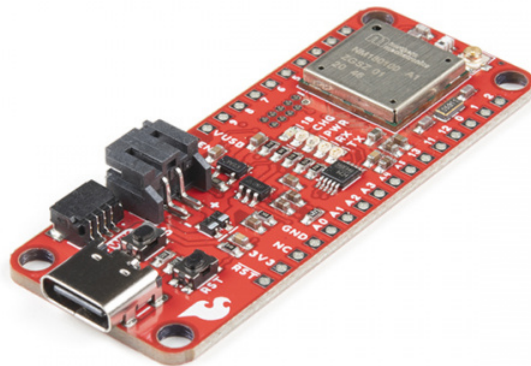


# SparkFun expLoRaBLE Hookup Guide

## Introduction

The SparkFun LoRa Thing Plus – expLoRaBLE is a feather form-factor development board with the NM180100 system in package (SiP), from Northern Mechatronics. The NM180100 SiP includes a Semtech SX1262 LoRa module paired with the Apollo3 MCU, which is used in the SparkFun Artemis module. This provides the board with compatibility in the Arduino IDE, through our Apollo3 Arduino core. The Semtech SX1262 is a long range, low power RF transceiver introduces LoRaWAN capabilities to the board. Additionally, both the BLE and LoRa capabilities of the expLoRaBLE allow it to operate as a Bluetooth enabled LoRa node.



SparkFun LoRa Thing Plus - expLoRaBLE

© WRL-17506

Product Showcase: SparkFun LoRa Thing Plus - ExpLoRaBLE



## Required Materials

To get started, users will need a few items. Now some users may have a few of these items, feel free to modify your cart accordingly.

- SparkFun LoRa Thing Plus – expLoRaBLE
- USB 3.1 Cable A to C - 3 Foot - The USB interface serves two purposes: it powers the board and allows you to upload programs to it. (*\*If your computer doesn't provide a USB-A slot, then you will need to choose an appropriate cable or purchase an adapter as well.*)
- Wide Band 4G LTE Internal FPC Antenna
- Access to a LoRa gateway (*connected to The Things Network*)
- An account with The Things Network
- Computer with the an operating system (OS) that is compatible with all the software installation requirements.



SparkFun LoRa Thing Plus - expLoRaBLE

🕒 WRL-17506



USB 3.1 Cable A to C - 3 Foot

🕒 CAB-14743



Wide Band 4G LTE Internal LoRa Antenna

🕒 WRL-17841

GATEWAYS

ANTENNAS

HEADERS

BATTERIES

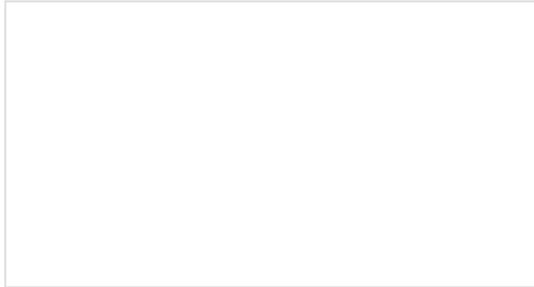
JUMPER MODIFICATION

JTAG FUNCTIONALITY

*Click the buttons above to toggle the **additional materials** based on the options you wish to use. Feel free to modify the items in your cart to fit your needs.*

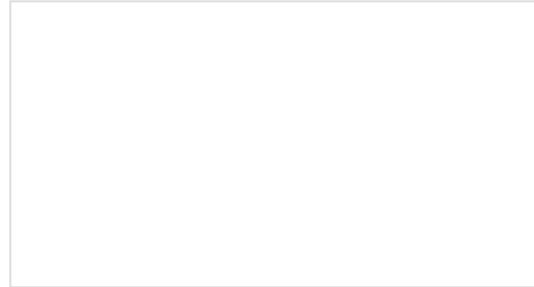
## Suggested Reading

As a more professionally oriented product, we will skip over the more fundamental tutorials (i.e. **Ohm's Law** and **What is Electricity?**). However, below are a few tutorials that may help users familiarize themselves with various aspects of the board.



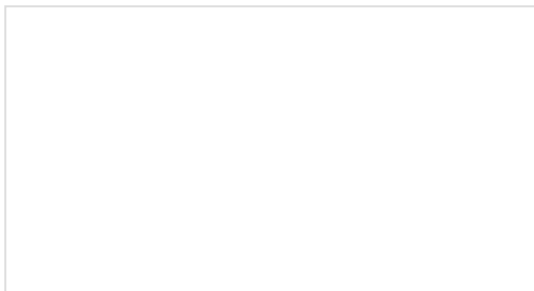
### Serial Communication

Asynchronous serial communication concepts: packets, signal levels, baud rates, UARTs and more!



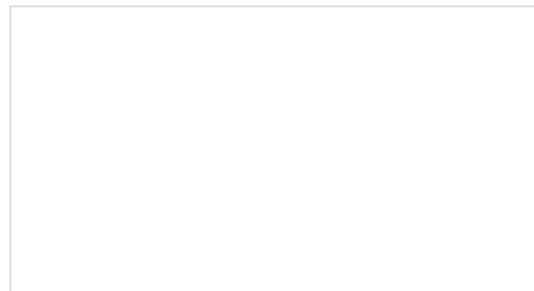
### I2C

An introduction to I2C, one of the main embedded communications protocols in use today.



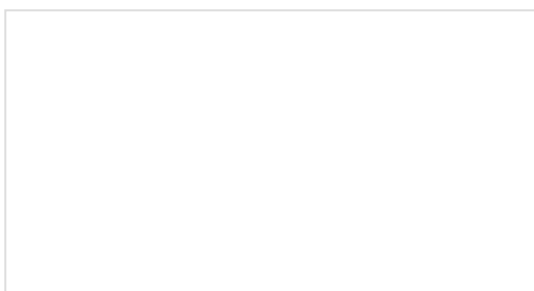
### Bluetooth Basics

An overview of the Bluetooth wireless technology.



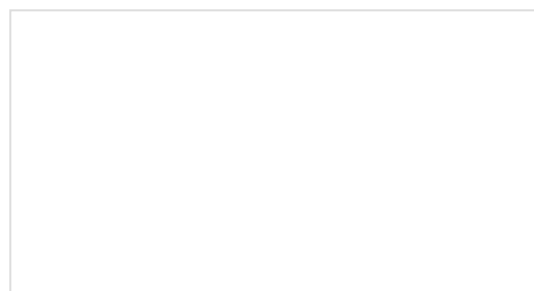
### Logic Levels

Learn the difference between 3.3V and 5V devices and logic levels.



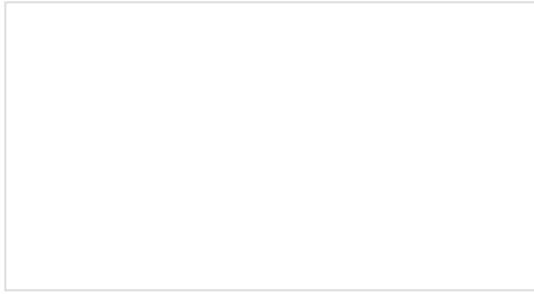
### Serial Peripheral Interface (SPI)

SPI is commonly used to connect microcontrollers to peripherals such as sensors, shift registers, and SD cards.



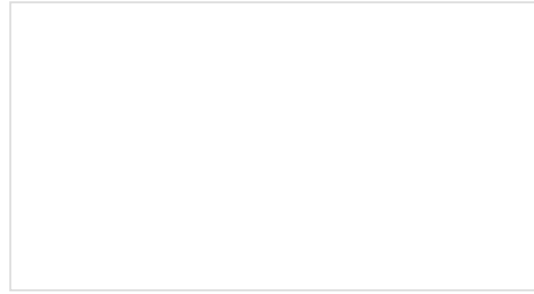
### ARM Programming

How to program SAMD21 or SAMD51 boards (or other ARM processors).



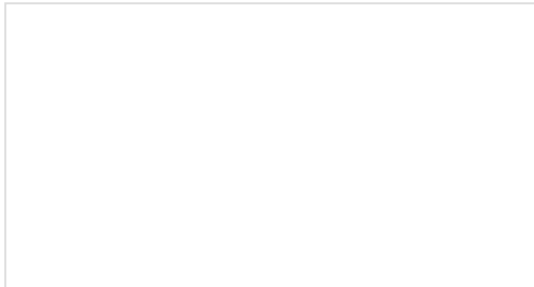
### Installing an Arduino Library

How do I install a custom Arduino library? It's easy! This tutorial will go over how to install an Arduino library using the Arduino Library Manager. For libraries not linked with the Arduino IDE, we will also go over manually installing an Arduino library.



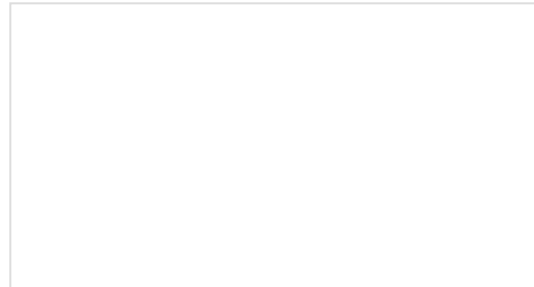
### Three Quick Tips About Using U.FL

Quick tips regarding how to connect, protect, and disconnect U.FL connectors.



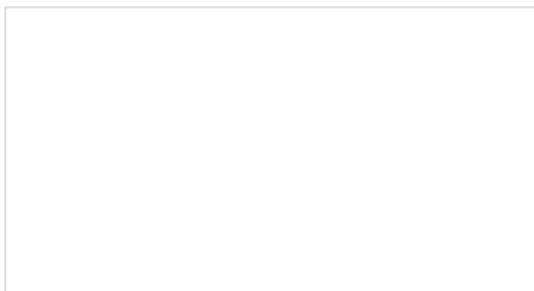
### LoRaWAN with ProRF and The Things Network

Learn how to make a LoRaWAN node for your next long range IoT project and connect it to the internet with The Things Network!



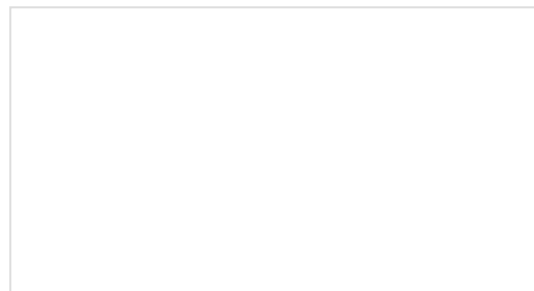
### Artemis Development with Arduino

Get our powerful Artemis based boards (Artemis Nano, BlackBoard Artemis, and BlackBoard Artemis ATP) blinking in less than 5 minutes using the SparkFun Artemis Arduino Core!



