P524: Survey of Instrumentation and Laboratory Techniques
Week 8

10/15/2024

Week 8: sensors-5

Measuring sound waves:

Electret microphone MAX4466

- a 20Hz-20KHz electret microphone soldered on board.
- a small trimmer pot to adjust the gain. You can set the gain from 25x to 125x
- For the amplification, we use the Maxim MAX4466, an op-amp specifically designed for this delicate task! The amplifier has excellent power supply noise rejection, so this amplifier sounds really good and isn't nearly as noisy or scratchy as other mic amp breakouts we've tried!

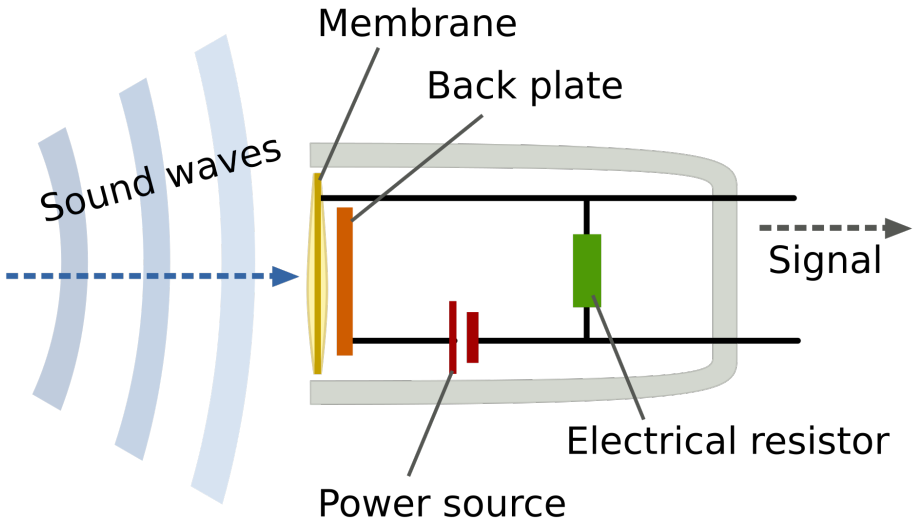
MAX4466 preamp:

https://www.analog.com/en/products/max4466.html#part-details

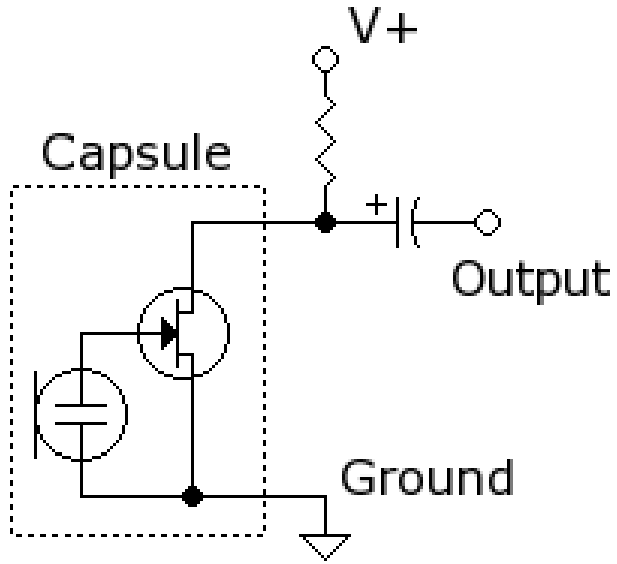
VCC: 2.4-5VDC
OUT: the voltage waveform out from the microphone! The voltage will range from 0V to VCC.

The output will have a DC bias of VCC/2 so when it's perfectly quiet, the voltage will be a steady VCC/2 volt (it is DC coupled).
If AC coupled audio equipment is used, place a 100uF capacitor between the output pin and the input of your device.

