Team π

Final Report

EEC 134AB

By: Oleg Khokhlan
Chuong Khong
Lhawang Thaye
Ferris Chu

June 17, 2016

Abstract

This paper will discuss the design, implementation, and testing of a 24 GHz UAV radar system. The goal of this project is to design a small, compact radar that could be mounted on a UAV drone which may be use for collision avoidance applications, and range and speed measurements applications. The overall design of this radar utilizes the Infineon's 24-26 GHz MMIC transceiver, BGT24MTR11. The reasoning behind making this chip the central part of our design is because the transceiver has all of the necessary components of a FMCW radar placed in a small, compact package. Considering the high operating frequency of this transceiver, the advantages lie in the fact that a custom designed microstrip patch array antenna would be small enough to fit within the dimensions in the design specification.

Introduction

Design Specifications

The required specifications for this project are as follows:

1. At least 20 meters of range for a target with an RCS of 1 m^2.
2. Must fit within the form factor of a GoPro (41 mm × 59 mm × 30mm), 152 g.
3. Include an on-board processor to process and send information with a greater than 10 Hz refresh rate or save the raw data for computer processing.
4. Must develop the receiving end of the communication link and demonstrate the working system on a computer.
5. Needs to work from a 12V, 1 A power supply.

To meet required specifications, this design had to be compact and lightweight. This design was also intended to improve on the design from previous years. Following in the steps of previous teams, we separated our system between 3 boards: RF board, Baseband, and Signal Processing in order simplify the system. Our team designed and used 2 sets of 4x4 array patch antenna for transmitting and receiving the signal. We used ANSYS HFSS to simulate the patch antenna array, Cadence Allegro to design the RF board, and had the board fabricated by Bay Area Circuit on a 10 mil thick Rogers RO4350B board with 1 oz copper. This patch antenna array was implemented on the RF board, where the Infineon BGT21MTR11 MMIC transceiver and the rest

of the RF circuit was located. The Baseband board includes the voltage regulator, signal generator, filter and amplifiers. The Signal Processing board stores the raw data to an SD card for processing on the computer. We used KiCad to design the PCBs which were fabricated by Bay Area Circuits.

Block Diagram

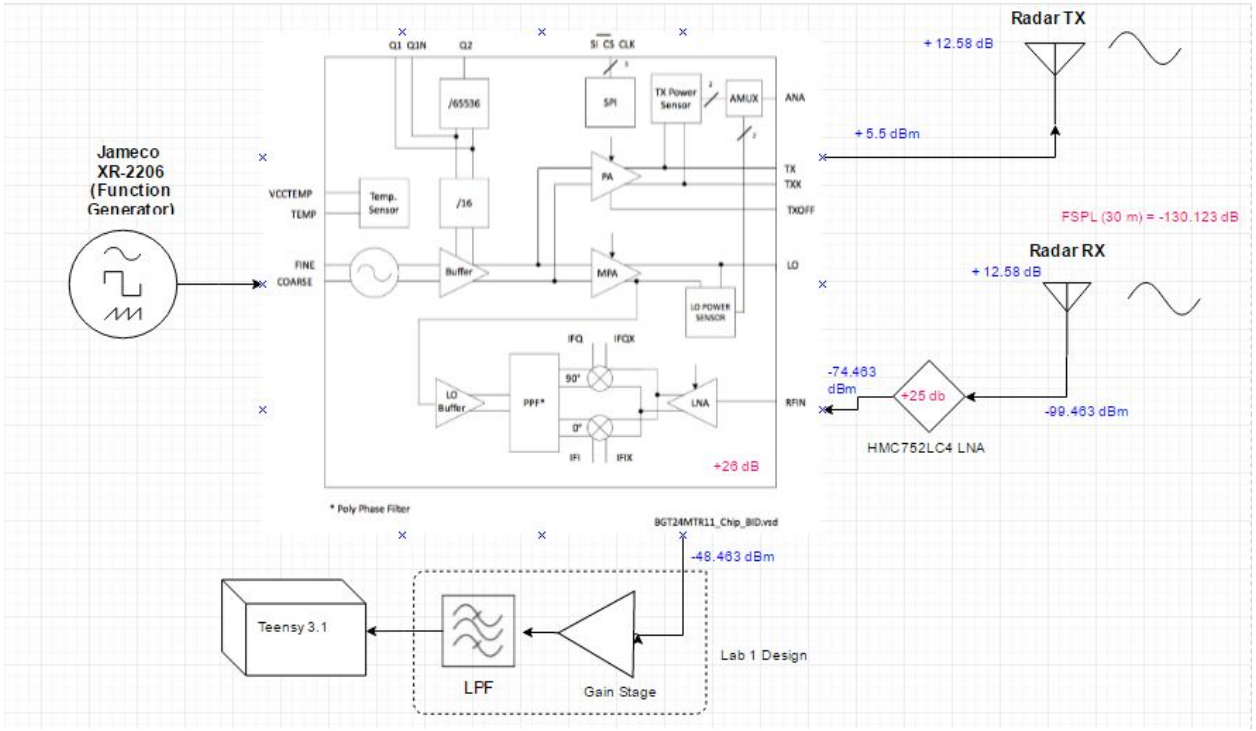

Fig. 1 - Block Diagram

Design

After reading previous years design, we decided to use separate boards as well (RF, Baseband, Signal Processing). The benefit is that simple mistakes cannot affect the entire system, and parts of the baseband board can be temporarily substituted with a breadboard or prototype equivalent. Oleg had designed the baseband board and the board that housed the Teensy microcontroller and Chuong had designed the RF Board and the microstrip patch antennas. Heavy communication was imperative to make sure that the pins of the boards would line up and so signal transmission would be fluid.

RF Board Design

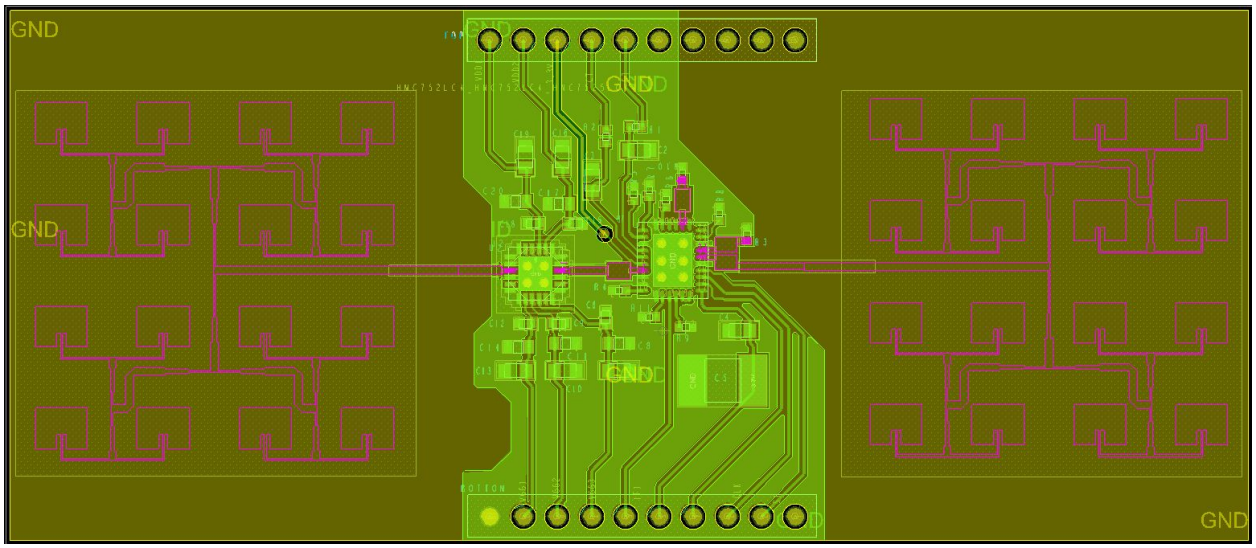

Fig. 2 - The first variation of the RF board. This version has the 24-28 GHz LNA

The most important and critical aspect of our system is the top layer RF board. The RF board carries 2 sets of 4x4 microstrip patch antenna array, the BGT21MTR11 MMIC transceiver, and the necessary resistors and capacitors used to bias the DC inputs. The goal of the RF Board was to transmit a frequency modulated (FM) signal that varies between 24 - 26 GHz through one set of the 4x4 microstrip patch antenna and to receive the reflected signal through a second set of 4x4 microstrip patch antenna. The reflected signal is taken into the RX port of the BGT21MTR11 transceiver, amplified by the LNA, and downconverted by the mixer. The IF (Intermediate Frequency) signal, which is the signal that comes out of the mixer that is located inside of the transceiver, is sent to the baseband system that is located on the PCB board directly

under the RF Board. The RF Board was designed to be fabricated on 10 mil thick Rogers 4350B substrate with 1 oz copper cladding. Our group decided to use the Rogers 4350B material over the standard FR4 substrate for the RF Board because the 4350B substrate is less lossy at 24 GHz than compared to the FR4 substrate. For our design, we had fabricated two different RF boards. One board had included an additional low noise amplifier on the receiver side while the other one didn't include a low noise noise amplifier. Our group decided to use an additional low noise amplifier to provide more gain instead of a power amplifier because the LNA had less power consumption and the LNA was more readily available. Also, the additional amplifier would significantly improve the performance of the radar.

