

ECE 341 ACCELERATED PROJECT

Project Artifacts

Authors: Shelby Westerberg, Trevor Horine, Bryson Goto

December 3, 2020

Instructor: Matthew Shuman

Grading TA: Shane Witsell

Contents

1	Simulation	3
2	Schematic	5
3	Block Diagram	6
4	PCB Layout	7
5	Built Circuit	8
6	Bill of Materials	9
7	Time Report	10
7.1	Hours Spent per Week	10
7.2	What We Worked on by Percentage	10
7.3	Hours Spent per Person	11
7.4	Time sheet	12
8	Code	13
8.1	Arduino Sample Collection Code	13
8.2	Matlab FFT and Graphing Single Set of Data	15
8.3	Matlab Repeated Calculations and LED Communication	18

1 Simulation

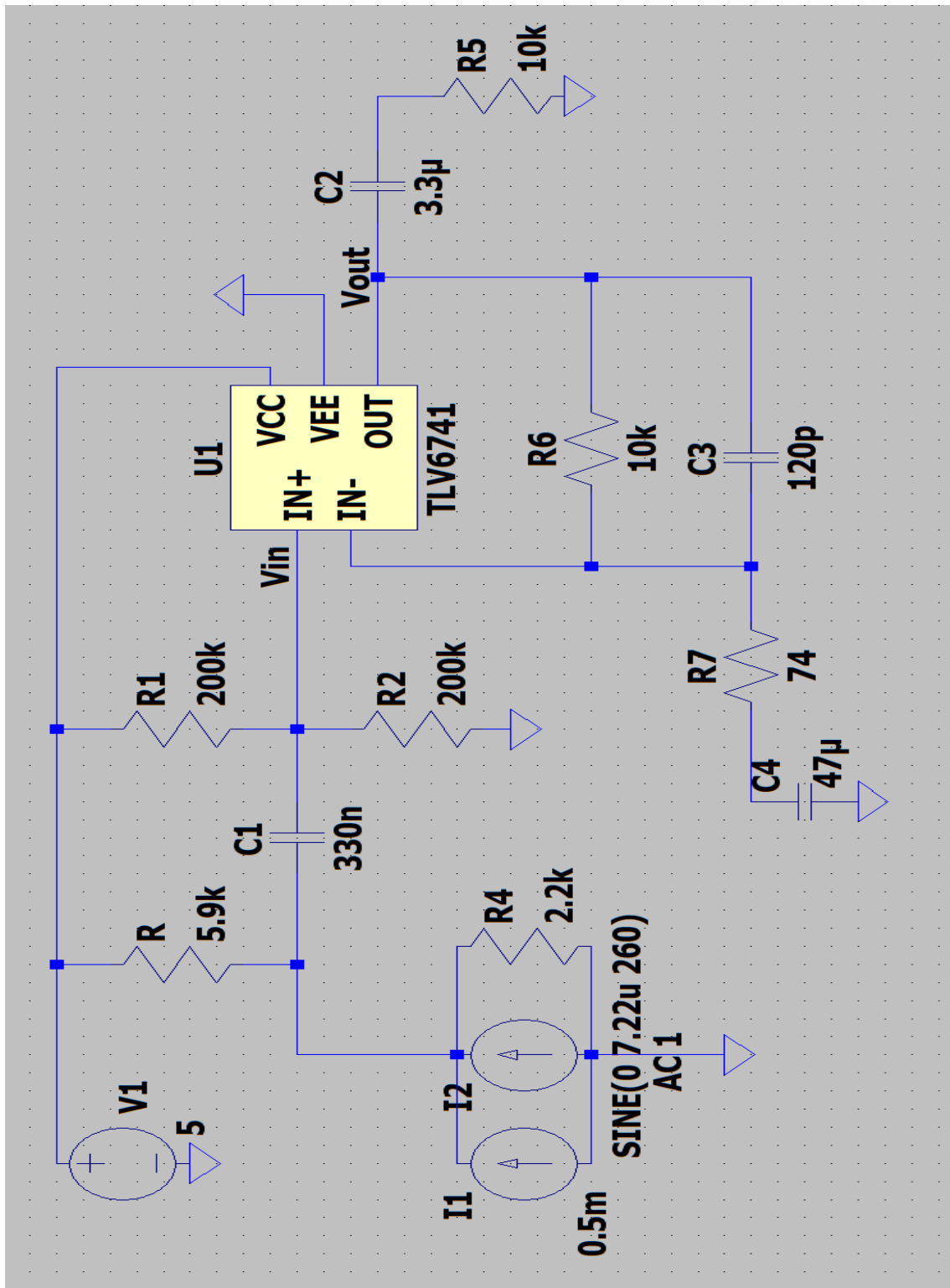


Figure 1: LTSpice Circuit Simulation

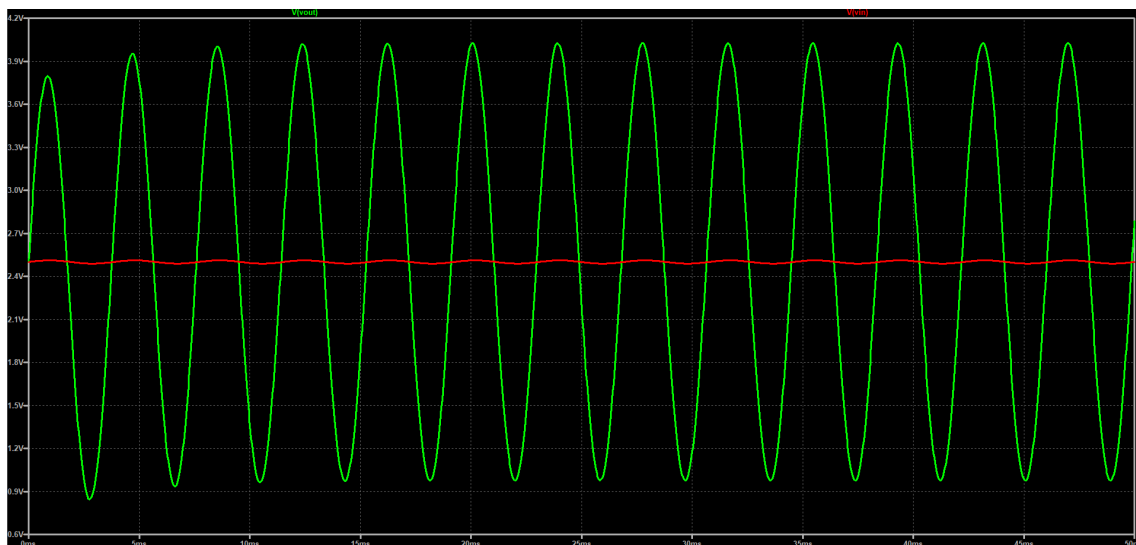