Microphones



- **Capacitor microphone** or **electrostatic microphone**—The diaphragm acts as one plate of a capacitor (historically called condensers), and audio vibrations produce changes in the distance between the plates. Because the capacitance of the plates is inversely proportional to the distance between them, the vibrations produce changes in capacitance. $\Delta V = Q/\Delta C$. These changes in capacitance are used to measure the audio signal.
  - The assembly of fixed and movable plates is called an *element* or *capsule*.
- **Electret microphone:** The externally applied charge used for a conventional condenser microphone is replaced by a permanent charge in an electret material.
  - An electret is a ferroelectric material that has been permanently electrically charged or *polarized*. The name comes from *electrostatic* and *magnet*; a static charge is embedded in an electret by the alignment of the static charges in the material, much the way a permanent magnet is made by aligning the magnetic domains in a piece of iron.
  - The electret's constant charge eliminates the need for the polarizing power supply required for non-electret condenser microphones, though a preamplifier is typically incorporated to boost the audio voltage signal.



Making sound

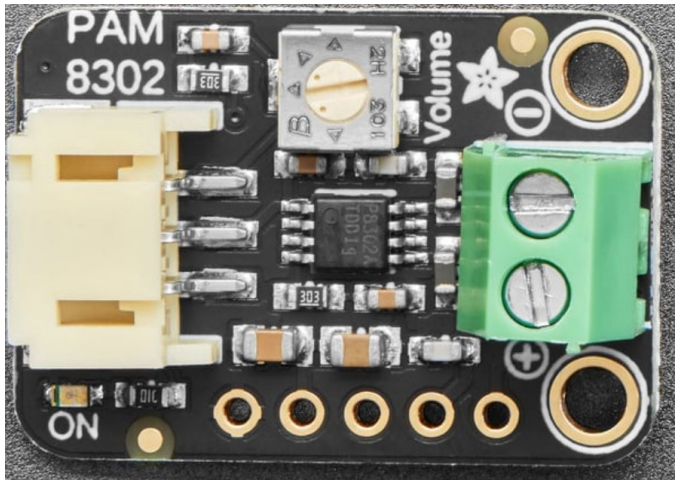
Making sound waves:

Mini Metal Speaker (8 ohm 0.5 W)

- Arduino analog out or digital out to make a voltage waveform.

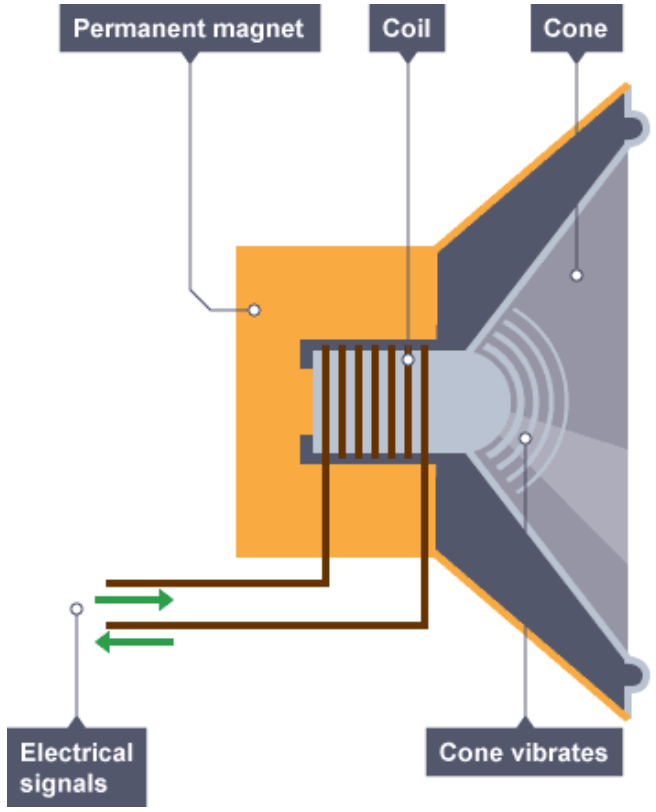
For the amplification, we can use PAM8302, 2.5W Class D Mono Audio Amp:

•Output Power: 2.5W at 4Ω, 10% THD, 1.5W at 8Ω, 10% THD, with 5.5V Supply
•50dB PSRR at 1KHz
•Filterless design, with ferrite bead + capacitors on output.
•Fixed 24dB gain, onboard trim potentiometer for adjusting input volume.
•Thermal and short-circuit/over-current protection
•Low current draw: 4mA quiescent and 0.5mA in shutdown (due to pullup resistor on SD pin)



Loud Speakers

- Alternating current supplied to the loudspeaker creates sound waves in the following way:
  - A current in the coil creates a magnetic field.
  - The magnetic field interacts with the permanent magnet generating a force, which pushes the cone outwards.
  - When the current reverses, the cone is pushed inwards.
  - Alternating current, driven by the DAC, can create sound waves.
- A piezoelectric element may be driven by an oscillating electronic circuit or other audio signal source, driven with a piezoelectric audio amplifier. Sounds commonly used to indicate that a button has been pressed are a click, a ring or a beep.
- Piezoelectric speakers have several advantages over conventional loudspeakers: they are resistant to overloads that would normally destroy most high frequency drivers, and they can be used without a crossover due to their electrical properties. There are also disadvantages: some amplifiers can oscillate when driving capacitive loads like most piezoelectrics, which results in distortion or damage to the amplifier. Additionally, their frequency response, in most cases, is inferior to that of other technologies, especially with regards to bass and midrange. This is why they are generally used in applications where volume and high pitch are more important than sound quality.

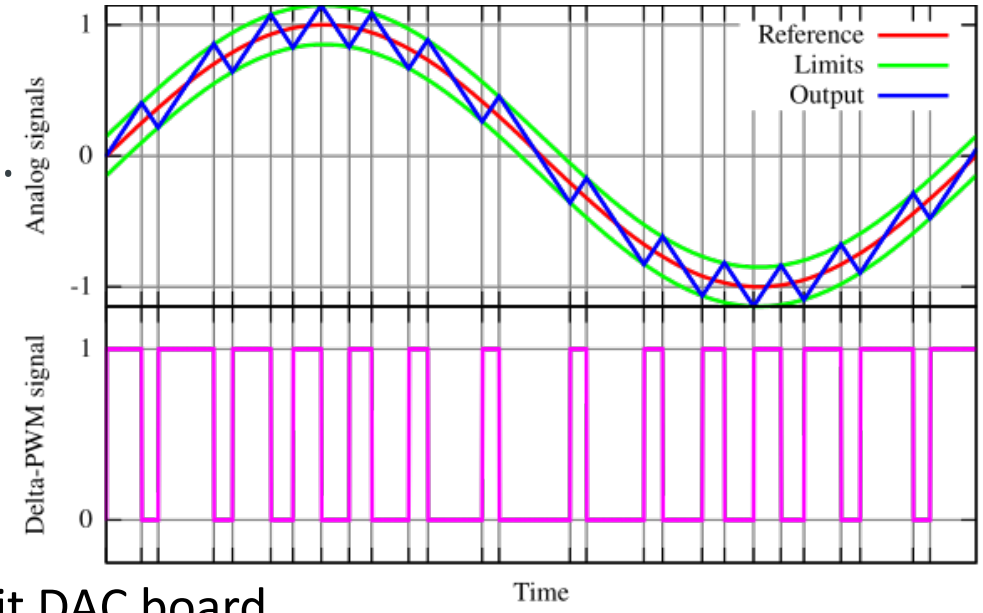


Sound Generation using Arduino

1. **Tone(pin, frequency, duration)**
   - Generates a *square wave* of the specified frequency (and 50% duty cycle) on a pin. A duration can be specified, otherwise the wave continues until a call to **noTone()**. The pin can be connected to a piezo buzzer or other speaker to play tones.
   - Only one tone can be generated at a time. If a tone is already playing on a different pin, the call to tone() will have no effect. If the tone is playing on the same pin, the call will set its frequency.