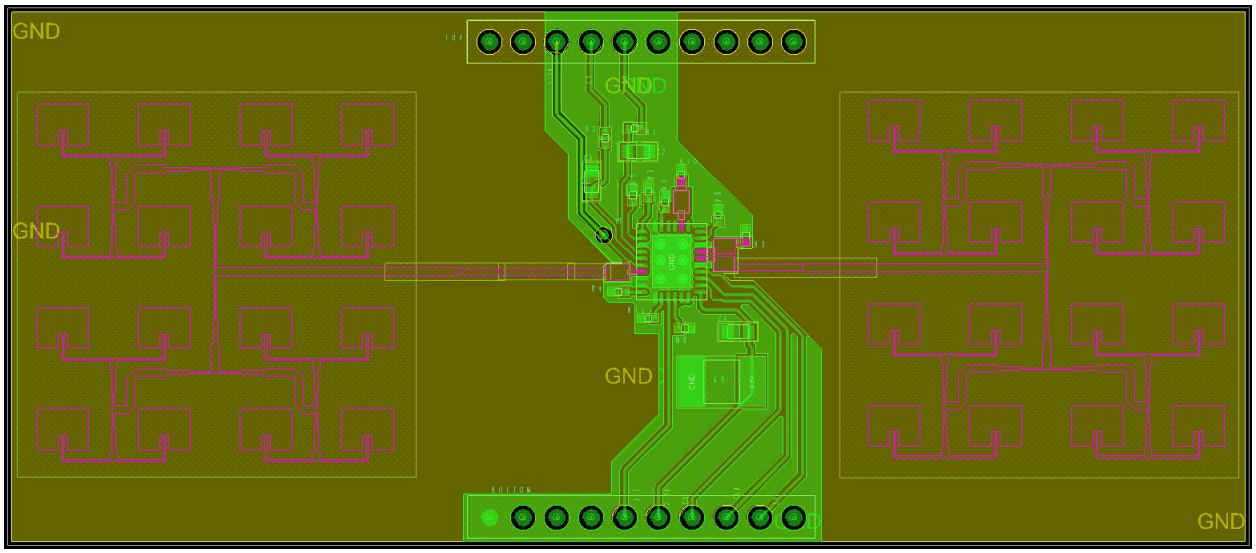

Fig. 3 - RF board without an LNA attached.

The reason why the transceiver was chosen to be used in our design was because it greatly simplified the design of the system. The transceiver contains a voltage controlled oscillator, power amplifier, low noise amplifier, and a mixer packaged in a 5.5 mm by 4.5 mm integrated circuit. All of these devices are common components that are used in the implementation of FMCW radars. By using the transceiver, we are able to avoid any possible complications that could arise from designs that implement these components separately such as soldering errors and the negative effects of parasitics, especially considering that our operating frequency is in the K-band.