Figure 2: LTSpice Circuit Simulation Voltage Graph

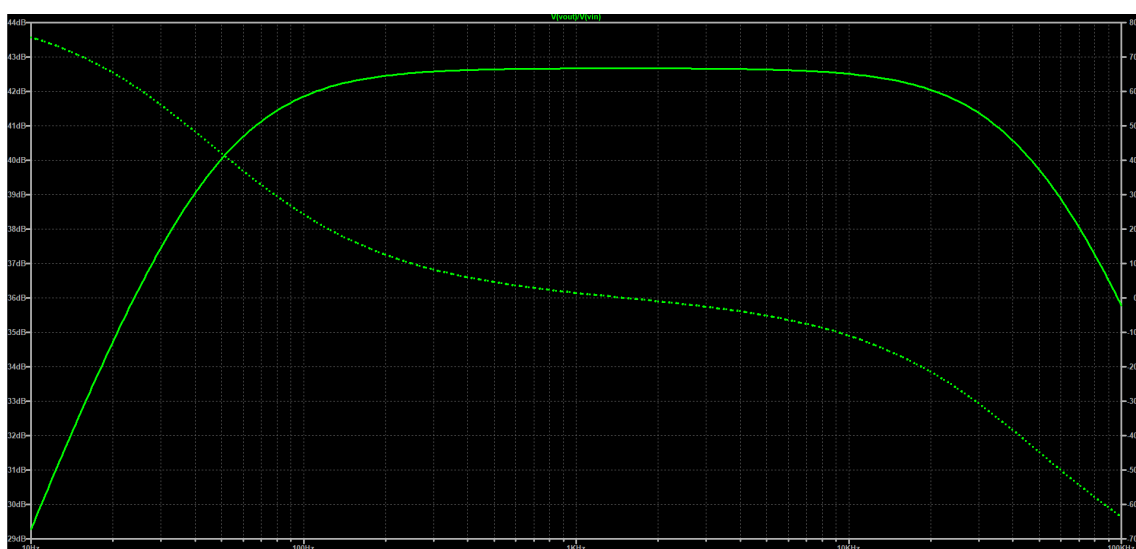


Figure 3: LTSpice Circuit Simulation Gain Graph

The diagram illustrates a 262Hz LED driver circuit. It features an Arduino Nano 3.0 microcontroller connected to an LMC6032A operational amplifier. The circuit is powered by a 5.9k resistor (R1) and a 330nF capacitor (C1). The op-amp's non-inverting input is connected to a voltage divider network consisting of resistors R2, R3, R4, and R5, and capacitors C2, C3, C4, C5, and C6. The op-amp's output, labeled VOUTBIAS@0, drives 18 LEDs (LED1-LED18) through resistors R11-R18. The LEDs are connected to a USB port, which is also connected to the Arduino Nano 3.0. The circuit is designed to operate at 262Hz.