### Artemis Development with the Arduino IDE

This is an in-depth guide on developing in the Arduino IDE for the Artemis module and any Artemis microcontroller development board. Inside, users will find setup instructions and simple examples from blinking an LED and taking ADC measurements; to more complex features like BLE and I2C.



### Pulse Width Modulation

An introduction to the concept of Pulse Width Modulation.





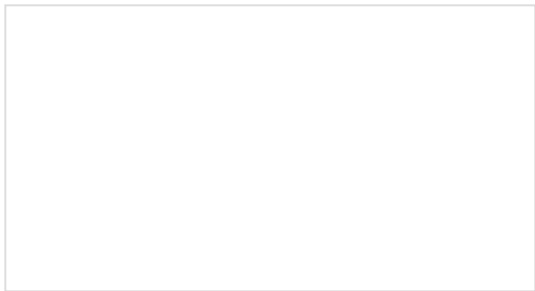
### Analog vs. Digital

This tutorial covers the concept of analog and digital signals, as they relate to electronics.



### Installing Board Definitions in the Arduino IDE

How do I install a custom Arduino board/core? It's easy! This tutorial will go over how to install an Arduino board definition using the Arduino Board Manager. We will also go over manually installing third-party cores, such as the board definitions required for many of the SparkFun development boards.



### Installing Arduino IDE

A step-by-step guide to installing and testing the Arduino software on Windows, Mac, and Linux.



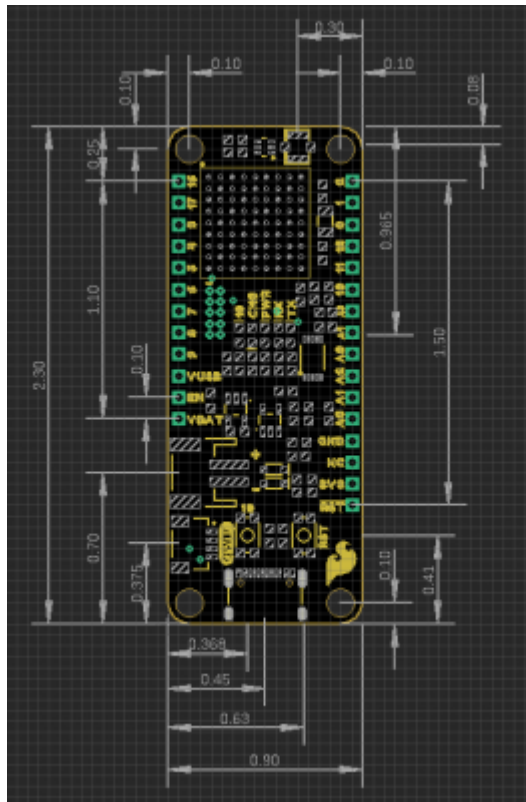
One of the new, advanced features of the board is that it takes advantage of the Qwiic connect system. We recommend familiarizing yourself with the **Logic Levels** and **I<sup>2</sup>C** tutorials. Click on the banner above to learn more about Qwiic products.



## Hardware Overview

### Board Dimensions

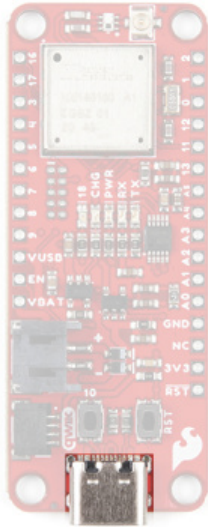
The board dimensions are illustrated in the drawing below. The listed measurements are in inches and the four mounting holes are compatible with 4-40 standoff screws.



*Board dimensions (PDF) for the SparkFun expLoRaBLE, in inches. (Click to enlarge)*

### USB-C Connector

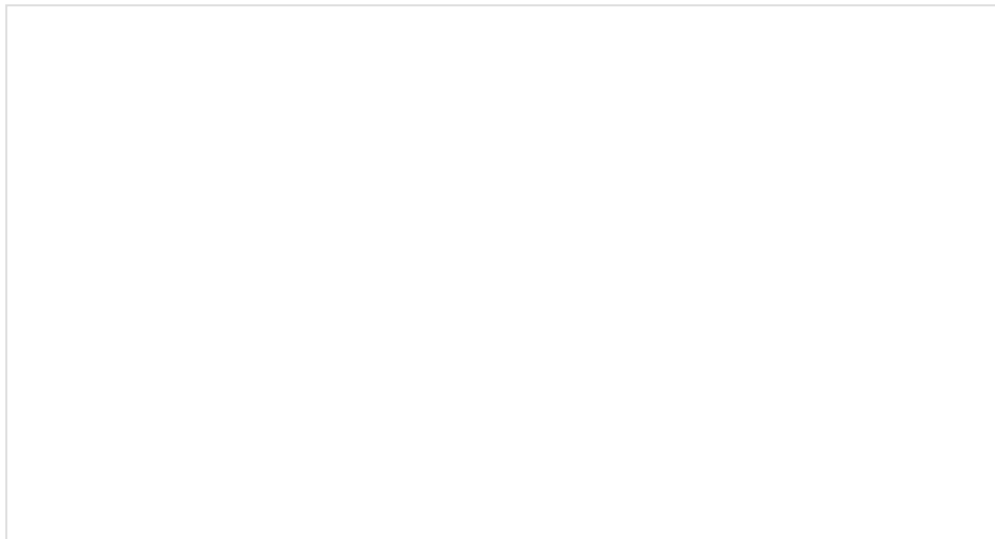
The USB connector is provided to power and program the board. For most users, it will be the primary programming interface for the NM180100.



*USB-C connector on the SparkFun expLoRaBLE. (Click to enlarge)*

## CH340E Serial-to-UART

The CH340E allows the NM180100 to communicate with a computer/host device through its USB-C connection. This allows the board to show up as a device on the serial (or COM) port of the computer. Users will need to install the latest CH340 driver for the computer to recognize the board.



## How to Install CH340 Drivers

AUGUST 6, 2019

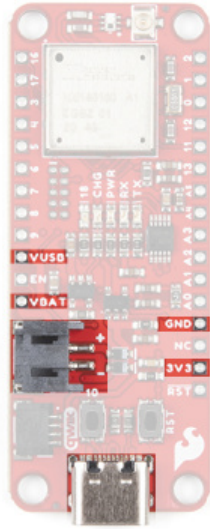
How to install CH340 drivers (if you need them) on Windows, Mac OS X, and Linux.

## Power

The SparkFun expLoRaBLE only requires 3.3V to power the board. However, the simplest method to power the board is with the USB-C connector. There are additional power pins available on the board:

- **3.3V** - A regulated 3.3V voltage source.
  - Regulated from the USB 5V power and/or battery connection.

- Used to power the NM180100 SiP and Qwiic I<sup>2</sup>C bus.
- **USB** - The voltage from the USB-C connector, usually 5V.
- **VBAT** - The voltage from the JST battery connector; meant for single cell LiPo batteries.
- **GND** - The common ground or the 0V reference for the voltage supplies.



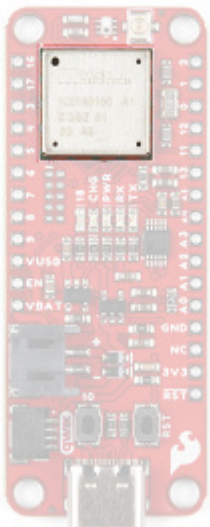
*SparkFun expLoRaBLE power connections. (Click to enlarge)*

## Charging Circuit

The charging circuit utilizes the MCP73831 linear charge management controller and is powered directly from the USB-C connector or USB. The controller is configured for a **500mA** charge rate and battery charging is indicated when the yellow, CHG LED. If the charge controller is shutdown or charging is complete, the CHG LED will turn off. For more information, please refer to the MCP73831 datasheet.

## NM180100 SiP

The NM180100 SiP from Northern Mechatronics is the brains of the SparkFun expLoRaBLE. The system in package (SiP) includes an Apollo3 MCU connected to a Semtech SX1262 radio transceiver. The connection between the two ICs is laid out in the table below.



*NM180100 SiP on the SparkFun expLoRaBLE. (Click to enlarge)*



Apollo3		SX1262		Description
Pin	Name	Pin	Name	
H6	GPIO 36	19	NSS	SPI Slave Select
J6	GPIO 38	17	MOSI	SPI Slave Input
J5	GPIO 43	16	MISO	SPI Slave Output
H5	GPIO 42	18	SCK	SPI Clock Input
J8	GPIO 39	14	Busy	Radio Busy Indicator
J9	GPIO 40	13	DIO1	Multipurpose Digital I/o
H9	GPIO 47	06	DIO3	Multipurpose Digital I/o
J7	GPIO 44	15	NRESET	Radio Reset Signal ( <i>Active Low</i> )

A breakdown of the NM180100 functionality is shown in the block diagram below and the functionality of each embedded chip is laid out in the following subsections. It should be noted, that the Bluetooth and LoRa antenna connections on the SparkFun expLoRaBLE share the same u.FL connector with the use of a diplexer. For more details on the NM180100 SiP, check out the datasheet.

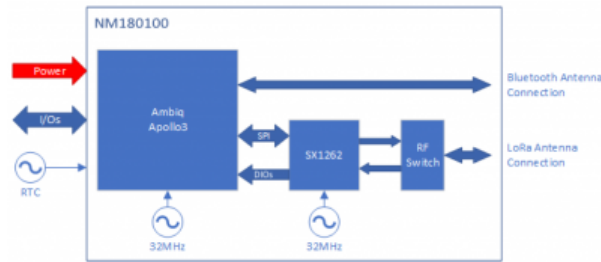


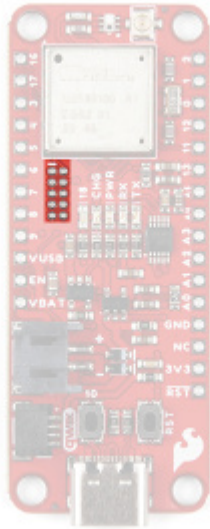
Figure 2. NM180100 block diagram.

