Arizona State University

Frequency–Modulated Continuous–Wave Millimeter–Band Radar for Volcanic Ash Detection

Barrett Undergraduate Honors Thesis

Andrew Jinchi Xi **Matthew Byeongmok Lee**

School of Electrical, Computer, and Energy Engineering (ECEE)

Ira A. Fulton Schools of Engineering

and

Barrett, The Honors College

Directed by

Dr. Christopher Groppi[i]

Supervised by

Dr. Philip Mauskopf[i]

Dr. Douglas Cochran[ii,iii]

Dr. Alicia Baumann[ii]

Spring 2019

[i]School of Earth and Space Exploration (SESE), Ira A. Fulton Schools of Engineering
[ii]School of Electrical, Computer, and Energy Engineering (ECEE), Ira A. Fulton Schools of Engineering
[iii]School of Mathematical and Statistical Sciences, The College of Liberal Arts and Sciences (CLAS)

Abstract

The use of conventional weather radar in vulcanology leads to two problems: the radars often use wavelengths which are too long to detect the fine ash particles, and they cannot be field–adjusted to fit the wide variety of eruptions. Thus, to better study these geologic processes, a new radar must be developed that is easily reconfigurable to allow for flexibility and can operate at sufficiently short wavelengths.

This thesis investigates how to design a radar using a field–programmable gate array board to generate the radar signal, and process the returned signal to determine the distance and concentration of objects (in this case, ash). The purpose of using such a board lies in its reconfigurability—a design can (relatively easily) be adjusted, recompiled, and reuploaded to the hardware with none of the cost or time overhead required of a standard weather radar.

The design operates on the principle of frequency–modulated continuous–waves, in which the output signal frequency changes as a function of time. The difference in transmit and echo frequencies determines the distance of an object, while the magnitude of a particular difference frequency corresponds to concentration. Thus, by viewing a spectrum of frequency differences, one is able to see both the concentration and distances of ash from the radar.

The transmit signal data was created in MATLAB®, while the radar was designed with MATLAB® Simulink® using hardware IP blocks and implemented on the ROACH2 signal processing hardware, which utilizes a Xilinx® Virtex®–6 chip. The output is read from a computer linked to the hardware through Ethernet, using a Python™ script. Testing revealed minor flaws due to the usage of lower–grade components in the prototype. However, the functionality of the proposed radar design was proven, making this approach to radar a promising path for modern vulcanology.

Contents

1 Introduction

1.1 Background

1.1.1 Motivation

Volcanic eruptions occur globally 50 – 60 times per annum [1]. These eruptions can form large clouds of fine volcanic ash and water vapor. These ash distributions are crucial to understanding volcanic activity on earth, yet modern weather radar systems are unsuited for tracking them due to 2 reasons:

1. Since weather imaging systems operate on wavelengths meant to detect large cloud groups, the fine particulate matter spewing out of volcanoes is too small to be "visible" to a weather radar. In order to discern it, millimeter wavelengths and Extremely High Frequencies (EHF)* are required [2].
2. The large variability in eruption conditions means that vulcanologists need to be able to make adjustments to the radar on a case–by–case basis. The inflexibility of modern weather radars means that they cannot be modified without very large cost and time overheads.

The objective of this project was to develop a radar which uses sufficiently small wavelengths to detect and map volcanic ash distributions, which is also easily reconfigurable. The usage of a Field–Programmable Gate–Array (FPGA) board facilitates this. Since designs can be easily changed, compiled, and then uploaded to the board, the time and cost overheads are both removed.

1.1.2 Frequency–Modulated Continuous–Wave Radar

Frequency–Modulated Continuous–Wave (FMCW) is a radar scheme which uses a variable–frequency wave—a "chirp"—as the transmitted signal. Fig. 1 displays this signal characteristic, with the top trace being frequency and the bottom being magnitude with respect to time.

*Defined as 30 to 300 GHz

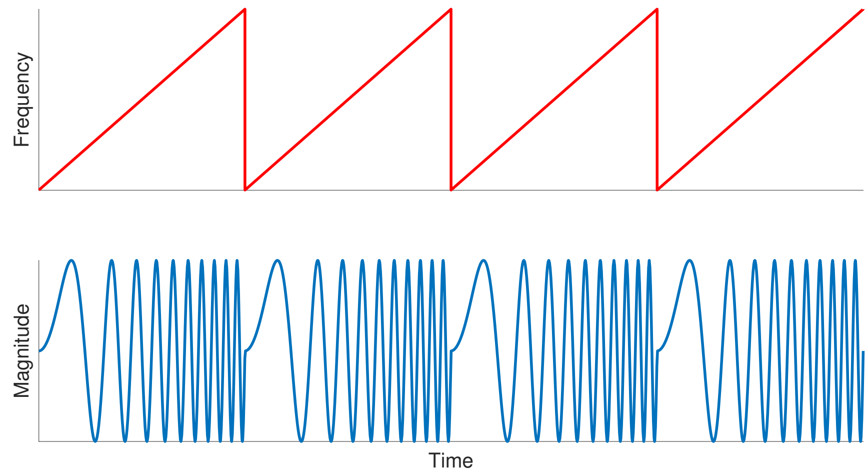


Figure 1: The “chirp” of a FMCW radar.

In this scheme, the chirp is sent out of the transmitter, reflects off a distant object, and returns with a time delay directly proportional to the distance of the object, seen in Fig. 2. This delay causes a difference in frequency between the outgoing and incoming signals. Measuring this difference allows the distance of the object to be ascertained.

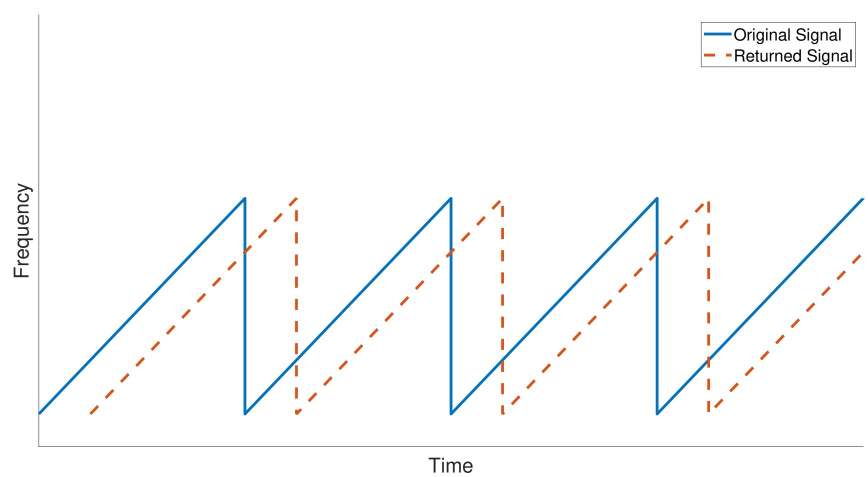


Figure 2: The returning “chirp” has a time delay and frequency difference directly proportional to the distance of the object

1.1.3 CASPER

This project used the Reconfigurable Open Architecture Computing Hardware 2 (ROACH2) Revision 2 FPGA board*, in addition to pre-written hardware IP blocks for MATLAB® Simulink®. These were designed by the Collaboration for Astronomy Signal Processing and Electronics Research, or CASPER, which is a UC Berkeley†–based research group that designs various astronomical instruments.

A photograph of the ROACH2 Rev. 2 FPGA board can be seen at Appendix A.1.

1.2 Mathematical Principles of FMCW

1.2.1 Assumptions

The following assumptions were made to simplify the approach for this project:

1. The object being measured (the ash distribution) is static and stationary. Therefore, the shift in frequency compared to a stationary object‡ is zero. The phase shift of the received signal is disregarded for the same reason.
2. Air has negligible effect on the speed of light. Therefore, c for our calculations is the same as *in vacuo*.
3. For a given maximum object distance, no reflective objects exist beyond that distance.

1.2.2 Derivation

The following equations and information are partially based on *Design and Implementation of an FMCW Radar Signal Processing Module for Automotive Applications* by Suleyman Suleymanov [3].

Several modulation schemes exist for FMCW, including the sawtooth wave, the triangle wave, and sinusoidal. The scheme used in this project is the sawtooth, shown in Fig. 1.

