Introduction to Microcontrollers

Experiment 4:

Bluetooth Audio Transmission

Developed 2020 by S. Rowland and M. Eckels

Contents

1 Objectives

To learn to use Bluefruit Low-Energy modules for UART transmission.

2 Background

2.1 What is Bluetooth Low Energy?

Bluetooth Low Energy or BLE is a wireless personal area network technology intended to provide similar features and functionality to standard Bluetooth while running at a reduced power consumption.

2.2 What is an Electret Microphone?

Electret microphones are a type of electrostatic capacitor-based microphone. These microphones do not require a polarizing power supply such as condenser microphones by utilizing a permanently charged material. These microphones provide a simple and cost-effective way of capturing audio.

2.3 UART Communication

UART or Universal Asynchronous Receiver/Transmitter is used to transmit and receive serial data over to wires and provides duplex communication. UART with Arduinos can be handled with hardware and software. The Arduino Mega supports 4 hardware UART ports but can also simulate UART RX and TX with software serial over GPIO pins.

2.3.1 Hardware Vs. Software UART

Hardware serial provides hardware data buffers that will allow for simultaneous receipt of data packets and allows for higher data rates than software serial. Software serial is limited to single port receipt due to utilizing system interrupts for receiving data. Software serial does have its place though i.e. on the Arduino Uno, only one hardware serial port is available; if more than one serial port must be used on an Arduino Uno then software serial could be used.

2.4 Bluefruit LE UART Friend Module

The Bluefruit Low-Energy Bluetooth module provides a simple way to communicate over Bluetooth with Bluetooth devices such as a mobile phone. These modules can be easily configured to communicate and once paired will automatically connect on power up to provide quick and easy communication over UART.

2.4.1 Command and UART Mode

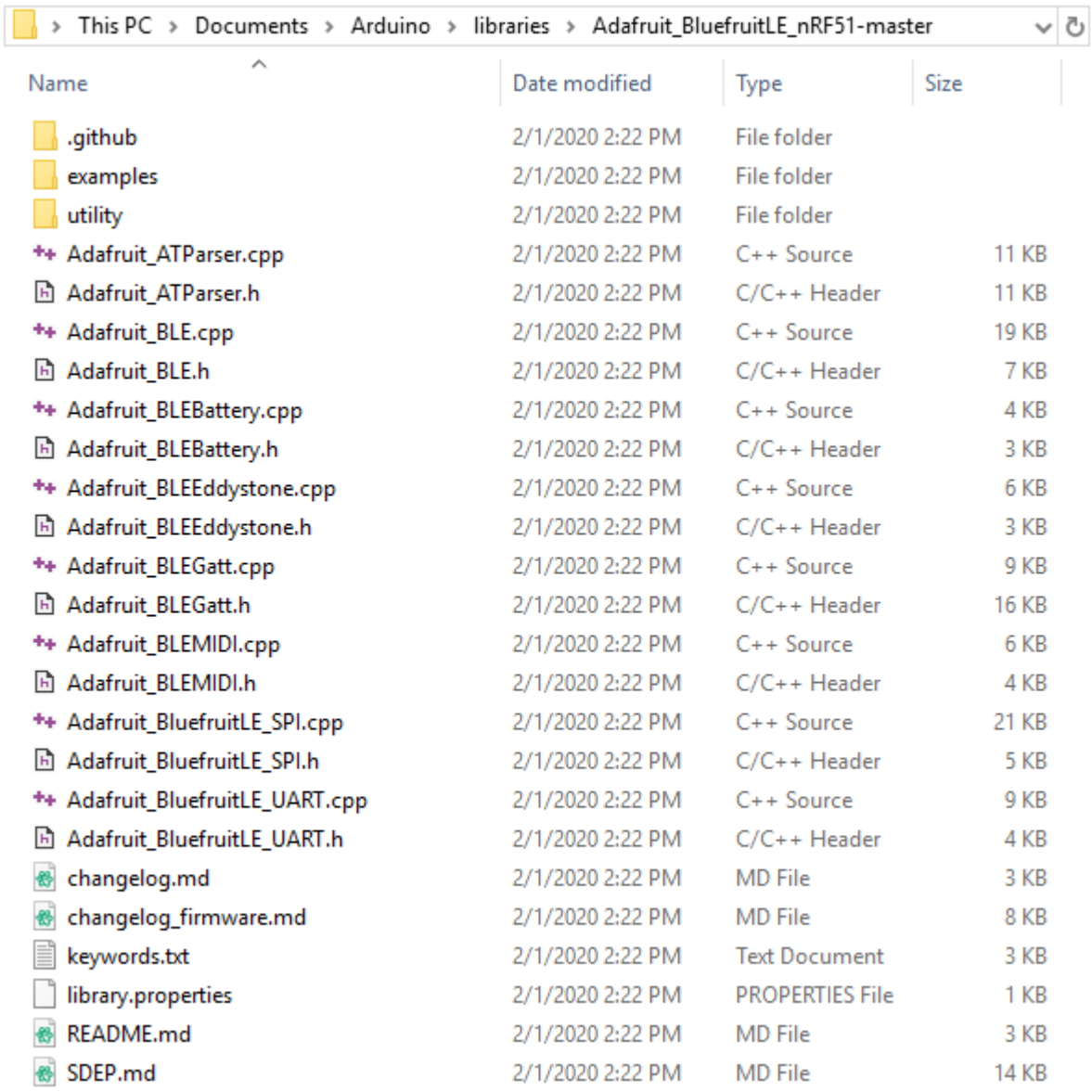
The Bluefruit module provides a switch to enter a configuration mode or to run in standard UART transmit mode. When the switch is in Command mode, the device can be configured to connect with a secondary device or to view information about the device such as internal temperatures or the RSSI value to check the reception with another Bluetooth device. In this lab we will pair two Bluefruit modules to communicate

with one another in command mode and then will switch to UART mode for communicating over serial wirelessly – like the previous two labs.