**Note:** While most users will utilize the USB connection for serial programming, the Apollo3 MCU in the NM180100 SiP can also be programmed through its JTAG or SWD pins. This might be useful for individuals developing and testing firmware that would be flashed directly onto the NM180100 SiP, such as in production for commercial applications. For more details on programming, please check out our ARM Programming tutorial

## ARM Programming

MAY 23, 2019

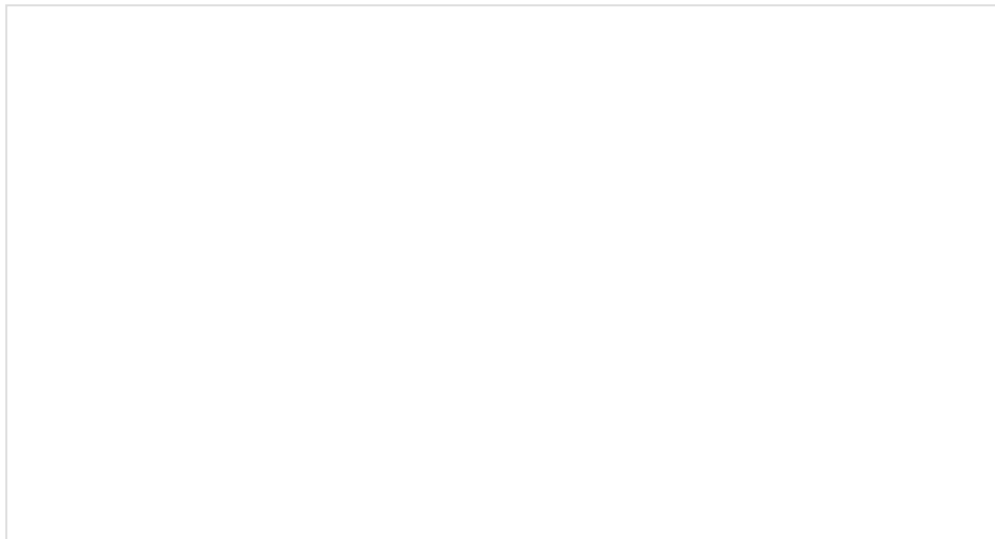
How to program SAMD21 or SAMD51 boards (or other ARM processors).



*The JTAG pins on the SparkFun expLoRaBLE. (Click to enlarge)*

## Apollo3 MCU

For details on the Apollo3, users should refer to the Designing with the SparkFun Artemis hookup guide. Additionally, users can reference the following resources for more technical information:



## Designing with the SparkFun Artemis

JUNE 20, 2019

Let's chat about layout and design considerations when using the Artemis module.

## Hardware Information:

- Ambiq MCU Product Page
  - Apollo3 DataSheet (PDF)
- Artemis GitHub Repository
  - Artemis Bootloader GitHub

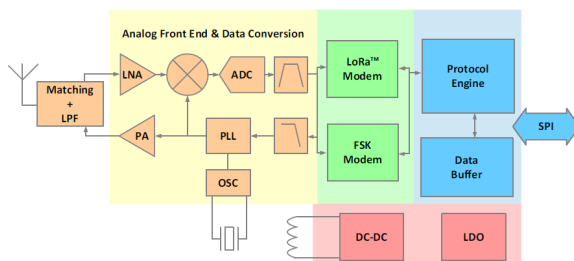
## Development Platforms:

- SparkFun Ambiq Apollo3 Arduino Core
- AmbiqSDK
- pyOCD

## SX1262 RF Transceiver

**Note:** In the US, LoRaWAN operates in the 915MHz frequency band (*i.e.* 902 to 928 MHz).

The Semtech SX1262 module is a long range, low power, half-duplex transceiver with global LoRa® frequency coverage, capable of operating as a Long range (LoRa) wide area network (LoRaWAN) or frequency-shift keying (FSK) modem. The module was designed for long battery life with just **4.2 mA** of active receive current consumption and it can transmit up to **+22 dBm** with the use of its highly efficient, integrated power amplifiers. The SX1262 is optimal for devices that are designed to comply with the physical layer requirements of the LoRaWAN specification released by the LoRa Alliance™. For more details on the SX1262, check out the datasheet.



Functional block diagram for the SX1262. (Click to enlarge)

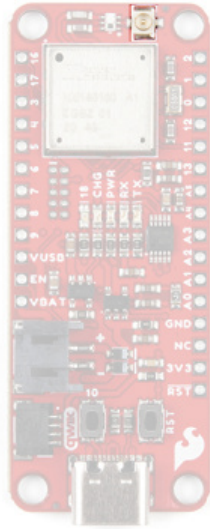
Characteristic	Description
Current Consumption	4.2 to 10.1 mA (RX) 32 to 118 mA (TX)
Frequency Range	150 to 960 MHz
Modulation	FSK, GFSK, MSK, GMSK and LoRa
Link Budget	170 dB (max)
Bit Rate (programmable)	FSK: .6 to 300 kb/s LoRa: .018 to 62.5 kb/s
RF Sensitivity	-104 to -148 dBm

RF Output Power	+14 to +22 dBm
-----------------	----------------

## u.FL Antenna Connector

**Note:** Make sure to utilize an antenna that is appropriate for your RF and/or BLE application.

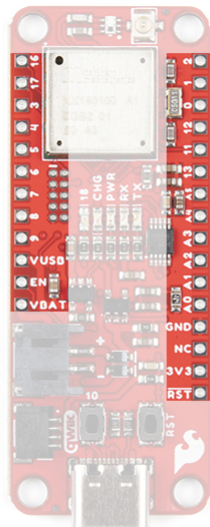
The Apollo3 BLE and SX1262 RF antenna connections share a single U.FL antenna connector with the use of a diplexer.



*Antenna connection on the SparkFun expLoRaBLE. (Click to enlarge)*

## Breakout Pin Connections

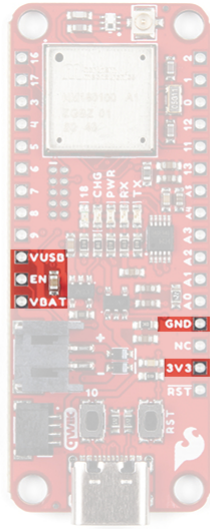
The pins from the NM180100 SiP are broken out into a feather form factor layout.



*SparkFun expLoRaBLE breakout pins. (Click to enlarge)*

## Power Pins

The power pins aren't really **I/O** (Input/Output) connections for the microcontroller; however, they are pertinent to the board.



*SparkFun expLoRaBLE power pins. (Click to enlarge)*

The power I/O mostly consists of voltage supply pins. These pins are traditionally **used as power sources** for other pieces of hardware (like LEDs, potentiometers, and other circuits).

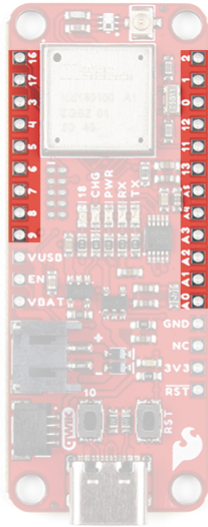
- **3.3V** - A regulated 3.3V voltage source.
  - Regulated from the USB 5V power and/or battery connection.
  - Used to power the NM180100 SiP and Qwiic I<sup>2</sup>C bus.
- **USB** - The voltage from the USB-C connector, usually 5V.
- **VBAT** - The voltage from the JST battery connector; meant for single cell LiPo batteries.
- **GND** - The common ground or the 0V reference for the voltage supplies.

**Note:** The **EN** pin is used to control the 3.3V voltage regulator output (*i.e. the boards power*) from the battery or USB connection to the rest of the board.

## I/O Pins

**⚡ Note:** All the GPIO on the SparkFun LoRa Thing Plus – expLoRaBLE are **3.3V** pins.

There are **21 I/O pins** broken out on this board, which can be used as digital **inputs** to or **outputs** from the NM180100 SiP.



*I/O breakout pins and headers. (Click to enlarge)*

All of the SparkFun expLoRaBLE pins are broken out with .1" pitch spacing for headers. It is best practice to define the `pinMode()` ([link](#)) in the setup of each **sketch** (programs written in the Arduino IDE) for the pins used.

### Input

When configured properly, an **input** pin will be looking for a **HIGH** or **LOW** state. **Input** pins are **High Impedance** and takes very little current to move the input pin from one state to another.

### Output

When configured as an **output** the pin will be at a **HIGH** or **LOW** voltage. **Output** pins are **Low Impedance**: This means that they can provide a relatively substantial amount of current to other circuits.

**⚡ Note:** It should be noted that there are electrical limitations to the amount of current that the NM180100 SiP can sink or source. For more details, check out the **Absolute Maximum Ratings** section of the **Electrical Characteristics** on page 774 of the Apollo3 datasheet.

### Additional Functions

There are several pins that have special functionality in addition to general **digital I/O**. These pins and their additional functions are listed in the tabs below. For more technical specifications on the **I/O** pins, you can refer to the Apollo 3 datasheet.

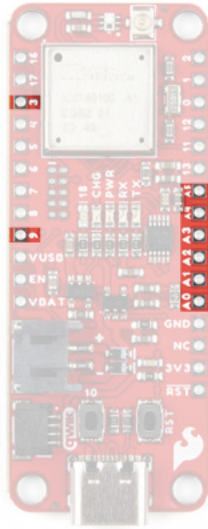
<b>ANALOG INPUT</b>	PWM OUTPUT	SERIAL COMM.	SPI	I <sup>2</sup> C
---------------------	------------	--------------	-----	------------------

### Analog Input Pins

**Note:** Be aware that the ADC input range is from **0 - 2V**. Although connecting a sensor with an output to 3.3V is safe for the NM180100 SiP, it will saturate the ADC at 2V.

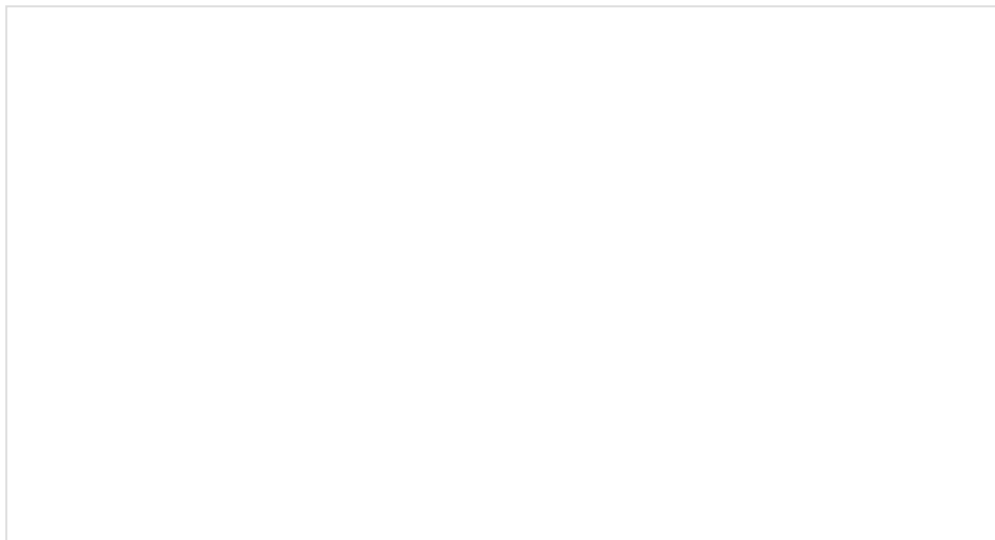
The NM180100 SiP offers a **14-bit ADC** input for eight of the SparkFun expLoRaBLE's I/O pins. This functionality is accessed in the Arduino IDE using the `analogRead(pin)` function.