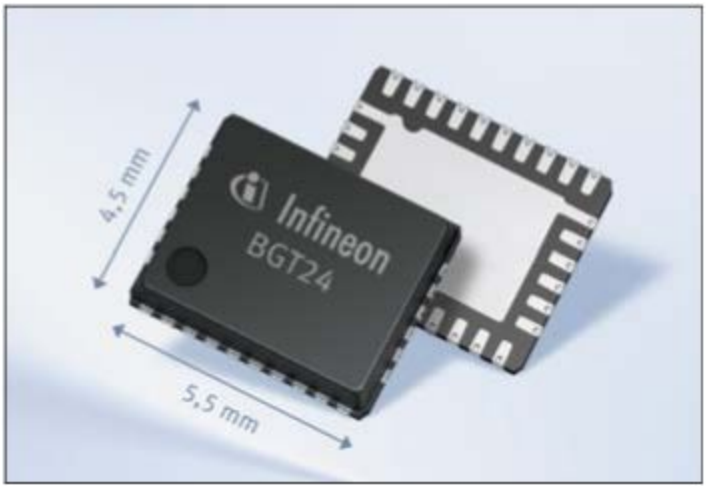

Fig. 4 - Infineon BGT21MTR11 Transceiver

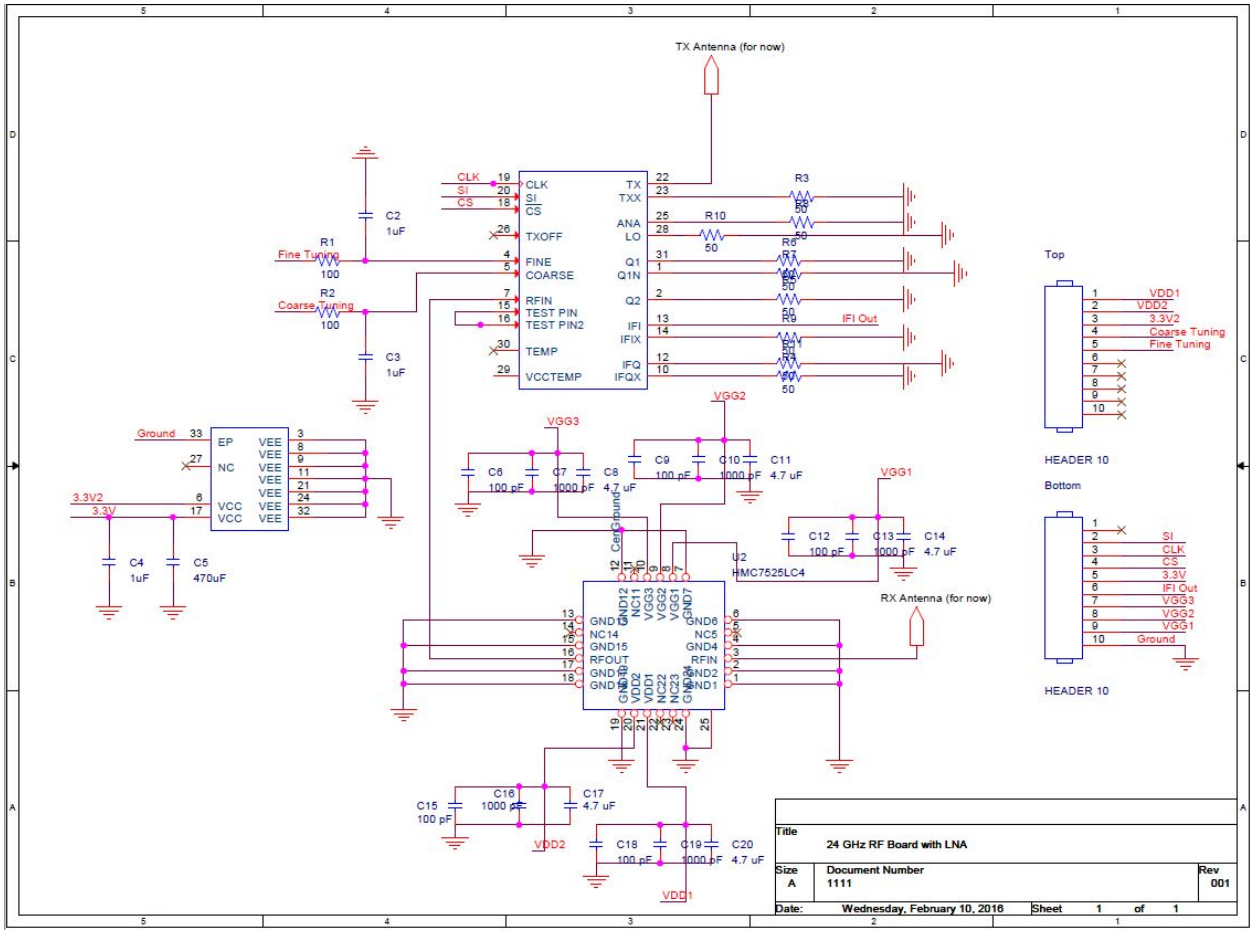

Fig. 5 - Schematic of the RF Board with LNA. Designed in OrCAD Schematic Capture

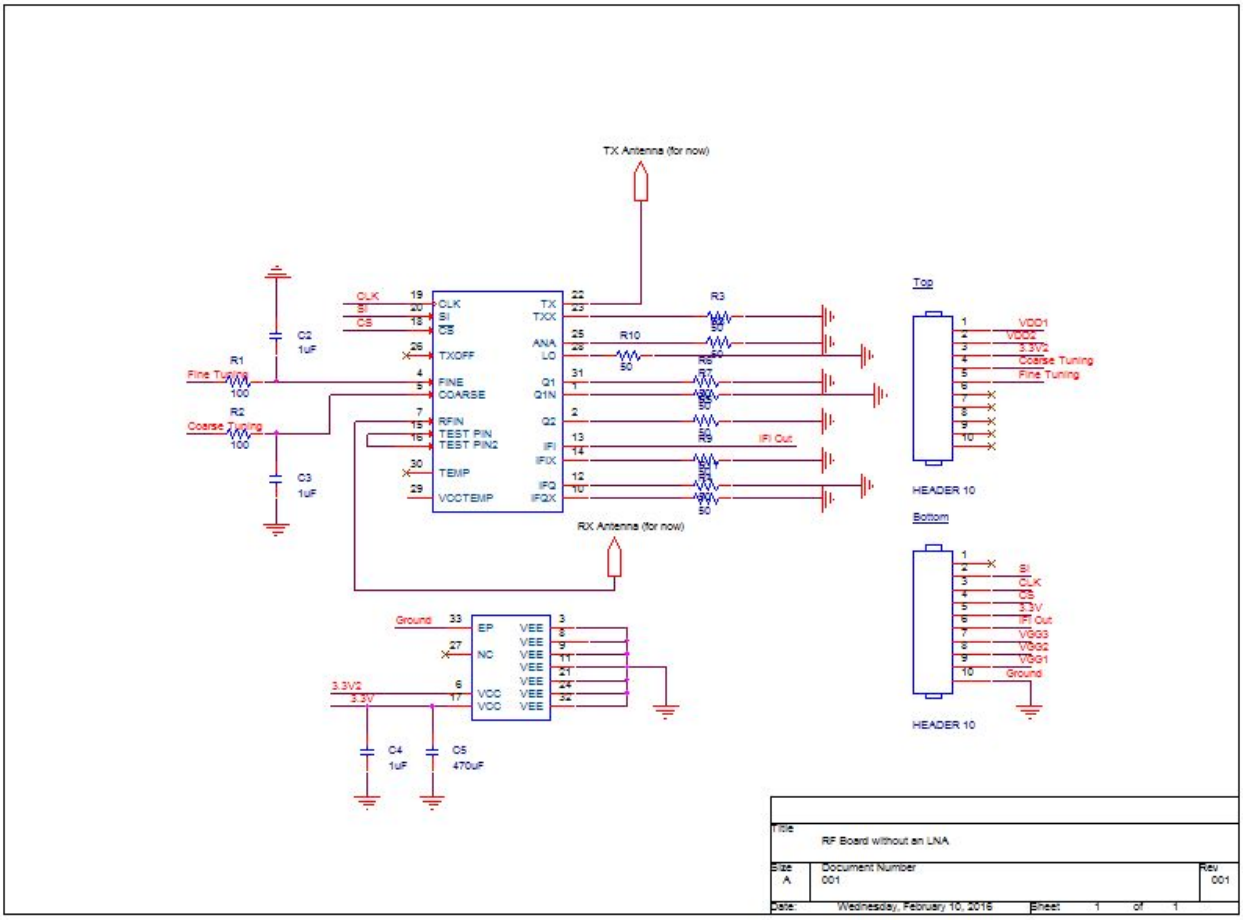

Fig. 6 - RF Board without an LNA, designed in OrCAD Schematic Capture

Patch Antenna Array

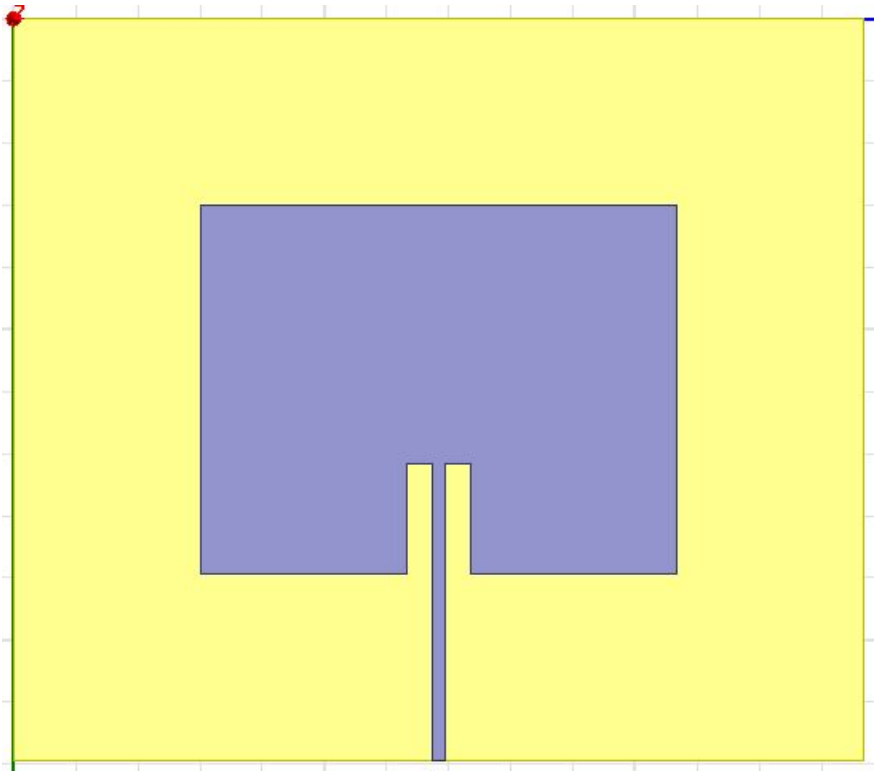

Fig. 7 - Patch Antennas

The 4x4 patch array microstrip patch array antenna was designed through HFSS and imported into Cadence Allegro so that it could be fabricated on top of the RF board. Due to the narrowband nature of microstrip antennas, the antenna was designed to have a center frequency of 25 GHz so that its bandwidth would span throughout the operating frequency of the transceiver. We came to this decision after looking at the transceiver's data sheet, we saw that the VCO inside of the transceiver operates only between 24-26 GHz. Thus, our group decided that the most optimal design would be to design the microstrip antenna so that its center frequency would be at 25 GHz, which would allow the entire bandwidth of the antenna to operate within the operating range of the transceiver's VCO. Due to the microstrip antennas operating at 25 GHz, the dimensions of the patch antennas are very small. Thus, we were able to fit at least 2 sets of 4x4 patch array antennas onto the board, while still meeting the designs specifications. The microstrip patch array utilizes a corporate feed network that feeds the signal to each element of the array. The feed network uses a 50 Ohm feedline, 100 Ohm line, and a 70.71 Ohm quarter wavelength transformer that transforms the 100 Ohm line back into a 50 Ohm line. The input impedance of each of the microstrip antennas are designed to be 100 Ohms in

order to eliminate the need for an additional quarter wavelength transformer. Each set of 4x4 patch array antennas is matched to the TX and RX ports of the BGT21MTR11 transceiver. The matching network for the transceiver was provided in the Infineon's datasheet and was followed exactly as shown. The dimensions of a single microstrip patch antenna can be readily calculated by hand, or it can be found by using an online calculator (**http://www1.sphere.ne.jp/i-lab/ilab/tool/ms_line_e.htm** and **https://www.pasternack.com/t-calculator-microstrip-ant.aspx**). In our design, we had utilize an online calculator to get a rough approximation of the dimensions and then tuned the dimensions till the desired S11 is reached with it being resonant at 25 GHz. The required parameters needed to calculate the dimensions are the resonant (center) frequency, effective dielectric constant, and the dielectric thickness. The dimensions of the 50 ohm, 70.71 ohm, and 100 ohm lines can be calculated using a calculator provided by Rogers Corporation, assuming that you're using their material, or it can be calculated by using ADS' LineCalc program.

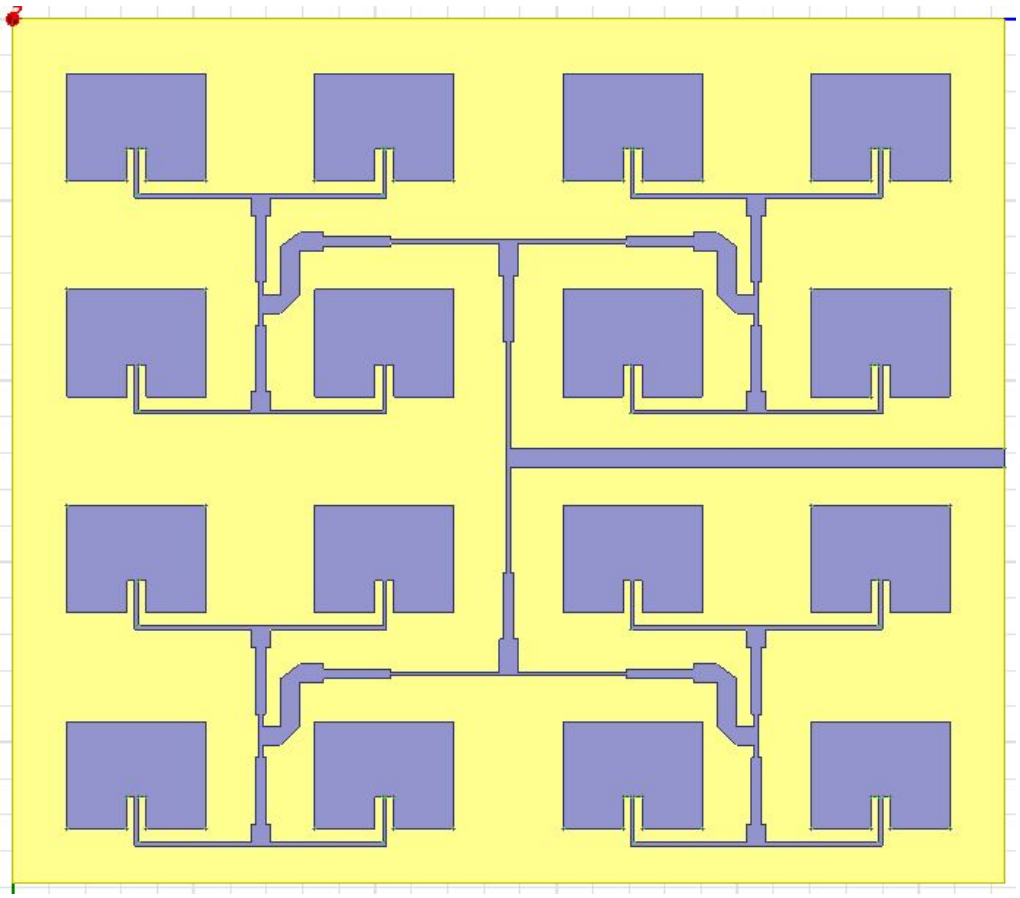

Fig. 8 - Patch Array Antennas, designed on HFSS

Baseband Design

The role of the baseband is to provide the basic needs of the circuit. This includes power input, regulating the voltage, controlling the signal generator, receiving the signal from RF board, filtering the signal, and directing the filtered signal to the signal processing board. It is the "middleman" between RF and Signal Processing.

Voltage Regulation

We are provided with a 12 V 1 Amp power source. Some components require a voltage of 12V, 5 V, 3.3V and 3 V. A total of 3 voltage regulators were used to convert the 12 V to the desired voltage. The voltage regulators were chosen by price and availability. Power consumption and heat dissipation were a concern, however, we have no way around power consumption since we needed the voltage regulators. As for heat dissipation, the voltage regulators we used were through-hole and had a large metal backside for heat dissipation. We bent the voltage regulators out into the open to help keep them cool. (See image)

1) LM317 (also used in lab) was chosen for the 3 V regulator because the SMD regulator we bought had an incorrect data sheet, which made the footprint tricky.
2) LM2937ET was used for 3.3 V. It looks like the LM317, but with slightly different pin assignments.
3) L7805CV was used for 5 V.