The frequency of the outgoing chirp is a periodic, linear function with respect to time. If the period of the chirp is T, the bandwidth§ of the signal is B and the starting frequency is f_c (160

*Contains Xilinx® Virtex®–6 FPGA chip

†University of California, Berkeley

‡This Δf is known as the Doppler effect, and is in fact the basis of Doppler radar

§Bandwidth in this context refers to the difference between the maximum and minimum frequencies in the chirp, i.e. $\max f_{\text{chirp}} - \min f_{\text{chirp}}$

MHz in our calculation), then the frequency with respect to time within a single chirp is:

$$f(t) = f_c + \frac{B}{T}t \qquad \text{where} \qquad 0 \le t \le T \tag{1.1}$$

The rate of change of $f(t)$, which is the slope of the chirp's frequency with respect to time, is called the "chirp rate", or α:

$$\alpha = \frac{B}{T} \tag{1.2}$$

Then, the signal of one chirp with amplitude A with respect to time can be defined as:

$$\begin{aligned} S_{\mathrm{Tx}}(t) &= A_1 \sin\Big(2\pi(f_c + \alpha t)t\Big) \\ &= A_1 \sin\Big((\omega_c + 2\pi\alpha t)t\Big) \qquad \text{where} \qquad 0 \le t \le T \end{aligned} \tag{1.3}$$

Since the signal is periodic, then any arbitrary time t can be expressed as the sum of an integer number of full periods plus the time from the start of the n^{th} chirp:

$$t = nT + t_s \qquad \text{where} \qquad 0 < t_s < T, \quad n \in \mathbb{Z} \tag{1.4}$$

Thus the output signal of the FMCW radar at any time t can be redefined in terms of t_s:

$$\begin{aligned} S_{\mathrm{Tx}}(t) &= A_1 \sin\Big((\omega_c + 2\pi\alpha t)t\Big) \\ &= A_1 \sin\Big((\omega_c + 2\pi\alpha(\cancel{nT} + t_s))(\cancel{nT} + t_s)\Big) \\ &= A_1 \sin\Big((\omega_c + 2\pi\alpha t_s)t_s\Big) \qquad \text{where} \qquad t_s = t \bmod T \end{aligned} \tag{1.5}$$

The frequency of the outgoing signal then is:

$$\omega_{\mathrm{Tx}} = \omega_c + 2\pi\alpha t_s \tag{1.6}$$

Now, let an object to be measured be located at a distance of D meters from the radar. The time for a signal to reach the object is therefore:

$$\tau_{\mathrm{Tx}} = \frac{D}{c} \tag{1.7}$$

Since the time for the signal to return is equivalent, then total time delay from transmission to reception is:

$$\tau = \frac{2D}{c} \tag{1.8}$$

The period T is necessarily at least long enough that a signal will be able to reach an object at the maximum distance $D_{\max}$ and return within 1 chirp:

$$\begin{aligned} \tau_{\max} &= \frac{2D_{\max}}{c} \\ T &\geq \tau_{\max} \end{aligned} \tag{1.9}$$

Due to incomplete reflection of the signal, the return signal will have a smaller amplitude A_2 than the outgoing amplitude A_1. Hence, the return signal can be expressed as:

$$S_{\mathrm{Rx}}(t) = A_2 \sin\Big((\omega_c + 2\pi\alpha(t_s - \tau))(t_s - \tau)\Big) \qquad \text{where} \qquad t_s = t \bmod T \tag{1.10}$$

The frequency of this return signal is:

$$\omega_{\mathrm{Rx}} = \omega_c + 2\pi\alpha(t_s - \tau) \tag{1.11}$$

The return signal is mixed by a frequency mixer with the outgoing signal:

$$S_m(t) = S_{\mathrm{Tx}}(t) S_{\mathrm{Rx}}(t) \tag{1.12}$$

Since this is a multiplication of sine waves with frequencies ω_{Tx} and ω_{Rx} (Eqs. 1.6, 1.11), then by the sine product identity:

$$\sin(\omega_{\mathrm{Tx}} t_s) \sin(\omega_{\mathrm{Rx}} t_s) = \frac{\sin\big((\omega_{\mathrm{Tx}} + \omega_{\mathrm{Rx}})t_s\big) + \sin\big((\omega_{\mathrm{Tx}} - \omega_{\mathrm{Rx}})t_s\big)}{2} \tag{1.13}$$

Thus, the mixed signal will contain both the sum and difference in frequency of the transmitted and received signals. The lower of the two, $\omega_{\mathrm{Tx}} - \omega_{\mathrm{Rx}}$, is also called the "beat" frequency. This beat frequency then is directly proportional to the distance of the object:

$$\begin{aligned} \omega_{\mathrm{Tx}} - \omega_{\mathrm{Rx}} &= (\omega_c + 2\pi\alpha t_s) - (\omega_c + 2\pi\alpha(t_s - \tau)) \\ &= 2\pi\alpha\tau \\ &= 2\pi\alpha\frac{2D}{c} \end{aligned} \tag{1.14}$$

$$f_{\mathrm{Tx}} - f_{\mathrm{Rx}} = \frac{\omega_{\mathrm{Tx}} - \omega_{\mathrm{Rx}}}{2\pi} = \alpha\frac{2D}{c} \tag{1.15}$$

Conversely, the distance of the object can be calculated by rearranging Eq. 1.15 in terms of the frequency difference, Δf, which can be isolated from the frequency sum using single-sideband mixers:

$$D = \Delta f \frac{c}{2\alpha} \qquad \text{where} \qquad \Delta f = f_{\mathrm{Tx}} - f_{\mathrm{Rx}} \tag{1.16}$$

The radar then samples the incoming mixed signal and takes an FFT*. If the FFT has n bins†, and the ROACH2, clocked at clk Hz, has a bandwidth of BW Hz, then each bin will represent a frequency range f_{bin}:

$$BW = \frac{clk}{2} \quad {}^{\ddagger} \tag{1.17}$$

$$f_{\mathrm{bin}} = \frac{BW}{n} \tag{1.18}$$

Recalling Eq. 1.16, the distance d represented by each bin is:

$$\begin{aligned} d &= f_{\mathrm{bin}} \frac{c}{2\alpha} \\ &= \frac{BW \times c}{2\alpha n} \end{aligned} \tag{1.19}$$

Thus, the radar is capable of a "distance resolution" of d meters.

*Fast Fourier Transform, an algorithm implementing the discrete Fourier Transform
†Distinct frequency groupings, akin to bars on a histogram
‡Shannon–Nyquist theorem

2 Design

2.1 Parameters

The initial specifications given for this design are as follows:

- Minimum object distance: 5 km
- Maximum object distance: 50 km
- Approximate distance resolution: 10 m
- Frequency multiplier*: 16

Using these conditions, the design parameters were obtained by iterating the chirp bandwidth, BW_{chirp}, and FFT size, n_{bins} to obtain a suitable distance resolution. The parameters are presented below:

1. The ROACH will be clocked at 200 MHz.

$$\text{clk}_{\text{ROACH}} = 200\,\text{MHz} \tag{2.1}$$

2. The maximum time delay between the transmission and reception of the chirp is:

$$d_{\text{far}} = 50\text{km} \tag{2.2}$$

$$\Delta t = \frac{2d_{\text{far}}}{c} \approx 333\text{µs} \tag{2.3}$$

The period of the chirp will be about double this time delay to account for synchronization (see Section 2.3). It will also be a power–of–2 number of clock cycles:

$$T_{\text{chirp}} = 2^{17}\frac{1}{\text{clk}_{\text{ROACH}}} \approx 655\text{µs} \tag{2.4}$$

3. Choosing pre-multiplication chirp bandwidth of 1.5 MHz allows us to calculate the multiplied bandwidth:

$$BW_{\text{chirp}} = 16 \times BW_{\text{pre-mult}} = 24\,\text{MHz} \tag{2.5}$$

Then, the chirp rate is:

$$\alpha = \frac{BW_{\text{chirp}}}{T_{\text{chirp}}} = 36.621\frac{\text{GHz}}{\text{s}} \tag{2.6}$$

*This multiplication is achieved through the use of four frequency doublers in series

4. The chirp will first be output by the ROACH2 with a $\min(f_{\text{roach}})$ of 10 MHz. This means that the $\max(f_{\text{roach}})$ will be 11.5 MHz.

5. The ROACH2 output will be fed into the Radio Frequency (RF) input of a single–sideband (upper) frequency mixer, known as an upconverter, with a Local Oscillator (LO) input of 13.75 GHz. Then, the Intermediate Frequency (IF) output will start at:

$$f_{\text{mixed}} = f_{\text{LO}} + f_{\text{RF}} = 13.76\,\text{GHz} \tag{2.7}$$

The output upper frequency will be 1.5 MHz higher, or 13.7615 GHz. After frequency multiplication, the chirp frequencies will begin at:

$$f_{\text{chirp}} = 16 f_{\text{mixed}} = 220.16\,\text{GHz} \tag{2.8}$$

They will end at 24 MHz above this value, or 220.184 GHz, as per Eq. 2.5. This frequency was chosen because it maximizes the signal return for the typical ash particle size in volcanic plumes [2]. It corresponds to a wavelength of about:

$$\lambda_{\text{chirp}} = \frac{c}{f_{\text{chirp}}} \approx 1.362\,\text{mm} \tag{2.9}$$

Thus the signal is operating within the EHF band and millimeter–wave region.

