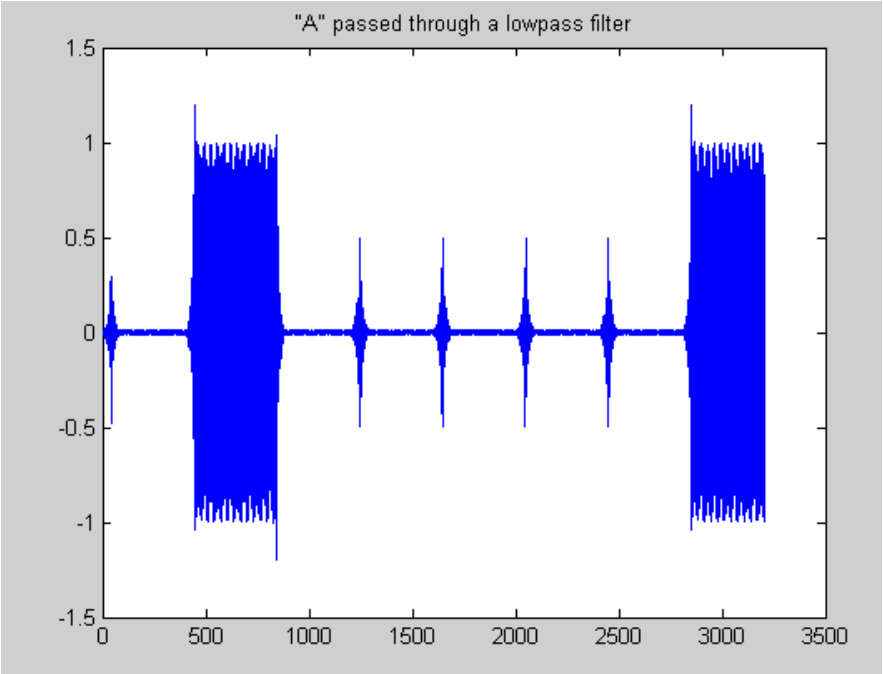
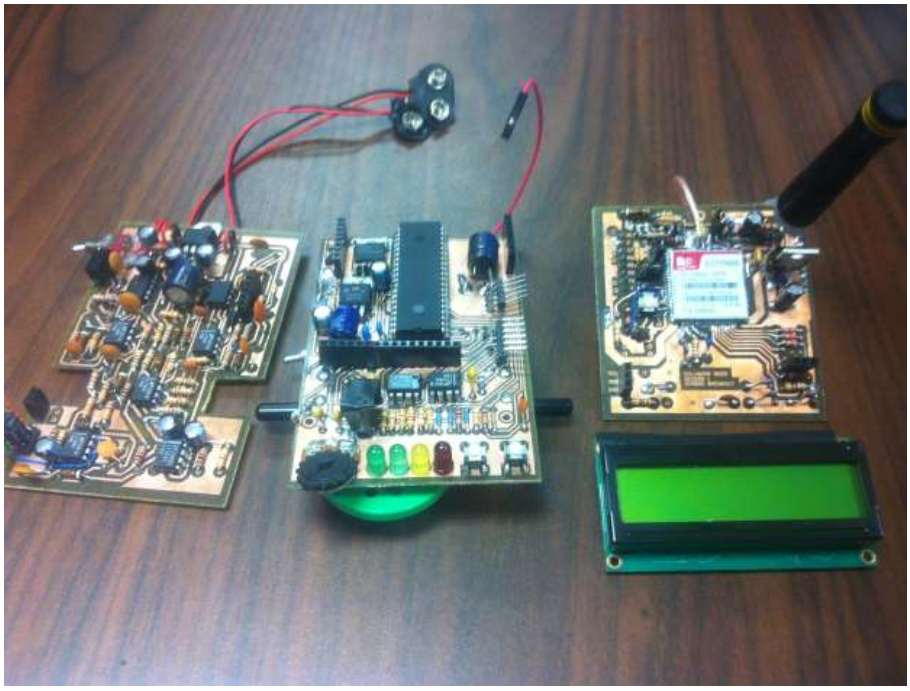
FSK modem over a GSM network

Salvador Razo

Design 1 Final project

Report Due: Friday 24th of May, 2014





Contents

Introduction

The original idea for this project was taken from a lab in Signal and Systems (EEL3135). I wanted to create an FSK modem and transmit the signal through a GSM chip to my device. I would use analog filters because the PIC and none of the microcontrollers available would provide the speed required for DSP in hardware.

Proposal

I proposed using a SIM900 chip as the GSM chip and would run the audio to analog filters. Digital IO would be achieved in the form of UART communication with that device and with status LEDs. For analog in I proposed to use a pot for volume. Analog out would be a sine wave sent to the dac and amplified with the LM386.

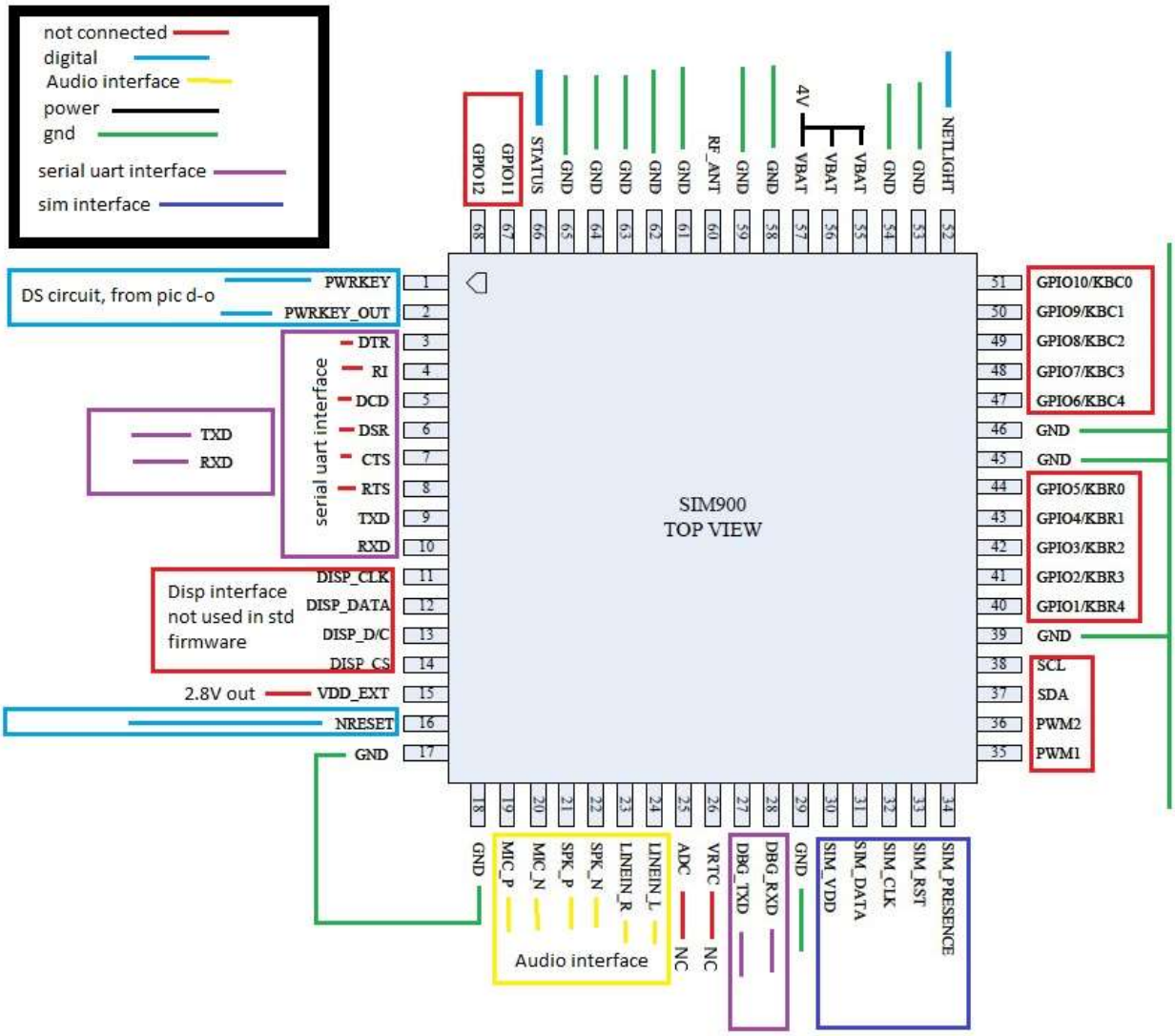
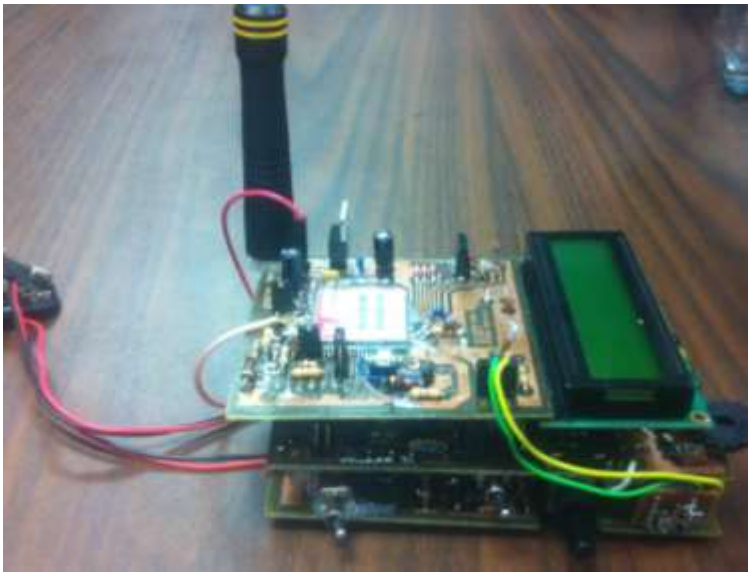