Signal Generator

We used the XR-2206 Analog Signal Generator because a previous team recommended it. Aside from its simplicity, the noise created by the XR-2206 is much less than the noise that would be created by using SPI. The simplicity comes from the fact that no programming is required for the XR-2206 to run. In the datasheet, you can find sample circuit layouts which describe the function of XR-2206 in that particular setup. We used Figure 11 from the datasheet as our design in the PCB. It gives us a triangular output which is sent to the RF board (top board). The equation $F = \frac{1}{RC}$ set the frequency. This means that the frequency can be manipulated and adjusted if you can change R or C. We chose to set C to a constant value, and used a variable resistor for R so we can change the frequency. **Important:** When you set C (since you already know what frequency you will use), solve for R. Make sure that R value found is reasonable. If not, adjust the C and redo the calculation.

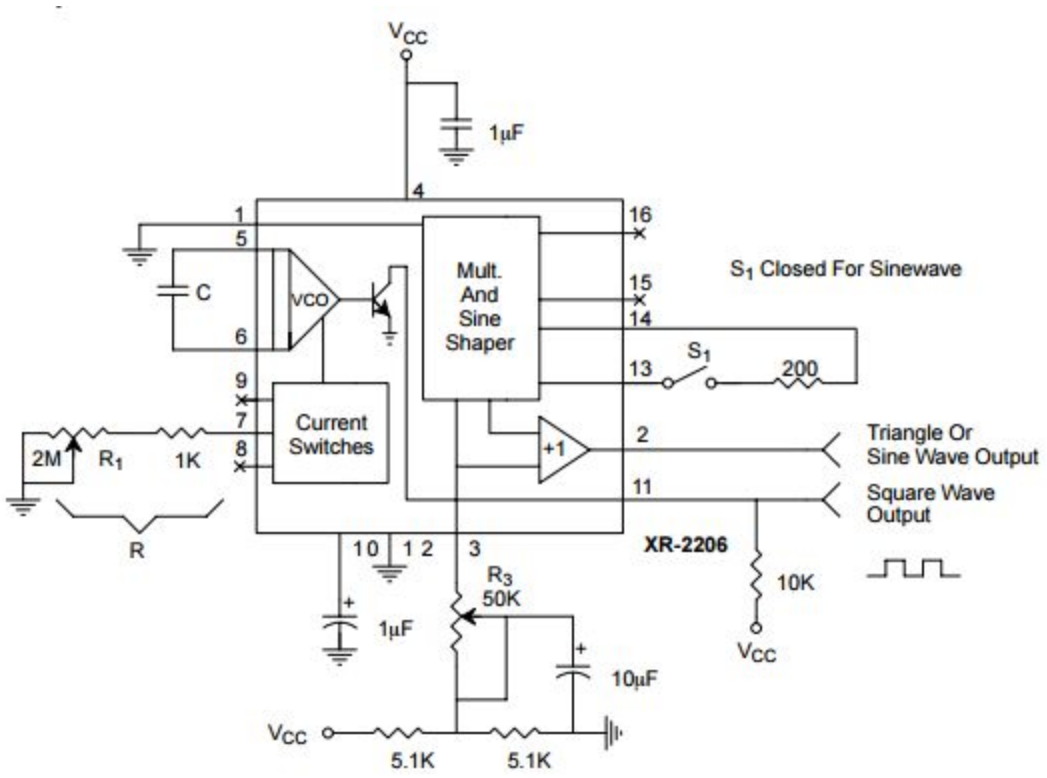

Fig. 9 - Figure 11 from XR-2206 Datasheet

Amplifier & Low Pass Filter

Once a signal comes from the RF board (top) board it is taken through the low pass filter and a 2-stage gain op amp to amplify the signal from processing. We used the same Op Amp and Filter design as in lab 6. We thought it would be best to use components we had experience and success with.

Header Pin Location and Stacking

Since we planned to stack the boards together, precise measuring and head pin placing was vital. We decided to place the pins in two columns on the left and right, and also a row of pins on the top. That way we had a solid connection, and a solid frame. Note: It is best to pick pins which are very long so that large components will not be an issue. What we did was decide to put female headers back to back and solder them to the baseband while the bottom and top boards have longer pins. (see image)

Baseband Schematic Design

For the baseband, I used Kicad to design the PCB. It is simple to learn and use. See related app notes for a detailed discussion of the design with KiCad.

Schematic of the Design: (see Design File for a closer look)

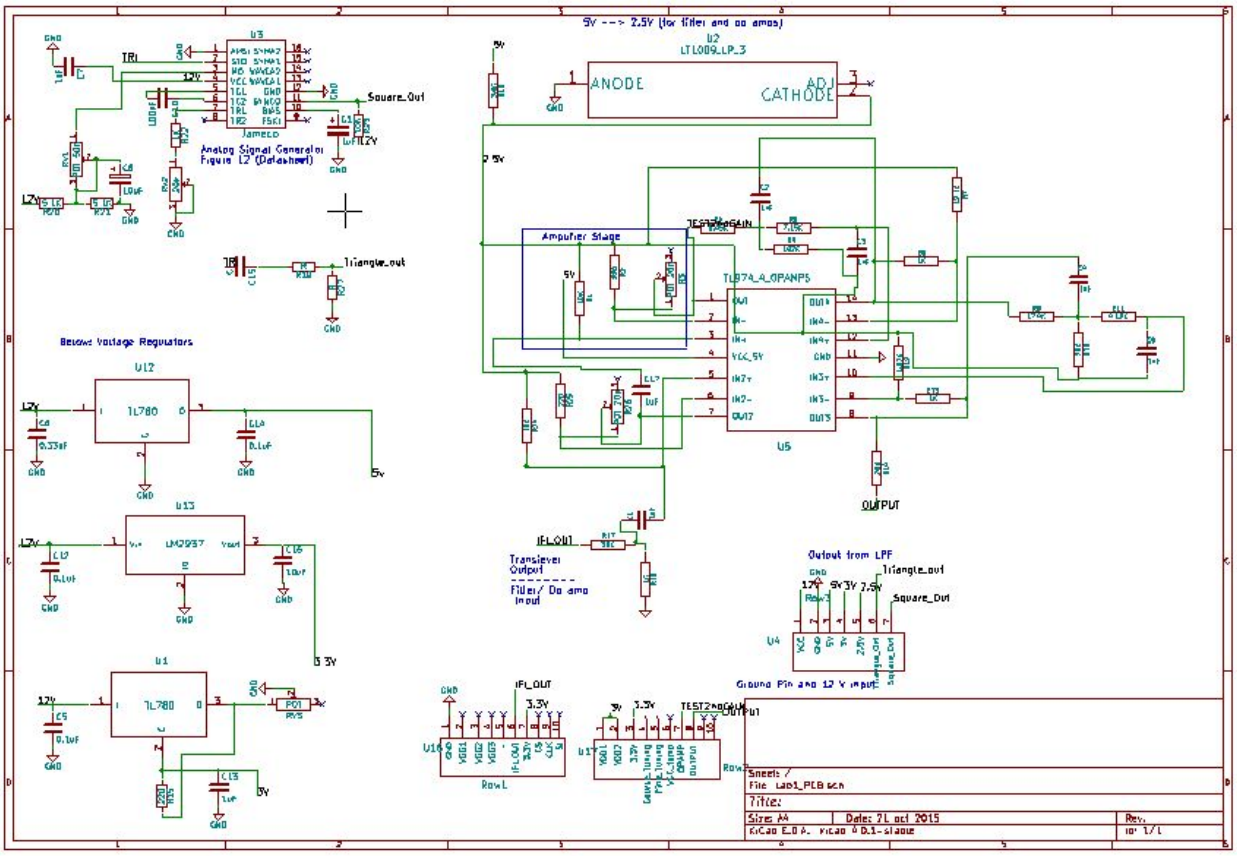

Fig. 10 - Baseband Schematic

Signal Processing

The role of the Signal Processing board is to use the Teensy 3.1 / 3.2 as both an ADC and a data storage device through SPI. The analog data coming from the output of the filters and amplifiers is sent to the analog input pin of the Teensy. Pressing the reset button on the Teensy will initialize and confirm that an SD card has been inserted, and the Teensy will begin taking samples and storing data. The Teensy uses a sampling system to obtain data points from the input analog signal and calculates a value with mV units from the number sampled. These data points are then stored to an SD card through an SD card-module using SPI protocol. This data, saved in a .txt file, is then saved onto the SD card for computer processing.

We did not complete the signal processing portion of the project but a way to implement this would be to take the .txt file that we saved from earlier and convert this to a .wav file using MATLAB. The code for this is very simple and is posted below. The data processing can then be done using Python, just like that which was done for quarter 1. This would give us a range versus time plot and a velocity versus time plot which we can then use to gauge our radar's performance.

```
//Converts .txt file to .wav file. The .txt file must be in the matlab folder you saved and
//the .wav file will save in that same folder.
A = importdata(filename.txt)
//Fs is sampling rate and A is array that is the output of the .txt file
audiowrite(filename.wav,A,Fs);
```

Budget

The BOM below is all of the components we purchased for our project. However, we also used components from the class inventory which are not included in the list.

