Arduino Compatible Microphones

Testing for use with Sound Reactive Displays on the Arduino Platform

By: Andrew Tuline

Date: April 13, 2020

Initial Release V1.01

Table of Contents

Table of Figures

Introduction

This is a review of various microphone/pre-amp boards for the Arduino. What is included:

- Analog microphones with built-in pre-amp.
- Sound sensors.

My goal is to determine which analog microphones can be used in conjunction with WLED on the ESP8266, the ESP32 and Nano platforms as I have over 40 portable WLED controlled lanterns, about 5-10 of which will be sound reactive. The questions to answer are how well do these microphones work, how do they work with WLED on the ESP platforms with my sound sampling routines in WLED (currently ASound1 through ASound9), and how reliable is the ESP8266 at sampling analog values.

Testing includes:

- Testing on ESP8266, ESP32 and Nano using analogRead(MIC_PIN).
- Measuring background noise levels (i.e. INMP401 on Nano quiet).
- Measuring loud signals (i.e. INMP401 on Nano noisy).
- Testing with sound reactive routines written for WLED (summarized in table only).
- All microphones powered with 3.3V.
- Sampling rate is ~1Khz. These are not frequency reactive.

What is NOT included:

- Frequency domain testing, both FFT libraries and the MSGEQ7 chip.
- Digital I2S microphones. That'll be for another day (hopefully soon).
- Testing with 3.5mm line-in (which works well with sound reactive WLED).
- This is not a comprehensive series of tests.

Links:

- Andrew's sound testing GitHub: https://github.com/atuline/Arduino-Sampling
- Andrew's sound reactive WLED fork: https://github.com/atuline/WLED

Acknowledgements

This project is self sponsored/published, however I'd like to thank the Maker community and the WLED and FastLED communities for their continued Open Source efforts and the manufacturers who make related products available for our use. Particular thanks to the WLED Discord community for their enthusiasm and feedback in the creation of this document.

ESP8266, ESP32 and Arduino Nano

Both the ESP8266 and the Arduino Nano have 10-bit A/D converters, thus providing 10 bits of sampling resolution. The ESP32, on the other hand has several 12-bit A/D converters. During testing, the ESP8266 exhibited unexpected anomalies when the room was quiet and the results are very noticeable in many of the accompanying figures. As a result, I would not recommend that platform for accurate A/D usage when Wi-Fi is enabled. See this GitHub issue:

https://github.com/esp8266/Arduino/issues/2070

This will be further investigated in the '**ESP8266 Anomalies**' section later in this document.

Table of Microphones Used

Model	Type	Comment	WLED Compatibility
INMP401	Analog mic/pre-amp	Test before use!	Good.
MAX4466	Analog mic/pre-amp	Good!	Good.
MAX9812	Analog mic/pre-amp	Only 20dB gain.	OK.
MAX9814	Analog mic/pre-amp	Configure for 40dB gain.	OK.
Sparkfun electret	Analog mic/pre-amp	High background noise.	Increase squelch.
SPW2430	Analog mic/pre-amp	No amplification.	Not good.
LM393 sensor	Sensor	Digital output, adjustable.	Marginally OK.
KY-038 sensor	Sensor	Little amplification.	Not good.

Considerations

The Arduino Nano (where most of my sound development was done) as well as the ES8266 each have a 10 bit A/D converter that can return values ranging from 0 to 1023, whereas the ESP32, with 12 bit can range from 0 to 4095.

In order to find the volume of a sample, we must first determine the voltage when there is no sound. On the Arduino Nano, we:

- Connected the Vin of the microphone to the 3.3V pin
- Connected 3.3V to the AREF pin
- Added to the setup function: analogReference(EXTERNAL);

Experimentation determined that the 'no sound' value returned with a Serial.println(MIC_PIN) was about 510. As a result, we would typically 'zero' out the samples and get their absolute value by:

```
int micIn = analogRead(A5);
micIn = micIn – 510;
micIn = abs(micIn);
```

Center point – Early sampling routines for the INMP401 on the Nano used that hard-coded DC offset value of 510 to get the '0' value of the signal, however an averaging routine will perform this DC offset calculation automatically. This is critical when using different microphones or microcontrollers, each with a different center point and output range. The calculation requires a floating-point variable and is:

```
static float micLev = ((micLev * 31) + micIn) / 32;
```

Background Noise – The INMP401 has a background noise typically ranging from 504 through 516 with the Nano. With a center point of ~510, the absolute value of amplitude of those sample signals is about 6. These results are similar using the ESP8266, so in my WLED sound sampling routines, I have configured a **squelch** value of 10, below which any samples are set to '0'. This works well for the INMP401, MAX4466 and MAX9812 microphones. The MAX9814 works well only if you connect VDD to the Gain pin in order to reduce gain from 60dB (default) to 40dB. Otherwise, it will require an increased squelch value. The Sparkfun Electret microphone has MUCH higher background noise. The only way to account for that increased noise would be to either change the squelch value at compile time for the Electret, or to create a field in WLED's HTML to allow for an adjustable squelch value.

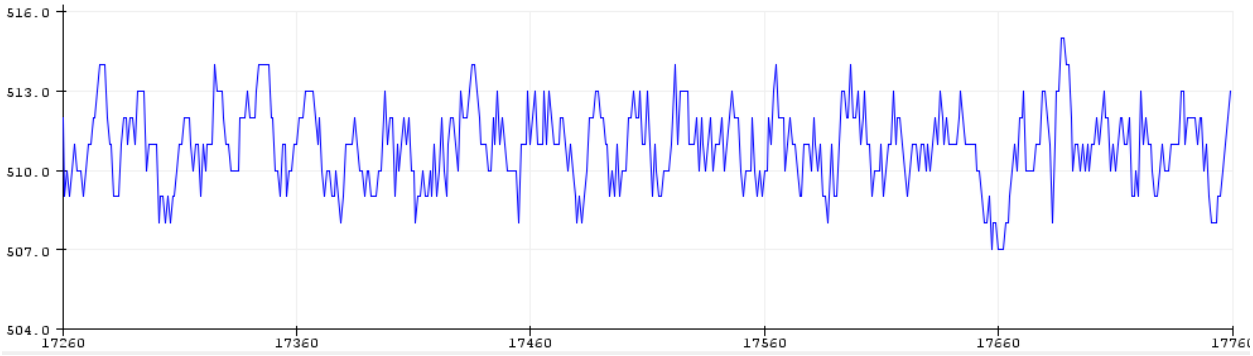
We also need to account for the ESP32's 12 bit A/D values, and can do so by dividing the results by 4. We just need to add a pre-processor directive in the code if an ESP32 is present.

Amplification – As long as there was a significant voltage range (up to 3.3V) for the built-in A/D converter when sampling and the background noise was minimal, then most of these microphone/pre-amp combination boards should be compatible with the WLED sound sampling routines. Unfortunately, the SPW2430 and the KY-038 did not have adequate amplification and will not work with my routines.

Measurement Example

This section describes the measurements taken for each microphone. The first is in a quiet environment in order to determine the background noise, while the second is in a loud environment.

Figure 1 - INMP401 on Nano quiet



The ‘y’ (or vertical) axis above is the linear Serial.println() output of analogRead(MIC_PIN). The ‘x’ axis is time in microseconds and is not really used except for finding maximums and minimums over a period of time.

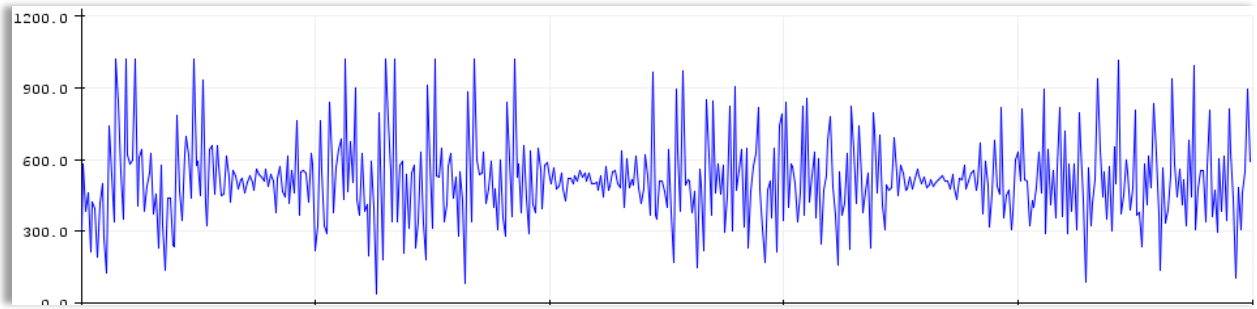
Center: 511, Min: 507, Max: 514

Using the simple arithmetic from the considerations section:

Noise: abs(507 – 511) = **4** or abs(514 – 511) = 3

In this environment, you could use a squelch value of 4 plus a safety factor to get rid of background noise. After testing with several (good) INMP401 microphones, I use a default value of 10 with WLED.

Figure 2 - INMP401 on Nano noisy



In the above figure, I'm making loud, but inconsistent noise (these tests are not rigorous).