5

3 Block Diagram

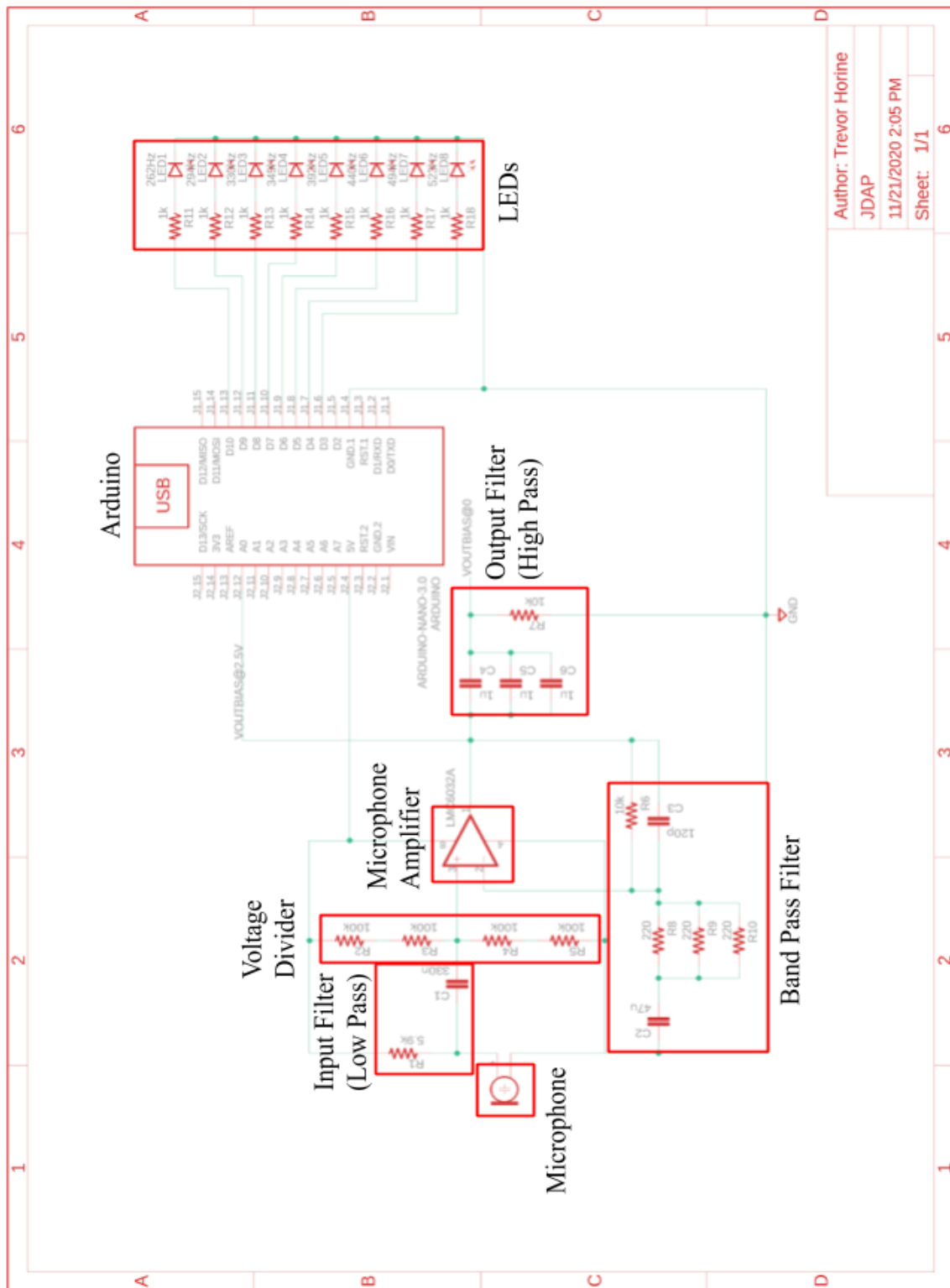


Figure 5: Block Diagram of Audio Detection Circuit

4 PCB Layout

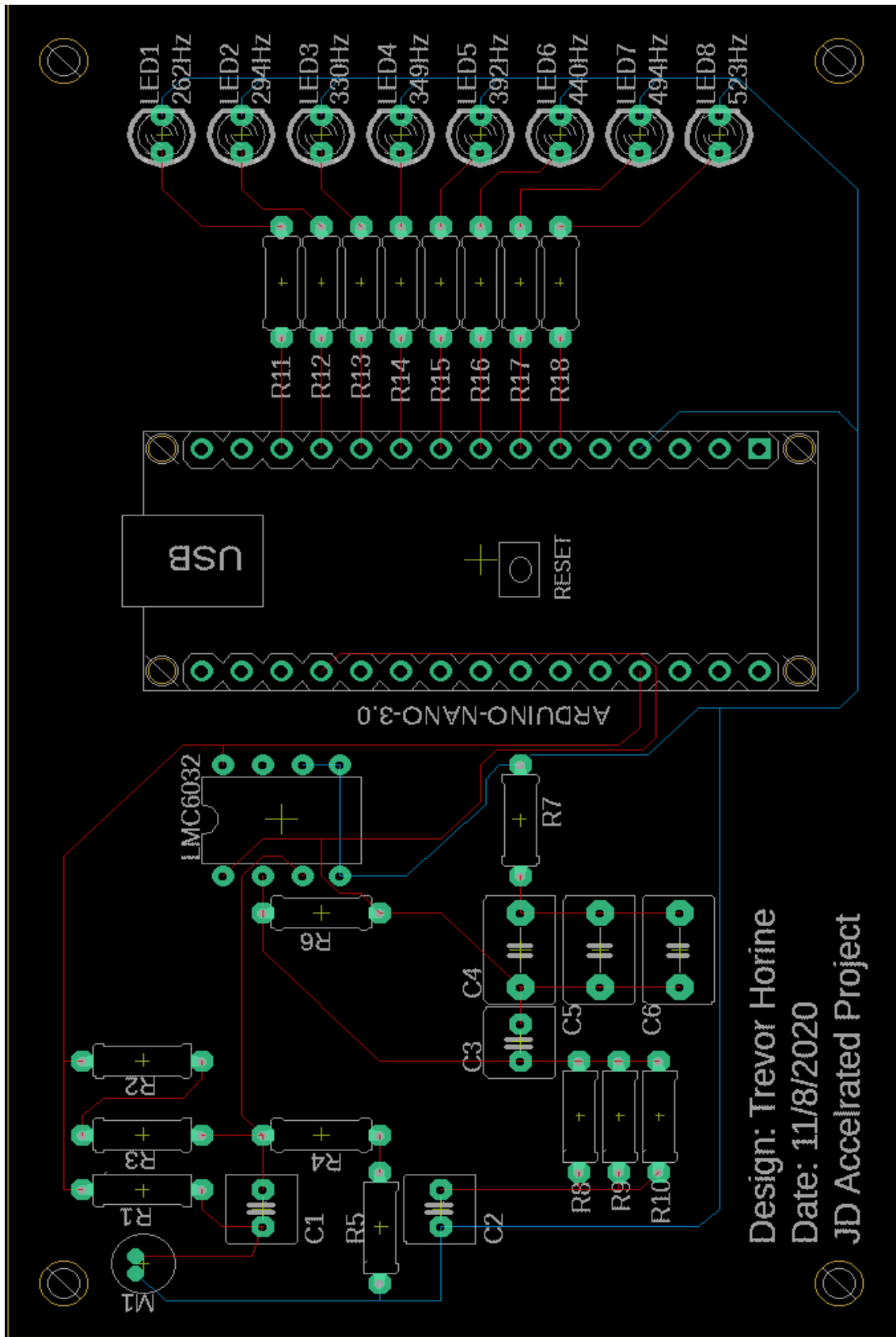


Figure 6: Possible layout of a PCB we designed but did not get printed.

