MicroMod nRF52840 Processor Hookup Guide

Introduction

The MicroMod nRF52840 Processor offers a powerful SoC (system on chip) combination of ARM Cortex-M4 CPU and 2.4 GHz Bluetooth transceiver in the MicroMod form-factor so you can easily plug it in to the Carrier Board of your choice. The specific nRF module on this Processor board features an integrated PCB antenna for the Bluetooth transceiver.

SparkFun MicroMod nRF52840 Processor

WRL-16984

Product Showcase: SparkFun MicroMod Weather Carrier & nRF5...

The nRF52840 module includes a BT 5.1 stack and supports Bluetooth 5, Bluetooth mesh, IEEE 802.15.4 (Zigbee & Thread) and 2.4GHz RF wireless protocols (including Nordic's proprietary RF protocol) allowing you to pick which option works best for your application.

On top of the processing power and Bluetooth capability of the nRF52840, this board features two I^2C buses, 2 SPI buses, eleven GPIO, dedicated digital, analog, and PWM pins along with multiple serial UARTs to cover all your peripheral needs.

Required Materials

Along with your MicroMod nRF52840 Processor, obviously need a Carrier Board to plug your Processor into. SparkFun offers a variety of MicroMod Carrier Boards to fit your project. Below you can see a few options:

SparkFun MicroMod Weather Carrier Board
SEN-16794

SparkFun MicroMod Input and Display Carrier Board
DEV-16985

SparkFun MicroMod Data Logging Carrier Board
DEV-16829

SparkFun MicroMod ATP Carrier Board
DEV-16885

You'll also need a USB-C cable to connect the Carrier Board to your computer and if you want to add some Qwiic breakouts to your MicroMod project you'll want at least one Qwiic cable to connect it all together. Below are some options for both of those cables:

SparkFun Qwiic Cable Kit
KIT-15081

Flexible Qwiic Cable - 100mm
PRT-17259

USB 3.1 Cable A to C - 3 Foot
CAB-14743

Reversible USB A to C Cable - 2m
CAB-15424

Depending on which Carrier Board you choose, you may need a few extra peripherals to take full advantage of them. Refer to your Carrier Board's Hookup Guide for specific peripheral recommendations.

Suggested Reading

The SparkFun MicroMod ecosystem offers a unique way to allow users to customize their project to their needs. Do you want to communicate with your MicroMod circuit via a wireless signal (e.g. Bluetooth or WiFi)? There's a MicroMod Processor for that. Looking to instead maximize efficiency and processing power? You guessed it, there's a MicroMod Processor for that. If you are not familiar with the MicroMod ecosystem, take a look here:

MicroMod

MicroMod Ecosystem

We also recommend reading through the following tutorials if you are not familiar with the concepts covered in them:

Serial Communication

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

Serial Peripheral Interface (SPI)

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

Installing Arduino IDE

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

Logic Levels

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

I2C

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

Bluetooth Basics

An overview of the Bluetooth wireless technology.

Hexadecimal

How to interpret hex numbers, and how to convert them to/from decimal and binary.

Getting Started with MicroMod

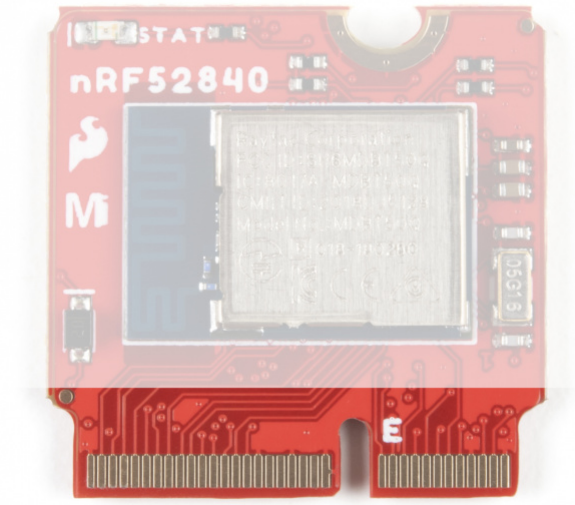
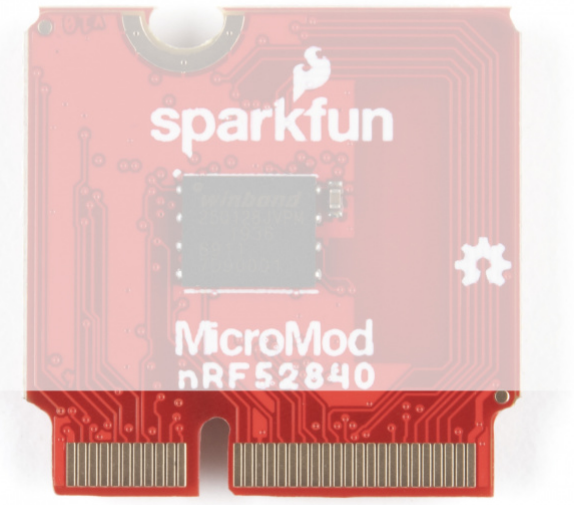
Dive into the world of MicroMod - a compact interface to connect a microcontroller to various peripherals via the M.2 Connector!

Hardware Overview

In this section we'll cover the nRF52840 SoC in more detail as well as other components included on the MicroMod nRF52840 Processor.

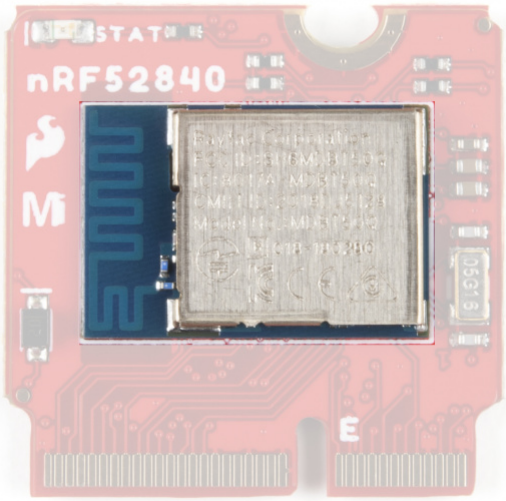
M.2 Connector

All of our MicroMod Processors come equipped with the **M.2 MicroMod Connector**, which leverages the M.2 standard and specification to allow you to install your MicroMod Processor board on your choice of Carrier Board.