Center: ~500, Noise: 4, Min: 50, Max: 1000, Vol max: 500

As long as a microphone has sample readings greater than the squelch, we'll be able to get something out of it. Different WLED sound reactive animation routines may use:

- Peak detection
- Current sample
- Automatic gain control of current sample

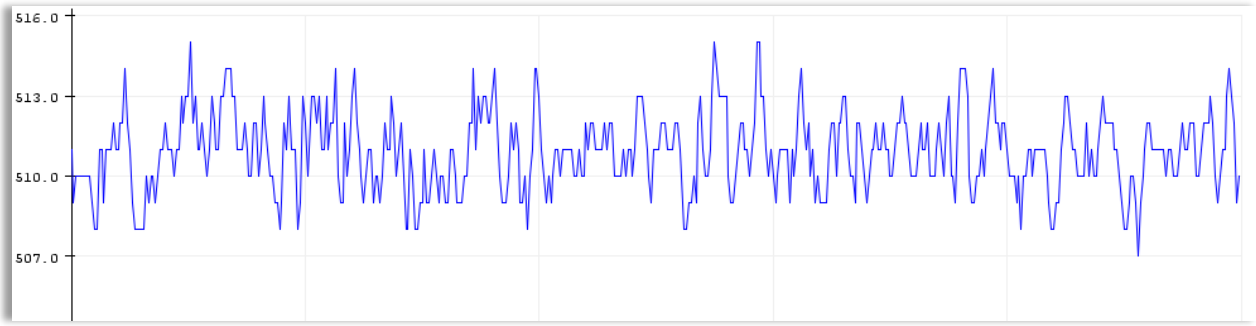
INMP401 Microphone

See: https://www.sparkfun.com/products/9868

These are available from both Sparkfun as well as AliExpress.

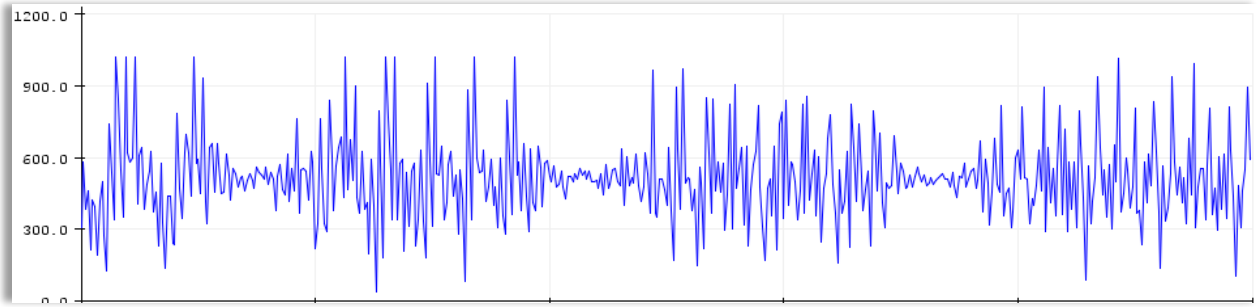
- Based on the ADMP401 microphone with 67db gain.
- Chinese ones can be sketchy. Some are very noisy, and some don't center correctly.
- Otherwise, this **has** been my primary microphone for the Arduino.

Figure 3 – INMP401 on Nano quiet



This microphone centers around 510 with background noise ranging from 504 through 516. This one is from Sparkfun and is probably the best INMP401 microphone I have.

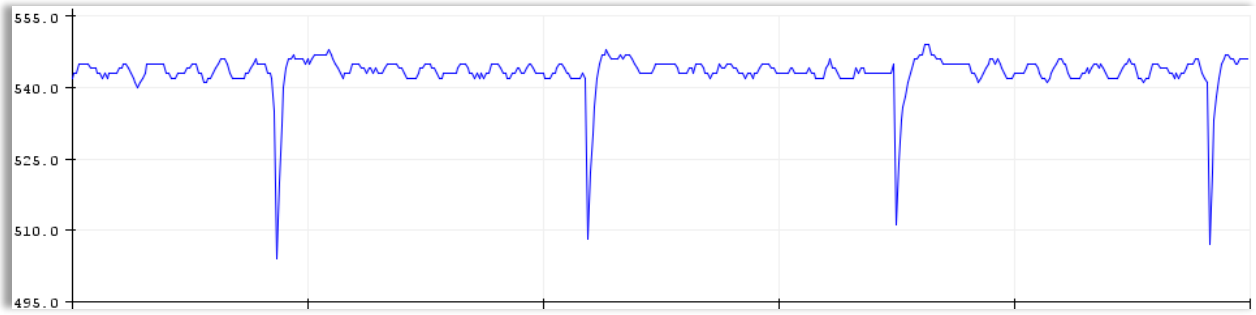
Figure 4 - INMP401 on Nano noisy



Exhibits a good range of values from about 20 to 1000, so a volume of about ~500 and can easily be used for sound reactive displays.

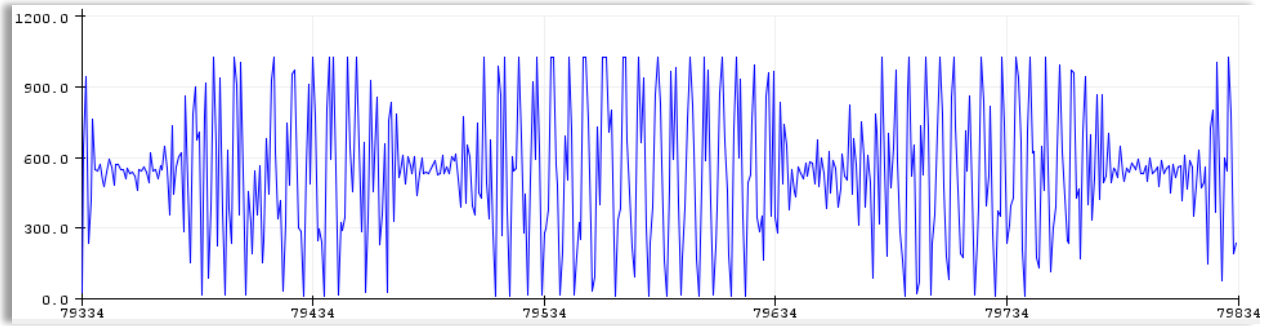
Center: ~510, Noise: ~6, Min: 50, Max: 1000, Vol max: 500

Figure 5 - INMP401 on ESP8266 quiet



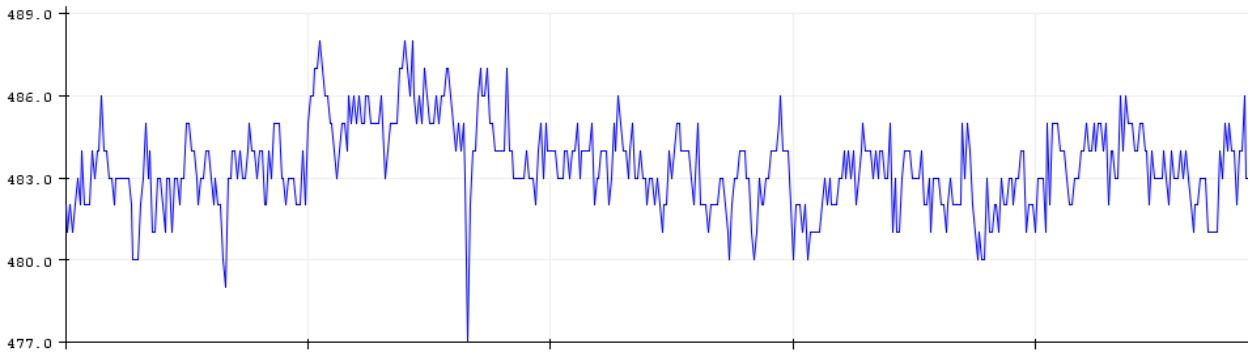
The ESP8666 alternated between the spikes displayed above and the expected response centering around 510 that we see on the Nano. Very disconcerting and repeatable across multiple tests, even with small changes to the code. It didn't really show up on the WLED display though, so that's a relief.

Figure 6 - INMP401 on ESP8266 noisy



Exhibits a good range of values and can easily be used for sound reactive displays. I'm wondering what happened to the spikes.

Figure 7 - Clone INMP401 on Nano quiet



I have several INMP401 microphones from AliExpress (from more than 1 batch). The quality of several of these microphones from AliExpress made them all but useless. Test them BEFORE you put them into production.

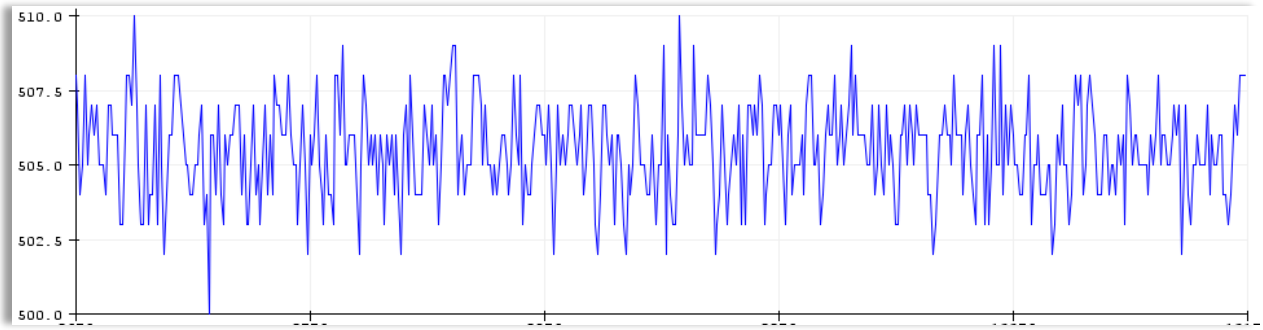
As much as I've used the INMP401 microphone in various projects, I've had to be very careful to pick and choose microphones that worked correctly. The results of these tests may push me towards another microphone, probably the MAX9814 or the MAX4466.