2.4.2 Installing the Arduino Library

To install an external library that is not included in a repository, we will need to start with finding and downloading the library. The Bluefruit LE library can be found at github.com/adafruit/Adafruit_BluefruitLE_nRF51

On GitHub, click the Clone or download drop down and download the Zip file. Extract the zip file to the libraries folder under the Arduino folder in the Documents folder in your user account (C:\Users\xxxxxxxx\Documents\Arduino\libraries\Adafruit_BluefruitLE_nRF51). Verify that the Adafruit_BluefruitLE_nRF51 folder at this location contains something like the following:



2.4.3 Bluefruit App for iOS and Android

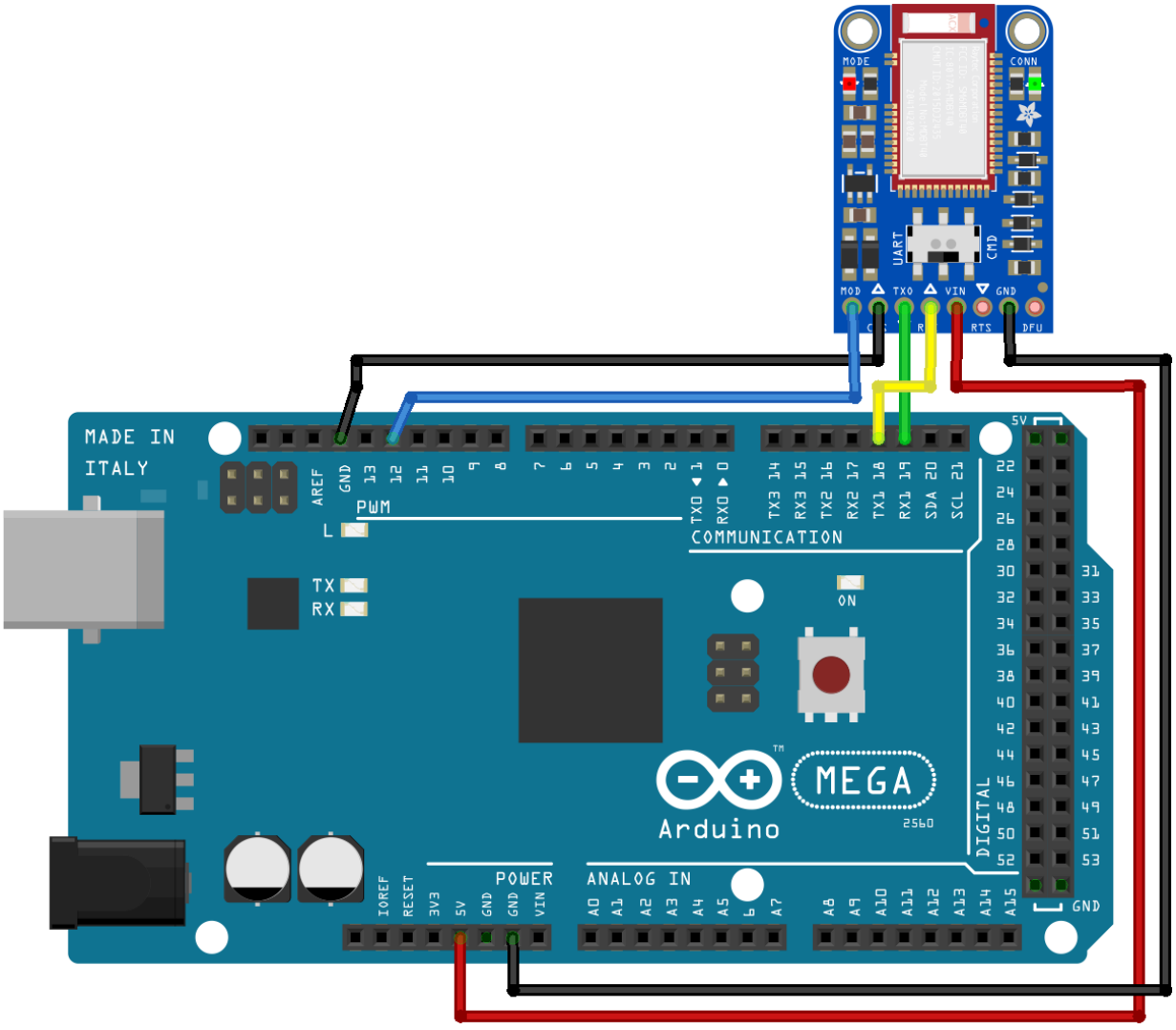
Adafruit provides an easy way for communicating with these BLE devices, download the Adafruit Bluefruit LE Connect app from either the iOS store or Android Play Store.