	
M2 Connector from the Front	*M2 Connector from the Back*

nRF52840 MDBT50Q Module

This Processor features the nRF52840 SoC from Nordic Semiconductor. The specific module is from Raytac which includes a built-in PCB antenna. Refer to the MDBT50Q Module Datasheet and nRF52840 IC Datasheet for full specifications on the IC and module.

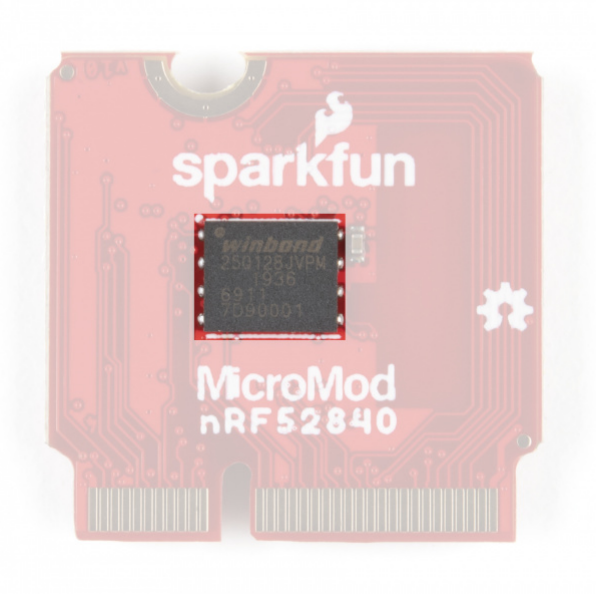


The nRF52840 combines an ARM Cortex-M4 CPU and 2.4 GHz Bluetooth transceiver to provide a versatile and powerful microcontroller with Bluetooth 5.0 wireless capability. The transceiver also works with Bluetooth mesh, IEEE 802.15.4(Thread & Zigbee), ANT and Nordic's proprietary 2.5 GHz RF protocol to communicate with other Nordic devices.

The nRF52840 Processor board uses the Arduino Nano 33 BLE Bootloader as opposed to the USB Bootloader found on the SparkFun Pro nRF52840 Mini.

Flash IC

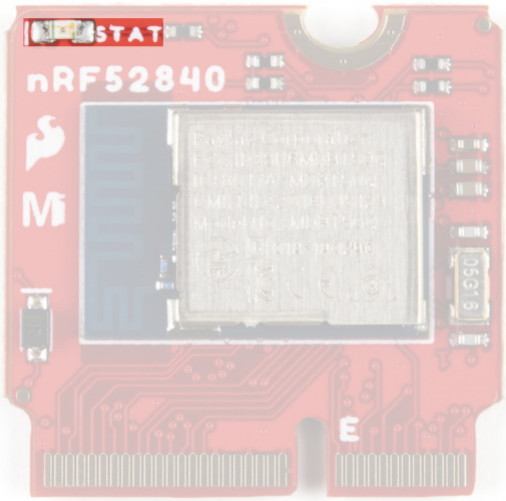
Along with the nRF52840 module, the Processor also includes external flash memory in the form of the 128Mb (16M x 8) W25Q128JV SpiFlash IC from Winbond Electronics. The flash IC is connected via SPI on the nRF52840. For full details on the Flash IC, refer to the datasheet.



The flash IC uses the secondary SPI bus (SPI1) and its Chip Select pin is tied to a dedicated pad on the nRF module (P0.12). Writing to the flash IC requires some extra work involving adjusting the SPI definitions in the variant file and is not recommended for anyone but advanced users. Writing to the Flash IC is beyond the scope of this tutorial.

STAT LED

The STAT LED is tied to a dedicated pin on the nRF module (P0.13) and can be used as a status LED or as a built-in LED. Control it directly in Arduino by writing to `LED_BUILTIN`.



nRF52840 Processor Pin Functions

The nRF52840 has a ton of awesome features including (but not limited to):

- **ARM Cortex-M4 CPU** with floating point unit (FPU), 64MHz
  - **1MB internal Flash** -- For all of your program, SoftDevice, and file-storage needs!
  - **256kB internal RAM** -- For your stack and heap storage.
- **Integrated 2.4GHz radio** with support for:
  - **Bluetooth Low Energy (BLE)** -- With peripheral and/or central BLE device support
  - **Bluetooth 5** -- Mesh Bluetooth!
  - **Nordic's proprietary RF protocol** -- If you want to communicate securely with other Nordic devices.
- **Interface Options**
  - **USB** -- Turn your nRF52840 into a USB mass-storage device, use a CDC (USB serial) interface, and more.
  - **UART** -- Serial interfaces with support for hardware flow-control if desired.
  - **I^{2}C** -- SparkFun's favorite 2-wire bi-directional bus interface
  - **SPI** -- If you prefer the 3+-wire serial interface
  - **Analog-to-digital converters (ADC)** -- Six pins on the nRF52840 Processor support analog inputs though four are used for other functionality on the MicroMod Processor.
  - **PWM** -- Two dedicated Pulse-Width Modulation interface pins.
  - **PDM** -- Pulse Density Modulation interface. The PDM module generates the PDM clock and supports single and dual channel data input.
  - **GPIO** -- Eleven dedicated General Purpose I/O pins.

Note: While nRF52840 supports peripheral-multiplexing to alter the functionality of almost any pin on the IC, in order to maintain compatibility with all SparkFun MicroMod Carrier Boards the pins are software defined to specific functions in the Arduino Board Definition. The pinout table at the end of this section outlines these definitions in totality and more detail.

