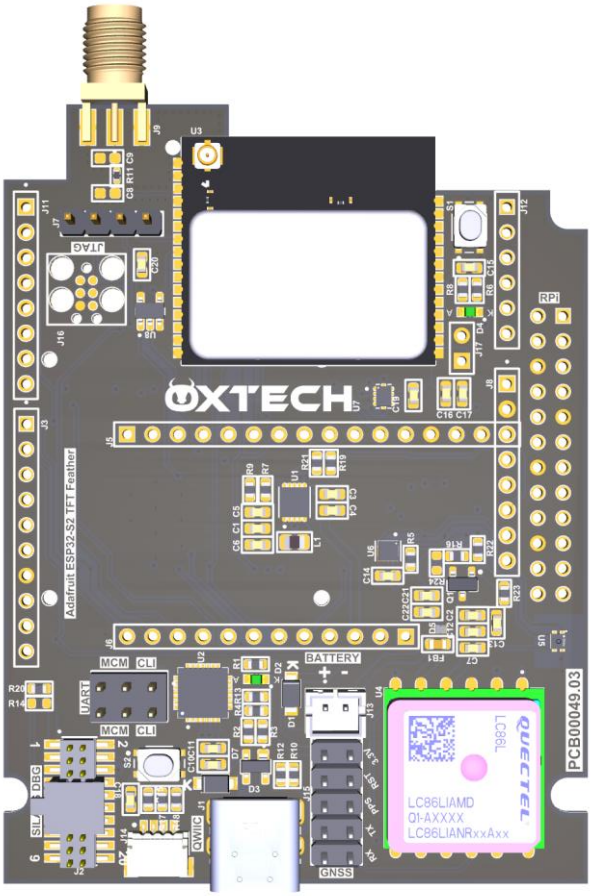
OxTech Multi-Connectivity Module



Demo Kit Guide

Document Status: Preliminary – Gen 2

Revision History

Date	Version	Author	Descriptions
08/13/2024	1.0	Oxit LLC	Initial Version
11/21/2025	1.1	Oxit LLC	Update Manufacturing File generation
12/13/2024	1.2	Oxit LLC	Added information about new Host and MCM FUOTA feature
03/21/2025	1.3	Oxit LLC	Updated according to new demo repo structure
04/10/2025	1.4	Oxit LLC	Condensed version

Table of Contents

1.Introduction

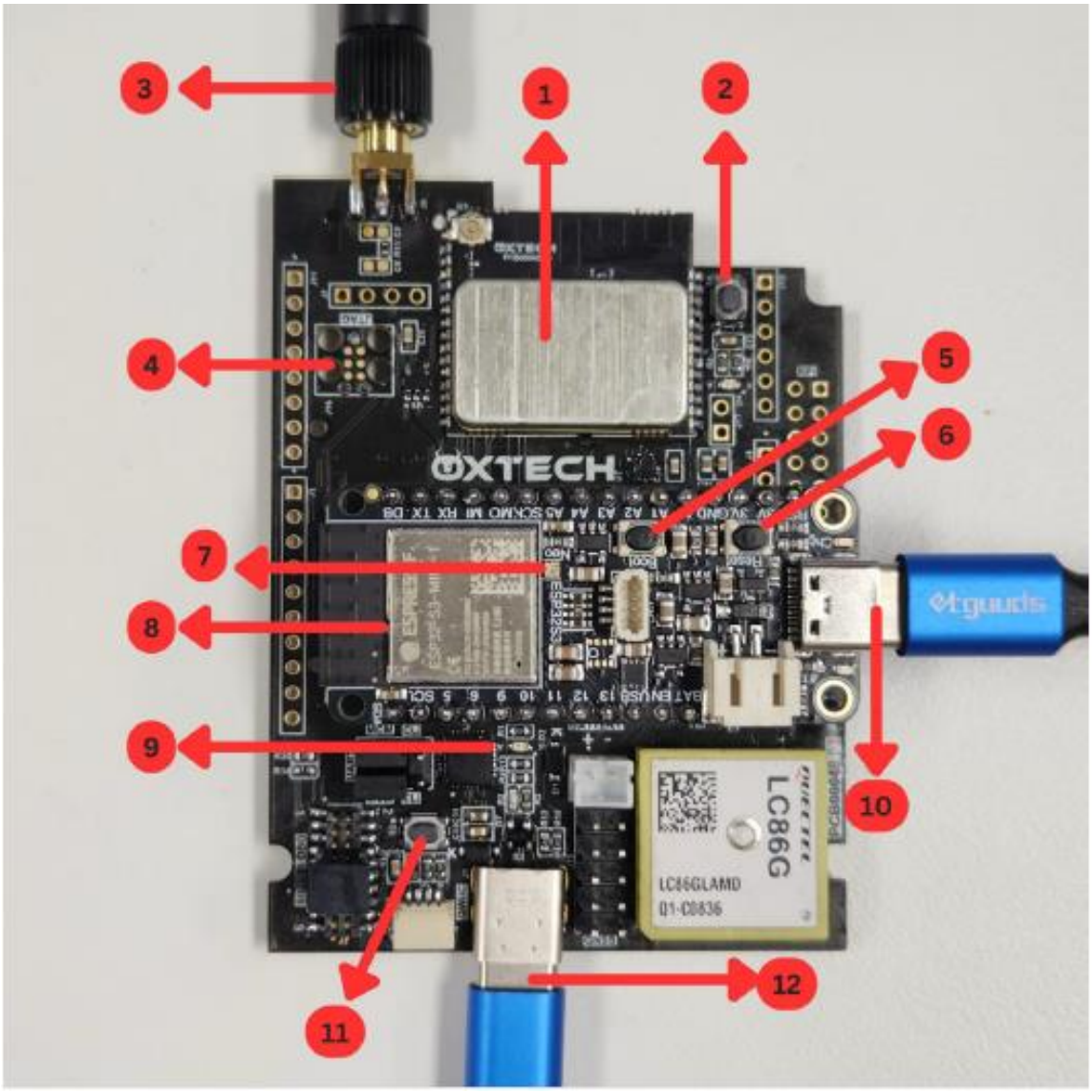
The OxTech Demo Kit enables seamless switching between LoRaWAN and Sidewalk networks without hardware or firmware modifications, enhancing deployment flexibility. It includes an ESP32-S3 Adafruit Feather as the host processor and the OxTech Multi-Connectivity Module (MCM) as the network co-processor, supporting LoRaWAN and Sidewalk. Additional components include a humidity & temperature sensor, RGB LED, buttons, and a USB-C connector.

The MCM combines a powerful EFR32MG24x microcontroller and a Semtech Sx126x LoRa transceiver, offers:

1. **Advanced security:** with robust data protection.
2. **Low power consumption:** for extended operation.
3. **Multi-connectivity:** supports Bluetooth Low Energy and Zigbee.
4. **LoRa connectivity:** for long-range, low-power LoRaWAN and Sidewalk communication.

1.1 Demo Kit

Onboard components:



1. **MCM**: Works in the network co-processor mode. Responsible for maintaining the LoRaWAN connection.
2. **User Button:** This button's action is currently undefined.
3. **Antenna**: An external antenna has been provided for the better reception of the signal. Please verify it's properly connected and there is no physical damage to the antenna.
4. **Jlink Tag connector footprint:** This connector is used to attach the Jlink to the board. Its whole purpose is to attach the Jlink here and update the MCM firmware.
5. **Boot Button:** Pressing this button along with the reset button will cause the ESP32-S3 feather board to enter in the bootloader mode.
6. **HOST Reset Button:** Pressing this button will reset the host module.
7. **RGB Led**: NeoPixel LED, it indicates various statuses such as device boot-up and uplink activity.
8. **Adafruit ESP32 S3 Feather**: Kit contains Adafruit esp32 s3 feather board, a powerful board that acts as a host for the MCM.
9. **Power LED:** This LED is on when the device is powered with the USB type C.
10. **Host USB Type-C:** Used to power the Adafruit ESP32-S3 Feather device, view logs from the host, and send CLI commands to control the host.
11. **MCM Reset Button:** Pressing this button will reset the MCM.
12. **MCM USB TYPE-C:** Used to power the MCM Devkit, view logs from the device, and send CLI commands to the devkit.

1.2 Demon flow

1. Read reboot count from ESP32-S3 Feather internal flash, update and store again.
2. Start connecting to LoRaWAN network or sidewalk, whichever is the active network.
3. Periodically read the temperature and humidity values from sensor.
4. Send confirmed uplink (only with LoRaWAN) with temperature, humidity and reboot count, according to connected network.
5. Print serial logs if acknowledged is received or not.
6. Print in serial logs if the downlink is available on terminal.
7. Features console commands for easy injection of LoRaWAN credentials.
8. Save the LoRaWAN credentials on internal flash if changed from console commands.
9. Various LED patterns to know the state of the device.
10. On pressing the user button, the kit will change its network configuration, which means it switches from LoRaWAN to Sidewalk and vice versa.
11. Firmware update of host as well as mcm module firmware. MCM can download the firmware file and after download, mcm will notify the host. The host will either download the file or can command the mcm to update itself.