2. **AnalogWrite(pin, value)**
   - Pulse width modulation (PWM) output from digital pins.
     - Mega: 2 - 13, 44 – 46
     - Adalogger: nearly all pins can do PWM output
   - DAC (true analog output)

```
// SYNTAX
pinMode(DAC1, OUTPUT);
analogWrite(DAC1, 1024);
```



   - Arduino has no built-in DAC, could connect to a MCP4725 12-bit DAC board
   - Adalogger M0: A0 pin, 10-bit analog output. You can set the raw voltage to anything from 0 to 3.3V, unlike PWM outputs this is a true analog output

Exercises (Tuesday)

1. Measure sound waves w/ the electret microphone ***MAX4466***
2. Making a tone w/ the Speaker (mini metal speaker)
    1. Compare the sound of 440Hz, generated using tone() and a sine wave using analogWrite()
    2. Compare connecting Arduino output to the speaker with and without the PAM8302 amplifier
3. Measure the *rms (root-mean-square)* noise of your microphone.

Homework: Triangulation

First, synchronize your boards' clocks (using the GPS module is recommended). Next, place two microphones at different distances from a speaker. Measure the phase lag between the signals at the two microphones. Does this phase lag agree with your expectations based on the speed of sound and distance?

Challenge: with a third microphone, triangulate the location of the speaker. How close do you get?

Fourier expansion

- Just like Taylor expansion, which expand an arbitrary function with a series of power laws,

$$f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \cdots = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n.$$

we can expand any function with a series of sine and cosine functions.

Fourier series, sine-cosine form

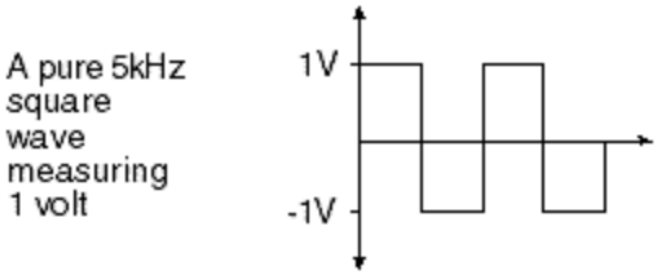
$$s_N(x) = A_0 + \sum_{n=1}^{N} \left(A_n \cos\left(2\pi \frac{n}{P} x\right) + B_n \sin\left(2\pi \frac{n}{P} x\right) \right)$$

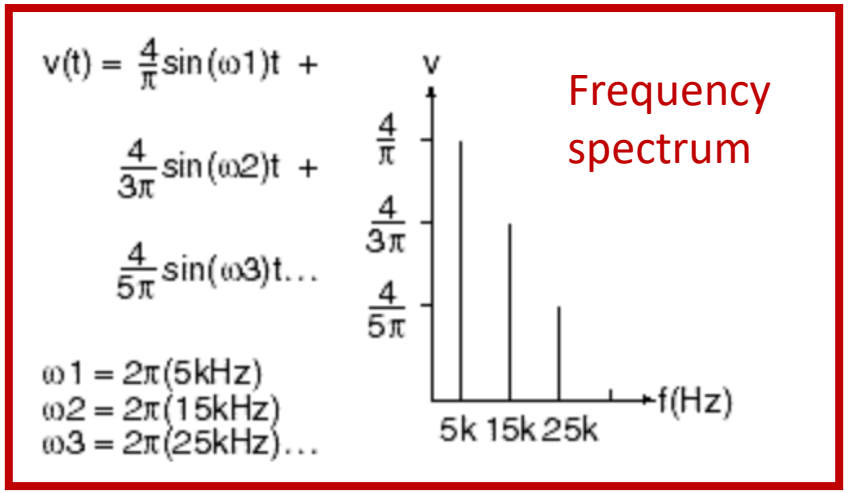
Making a square wave:

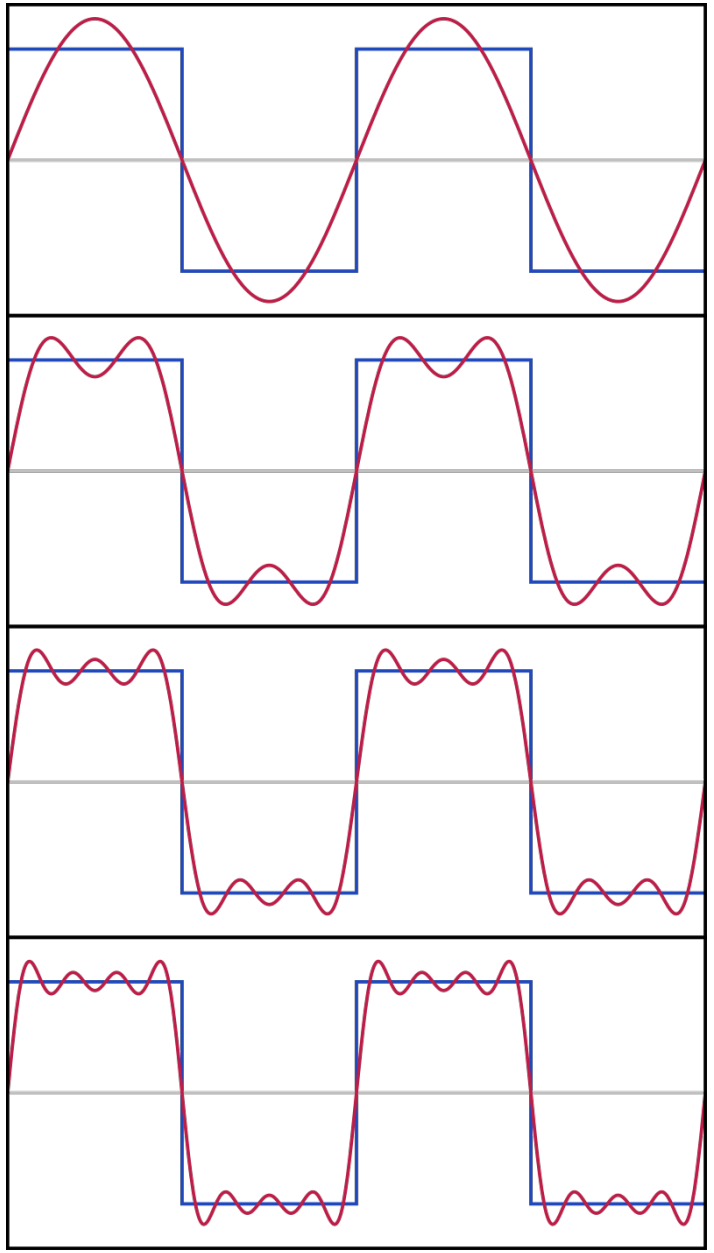
It is hard to Taylor expand a square function, but a square wave can be easily assembled by a series of sinusoidal waves:

$$x(t) = \frac{4}{\pi}\sum_{k=1}^{\infty}\frac{\sin(2\pi(2k-1)ft)}{2k-1}$$
$$= \frac{4}{\pi}\left(\sin(\omega t) + \frac{1}{3}\sin(3\omega t) + \frac{1}{5}\sin(5\omega t) + \ldots\right), \qquad \text{where } \omega = 2\pi f.$$

- https://www.desmos.com/calculator/baqmhg0orv







Discrete Fourier Transform (DFT)

When a signal is discrete and periodic, we don't need the continuous Fourier transform. Instead we use the discrete Fourier transform, or DFT. Suppose our signal is a_n for $n = 0 \ldots N-1$, and $a_n = a_{n+jN}$ for all n and j. The discrete Fourier transform of a, also known as the spectrum of a, is:

$$A_k = \sum_{n=0}^{N-1} e^{-i\frac{2\pi}{N}kn} a_n$$

In audio signals,
A_k is the amplitude of sound of different pitches (frequencies).
a_n is the amplitude of harmonics of that pitch.