Figure 1: SIM 900 chip

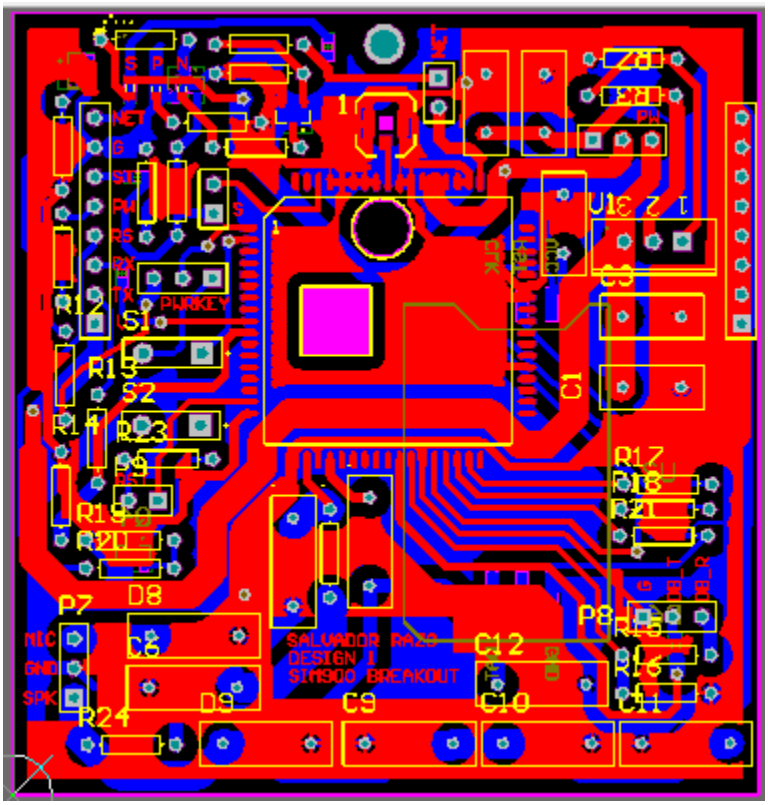
In Figure 1 I presented a plan on how to use the pins of the SIM900. The proposal also outlined the other hardware required and how the digital pins were to be used. That document can be viewed on E-Learning and also outlines suggested improvements on this project.

Project



SIM900 board

First I created a SIM900 breakout board to prototype with it; I was not able to prototype with the chip directly because it was a SMD component.



This board included the breakout to the main digital pins that would be used as well as a small passive filter for the audio. This board included a mount for the U.FL IPX connector an a power supply for the sim 900.

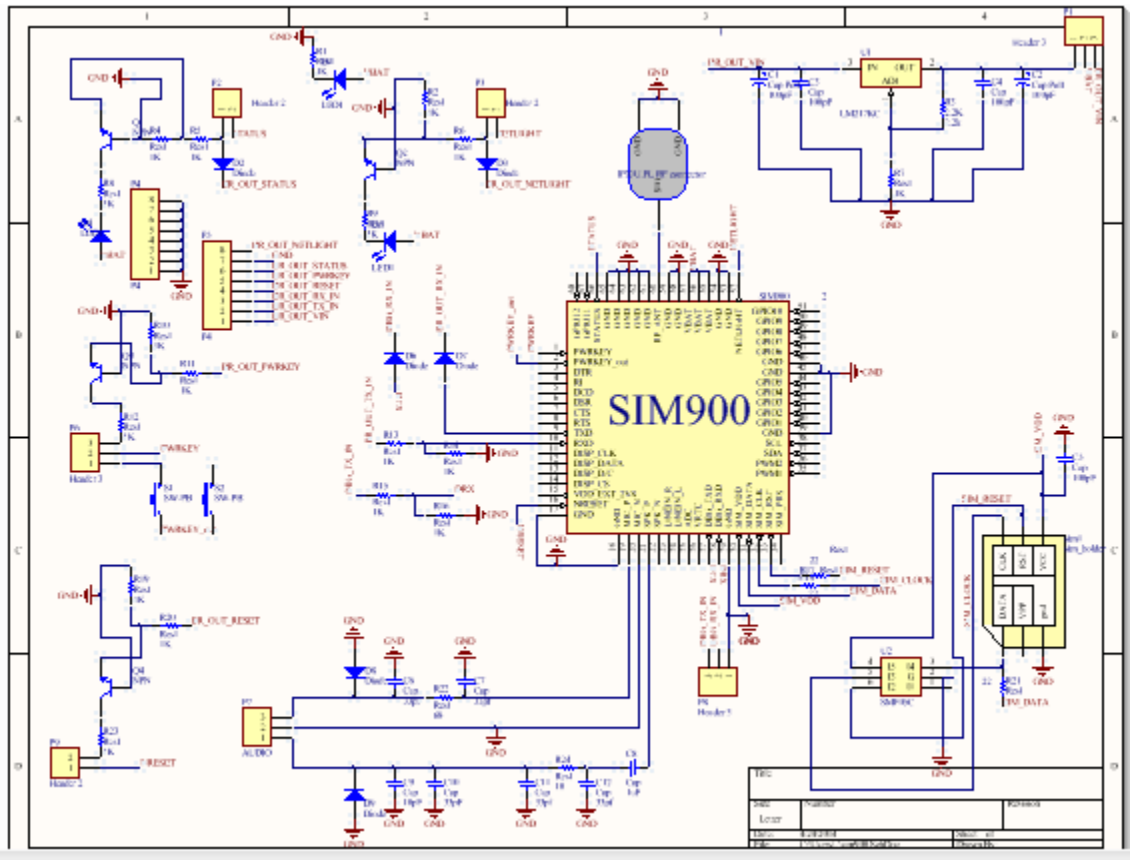
Problems encountered

The sim900 may draw upto 2A when powering up or when initializing a call. I chose a LM350T because the datasheet states that it can handle up to 3A at the required voltage of my device. When that much current is being drawn, however, my adjustment circuit was affected and the power dropped. I instead opted for a MIC29302BT and that solved the issue.

The second regulator was required because this device required 3.5-4.5 volts. Instead of converting the logic of this device to 5V I opted to run the pic on this power supply as well. I still required a 7805 for 5V to drive the lcd and filters.

My SIM card holder did not arrive so I had to order another one. It was a smaller size however and I modified the pcb by scraping away copper and running wires.

I also had trouble soldering the smd components such as the LEDs and the smF05c. These were more frustrations over patience requirements than actual problems.

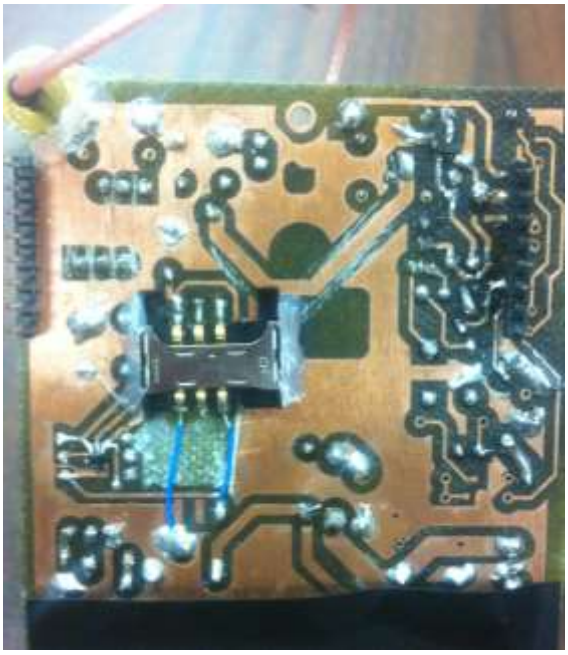


Altium footprints

I had to create a footprint for the SIM900, the IPX connector, the sim card holder, and the SMF05C. In this schematic one can see the power supply and see why increases current on the output pin would affect the ADJ pin.

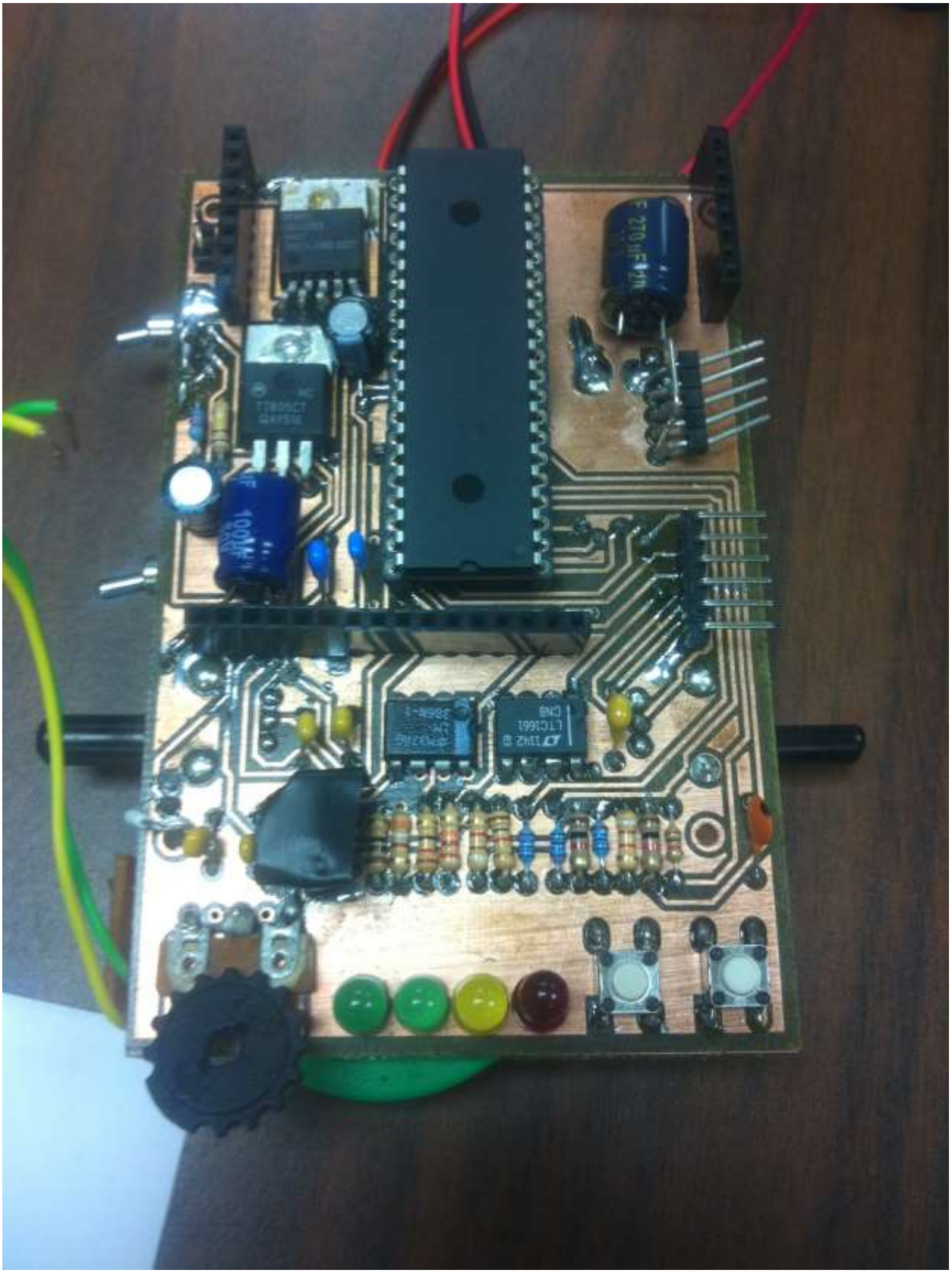
Also note the headers. I used them to isolate certain parts of the circuit and to have probe points readily available. Prototyping with this device was done using an Arduino Due with UART pass-through between two modules to convert logic levels from the computer to the sim900. I used a terminal to send commands to the device and understand its behavior. The whole motivation of this project was to use this chip in Senior design and after reviewing the documentation and gaining experience with it, I feel much more comfortable using it in Senior design and confident it is the right chip to use.