Users who wish to alter pin functionality can modify the default board definition or use their own custom one but these can result in issues interacting with SparkFun Carrier Boards. If you decide to create or use a custom nRF52840 Processor board definition with a SparkFun Carrier Board, take note of the pins used for

components (sensors, displays, etc.) to maintain compatibility. The board variant files can be found in the nRF52840 Processor Hardware GitHub Repository.

I²C

As you may have guessed from our extensive Qwiic System, we love communicating with devices using I²C! The nRF52840 Processor features two I²C buses. The main I²C bus has dedicated pins connected to MiroMod pads 12/14, along with a dedicated Interrupt pin connected to MicroMod pad 16. The primary I²C Bus will almost always be connected to a Qwiic connector on your Carrier Board.

If you want a second I²C bus, the nRF52840 has SDA1 and SCL1 on MicroMod pads 51 and 53. To use this second bus, initialize it by calling `Wire1.begin();`.

UART

The nRF52840 Processor has three UARTs available. The primary UART is tied to USB_D± (MicroMod pads 3 and 5) for serial communication over USB. This is used for your standard serial upload as well as serial prints in Arduino. The secondary UART is a hardware UART tied to MicroMod pads 19 (RX1) and 17 (TX1). The tertiary UART is another hardware UART tied to MicroMod pads 20 (RX2) and 22 (TX2).

In Arduino, use the following calls to send data over the three UARTs:

- `Serial` - Communication over USB
- `Serial1` - Communication over RX1/TX1
- `Serial2` - Communication over RX2/TX2

SPI

The nRF52840 Processor features two SPI buses; SPI & SPI1. The primary SPI bus is tied to MicroMod pads listed below:

- CIPO (Controller In/Peripheral Out) - MicroMod pad 61
- COPI (Controller Out/Peripheral In) - MicroMod pad 59
- SCK (Serial Clock) - MicroMod pad 57
- CS (Chip Select) - MicroMod pad 55

SPI1 is tied to the flash IC and the pins are broken out to the M.2 connector on the MicroMod pads listed below:

- CIPO1/SDIO_DATA0 - MicroMod pad 64
- COPI1/SDIO_CMD - MicroMod pad 62
- SCK1/SDIO_CLK - MicroMod pad 60
- CS1/SDIO_DATA3 - MicroMod pad 70
- SDIO_DATA1 - MicroMod pad 66
- SDIO_DATA2 - MicroMod pad 68

Note: SparkFun has joined with other members of OSHWA in a resolution to move away from using "Master" and "Slave" to describe signals between the controller and the peripheral. This includes changing the traditional MISO/MOSI standard to CIPO/COPI. Check out this page for more on our reasoning behind this change. You can also see OSHWA's resolution here.

Audio

The nRF52840 supports audio input processing through Pulse-Density Modulation (PDM). The pins used are:

- PDM_DATA - P0.26 , MicroMod pad 52. This is the PDM data signal.
- PDM_CLOCK - P0.25, MicroMod pad 50. This is the PDM clock signal.

For more information on how to process audio signals using PDM on the nRF52840 Processor, refer to section 6.15 of the nRF52480 Datasheet as well as this example for the MicroMod Machine Learning Carrier Board.

Dedicated Pins/GPIO

Finally, along with all the interface options, the nRF52840 Processor has two pins dedicated for each of the following functionalities: Analog Read/Input, Digital I/O and Pulse Width Modulation. The nRF52840 Processor also has eleven general purpose I/O pins.

Dedicated Pins

- A0 - ADC0/P0.04 , MicroMod pad 34 (Input Only!)
- A1 - ADC1/P0.05 , MicroMod pad pad 38 (Input Only!)
- D0 - P0.27 , MicroMod pad pad 10
- D1 - P1.08 , MicroMod pad pad 18
- PWM0 - P0.06 , MicroMod pad pad 32
- PWM1 - P0.16 , MicroMod pad pad 47

General Purpose I/O pins

- G0 - GPIO0/P0.29 , MicroMod pad 40
- G1 - GPIO1/P0.03 , MicroMod pad 42
- G2 - GPIO2/P1.13 , MicroMod pad 44
- G3 - GPIO3/P1.12 , MicroMod pad 46
- G4 - GPIO4/P1.11 , MicroMod pad 48
- G5 - GPIO5/P0.17 , MicroMod pad 73
- G6 - GPIO6/P1.06 , MicroMod pad 71
- G7 - GPIO7/P1.04 , MicroMod pad 69
- G8 - GPIO8/P1.14 , MicroMod pad 67
- G9 - GPIO9/P0.09 , MicroMod pad 65 - Shared with NFC1
- G10 - GPIO10/P0.10 , MicroMod pad 63 - Shared with NFC2

MicroMod Pinout

Users looking for a complete pin map can find it the table below or you can refer to the schematic.

> **Heads up!** For many of the General Purpose I/O pins and other pins with multiple signal options, refer to your Carrier Board's Hookup Guide for information on how those pins are configured what they are used for. Not all pins are used on every Carrier Board.

NRF52840 PROCESSOR PINOUT TABLE

MICROMOD GENERAL PINOUT TABLE

MICROMOD GENERAL PIN DESCRIPTIONS

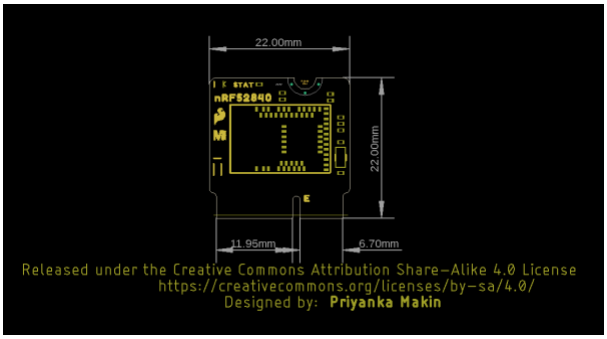
AUDIO	UART	GPIO/BUS	I²C	SDIO	SPI0	Dedicated