This is more commonly written:

$$A_k = \sum_{n=0}^{N-1} W_N^{kn} a_n \qquad (1)$$

where

$$W_N = e^{-i\frac{2\pi}{N}}$$

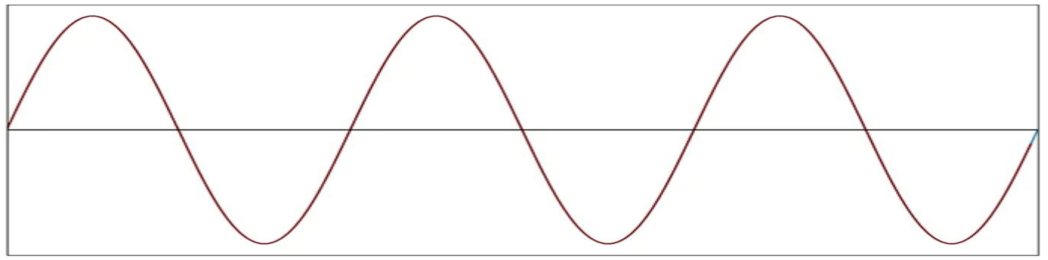
and W_N^k for $k = 0 \ldots N-1$ are called the *Nth roots of unity*.

There are several Arduino libraries available to use the Arduino processors to calculate the frequency spectrum (A_k) using a Fast Fourier Transform (FFT) algorithm, e.g., *ArduinoFFT, Adafruit Zero FFT,* etc…

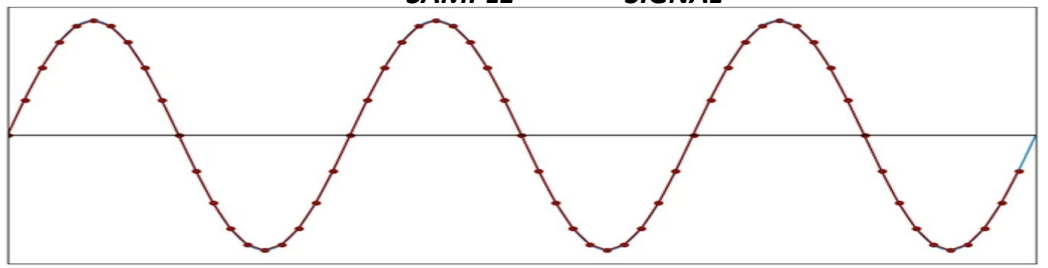
Nyquist-Shannon Theorem

If we have a signal of bandwidth f_0 , in order to reconstruct it perfectly we need to sample at $f_s >= 2f_0$.

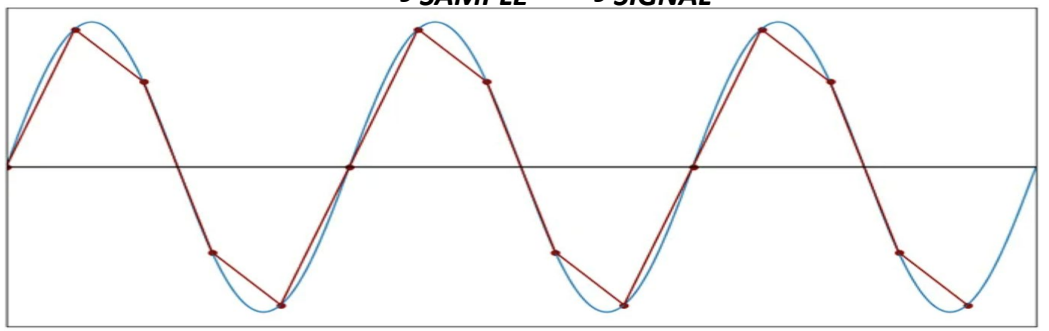
This minimum sampling frequency, f_s, is called the Nyquist frequency.