Final sim900 board

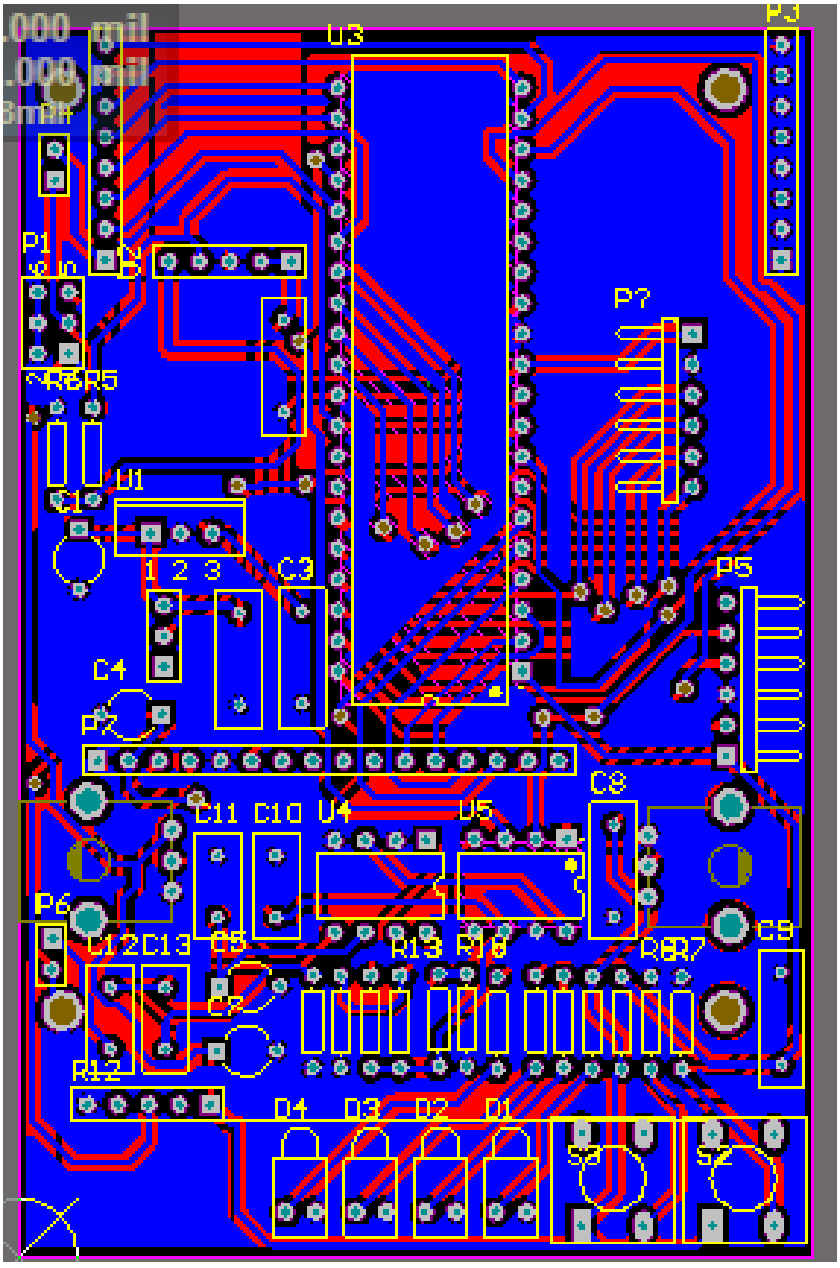


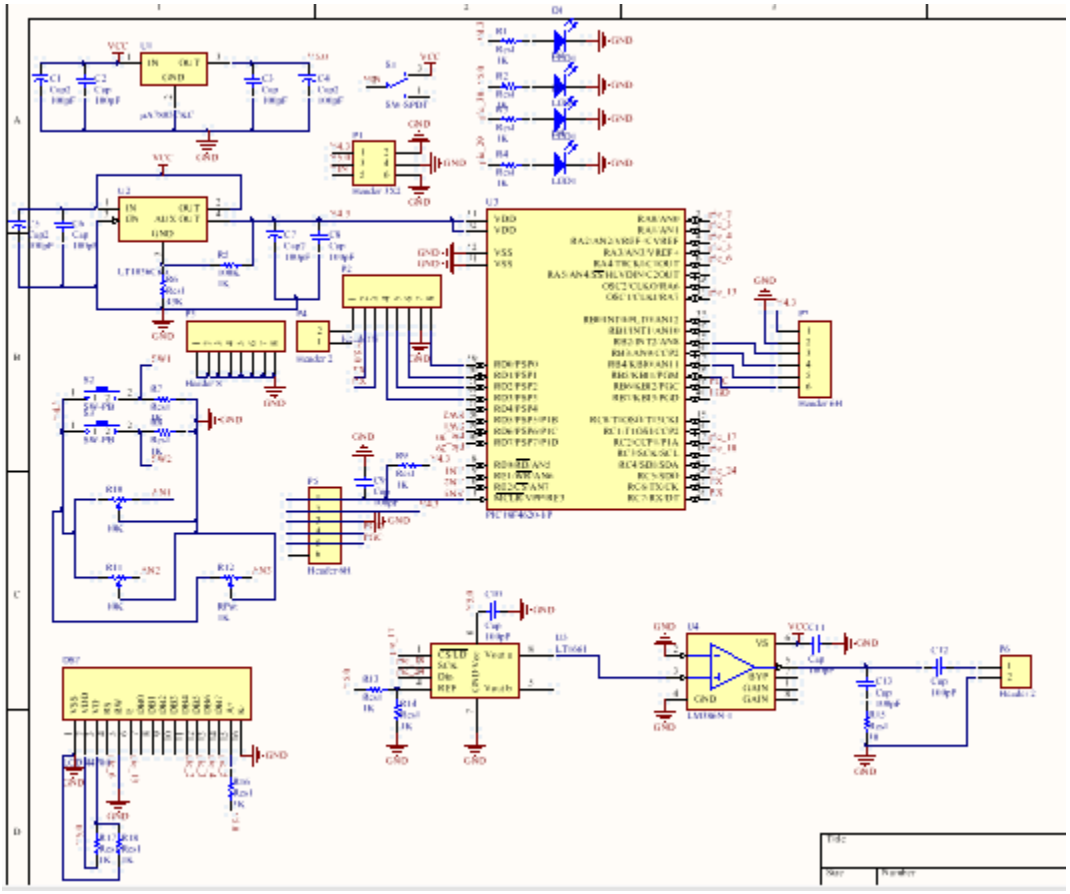
This board was manually routed. Notice that the top layer has some components not soldered on. These were TVS protection diodes that I opted not to use, aswell as logic conversion circuitry that was not required when I ran the PIC at the lower voltage. I did use TVS diodes on the bottom layer however, to the bottom left of the sim socket. This was the hardest component to solder but I was able to short an unused pin to ground to make it easier on me. The blue wire on the top layer is from a trace that came off after overheating a pad. The upper left of the board has the SMD LEDs. The upper right has the power supply header that is connected to with a wire to bypass the regulator after it was discovered that it would not function as desired.