**Note:** By default, in the Arduino IDE, `analogRead()` returns a 10-bit value. To change the resolution of the value returned by the `analogRead()` function, use the `analogReadResolution(bits)` function.



*Analog input pins on the SparkFun expLoRaBLE. (Click to enlarge)*

**Note:** To learn more about analog vs. digital signals, check out this great tutorial.



## Analog vs. Digital

JULY 18, 2013

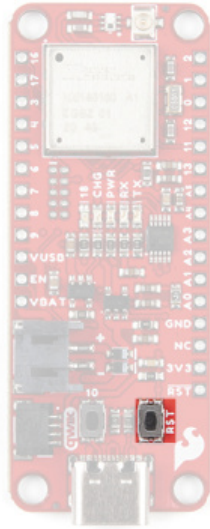
This tutorial covers the concept of analog and digital signals, as they relate to electronics.

## Buttons

There are two buttons on SparkFun expLoRaBLE; a reset and GPIO button.

## Reset Button

The reset ( **RST** ) button allows users to reset the program running on the NM180100 SiP without unplugging the board.



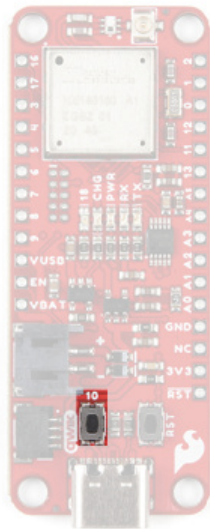
*Reset button on the SparkFun expLoRaBLE. (Click to enlarge)*

## User Button

**Note:** In order to utilize the user button, connected to GPIO 10 , the pin mode will need to be configured as an input with an internal pullup (i.e. `INPUT_PULLUP` ); see an example below.

```
pinMode(D10, INPUT_PULLUP);
```

The user ( **10** ) button allows users to short GPIO 10 to ground ( **GND** ).



*GPIO 10 button on the SparkFun expLoRaBLE. (Click to enlarge)*

## Indicator LEDs

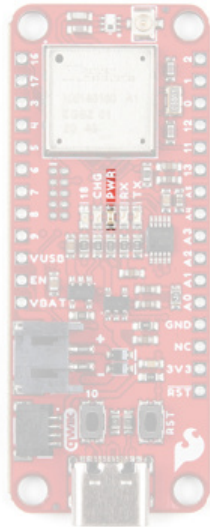


There are five indication LEDs on the SparkFun expLoRaBLE:

- Status/Pin 18 (Blue)
- Power (Red)
- Battery Charging (Yellow)
- RX (Yellow) and TX (Green)

### Power LED

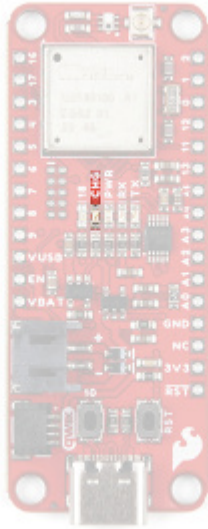
The red, **PWR** LED will light up once **3.3V** is supplied to the board. For most users, it will light up when **5V** is supplied through the USB connection and/or when a LiPo battery is attached to the JST connector.



*SparkFun expLoRaBLE Thing Plus PWR status LED indicator. (Click to enlarge)*

### Battery Charging LED

The yellow, **CHG** LED will light while a battery is being charged through the charging circuit. The LED will be off when no battery is present (*\*dimmed*), when the charge management controller is in standby (*after the battery charging has been completed*), or when the charge management controller is shutdown. The LED will be on when the charge management controller is in the process of charging the battery. For more information, please refer to the MCP73831 datasheet.

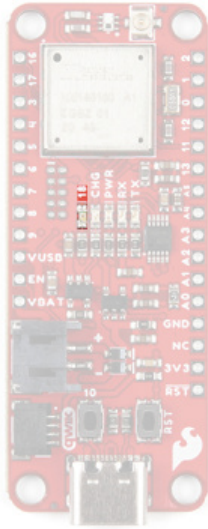


The battery charging ( CHG ) LED indicator on the SparkFun expLoRaBLE Thing Plus. (Click to enlarge)

Charge Cycle State	STAT1
Shutdown <ul style="list-style-type: none"> <li>• Thermal Shutdown</li> <li>• <math>V_{DD} &lt; V_{BAT}</math></li> </ul>	<b>Off</b> (High Z)
No Battery Present	<b>Dimmed</b> (High Z)
Charge Complete – Standby	<b>Off</b> (H)
Preconditioning	<b>On</b> (L)
Constant-Current Fast Charge	<b>On</b> (L)
Constant Voltage	<b>On</b> (L)

## STAT LED

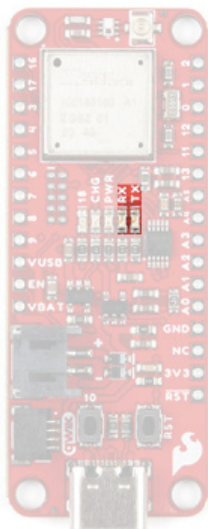
The blue, status ( 18 ) LED is typically used as a test or status LED to make sure that a board is working or for basic debugging. This indicator is connected to GPIO 18 .



*The status ( 18 ) LED indicator on the SparkFun expLoRaBLE Thing Plus. (Click to enlarge)*

## RX and TX LEDs

The yellow, RX and green, TX LEDs are used to indicate serial communication and programming between the CH340E serial-to-UART and NM180100 SiP.



*The RX and TX LED indicators on the SparkFun expLoRaBLE Thing Plus. (Click to enlarge)*

## Jumpers

There are three jumpers on the back of the board that can be used to easily modify the hardware connections on the board.

- **ISO** - This jumper can be used to isolate the 3.3V connection from the voltage regulator.
- **LED** - This jumper can be used to remove 3.3V power to the PWR LED and reduce power consumption on the board.
- **RTS** - This jumper can be used to disconnect the CH340E from the RESET pin of the Apollo3; effectively, disabling the ability of the CH340E to reset the MCU through software.

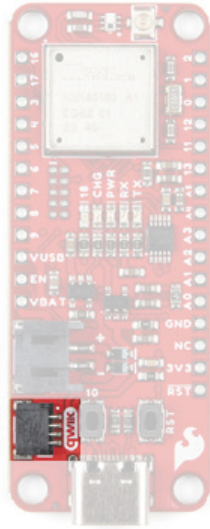
## Primary I<sup>2</sup>C Bus

The Qwiic connector is attached to the primary I<sup>2</sup>C bus. The primary I<sup>2</sup>C bus for this board utilizes the pin connections, detailed in the table below:

Connection	VDD	GND	SCL	SDA
<b>Pad Number (NM180100 SiP)</b>	<b>3.3V</b>	<b>GND</b>	D15	D14

## Qwiic Connector

A Qwiic connector is provided for users to seamlessly integrate with SparkFun's Qwiic Ecosystem.



*Qwiic connector on the SparkFun expLoRaBLE. (Click to enlarge)*

## What is Qwiic?

The Qwiic system is intended a quick, hassle-free cabling/connector system for I<sup>2</sup>C devices. Qwiic is actually a play on words between "quick" and I<sup>2</sup>C or "iic".

## SparkFun's Qwiic Connect System



## Features of the Qwiic System

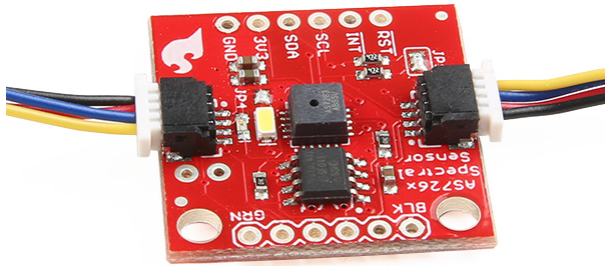
**NO SOLDERING**

POLARIZED CONNECTOR

DAISY CHAIN

Keep your soldering iron at bay.

Cables plug easily between boards making quick work of setting up a new prototype. We currently offer three different lengths of Qwiic cables as well as a breadboard friendly cable to connect any Qwiic enabled board to anything else. Initially you may need to solder headers onto the shield to connect your platform to the Qwiic system but once that's done it's plug and go!

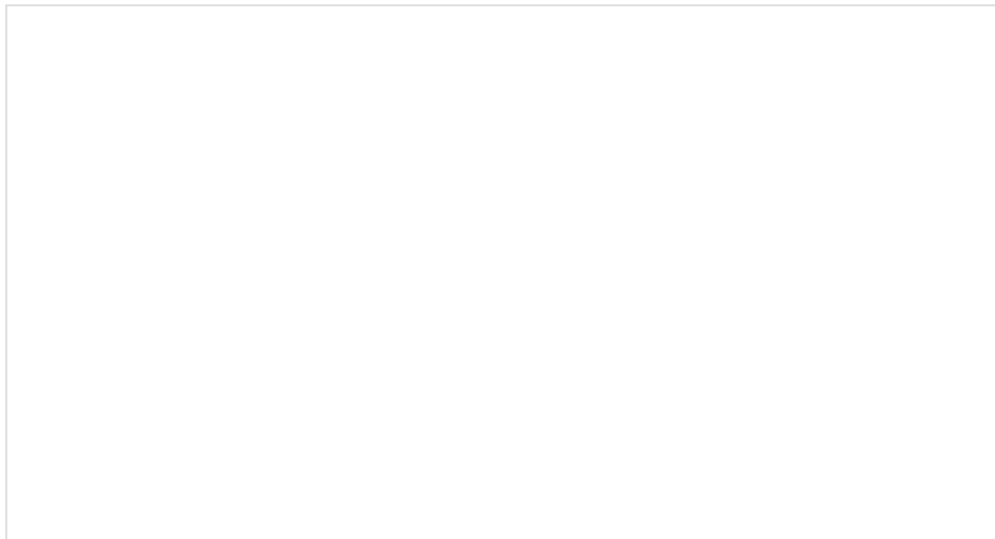


Qwiic cables connected to Spectral Sensor Breakout

## Software Overview

### CH340 Driver

Users will need to install the latest CH340 driver on their computer, in order for it to properly recognize the board.