5 Built Circuit

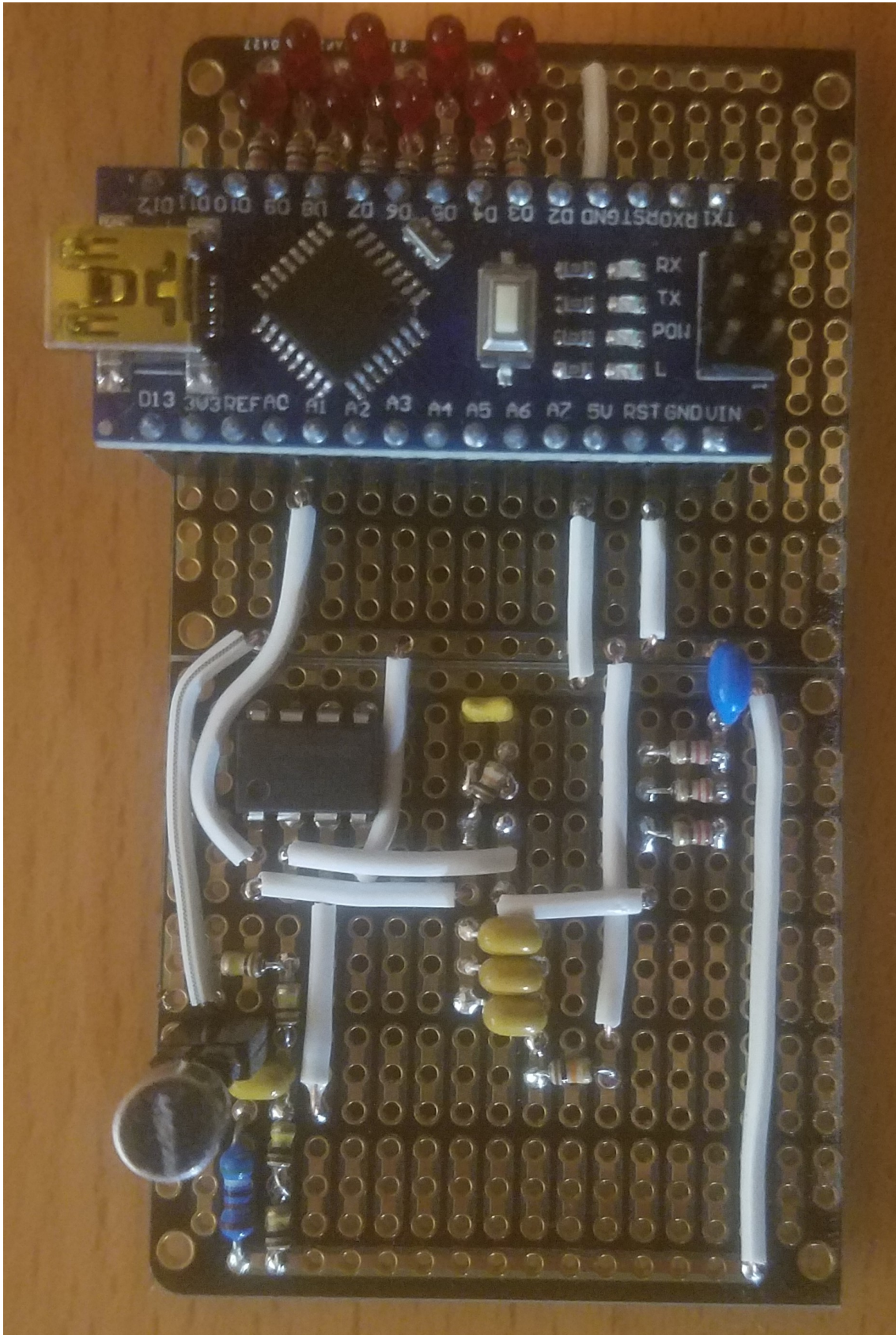


Figure 7: Picture of our project built.

6 Bill of Materials

Part	Value	Package	Description
ARDUINO	ARDUINO-NANO-3.0	ARDUINO-NANO-3.0	Arduino Nano 3.0
C1	330n	Through Hole	CAPACITOR
C2	47u	Through Hole	CAPACITOR
C3	120p	Through Hole	CAPACITOR
C4	1u	Through Hole	CAPACITOR
C5	1u	Through Hole	CAPACITOR
C6	1u	Through Hole	CAPACITOR
LED1		LED3MM	LED
LED2		LED3MM	LED
LED3		LED3MM	LED
LED4		LED3MM	LED
LED5		LED3MM	LED
LED6		LED3MM	LED
LED7		LED3MM	LED
LED8		LED3MM	LED
LMC6032		DIL08	OP AMP
M1	ELECTRET_MICROPHONE	CMC-5042PF-AC	Electret Condenser Microphone
R1	5.9k	Through Hole	RESISTOR
R2	100k	Through Hole	RESISTOR
R3	100k	Through Hole	RESISTOR
R4	100k	Through Hole	RESISTOR
R5	100k	Through Hole	RESISTOR
R6	10k	Through Hole	RESISTOR
R7	10k	Through Hole	RESISTOR
R8	220	Through Hole	RESISTOR
R9	220	Through Hole	RESISTOR
R10	220	Through Hole	RESISTOR
R11	1k	Through Hole	RESISTOR
R12	1k	Through Hole	RESISTOR
R13	1k	Through Hole	RESISTOR
R14	1k	Through Hole	RESISTOR
R15	1k	Through Hole	RESISTOR
R16	1k	Through Hole	RESISTOR
R17	1k	Through Hole	RESISTOR
R18	1k	Through Hole	RESISTOR