Function			Bottom Pin	Top Pin	Function		
			(Not Connected)	**75**	GND		
		3.3V	**74**	**73**	G5		
		RTC_3V_BATT	**72**	**71**	G6		
	SDIO_DATA3	SPI_CS1	**70**	**69**	G7		
		SDIO_DATA2	**68**	**67**	G8		
		SDIO_DATA1	**66**	**65**	G9	NFC1	
	SDIO_DATA0	SPI_CIPO1	**64**	**63**	G10	NFC2	
	SDIO_CMD	SPI_COPI1	**62**	**61**	SPI_CIPO		
	SDIO_SCK	SPI_SCK1	**60**	**59**	SPI_COPI		
		Not Connected	**58**	**57**	SPI_SCK		
		Not Connected	**56**	**55**	SPI_CS		
		Not Connected	**54**	**53**	I2C_SCL1		
		PDM_DATA	**52**	**51**	I2C_SDA1		
		PDM_CLK	**50**	**49**	BATT_VIN / 3 (0 to 3.3V)		
		G4	**48**	**47**	PWM1		
		G3	**46**	**45**	GND		
		G2	**44**	**43**	Not Connected		
		G1	**42**	**41**	Not Connected		
		G0	**40**	**39**	GND		
		A1	**38**	**37**	Not Connected		
		GND	**36**	**35**	Not Connected		

		A0	**34**	**33**	GND		
		PWM0	**32**	**31**	Module Key		
		Module Key	**30**	**29**	Module Key		
		Module Key	**28**	**27**	Module Key		
		Module Key	**26**	**25**	Module Key		
		Module Key	**24**	**23**	SWDIO		
		UART_TX2	**22**	**21**	SWDCK		
		UART_RX2	**20**	**19**	UART_RX1		
		D1	**18**	**17**	UART_TX1		
		I2C_INT	**16**	**15**	UART_CTS1		
		I2C_SCL	**14**	**13**	UART_RTS1		
		I2C_SDA	**12**	**11**	BOOT (Open Drain)		
		D0	**10**	**9**	USB_VIN		
			8	**7**	GND		
		RESET# (Open Drain)	**6**	**5**	USB_D-		
		3.3V_EN	**4**	**3**	USB_D+		
		3.3V	**2**	**1**	GND		

Board Dimensions

MicroMod Processor Boards all measure in at 22mm x 22mm, with 15mm to the top notch and 12mm to the E key. For more information regarding the processor board physical standards, head on over to the Getting Started with MicroMod tutorial and check out the Hardware Overview section.

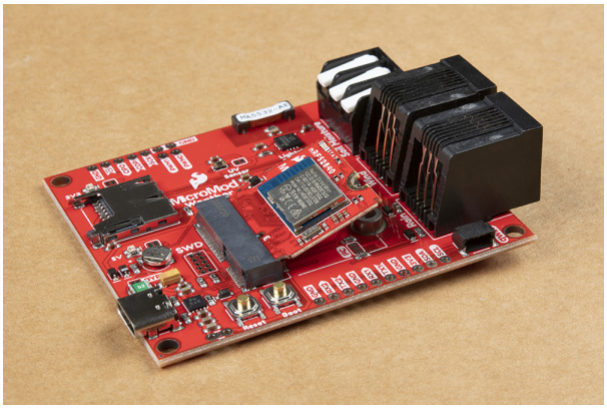


Hardware Assembly

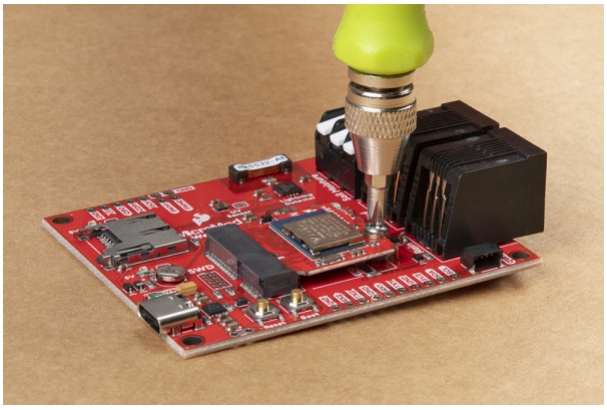
Now that we are familiar with the components on the nRF52840 Processor, it's time to assemble it with your chosen MicroMod Carrier Board and connect it to your computer. For this guide, we'll be using the MicroMod Weather Carrier Board.

Inserting the Processor Board

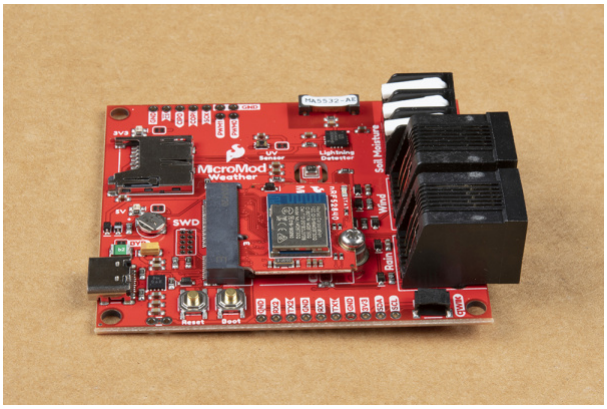
With the M.2 MicroMod connector, connecting your Processor board is a breeze. Simply match up the key on your Processor's beveled edge connector to the key on the M.2 connector. At a 45° angle, insert the Processor board to the M.2 connector. The Processor will stick up at an angle as seen here:



Once the board is in the socket, gently press the Processor board down, grab the set screw and tighten it with a Phillip's head screwdriver:



Once the Processor is secure, your assembled MicroMod system should look similar to the image below!