6. Based on Eq. 2.2 – 2.3 the greatest possible frequency difference is:

$$\max \Delta f = \Delta t \times \alpha \approx 12.207\,\text{MHz} \tag{2.10}$$

According to the Shannon–Nyquist sampling theorem, the spectrometer has a bandwidth that is half the clocking frequency of the ROACH. This bandwidth must be at least the highest frequency in the signal, or $max\Delta f$:

$$BW_{\text{Rx}} = \frac{\text{clk}_{\text{Rx}}}{2} = 100\,\text{MHz} \tag{2.11}$$

7. The number of bins in the FFT must be a power of two. Note that half of the spectrum will be negative frequencies:

$$\begin{aligned} n_{\text{bins}} &= 2^{16} = 65536 \\ n_{\text{bins}^+} &= \frac{n_{\text{bins}}}{2} = 32768 \end{aligned} \tag{2.12}$$

Hence the frequency range of each bin is:

$$f_{\text{bin}} = \frac{BW_{\text{Rx}}}{n_{\text{bins}^+}} = 3.052\,\text{kHz} \tag{2.13}$$

8. Following Eq. 1.19, the distance each bin represents is:

$$d_{\mathrm{bin}} = \frac{BW_{\mathrm{Rx}} c}{2\alpha n_{\mathrm{bins}}} = 12.500\,\mathrm{m} \tag{2.14}$$

This is sufficiently close to the desired 10 m distance resolution specification.

The wave vector used for the chirp was then generated in MATLAB® using the script in Appendix B.1.

2.2 Components

The system consists of a computer, a ROACH2 FPGA board, and other hardware, in addition to lab instruments.

2.2.1 Computer

- Ethernet port (for connecting to ROACH2)
- MATLAB® 2012a (for computing vectors) with Simulink® (for generating designs)
- `casperfpga`* Python™ library (for communicating with ROACH2)

2.2.2 FPGA

- CASPER ROACH2 Revision 2 (FPGA board)
- Techne Instruments MKID† DAC/ADC (music board)

2.2.3 Other Components

- Mini–Circuits® ZFSC–2–2500–S+ (2–way power splitter)
- Mini–Circuits® ZLW–6+ (double–sideband frequency mixer)
- Mini–Circuits® 15542 SMA Fixed Attenuator
- Single–sideband frequency mixer (upconverter)‡
- Frequency doublers‡

*`https://github.com/ska-sa/casperfpga`
†Microwave kinetic inductance detector
‡Not used in testing

- Various SMA* and BNC† cables

2.2.4 Lab Instruments

- SIGLENT® SDG6032X (signal generator)
- Agilent 33220A (signal generator)
- Rigol DSA 815 (spectrum analyzer)
- Tektronix® TDS7104 (oscilloscope)

2.3 Synchronization

The design of the radar presents a problem in that at certain points during operation, the difference in the outgoing and incoming frequencies jumps to a large negative value, seen in Fig. 3.

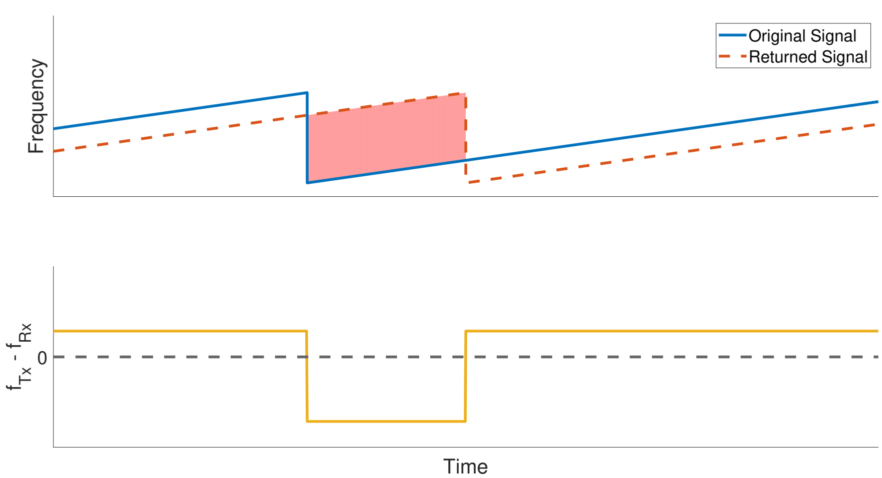


Figure 3: At certain times during the chirp (top, shaded), the frequency difference $f_{\mathrm{TX}} - f_{\mathrm{RX}}$ (bottom) will jump from a relatively small positive value to a relatively large negative value.

The large discontinuity in frequencies presents two issues:

1. It can lead to faulty data during the transition to the negative value.

*SubMiniature Version A
†Bayonet Neil–Concelman

2. The negative frequency has a different proportional relationship to the object's distance, and will appear in the spectrum as an extra, "phantom" object with a different distance than in reality.

To avoid these issues, the radar had to have a system for synchronizing its FFT to only read data when the difference was positive. The solution used in this design was to make the length of the chirp sufficiently long that the second half of the chirp could always be expected to have a positive frequency difference (Fig. 4).

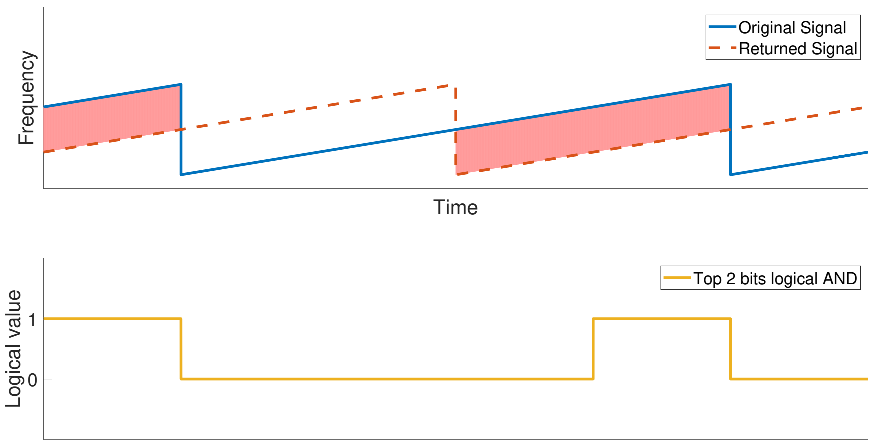


Figure 4: The period of the chirp is double the maximum expected delay. This means the second half of each outgoing chirp will contain a positive frequency difference (shaded). Thus the top two bits of the counter are high only during the positive frequency difference region.

By making the period of the chirp double the length of the maximum time delay, the design is able to very easily discern when the outgoing chirp is currently in the shaded region. Since the design uses a binary counter as a data pointer for reading out the ROACH2 output vector, taking an FFT when the top two bits are 1 allows the design to only take an FFT when outputting the last quarter of the wave, when the frequency difference is guaranteed to be positive.

The design methodology for the synchronization scheme constitutes an original scheme for accounting for the portions of data which may result in phantom objects.

2.4 Block Diagrams

2.4.1 Final ROACH2 Design

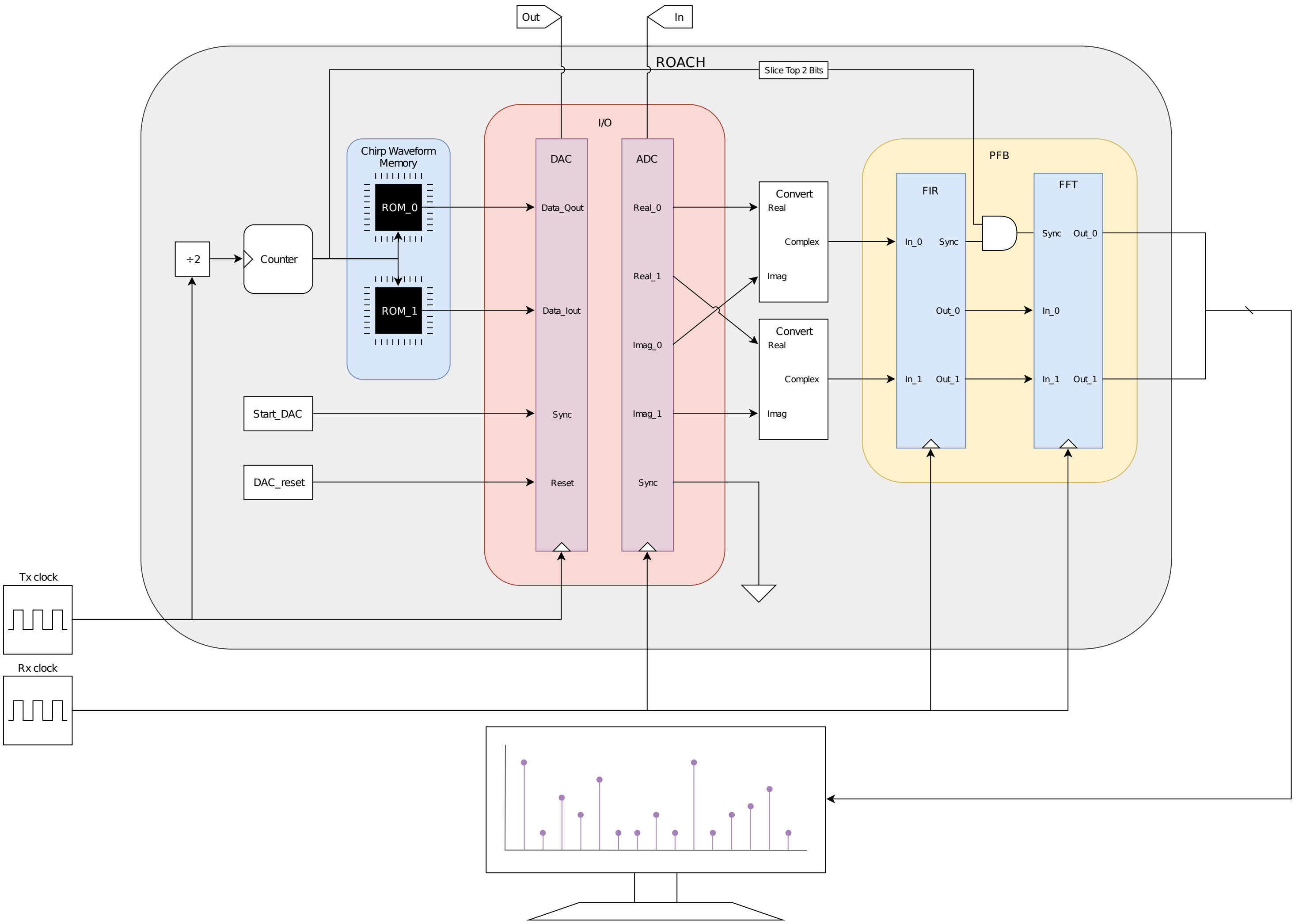


Figure 5: Block diagram of ROACH2 design for FMCW.