7 Time Report

This time report is based on when the report was finished and does not include the hours put in to record the project video submission.

7.1 Hours Spent per Week

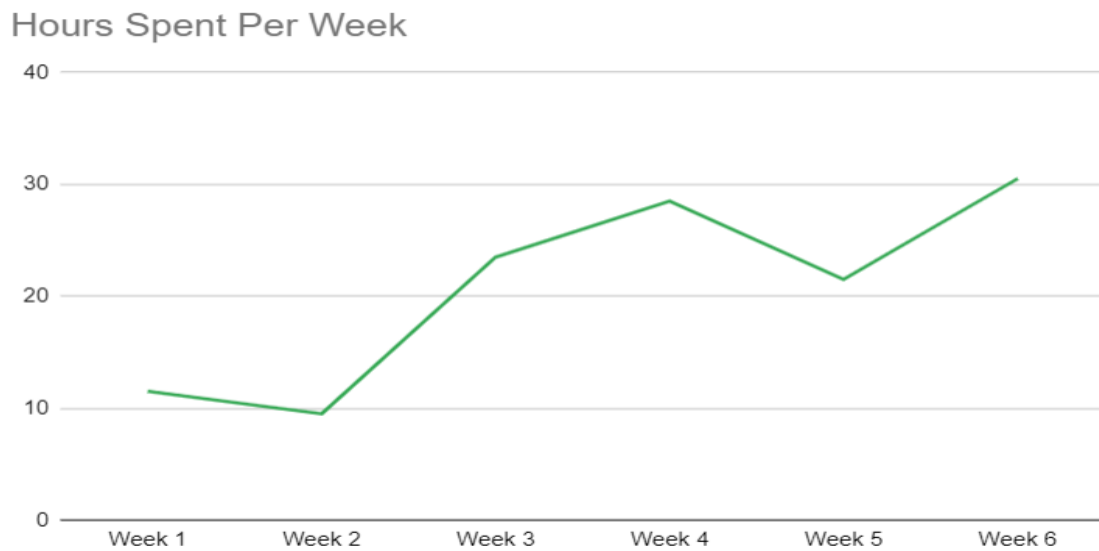


Figure 8: Hours spent by the team each week on the project.

7.2 What We Worked on by Percentage

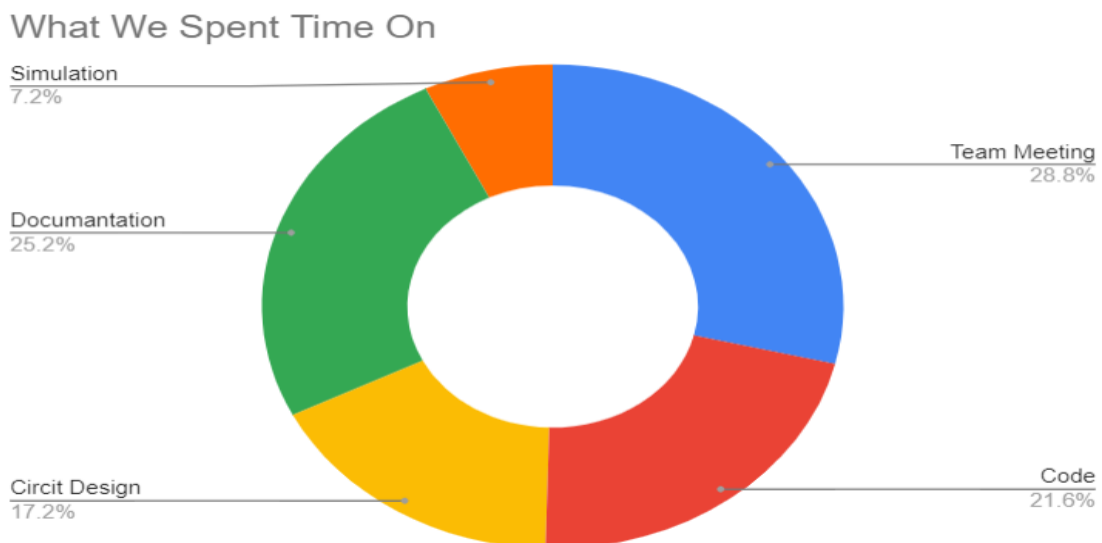


Figure 9: What the team worked on by percentage.