2.Prerequisite

2.1 Hardware

1. Windows PC (preferably Windows 11)
2. USB-C data cable (for power)
3. Oxit Demo Kit
4. LoRaWAN gateway connected to the preferred LNS
5. Sidewalk gateway (e.g. Echo). For comprehensive list, visit Sidewalk Gateway
6. JLink for flashing manufacturing file (Sidewalk)
7. Familiarity with Arduino IDE is required, if you're not familiar with click here for resources to get started.

2.2 Software

1. Arduino IDE version 2.3.2

2.3 Network

1. **LoRaWAN Network Credentials** This includes DevEUI, JoinEUI, and AppKey.
2. **LoRaWAN Network Server (LNS) Setup** with access to:
    a. **Access to add device Credentials** (for registering demo kit on the network)
    b. **Access to view uplinks from device** (to see the data sent by demo kit)
    c. **Access to send downlinks to device** (optional, for sending downlinks to demo kit)
3. **AWS Account:** AWS account would be needed for:
    a. **Access to generate manufacturing files** This is like the credentials to connect with Sidewalk network.
    b. **Access to view uplinks from device** (to see the data sent by demo kit)
    c. **Access to send downlinks to device** (optional, for sending downlinks to demo kit)

3.Hardware Setup

Device can be powered with any data capable USB type C cable. To power the device, connect one end of the USB type C cable to the kit and another to the PC.

ⓘ * Be sure that USB port can supply enough power to demo kit (~500mA max), as required to drive both mcm kit and ESP32-S3 feather.

3.1 Setup for programming the MCM through Jlink

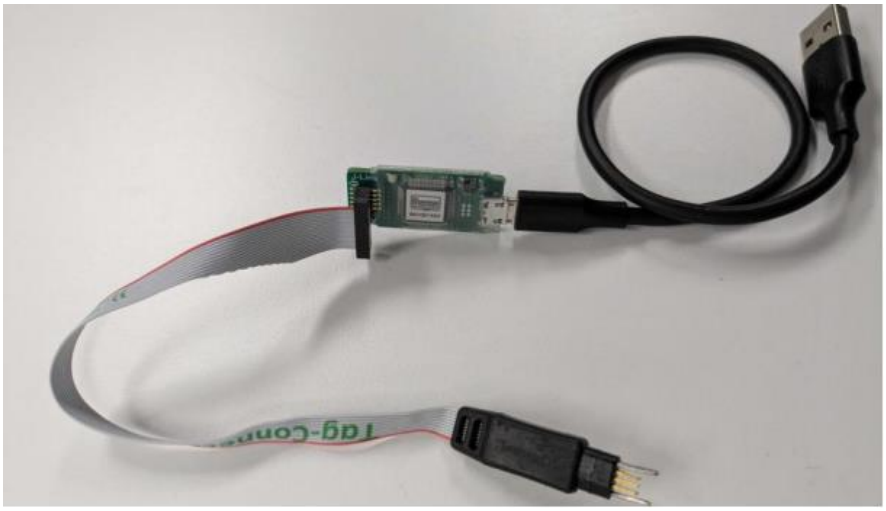
ⓘ *User may have different versions of Jlink and its associated cable. Please use whichever is appropriate for your device.

To perform this action, you may need administration rights. Please login with the administrator account.

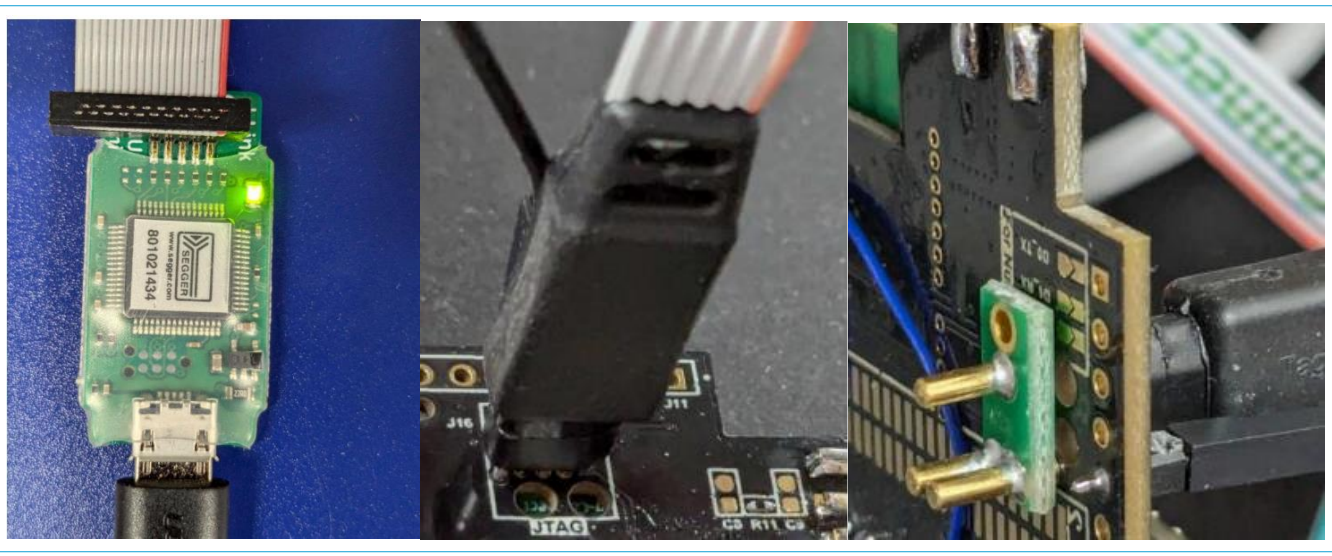
3.2.1 Hardware Setup

1. Connect the Jlink to the board using the tag connect cable.

2. Connect the JLink to PC using the Micro USB to USB cable.



3. Connect tag connector to the tag connector footprint on demo board.



4. Connect the demo board with PC using the USB-C to USB cable.

3.2.2 Software setup

1. Visit JLink Commander and download the JLink Commander and install it, according to the OS.
2. Download the **Simplicity Commander** according to the OS. Click on the links below to download the zip file without a Silicon Lab's account.
    a. For MacOS: https://www.silabs.com/documents/public/software/SimplicityCommanderMac.zip
    b. For Windows: https://www.silabs.com/documents/public/software/SimplicityCommanderWindows.zip
    c. For Linux: https://www.silabs.com/documents/public/software/SimplicityCommanderLinux.zip
3. Unzip the downloaded file.
    a. For MacOS traverse to **SimplicityCommander-*Mac/*** folder**.**
        i. There will be multiple zipped files, unzip *Commander-cli_win32_x64_1v1611b1660.zip*
    b. For Windows *unzip Commander-cli_win32_x64_1v16p11b1660.zip*
    c. For Linux unzip Commander_linux_x86_64_1v16p11b1660.tar.bz

> *** Filenames shown above may not match exactly due to updates from Silicon Labs**

4. Traverse to unzipped folder and you should see:
    a. For Windows: **Simplicity Commander/**
    b. For mac: Traverse to **Commander.app > Contents > MacOS**, right-click on **commander**, click on the `Show Package Contents` option. It will show the contents.
    c. For Linux: **Commander.**
5. Ensure that the new folder contains:
    a. For Windows: ▶ `commander.exe`
    b. For Linux/MacOS: ▶`commander`
6. Open the system provided shell:
    a. In Windows use command prompt (cmd)
    b. In Linux/MacOS use bash/zs

> *** You can open the terminal in current directory by right-clicking the folder and selecting, New Terminal at Folder (MacOS) or Open in Terminal (Windows/Linux)**

7. Users need to set the current folder (the folder which contains commander.exe or commander) to its current path. For this issue command:
    a. For Windows open the command prompt and use the command

```
`set PATH=%PATH%;{path to folder containing the commander}`
```

e.g.

```
`set PATH=%PATH%;C:\Users\OxitLab037\Downloads\SimplicityCommander-Windows\SimplicityCommander-Windows\Commander_win32_x64_1v16p11b1660\Simplicity-Commander`
```

    b. For Linux and MacOS open the shell and use the command

```
`export PATH=$PATH:{Path to folder containing the commander}`
```

e.g.

```
`export PATH=$PATH:/Users/oxitdeveloper/Downloads/SimplicityCommander-Mac/Commander.app/Contents/MacOS`
```

8. Download and go to the demo repository ***mcm-playground***, from the shell or terminal (depending on OS i.e. command prompt in Windows and bash in the case of MacOS and Linux).
    a. Folder includes:
        i. ***Documents/***
        ii. ***Examples/***
        iii. ***lib/***
        iv. ***tools/***
        v. ***README.md***
9. From shell, Navigate to the folder ***tools/oxtech-mcm-support/MCM-programming-tools/***
10. Test if commander executable can be accessed from the current folder.
    a. For Windows issue the command `` `commander.exe -version` ``, output should look like:

    b. For Linux/MacOS issue the command `` `commander --version` ``. Output should be similar.