Fig. 5 displays the block diagram of the ROACH2 design. The transmit and receive portions of the radar have different clock frequencies. The transmit clock is first linked to a circular counter (i.e. the count loops back to 0 upon reaching the maximum count). This counter is connected as the read–address of the two ROMs, which hold the even and odd elements of the chirp vector. The ROMs output the current chirp magnitude to the DAC, which reads them out sequentially.

The incoming frequency–difference wave is input to the ADC, which converts the analog data to two real and two imaginary values. These values are then concatenated into complex numbers, which are placed in a Finite Impulse Response (FIR) filter before being sent to the FFT when the two most significant bits of the transmit clock are high (and thus the frequency difference is positive)*. The FFT block uses the butterfly scheme to transform the data from the time domain to the frequency domain. Finally, the output of the FFT is sent through Ethernet to a computer, which displays the data.

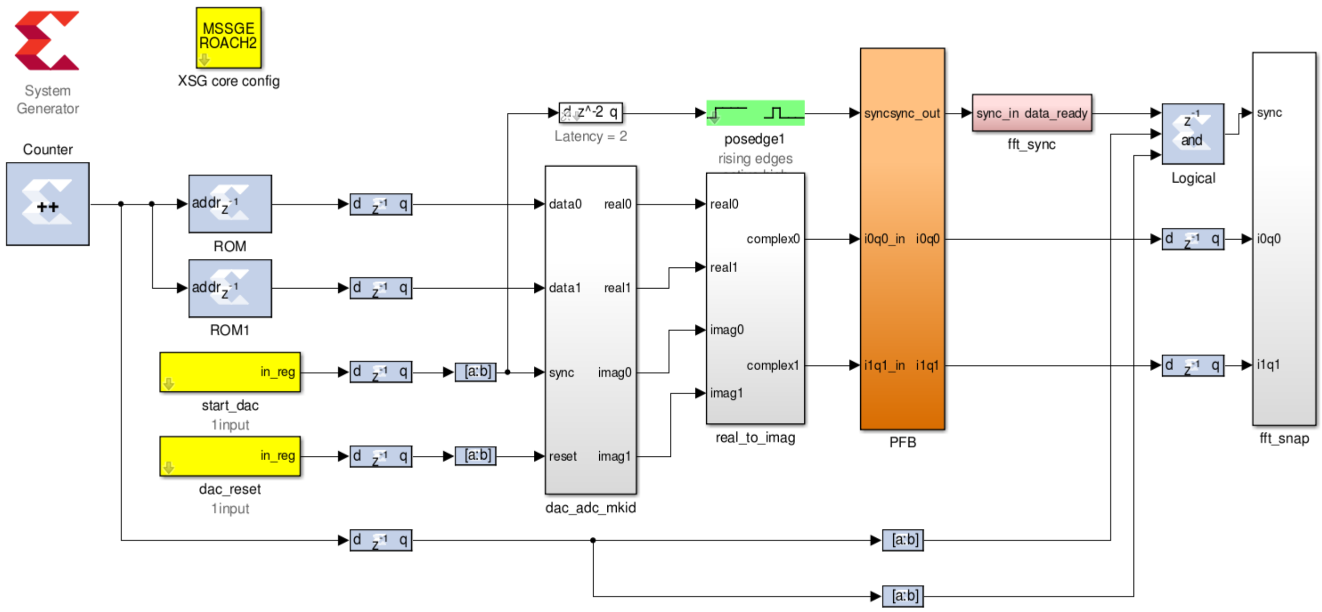


Figure 6: Simulink block diagram of FMCW.

Fig. 6 shows the Simulink® block diagram implementing this design in Fig. 5.

*This is the synchronization discussed in Section 2.3

2.4.2 FMCW Design

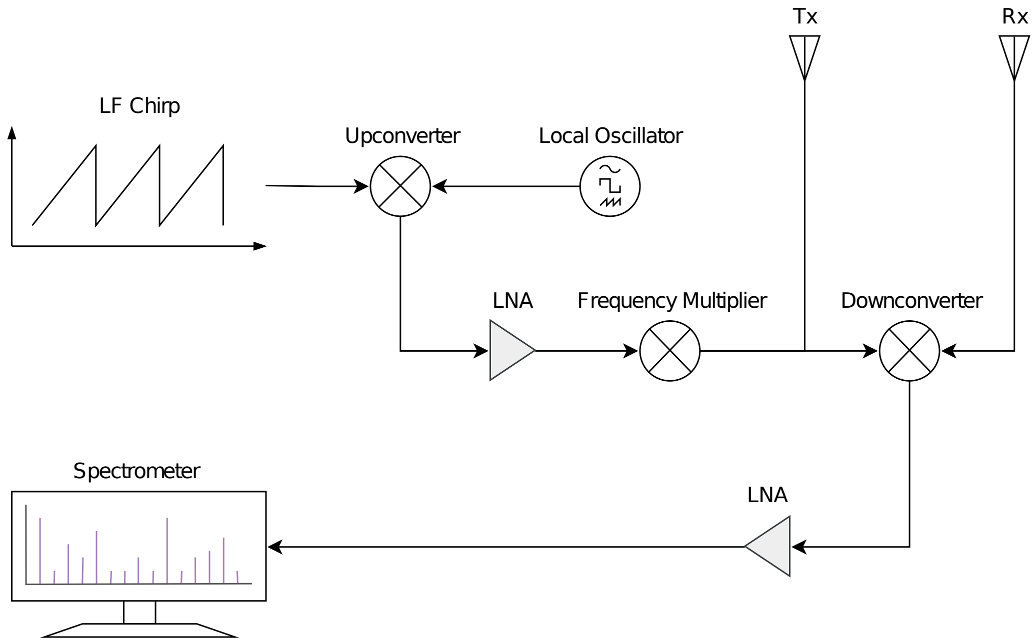


Figure 7: Block diagram of FMCW radar using ROACH2.

The off-board portion of the FMCW design is displayed in Fig. 7. First, the outgoing chirp (LF Chirp) from the DAC is sent through a single–sideband frequency mixer (upconverter), which outputs the sum of the RF and LO frequencies. Since the LO frequency will be a separately generated 13.75 GHz, and the ROACH2 output wave has a frequency ranging from 10 – 11.5 MHz, the mixer's output will be 13.76 – 13.7615 GHz.

This upconverted wave is then amplified by a low–noise amplifier and fed into a frequency multiplier, which outputs a harmonic of its input (in this case, 16 times the original frequency), resulting in a frequency of 220.16 – 220.184 GHz. The harmonic multiplication strongly attenuates the signal, which is corrected for by the pre-amplification. The multiplied chirp is then connected to the transmitting antenna as well as the LO input of the second single–sideband mixer (downconverter).

The received (and delayed) chirp is routed from the receiver antenna to the RF input of the downconverter. The IF output of the downconverter contains the difference in frequencies of the LO and RF signals, which is amplified and sent back to the spectrometer to process.

3 Testing

3.1 Overview

The testing was done in 3 major phases. First, the transmitter was tested to see if it correctly outputted the chirp waveform. The next step was to check if the spectrometer was able to properly plot the FFT, as well as if the spectrometer displayed the correct spectrum. Then, the entire design was tested to see if the combined transmitter and spectrometer functioned as a radar. For the purposes of testing, up–conversion and frequency multiplication of the ROACH2 output is unnecessary due to the fact only the outputted 10 – 11.5 MHz wave needed to be checked (frequency mixers and multipliers are discrete analog components with well–defined behaviors) and thus this step was omitted.

3.2 Setup

For these tests, the music board was attached to the ROACH2. A waveform generator operating at 200 MHz was first put through the 2–way power splitter, with the two ends connected to the DAC Clock In and the ADC Clock In ports. The compiled bitstream* of the design was uploaded through iPython using `upload.py`, found in Appendix B.2.

3.2.1 Transmitter

A male SMA to male BNC cable connected the DataQout port of the DAC to the input of the oscilloscope.