7.3 Hours Spent per Person

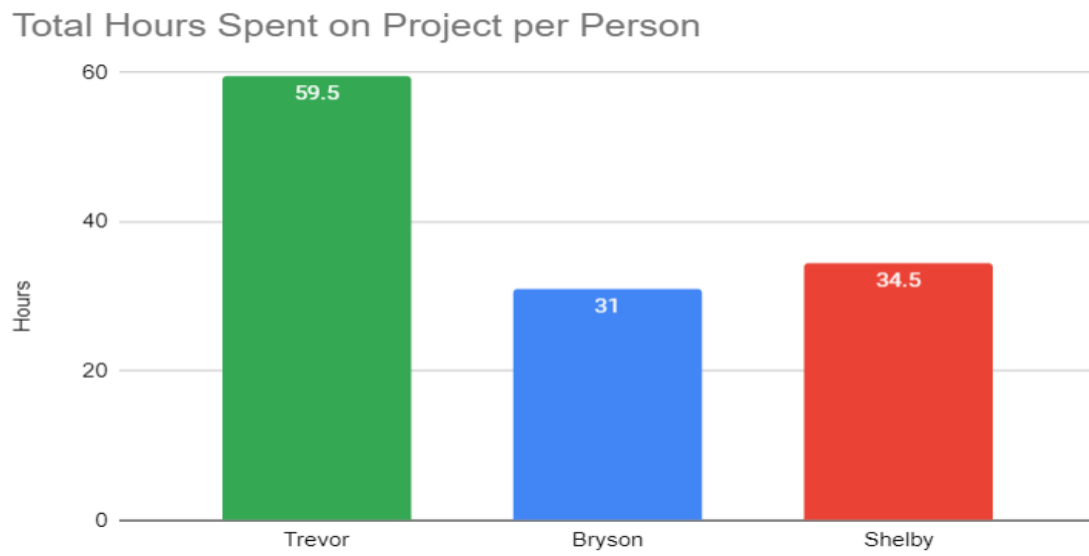


Figure 10: Breakdown of how many hours each team member put in to the project.

7.4 Time sheet

Name	Week	Time spent (Hours rounded to nearest .5)	What did you work on
Trevor	The Beginning of Time	0.5	Set up time sheet
All	10/22/2020	3	Team meeting, Project planning
Trevor	10/22-28	6	Circuit calculations, circuit building, LTSpice, Matlab, Arduino
Shelby	10/22-28	1	Review datasheet, circuit calculations, LTSpice
Bryson	10/22-28	1	Circuit calculations
All	10/29/2020	3	Team meeting, Simulation, circuit calculations
Trevor	11/3	2.5	simulation, circuit calculations, data gathering
Shelby	11/3/2020	1.5	Report introduction and frequency calculations
Bryson	11/3/2020	2.5	Calculations, circuit simulations
Trevor	11/5	6	Questions in lab, debugging/matlab/arduino code
All	11/5/2020	3	Group meeting: simulation and debugging
Trevor	11/6	4	debug and testing, some code "magic"
Bryson	11/8	2	Assembling circuit
Trevor	11/8	2	code commenting and schematics
Bryson	11/9	1.5	Documentation, LTSpice simulations + screenshots
Shelby	11/9	1	Documentation
Trevor	11/11	4	codeing (MATLAB and Arduino) (regerster programming)
Bryson	11/12	3	Group meeting, troubleshooting frequency response
Trevor	11/12	3	Group meeting, troubleshooting frequency response
Shelby	11/12	0.5	Group meeting
Shelby	11/13	2	Circuit assembly, Matlab code
Shelby	11/14	1	Matlab code
All	11/14	12	Group meeting, troubleshooting
Trevor	11/14	0.5	Time sheet formulas
Trevor	11/16	0.5	Troubleshooting, office hour prep
All	11/17	6	Office hours, troubleshooting
Trevor	11/19	4	Coding and troubleshooting
Bryson	11/19	2	Circuit building, video script writing
Shelby	11/19	2	Coding and troubleshooting
Shelby	11/20	0.5	Shielding design
Trevor	11/20	1	Circuit soldering
Shelby	11/21	1	Circuit troubleshooting
All	11/21	7.5	Complete circuit build, documentation
Trevor	11/21	2.5	Circuit Soldering, and testing
Shelby	11/24	1	Report writing
Shelby	11/26	2	Report writing
Shelby	11/29	1.5	Report writing, sampling frequency measurement, SNR setup
Trevor	11/29	3	Demo video
Shelby	11/30	1	Report writing
Trevor	11/30	1.5	Artifacts document
Bryson	12/1	0.5	Report + Artifact
All	12/1	9	Presentation prep
Trevor	12/2	0.5	Graphs and documentation
All	12/3	12	Documentation and report writing
		Total Manhours	
	Percentage per Person	125	
Trevor	47.6	59.5	
Bryson	24.8	31	
Shelby	27.6	34.5	

8 Code

8.1 Arduino Sample Collection Code

```
1  /*
2  This program waits for an input in the serial monotor then prints
   the next 10500 values captred by the arduino to the serial port.
   It also passes the time it took to collect all the samples the
   time it took to gather thoes samples to the serial port.
3  Trevor Horine
4  11/6/2020
5  */
6
7  //Pin assinments and global variables
8  const int analogInPin = A0;  // Analog input pin that is used as
   input form amplifier
9  int sensorValue = 0;          // value read from the output of
   amplifier
10 const int MAX_SAMPLES = 10500; //number of samples collected before
   sending
11 int last;
12
13 void setup() {
14   Serial.begin(230400); // initialize serial communications at
   230400 baud rate
15   pinMode(LED_BUILTIN, OUTPUT); //built in led
16   digitalWrite(LED_BUILTIN, HIGH);
17 }
18
19 void loop() {
20   int values = MAX_SAMPLES+100;
21   digitalWrite(LED_BUILTIN, HIGH);
22   while(!(Serial.available() > 0)){ //while nothing in serial port
   (waiting for matlab to reach out)
23     Serial.read();
24     values = 0;
25     last = true;
26   }
27   digitalWrite(LED_BUILTIN, LOW); //when found something in serial
   port (matlab has reached out)
28   delay(10);
29   long before = micros(); //current time in microseconds before
   collecting data
30   while(values < MAX_SAMPLES) { //while less then desired number of
   samples has been printed to the serial port
   // read the analog in value form amplifier
31     sensorValue = analogRead(analogInPin);
32     byte buf[2]; //split in to bytes
33     buf[0] = sensorValue & 255;
34     buf[1] = (sensorValue >> 8) & 255;
35     Serial.write(buf, sizeof(buf)); // write to serial port
36     values++;
37   }
38   if (last) {
39     long diff = micros()-before; //calculate and write time it took
   to get samples to serial port
40     byte buf[4];
41     buf[0] = diff & 255;
```

```
43     buf[1] = (diff >> 8) & 255;
44     buf[2] = (diff >> 16) & 255;
45     buf[3] = (diff >> 24) & 255;
46     Serial.write(buf, sizeof(buf));
47     last = false;
48 }
49 }
```