> ***If commander or commander.exe is not accessible, please follow step 8.**
> **Please note above steps are for the x86_64 Linux PC not for Raspberry Pi.**

3.2.3 Steps to update the MCM firmware

1. Power cycle the demo board by disconnecting and reconnecting the USB cable.
2. Run script `erase.bat` in Windows and for MacOS and Linux run, `sh erase.sh`, the chip should unlock successfully.
3. Power cycle the demo board by disconnecting and reconnecting the USB cable.
4. Run the script *app_flasher.bat* `studio` for Windows and for mac and Linux run `sh app_flasher.sh studio`. On success this message would appear on console. It should say flashing completed successfully.
5. Power cycle the demo board by disconnecting and reconnecting the USB cable. Now MCM firmware has been updated and it's ready for testing.

> ***If a success message does not appear on the console, the user can repeat from step 1 to 4 again, any number of times.**

4.Software Setup

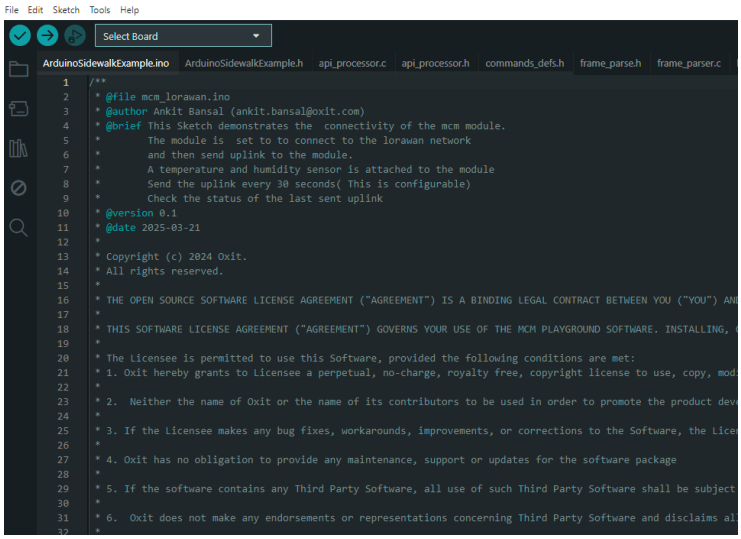
For detailed instructions on setting up the Arduino IDE and configuring the Adafruit ESP32-S3 Feather board, refer to the official Adafruit documentation: https://learn.adafruit.com/adafruit-esp32-s3-feather/arduino-ide-setup.

Once you have completed the setup as outlined in the guide, proceed with the following sections.

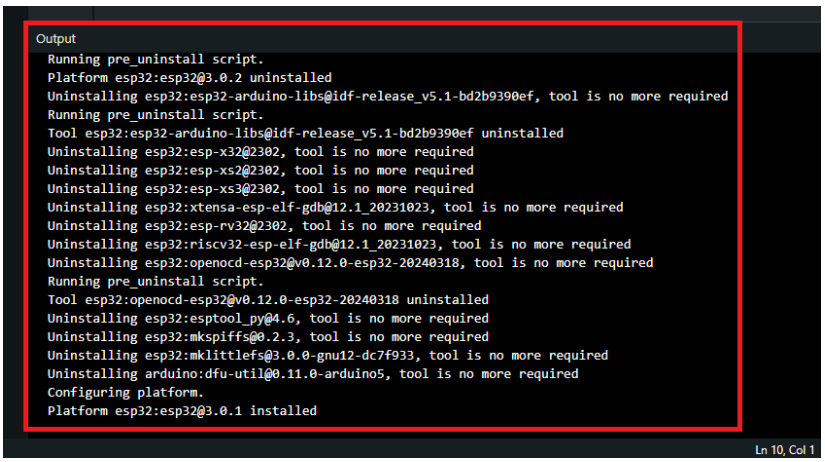
4.1 Board Setup

1. Open the Arduino IDE, navigate to the **mcm-playground** repository and open file ***mcm-playground/Examples/ArduinoESP32S3FeatherMultiProtocol/ArduinoMultiprotocolExample/ArduinoMultiprotocolExample.ino*** to open the demo project for Sidewalk.

2. Open the **board manager** by clicking on the board manager icon on the left side of the Arduino IDE.

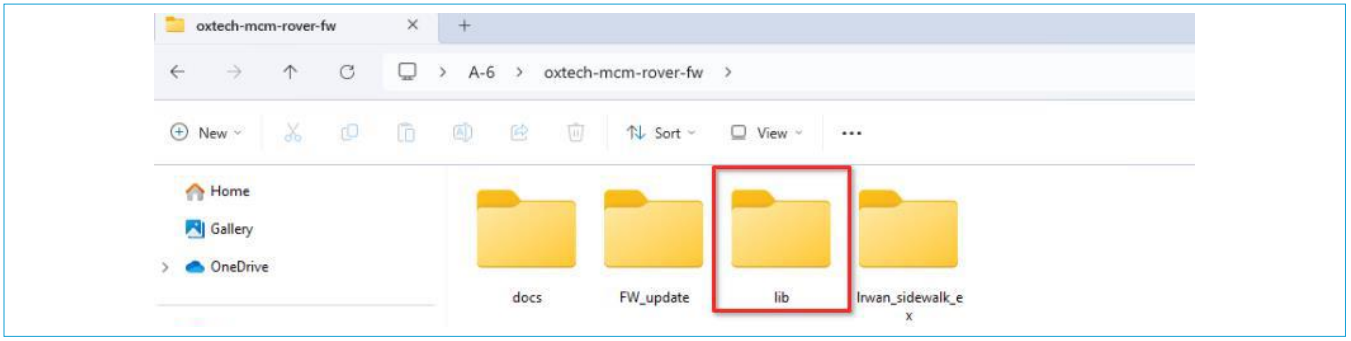


3. In the open window, type *esp32* and select the second option esp32 with version **no. 3.0.1** and click **install.**
4. Ensure that the installation process is completed successfully, in the output window you should see confirmation of the installation.



4.2 Adafruit Feather Library Setup

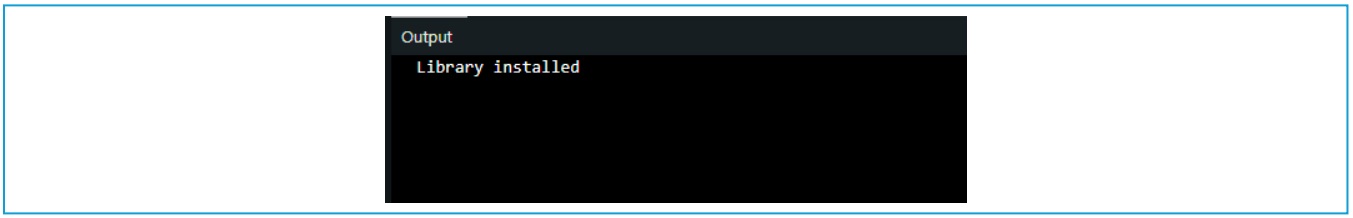
1. Open the **library manager** by clicking on the library manager icon.
2. First, install the **Adafruit neopixel** library by typing the name in the search box. Select **version 1.12.2** and click install, you should see confirmation of the successful installation.
3. Next install CLI library which is responsible for functioning of CLI commands. This library is available at path mcm-playground/lib/cli-lib.zip.



4. To install the library, click on **sketch > Include Library > Add .ZIP Library**
5. A dialog box appears, now navigate to the lib folder, as stated on step 3, point to the CLI library, cli-lib.zip, and click open. This will install the library, on the output window, confirmation of the installation should be displayed

4.3 CLI Commands

The



provided CLI commands allow you to manage LoRaWAN credentials on the device. These commands include setting LoRaWAN credentials, erasing LoRaWAN credentials and rebooting the device.