Carrier board



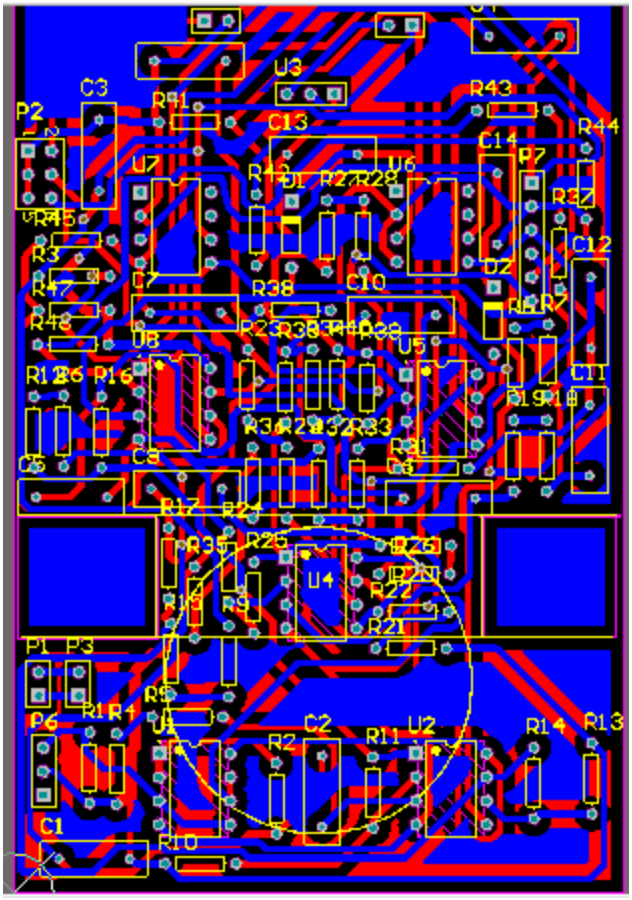
The carrier board includes the pic and all the digital components of the board. The red LED is a 4.3V indicator and the Yellow is a 5V indicator. I added two switches (only one on this board) to turn off each supply. The headers on the right are a programming header and a debug header for certain digital and analog pins. This board was also completely manually routed. I did not have many issues with this board because it included previously tested components from other labs. The electrical tape is covering capacitors because the LCD rested on it. The top regulator is my 4V one and the lower is a 7805. There are two logarithmic pots on the bottom. This was done for space reasons and I cut a hole in the analog board to accommodate.

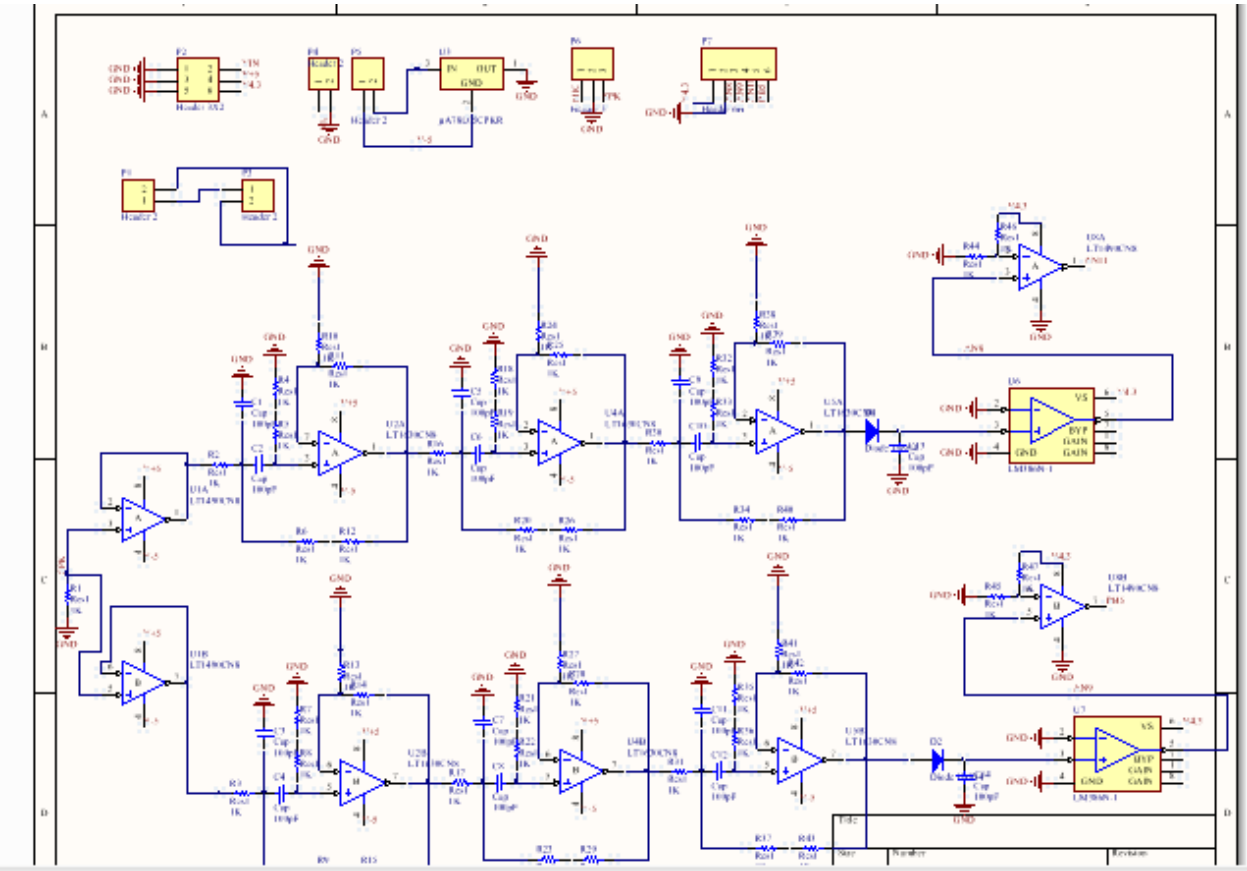




Analog board

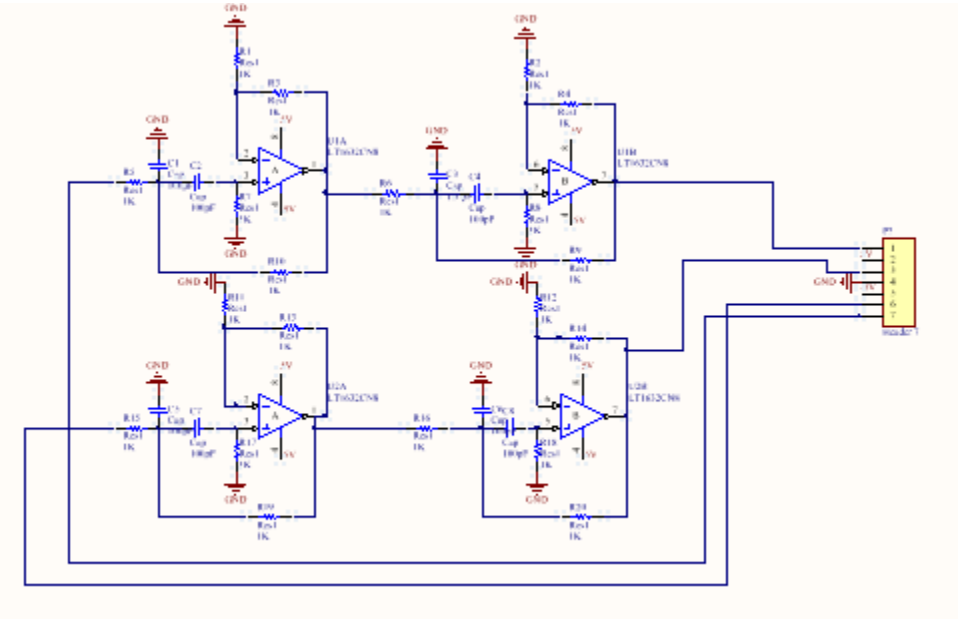
This was by far the most difficult board. Noise on this board was due to not separating it from the digital components above and not having a ground plane. After errors with this board I read up on separate grounding planes, separating return lines and various techniques. This board has two bandpass filters coming from an amplifier. They then are rectified, passed through a lowpass filter, and amplified again. I used the filter software from TI and simulated them with LTspice.

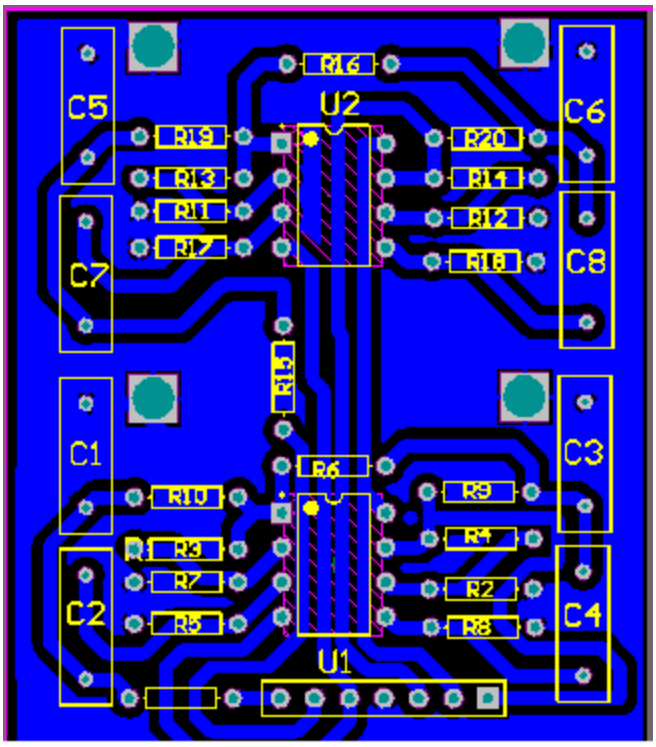

U3
C3
R41
P2
C13
R43
R44
C14
U7
U6
R45
D1
R27
R28
P7
R37
C12
R3
R47
C7
R38
C10
D2
R48
U8
U5
C11
R16
C9
C5
C8
R35
R25
U4
R22
R21
P1
P3
R5
R1
P6
U1
C2
R2
R11
U2
R14
R13
C1
R10



After this design I optimized the Sallen-Kay layout on a pcb. I will use the optimized version for Senior design.