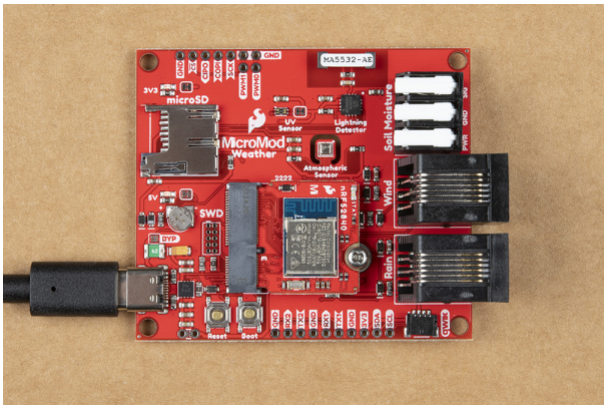
Note: There is *technically* no way to insert the processor backward since the key prevents it from mating with the M.2 connector. As an extra safeguard to prevent inserting a processor improperly or with a poor

connection, the mounting screw will not match up if the Processor is not plugged in correctly.

Connecting Everything Up

Depending on which Carrier Board you are using with your nRF52840 Processor, you may want to plug in any other devices (Qwiic breakouts, UART devices, SD cards, I/O devices, etc.) prior to plugging in your Carrier Board to USB. Refer to your Carrier Board's Hookup Guide for specific instructions for Carrier Board peripheral device Hardware Assembly.

With your nRF52840 Processor inserted and secured and your other devices connected, it's time to connect your MicroMod Carrier Board to your computer using the USB-C connector.



That's it! Now that our MicroMod circuit is assembled, we can move on to setting up the nRF52840 Processor in Arduino to start coding. Read on for detailed instructions on how to add the board to Arduino.

Arduino Software Setup

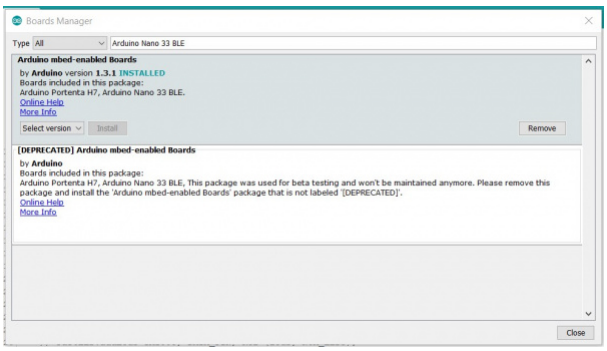
In this section we'll outline how to install the MicroMod nRF52840 board definitions and drivers. As we mentioned in the Hardware Overview section, the nRF52840 Processor ships with the same bootloader found on the Arduino Nano 33 BLE and takes advantage of Arduino's Mbed Core but we'll need to modify the board files a bit to add the MicroMod nRF52840 Processor to Arduino.

Arduino Mbed Core Installation

Heads up! If this is your first time using Arduino IDE or board add-ons, please review the following tutorials.

- Installing the Arduino IDE
- Installing Board Definitions in the Arduino IDE

Before we can use the nRF52840 Processor, we need to install the Arduino Mbed Core. The easiest way to do that is to use the Boards Manager tool in Arduino. Open the tool by navigating to the **Tools > Boards** menu and select **Boards Manager** at the top of the list. With the Boards Manager tool opened, search for "**Arduino Nano 33 BLE**", select the board package titled "**Arduino mbed-enabled Boards**" and click "Install".



Installation may take a few minutes as all necessary source files for the Mbed Core, plus all of the compiler and software-upload tools needed to use the boards package with Arduino, are included with this install.

Once the board definitions have been installed, you should see a new set of Arduino Mbed OS Boards under your **Tools > Board** menu. However, you won't see an option for the SparkFun MicroMod nRF52840 Processor as we need to manually add the board to the Arduino Mbed OS board package.

Adding the MicroMod nRF52840 Board Definition

Now that the Arduino Mbed Core is installed, we need to modify it to add the MicroMod nRF52840 Processor. To do this we need to add and replace a few files in the Arduino Mbed Core.

If you are familiar with modifying an Arduino boards package with a custom board, you can skip the following instructions and use the links below to get the necessary files from the Hardware GitHub repo.

To modify the Arduino Mbed Core we need the necessary files from the nRF52840 Processor Hardware GitHub Repository. You can download the repository from the GitHub following the link above or you can download the repository in a ZIP by clicking the button below:

DOWNLOAD THE MICROMOD NRF52840 HARDWARE REPOSITORY

With the GitHub repository downloaded we need to open the Arduino Mbed Core folder. If you used the Boards Manager tool to install it, the folder should be in the following location:

- **Windows**: `%LOCALAPPDATA%\Arduino15\packages\arduino\hardware\mbed\<version>`
- **OSX**: `~/Library/Arduino15/packages/arduino/hardware/mbed/<version>`
- **Linux**: `~/.arduino15/packages/arduino/hardware/mbed/<version>`

If you installed the core manually into your Arduino sketchbook, it will be in `"hardware/arduino/mbed"`.

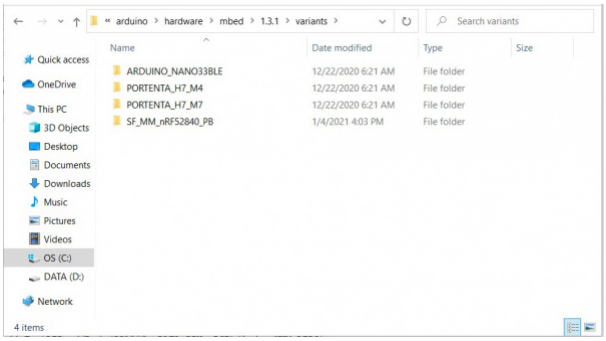
> **Note:** Windows users struggling to locate the AppData folder should make sure they have their View settings to show Hidden Items and are logged in as an administrative user.

Open the folder with the version number (v1.3.1 at the time of this writing). We need to replace or modify the `boards.txt` file and add the `SF_MM_nRF52840_PB` folder to the Variants folder.