MAX4466 Microphone

See: https://www.adafruit.com/product/1063

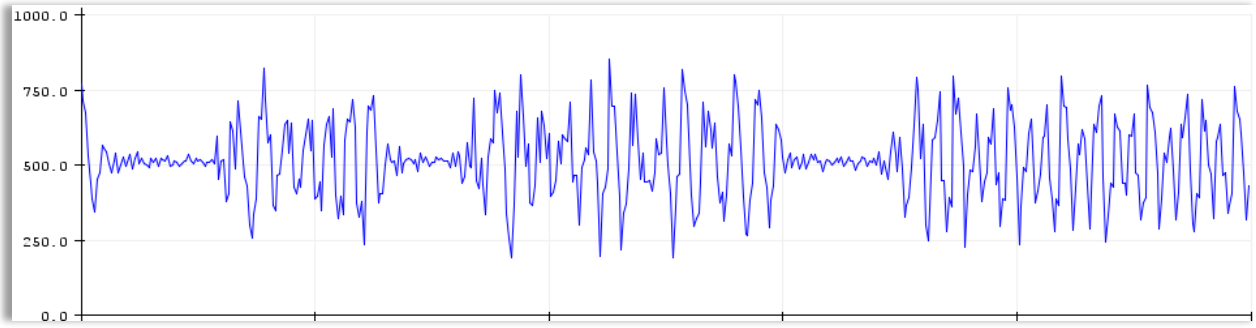
This microphone has a small potentiometer on the back to adjust the gain. Without any adjustments, this microphone tests OK and works fine with our sound reactive routines.

Figure 8 - MAX4466 on Nano quiet



The center is slightly off from the INMP401, however the automatic sound centering in the routines will take that into account. Otherwise the background noise is acceptable.

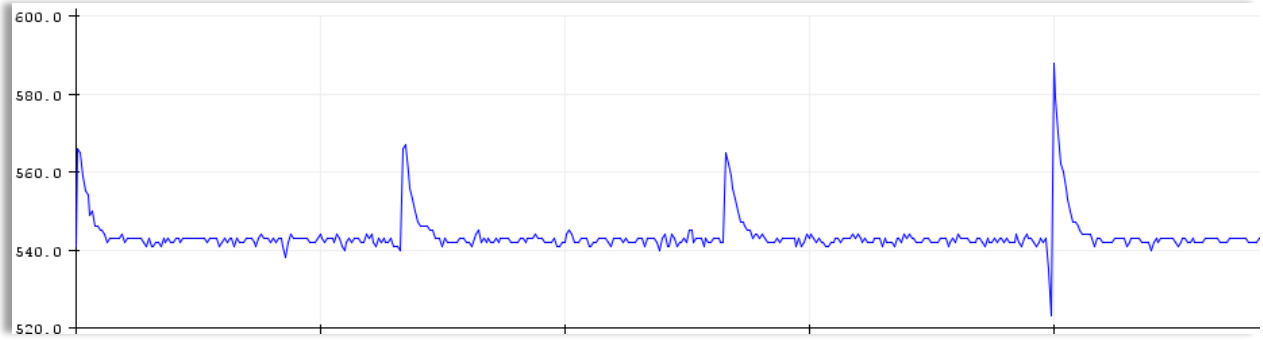
Figure 9 - MAX4466 on Nano noisy



These exhibit a reasonable range of values for use with sound reactivity.

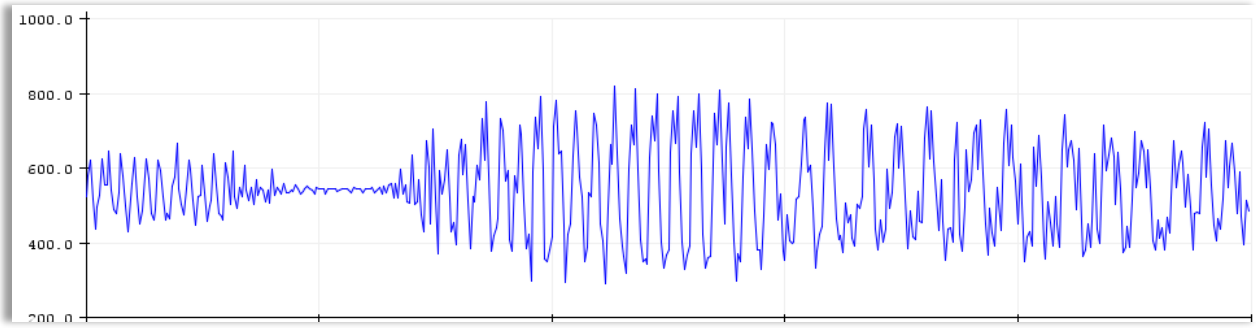
Center: ~505, Noise: ~6, Min: 250, Max: 800, Vol max: ~300

Figure 10 - MAX4466 on ESP8266 quiet



These spikes are apparently not just an INMP401 thing, but generic to the ESP8266. It's interesting that they don't look the same as the INMP401. I really need a digital storage scope and a signal generator.

Figure 11 - MAX4466 on ESP8266 noisy



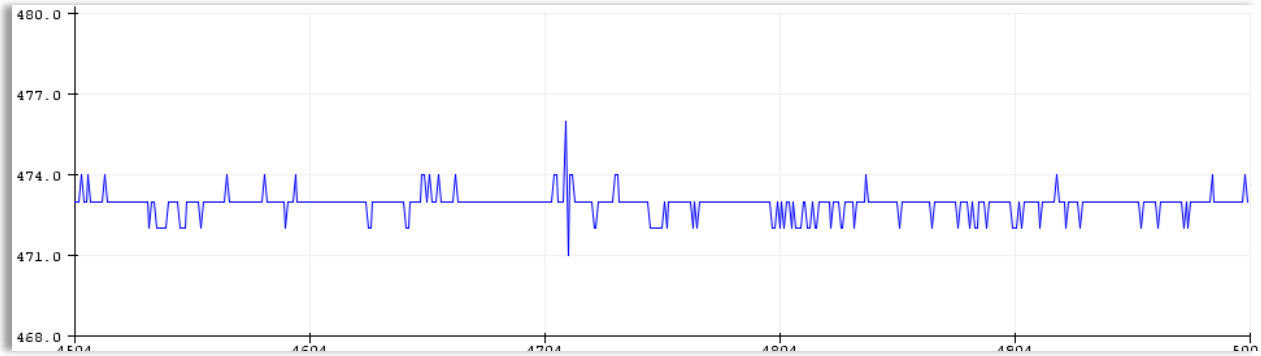
Yep, this still looks good for sound reactive routines.

MAX9812 Microphone

See: https://leeselectronic.com/en/product/16042.html

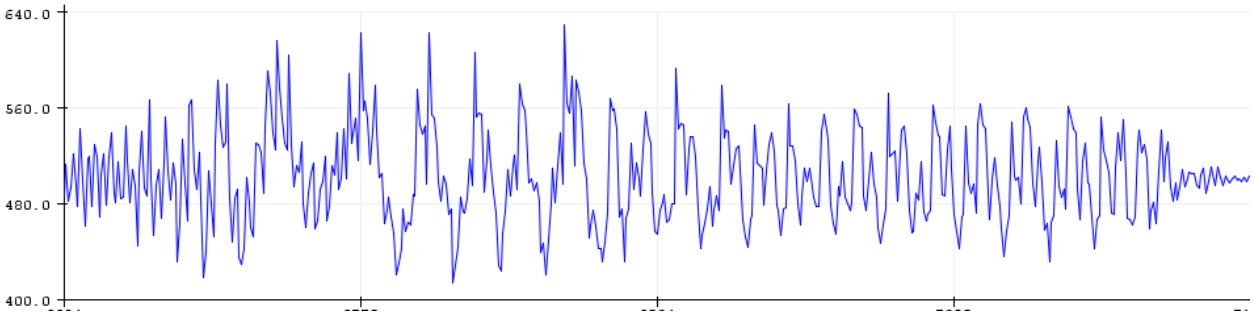
Based on the MAX9812 chip, this microphone has a marginal 20dB gain, but appears to be no longer sold on AliExpress and has probably been replaced by the MAX9814.

Figure 12 – MAX9812 on Nano quiet



There's hardly any background noise on this microphone, which is due to the 20dB amplification.