3 Procedures

3.1 Configuring BLE Module for UART

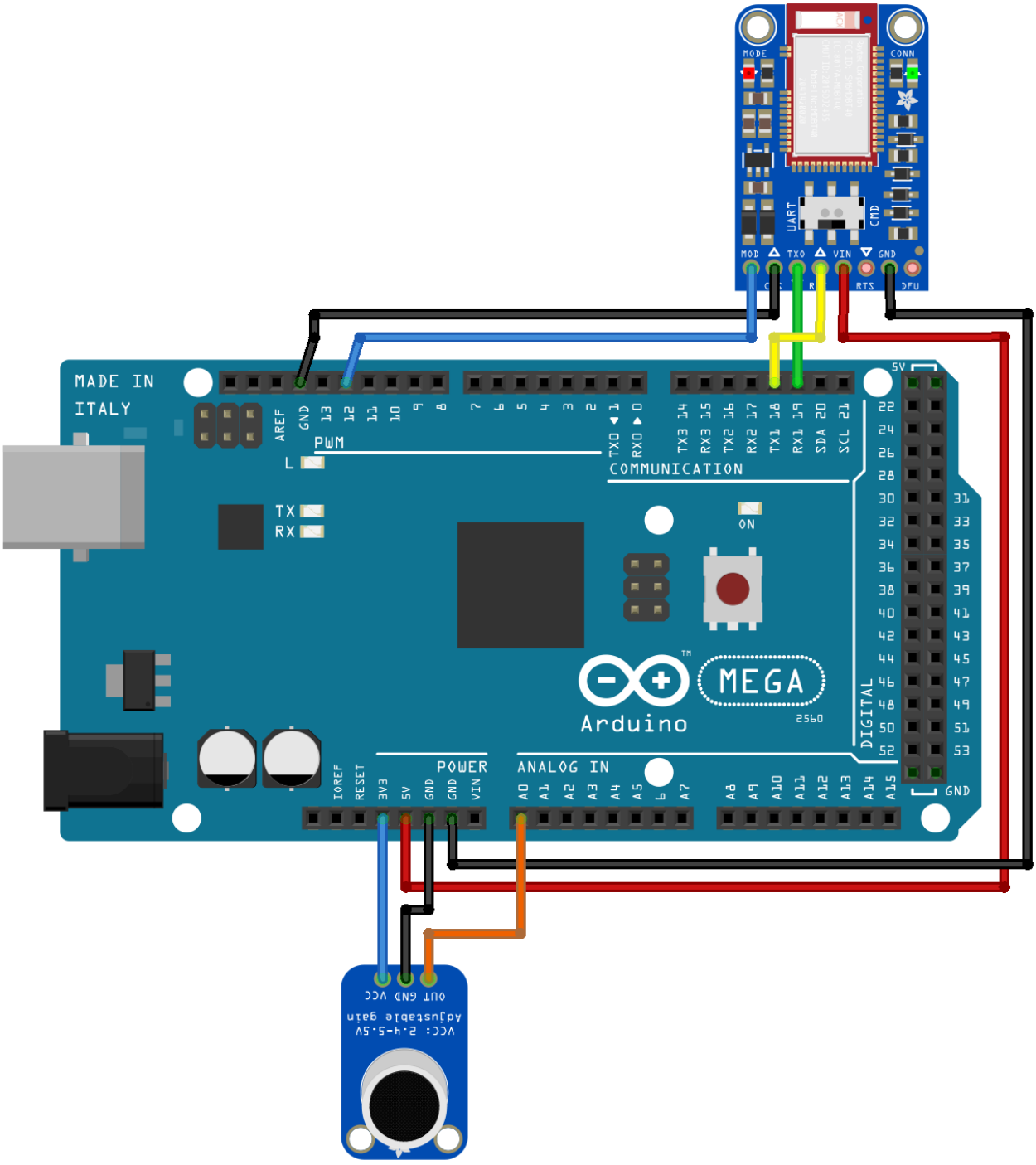
3.1.1 Wiring BLE Module for Configuration

Wire the BLE module to the Arduino as follows:



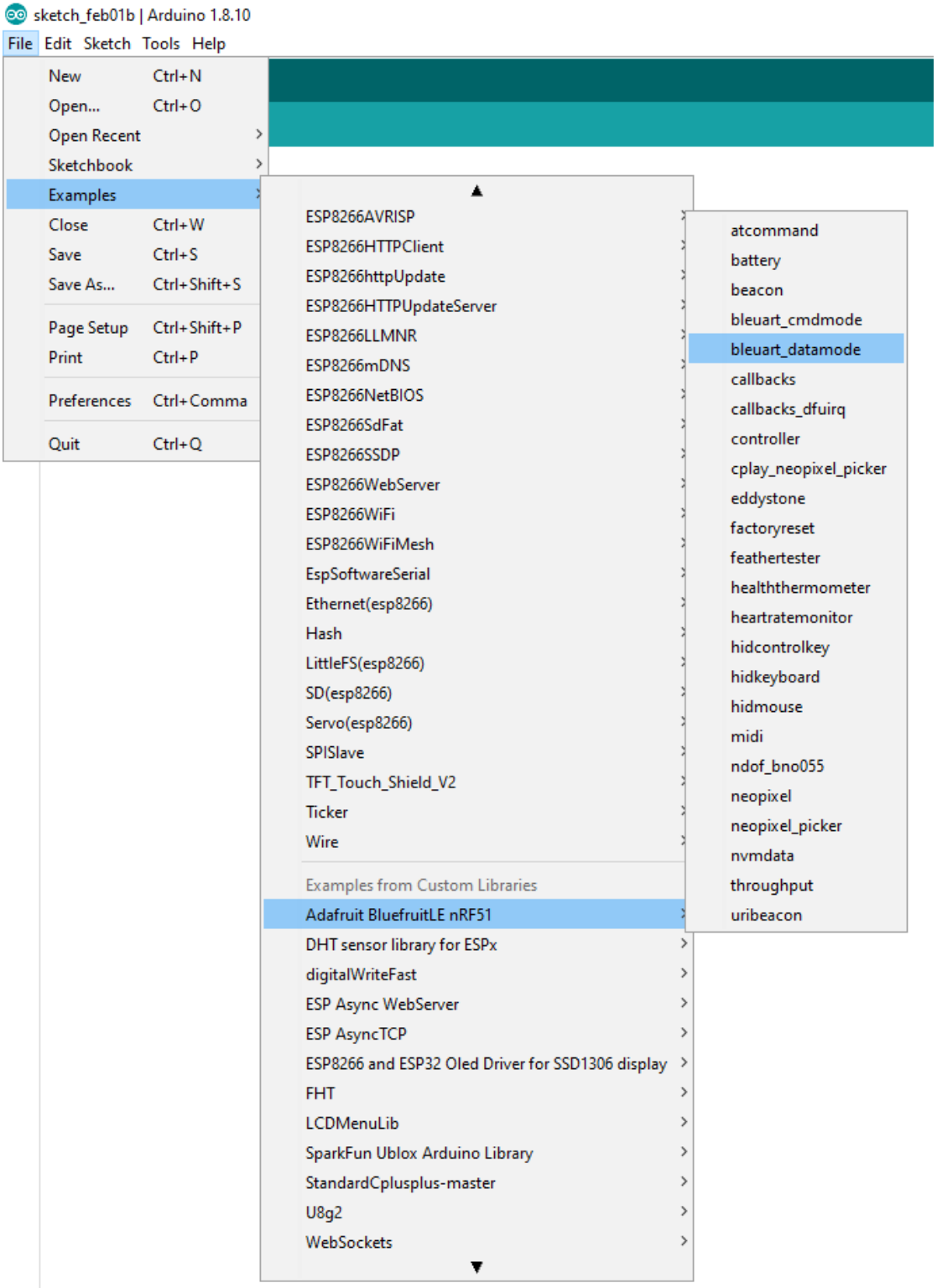
3.1.2 Wiring Electret Microphone with Arduino Mega