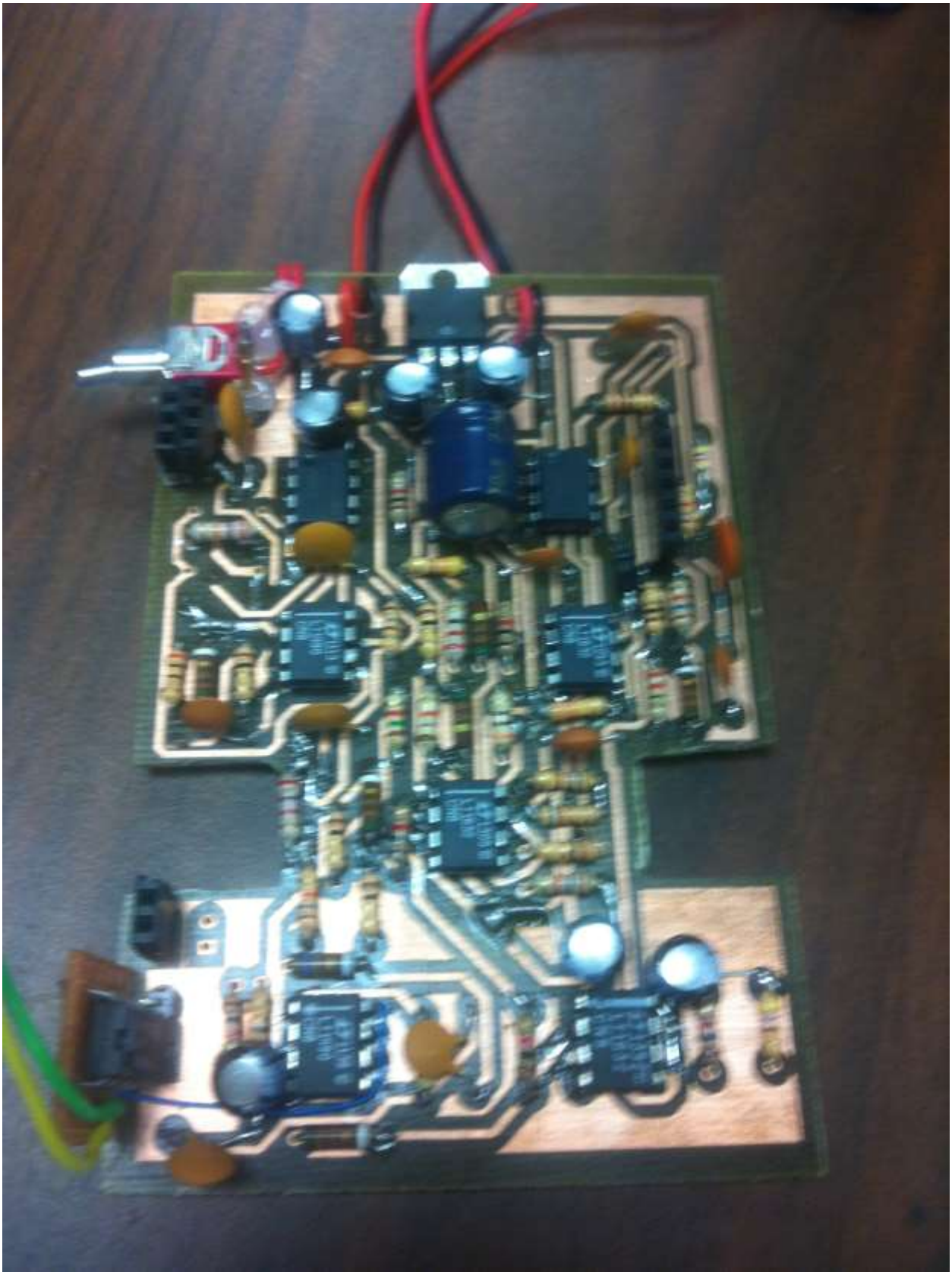
Below is another project with two simple two stage filters.





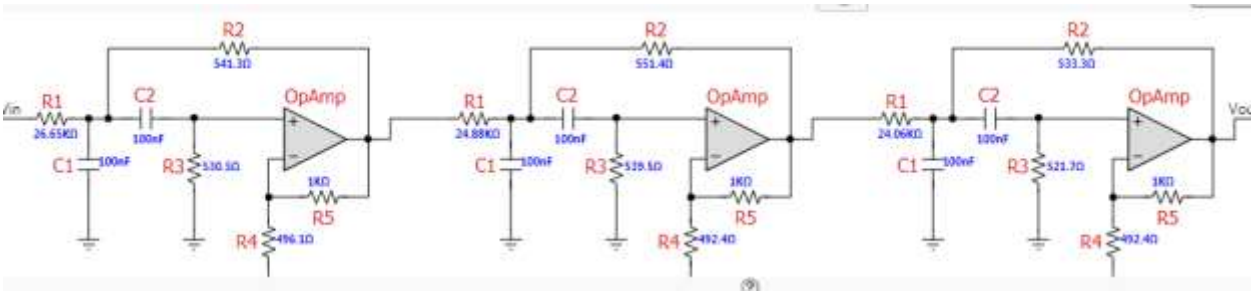
Above is the optimized layout I found. It uses one layer so a nice grounding plane can be used on the top layer. Unfortunately I rushed my design and did not use this board in Design 1. This made the circuit more complicated and harder to debug.