Item	Quantity	Unit Cost $	Total $
Transciever	20	$19.96	$399.20
3.3V Regulator	3	$1.53	$5
5.1k resistors	5000	-	$6.46
20k Resistor	15	$0.10	$1.50
50k Trim Pot	12	$1.23	$14.76
5V Regulator	3	$0.44	$1.32
3V Regulator	3	$0.72	$2.16
100 Ohm	15	$0.10	$1.50
470uF	4	$5.34	$21.36
100pF	15	$0.10	$1.50
4.7uF	15	$0.77	$11.55
Jameco	2	$7.25	$14.50
			$480.40

Fig. 11 - Bill of Materials

Implementation

RF Board

The RF board was laid out in a way such that the transceiver was at the center of the design and the patch antenna array laid on both sides of the transceiver. This layout worked best because the TX and RX ports laid on opposite sides of each other. Another design choice was to lay the capacitors as close to the transceiver as possible. The 50 ohm terminations were also placed as close as possible to the transceiver in order to minimize the effects of parasitic capacitances in the RF lines. Laying out the traces and microstrip lines was a challenging task due to the pins of the transceiver being extremely close together. However, this was accomplished by varying the width of the traces for the DC line in order for it to pass the DFM check done by Bay Area Circuits. This is possible because the widths of the DC line has no effect on the line impedance. The dimensions of the microstrip lines, which are the 50 ohm feedlines (RF lines) of the antennas was calculated by using LineCalc, which can be found in Keysight's Advanced Design System (ADS) software. For the purpose of the 50 Ohm feedlines, the width is the most important aspect of the microstrip line because varying the width results in a change in the line impedance. There was a slight difference in widths of the matching network provided by the transceiver data sheet and the feed lines to a 25 GHz patch array antenna, but the difference is negligible. The connection between the antenna and the RX ports are created by setting the "Net name" of the RX port and the antenna to the same name. This could easily be done in Cadence Allegro.

The more difficult part in the implementation of the RF board was to successfully solder the transceiver onto the printed circuit board (PCB). Although the RF board was already coated in a layer of tin, provided by BAC, the melting point of the tin is noticeable higher than the maximum temperature rating of the transceiver. As a result, if care is not taken then the most likely outcome will be that the transceiver becomes broken due to prolonged exposure to excessive heat. Chuong found that the most effective way to solder the transceiver onto the RF board is to apply a different coat of solder, such as "Low temperature solder paste". In addition to using low temperature solder paste, Chuong had use a hot air gun, which was set to around 100 degrees celsius , to melt the solder instead of placing the board on a hot plate. After the board is tinned with low temperature solder, the next step is to place flux over the pads where the transceiver will be soldered to. The reason for this is because the flux is intended to hold the IC to the board during the soldering process and that it will assist in lining up the pins once it begins to melt. Although this process might take a little longer, it eliminates the possibility of destroying

the transceiver due to heat exposure. It should it noted that there should be a solid connection between the pins of the transceiver and the pads of the PCB and that it should be checked, along with shorts, prior to powering the transceiver on.

Baseband Board

This section will discuss PCB layout and soldering.

The goal of the PCB layout was to have the pins on the sides and at the top. All components were placed as close as possible to their voltage source which comes from the voltage regulators. The voltage regulators were placed on the sides to help with heat dissipation.

PCB View
(with copper fill and solder mask)

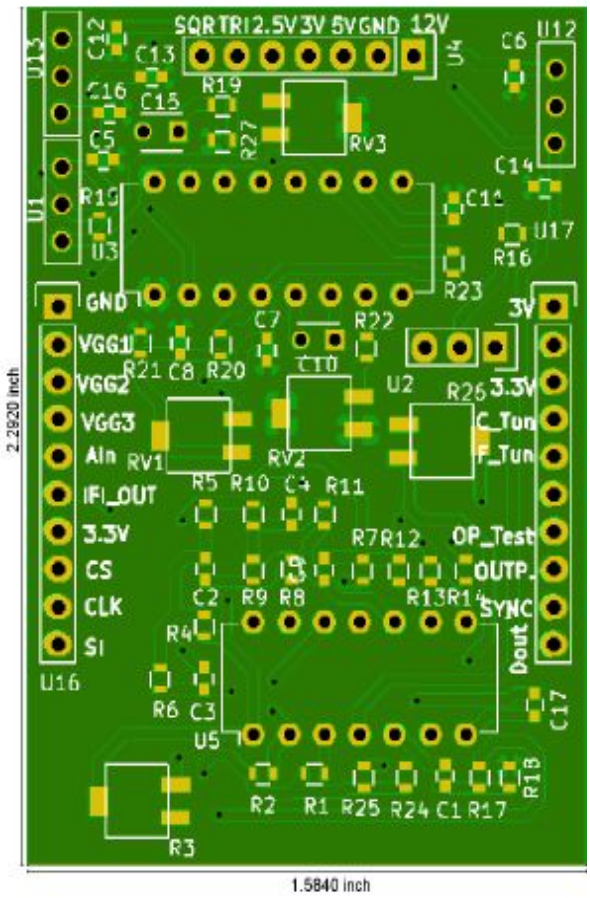

Fig. 12

PCB View
(without copper fill)

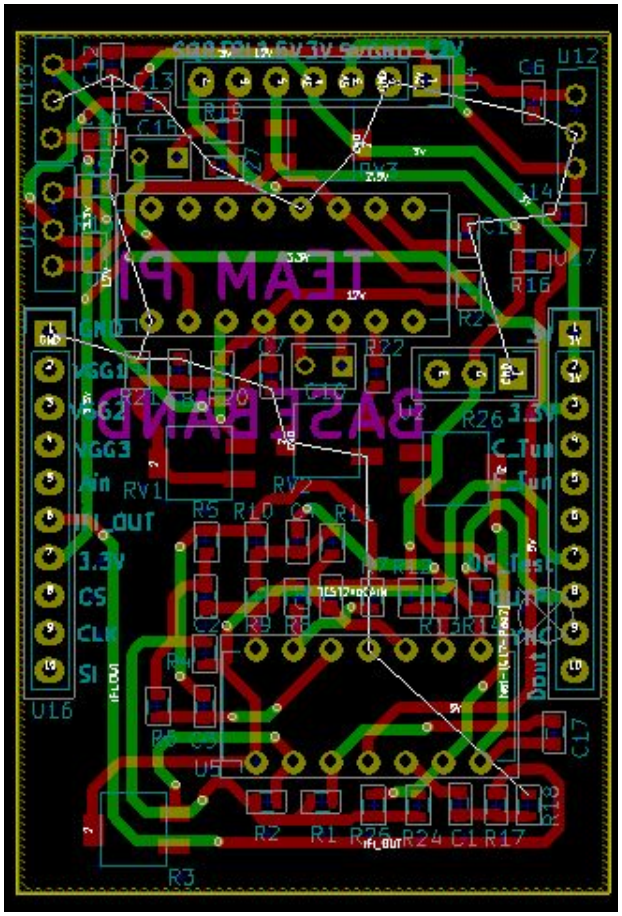

Fig. 13

Soldering was not difficult at all, just time consuming. Although we had quite a number of small and large components, certain steps can simplify the process.

1) Solder the smaller SMD components first.
2) Solder in small batches. In other words, solder 6 or 7 SMD components at a time if you are using a hot plate. Reasoning: Many times the PCB tips and all 30 or so SMD components will slide out of place. By securing them every so often, that becomes less of a problem.
3) Check for shorts using multimeter.
4) When doing through-hole components, solder DIPs instead of soldering your components directly. This allows you to replace components without de-soldering which gets very messy, is time consuming, keeps the board clean, and can save the component from damage (since it is easier to replace in the circuit).

Stacking PCB and Fixing Errors on Board

Sometimes mistakes in KiCad, or an actual design flaw can result in functional PCB which cannot work due to certain errors. It is possible to improvise and correct some of those errors. For example, if you forgot a connection between two components, you can use an actual wire to solder a connection directly on the PCB. We had an error in KiCad where a junction did not recognize a connection so we had to improvise a connection in such a way. In another example. I (Oleg) accidently included a pin twice, but named the pin differently the second time. This resulted in a misalignment with the pins from the RF board. To solve this, we forced the entire row of pins down one spot to line up again. For the top pin, which was aligned initially, we clipped the pin and used a wire to solder a connection.

Signal Processing

The code for Teensy 3.1 / 3.2 is provided. It includes the ADC with adjustable reference voltage and SD card read/write. We programmed this code to the Teensy using the TeensyDuino interface. We used the Teensy ADC library provided by pedvide on GitHub: www.github.com/pedvide/adc . The *analogread* example provided in this library was the basis for the analog-to-digital converter code. We modified the code to read and write the analog signals onto the SD card as well. We performed early tests on the ADC, using a breadboard, the Teensy 3.1, a function generator, oscilloscope, and TeensyDuino Serial Monitor. The first iteration of this Teensy ADC was a single ADC that displayed results to the serial monitor at the rate specified in the code. After studying the results of the sampling, we then modified the code for the Teensy to have two ADCs, with each providing separate readings. After this, we added

code that would access the SD card module through SPI connections and write the samples from the two ADCs to two separate .txt files. The connections are as follows:

Teensy 3.1 / 3.2 Pin:	Connects to:
GND	SD GND, Board GND
Pin 10	SD CS
Pin 11	SD MOSI
Pin 12	SD MISO
Pin 13	SD SCK
Pin 16	Analog Input 1
Pin 23	Analog Input 0
3.3V	SD 3v3

Since we made many modifications to the code and the connections to the Teensy after the PCB runs, we continued to use a breadboard for the Teensy's operation without making a new PCB.