Once the BLE is wired and configured, wire the electret microphone to the Arduino as follows:



3.1.3 Programming the Arduino to Read and Write from the BLE Module

With the installed BLE library (Section 2.4.2), in Arduino open File, Examples, Adafruit BluefruitLE nRF51, and bleuart_datamode. Ensure that the switch on the BLE module is set to UART.



This will give the following example code:

```
/*********************************************************************
 This is an example for our nRF51822 based Bluefruit LE modules

 Pick one up today in the adafruit shop!

 Adafruit invests time and resources providing this open source code,
 please support Adafruit and open-source hardware by purchasing
 products from Adafruit!

 MIT license, check LICENSE for more information
 All text above, and the splash screen below must be included in
 any redistribution
*********************************************************************/

#include <Arduino.h>
#include <SPI.h>
#include "Adafruit_BLE.h"
#include "Adafruit_BluefruitLE_SPI.h"
#include "Adafruit_BluefruitLE_UART.h"

#include "BluefruitConfig.h"

#if SOFTWARE_SERIAL_AVAILABLE
  #include <SoftwareSerial.h>
#endif

/*=========================================================================
    APPLICATION SETTINGS

    FACTORYRESET_ENABLE       Perform a factory reset when running this sketch

                              Enabling this will put your Bluefruit LE module
                              in a 'known good' state and clear any config
                              data set in previous sketches or projects, so
                              running this at least once is a good idea.

                              When deploying your project, however, you will
                              want to disable factory reset by setting this
                              value to 0.  If you are making changes to your
                              Bluefruit LE device via AT commands, and those
                              changes aren't persisting across resets, this
                              is the reason why.  Factory reset will erase
                              the non-volatile memory where config data is
                              stored, setting it back to factory default
                              values.

                              Some sketches that require you to bond to a
                              central device (HID mouse, keyboard, etc.)
                              won't work at all with this feature enabled
                              since the factory reset will clear all of the
                              bonding data stored on the chip, meaning the
```

```
                            central device won't be able to reconnect.
    MINIMUM_FIRMWARE_VERSION  Minimum firmware version to have some new
features
    MODE_LED_BEHAVIOUR        LED activity, valid options are
                              "DISABLE" or "MODE" or "BLEUART" or
                              "HWUART"  or "SPI"  or "MANUAL"
    ------------------------------------------------------------------------*/
    #define FACTORYRESET_ENABLE         1
    #define MINIMUM_FIRMWARE_VERSION    "0.6.6"
    #define MODE_LED_BEHAVIOUR          "MODE"
/*=========================================================================*/

// Create the bluefruit object, either software serial...uncomment these
lines
/*
SoftwareSerial bluefruitSS = SoftwareSerial(BLUEFRUIT_SWUART_TXD_PIN,
BLUEFRUIT_SWUART_RXD_PIN);

Adafruit_BluefruitLE_UART ble(bluefruitSS, BLUEFRUIT_UART_MODE_PIN,
                      BLUEFRUIT_UART_CTS_PIN, BLUEFRUIT_UART_RTS_PIN);
*/

/* ...or hardware serial, which does not need the RTS/CTS pins. Uncomment
this line */
Adafruit_BluefruitLE_UART ble(BLUEFRUIT_HWSERIAL_NAME,
BLUEFRUIT_UART_MODE_PIN);

/* ...hardware SPI, using SCK/MOSI/MISO hardware SPI pins and then user
selected CS/IRQ/RST */
//Adafruit_BluefruitLE_SPI ble(BLUEFRUIT_SPI_CS, BLUEFRUIT_SPI_IRQ,
BLUEFRUIT_SPI_RST);

/* ...software SPI, using SCK/MOSI/MISO user-defined SPI pins and then user
selected CS/IRQ/RST */
//Adafruit_BluefruitLE_SPI ble(BLUEFRUIT_SPI_SCK, BLUEFRUIT_SPI_MISO,
//                             BLUEFRUIT_SPI_MOSI, BLUEFRUIT_SPI_CS,
//                             BLUEFRUIT_SPI_IRQ, BLUEFRUIT_SPI_RST);


// A small helper
void error(const __FlashStringHelper*err) {
  Serial.println(err);
  while (1);
}

/**************************************************************************/
/*!
    @brief  Sets up the HW an the BLE module (this function is called
            automatically on startup)
*/
/**************************************************************************/
void setup(void)
{
```

```
  while (!Serial);  // required for Flora & Micro
  delay(500);

  Serial.begin(115200);
  Serial.println(F("Adafruit Bluefruit Command <-> Data Mode Example"));
  Serial.println(F("------------------------------------------------"));

  /* Initialise the module */
  Serial.print(F("Initialising the Bluefruit LE module: "));

  if ( !ble.begin(VERBOSE_MODE) )
  {
    error(F("Couldn't find Bluefruit, make sure it's in CoMmanD mode & check
wiring?"));
  }
  Serial.println( F("OK!") );

  if ( FACTORYRESET_ENABLE )
  {
    /* Perform a factory reset to make sure everything is in a known state */
    Serial.println(F("Performing a factory reset: "));
    if ( ! ble.factoryReset() ){
      error(F("Couldn't factory reset"));
    }
  }

  /* Disable command echo from Bluefruit */
  ble.echo(false);

  Serial.println("Requesting Bluefruit info:");
  /* Print Bluefruit information */
  ble.info();

  Serial.println(F("Please use Adafruit Bluefruit LE app to connect in UART
mode"));
  Serial.println(F("Then Enter characters to send to Bluefruit"));
  Serial.println();

  ble.verbose(false);  // debug info is a little annoying after this point!

  /* Wait for connection */
  while (! ble.isConnected()) {
      delay(500);
  }

  Serial.println(F("******************************"));

  // LED Activity command is only supported from 0.6.6
  if ( ble.isVersionAtLeast(MINIMUM_FIRMWARE_VERSION) )
  {
    // Change Mode LED Activity
    Serial.println(F("Change LED activity to " MODE_LED_BEHAVIOUR));
    ble.sendCommandCheckOK("AT+HWModeLED=" MODE_LED_BEHAVIOUR);
```

```
  }

  // Set module to DATA mode
  Serial.println( F("Switching to DATA mode!") );
  ble.setMode(BLUEFRUIT_MODE_DATA);

  Serial.println(F("******************************"));
}

/**************************************************************************/
/*!
    @brief  Constantly poll for new command or response data
*/
/**************************************************************************/
void loop(void)
{
  // Check for user input
  char n, inputs[BUFSIZE+1];

  if (Serial.available())
  {
    n = Serial.readBytes(inputs, BUFSIZE);
    inputs[n] = 0;
    // Send characters to Bluefruit
    Serial.print("Sending: ");
    Serial.println(inputs);

    // Send input data to host via Bluefruit
    ble.print(inputs);
  }

  // Echo received data
  while ( ble.available() )
  {
    int c = ble.read();

    Serial.print((char)c);

    // Hex output too, helps w/debugging!
    Serial.print(" [0x");
    if (c <= 0xF) Serial.print(F("0"));
    Serial.print(c, HEX);
    Serial.print("] ");
  }
}
```