Final analog board

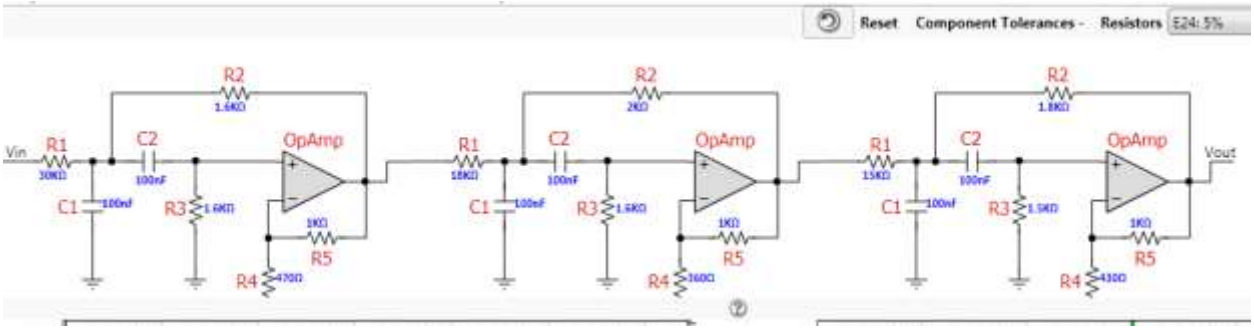


Filter design

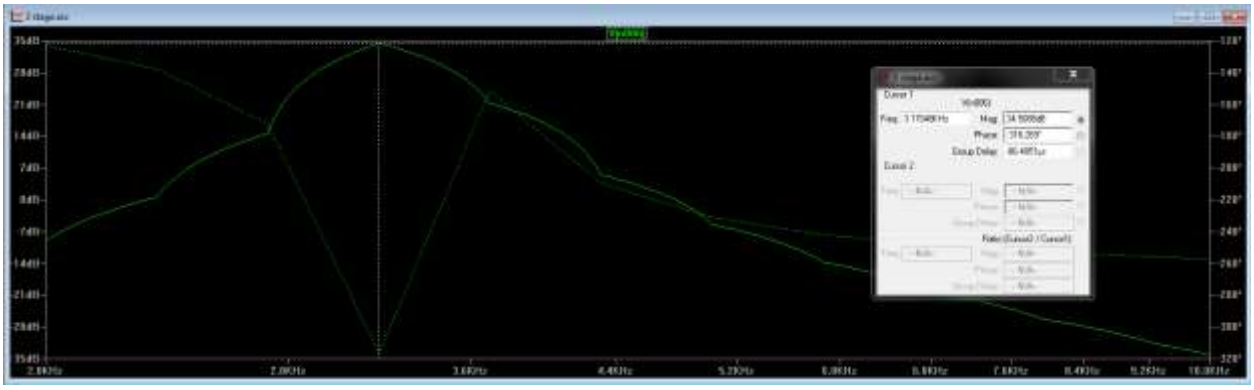
3k filter

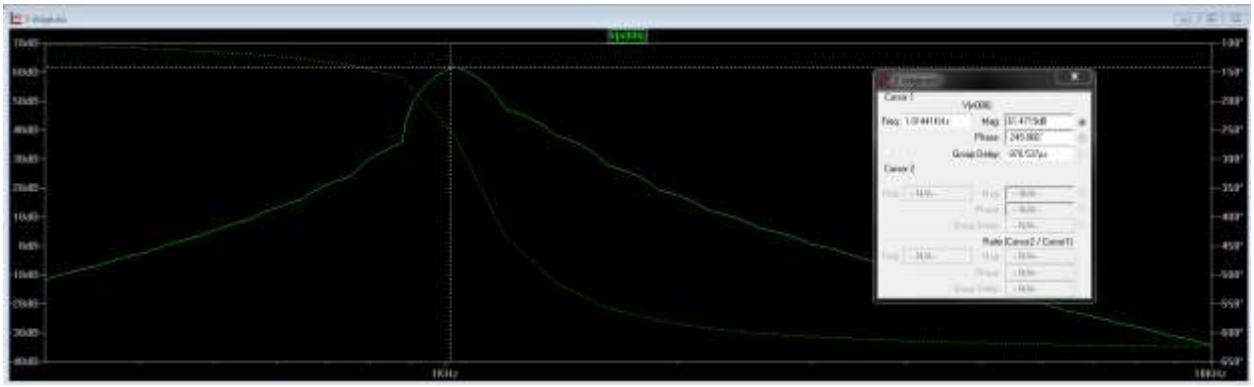


1k filter



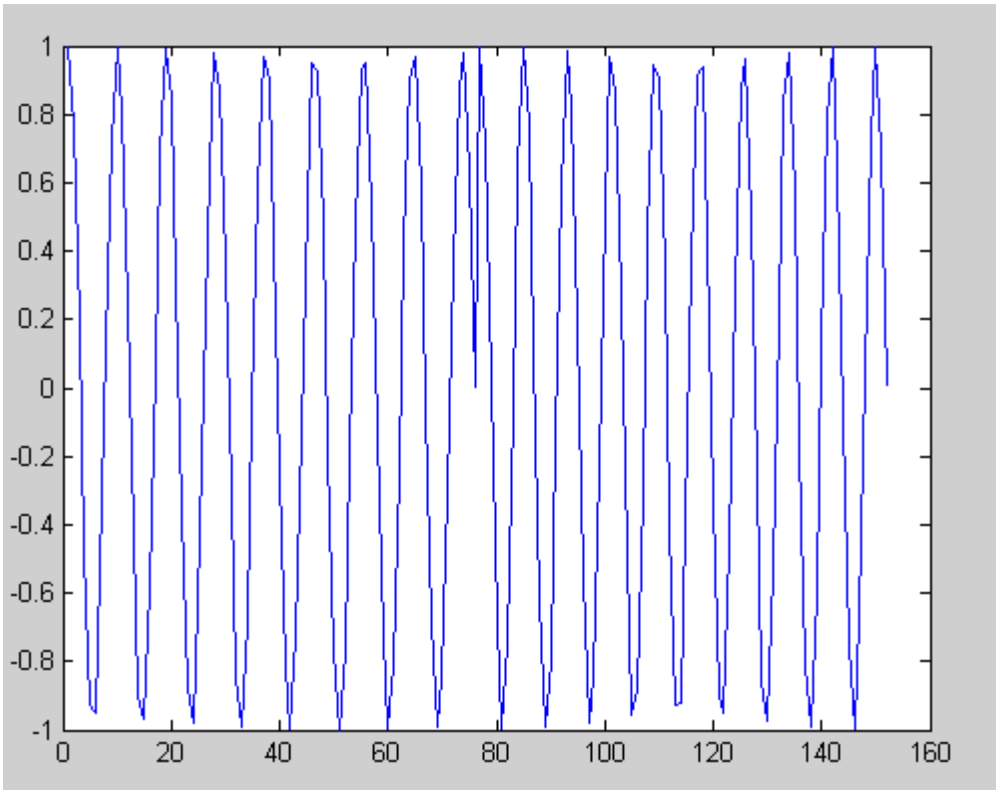
These were simulated and the gain adjusted slightly in simulations. Afterwards the gain was again adjusted on the final circuit.





MATLAB

I rewrote my script from signals to implement it for this project.



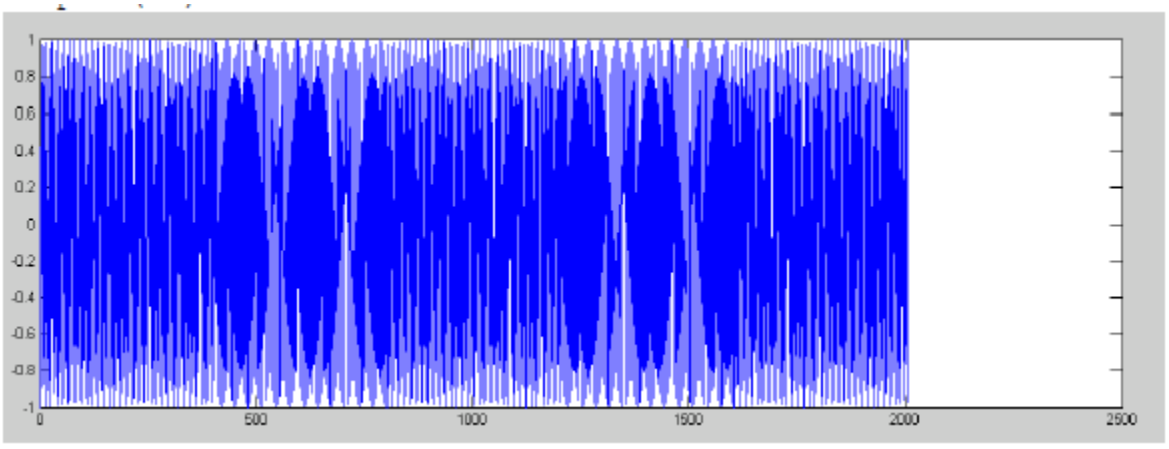
Notice the jump in the naïve implementation above. My code from signals did not fix this correctly. I fix this in the next code.

```
tn = 0:1/fs:(1/bps);
bitstr=[preamble bitstr endstring];

[b length]=size(bitstr); %number of bits to be sent
[b a]=size(tn);

freqonw=1000;
freqoffw=3000;

phaseon=2*pi*mod(freqonw*(1/bps-1/fs),1); %phase added after each onpulse
phaseoff=2*pi*mod(freqoffw*(1/bps-1/fs),1); %phase added after each off pulse

phase=kron(bitstr, phaseon)+kron(abs(bitstr-1),phaseoff); %array of phases per bit
[b c]=size(phase);
phase=[0 phase(1:c-1)];
phase=cumsum(phase);
phase=kron(phase,ones(1,a));

time=kron(ones(1,length),tn); %an array with time allocated for eachbit
[b a]=size(tn);
freqs=kron(bitstr,ones(1,a));
freqs=freqs*freqonw+abs(freqs-1)*freqoffw;
xx=cos(2*pi*freqs.*time-phase);
```

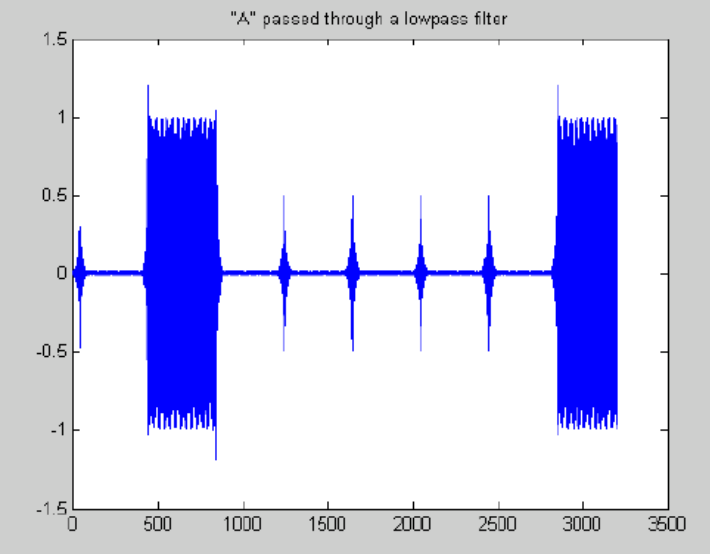
I then output the sound to a wav file.

```
xx=fsk_genc([message],44100,baud,1,Preamble,End);
plot(xx);
wavwrite( xx,44100,filename);
```

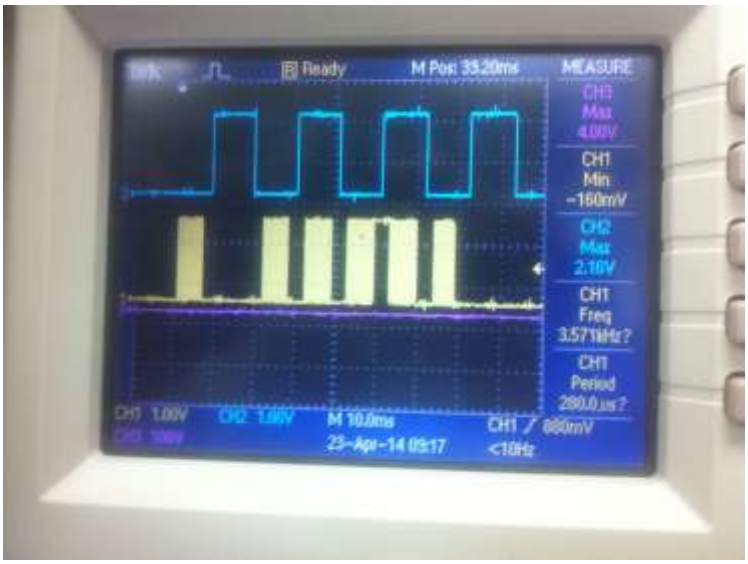
This would then be played through my speakers to the circuit.



Notice the faster and slower portions of this wave.



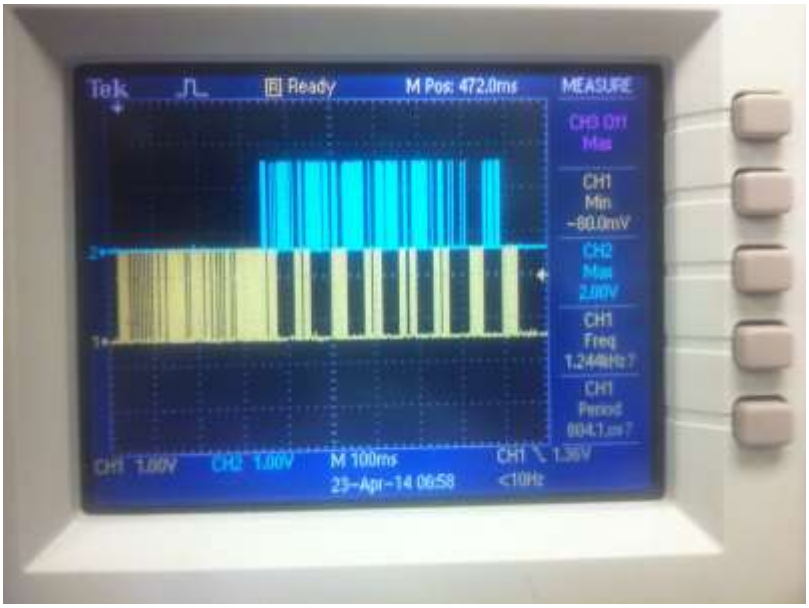
The above version is a filtered version of the character A.