8.2 Matlab FFT and Graphing Single Set of Data

```
1 %This program will open a serial port, then send a signal to an
   Arduino. It
2 %will then read in a set number of samples and how long it took to
   collect
3 %them. The arduino ADC values will then be converted to voltages.
   The
4 %captured signal will be graphed and run through the MATLAB FFT to
5 %determine the frequency of the signal. The Single-Sided Amplitude
   Spectrum
6 %is graphed and the highest frequency in the range between 250 and
   540 is
7 %reported to the command window.
8 %Trevor Horine
9 %11/6/2020
10
11 %start with a clean workspace
12 clear
13
14 %Number of samples
15 numSamples = 10500;
16
17
18 %open serial port set this to what port the Arduino is conected to
19 device = serialport("COM6",500000);
20 configureTerminator(device,"CR/LF")
21 %create empty vector to add values of the signal to
22 y = 0:0:0;
23 %read in line specified number (50) times from serial port
24 pause(5);
25 write(device,"getval","string");
26 %read in samples from Arduino
27 tic;
28 num = read(device,numSamples,"uint8")
29 toc
30 %read in time to collect samples from Arduino
31 timeIn = read(device,1,"uint32");
32 %convert time to seconds from micro seconds
33 captureSeconds = timeIn/1000000;
34 %print time to capture samples
35 fprintf("Capture Time: %f\n",captureSeconds);
36 %time per sample in seconds
37 sampleTime = captureSeconds/numSamples;
38 %create time vector to graph signal against
39 %x is the vector of the x-axis and has increments of time that
40 x = 0:sampleTime:captureSeconds-sampleTime;
41 %for loop to covert from ADC value and place it in the vector
42 for i = 0:numSamples-1
43     %num is the string that is read in from the serial port
44     curnum = num(i+1);
45     %add it to the end of the y vector
46     y(end +1) = (curnum*5)/256;
47 end
48 %release the serial port so it can be used by something else
49 %clear device
50 %create figure with a 1x3 array of graphs
51 figure
```

```

52 %plot the graph using the x and y vectors
53 %plot the whole signal
54 subplot(3,1,1)
55 plot(x,y)
56 %title graph and axes
57 title('Mic voltage')
58 xlabel('Time (seconds)')
59 ylabel('Voltage (Volts)')
60 %range of axis
61 xlim([0 1.05])
62 ylim([0 5])
63
64 %plot small section of signal so can see the period (might not line
    up with
65 %the window perfectly, if not pan up or down)
66 subplot(3,1,2)
67 plot(x,y)
68 %title graph and axes
69 title('Mic voltage')
70 xlabel('Time (seconds)')
71 ylabel('Voltage (Volts)')
72 %range of axis
73 xlim([.1 .2])
74 ylim([2 4.5])
75
76 %MATLAB FFT
77 Fs = numSamples/captureSeconds; % Sampling frequency
78 T = 1/Fs; % Sampling period
79 L = numSamples; % Length of signal (in milliseconds)
80 t = (0:L-1)*T; % Time vector
81 Y = fft(y);
82 P2 = abs(Y/L);
83 P1 = P2(1:floor(L/2+1));
84 P1(2:end-1) = 2*P1(2:end-1);
85 %0.74074074074074 is weird factor that we found in testing, dont
    know where
86 %it came from but the FFT is off by this factor every time.
87 f = Fs*(0:(L/2))/L;
88 %f = (1/captureSeconds)*Fs*(0:(L/2))/L;
89
90 %SNR computation:
91 r = snr(y,Fs,3); %outputs snr in decibels, ignoring the first 3
    harmonics
92 fprintf("Signal to Noise Ratio %f\n", r)
93
94
95 %plot the Single-Sided Amplitude Spectrum (intensity at difrent
    frequancies)
96 subplot(3,1,3);
97 plot(f,P1)
98 title('Single-Sided Amplitude Spectrum of X(t)')
99 xlabel('f (Hz)')
100 ylabel('|P1(f)|')
101 xlim([200 600])
102 ylim([0 .075])
103
104 %find maximum value in the intensity vector from the FFT and it's
    index

```



```

105 %ignore anything that is not between 250ish to 530ish
106 b = P1(53:120);
107 [maximum, maxindex] = max(b);
108 %if maximum is really small, liklely no signal
109 feq = round(f(maxindex+52));
110 if maximum < .001
111     fprintf("No Signal\n")
112     write(device, "N", "string");
113     %print the frequency of signal
114 else
115     fprintf("Highest frequency is %i\n", feq)
116     if feq >= 260 && feq <= 263
117         write(device, "MC","string");
118     end
119     if feq >= 292 && feq <= 295
120         write(device, "D","string");
121     end
122     if feq >= 327 && feq <= 331
123         write(device, "E","string");
124     end
125     if feq >= 347 && feq <= 351
126         write(device, "F","string");
127     end
128     if feq >= 390 && feq <= 394
129         write(device, "G","string");
130     end
131     if feq >= 437 && feq <= 442
132         write(device, "A","string");
133     end
134     if feq >= 491 && feq <= 496
135         write(device, "B","string");
136     end
137     if feq >= 520 && feq <= 526
138         write(device, "HC","string");
139     end
140 end
141 %clear workspace when done
142 clear device