On line 75, add a comment to disable the BLE object from being created as an SPI type device:

```
/* ...hardware SPI, using SCK/MOSI/MISO hardware SPI pins and then user
selected CS/IRQ/RST */
```

```
//Adafruit_BluefruitLE_SPI ble(BLUEFRUIT_SPI_CS, BLUEFRUIT_SPI_IRQ,
BLUEFRUIT_SPI_RST);
```

On line 72, remove the comment to enable the BLE object as a UART device and replace `BLUEFRUIT_HWSERIAL_NAME` with `Serial1`:

```
/* ...or hardware serial, which does not need the RTS/CTS pins. Uncomment
this line */
Adafruit_BluefruitLE_UART ble(Serial1, BLUEFRUIT_UART_MODE_PIN);
```

With this code the Serial1 port can be read for communicating over the BLE device. To read from the electret microphone the code within the main loop will be modified and a line added to the setup. Start by deleting all the code within the void loop to give the following:

```
void loop(void)
{

}
```

At the end of the void setup function, add the following at the end of the function to enable the A0 input pin on the Arduino Mega:

```
pinMode(A0, INPUT);
```

Now with the A0 pin initialized, add the following to the void loop function:

```
void loop(void)
{
  uint8_t value = map(analogRead(A0), 0, 1023, 0, 255);

  ble.print(value);
  ble.print(',');
  ble.print('\n');

  delay(10);
}
```

This will continuously read from the electret microphone and write the data to the UART port.

Once modified, upload this code to the Arduino Mega and connect to the Serial Port. You should see the following:

```
Adafruit Bluefruit Command <-> Data Mode Example
------------------------------------------------
Initialising the Bluefruit LE module: ATZ

<- OK
OK!
Performing a factory reset:
AT+FACTORYRESET

<- OK
ATE=0

<- ATE=0
OK
Requesting Bluefruit info:
----------------
BLEFRIEND32
nRF51822 QFACA00
2C39C03CA93402D4
0.6.2
0.6.2
Apr 30 2015
S110 8.0.0, 0.2
----------------
Please use Adafruit Bluefruit LE app to connect in UART mode
Then Enter characters to send to Bluefruit
```

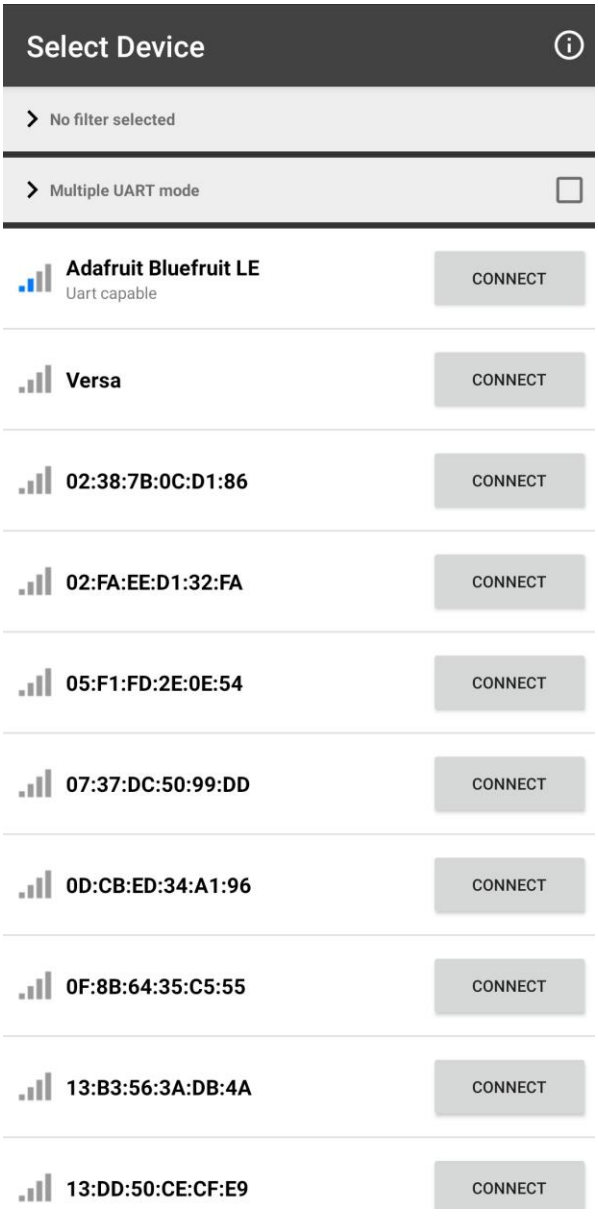
If you instead see this, verify the connections and ensure that the switch on the BLE is set to UART:

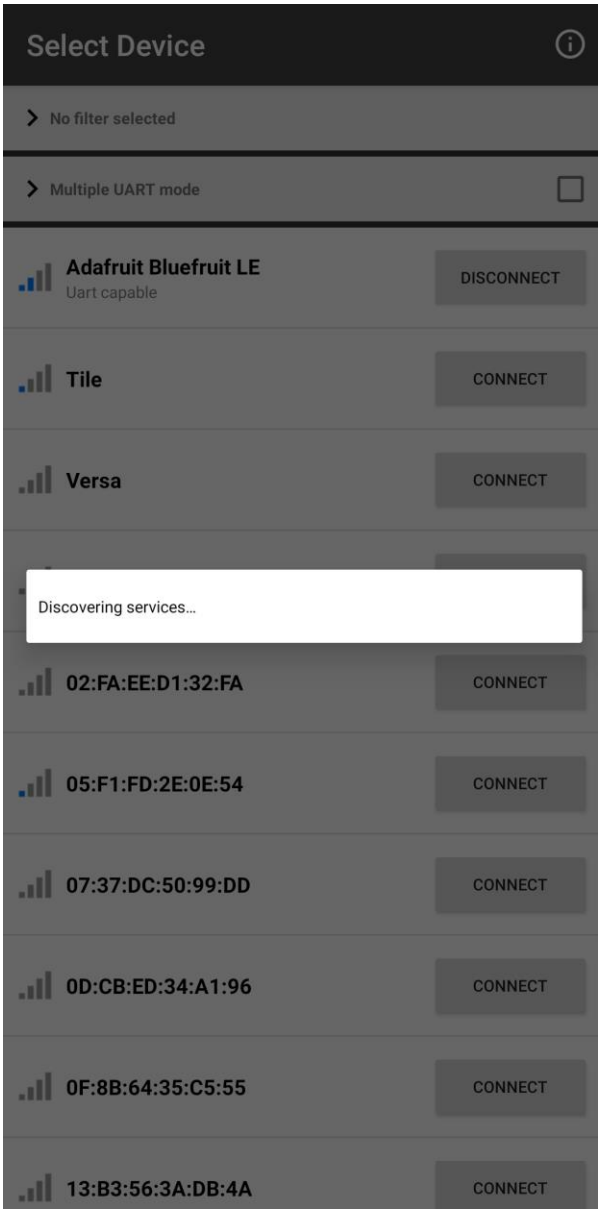
```
Adafruit Bluefruit Command <-> Data Mode Example
------------------------------------------------
Initialising the Bluefruit LE module: ATZ

<- ATZ

<- ATZ

<- ATZ

<- ATZ

<- ATZ

<- ATZ

<- ATZ

<- ATZ

<- ATZ

<- Couldn't find Bluefruit, make sure it's in CoMmanD mode & check wiring?
```

3.2 Transmitting and Receiving Audio

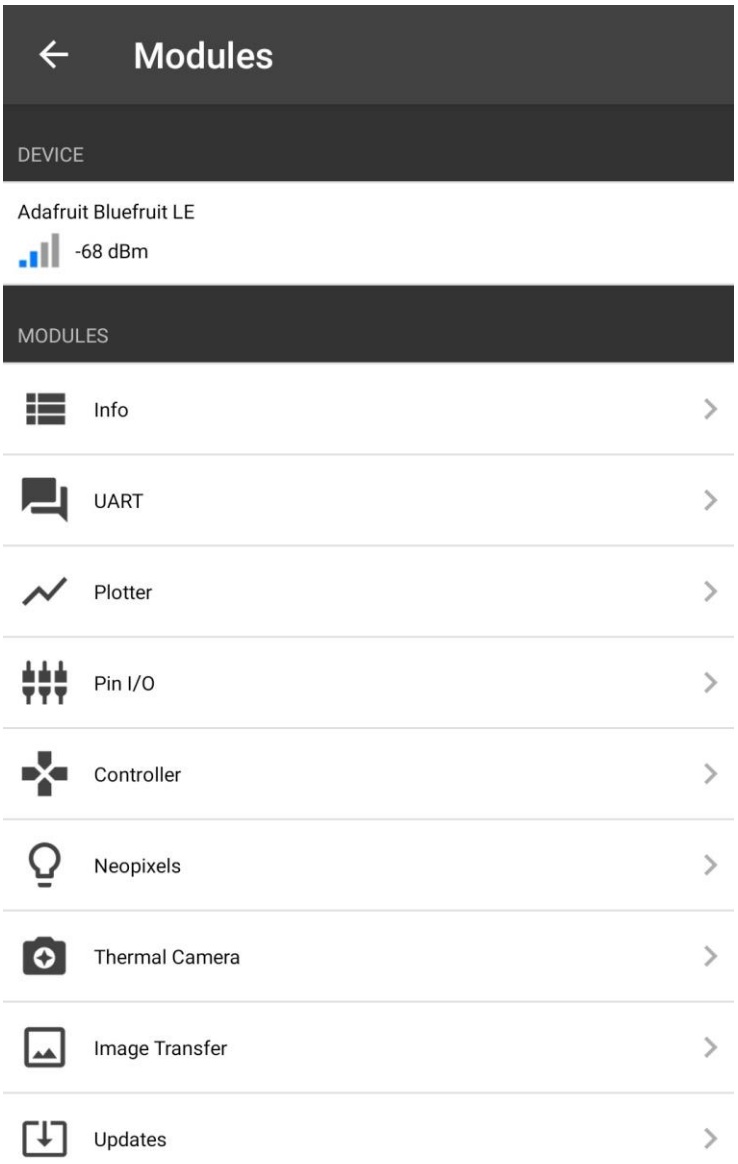
3.2.1 Transmitting Audio Over UART

In the Bluefruit App on your mobile device, refresh devices and search for Adafruit Bluefruit LE then click Connect. Several may show up, make sure that you do not connect to a device that is being used by another team, do this by checking the signal strength to the left of the device.