1. **Setting Dev Eui**: To set the Dev Eui, type the command:
   `oxit_cli deveui 0011223344556677`
2. **Setting Join Eui**: To set the Join Eui, type the command:
   `oxit_cli joineui 0011223344556677`
3. **Setting App Key**: To set the App Key, type the command:
   `oxit_cli appkey 0011223344556677`
4. **Erasing LoRaWAN credentials**: To erase the saved credentials:
   `oxit_cli erase 0011223344556677`
5. **Reboot Host**: To reboot application processor:
   `oxit_cli reboot`

4.4 Cloud Setup

4.4.1 LoRaWAN

Recommended device configuration parameters for registration and correct operation:

4. LoRaWAN specification version: v1.0.4 (*supports v1.0.2)
5. LoRaWAN class: Class A
6. Regional parameters:
    i. Region: US915
7. Version: RP002-1.0.2

4.4.2 Sidewalk

Configuration for Sidewalk network:

1. **Link type for registration:** FSK
2. **Link type for communication:** CSS (LoRa)
3. **Power profile:** Profile A

4.5 Generate sidewalk manufacturing files

A manufacturing file is a credentials file for Sidewalk, necessary to enable the MCM to connect to and communicate over Amazon Sidewalk network.

The manufacturing page is now fully stored in the default NVM3 area, eliminating the use of dedicated NVM3 areas.

> *Please note that this new version does not support backward compatibility for the manufacturing page.

4.5.1 Prerequisites

1. Simplicity Commander
2. Python 3
3. pip3
4. AWS CLI configured on your machine
    I. Run `aws configure` in your terminal and make sure you are using correct AWS access key and secret keys for your AWS IAM User account where you wish to create the MFG.

4.5.2 Steps to Generate MFG Files

We will need to use the scripts provided in the Silicon Labs extension for Amazon Sidewalk GitHub repository.

1. Clone the repository based on your SDK version in your local machine – switch to the corresponding branch based on your SDK version.
    a. sisdk-2024.6: Currently being used for MCM application, users should switch to this branch.
    b. gsdk_4.3
    c. gsdk_4.4

> *For SDK 4.3.0, modify `ProvisionWrapper.py` file located in **amazon-sidewalk\tools\scripts\public\generate_prototype\libs** to ensure compatibility with Windows OS.

2. In the python script remove line 111 and 112 and replace with the following (careful to include correct indent and click save.) NOTE: script may match the following python code

```
      If (not self.is_context_studio):
            If platform.system() == "Windows":
                  args.insert(0, "python.exe")
      else:
            args.insert(0, "python3")
```

3. In your preferred terminal navigate to ***amazon-sidewalk\tools\scripts\public\generate_prototype\amazon_dependencies*** directory and install the necessary requirements using the command below:
    a. `pip3 install -r requirements.txt.`
4. Navigate back to ***amazon-sidewalk\tools\scripts\public\generate_prototype***
5. The script configuration can be managed using two methods and is shown in the following sections.

4.5.2.1 Method 1 (Using JSON Config)

1. Modify file in ***amazon-sidewalk\tools\scripts\public\generate_prototype\example_config.json*** and save it after modifying it as below

```
{
    "awsAccount": {
        "awsRegion": "us-east-1",
        "awsProfile": "default"
   },
"commanderPath":
"C:\\SiliconLabs\\SimplicityStudio\\v5\\developer\\adapter_packs\\commander\\commander.exe",
    "generateRequests": [
        {
         "deviceProfileId": null,
         "deviceName": null,
         "destinationName": "CFSDESTINATION",
         "targetPart": null,
         "quantity": 1
        }
    ]
}
```

2. The first block **awsAccount** is used to configure the connection to AWS cloud:
    a. **awsRegion**: The AWS region you would like to add your devices to (Note that only the **us-east-1** region is supported currently).
    b. **awsProfile**: Used to choose the AWS CLI profile linking to the correct credentials. If you do not use profiles, you can leave the default.
3. The second block defines the toolchain used to create the manufacturing page:
    a. **commanderPath**: Should link to the Simplicity Commander executable file, make sure you have the correct path.

4. The third block **generationRequests** controls the target device details:
    a. **deviceProfileId**: Replace with the Profile ID previously created with double quotes (example: "*7c51bc6b-0556-2083-6f0d-aeb750c94508*") or keep it empty, and script will automatically generate new device profile for the first run.
    b. **deviceName**: This is optional, you can choose to give a custom name to your devices.
    c. **destinationName**: Should be the name of the destination used by your AWS cloud application.
    d. **targetPart**: The OPN of the microcontroller target part. If empty, manufacturing pages for all supported radio boards will be generated. (can be "*xg21*", "*xg23*", "*xg24*", "*xg28*", example input: "*efr32mg24b220F1536*")
    e. **quantity**: The quantity of devices you want to create (default value is 1).

5. On the first run, the script creates a prototyping device profile if the corresponding field is kept empty and fills the ***deviceProfileId*** with the resulting device ID. All subsequent wireless devices will be created using this device profile.

6. Follow the steps below to run the script:

> ***That during prototyping, a device profile can be linked to a maximum of 1000 wireless devices.**

    a. Create an output folder where you wish to store the manufacturing file because the script will not automatically create it within the same path.
    b. `python3 generate_protoype.py -input <> --output <output_folder> --config example_config.json`
    c. Check the example below using command prompt on Windows.

```
py -3 generate_prototype.py --input . --output out --config example_config.json
```

    d. The script creates a directory structure as follows in the chosen output folder under **mfg_output**:

```
DeviceProfile_7c51bc6b-0556-2083-6f0d-aeb750c94508
├── DeviceProfile.json
└── WirelessDevice_db57b1c9-22ad-c052-07ae-1e1cae9bb384
    ├── SiLabs_MFG.nvm3
    ├── Silabs_xG21.s37
    ├── Silabs_xG23.s37
    ├── Silabs_xG24.s37
    ├── Silabs_xG28.s37
    └── WirelessDevice.json
```

4.5.2.2 Method 2 (Using CLI Arguments) (not recommended)

1. Using command line arguments – this will create new device profile within each run, read more about it https://docs.silabs.com/amazon-sidewalk/latest/sidewalk-app-development/.
   Example command:

```
py -3 generate_prototype.py --input . --output out --dst-name CFSDestination -aws profile default - -commander C:\\SiliconLabs\\SimplicityStudio\\v5\\developer\\adapter_packs\\commander\\commander.exe
```


4.5.3 Useful Links

1. https://docs.silabs.com/amazon-sidewalk/latest/sidewalk-app-development/
2. https://github.com/SiliconLabs/amazon-sidewalk

4.6 Programming Sidewalk Manufacturing File to MCM

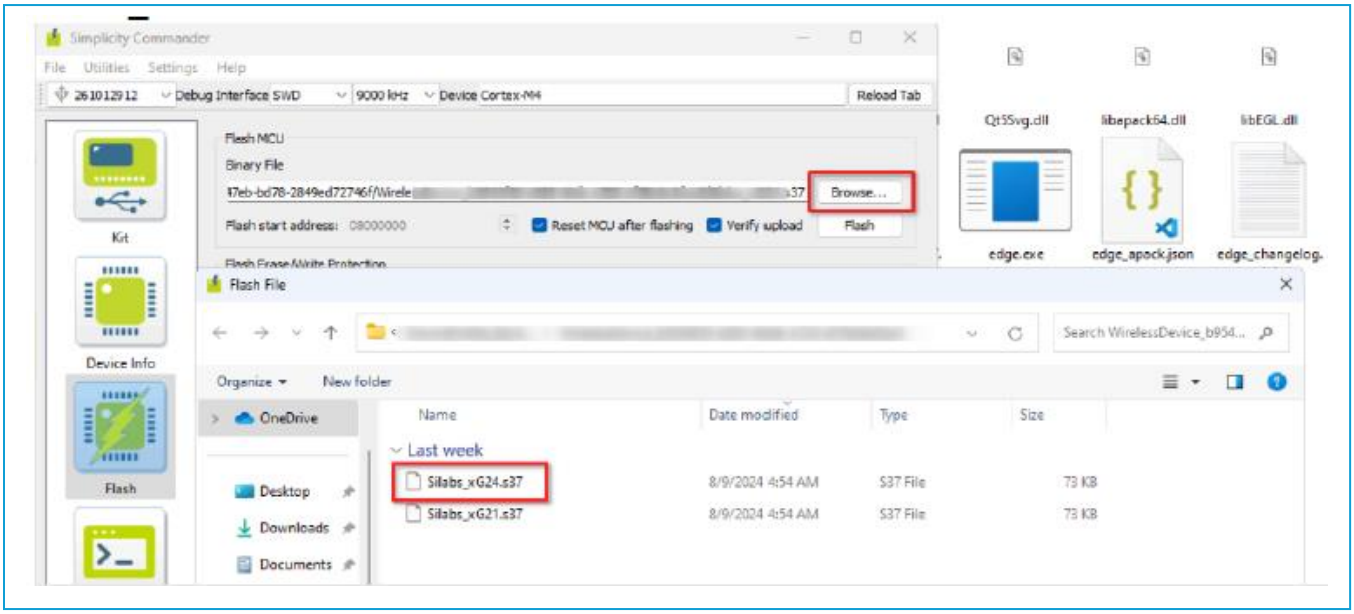
Users can use the 2 modes to program the manufacturing file, using JLink.
To program Sidewalk manufacturing file, please go through hardware setup and install the Jlink commander and unzip simplicity commander as explained above.