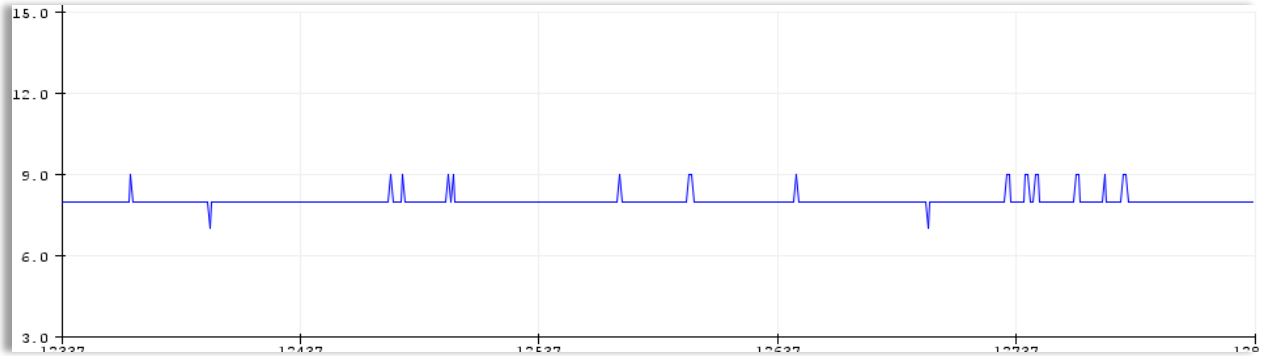
Figure 13 - MAX9812 on Nano noisy



It doesn't appear to have a great amplification, but it appears to be acceptable nonetheless.

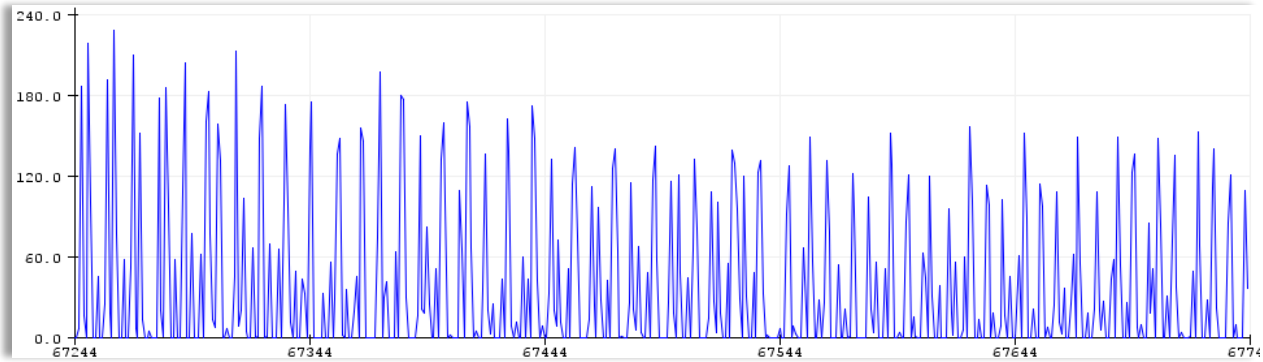
Center: 473, Noise: 2, Min: 420, Max: 600, Vol max: 120

Figure 14 – MAX9812 on ESP8266 quiet



Look at this! It's centered a lot closer to 0, and no spikes either.

Figure 15 – MAX9812 on ESP8266 noisy



The amplification is marginally acceptable for use with my WLED sound reactive routines.

MAX9814 Microphone

See: https://www.adafruit.com/product/1713

This seems to be a pretty popular board and there are several vendors on AliExpress that sell it. Adafruit mentions that it has Auto Gain Control, but then the board talks about jumpers, so I'm not sure where that 'Auto' part comes in. Based on the measurements here, I recommend tying the Gain pin to Vdd for decreased sensitivity. The first measurements taken use 60dB gain and the background noise is too great to work with my sound reactive routines unless you increase the squelch value. I later tested with 40dB gain, which was much better.

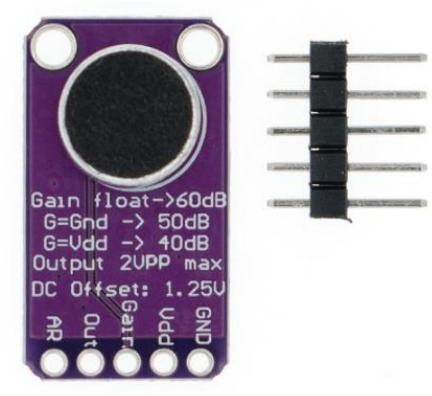
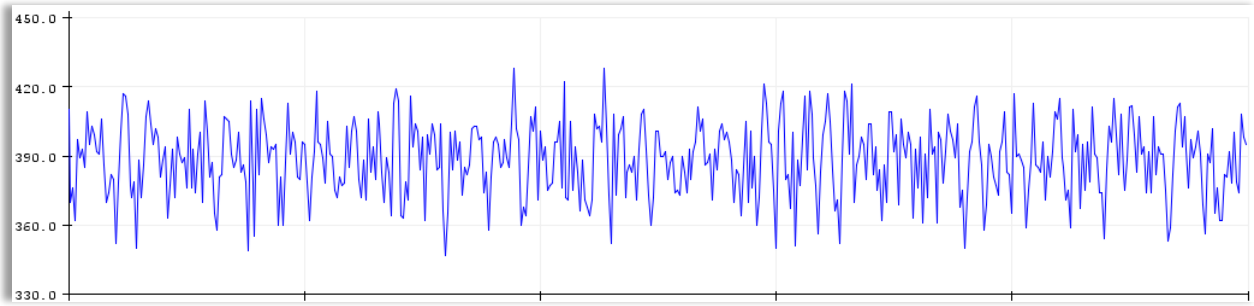
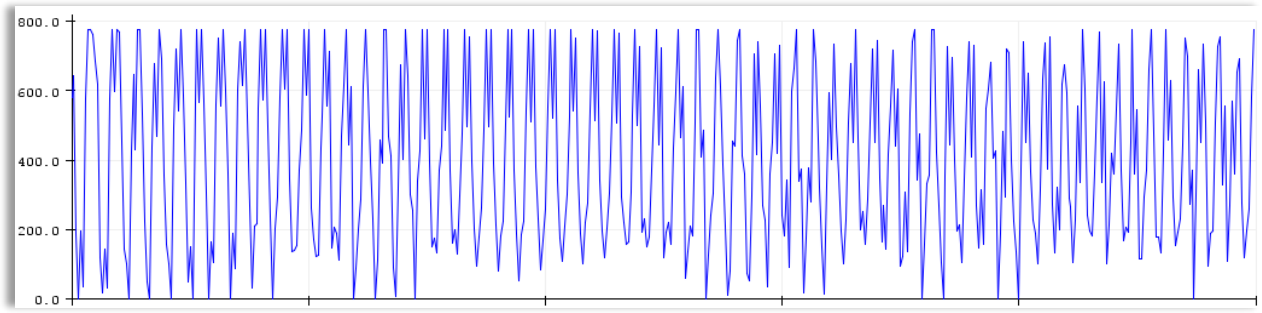


Figure 16 - MAX9814 on Nano quiet



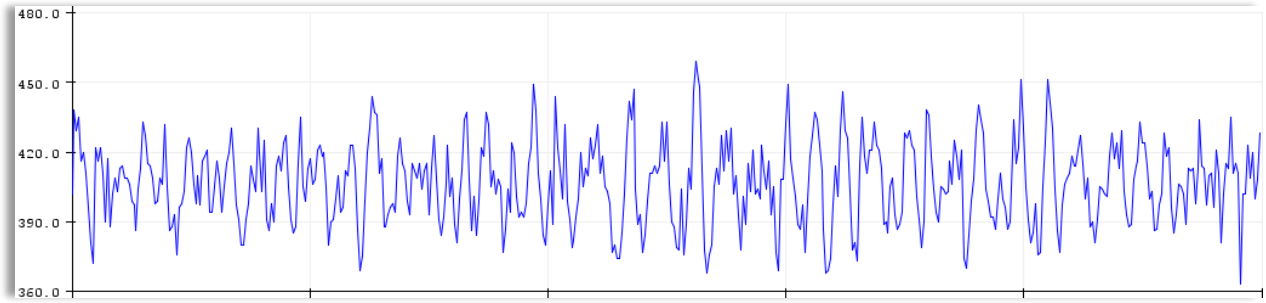
As shipped, this microphone has FAR more background noise than I am happy with. Either way, this requires much more squelch than other microphones.