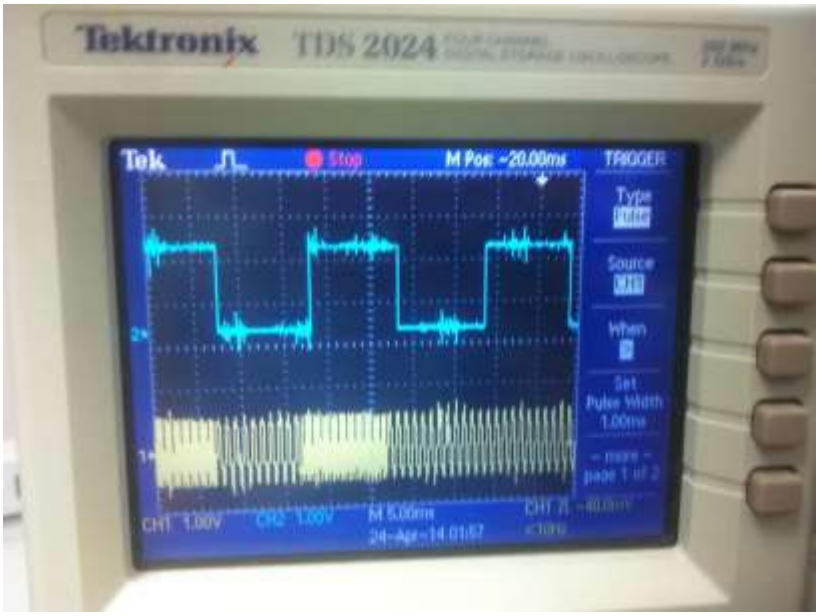
Above is a filtered version of the other channel in my circuit. I have an interrupt firing 160 times the baud of the signal. After a sync pulse it starts counting to 160 to capture the first bit. For debugging, my interrupt toggled every 160th pulse and output the sampled data as well.

I modelled UART in that a higher sampling is used and a sync occurs when data is detected. My code began smapling at 20% of the bit and stopped at 80%. That is why only the middle of the above yellow signals are on. The blue indicates when the sync occurred and where it thought every bit should be. The values inside were averaged and if the signal was on for the majority of the time then the bit was set to the corresponding channel (0 for the above example in yellow, and the blue signal would correspond to the on channel.) My circuit took a lot of tweaking. For a large part of the time only one or two characters would be received before the bit alignment went out of sync. This made me understand why

UART syncs at every bit. A naïve implementation of the standardized protocol would be out of sync quickly.



Here is a header used as sync and then blue corresponds to each bit captured.



Above we see the input to the analog filters (output from speakers) and each timing pulse the PIC sent out for the frame it expected a bit in. The analog signal would be split by the filters and digital signals would be sent to the PIC after that stage.

Above we see an intermediate output of both channels.

Overall

The sim900 made and received calls. The antennae used was not very good and I did not have good reception in the Design 1 room. It was also too noisy in my circuit. I need to learn more about analog ground planes and separating components on PCBs. I also should have had a low pass filter before the band pass filters. The call did not work during demo because of reception issues. I output the sound from my laptop directly to the circuit. I demonstrated an alphabet being sent, Dr. Turner's name, Kyle's name, "HELLO", and other test messages. My speaker made a tone on message reception but it was not loud enough. I demonstrated that it was indeed a sine wave outputting from the 386 and that my pots affected frequency and volume.

For improvements on this project I want to improve the sim900 board to have less noise and better reception. This would have completed the project. On the other boards I would like to have had debug pins for much more. I had to remove the lcd constantly to test it.

I already mentioned the analog and this was a big issue. I have started reading on ways to improve it. Seperating ground planes is a big one. They also suggest separating digital and analog sections of the board. I think stacking the boards added noise because performance was greatly reduced when I had the system together.

Parts:

Index	Quantity	Part Number	Manufacturer Part Number	Description	Customer Reference	Available Quantity	Backorder Quantity	Unit Price	Extended Price
1	2	SMF05CCT-ND	SMF05C.TCT	TVS DIODE 5VWM 12.5VC SC70-6		2 Immediate	0	0.55000	$1.10
2	10	MMBT2222ATPMSCT-ND	MMBT2222A-TP	TRANSISTOR NPN GP 40V SOT23		10 Immediate	0	0.12700	$1.27
3	5	1N4148WSFSCT-ND	1N4148WS	DIODE SML SIG 75V 0.15A SOD323F		5 Immediate	0	0.13000	$0.65
4	1	160-1446-1-ND	LTST-C191KGKT	LED GREEN CLEAR THIN 0603 SMD		1 Immediate	0	0.32000	$0.32
5	2	160-1434-1-ND	LTST-C190KFKT	LED YELLOW ORANGE CLEAR 0603 SMD		2 Immediate	0	0.28000	$0.56
6	2	160-1447-1-ND	LTST-C191KRKT	LED SUPER RED CLR THIN 0603 SMD		2 Immediate	0	0.32000	$0.64