1. Run simplicity commander by:
    a. For Windows, double click on ▶ **`commander.exe`**
    b. For MacOS, double click on ▶ **`commander.app`**
    c. For Linux, double click on ▶ **`commander`**

2. Click on **Select Kit** in the top left and click on Jlink that is connected.
3. Select the option **Flash**, if any terms and conditions are there, please click **Accept** after reading the terms and conditions.
4. Click on **browse** and select the path for the manufacturing file. Select the file *Silabs_xG24.s37*. If **targetPart** was left **NULL** during setup, the file name may be something like, *MFG_XG24.s37* just ensure the file ends in xG24.s237. Refer to 4.5.2.1 Method 1 (Using JSON Config) .
5. Click on **Flash** in the left column. This will write the manufacturing file to the device.
6. Reboot the device by disconnecting and re-connecting the USB cable. Now the device is ready for Sidewalk network.

4.7 FUOTA for MCM or Host

Since large file updates are not possible through LoRaWAN, updating the host as well as the MCM module is quite a challenge.
To mitigate this issue a strategy has been adopted. A large file can be broken into small pieces called segments. We can transmit these segments and reassemble them on MCM.
To reliably reconstruct these segments, we need to transfer the meta data related to these segments. This metadata will need to be transmitted in the form of downlink just before starting the file transfer using the LoRaWan.



The ***mcm-playground*** repository includes ***tools/MCM-FUOTA-tools/*** as a private submodule which contains the script to split the file to multiple segments and the method to transmit the file.
Follow the ***README.md*** file included in the folder, to know how to generate the segment and what downlinks needed to be done to perform the FUOTA.
When the MCM receives all the segments, it validates and notifies the host regarding it. Host will either start the file transfer in case the firmware files belong to the host or trigger the firmware if it belongs to the mcm.

5.Build and Run

***Before building the project, please connect the demo kit to the PC using the USB type C cable. Fore more info, please refer to the hardware setup section above.**

5.1 Selection of Module

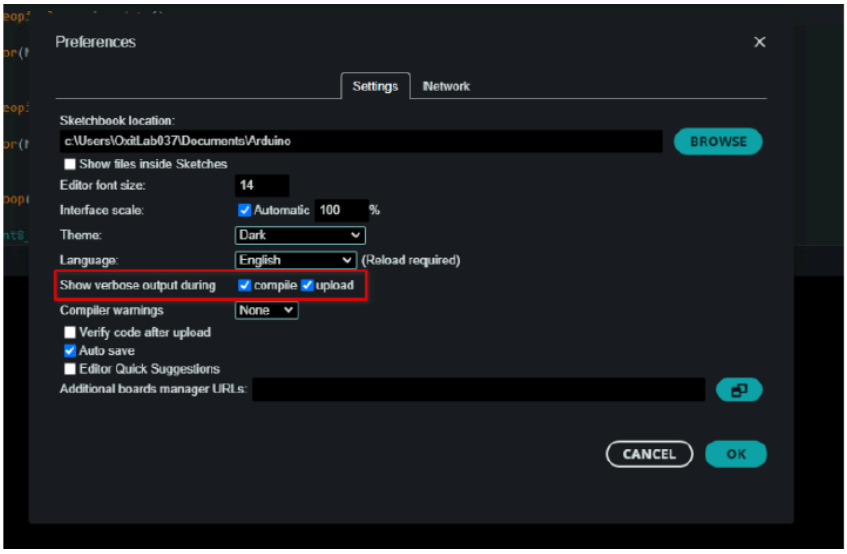
The demo kit uses the Adafruit Feather ESP32-S3 2MB PSRAM, for its selection go to **Tools > Board > esp32 > Adafruit Feather ESP32-S3 2MB PSRAM**

Similarly select the COM port to which the demo kit is connected. Select the COM port by clicking **Tools > Port**: > COMx (As shown in the picture, setup's com port)

5.2 Verbose Logs (Optional)

While not required, enabling verbose logging during the firmware compilation and upload process is highly recommended. This detailed output provides valuable insights for debugging any potential issues that may arise.

To enable verbose logging, click on **file > Preferences**, this opens a dialog box, check the ***compile*** and ***upload*** checkbox, in front of the ***Show verbose output during option***

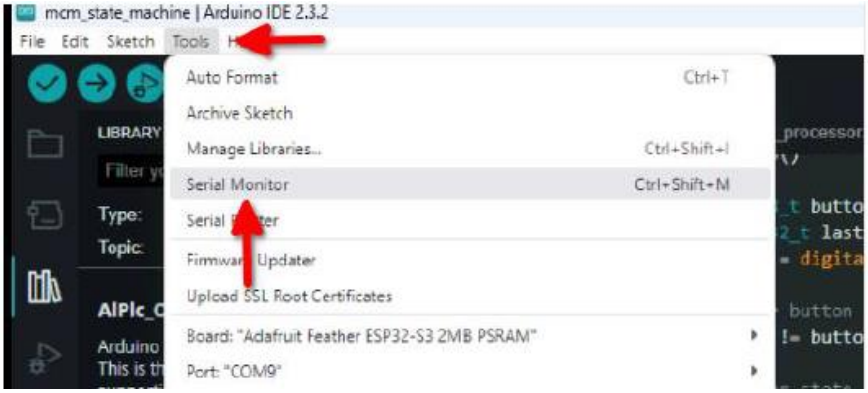


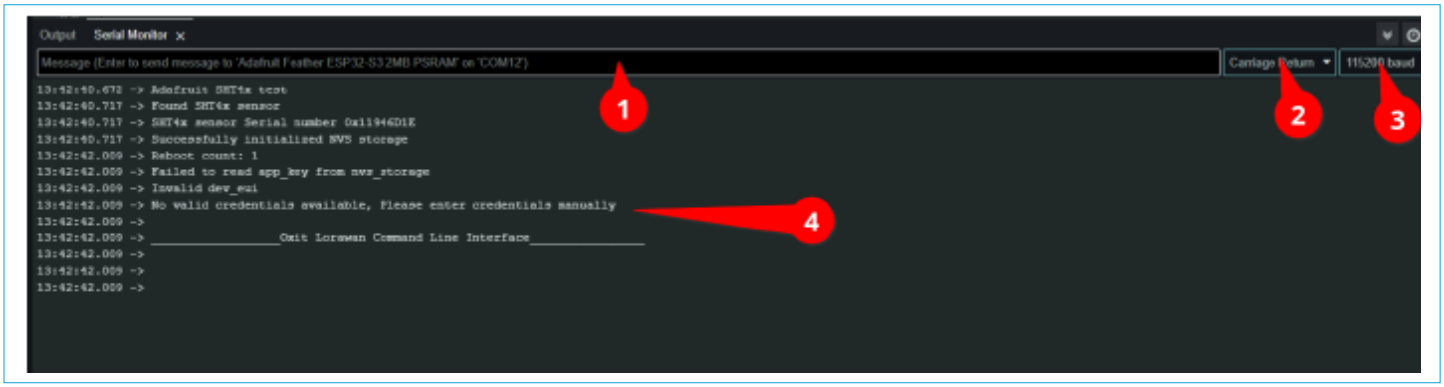
5.3 Programming the Arduino demo firmware

Firmware can be programmed by clicking the arrow button in the top left of the Arduino IDE. On the output screen, if everything is working properly. It should show "`writing at...`" logs and in the end, there should be log "`Leaving... Hard resetting via RTS pin.`"

5.4 Monitoring the serial logs

To monitor the logs from the Arduino IDE, click on **Tools > Serial Monitor**





In the serial monitor there are four things to look for:

1. CLI commands mentioned in the 4.3 CLI Commands can entered here and it will send the command to the demo kit.
2. Set this option to **carriage return**, this is necessary as our CLI commands terminates on the carriage return.
3. Select **baud rate** to **115200**
4. This is the viewing area where system logs can be observed.