### How to Install CH340 Drivers

AUGUST 6, 2019

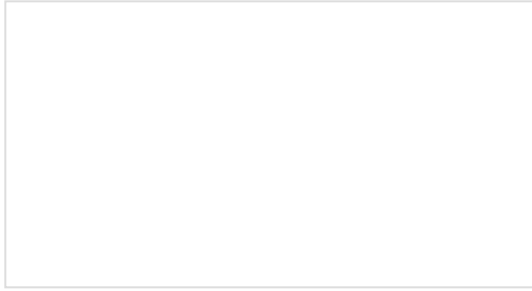
How to install CH340 drivers (if you need them) on Windows, Mac OS X, and Linux.

### Arduino IDE

**Note:** For first-time users, who have never programmed before and are looking to use the Arduino IDE, we recommend beginning with the SparkFun Inventor's Kit (SIK), which includes a simpler board like the Arduino Uno or SparkFun RedBoard and is designed to help users get started programming with the Arduino IDE.

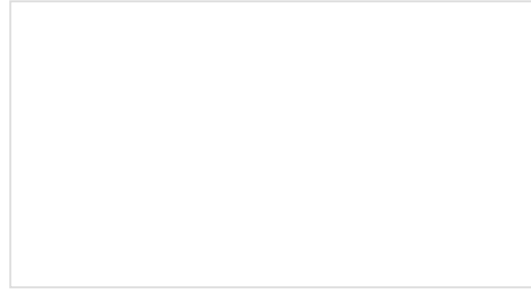
Most users will be familiar with the Arduino IDE and its use. As a point of reference for professional developers who aren't aware, the Arduino IDE is an open-source development environment, written in Java, that makes it easy to write code and upload it to a supported board. For more details, feel free to check out the Arduino website.

To get started with the Arduino IDE, check out the following tutorials:



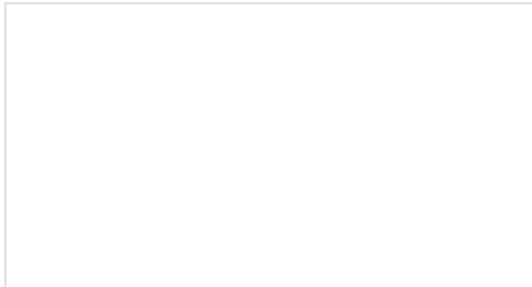
### Installing an Arduino Library

How do I install a custom Arduino library? It's easy! This tutorial will go over how to install an Arduino library using the Arduino Library Manager. For libraries not linked with the Arduino IDE, we will also go over manually installing an Arduino library.



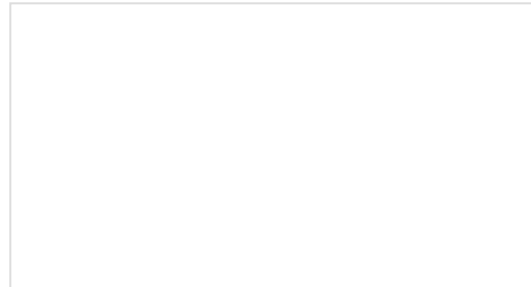
### What is an Arduino?

What is this 'Arduino' thing anyway? This tutorial dives into what an Arduino is and along with Arduino projects and widgets.



### Installing Arduino IDE

A step-by-step guide to installing and testing the Arduino software on Windows, Mac, and Linux.



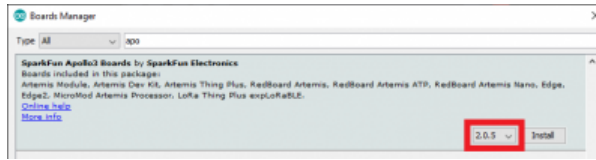
### Installing Board Definitions in the Arduino IDE

How do I install a custom Arduino board/core? It's easy! This tutorial will go over how to install an Arduino board definition using the Arduino Board Manager. We will also go over manually installing third-party cores, such as the board definitions required for many of the SparkFun development boards.

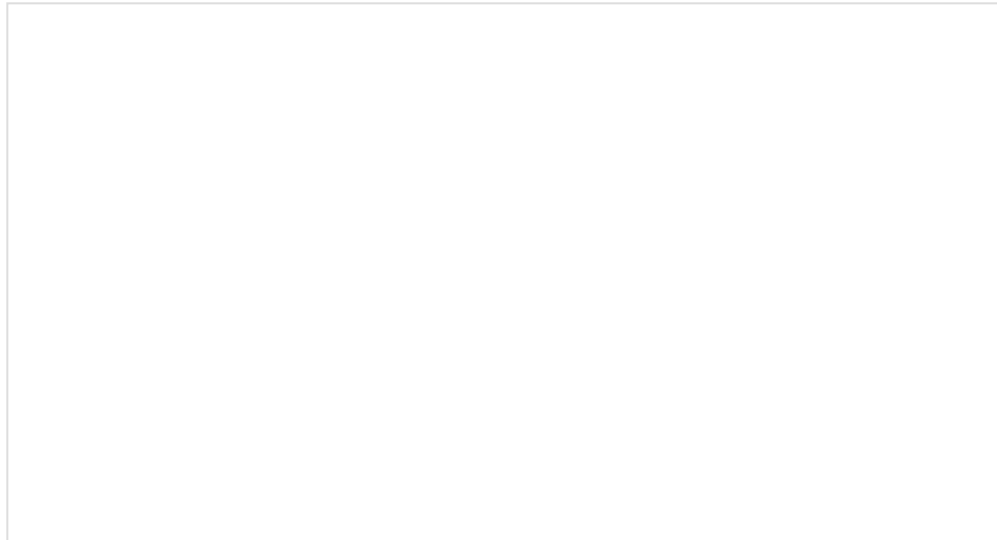
## Software Dependencies

- Install the latest **SparkFun Apollo3 Boards** board definitions in the Arduino IDE

**Note:** The installed *board definitions* for our Apollo3 development boards must be version 2.0.5 or later.



Adding the SparkFun *explORaBLE* in the **Board Manager**. (Click to enlarge)



## Installing Board Definitions in the Arduino IDE

SEPTEMBER 9, 2020

How do I install a custom Arduino board/core? It's easy! This tutorial will go over how to install an Arduino board definition using the Arduino Board Manager. We will also go over manually installing third-party cores, such as the board definitions required for many of the SparkFun development boards.

**Note:** For more instructions, users can follow this tutorial on Installing Additional Cores provided by Arduino. Users will also need the `.json` file for the SparkFun Ambiq Apollo3 Arduino Core:

[https://raw.githubusercontent.com/sparkfun/Arduino\\_Apollo3/master/package\\_sparkfun\\_apollo3\\_index](https://raw.githubusercontent.com/sparkfun/Arduino_Apollo3/master/package_sparkfun_apollo3_index)

- Install the associated Arduino libraries for the board's features:
  1. The **ArduinoBLE** Arduino library
  2. The **RadioLib** Arduino library

**Note:** This section has been updated to no longer use the modified RadioLib library. As mentioned in the pull request for our changes, users only need to declare that the SPI1 bus is used.

- Provides support for peer-to-peer RF communication

**Note:** Peer-to-peer refers to direct RF communication using FSK modulation and not a mesh network LoRa integration.

**DOWNLOAD THE RADIOLIB ARDUINO LIBRARY**

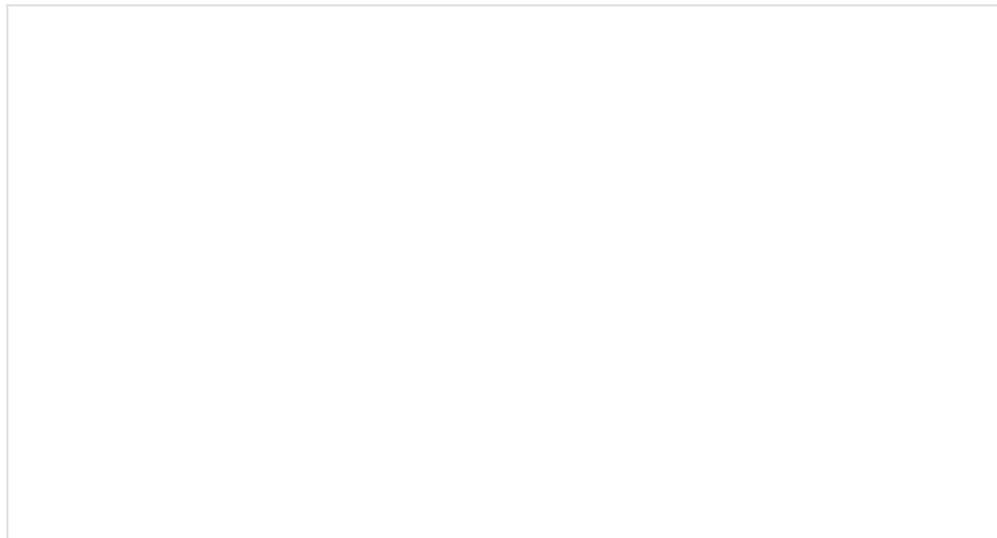
### 3. Our port of the **basicmac** code (*\*requires manual .zip file installation*)

**Note:** Currently, there isn't an Arduino library that provides a LoRa implementation, supporting the SX1262. However, we were able to modify and port the **basicmac** code as an Arduino library for the SparkFun expLoRaBLE. Users will need to manually install the library with this .zip file.

#### **DOWNLOAD THE PORTED BASICMAC ARDUINO LIBRARY**

The library includes modifications for the SPI pins use by the NM180100 to connect the Apollo3 MCU to the SX1262 module. Additionally, it also contains modifications to pre-configure the library to use the SX1262, the US LoRa frequency band, and defaults to a SF7 (spreading factor).

- Provides support for LoRa on The Things Network
- GitHub repository for **basicmac** (*original code*)



## Installing an Arduino Library

JANUARY 11, 2013

How do I install a custom Arduino library? It's easy! This tutorial will go over how to install an Arduino library using the Arduino Library Manager. For libraries not linked with the Arduino IDE, we will also go over manually installing an Arduino library.