Testing

RF Board

Prior to testing the board, we had to check whether or not the transceiver was properly soldered and that there was a solid connection between the pins of the BGT21MTR11 and the pads of the RF port. This is done by performing a continuity check using needle tipped probes, or test leads. It is vital that these specific probes are used to test for continuity because the pins and pads of the transceiver and the SMD components are very small. It is important to note that the continuity checks may produce a false alarm when checking the 50 Ohm terminations because the impedance seen by the multimeter is too low--the impedance has to be greater than ~100 Ohms in order to not trigger an alarm. After performing these checks, the next course of actions was to set the voltage to the 3.3V that the transceiver requires; this 3.3V requirement was based off of the data sheet of the transceiver. In order to test the board, the transceiver has to be initialized properly before it can begin transmitting RF signals due to the transceiver's TX outputs being disabled on startup, which is done by default. The transceiver was initialized through Teensy 3.2 via SPI. By looking at the transceiver's datasheet and "User's Guide" document, we saw that in order to initialize the transceiver correctly, we had to send a specific set of 16 bits from the Teensy 3.2 microcontroller to pin 20 of the Infineon BGT21MTR11 transceiver. The following 16 bits that had to be sent is "0b0000000001101000". Since we could not send 2 bytes, or 16 bits, of data at the same time, we decided to send the string in packets of 2. The first byte of data that was sent was called "HighByte" which had the following bits set to: 0b00000000. The second byte of data was called "LowByte" which had the following bits set to: 0b01101000. Each individual bits are set so that it turns on/off specific parts of the transceiver. For instance, with reference to "0b0000000001101000", the 12th bit in this string enables/disables the power output of the transceiver. Thus, in order to allow the transceiver to transmit signals through the TX port, we need to set the 12th bit to 0, or LOW.

After the transceiver is properly initialized, the next step that we took was to connect the IFI output to the input of the baseband amplifier circuit. The output of the baseband amplifier network was connected to a 3-lead microphone cord: the red lead goes to the output of the baseband amplifier network, the white lead goes to the square wave output from the Jameco, and the last wire goes to ground. The white lead connects to the square wave output of the Jameco because the square wave is used as a SYNC, which is needed to perform the signal processing for range measurements.

Baseband

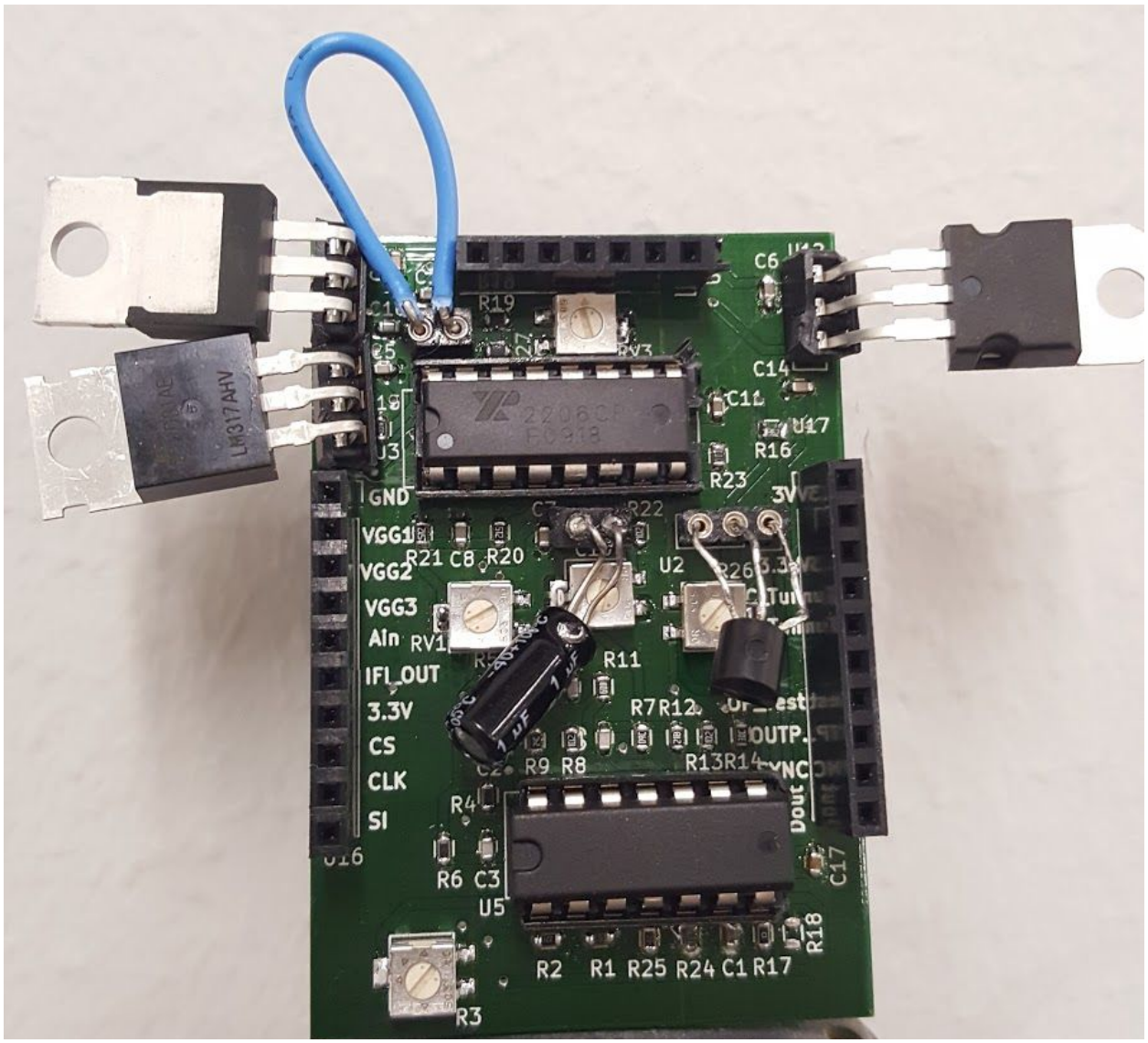

Fig. 14 - Baseband PCB

The voltage regulators performed as expected. We did not have an issue with them. It is noteworthy, that the 3V voltage regulator was extremely hot due to about 230 mA streaming through it to the transceiver (high power dissipation). We minimized the time in board was on to limit any damage that may occur as a result of the high temperature.

We had a strange issue with the LPF + Gain stage and the 2.5 V reference from the LT1009 component. At times the voltage would work as expected. We would see 2.5 V at the LT1009 which set the voltage at many points along the LPF + Gain Stage. Other times, the voltage at those points would spike to 5 V and supercede the 2.5 V reference voltage. It was hard to tell that the exact error was because changing the ICs did not fix the issue. The LT1009 and TL980 IC worked properly. The 5V is coming from the 5V bias of the TL980 IC. This is likely due to some error in the actual design which allowed the voltage to leak.

We did not end up using the 3 V since it was intended for a component we did not use. However, it is worth noting that using small potentiometers is best. For example, our potentiometers had a range from 20 Ohms to 2 Megaohms. This made getting 3 V extremely hard since one slight movement changes the voltage by ~0.3 V. Also, the resistance drifts slightly so the voltage would change by about 0.2 V overnight.

We were successful in getting the signal generator to work. We managed to get the triangular waveform with the correct bias, amplitude and frequency. Sometime the capacitor would act funny and the frequency we would see was in the MHz. To fix this, we simply shifted the capacitor.

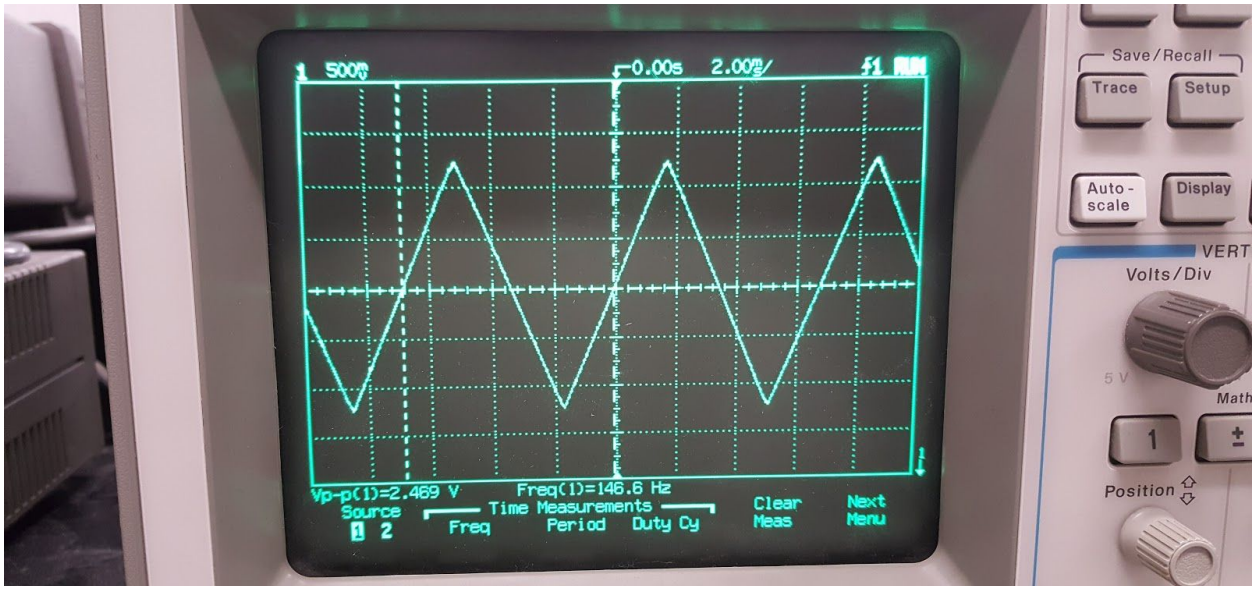

Fig. 15 - XR 2206 Triangular Waveform

Unfortunately we could not get this signal to work with the RF board. Upon connecting the boards together, the shape of the signal changes to sinusoidal and the amplitude falls a significant amount.