*File extension `.fpg`

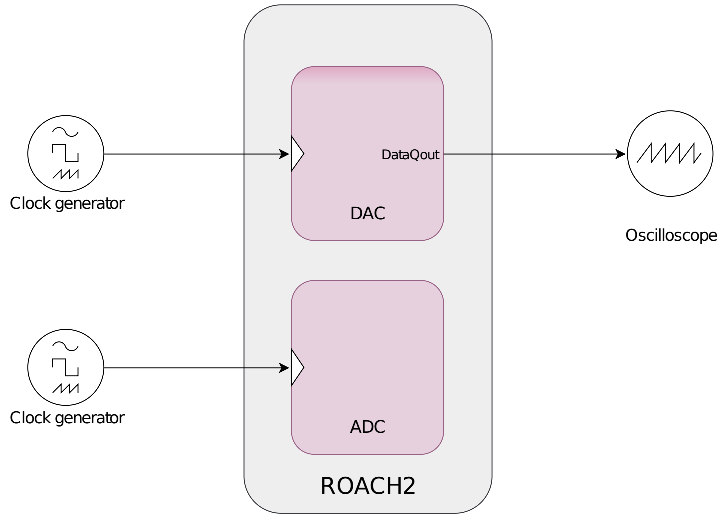


Figure 8: Setup diagram for testing transmitter.

The ADC and DAC are each connected to a clock signal generator, as in Fig. 8, and the output of the DAC is connected to a high–speed oscilloscope.

This test's purpose is to demonstrate that the ROACH2 is correctly outputting the initial wave which moves from 10 to 11.5 MHz. This shows that the transmitter is functioning properly.

3.2.2 Spectrometer

A plot of the FFT was generated in Python™ by calling the script `spec_plot.py`* found in Appendix B.3 which includes the code to upload the compiled bitstream.

*This script as well as the rest of the design can be found on GitHub at `powerfulmandrew.github.io/FMCW_Volcano_Radar`

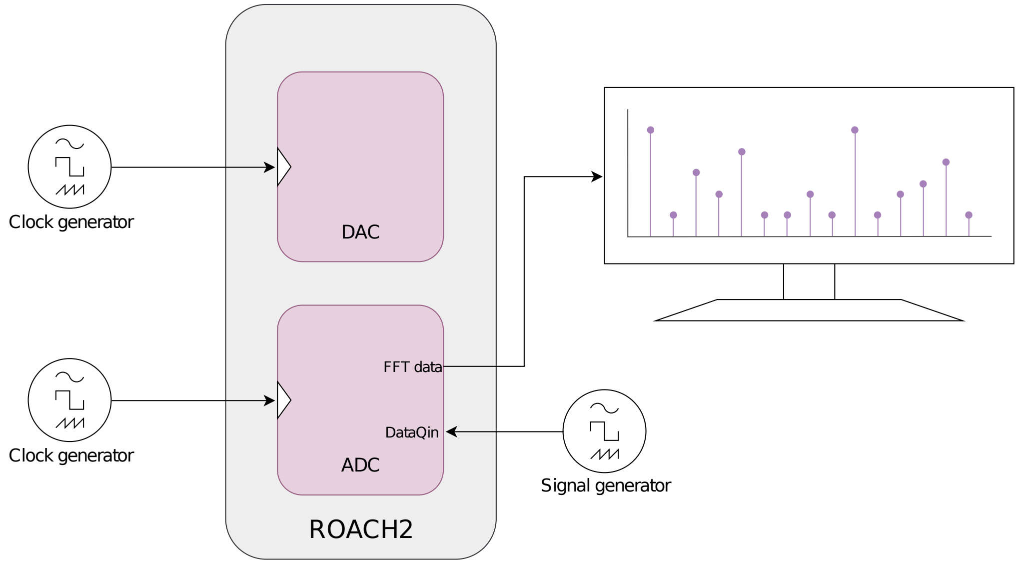


Figure 9: Setup diagram for testing spectrometer.

Fig. 9 shows the testing setup. The ADC and DAC were connected to 200 MHz clock signals, and a third signal generator generating a sine wave at various frequencies from 1 kHz – 50 MHz was fed into the DataQin port of the ADC. This frequency was adjusted while checking the output graph on the Ethernet–linked computer. The test was repeated with a 5 MHz square wave.

This test demonstrates that the spectrometer portion of the radar is able to accurately plot the frequencies it receives (which in the full radar is the frequency difference of outgoing and incoming waves).

3.2.3 Radar

1. The DataQout port was connected using a male SMA to male SMA cable to the 2–way power splitter. Both ends were then connected to a frequency mixer in the LO and RF inputs. The IF output was connected to the DataQin port, and the Python™ code for generating the spectrometer plot was run.

   This test demonstrates the removal of signals from an extremely close objects (zero distance), such as if the transmitting and receiving antennas are placed next to each other.

2. The LO input of the frequency mixer was connected to a 11 MHz sine wave, while the RF input was connected to the ROACH2 output. The IF output was then connected to the DataQin port of the ADC, shown in Fig. 10:

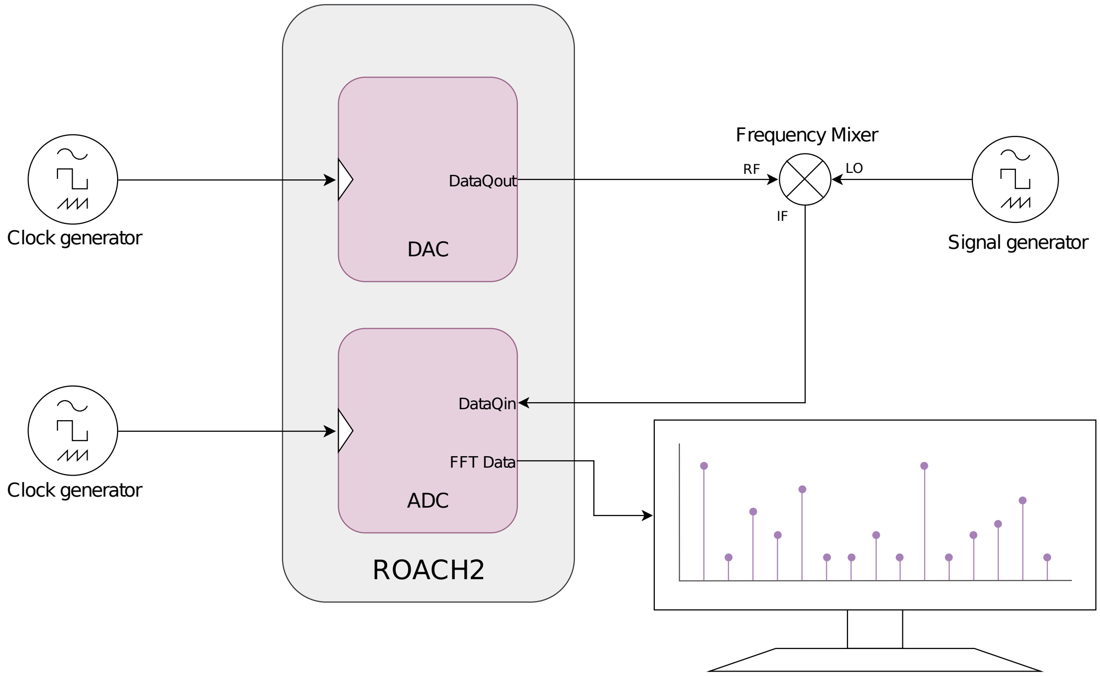


Figure 10: Setup diagram for testing synchronization.

Since the ROACH2 output ramps from 10 – 11.5 MHz, the IF output should output a frequency ramp from -1 to 0.5 MHz. This test demonstrates whether the synchronization portion of the design functions correctly. Functional synchronization should result in the FFT plot only containing a small band of frequencies from slightly above DC to 0.5 MHz, but should not contain anything above that.

A photograph of the synchronization testing setup can be seen at Appendix A.2.

4 Results

4.1 Transmitter

The transmitter was the first part of the radar built, and as such, the first part tested. The transmitter's wave on the oscilloscope was demonstrated to have an approximately sinusoidal form which fluctuated from 10 – 11.5 MHz (the pre-multiplication frequencies of the chirp).

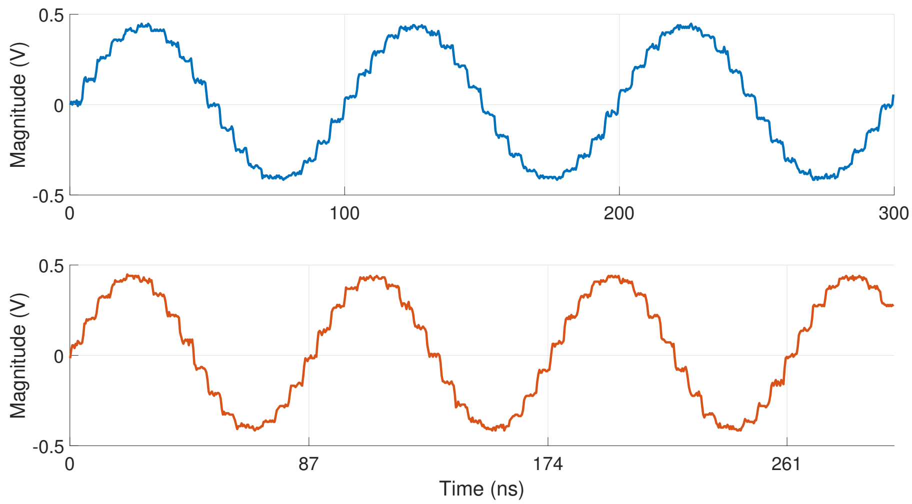


Figure 11: Oscilloscope waveform of chirp output, displaying varying frequencies near 10 and near 11.5 MHz.