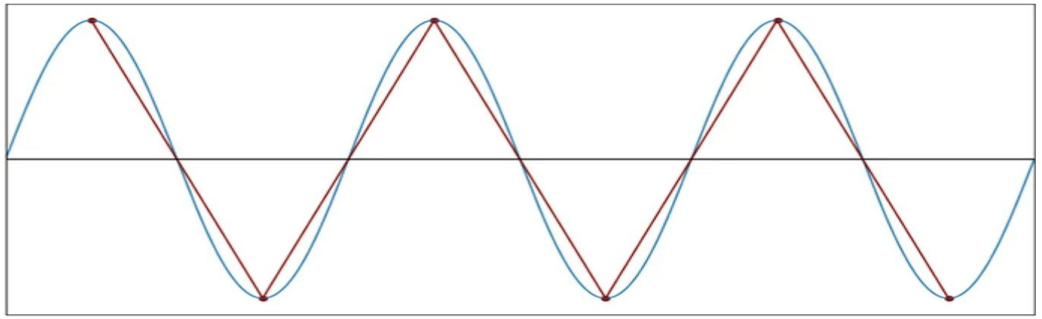
$f_{SAMPLE} = 10f_{SIGNAL}$



$f_{SAMPLE} = 5f_{SIGNAL}$

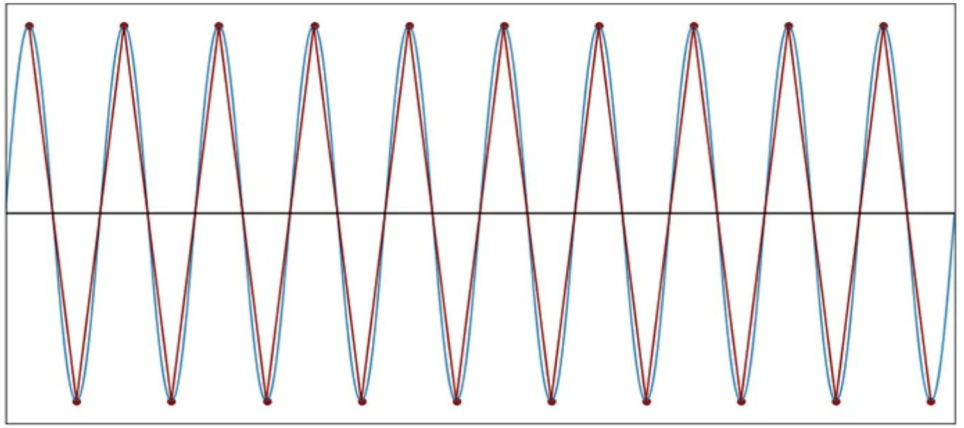


$f_{SAMPLE} = 2f_{SIGNAL}$

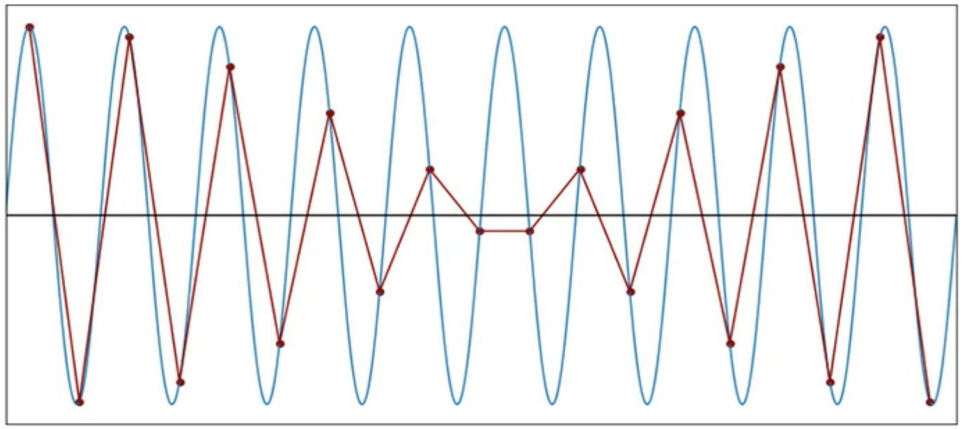


Aliasing

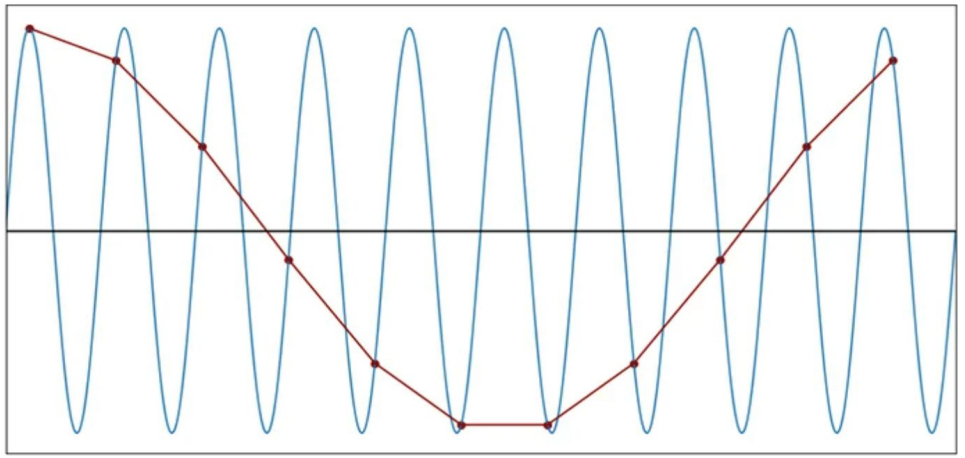
- Undersampling (sampling at lower than the Nyquist rate) causes a phenomenon known as aliasing.
- Aliasing causes a higher-frequency signal to appear at lower-frequency. This can cause noise, distortion, etc. and leads to imperfect reconstruction



$f_{SAMPLE} = 2f_{SIGNAL}$



$f_{SAMPLE} = 1.9f_{SIGNAL}$



$f_{SAMPLE} = 1.1f_{SIGNAL}$

Frequency resolution of the FFT

- The frequency resolution $\Delta f = (f_{sample}/2)/N_{sample}$
- You can increase the frequency resolution, by taking more data points, i.e., increasing $N_{sample.}$
- Note: to use FFT library, the N_{sample} has to be an integer number: 2^N = 64, 128, 256, 512, 1024, 2048, ….

Exercises (Thursday): making a spectrogram

1. Install the “Adafruit Zero FFT” library, modify the ‘fft_test.ino’ example to analyze a digital output signal using the tone generation function: tone(). Connect the digital output pin to A0 and collect data with AnalogRead(A0). Display the FFT power spectrum on the serial monitor and serial plot.
2. Now send the digital tone output, through the amplifier PAM8203, to your mini metal speaker. The speaker should be making sounds (you have done this step on Tuesday)
3. Modify your Arduino sketch to record the voltage output from the electret speaker. Calculate the FFT spectrum. How does it compare with the spectrum you got from step 1?
4. Record a sound wave of your voice using the microphone, and display its FFT spectrum.
   - Pick the correct sampling frequency (at least twice the frequency of interest, according to the Nyquist theorem) and an appropriate number of samples (always a power of 2). You should ideally sample long enough to get down to 10-20 Hz.

Homework: connect a TFT display (ST7789 240x135) to your Adalogger; display the FFT spectrum of sounds recorded with your electret mic on the TFT display as a x-y plot. Check out the example ‘mic_tft.ino’ in the ‘Adafruit Zero FFT library’ for a similar implementation.

Follow this tutorial to learn how to draw on TFT displays: Overview | Adafruit GFX Graphics Library | Adafruit Learning System

(Challenge) create a spectrogram, like the waterfall plot, on your TFT display. You can accumulate enough time slices to fill the whole screen and then flash it all at once, or for an extra challenge make a scrolling display