Initial Testing (RF Board)

While testing the RF board with the amplifier circuit, Chuong managed to get some measurements. The range is about 10 meters. On the image it shows that the range is greater, but that is incorrect due to the way that the code operates. Through several iterations, we found that the most optimum tuning frequency that provides the best result is a 150 Hz triangle wave with a V_{p-p} of 2V and a DC offset of 3V, see Fig. 16C. The V_{p-p} and DC offset makes sense since the antenna was designed to have a center frequency of 25 GHz, which meant that there is an acceptable amount of reflection at this point such that there is still power transfer to the antennas. During these tests, a function generator was used to create the tuning triangle waves instead of using the Jameco.

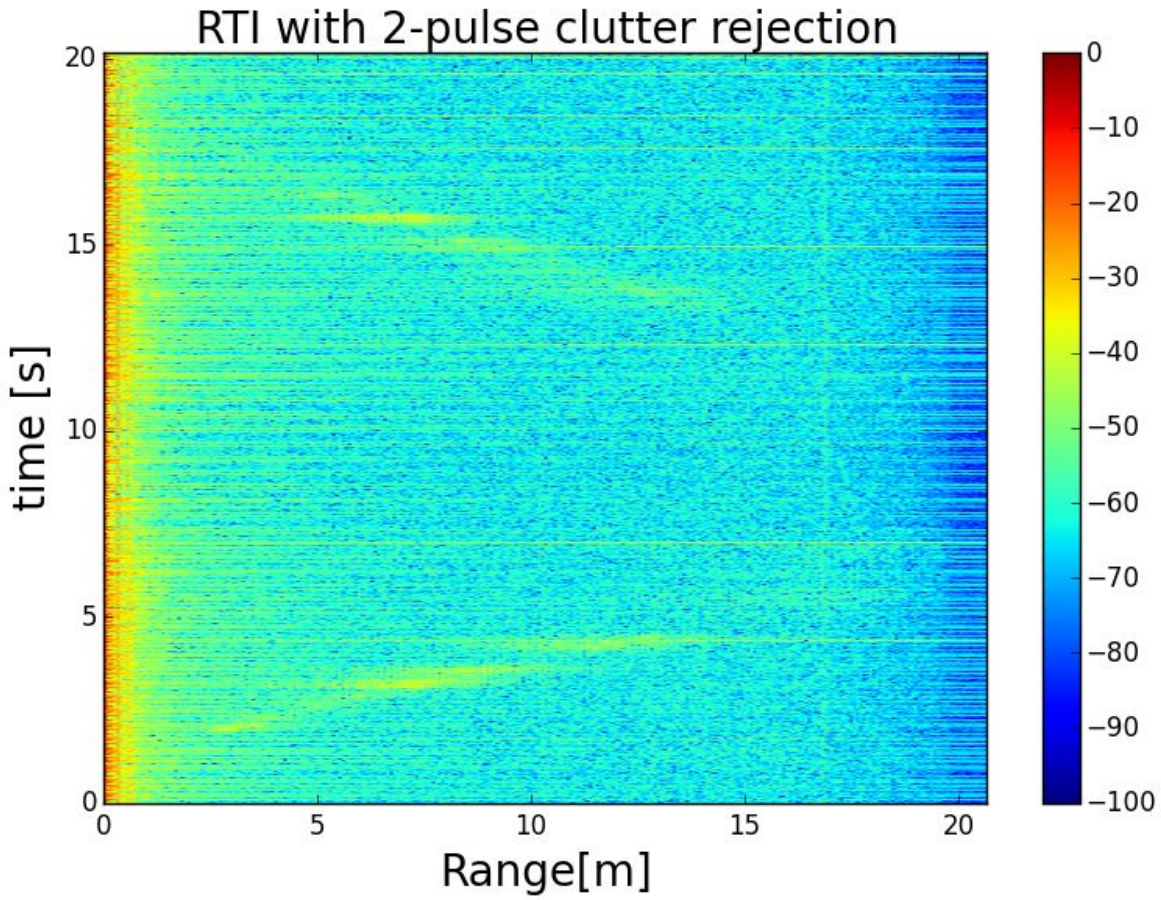

Figure 16A - Initial Testing Result

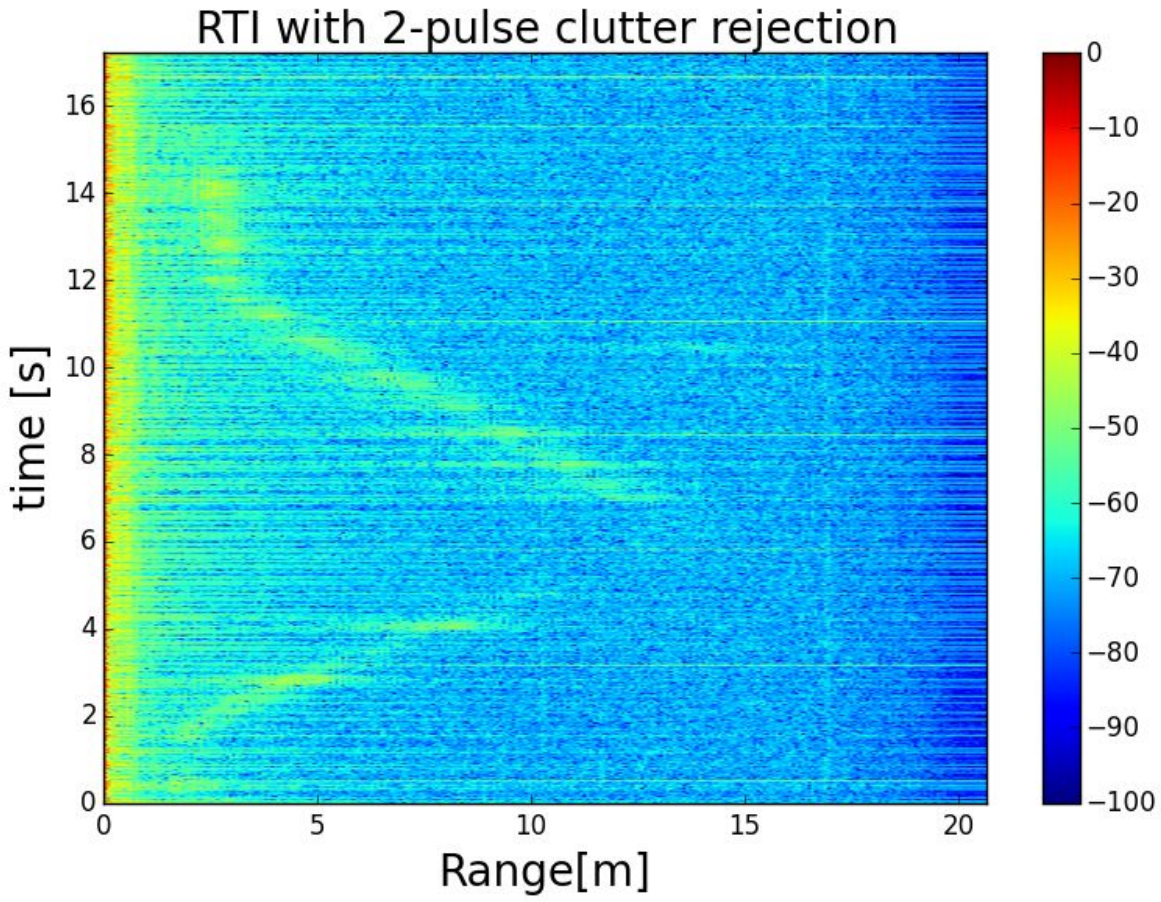

Figure 16B - Test 2 of the RF Board with a 100 Hz Triangle Wave

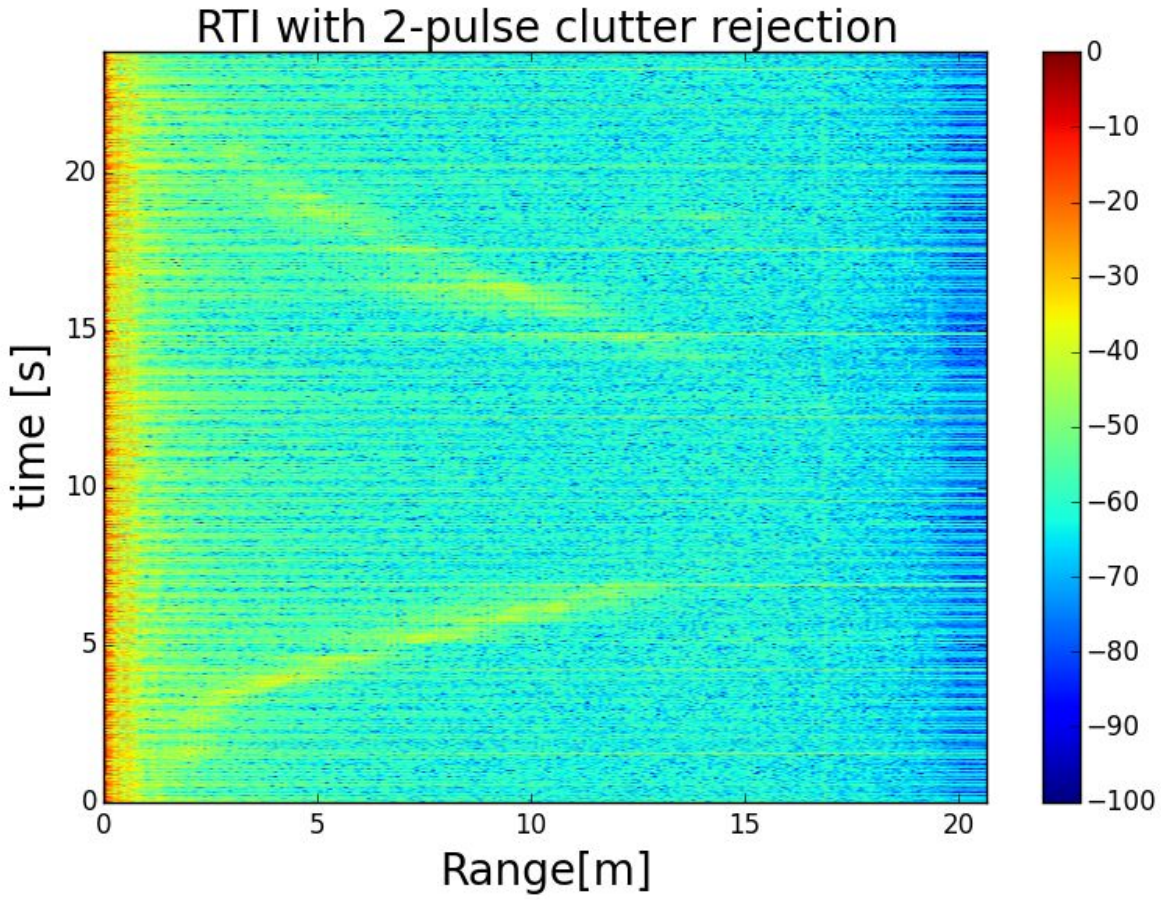

Fig. 16C - Test 3 of the RF Board with a 150 Hz Triangle Wave

Secondary Testing

Upon combining the baseband with the RF board, we could not replicate the same results. Initially we thought it could be due to the fact that the signal coming from the Jameco XR-2206 becomes sinusoidal when the baseband is combined with the RF board. However, when substituting the XR-2206 with an outside signal generator did not seem to fix the issue. Further testing and experimenting still could not help us figure out why. The RF board seemed to functioning as expected since it was drawing the required amount of current. The baseband by itself was also functioning as expected aside from some issues mentioned on page 20. We did not include the bottom board (teensy board), because of some pin issues and design flaws. Also, we did not end up saving data to a memory card as initially planned. We used Audacity on campus computers to process the data.

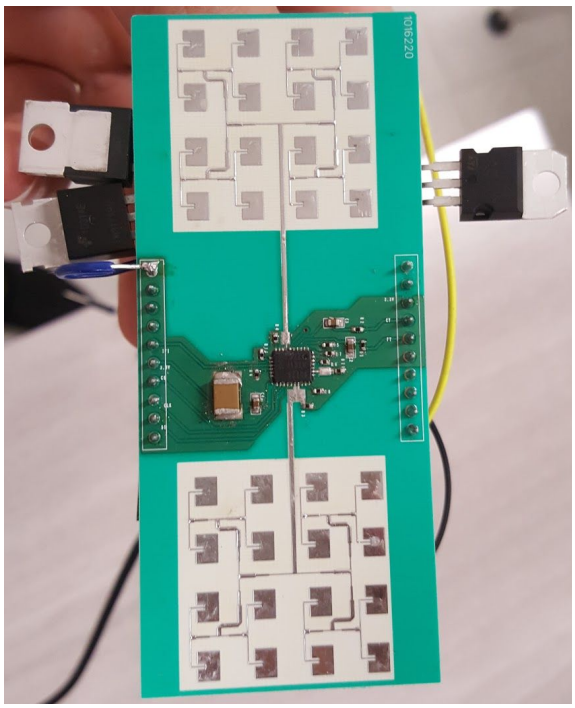

Fig. 17 - Top View

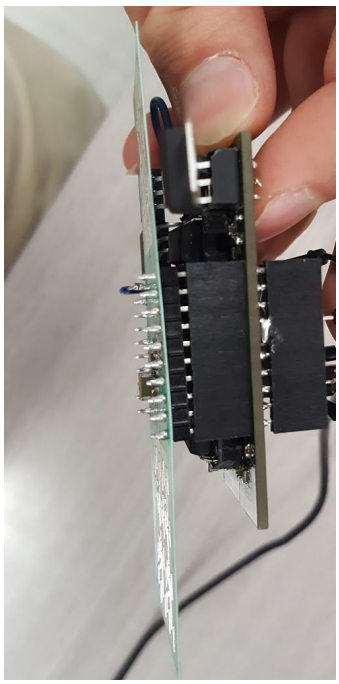

Fig. 18 - Side View

Conclusion

The design and implementation of this system was meant as a way to assist students in understanding the fundamental principles of a Frequency Modulated Continuous Wave (FMCW) Radar. The RF and baseband systems serve as the heart of a FMCW radar, with the RF containing the voltage controlled oscillator (VCO), power amplifiers (PA), antennas, low noise amplifiers (LNA), and mixers; while the baseband systems contained the necessary low frequency amplifiers and low pass filters, which is necessary to boost the IF signal from the mixer and to filter out unwanted harmonics. While testing the RF board and baseband system--by using a function generator signal to tune the VCO-- we found some results, which was shown in Fig. 16ABC. However, we ran into problems when we tried to to stack the system because the Jameco would fail to produce the triangle wave that was needed to tune the system. Overall, this project provided an invaluable experience that would be greatly beneficial in future work places. The project provided us with the opportunities to design PCBs, antennas, and system testing, which are valuable skills to possess. Special thanks to Professor Liu and Hao Wang for providing tips and assistance in the early stages of the project.

Suggestions for Future Students

Chuong's Suggestions:

1. If the students plan to work on the 24 GHz radar, then it is **extremely important** that the students have a good grasp of designing PCBs using Cadence Allegro and antenna design using HFSS because learning either one of these EDA tools from scratch is extremely time consuming.
2. If your group decides to build a microstrip patch antenna array, start as soon as possible. Tuning a single patch antenna is already time consuming, but factoring in an array using a corporate feed network will be many times more time consuming