As seen in Fig. 11, the output is a sinusoidal wave with stepped magnitudes. This quantization error is due to the relatively low sampling rate compared to the operating frequency, and results in the presence of other distorting frequencies in our signal spectrum.

The signal can also be compared to a perfect sine wave (Fig. 12):

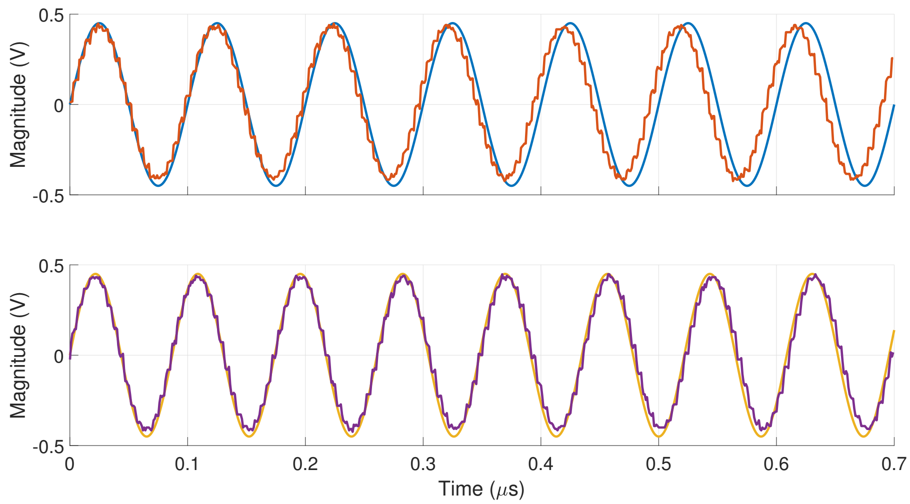


Figure 12: Oscilloscope waveform of chirp output compared to ideal sine wave at (top) 10 MHz and (bottom) 11.5 MHz.

Thus it can be concluded that the 10 – 11.5 MHz pre-multiplication chirp output of the ROACH2 is functioning correctly.

4.2 Spectrometer

The spectrometer was tested with various input frequencies and waveforms and displayed the following graph for the sine wave:

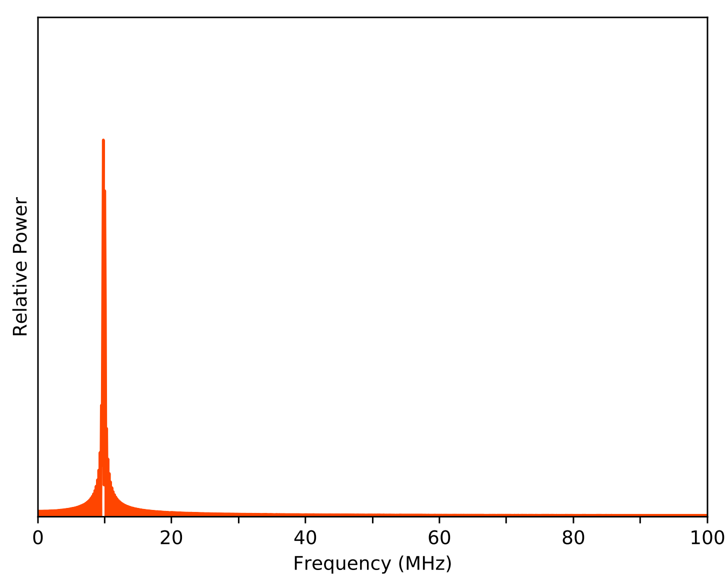


Figure 13: Spectrometer output for pure 10 MHz sine wave injection.

The spectrum of the sine wave (Fig. 13) is clean and accurate. Although frequencies outside of the exact 10 MHz are present next to the main peak (from 8 MHz to 12 MHz), this is due to the sampling length. The longer a signal is sampled, the more precise the spectrum will be* [5].

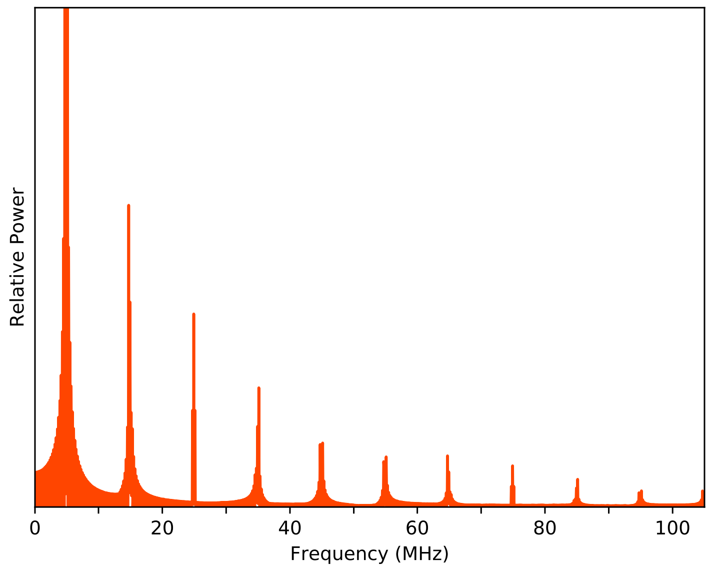


Figure 14: Spectrometer output for 5 MHz square wave injection.

The spectrum of the square wave (Fig. 14) is also correct, with the same uncertainty due to sampling length. An ideal square wave spectrum has peaks only at odd harmonics of the base frequency (i.e. 15 MHz, 25 MHz, 35 MHz, etc.), while the even harmonic peaks (i.e. 10 MHz, 20 MHz, 30 MHz, etc.) have zero magnitude [4].

4.3 Radar

1. The Python™ plot demonstrated strong attenuation of all harmonics of the chirp (though some leakage was still present) and removal of the near–DC region (Fig. 15):

*This is also the cause of the Heisenberg Uncertainty Principle

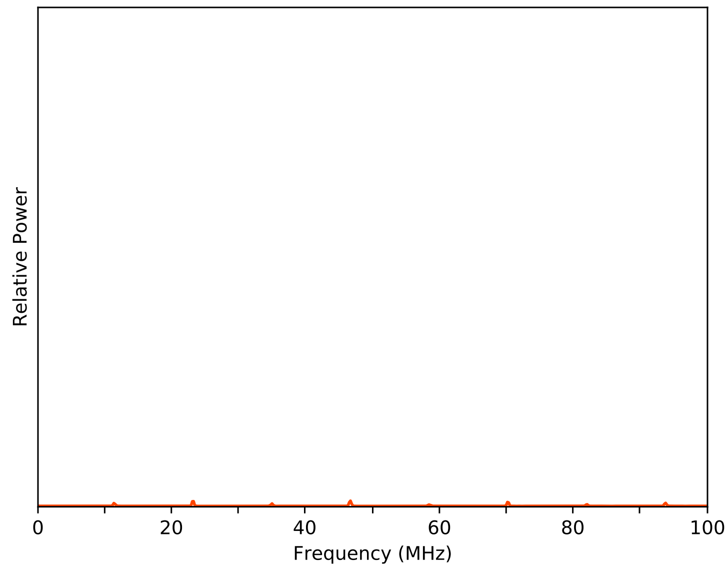


Figure 15: Spectrometer output for testing removal of near–DC signals.

The IF mixer in this case generates a DC bias due to the frequency difference being 0, which is then removed by the radar. This allows better signal quality as it helps alleviate the issue of a "drowned out" signal when the transmitter and receiver antennas are in close proximity.

2. The spectrometer plot of the chirp without frequency mixing shows some noise in the region near the central peaks, but overall has a clean spectrum, shown in Fig. 16.

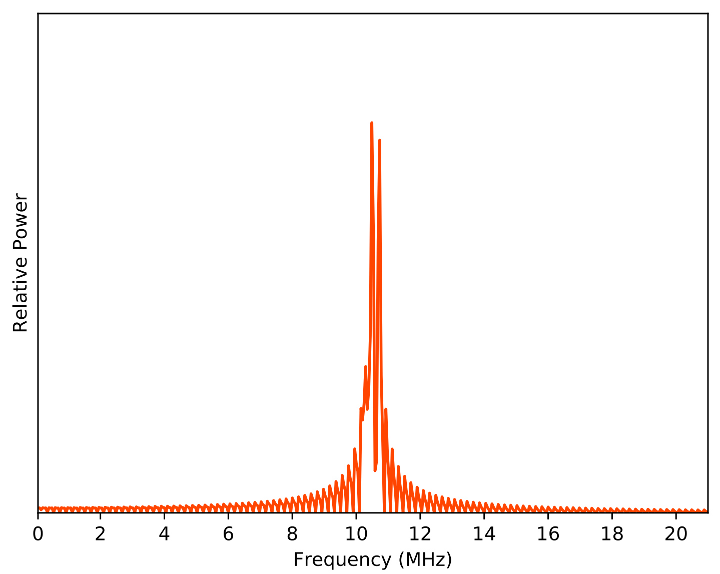


Figure 16: Spectrometer output for chirp.