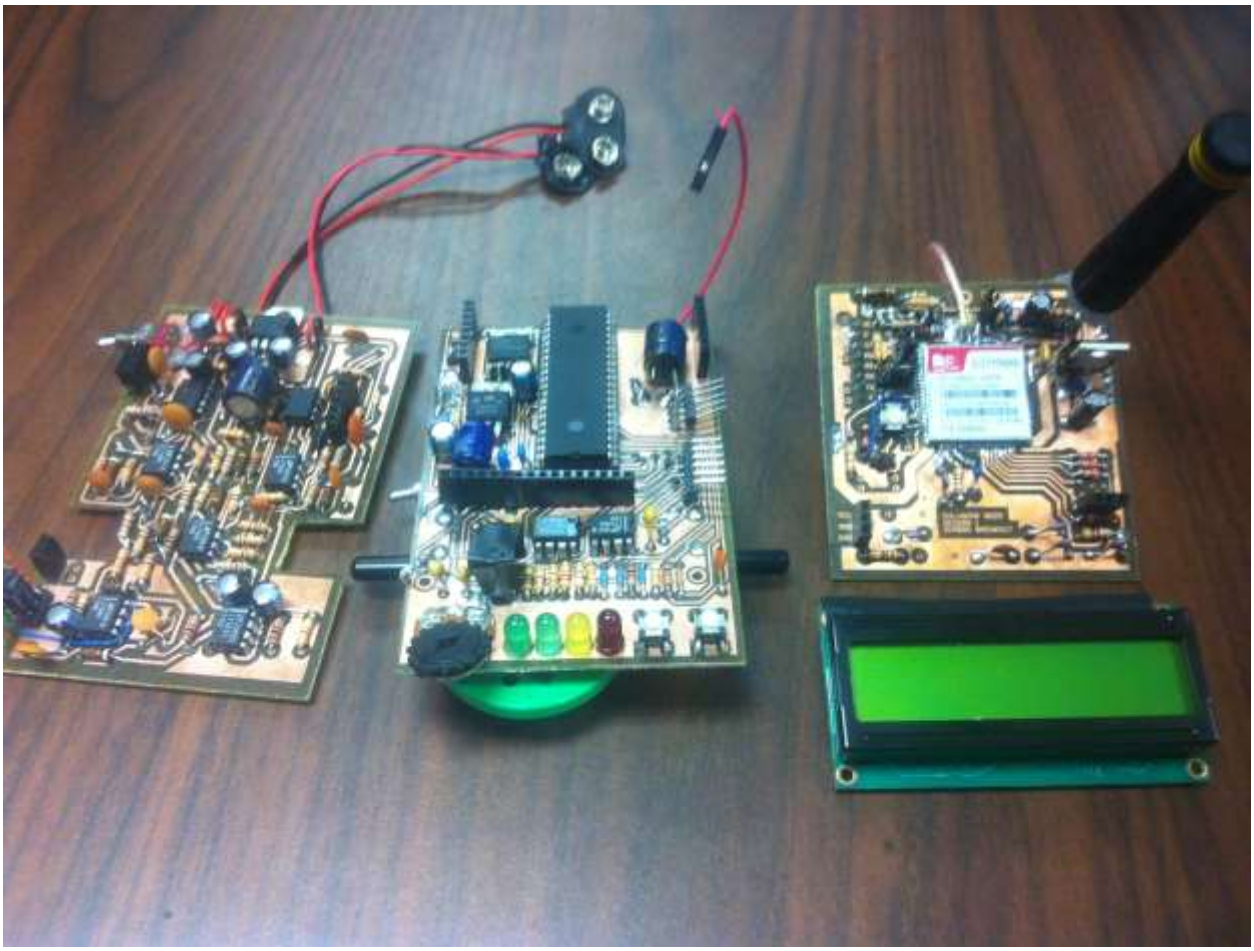
Sim900 chip: $16 includeing shipping

BT29302BT regulator: 4.95 including shipping

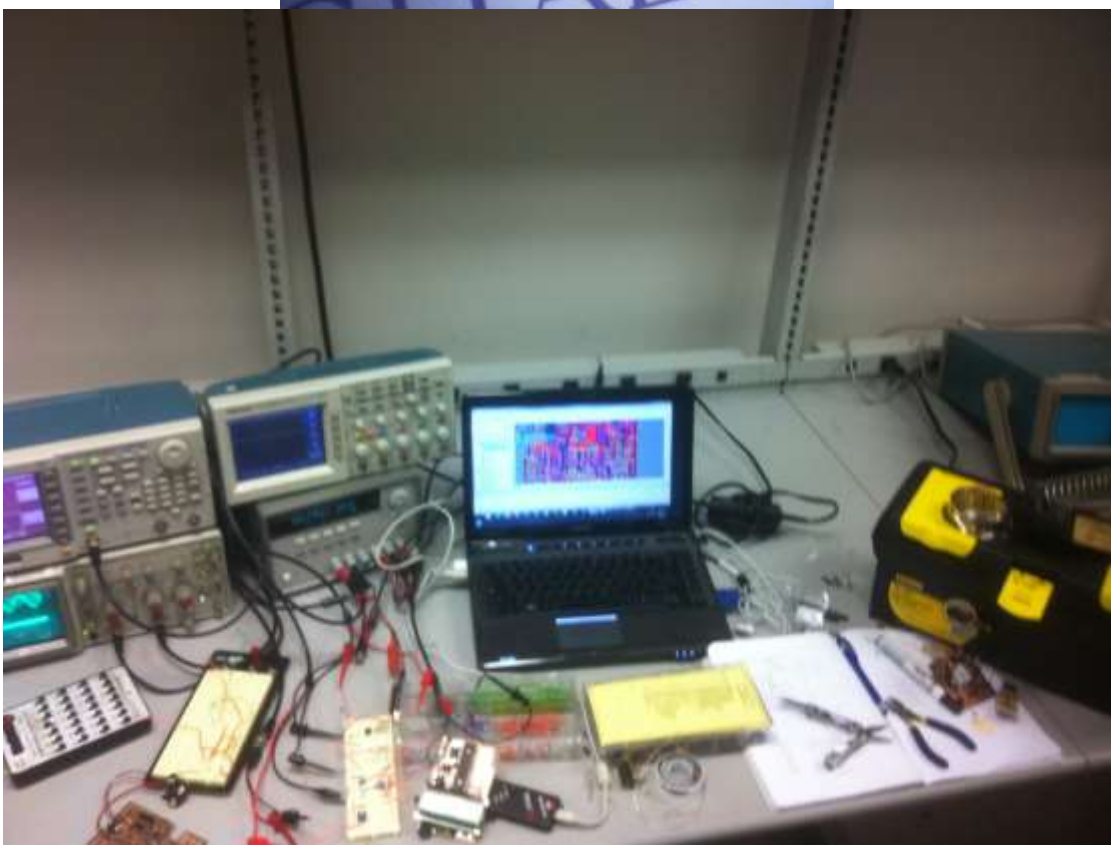
Other devices such as sim card holder, antennae connectors, antennae, and various devices did not exceed 15 dollars.

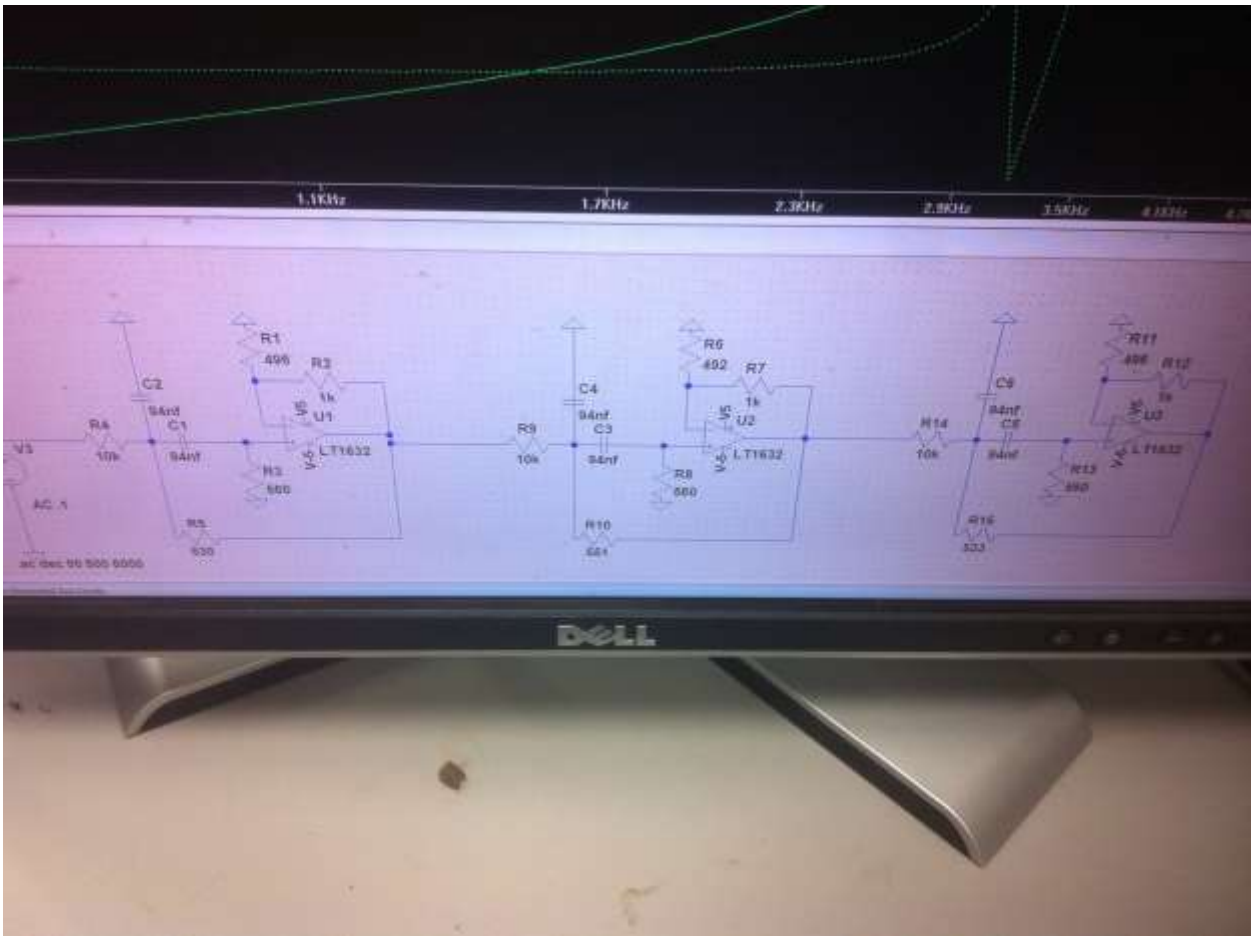
Resistors and caps from lab fee in Design 1.

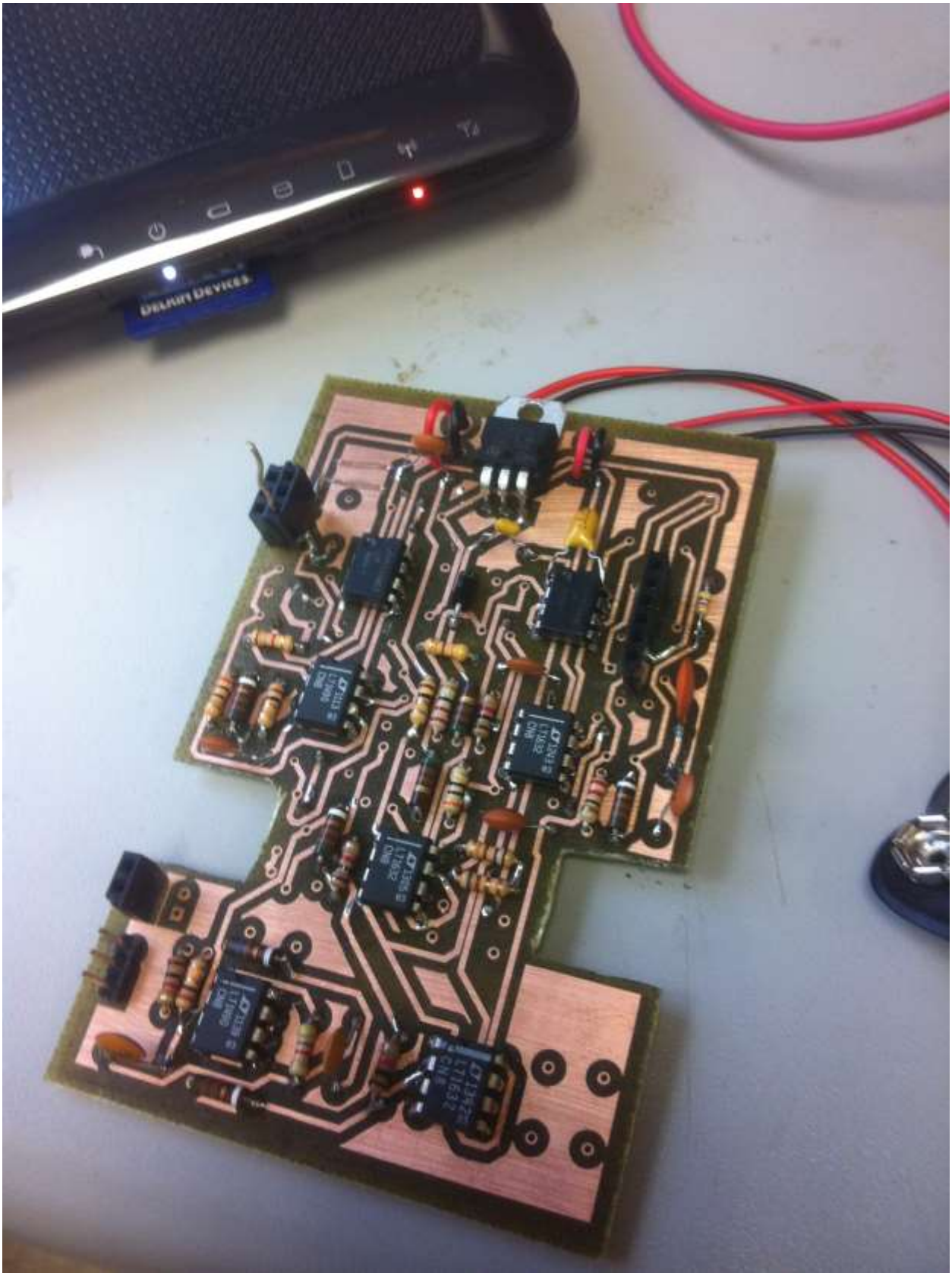
The Op amps and amplifiers were from Eric Liebner.



Images




1.1KHz
1.7KHz
2.3KHz
2.9KHz
3.5KHz
R1
49R
R3
1k
U1
LT1632
C2
94nf
C1
94nf
R4
10k
V1
AC 1
R2
560
R5
530
R9
10k
C4
94nf
C3
94nf
R10
661
R6
492
R7
1k
U2
LT1632
R8
560
R14
10k
C9
94nf
C8
94nf
R16
523
R11
49R
R12
1k
U3
LT1632
R13
560
DELL


DELUXE DEVICES