3. Always have a team member check your PCBs before sending it to be manufactured. Small things could easily be overlooked.

Oleg's Suggestions:

1. If the PCB has a lot of components, having multiple test pins will help with debugging. For example, after each gain stage in the LPF, it would help to have a test pin. You might also like to have test pins at the voltage regulator outputs. This way you can measure the voltages and quickly spot errors. Also, align these pins together in rows at the edges of the PCB. This make it neater and easier to test. Use silkscreen in the PCB software to label pins.
2. Always have at least one team member and a friend outside the team review the schematic and PCB design. It is much harder to spot your own errors.
3. The lower the frequency the better. We made the mistake of using 24 GHz. If I could go back, I would have worked with a frequency under 6 GHz. Reasoning: Much cheaper components and not as sensitive as 24 GHz.

Ferris' Suggestions:

1. When the team is working remotely, it is imperative to meet up at least once or twice a week. Without weekly meetups, communication can dwindle and team members may work on overlapping parts.
2. When collecting data via microcontroller, make sure that the controller can take the desired samples per second. One problem we faced with using the Teensy to sample data was that the transfer of data through the MOSI/MISO pins of the SD card disrupted the operation of the program. The result of this was that we would observe a segment of a coherent wave, then suddenly the data would show a shift to another portion of the wave, with no apparent difference in frequency or amplitude. An example of this is shown below. After some consultation with students who have more experience in using

microcontrollers, it was suggested that this "skipping" may be caused by the microcontroller pausing the data-sampling task to perform the data-writing task. One method they suggested to alleviate the problem is to write the data samples into an array on the microcontroller, then when there is sufficient data to analyze, write the whole array to the SD card. That way, the microcontroller does not have to "skip" every time it writes a few data points. The skipping is also not periodic; it happens at various intervals and has different durations, so it is difficult to work with when analyzing data.

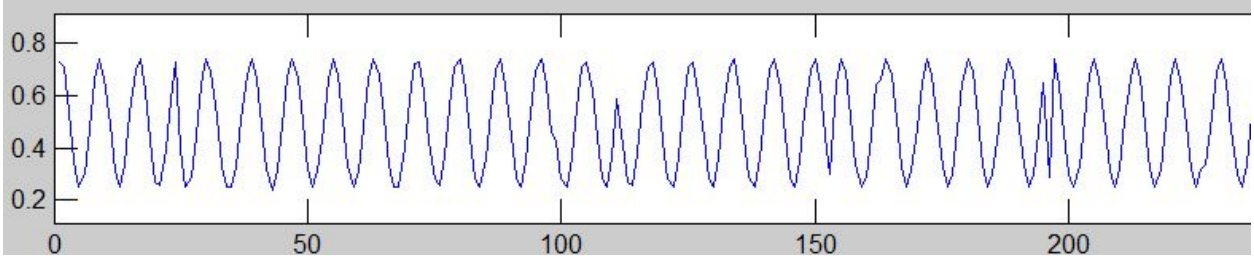

Fig. 19 - Taking samples from a non-changing (amplitude nor frequency) sinusoidal wave, but ending up with data that shows segments of the correct waveform, then "hiccups" happening at a non-periodic rate (they happen during the: 25th, 100th, 110th, 155th, 195th, and 225th samples on the graph).

Lhawang Thaye's Suggestions:

1. When coding on the Arduino software, using the Arduino library is very helpful especially for those that are new to coding. Here is a link to the reference page that is a library of functions, variables, and setup functions (structure) which will be used. https://www.arduino.cc/en/Reference/HomePage Just like Matlab documentation, particular functions that might come in handy have descriptions of the function, explainations of every parameter used by the function and examples of the said function used in a block of code to help understand how to use the function in a block of code.
2. When soldering the PCB by a soldering iron or a hot plate, always make sure that the soldering temperature is not above the recommended soldering temperature of the device. Always check the datasheet of the devices to find out the right temperature otherwise the device may not work afterwards due to the high temperature that it was exposed to. This is something is can be at times forgotten and should be a on the checklist before soldering.