The spectrometer output for the downconverted chirp, seen in Fig. 17, contains only low frequencies between 0 and 0.5 MHz. This meets the synchronization expected output. Further noise reduction as well as signal amplification to achieve a better Signal–To–Noise Ratio (SNR) will improve this result.

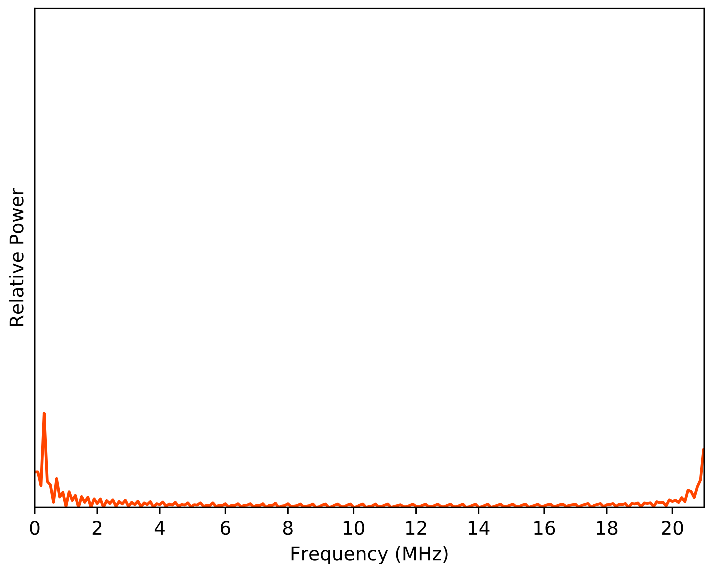


Figure 17: Spectrometer output for testing removal of near–DC signals.

4.4 Analysis

The transmitter wave was shown to have the correct waveform and frequency. The spectrometer accurately displays the frequency spectrum of the received signal. The synchronization scheme is also demonstrated to function correctly. Thus, all components of the system meet the design specifications, and it can be concluded that the system is a working FMCW radar. Usage of more precise components would eliminate the sum frequency and help make the radar more effective. However, such components are expensive and the above testing is sufficient to demonstrate proof–of–concept.

5 Conclusion

5.1 Summary

This thesis discusses the design of an FMCW radar built for the purpose of imaging ash distributions from volcanic eruptions. The design utilizes the ROACH2 signal processing board, which contains a Xilinx® Virtex®–6 FPGA . This design used a varying frequency output chirp—a sine wave from 10 to 11.5 MHz before multiplication and 220.16 to 220.184 GHz after up–conversion—following a sawtooth frequency modulation scheme.

The project was first designed in MATLAB® Simulink®, using pre-written HDL blocks from CASPER as well as Xilinx® IP blocks. The chirp wave vector was generated in MATLAB®.

The design as a whole (including off-board components) contains the entire chirp wave as a vector located in the read–only memory (ROM) blocks on the FPGA. A clocked counter tells the ROM which vector element to output to the DAC, which outputs the chirp waveform. The chirp then is mixed with a pure 13.75 GHz tone and multiplied by a factor of 16, bringing it up to about 220 GHz. This up–converted chirp is sent out through a transmitter antenna as well as looped back as the LO input of the (ideally single–sideband) IF mixer. The reflected wave is read in through a receiver antenna and connected to the RF input of the IF mixer. The IF output is then connected to the ADC input to pass onto the on-board implementation of the FFT algorithm. The data from the FFT is then sent through an Ethernet cable to the accompanying computer, which uses a Python™ script to display the data.

Testing showed the transmitter portion of the design outputs the correct waveform. It also demonstrated that the spectrometer is able to display the input frequencies accurately. Thus, it is shown that the design works, although minor issues such as quantization in the spectrometer output are present.

5.2 Future Work

5.2.1 Hardware

The testing of this design utilized many cheaper and lower–grade components that made the results less than ideal. Following are components which should be changed for high–performance applications:

- The use of a upconverter was omitted. The calculations assume a frequency of 13.75 GHz entering the LO port.
- The use of a frequency multiplier was also omitted. The calculations presented assume a frequency multiplication of 16. This can be accomplished by using 4 harmonic frequency doublers in series.

- The IF mixer used in testing had both sidebands. However, for more accurate spectrometer plots, a single–sideband frequency mixer should be used instead. Note that unlike the other mixer, this one should be a downconverter.
- A low–pass filter should be attached in series with the output of the lower–sideband IF mixer, to remove the leaking signal of the chirp as well as its higher frequency harmonics. The cutoff frequency should be between the highest expected frequency difference (in our case, about 8 MHz) and the lowest frequency in the chirp (when up–converted, about 220 GHz).
- A low–pass filter should be attached in series with the output of the frequency multiplier to remove the sampling artifacts and smooth the waveform of the chirp. The cutoff frequency should be around 250 GHz to remove these high–frequency distortions.
- A low–noise high–gain amplifier should be used to amplify the frequency difference, or IF output of the downconverter.

5.2.2 Design Improvements

The most obvious next step is to make the radar more generalized by accounting for assumptions which simplified this design. Most notably, this radar assumed that the object being measured was stationary; however, it should be noted that volcanic eruptions do not tend to have stationary ash clouds. In addition, the relationship of the frequency shift to the speed is non-linear and depends also on the original frequency [6]. Thus, accounting for the Doppler effect due to moving distributions would improve the design.

The ease–of–adjustment of the design can be improved by switching the use of a ROM to RAM for holding the chirp vector. Doing this allows the chirp to be adjusted and reuploaded without having to recompile the design. Other potential improvements include implementing the other modulation schemes mentioned earlier and evaluating the performance of each.

References

[1] United States Geological Survey, "Volcano Hazards Program FAQs," *United States Geological Survey*. [Online]. Available: https://volcanoes.usgs.gov/vsc/file_mngr/file-153/FAQs.pdf. [Accessed Feb. 10, 2019].

[2] S. Bryan et al, "Measuring Water Vapor and Ash in Volcanic Eruptions with a Millimeter-Wave Radar/Imager", IEEE Trans. Geosci. Remote Sens., vol. 55, no. 6, pp. 3177–3185, Feb. 2017.

[3] S. Suleymanov, "Design and Implementation of an FMCW Radar Signal Processing Module for Automotive Applications," Master's thesis, Univ. of Twente, Enschede, The Netherlands, 2016. [Online]. Available: https://essay.utwente.nl/70986/1/Suleymanov_MA_EWI.pdf. [Accessed May 10, 2018].

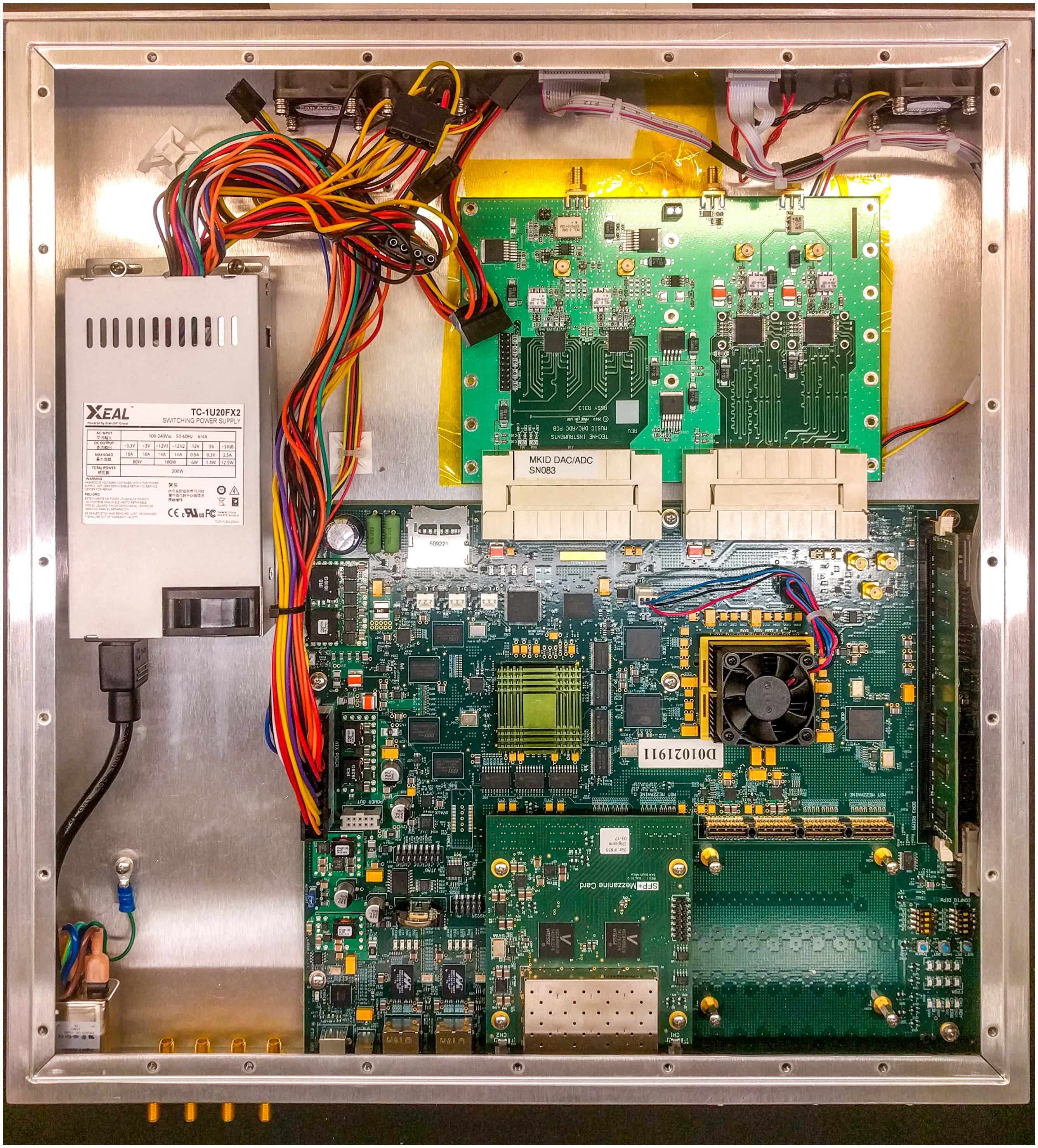
[4] E. Bogatin, Signal and Power Integrity–Simplified, NJ, Englewood Cliffs: Prentice-Hall, 2009.