Figure 17 - MAX9814 on Nano noisy



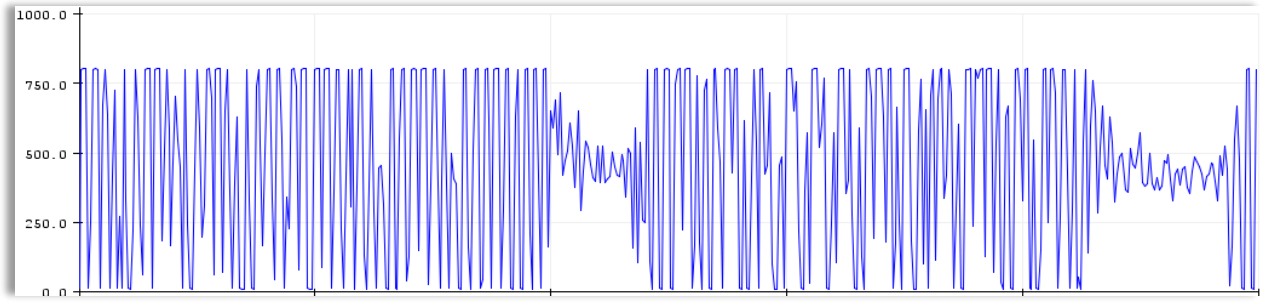
Center: 385, Noise: 35, Min: 0, Max: 780, Vol max: 400

Figure 18 - MAX9814 on ESP8266 quiet



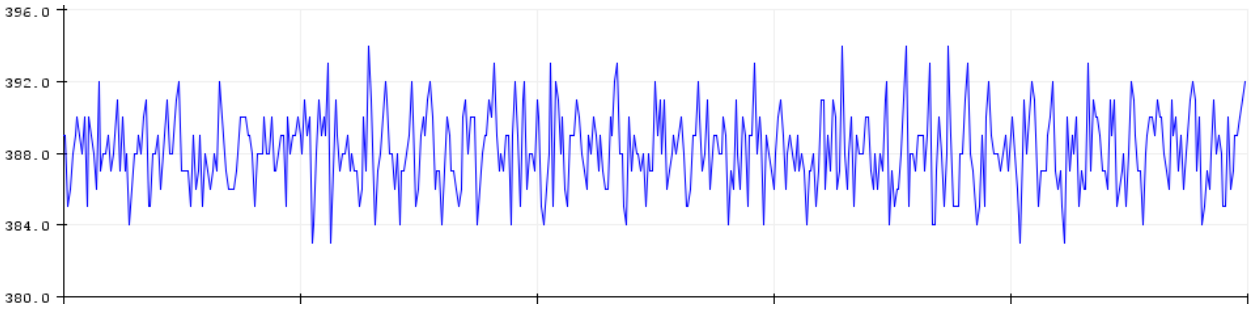
Again, there's far too much background noise detected for these to be used if we can't adjust our squelch value.

Figure 19 - MAX9814 on ESP8266 noisy



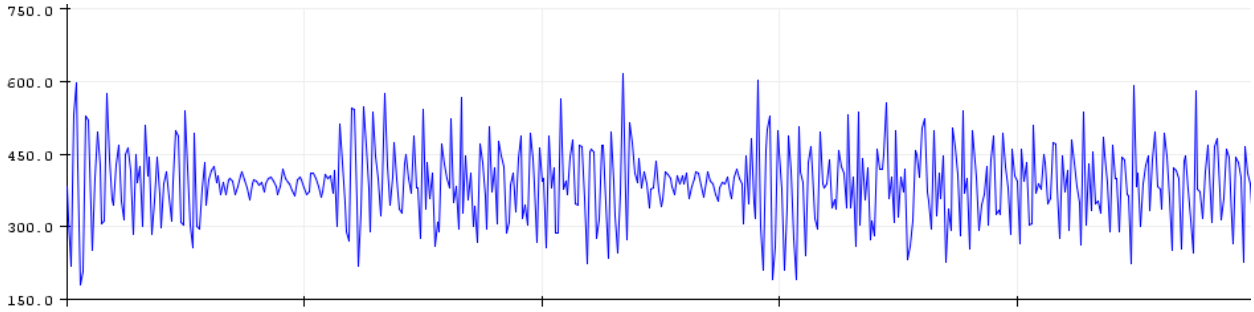
As a result of the high background noise values with the MAX9814 microphone, the squelch value would need to be adjusted significantly. Although this can be changed at compile time in WLED, there is no slider to change the squelch at run time.

Figure 20 - MAX9814 on Nano quiet (VDD to Gain)



If you connect VDD to the Gain pin, amplification is adjusted to 40dB and the MAX9814 will then work well with my sound reactive WLED routines. This is looking to be a real good microphone if you configure it appropriately.

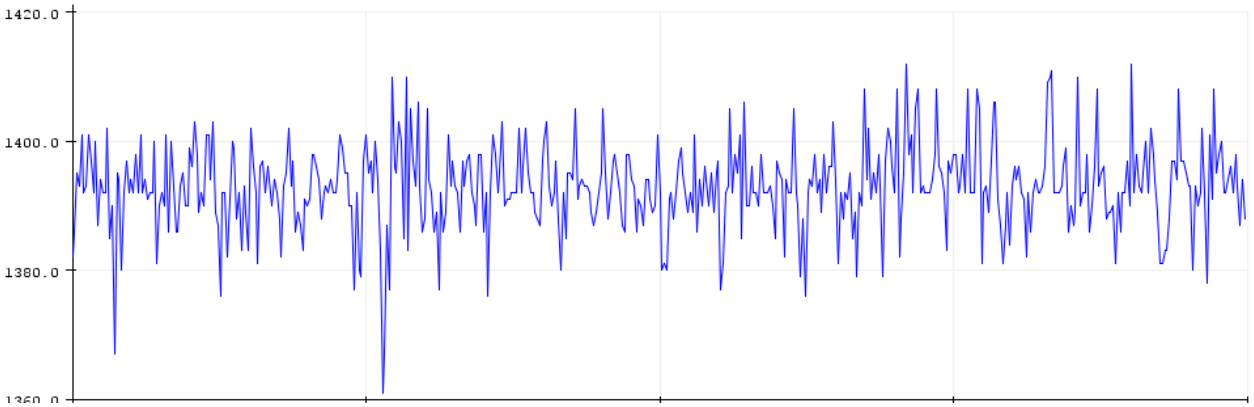
Figure 21 - MAX9814 on Nano noisy (VDD to Gain)



Not as much gain as I would like, but it now works well with WLED.

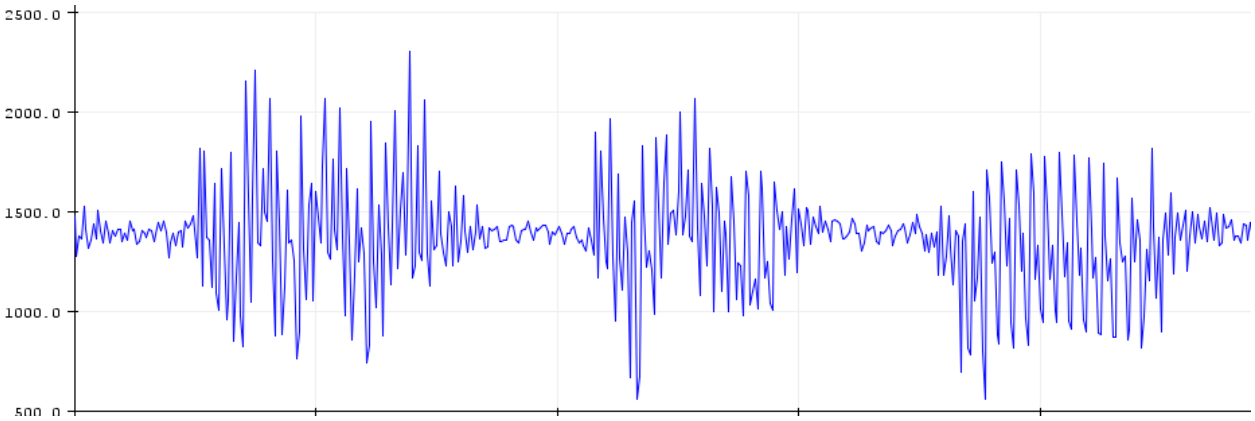
Center: 388, Noise: 4, Min: 180, Max: 600, Vol max: 220

Figure 22 - MAX9814 on ESP32 quiet (VDD to Gain)



After multiple samples, I have determine that the ESP32 doesn't exhibit the same anomalies that the ESP8266 does.

Figure 23 - MAX9814 on ESP32 noisy (VDD to Gain)



Center: 1390, Noise: 15, Min: 700, Max: 2000, Vol max: 700

Even if you divide the values by 4, we won't match the INMP401 'quiet' value, which makes the auto-centering calculation critical for determining volume in the sampling routines.

Further notes on MAX9814

Here are a couple of YouTube videos on comparing the MAX9814 in different configurations:

- https://www.youtube.com/watch?v=3rUBtNlOT2I
- https://www.youtube.com/watch?v=5vY2BWPqv6U

Sparkfun Electret Microphone Breakout

See: https://www.sparkfun.com/products/12758

This simple microphone uses an OPA344 amplifier. It doesn't appear to have any sellers on AliExpress.