To replace the `boards.txt` file simply delete the `boards.txt` file in the `mbed/<version>` folder and copy/paste the `boards.txt` file from the `nRF Hardware GitHub Repository` folder. Alternatively, you can modify the `boards.txt` in the `mbed/<version>` folder by opening the file in a text editor, scrolling to the bottom of the file and copy/pasting the below to the file:

```
##################################################################

sfnrf52840pb.name=SparkFun MicroMod nRF52840 Processor Board
sfnrf52840pb.build.core=arduino
sfnrf52840pb.build.crossprefix=arm-none-eabi-
sfnrf52840pb.build.compiler_path={runtime.tools.arm-none-eabi-gcc.path}/bin/

sfnrf52840pb.build.variant=SF_MM_nRF52840_PB
sfnrf52840pb.build.mcu=cortex-m4
sfnrf52840pb.build.extra_flags=
sfnrf52840pb.build.architecture=cortex-m4
sfnrf52840pb.build.fpu=fpv4-sp-d16
sfnrf52840pb.build.float-abi=softfp
sfnrf52840pb.build.board=ARDUINO_NANO33BLE
sfnrf52840pb.build.ldscript=linker_script.ld
sfnrf52840pb.compiler.mbed.arch.define=-DARDUINO_ARCH_NRF52840
sfnrf52840pb.compiler.mbed.defines={build.variant.path}/defines.txt
sfnrf52840pb.compiler.mbed.ldflags={build.variant.path}/ldflags.txt
sfnrf52840pb.compiler.mbed.cflags={build.variant.path}/cflags.txt
sfnrf52840pb.compiler.mbed.cxxflags={build.variant.path}/cxxflags.txt
sfnrf52840pb.compiler.mbed.includes={build.variant.path}/includes.txt
sfnrf52840pb.compiler.mbed.extra_ldflags=-lstdc++ -lsupc++ -lm -lc -lgcc -lnosys
sfnrf52840pb.compiler.mbed="{build.variant.path}/libs/libmbed.a" "{build.variant.path}/libs/libc
c_310_core.a" "{build.variant.path}/libs/libcc_310_ext.a" "{build.variant.path}/libs/libcc_310_t
rng.a"
sfnrf52840pb.vid.0=0x2341
sfnrf52840pb.pid.0=0x005a
sfnrf52840pb.vid.1=0x2341
sfnrf52840pb.pid.1=0x805a

sfnrf52840pb.upload.tool=bossac
sfnrf52840pb.upload.protocol=
sfnrf52840pb.upload.use_1200bps_touch=true
sfnrf52840pb.upload.wait_for_upload_port=true
sfnrf52840pb.upload.native_usb=true
sfnrf52840pb.upload.maximum_size=983040
sfnrf52840pb.upload.maximum_data_size=262144

sfnrf52840pb.bootloader.tool=openocd
sfnrf52840pb.bootloader.extra_action.preflash=echo INFO:removed_mass-erase
sfnrf52840pb.bootloader.config=-f target/nrf52.cfg
sfnrf52840pb.bootloader.programmer=-f interface/cmsis-dap.cfg
sfnrf52840pb.bootloader.file=nano33ble/bootloader.hex
```

With the boards file updated or replaced, add the **`SF_MM_nRF52840_PB`** folder to **`mbed/<version>/variants`** .



After completing these steps, **restart the Arduino IDE** if it was open. After restarting, you should be able to select **"SparkFun MicroMod nRF52840 Processor Board"** from the **Boards** menu.

Arduino Example: Blink

With the SparkFun MicroMod nRF52840 Processor Board Definitions added to the Arduino nRF528X mbed core, let's do some quick code examples to make sure everything went correctly during the Arduino Software Setup.

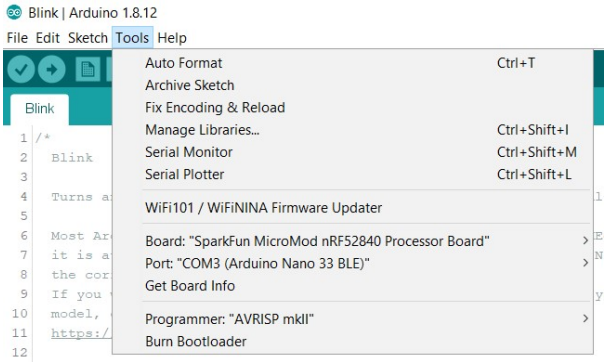
Selecting and Loading Blink

We'll start off with a basic Blink example to turn the STAT LED on and off just to make sure everything is working properly and your Processor can accept code.

Open up the Arduino IDE and select the "Blink" example by navigating to **"File > Examples > 01.Basics > Blink"** or by copying the code below into a blank sketch:

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

With the example opened, select your Board (SparkFun MicroMod nRF52840 Processor Board) and Port using the **Tools > Board** and **Tools > Port** menus and click the **"Upload"** button. Barring any issues during compilation and upload, the STAT LED on your nRF52840 Processor should be blinking on and off every second.



If the STAT LED remains off, the Processor may still be sitting in the bootloader. After uploading a sketch, you may need to **tap the reset button** to get your nRF52840 Processor to exit bootloader mode and run the sketch.

Arduino Example: Bluetooth Low Energy

Note: This example uses a SparkFun Arduino library that needs to be installed to Arduino. If you have not previously installed an Arduino library, please check out our installation guide for help.

This example demonstrates how to send sensor data (in this case, temperature data from a BME280) over Bluetooth Low Energy (BLE). The code creates a BLE peripheral with the temperature service and reading characteristic. This example can be used as a template for creating BLE peripherals and sending the data to another BLE device.

Prior to uploading the example, we'll need to install a couple of things to use and interact with the code.

BME280 Hardware

First off, we of course need a BME280 Atmospheric sensor. The Weather Carrier Board we used in the Hardware Assembly section includes a BME280 so if you are using that as your Carrier Board you're all set. If you are using a different Carrier Board you can attach a BME280 breakout like this Qwiic version to the Qwiic connector on your Carrier or to the primary I^2C bus pins.

SparkFun BME280 Arduino Library

If you wish to use the code as is, you'll need to install the SparkFun BME280 Arduino Library. Install the library using the Arduino Library Manager by searching for **"SparkFun BME280 Arduino Library"**. Users who prefer to manually install the library can download the GitHub Repository or by clicking the button below:

DOWNLOAD THE SPARKFUN BME280 LIBRARY (ZIP)

A detailed overview of the library can be found here in our Qwiic Atmospheric Sensor (BME280) Hookup Guide.