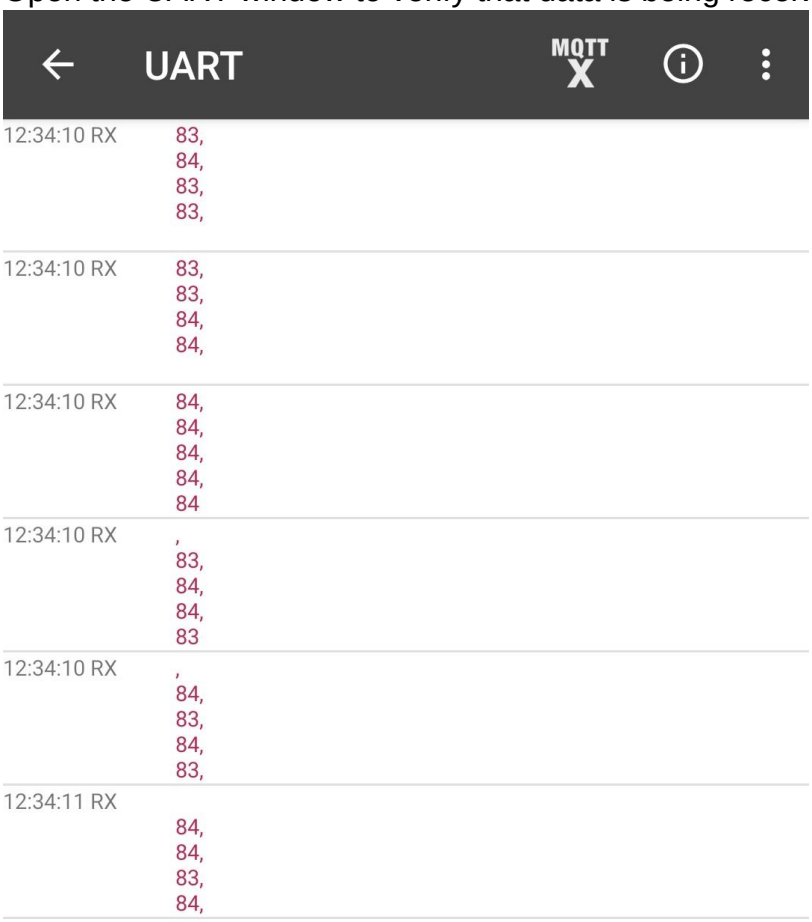




Now with the device connected, you can now look at the device information and use several other features.