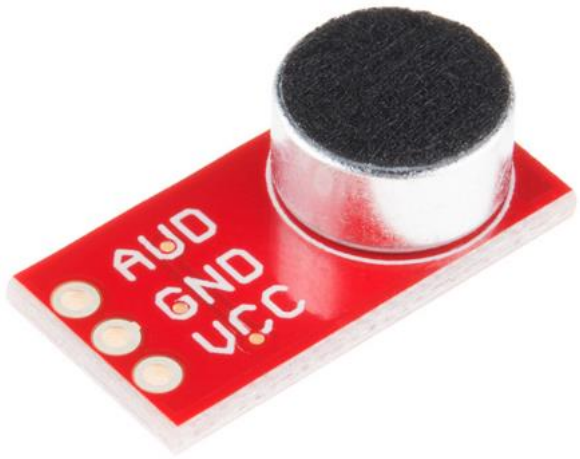
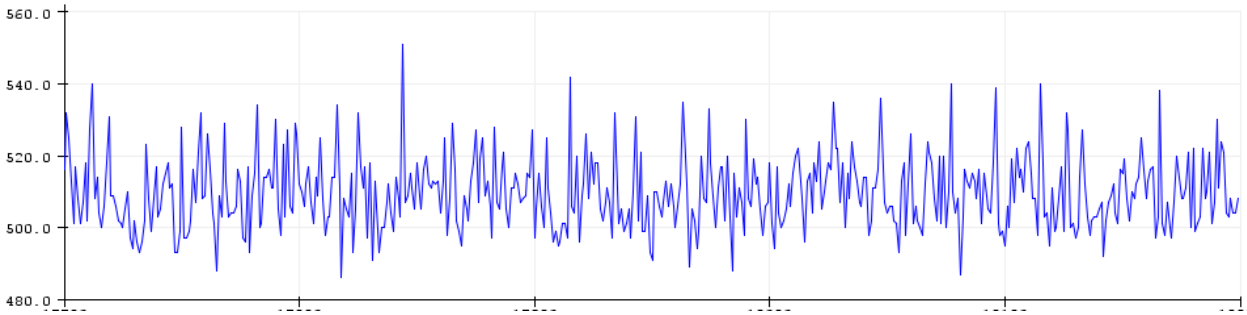
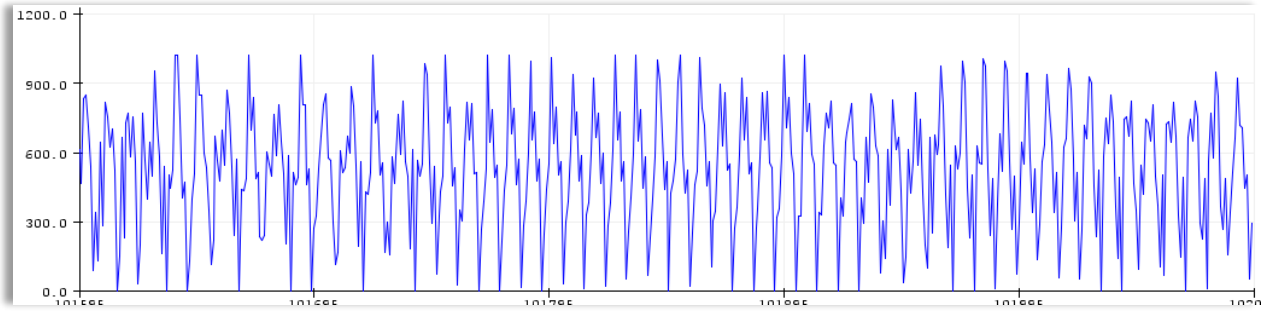


Figure 24 - Electret on Nano quiet



This will require additional squelch to remove the background noise.

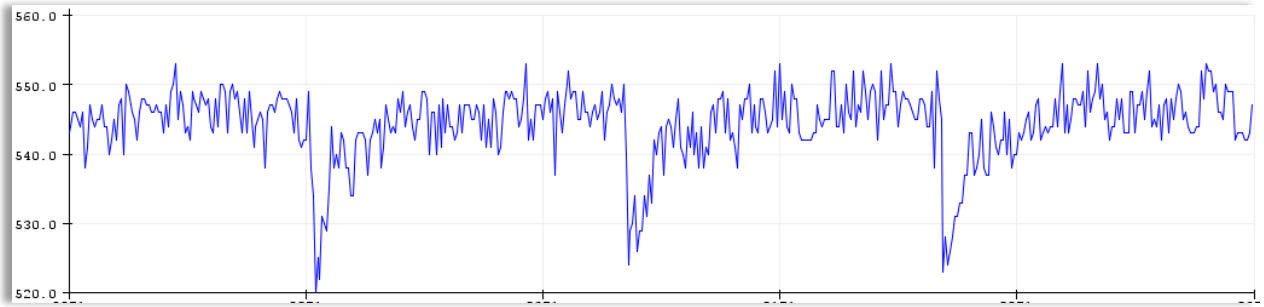
Figure 25 - Electret on Nano noisy



This has a reasonable level of sensitivity, but the background noise value is significant.

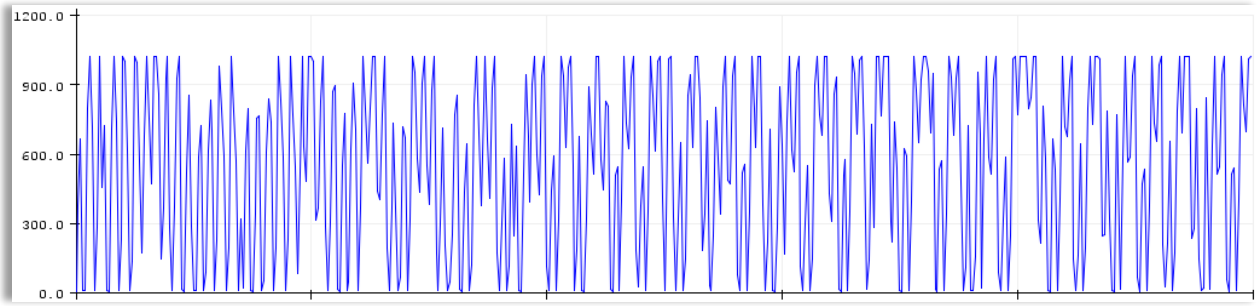
Center: 510, Noise: 20, Min: 0, Max: 1000, Vol max: 500

Figure 26 - Electret on ESP8266 quiet



There's those spikes again.

Figure 27 - Electret on ESP8266 noisy



Overall a good microphone, however it requires more squelch than what is used by several other microphones. We would need to increase the squelch value at compile time.

SPW2430 MEMS Microphone

See: https://www.adafruit.com/product/2716

This microphone has both an AC output and a DC output. We'll use the DC output.

Figure 28 - SPW2430 DC pin on Nano quiet

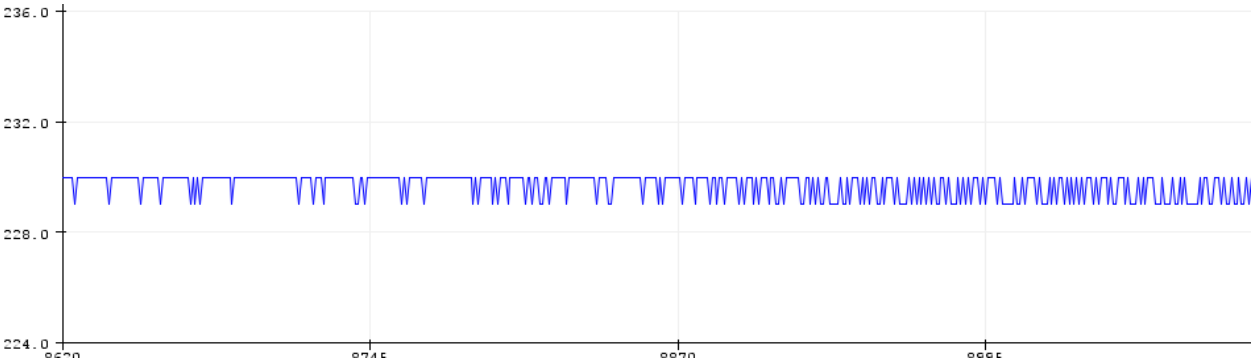
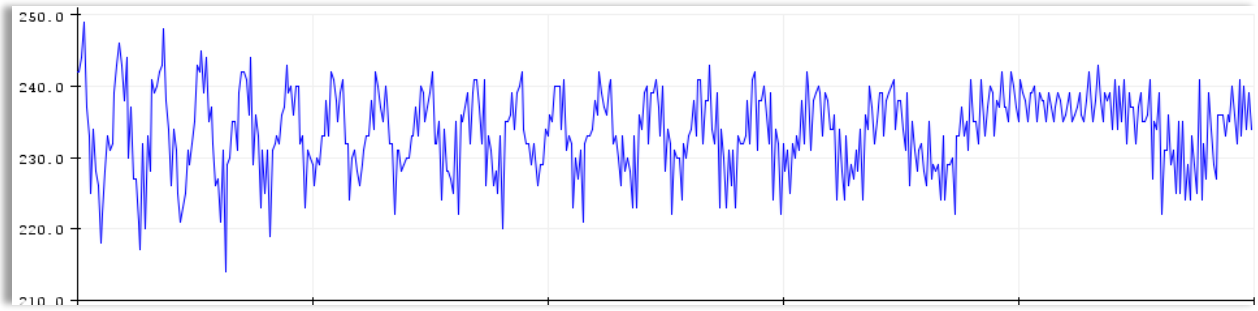


Figure 29 - SPW2430 DC pin on Nano noisy



This board has minimal noise, but also minimal amplification. Combined with the default squelch value of 10, this board won't work well at all.