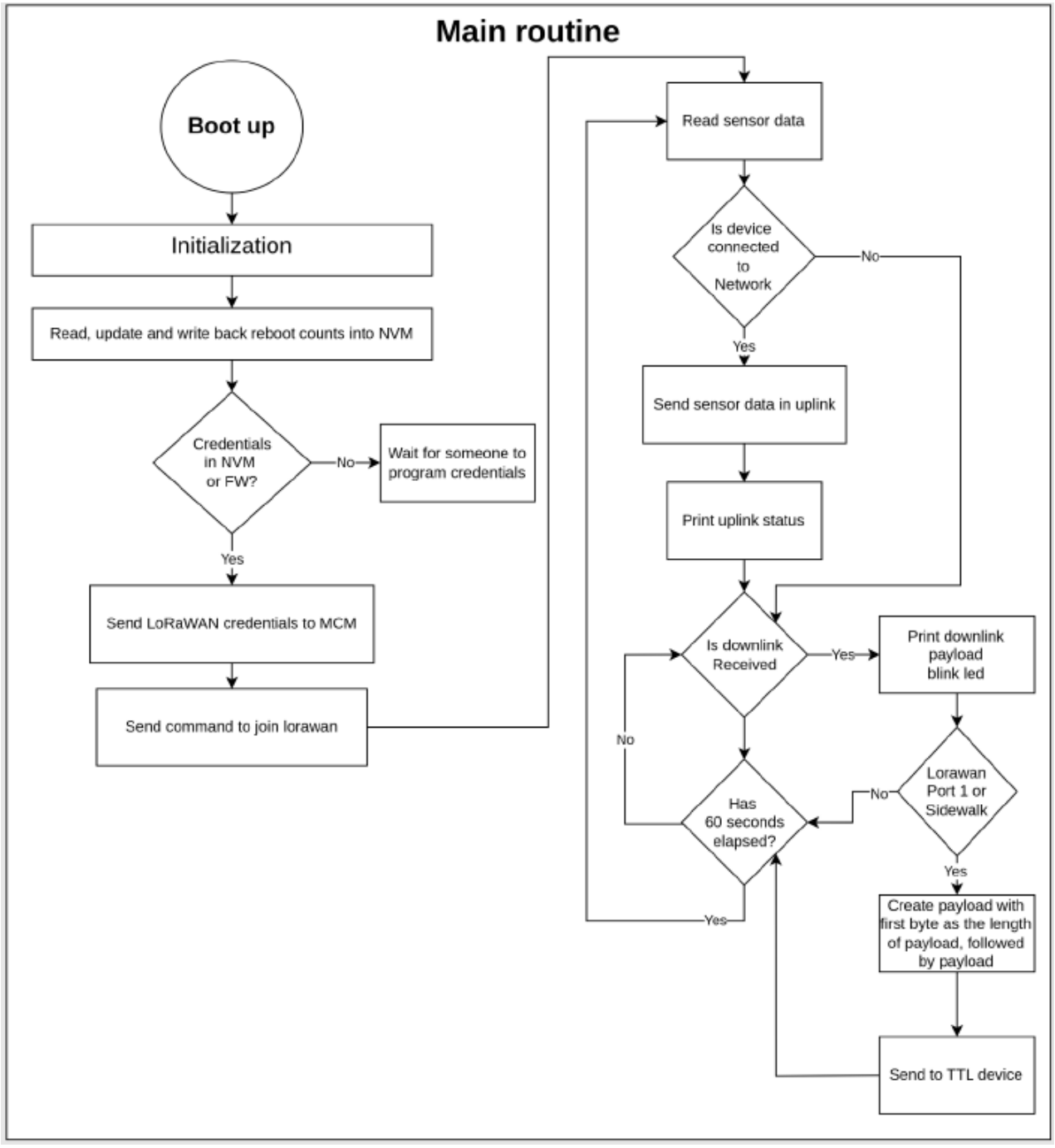
When the device starts, a window called Serial Monitor may not display logs immediately. This is because the monitor may not be active until sometime after startup (Feather ESP32-S3 kit uses the same UART for programming and viewing the serial logs in our setup).

To view the complete system logs, including those from startup, you will need to reset the host device. You can do this in two ways:

1. Press the physical **RESET** **(Rst)** button on the Host (e.g. ESP32-S2/S3).
2. Use the CLI command `oxit_cli reboot` through the existing connection.

Both methods will restart the device and activate the Serial Monitor the same way, allowing you to see all logs from the initial boot process.

6.General demo flow walkthrough



6.1 initialization

During boot up, the device follows a specific order to initialize essential components as mentioned below:

1. **Serial Logs:** Initialize host serial logs. The baud rate is set to 115200.
2. **EVK Led:** Initialize the user LED on the MCM evaluation kit.
3. **I2C bus:** Initialize I2C bus.
4. **SHT4X sensor:** Demo kit has the i2c based temperature and humidity sensor. This sensor is based on the i2c bus.
5. **RGB LED:** Demo kit contains the neo pixel RGB LED, for status display.
6. **CLI console:** Device initializes a CLI console for entering the cli commands.

6.2 Get Data from Internal Flash

1. **Reboot count**: Whenever device reboots, it gets the reboot counter from internal flash. It increments the reboot counter and saves it again on internal flash.
2. **LoRaWAN Credentials**: When the device boots up it finds the LoRaWAN credentials from internal flash, which was saved when last time user entered them from CLI. These credentials are DevEUI, JoinEUI and AppKey.

6.3 Choosing the Credentials

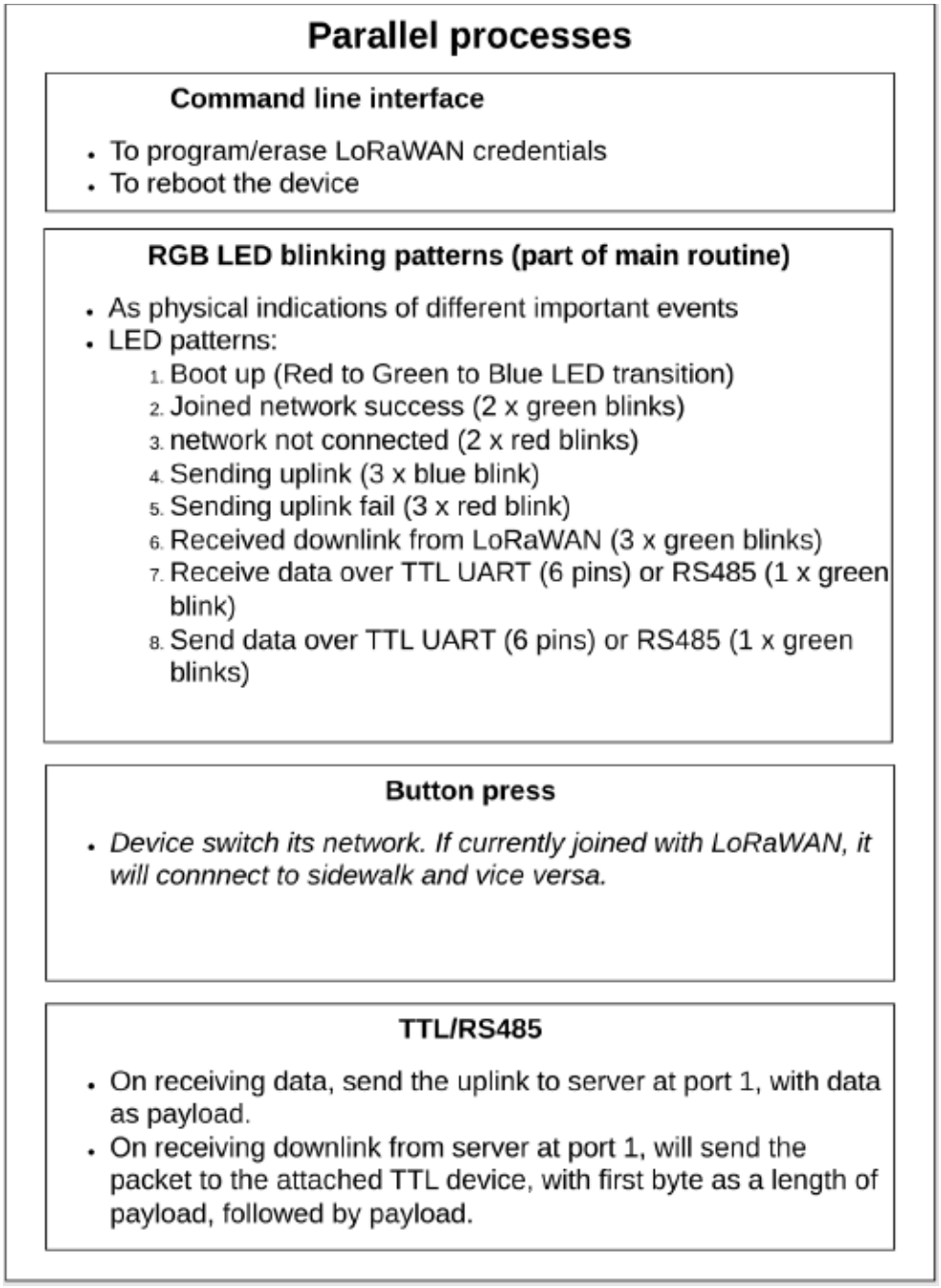
1. Firmware validates the credentials from internal flash and uses them for joining process.