## The Things Network

Users will need an account on The Things Network for the example that demonstrates the LoRaWAN capabilities of the expLoRaBLE. Please, use the information in either of the following tutorials to create an account on register a node on The Things Network.



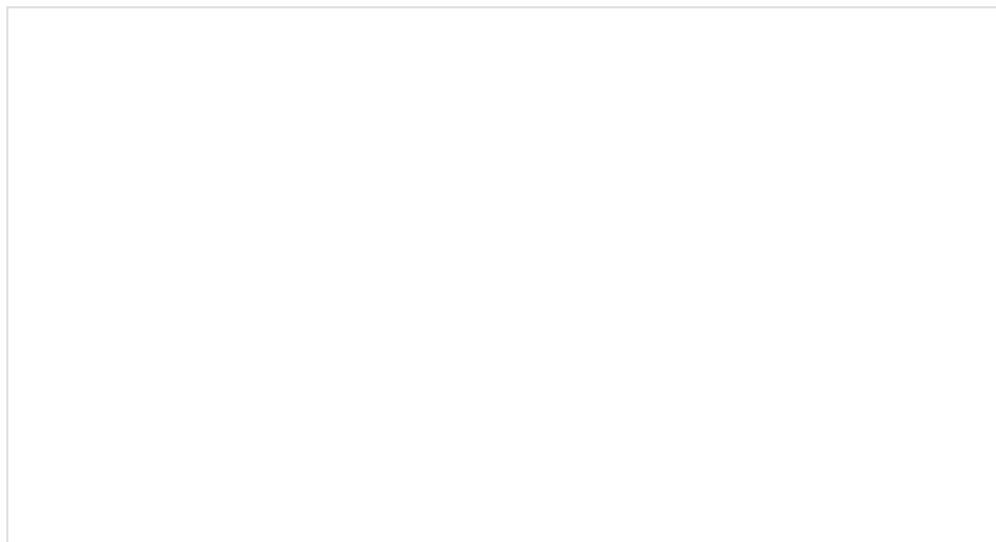




## LoRaWAN with ProRF and The Things Network

JULY 3, 2018

Learn how to make a LoRaWAN node for your next long range IoT project and connect it to the internet with The Things Network!



## SparkFun SAMD21 Pro RF Hookup Guide

OCTOBER 4, 2018

Using the super blazing, nay blinding, fast SAMD21 whipping clock cycles at 48MHz and the RFM96 module to connect to the Things Network (and other Radio woodles).

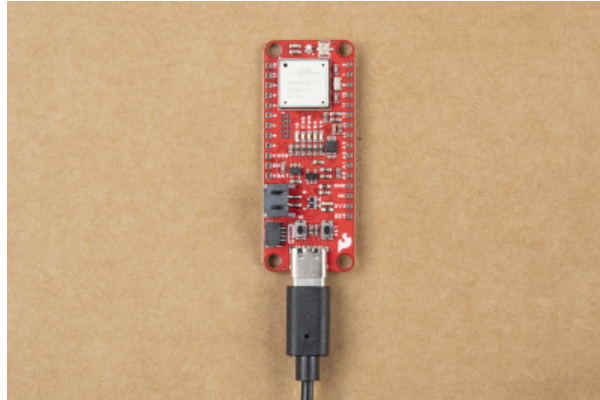
**Note:** For more information, feel free to utilize the documentation from The Things Network to:

1. Create or Register an Account
2. Add an Application through the Console
3. Register a Device in the Application
4. Once registered, in the device settings:
  - Configure the activation method to ABP (activation by personalization)
  - Disable (uncheck) the **Frame counter checks**
5. Learn about Working with Bytes

## Hardware Assembly

### USB Programming

The USB connection is utilized for programming and serial communication. Users only need to plug their SparkFun expLoRaBLE into a computer using a USB-C cable.

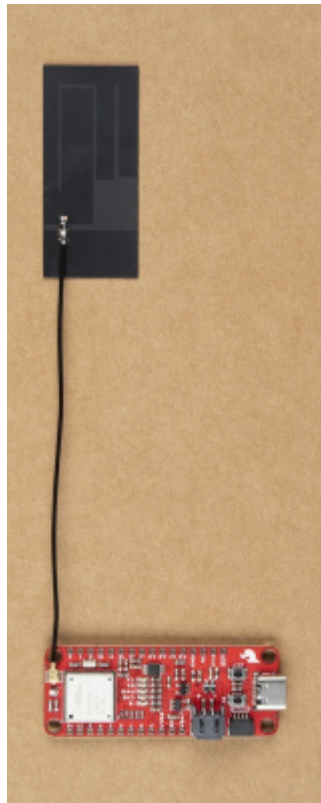


*The SparkFun expLoRaBLE attached to a computer. (Click to enlarge)*

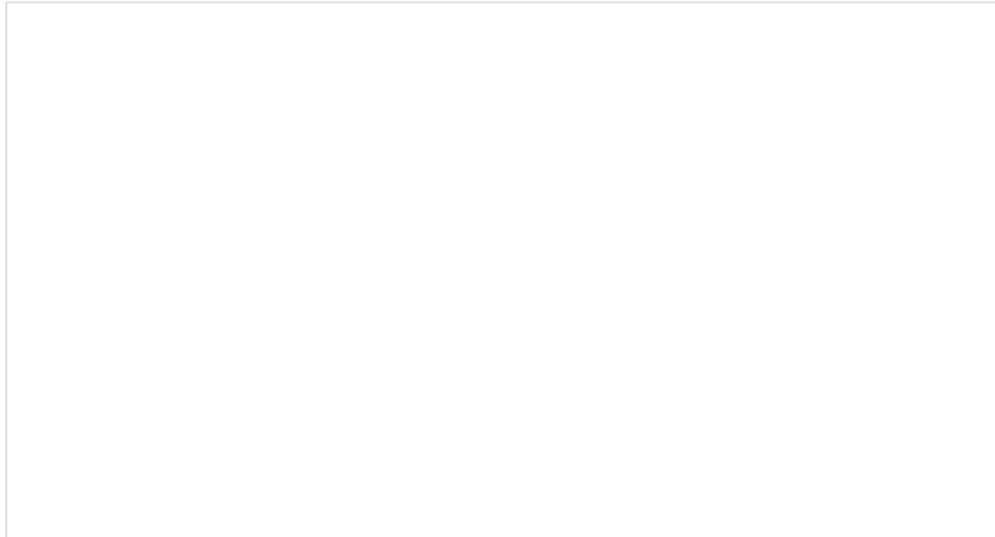
### Antenna

In order to utilize the LoRa and RF capabilities of the SX1262, users will need to attach an antenna. Check out the tutorial below for tips on using U.FL connectors.





The SparkFun expLoRaBLE with a whip or patch antenna attached. (Click to enlarge)



## Three Quick Tips About Using U.FL

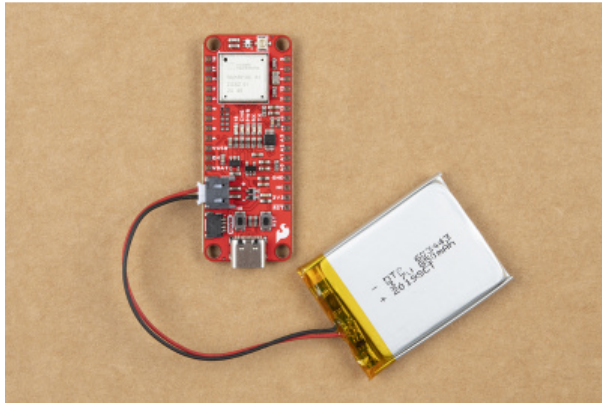
DECEMBER 28, 2018

Quick tips regarding how to connect, protect, and disconnect U.FL connectors.

**Note:** Users should be aware that the transmission signal from the Wide Band 4G LTE Internal FPC antenna emits orthogonally to the top surface (*i.e. towards the camera in the image above*). Additionally, from our experience the antenna runs better in the wire down orientation as well.

### Battery

For remote LoRa applications, a Li-Po battery can be connected. Additionally, users may be interested in utilizing a solar panel and USB-C cable to recharge their battery.



*The SparkFun expLoRaBLE with a battery connected. (Click to enlarge)*



**Solar Panel Charger - 10W**  
© PRT-16835

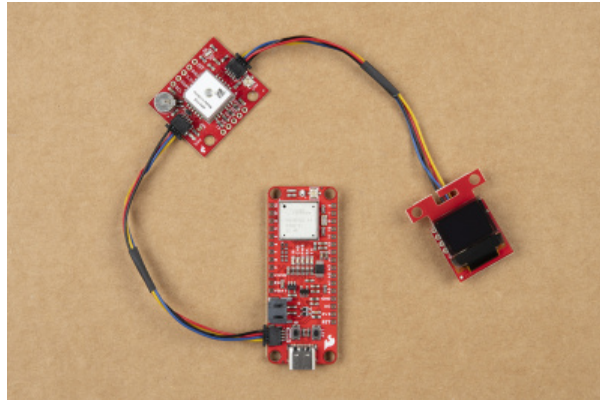


# USB 3.1 Cable A to C - 3 Foot

© CAB-14743

## Qwiic Devices

The Qwiic system allows users to effortlessly prototype with a Qwiic compatible I<sup>2</sup>C device without soldering. Users can attach any Qwiic compatible sensor or board, with just a Qwiic cable. (*\*The example below, is for demonstration purposes and is not pertinent to the board functionality or this tutorial.\**)



The Qwiic XA110 GPS breakout board and Micro OLED board connected to the SparkFun expLoRaBLE.

## LoRaWAN Example

### **LEGAL IMPLICATIONS!!!**

**Regional Configuration:** The (*ported*) **basicmac** library includes configurations required for the SPI pins use by the NM180100 to connect the Apollo3 MCU to the SX1262 module. Additionally, it also contains modifications to pre-configure the library to use the SX1262, the **US LoRa frequency band**, and defaults to a spreading factor of 7 (SF7).