Center: 230, Noise: 2, Min: 218, Max: 245, Vol max: 15

Figure 30 - SPW2430 DC pin on ESP8266 quiet

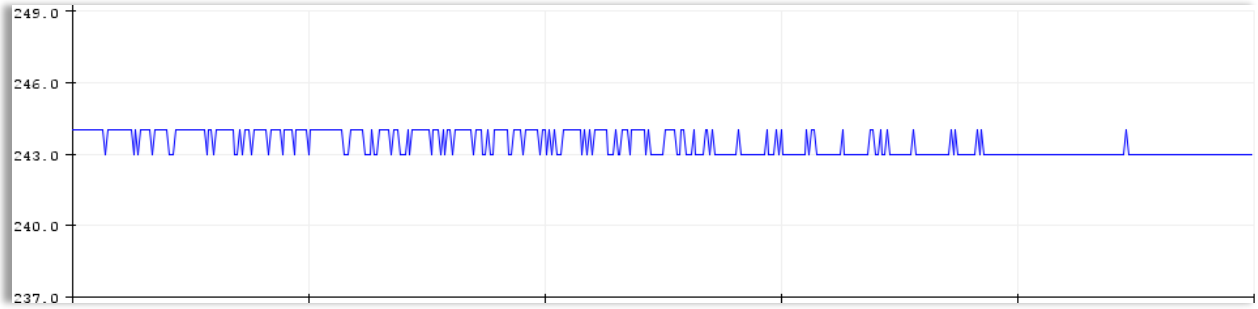
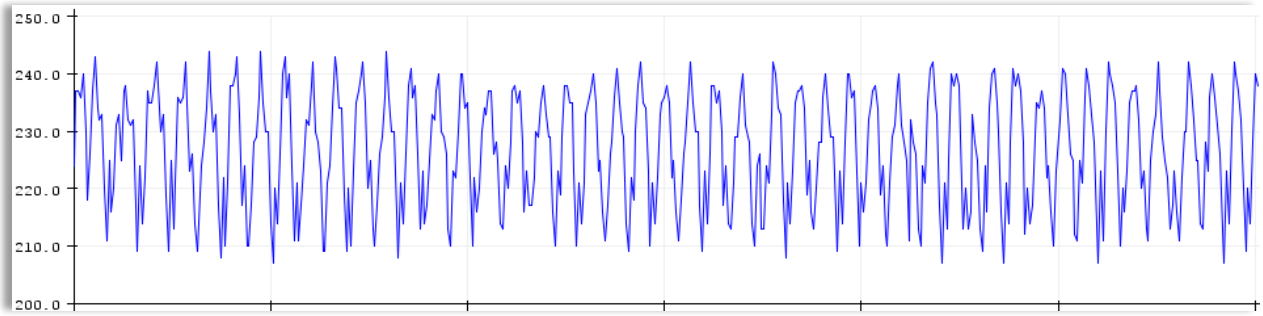


Figure 31 - SPW2430 DC pin on ESP8266 noisy



This has minimal amplification and is not appropriate for our use.

LM393 Sound Sensor

This is a digital sound sensor, as opposed to an analog amplifier. The potentiometer needs to be adjusted to a sensitivity appropriate to the environment. I'm not providing a link for it, because I don't recommend it for sound reactive projects.

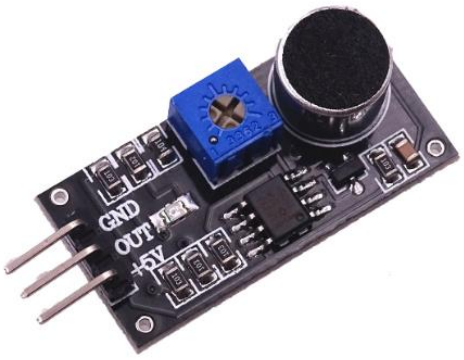
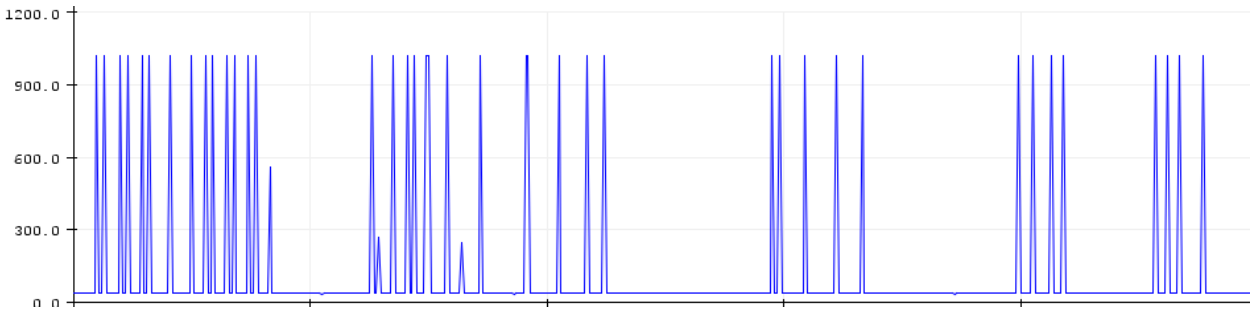


Figure 32 - LM-393 Nano quiet/noisy



Surprisingly, this microphone worked to a degree when used with the sound reactive WLED code. I wouldn't recommend it for general sound reactive use.

Cannot use standard centering/volume measurements.

KY-038 Microphone Sound Microphone/Sensor

I'm not going to publish a link for this sensor either as it's probably the worst of the lot for sound reactive usage. I connected the AC output of the sensor to the microcontroller.

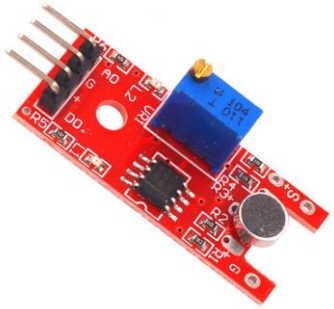
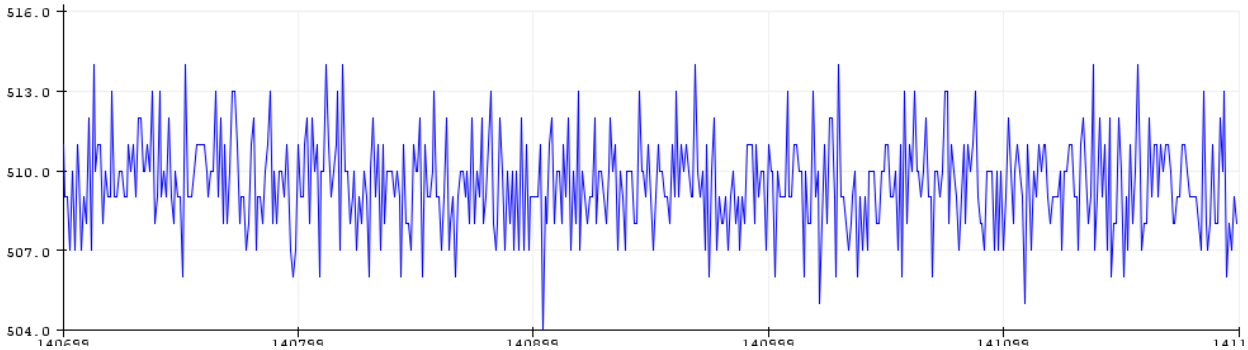
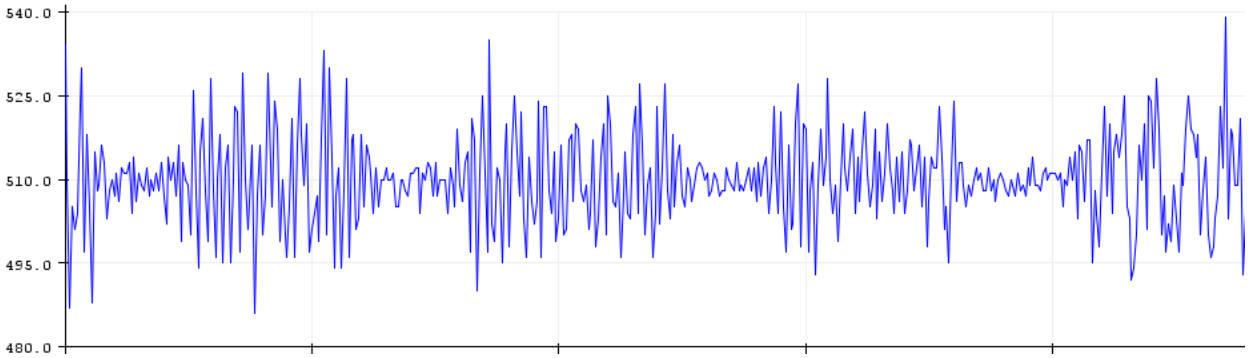


Figure 33 - KY-038 Nano quiet



I turned the potentiometer counter clockwise until the green output reached 512.

Figure 34 - KY-038 Nano noisy



This has nowhere near the amplification of other microphones.

Center: 510, Noise: 3, Min: 490, Max: 525, Vol max: 20

Results

The following table show the center point and volume range for each tested microphone.