2. If credentials from internal flash are not valid, it selects the credentials from firmware itself. Users can provide their own credentials while building the firmware. Instead of the 0xFF, please place valid credentials.

***If the credentials are not provided from firmware or CLI, the device will no do anything regarding the LoRaWAN connection.**

6.5 Other Parallel Processes

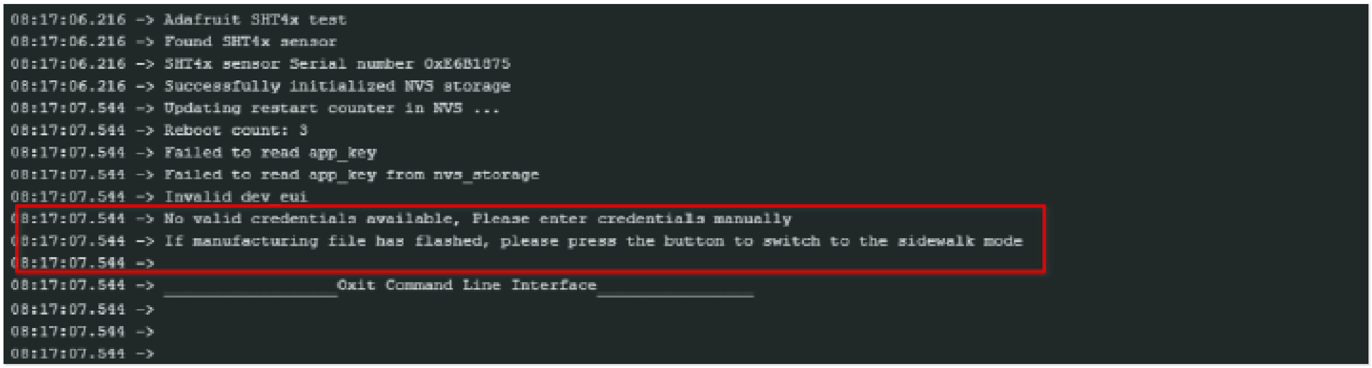
The MCM's Parallel Process are shown in Figure below.



7.Monitor logs

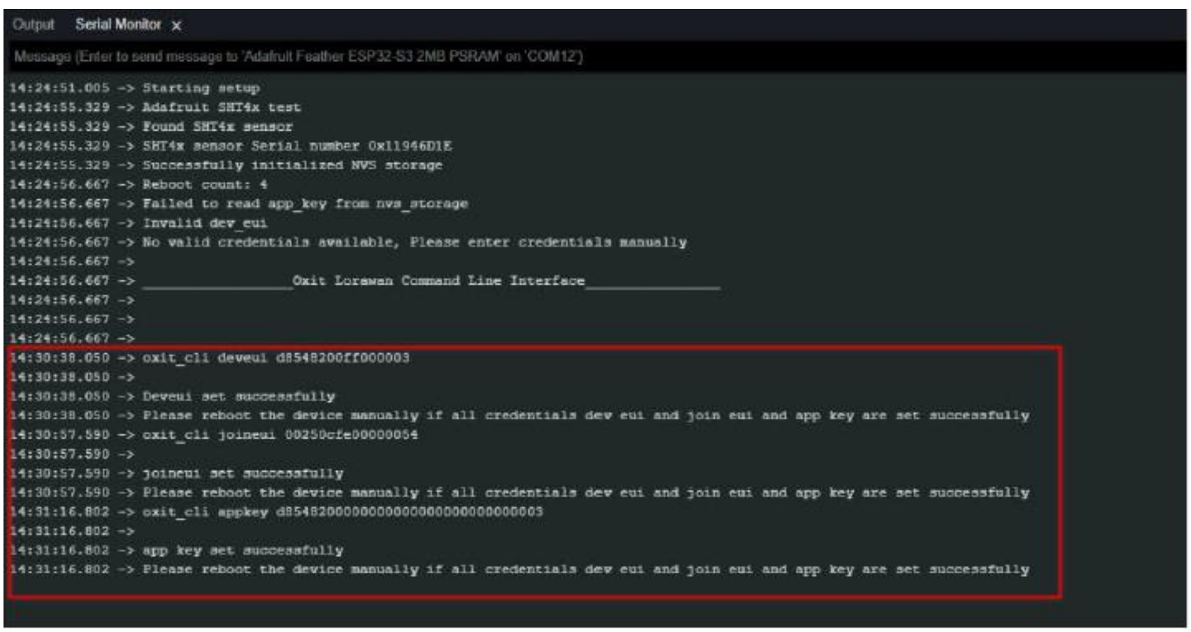
7.1 Device Logs

There are no LoRaWAN credentials and Sidewalk manufacturing file present in the new demo kit. Users need to provide the credentials using the CLI. Refer to 4.3 CLI Commands .



```
08:17:06.216 -> Adafruit SHT4x test
08:17:06.216 -> Found SHT4x sensor
08:17:06.216 -> SHT4x sensor Serial number 0xE6B1875
08:17:06.216 -> Successfully initialized NVS storage
08:17:07.544 -> Updating restart counter in NVS ...
08:17:07.544 -> Reboot count: 3
08:17:07.544 -> Failed to read app_key
08:17:07.544 -> Failed to read app_key from nvs_storage
08:17:07.544 -> Invalid dev eui
08:17:07.544 -> No valid credentials available, Please enter credentials manually
08:17:07.544 -> If manufacturing file has flashed, please press the button to switch to the sidewalk mode
08:17:07.544 ->
08:17:07.544 -> ________________Oxit Command Line Interface________________
08:17:07.544 ->
08:17:07.544 ->
08:17:07.544 ->
```

***The device (MCM) is not pre-flashed with the manufacturing file. User needs to flash the manufacturing file. Please check the above section, how to generate and flash the manufacturing file.**



```
Output   Serial Monitor x
Message (Enter to send message to 'Adafruit Feather ESP32-S3 2MB PSRAM' on 'COM12')
14:24:51.005 -> Starting setup
14:24:55.329 -> Adafruit SHT4x test
14:24:55.329 -> Found SHT4x sensor
14:24:55.329 -> SHT4x sensor Serial number 0x11946D1E
14:24:55.329 -> Successfully initialized NVS storage
14:24:56.667 -> Reboot count: 4
14:24:56.667 -> Failed to read app_key from nvs_storage
14:24:56.667 -> Invalid dev_eui
14:24:56.667 -> No valid credentials available, Please enter credentials manually
14:24:56.667 ->
14:24:56.667 -> ________________Oxit Lorawan Command Line Interface________________
14:24:56.667 ->
14:24:56.667 ->
14:24:56.667 ->
14:30:38.050 -> oxit_cli deveui d8548200ff000003
14:30:38.050 ->
14:30:38.050 -> Deveui set successfully
14:30:38.050 -> Please reboot the device manually if all credentials dev eui and join eui and app key are set successfully
14:30:57.590 -> oxit_cli joineui 00250cfe00000054
14:30:57.590 ->
14:30:57.590 -> joineui set successfully
14:30:57.590 -> Please reboot the device manually if all credentials dev eui and join eui and app key are set successfully
14:31:16.802 -> oxit_cli appkey d8548200000000000000000000000003
14:31:16.802 ->
14:31:16.802 -> app key set successfully
14:31:16.802 -> Please reboot the device manually if all credentials dev eui and join eui and app key are set successfully
```

Provide the DevEUI, JoinEUI and AppKey.