BLE Temperature Example

Copy the code below into a blank sketch or you can also find the example included the Hardware GitHub Repository. If you are opening the example from the downloaded repository, navigate to the **"Test Sketches/ble_temp"** folder and open the sketch.

Select your **Board** and **Port** from the **Tools** menu just as you did for the "Blink" example and click "Upload".

```
#include <ArduinoBLE.h>
#include "SparkFunBME280.h"

BME280 tempSensor;

 // BLE Temperature Service
BLEService temperatureService("9feb1060-0814-11eb-adc1-0242ac120002");

// BLE Temperature Reading Characteristic
BLEIntCharacteristic temperatureReadingChar("9feb1060-0814-11eb-adc1-0242ac120002",  // standard
16-bit characteristic UUID
    BLERead | BLENotify); // remote clients will be able to get notifications if this characteri
stic changes

int oldTemperatureF = 0; //last temperature reading from BME280

void setup() {
  Serial.begin(115200);    // initialize serial communication
  while (!Serial);

  pinMode(LED_BUILTIN, OUTPUT); // initialize the built-in LED pin to indicate when a central is
connected

  // begin initialization
  if (!BLE.begin()) {
    Serial.println("starting BLE failed!");

    while (1);
  }

  if (tempSensor.begin() == false) { //Connect to BME280
    Serial.println("BME280 did not respond.");
    while(1); //Freeze
  }

  /* Set a local name for the BLE device
     This name will appear in advertising packets
     and can be used by remote devices to identify this BLE device
     The name can be changed but maybe be truncated based on space left in advertisement packet
  */
  BLE.setLocalName("TemperatureReading");
  BLE.setAdvertisedService(temperatureService); // add the service UUID
  temperatureService.addCharacteristic(temperatureReadingChar); // add the temperature reading c
haracteristic
  BLE.addService(temperatureService); // Add the temperature service

  /* Start advertising BLE.  It will start continuously transmitting BLE
     advertising packets and will be visible to remote BLE central devices
     until it receives a new connection */

  // start advertising
  BLE.advertise();
```

```
  Serial.println("Bluetooth device active, waiting for connections...");
}

void loop() {
  // wait for a BLE central
  BLEDevice central = BLE.central();

  // if a central is connected to the peripheral:
  if (central) {
    Serial.print("Connected to central: ");
    // print the central's BT address:
    Serial.println(central.address());
    // turn on the LED to indicate the connection:
    digitalWrite(LED_BUILTIN, HIGH);

    // check the temperature every second
    // while the central is connected:
    while (central.connected()) {
        updateTemperature();
        delay(1000);
    }
    // when the central disconnects, turn off the LED:
    digitalWrite(LED_BUILTIN, LOW);
    Serial.print("Disconnected from central: ");
    Serial.println(central.address());
  }
}

void updateTemperature() {
  /* Read the current temperature from the BME280
  */
  int temperatureF = (int)tempSensor.readTempF();

  if (temperatureF != oldTemperatureF) {
    Serial.print("Temperature in F is now: "); // print it
    Serial.println(temperatureF);
    temperatureReadingChar.writeValue(temperatureF);  // and update the temperature reading char
acteristic
    oldTemperatureF = temperatureF;
  }
}
```

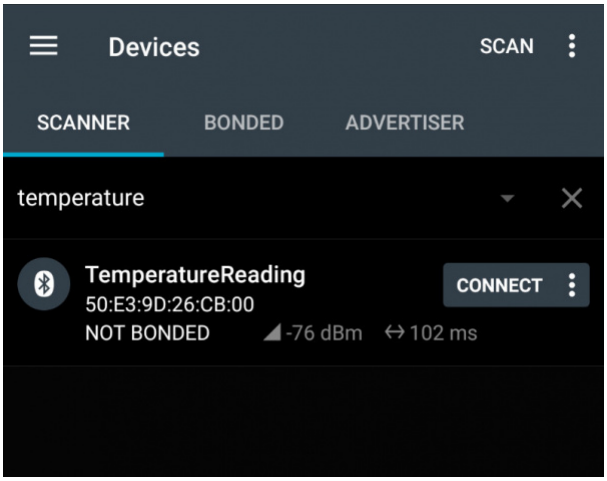
Reading the Temperature Data with the nRF Connect App

In order to interact with this example we need another device (phone/computer/tablet) connected to the nRF52840. There are plenty of BLE Central applications out there but for this tutorial we'll use Nordic's helpful (and free) testing tool called **nRF Connect for Mobile** for interfacing between our two devices. You can find the application on both the Google Play Store or Apple's App Store or you can find them by clicking the buttons below. Go ahead and install the app before moving on with this example.

NRF CONNECT FOR ANDROID **NRF CONNECT FOR IOS**

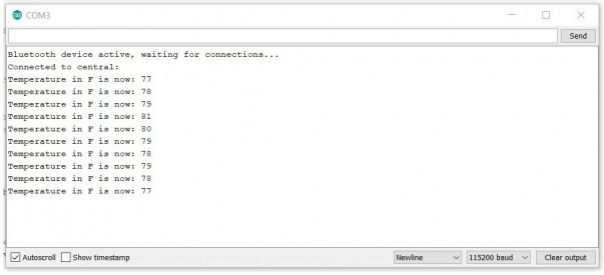
Open the serial monitor after uploading finishes and set the baud rate to **115200**. If there are no failures initializing BLE and the BME280, the code will start advertising the nRF52840 Processor until it receives a new connection.

At this point (if you have not already), open your BLE Central App and scan for devices. The nRF52840 Processor should appear as an available BLE device named "**TemperatureReading**".