Model	Center	Noise	Volume	WLED Compatibility
INMP401	510	6	~500	Good.
MAX4466	505	6	~300	Good.
MAX9812	473	2	~120	OK.
MAX9814	385	35	400	Increase squelch.
MAX9814 40dB	388	4	200	OK.
Sparkfun electret	510	20	500	Increase squelch.
SPW2430	230	2	15	Not good.
LM393 sensor	0	n/a	1023	Digital output only. Kind of works
KY-038 sensor	510	3	20	Not good.

Conclusions

The INMP401 (when they work) and the MAX4466 microphones both work well out of the box. The MAX89812 could do with the squelch value reduced as it does not have a lot of amplification but should otherwise be OK. The MAX9814 and the Sparkfun Electret both have significant background noise, however the MAX9814 can be jumpered to provide a lower gain, thus lower background noise. I wouldn't recommend the other devices for sound reactive use.

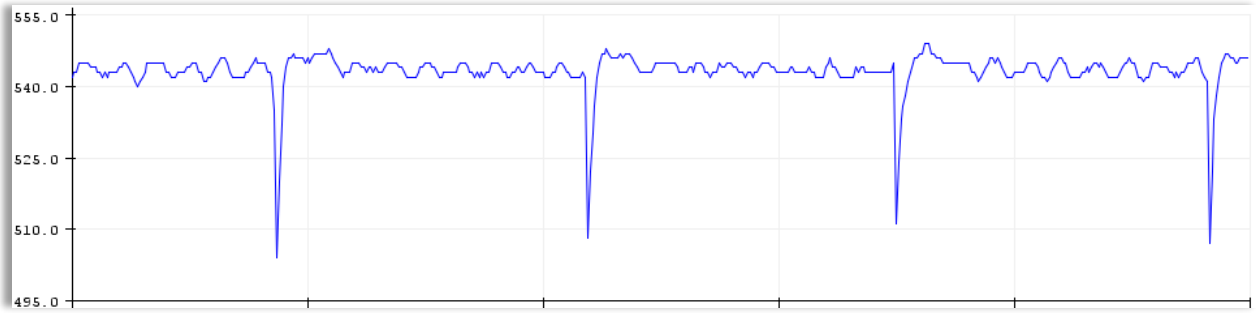
Currently, the MAX9814 and the MAX4466 are widely available and at great prices.

ESP8266 Anomalies

As we saw earlier, we have encountered some issues with sampling on the ESP8266. Let's delve into that further.

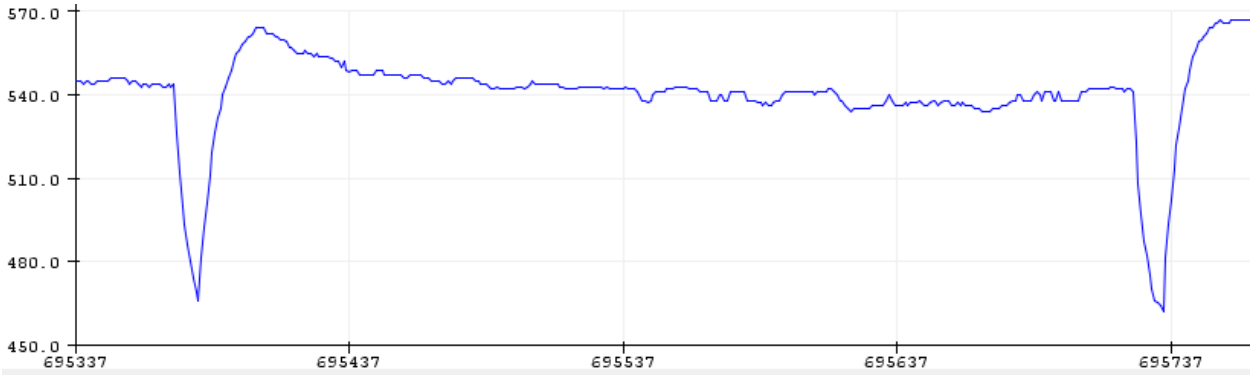
First off, we're sampling with a 1ms delay, which puts our sample rate at about 1KHz. This will provide anomalous readings at around that frequency, but should not provide the issue that we're seeing:

Figure 35 - INMP401 on Nano quiet



The 1ms delay has been added in order to allow the ESP8266 watchdog to function, or the device may crash and reboot. If we replace the delay(1) with a yield(), we get the following similar results:

Figure 36 - INMP401 on Nano quiet with yield()



From the X axis, it appears that the timing for these is:

695737 – 695387 = 350 or .35mS

Surprisingly, removing the yield statement did not crash the ESP8266, and yielded the same results.

Let's follow the GitHub issue at: https://github.com/esp8266/Arduino/issues/2070

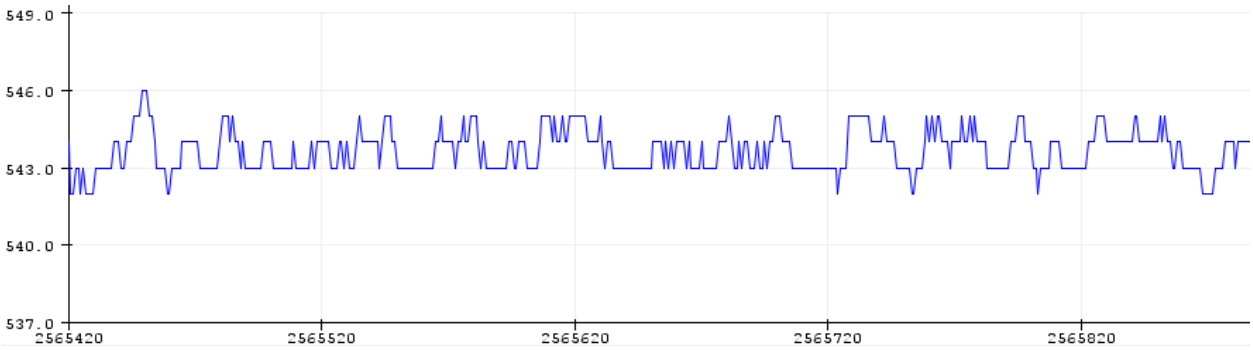
As mentioned in the link, the issue may be related to the Wi-Fi power draw (or other internal issue), so the following was added to the code below:

#include <<ESP8266WiFi.h>

and in setup:

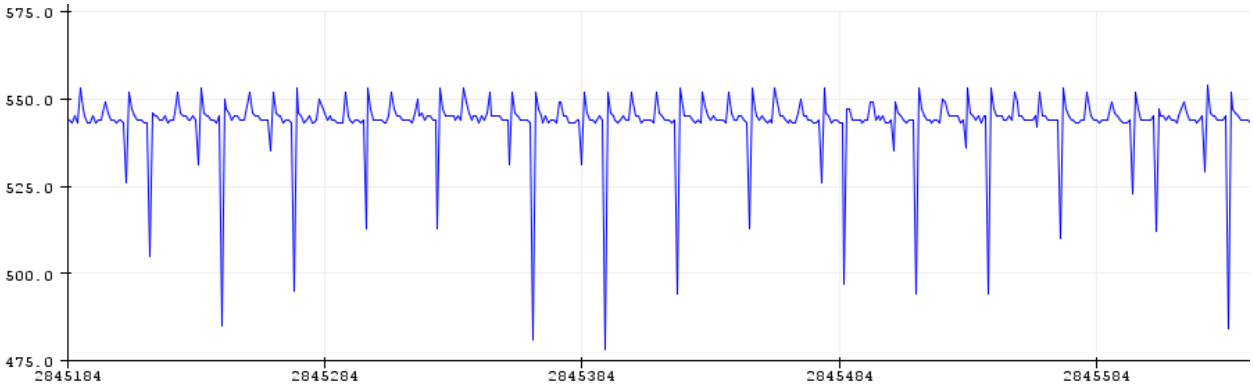
WiFi.disconnect()

Figure 37 - INMP401 on Nano quiet and Wi-Fi disabled



That cleared everything up, however that mode of operation won't work while using WLED. Let's see what the ESP8266 sampling routine looks like while using WLED. I have a delay(10) in there, which will slow things down.

Figure 38 - INMP401 on Nano quiet running WLED



Commenters in the previously mentioned thread thought that the Wi-Fi issue was related to power drain and then powered the ESP8266 directly on the 3.3V of the board. This still resulted in the same jumpy values. This issue never was resolved.

The good thing is that the sound reactive visuals don’t exhibit this same jumpiness, so I’ll continue to work on sound reactive coding with the internal ADC for the ESP8266 platform.

The Code

```
/* Basic sound sampling
 *
 *  By: Andrew Tuline
 *
 *  Date: April, 2020
 *
 */

#ifdef ESP8266
#define MIC_PIN A0
#endif

#ifdef ESP32
#define MIC_PIN 36
#endif

#ifdef __AVR__
#define MIC_PIN A5
#endif

void setup() {
  Serial.begin(115200);

#ifdef __AVR__
  analogReference(EXTERNAL);
#endif

  delay(1000);

} // setup()



void loop() {
  Serial.println(analogRead(MIC_PIN));
  delay(1);
} // loop()
```