In order to change the regional configuration (**for outside the US**), users will need to go into the library files to make the necessary modifications to the target configuration file ( `target-conf.h` ). The configuration file can be found with the following file path `BasicMAC>src>hal>target-config.h` ; on Windos 10, the file path is `C:\Users\"username"\Documents\Arduino\libraries\BasicMAC\src\hal\target-config.h` .

```
C target-config.h X
C:\Users > > Documents > Arduino > libraries > BasicMAC > src > hal > C target-config.h > ...
32
33 // This defines the region(s) to use. You can enable more than one and
34 // then select the right region at runtime using os_getRegion() and/or
35 // LMIC_reset_ex().
36 #if defined(CFG_eu868) && defined(CFG_us915)
37 // #define CFG_eu868 1
38 #define CFG_us915 1
39 // #define CFG_as923 1
40 // #define CFG_11915 1
41 // #define CFG_kr920 1
42 // #define CFG_au915 1
43 #endif // !defined(CFG_eu868) && !defined(CFG_us915)
44
```

Lines **36-43** in the `target-conf.h` file used to configure the regional settings. (*Click to enlarge*)

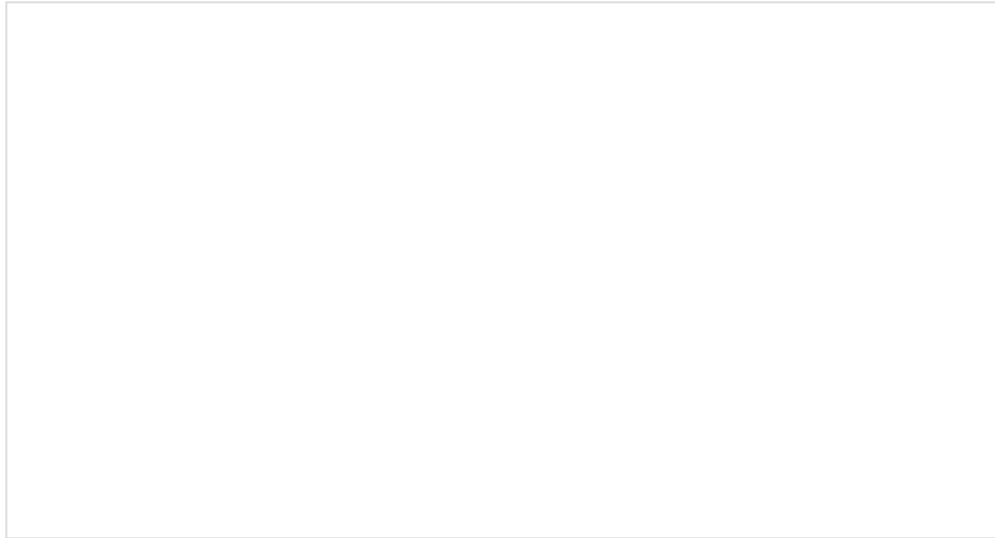
On lines **36-43**, users will find the available regional options. To make the required modifications, users simply need to comment in/out their configuration choice and save the file, before uploading their code. (*\*It is recommended that users also close and re-open the Arduino IDE before uploading their code, to ensure the modifications are in place.\**)

## Register a Device on The Things Network

In order to utilize this example, users will need access to a gateway that is connected to The Things Network. Additionally, users should have their own account on The Things Network and have registered a device for their node. A device must be registered, in order to receive the *Network Session Key*, *App Session Key*, and *Device Address* required for the SparkFun expLoRaBLE to operate as a node and have its data passed to The Things Network servers. For those who are unfamiliar with LoRaWAN and The Things Network, please refer to the information provided below.

### **Registering a Device on The Things Network:**

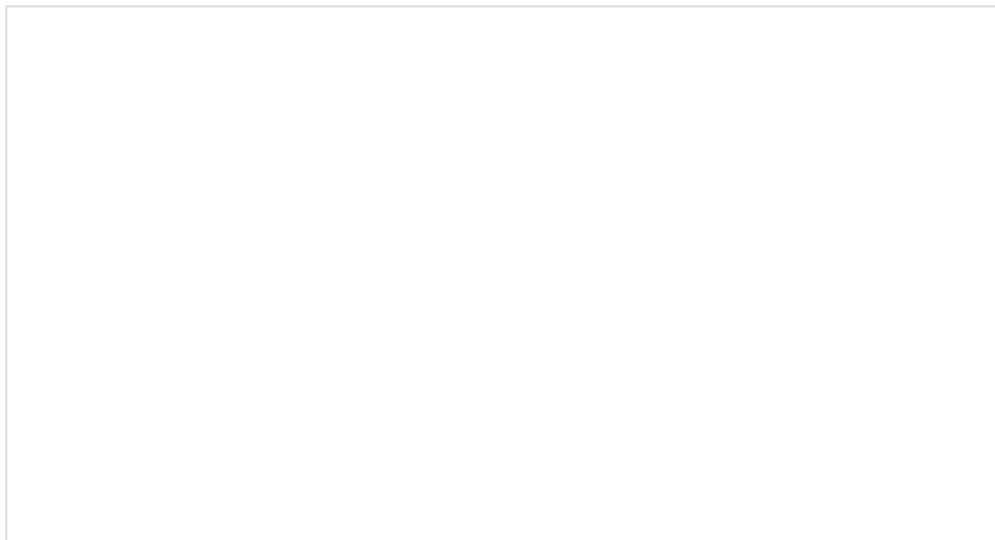
Users will need an account on The Things Network for this example. Please, use the information in either of the following tutorials to create an account and register a device on The Things Network.



### **LoRaWAN with ProRF and The Things Network**

JULY 3, 2018

Learn how to make a LoRaWAN node for your next long range IoT project and connect it to the internet with The Things Network!





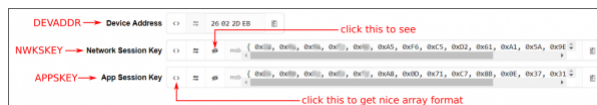
### **SparkFun SAMD21 Pro RF Hookup Guide**

OCTOBER 4, 2018

Using the super blazing, nay blinding, fast SAMD21 whipping clock cycles at 48MHz and the RFM96 module to connect to the Things Network (and other Radio woodles).

### **Credentials:**

Once a device has been registered (*\*the device activation method should be configure to **ABP***), the **Network Session Key**, **App Session Key**, and **Device Address** are displayed on the **Device Overview** page. In order for the expLoRaBLE to have its data passed to The Things Network, the *Network Session Key*, *App Session Key*, and *Device Address* must be hard coded on the device. By default, the *Network Session Key* and *App Session Key* fields are obscured for security reasons. Click the  icon to show the code and click the  button to show the codes in a C-style format. Below, is a diagram explaining which field on the **Device Overview** page corresponds to the variables in the provided example code.



A screen shot from the **Device Overview** page with labels. (Click to enlarge)

### Additional Resources

For more information, feel free to utilize the documentation from The Things Network to:

1. Create or Register an Account
2. Add an Application through the Console
3. Register a Device in the Application
4. Once registered, in the device settings:
  - o Configure the activation method to ABP (activation by personalization)
  - o Disable (uncheck) the **Frame counter checks**
5. Learn about Working with Bytes

### LoRa Example Code

Once users have a device registered on The Things Network, the example code will need to be downloaded and a few modifications are required before uploading.

[DOWNLOAD THE LORAWAN EXAMPLE CODE](#)

### Credentials

Lines **36-47** of the example code, need to be modified to hard code the credentials required for the payload from the node to be passed to The Things Network:

```

// LoRaWAN NwkSKey, network session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
static const PROGMEM u1_t NWKSKEY[16] = { 0x##, 0x##, 0x##, 0x##, 0x##, 0x##, 0x##, 0x##, 0x##,
0x##, 0x##, 0x##, 0x##, 0x##, 0x##, 0x## };

// LoRaWAN AppSKey, application session key
// This is the default Semtech key, which is used by the early prototype TTN
// network.
static const u1_t PROGMEM APPSKEY[16] = { 0x##, 0x##, 0x##, 0x##, 0x##, 0x##, 0x##, 0x##, 0x##,
0x##, 0x##, 0x##, 0x##, 0x##, 0x##, 0x## };

// LoRaWAN end-device address (DevAddr)
static const u4_t DEVADDR = 0x##### ; // <-- Change this address for every node! For example
, our device address is 26022DEN. We will need to replace "DEVICE_ADDRESS_HERE" as 0x26022DEB.

```

## Transmission Interval

Lines **62-64** of the example code, are used to set the transmission intervals between payloads:

```

// Schedule TX every this many milliseconds (might become longer due to duty
// cycle limitations).
const unsigned TX_INTERVAL = 20000;

```

## Payload Data

Lines **329-336** of the example code, is where the contents of the payload can be modified. The contents of the transmission on can be viewed on The Things Network through the **Application Data** page, once the node begins broadcasting. To access the **Application Data** page, click on the `data` tab from the **Device Overview** page on The Things Network console.

```

void send_packet() {
  // Prepare upstream data transmission at the next possible time.
  uint8_t mydata[] = "<Enter Text>";
  LMIC_setTxData2(1, mydata, sizeof(mydata) - 1, 0);
  Serial.println(F("Packet queued"));

  last_packet = millis();
}

```

**Note:** The data is limited to **13 bytes** in size. The data ( `mydata[]` ) can take 3 different formats: text (as shown above), integers (*8-bit unsigned*), or an array. Any data sent as an array, must be in the following format: `{ 0x01, 0x02, 0x03, ... up to ... 0x13 }` (*i.e. up to 13 hexadecimal numbers in an array*).

## Before You Upload the Code

### **STOP!!!**

**Regional Configuration:** Before uploading the code, be sure to make any necessary regional configuration changes, as mentioned at the beginning of this section, to **avoid breaking any laws**.



**Note:** For those utilizing our LoRa Gateway - 1-Channel (ESP32), the code on lines 288-300 will need to be uncommented. The modification forces the device/node to only use channel 8; instead of randomly picking a channel to broadcast on. This is important as the single channel gateway only receives transmissions on channel 8 and if the device were broadcasting on channels randomly, only a portion of the transmissions would reach The Things Network.