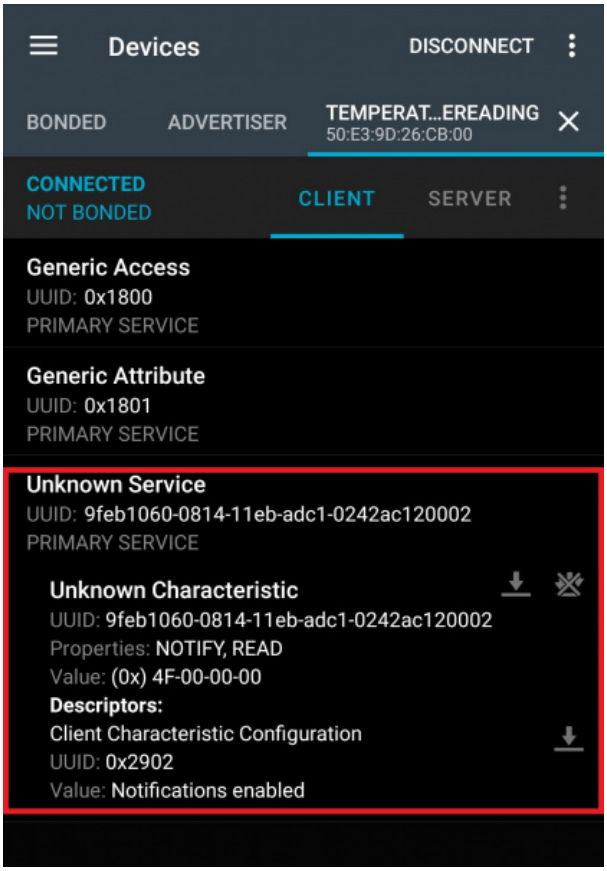


You can see here we've added a filter for "Temperature" to narrow down the scan results.

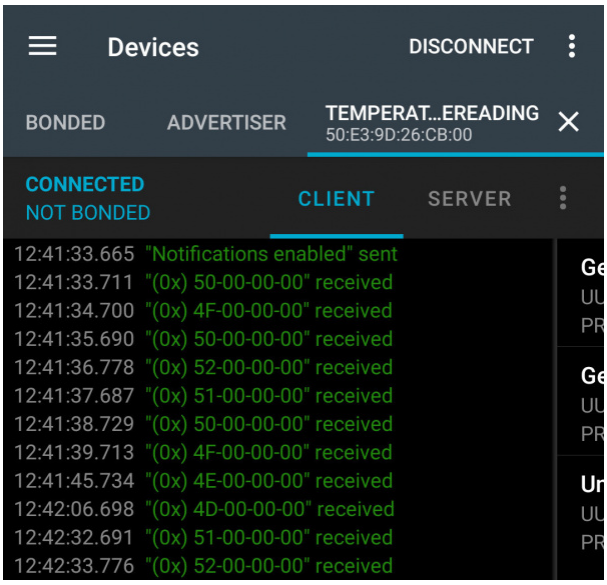
Once the nRF52840 connects to the nRF Connect App, the code will print out `Connected to central:` along with the central Bluetooth's address. The STAT LED on the Processor should also turn on indicating a successful pairing and connection.



In order to receive temperature data from the nRF52840 on your nRF Connect App you will need to enable notifcations by opening the BLE Temperature Reading Characteristic (shown below as Unknown Service with the full UUID set in the example code) and clicking the icon with three downward-facing arrows.



While paired, and if the temperature recorded by the BME280 changes, the code prints the new temperature value to the serial terminal and sends a data packet of the temperature value in Hexadecimal to the paired central device. On the nRF Connect app, swipe to the right to open the data log to see the Hex values for the temperature:



From here, you can take the Hexadecimal values from the BME280 and parse them however you choose to use for your BLE application.

Troubleshooting

We've got some troubleshooting tips for a couple of common "gotchas" you may run into using the nRF52840 Processor.

Force Bootloader Mode

In case your board gets stuck in an unresponsive state you can force the nRF52840 into Bootloader mode by double-tapping the RESET button on your Carrier Board. This can help recover the board from an unknown state or you can use it to directly interact with the board in bootloader mode. You can also use the RESET button to start any code that hangs after upload in case the Processor is still in Bootloader mode.

MicroMod Board Installation

If you run into problems after modifying the Arduino nRF5284X core the easiest way to remedy that is to use the Boards Manager tool in Arduino to switch to an older version of the core and then revert to the current version. Alternatively, you can use the Boards Manager to uninstall and reinstall the Arduino core.

After you have reverted or reinstalled the core go through the steps outlined in the Arduino Software Setup section of this guide to install the custom MicroMod nRF52840 Processor board definition.

General Troubleshooting Help & Technical Support

Not working as expected and need help?

If you need technical assistance or more information regarding this or another SparkFun product that is not working as you expected, we recommend heading on over to the SparkFun Technical Assistance page for some initial troubleshooting.

SPARKFUN TECHNICAL ASSISTANCE PAGE

If you don't find what you need there, the SparkFun Forums: MicroMod are a great place to find and ask for help. If this is your first visit, you'll need to create a Forum Account to search product forums and post questions.

SPARKFUN FORUMS: MICROMOD

Resources and Going Further

For more information about the MicroMod nRF52840 Processor, check out the following links:

- Schematic (PDF)
- Eagle Files (ZIP)
- Board Dimensions (PNG)
- Datasheet - MDBT50Q Module
- Datasheet - nRF52840 IC
- GitHub Hardware Repo

For more information about the SparkFun MicroMod Ecosystem, take a look at the links below:

- Getting Started with MicroMod
- Designing with MicroMod
- MicroMod Info Page
- MicroMod Forums

Need some inspiration for a project using your nRF52840 Processor? The tutorials below may help you get started!

Designing with MicroMod

This tutorial will walk you through the specs of the MicroMod processor and carrier board as well as the basics of incorporating the MicroMod form factor into your own PCB designs!

SparkFun MicroMod Input and Display Carrier Board Hookup Guide

A short Hookup Guide to get started with the SparkFun MicroMod Input and Display Carrier Board

MicroMod Data Logging Carrier Board Hookup Guide

Get started with some customizable MicroMod data logging with the Data Logging Carrier Board.

MicroMod Machine Learning Carrier Board Hookup Guide

Get hacking with this tutorial on our Machine Learning Carrier Board!