[5] N. G. de Bruijn, "Uncertainty principles in Fourier analysis" in Inequalities, New York: Academic Press, pp. 57-71, 1967.

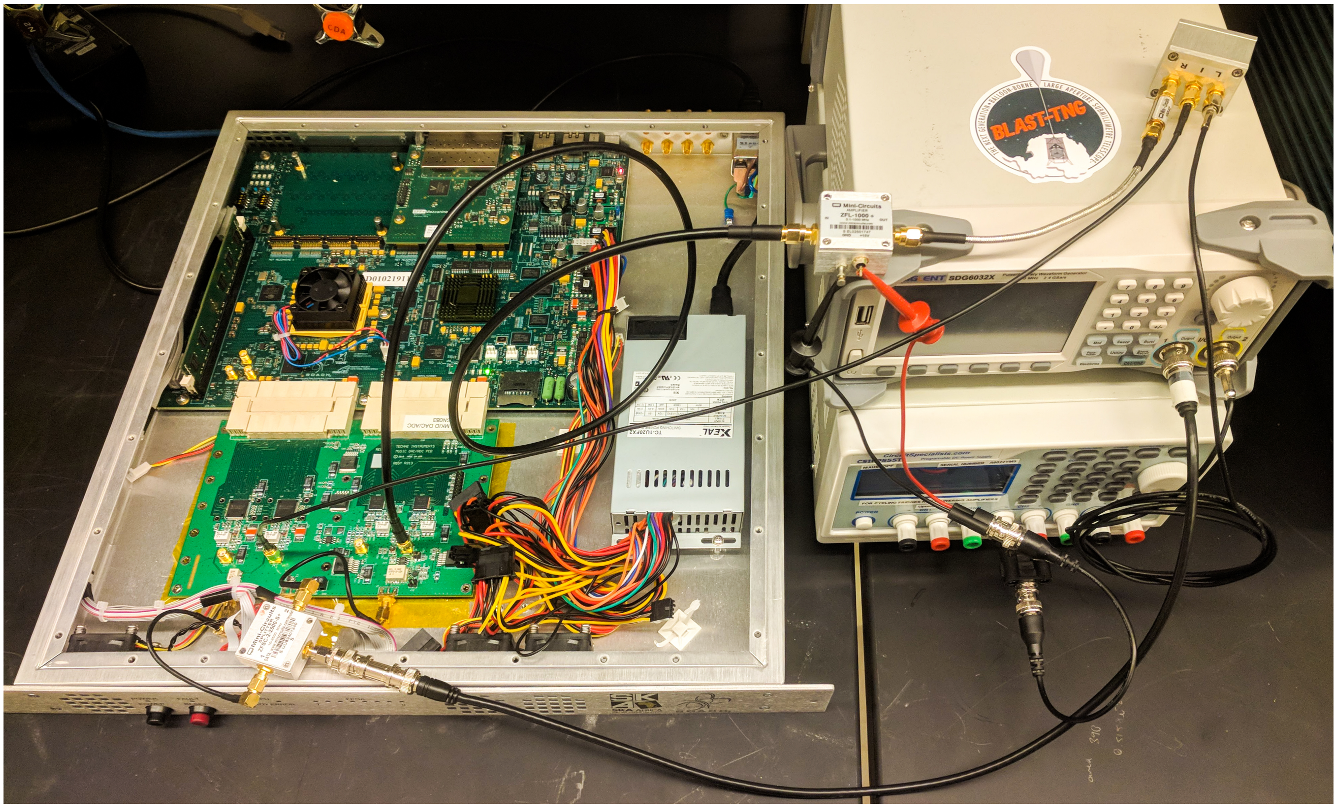
[6] A. Meta, P. Hoogeboom, L. P. Lighthart, "Signal processing for FMCW SAR", IEEE Trans. Geosci. Remote Sens., vol. 45, no. 11, pp. 3519–3532, Nov. 2007.

Appendix A Pictures

A.1 ROACH2 Revision 2 FPGA



A.2 Synchronization Testing Setup



Appendix B Code

B.1 MATLAB® code for generating wave vector

```
% Define parameters
    clk_tx = 200e6;
    size_rom = 2^16;
    num_pts = 2 * size_rom;
    bw_lf_chirp = 1.5e6;

% Create frequency vector
    f = 10e6 + linspace(0, bw_lf_chirp, num_pts);
    w = 2 * pi * f;

% Create chirp vector
    t = (0:1:num_pts-1) / clk_tx;
    chirp = sin(w .* t);

% Split chirp into 2 vectors, even and odd
    S0 = chirp(1:2:end);
    S1 = chirp(2:2:end);
```

B.2 Python™ script for uploading wave vector

```
import casperfpga

IP = '192.168.40.67'        # This was the IP address of the ROACH2 used

fpga = casperfpga.KatcpFpga(IP)

fpga.upload_to_ram_and_program('/path/to/trans_spec.fpg')

fpga.write_int('dac_reset',1)
fpga.write_int('dac_reset',0)
fpga.write_int('start_dac',0)
fpga.write_int('start_dac',1)
```

B.3 Python™ script for generating spectrometer plot

```
import casperfpga
import numpy as np
import matplotlib.pyplot as plt

def plotFFT(fft_len):

    fig = plt.figure(figsize(8,6), facecolor = 'w', edgecolor = 'w')
    plot1 = fig.add_subplot(1, 1, 1)

```

```
    line, = plot1.plot(np.arange(0, fft_len), np.zeros(fft_len), '#FF4500', alpha = 0.8)
    line1.set_marker('.')

    plt.ylim(0, 5000)       # Can be adjusted for height of peaks
    plt.xlim(0, fft_len/2)  # Removes negative frequencies, can be further adjusted

    while True:

        fpga.write_int(regs[np.where(regs == 'fft_snap_ctrl_reg')[0][0]][1], 0)
        fpga.write_int(regs[np.where(regs == 'fft_snap_ctrl_reg')[0][0]][1], 1)
        fft_snap = (np.fromstring(fpga.read(regs[np.where(regs ==
         ↪  'fft_snap_bram_reg')[0][0]][1], (fft_len/2)*8), dtype='>i2')).astype('float')

        i0 = fft_snap[0::4]
        q0 = fft_snap[1::4]
        i1 = fft_snap[2::4]
        q1 = fft_snap[3::4]

        mag0 = np.sqrt(i0**2 + q0**2)
        mag1 = np.sqrt(i1**2 + q1**2)
        mags = hstack(zip(mag0,mag1))

        line1.set_ydata(mags)
        fig.canvas.draw()

    return

#########################################################
#                    Upload to FPGA                     #
#########################################################

run /path/to/upload.py

fft_len = 2**16
roach.write_int(regs[np.where(regs == 'fft_shift_reg')[0][0]][1], (fft_len/2) -1)

#########################################################
#                        Plot                           #
#########################################################

plt.ion()
plotFFT(fft_len)
```

A VERY special thank you to the following people:

Dr. Groppi

Dr. Mauskopf

Dr. Cochran

Dr. Baumann

Adrian and Samuel

Hamdi, Jonathan, Cassie, and the rest of the ASU Cosmology Instrumentation Group

Youhao, Wen, Sarah, Kevin

Jaesang, Kyeongshin, Jake, Alma

Mariam

Nicole

Yifan

Shaojia

Anthony