```

8.3 Matlab Repeated Calculations and LED Communication

```
1 %This program will open a serial port, then send a signal to an
  Arduino. It
2 %will then read in a set number of samples and how long it took to
  collect
3 %them. The arduino ADC values will then be converted to voltages.
  The
4 %captured signal will be graphed and run through the MATLAB FFT to
5 %determine the frequency of the signal. The Single-Sided Amplitude
  Spectrum
6 %is graphed and the highest frequency in the range between 250 and
  540 is
7 %reported to the command window.
8 %Trevor Horine
9 %11/6/2020
10
11 %start with a clean workspace
12 clear
13
14 %Number of samples
15 numSamples = 10500;
16 %open serial port set this to what port the Arduino is conected to
17 device = serialport("COM6",500000);
18 configureTerminator(device,"CR/LF")
19
20 while 1
21     pause(2);
22     %create empty vector to add values of the signal to
23     y = 0:0:0;
24     write(device,"getval","string");
25     %read in samples from Arduino
26     tic;
27     num = read(device,numSamples,"uint8");
28     toc
29     %read in time to collect samples from Arduino
30     timeIn = read(device,1,"uint32");
31     %convert time to seconds from micro seconds
32     captureSeconds = timeIn/1000000;
33     %print time to capture samples
34     fprintf("Capture Time: %f\n",captureSeconds);
35     %time per sample in seconds
36     sampleTime = captureSeconds/numSamples;
37     %create time vector to graph signal against
38     %x is the vector of the x-axis and has increments of time that
39     x = 0:sampleTime:captureSeconds-sampleTime;
40     %for loop to covert from ADC value and place it in the vector
41     for i = 0:numSamples-1
42         %num is the string that is read in from the serial port
43         curnum = num(i+1);
44         %add it to the end of the y vector
45         y(end +1) = (curnum*5)/256;
46     end
47     %release the serial port so it can be used by something else
48     %clear device
49     %create figure with a 1x3 array of graphs
50     % figure
```

```

51 % %plot the graph using the x and y vectors
52 % %plot the whole signal
53 % subplot(3,1,1)
54 % plot(x,y)
55 % %title graph and axes
56 % title('Mic voltage')
57 % xlabel('Time (seconds)')
58 % ylabel('Voltage (Volts)')
59 % %range of axis
60 % xlim([0 1.05])
61 % ylim([0 5])
62 %
63 % %plot small section of signal so can see the period (might
not line up with
64 % %the window perfectly, if not pan up or down)
65 % subplot(3,1,2)
66 % plot(x,y)
67 % %title graph and axes
68 % title('Mic voltage')
69 % xlabel('Time (seconds)')
70 % ylabel('Voltage (Volts)')
71 % %range of axis
72 % xlim([.1 .2])
73 % ylim([2 4.5])
74
75 %MATLAB FFT
76 Fs = numSamples/captureSeconds; % Sampling frequency
77 T = 1/Fs; % Sampling period
78 L = numSamples; % Length of signal (in milliseconds
)
79 t = (0:L-1)*T; % Time vector
80 Y = fft(y);
81 P2 = abs(Y/L);
82 P1 = P2(1:floor(L/2+1));
83 P1(2:end-1) = 2*P1(2:end-1);
84 %0.74074074074074 is weird factor that we found in testing,
dont know where
85 %it came from but the FFT is off by this factor every time.
86 f = Fs*(0:(L/2))/L;
87 %f = (1/captureSeconds)*Fs*(0:(L/2))/L;
88
89 %plot the Single-Sided Amplitude Spectrum (intensity at different
frequencies)
90 % subplot(3,1,3);
91 % plot(f,P1)
92 % title('Single-Sided Amplitude Spectrum of X(t)')
93 % xlabel('f (Hz)')
94 % ylabel('|P1(f)|')
95 % xlim([200 600])
96 % ylim([0 .075])
97
98 %find maximum value in the intensity vector from the FFT and it
's index
99 %ignore anything that is not between 250ish to 530ish
100 b = P1(50:120);
101 [maximum, maxindex] = max(b);
102 feq = round(f(maxindex+49));
103 %if maximum is really small, likely no signal

```

```

104     if maximum < .001
105         fprintf("No Signal\n")
106         write(device, "N", "string");
107     %print the frequency of signal
108     else
109         fprintf("Highest frequency is %i\n", feq)
110         if feq >= 260 && feq <= 263
111             write(device, "MC","string");
112         end
113         if feq >= 292 && feq <= 295
114             write(device, "D","string");
115         end
116         if feq >= 327 && feq <= 331
117             write(device, "E","string");
118         end
119         if feq >= 347 && feq <= 351
120             write(device, "F","string");
121         end
122         if feq >= 390 && feq <= 394
123             write(device, "G","string");
124         end
125         if feq >= 437 && feq <= 442
126             write(device, "A","string");
127         end
128         if feq >= 491 && feq <= 496
129             write(device, "B","string");
130         end
131         if feq >= 520 && feq <= 526
132             write(device, "HC","string");
133         end
134     end
135 end
136 %clear device when done
137 clear device

```