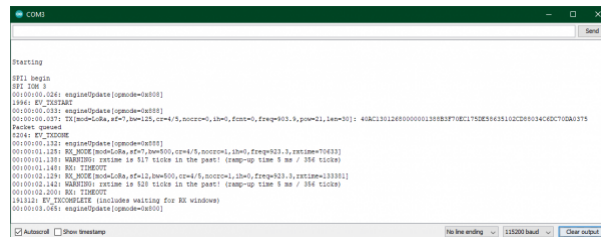
```
// disable channels 0-7
for (int i = 0; i <= 7 ; i++)
{
  LMIC_disableChannel(i);
}

// note, leave channel 8 enabled

// disable all other higher channels
for (int i = 9; i <= 63 ; i++)
{
  LMIC_disableChannel(i);
}
```

## Code in Operation

Once the example code has been modified and uploaded to the SparkFun expLoRaBLE, users can monitor the serial terminal (or SerialMonitor on the Arduino IDE) for successful transmissions. Additionally, users can use The Things Network to decode the data packets to verify the data transmission.



```
STARTING
RFI DONE
000000.026: rxFinalState (opmode=TX00)
19464: RF_TX_START
000000.033: rxFinalState (opmode=TX00)
000000.037: TX (mod=LoRa, sf=7, bw=125, cr=4/5, sccor=0, ldm=0, fcs=0, freq=903.3, pwr=21, lse=0) | 48421312600000128883F70E175D55563102205034C0C70A0375
Packet queued
52044: RF_TX_DONE
000000.123: rxFinalState (opmode=TX00)
000001.125: RF_NACK (mod=LoRa, sf=7, bw=125, cr=4/5, sccor=0, ldm=0, fcs=0, freq=903.3, rxTime=75633)
000001.129: NABORT() reason is 817 ticks in the past! (sleep-up time 5 ms / 354 ticks)
000001.148: RF_TX_START
000002.129: RF_NACK (mod=LoRa, sf=7, bw=125, cr=4/5, sccor=0, ldm=0, fcs=0, freq=903.3, rxTime=13381)
000002.142: NABORT() reason is 828 ticks in the past! (sleep-up time 5 ms / 354 ticks)
000002.200: RF_TX_START
181321: RF_TXCOMPLETE (includes waiting for RX window)
000003.045: rxFinalState (opmode=TX00)
```

Example of serial output from successful transmissions. (Click to enlarge)

## Mapping with Node

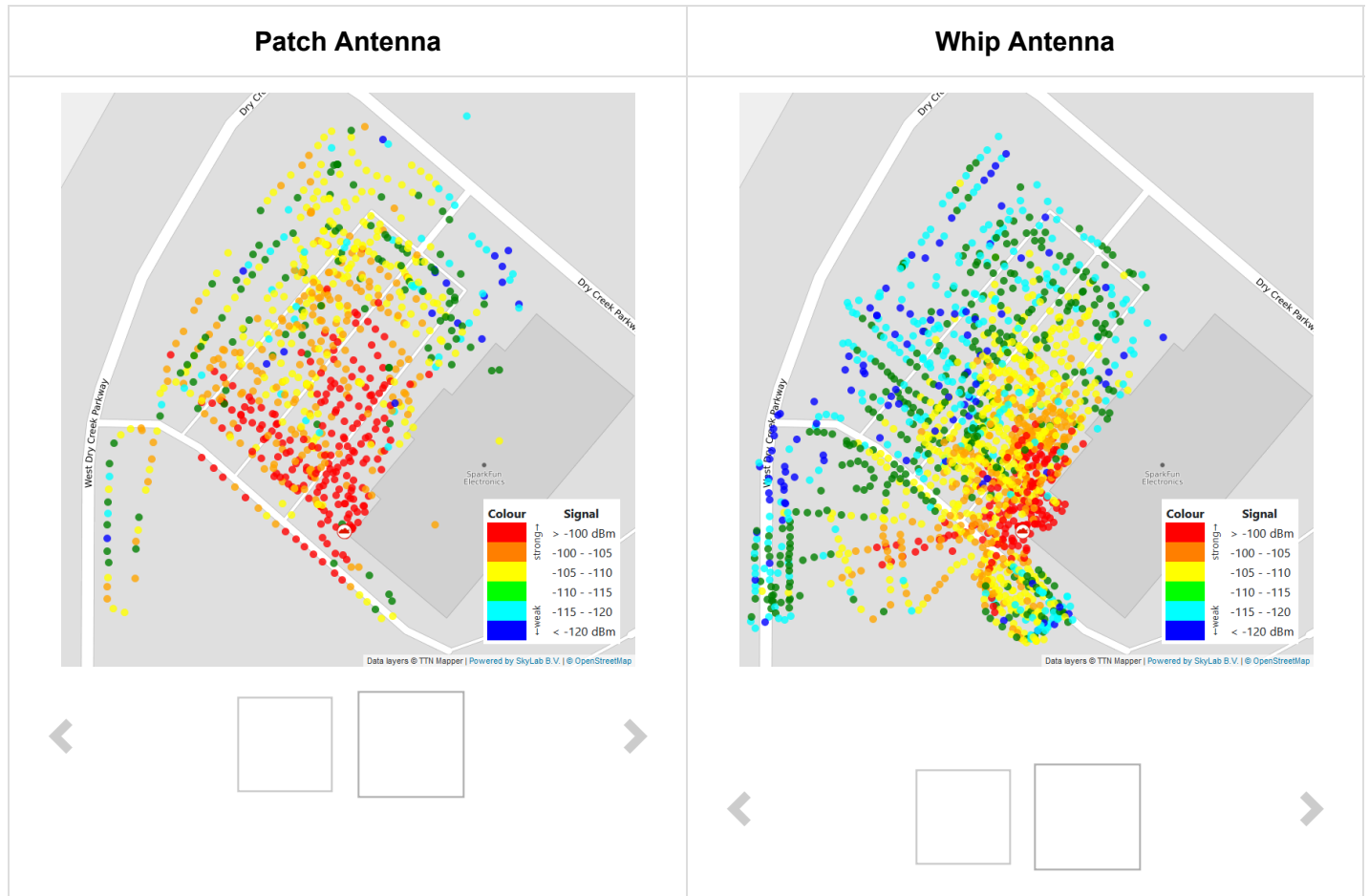
For those looking to test out the range of their setup configuration, The Things Network has an application called TTN Mapper. To get started follow the Mapping Gateway Coverage using a Things Node. There is also an option to map using a either an Android phone or iPhone. For more information on the app, please check out the documentation on The Things Network.

To generate the images below, the Android guide was followed. The steps were relatively simple:

1. Download the App
2. Link a Device by:
  1. Logging-in to The Things Network
  2. Select an Application
  3. Select Device
3. Power on Device

#### 4. Walk Around

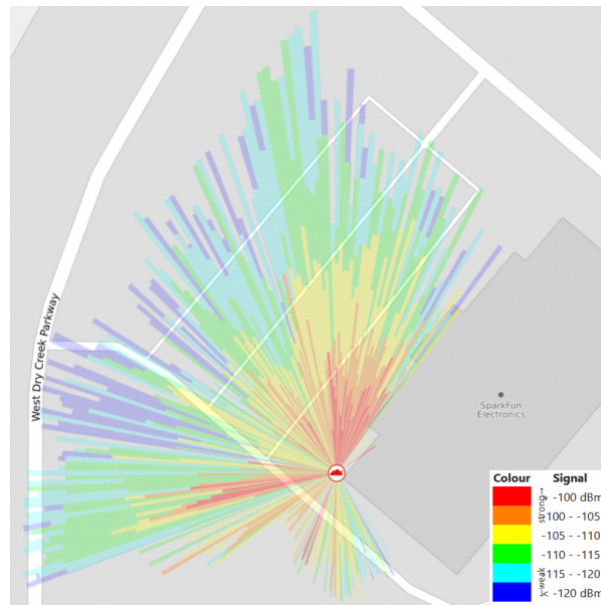
The **TTN Mapper** app uses the phone's positioning and ties it to the data transmission from the linked device. Therefore, users will need to enable their phone's data and location for the application to work properly.



*Example mapping of the signal coverage, using a spreading factor of 7 ( SF7 ), for two different antennas.*

**Note:** To maximize the range of the device, the spreading factor should be changed to **SF12**. However, this modification will reduce the bitrate; and therefore, lowers the data rate; and therefore, minimizes the packet size that can be transmitted.

You can even publish your data online, to get a *radar-like* coverage map for a gateway:



A radar, coverage map for a gateway. (Click to enlarge)

There were some vehicles parked to the northwest and northwest-by-north of where the gateway was positioned; as well as a tree and post to the immediate west and west-by-north. The image illustrates how the signal strength to the west drops off significantly, due to the proximity of the trees and poles. Additionally, how the signal strength in the northwest gradually falls from interference by the vehicles. The signal coverage to the south of the gateway was spotty since it was around the corner of the building.

## Peer-to-Peer Example

**Note:** This section has been updated to no longer use the modified RadioLib library. As mentioned in the pull request for our changes, users only need to declare that the SPI1 bus is used.

**Important:** Users should also use the example codes that we provide with the download buttons below. There seem to be several users, trying to use the built-in examples and running into issues, even though we provide example codes below. The example codes below have been written and tested for users to be able to upload directly to their expLoRaBLE and execute without any issues. *(Obviously because we are based in the US, the example code has been written for use at 915MHz (the reserved frequency band for our region). Users outside the US and using a different frequency band, will need to reference the library and modify the code on their own.)*

In this example, the **RadioLib** library utilizes the frequency shift keying (FSK) capabilities of the SX1262 transceiver. For this example, users will need two expLoRaBLEs; one to transmit data and the other to receive that data. For more information on this library, check out the GitHub repository Wiki. Users simply need to download and upload the code below to separate expLoRaBLEs:

**DOWNLOAD THE TRANSMITTER EXAMPLE CODE**

**DOWNLOAD THE RECEIVER EXAMPLE CODE**

Once the example codes have been uploaded to their respective boards, users should see the data transmissions in the Serial Monitor:



- SFE Product Showcase

Need some inspiration for your next project? Check out some of these related tutorials:

**ARDUINO IDE**    **BOARDS & SHIELDS**    **BOARD FUNCTIONALITY**

**SOFTWARE DEVELOPMENT GUIDES**

## SparkFun Tutorials

### Installing an Arduino Library

How do I install a custom Arduino library? It's easy!

This tutorial will go over how to install an Arduino library using the Arduino Library Manager. For libraries not linked with the Arduino IDE, we will also go over manually installing an Arduino library.

### What is an Arduino?

What is this 'Arduino' thing anyway? This tutorial dives into what an Arduino is and along with Arduino projects and widgets.

### Installing Arduino IDE

A step-by-step guide to installing and testing the Arduino software on Windows, Mac, and Linux.

### Installing Board Definitions in the Arduino IDE

How do I install a custom Arduino board/core? It's easy! This tutorial will go over how to install an Arduino board definition using the Arduino Board Manager. We will also go over manually installing third-party cores, such as the board definitions required for many of the SparkFun development boards.

## Arduino Tutorials

- Getting Started > Introduction: What is Arduino and what I can use it for?
- Getting Started with Arduino and Genuino products

- [Arduino Software \(IDE\)](#)
- [Arduino Troubleshooting](#)
- [Arduino: Contact Us](#)