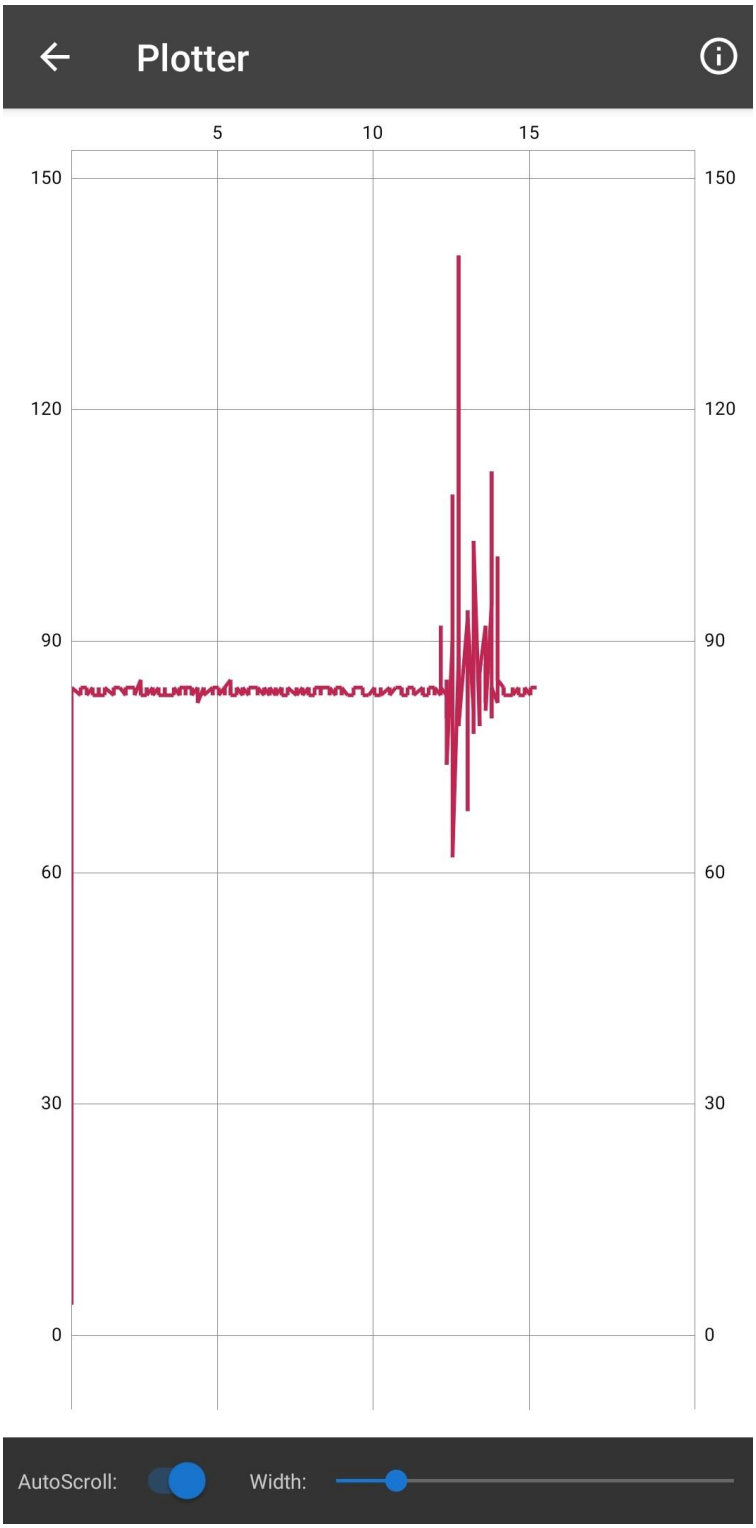


Open the UART window to verify that data is being received from the Bluetooth device.



3.2.2 Viewing Audio Transmission

With a verified stream of data being transmitted from the Arduino over Bluetooth to the mobile device, go to the previous device window and open the Plotter:



To verify that the microphone data is working properly, tap, scrap, or speak into the microphone to watch the audio levels change over time.

4 Study Questions & Deliverables

1. Provide a comprehensive report that demonstrates your completion of this laboratory assignment. Key sections to include in your report are "**Introduction and Background**", "**Methodologies**", and "**Results and Conclusions**" with inclusion of **figures**, **tables**, **codes**, **flowcharts**, and so forth in different sections.
2. What are some possible uses for low bitrate wireless audio transmission?
3. What are the pros and cons of using Bluetooth as opposed to LoRa or XBee?
4. Regarding IoT devices, why would using Bluetooth Low Energy be beneficial over standard Bluetooth?

Instructor: Dr. Shayan (Sean) Taheri.

Note – Cheating and Plagiarism: Cheating and plagiarism are not permitted in any form and cause certain penalties. The instructor reserves the right to fail culprits.

Deliverable: All your responses to the assignment questions should be included in a single compressed file to be uploaded in the Gannon University (GU) – Blackboard Learn environment.

5 Equipment

Name	Quantity
Arduino Mega Microcontroller	1
USB-A to USB-B Cable	1
Male-to-Female Dupont Jumpers	12
Breadboard	1
Bluefruit Low Energy Module	1
Electret Microphone	1
Android or iOS Device	1

Bluefruit Low Energy Module – References:

1. Adafruit Bluefruit LE SPI Friend - Bluetooth Low Energy (BLE) : ID 2633 : $17.50 : Adafruit Industries, Unique & fun DIY electronics and kits

2. Adafruit Bluefruit LE Micro - Bluetooth Low Energy + ATmega32u4 : ID 2661 : $26.95 : Adafruit Industries, Unique & fun DIY electronics and kits

3. Amazon.com: Adafruit Bluefruit LE UART Friend - Bluetooth Low Energy (BLE) : Electronics

4. Bluefruit LE - Bluetooth Low Energy (BLE 4.0) - nRF8001 Breakout - v1. (chicagodist.com)

5. Adafruit Bluefruit LE SPI Friend - Bluetooth Low Botland - Robotic Shop

6. getting-started-with-the-nrf8001-bluefruit-le-brea-932853.pdf (mouser.com)

7. Microsoft Word - Bendable EL Wire (digikey.com)

8. Adafruit Bluefruit LE SPI Friend - Bluetooth Low Energy (BLE) | ADA2633 | Core Electronics Australia (core-electronics.com.au)

9. 2633-Adafruit-Industries-datasheet-62119051.pdf (octopart.com)

10. Bluefruit LE SPI Friend - Bluetooth Low Energy (BLE) :: Micro JPM

11. CC2541 SimpleLink™ Bluetooth® low energy and proprietary wireless MCU datasheet (Rev. D) (ti.com)

12. Bluefruit LE - Bluetooth Low Energy (BLE 4.0) - nRF8001 Breakout (grobotronics.com)

13. Interfacing Adafruit Bluefruit LE SPI Friend with Arduino - Electropeak

14. Very Low Power BLE Made Easy With Arduino -- Part 1 : 15 Steps (with Pictures) - Instructables

Electret Microphone – References:

1. How to Use Microphones on the Arduino - Circuit Basics

2. Electret Mic Breakout Board Hookup Guide - SparkFun Learn

3. How to Use a Microphone with Arduino | Microcontroller Tutorials (teachmemicro.com)

4. Electret Microphone : 7 Steps - Instructables

5. Overview | Adafruit Microphone Amplifier Breakout | Adafruit Learning System

6. Assembly and Wiring | Adafruit Microphone Amplifier Breakout | Adafruit Learning System

7. How to Connect a Microphone to an Arduino (learningaboutelectronics.com)

8. Using a Microphone with Arduino • AranaCorp

9. Electret Microphone Amplifier (mtholyoke.edu)

10. How to Use a Microphone with Arduino? - ElectronicsHacks

11. Interfacing MAX9814 Microphone AGC Amplifier Module with Arduino (electropeak.com)

12. Electret Microphone Amplifier - MAX4466 with Adjustable Gain ID: 1063 - $6.95 : Adafruit Industries, Unique & fun DIY electronics and kits (farnell.com)

13. CMA-4544PF-W Datasheet - Electret Condenser Microphones | Microphones | CUI Devices

14. Electret Microphone - 20Hz-20KHz Omnidirectional : ID 1064 : $1.50 : Adafruit Industries, Unique & fun DIY electronics and kits

15. MAX4466 Electret Microphone Amplifier Adjustable Gain for Arduino, with a 20-20KHz electret microphone soldered on (rees52.com)

16. https://learn.sparkfun.com/tutorials/electret-mic-breakout-boar (digikey.com)

17. 001485300DS01-ml.pdf (conrad.com)

18. adafruit-agc-electret-microphone-amplifier-max9814.pdf (electrokit.com)