Reset the device with either the reset button on the ESP32-S3 or by entering the command ‘`oxit_cli reboot`’. In our case we used the CLI command to reset the device, and the device rebooted in 3 seconds.


```
Output  Serial Monitor ×
Message (Enter to send message to 'Adafruit Feather ESP32-S3 2MB PSRAM' on 'COM12')
05:26:11.607 -> Adafruit SHT4x test
05:26:11.642 -> Found SHT4x sensor
05:26:11.642 -> SHT4x sensor Serial number 0x11946D1E
05:26:11.642 -> Successfully initialized NVS storage
05:26:12.959 -> Reboot count: 1
05:26:12.959 -> Failed to read app_key from nvs_storage
05:26:12.959 -> Invalid dev_eui
05:26:12.959 -> No valid credentials available, Please enter credentials manually
05:26:12.959 ->
05:26:12.959 -> _________________Oxit Lorawan Command Line Interface_______________
05:26:12.959 ->
05:26:12.959 ->
05:26:12.959 ->
05:26:39.360 -> oxit_cli deveui d8548200ff000003
05:26:39.360 ->
05:26:39.360 -> Deveui set successfully
05:26:39.360 -> Please reboot the device manually if all credentials dev eui and join eui and app key are set successfully
05:26:49.250 -> oxit_cli joineui 00250cfe00000054
05:26:49.250 ->
05:26:49.250 -> joineui set successfully
05:26:49.250 -> Please reboot the device manually if all credentials dev eui and join eui and app key are set successfully
05:26:56.567 -> oxit_cli appkey d8548200000000000000000000000003
05:26:56.602 ->
05:26:56.602 -> app key set successfully
05:26:56.602 -> Please reboot the device manually if all credentials dev eui and join eui and app key are set successfully
05:27:14.840 -> oxit_cli reboot
05:27:14.840 -> Rebooting device in 3 seconds
```

7.2 Analyze logs from boot process

TIP: Review the following logs to understand the boot process after startup.

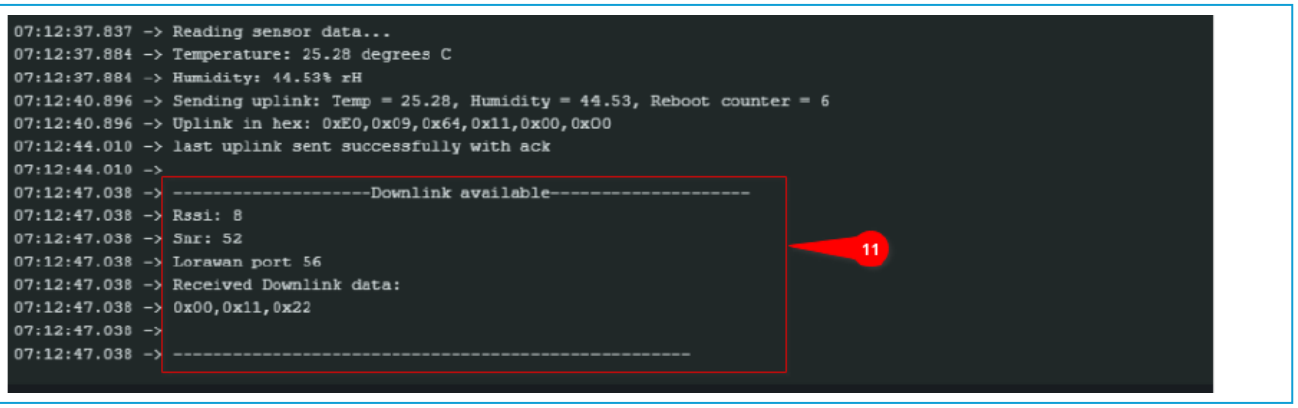
The images below show the process:

```
08:27:56.853 -> [0;32mI (101) boot:  5 ffat            Unknown data     01 81 Starting setup
08:28:01.196 -> Adafruit SHT4x test
08:28:01.196 -> Found SHT4x sensor
08:28:01.196 -> SHT4x sensor Serial number 0xE6B1875
08:28:01.196 -> Successfully initialized NVS storage
08:28:02.474 -> Updating restart counter in NVS ...
08:28:02.513 -> Reboot count: 5
08:28:02.513 -> Using saved lorawan credentials from nvs storage
08:28:02.513 -> Device EUI: 0x90,0x11,0x22,0x33,0x44,0x55,0x66,0x77
08:28:02.513 -> Join EUI: 0x77,0x66,0x55,0x44,0x33,0x22,0x11,0x00
08:28:02.513 -> Network key: 0x00,0x11,0x22,0x33,0x44,0x55,0x66,0x77,0x88,0x99,0x99,0x88,0x77,0x66,0x55,0x44
08:28:02.912 ->
08:28:02.912 -> _________________Oxit Command Line Interface_________________
08:28:02.912 ->
```

1. The sensor (SHT4x) and internal flash status. In our setup it shows both initialized properly.
2. The reboot count of device from internal flash.
3. The credentials have been found and read from internal Flash.
4. The credentials of the device being connected.
5. Command line interface started, and ready to accept commands.
6. Shows that kit is trying to join the network.

7. Shows device is trying to read the values of temperature and humidity.
8. Shows that the device is still not connected to the network.
9. After some seconds the device joins successfully.
10. The block of logs below does the following:

a. Read sensor data
b. If the device is connected to the network, it triggers an uplink.
c. Uplink data consists of temperature, humidity and reboot counts.
d. Shows the uplink in hex format.
e. Shows the last uplink status, here we do the confirm uplink.
f. It repeats the sequence from step 10 every 60 seconds



11. On receiving the downlink, logs show RSSI, SNR, Port, and the payload in hex format.
12. On pressing the button, the device switches from LoRaWAN to Sidewalk network and tries to join the network. On pressing another time, the device switches back to the LoRaWAN network.

***Before pressing user button, please ensure that the device has valid LoRaWAN credentials and for Sidwalke ensure a valid manufacturing file has been flashed.**

7.3 LNS Logs

Please login to the preferred LNS and monitor the uplinks and issue the downlink from there.

7.4 Interacting with Amazon Sidewalk

Users can observe the device uplink using the AWS MQTT test client. For initiating the downlink can use the AWS cli. For detailed procedures please follow this guide.

8.Troubleshooting

8.1 Unable to Flash Firmware

The ESP32-S3 Feather board should enter bootloader mode automatically during flashing, there may be times when you need to trigger it manually. Without the board in bootloader mode, the code will not be uploaded successfully.

Here is a typical error message you might see in such a scenario

Steps to solve the issue and boot up the device in normal mode:

1. Press and hold boot button, press and release the reset button, then release the reset button. The device will be in boot loader mode.
2. Flash the device as usual by clicking the flash button.
3. Press the reset button.
4. Open the serial monitor to observe the logs.

please refer to Adafruit documentation:
https://learn.adafruit.com/adafruit-esp32-s3-feather/using-with-arduino-ide https://learn.adafruit.com/adafruit-esp32-s3-feather/arduino-blink

8.2 Sensor Not Detected

If you see errors like "cannot find SHT4x" or "Failed to initialize NVS storage" during boot, it suggests potential issues with I2C device detection. Double-check the wiring connections for I2C devices.

```
08:09:58.470 -> entry 0x403c9880
08:09:59.760 -> Starting setup
08:10:04.105 -> Adafruit SHT4x test
08:10:04.105 -> Couldn't find SHT4x
08:10:04.105 -> Failed to initialize NVS storage
08:10:05.378 -> Reboot count: 0
08:10:05.378 -> Failed to read app_key from nvs_storage
08:10:05.378 -> Invalid dev_eui
08:10:05.378 -> No valid credentials available, Please enter credentials manually
08:10:05.378 ->
08:10:05.378 -> ________________Oxit Lorawan Command Line Interface______________
08:10:05.424 ->
08:10:05.424 ->
08:10:05.424 ->
```

8.3 Demo Kit Failing to Join the LoRaWAN Network

If the device fails to join LoRaWAN network, please ensure that:

1. Device is registered with the LNS. Please double check the credentials.
2. The boot logs will display the credentials used for joining the LoRaWAN network. Please verify that these credentials match the ones you've configured.

```
08:43:26.984 ->
08:43:26.984 -> Device EUI: 0xD8,0x54,0x82,0x00,0xFF,0x00,0x00,0x03
08:43:26.984 -> Join EUI: 0x00,0x25,0x0C,0xFE,0x00,0x00,0x00,0x54
08:43:26.984 -> Network key: 0xD8,0x54,0x82,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03
08:43:27.395 ->
08:43:27.395 -> __________________Oxit Lorawan Command Line Interface________________
```


8.4 Demo Kit Failing to Join the Sidewalk Network

If the device fails to join Sidewalk, please ensure that a valid manufacturing file has been flashed on the MCM.

For generating and flashing the manufacturing file please refer to Section 4.5.

9.Resources

7. Adafruit ESP32S Feather link https://learn.adafruit.com/adafruit-esp32-s3-feather/overview
8. Arduino IDE download, https://www.arduino.cc/en/software
9. Link to download the simplicity commander, https://www.silabs.com/developers/simplicity-studio/simplicity-commander?tab=downloads
10. Downlink JLink commander https://www.segger.com/downloads/jlink/
11. List of Sidewalk gateways https://docs.sidewalk.amazon/introduction/sidewalk-gateways.html