Upgrade and Re-Deployment of the Telescope Array RADAR (TARA) Cosmic Ray Observatory Remote Stations

By

Steven Prohira

Submitted to the Department of Physics and Astronomy and the
Graduate Faculty of the University of Kansas
in partial fulfillment of the requirements for the degree of
Master of Science

David Besson, Chairperson

Committee members

KC Kong

John P. Ralston

Alice Bean

Date defended: February 25, 2016

The Thesis Committee for Steven Prohira certifies
that this is the approved version of the following thesis :

Upgrade and Re-Deployment of the Telescope Array RADAR (TARA) Cosmic Ray Observatory Remote Stations

David Besson, Chairperson

Date approved: April 29, 2016

Abstract

The Telescope Array RADAR experiment is a bi-static radar search for Ultra-High-Energy Cosmic Rays (UHECR). Here we describe an upgrade and re-deployment of two of our detectors, which, owing to their isolation from the main detector apparatus on Long Ridge, Millard County, UT, are called the Remote Stations (RS). The upgrade described here comprises a total overhaul of the trigger and timing systems, with improvements in signal-to-noise ratio sensitivity of approximately 30 dB. Our new firmware-based trigger method is sensitive to expected signals at SNR of -4 dB at high efficiency. Bench-top tests indicate that this new system is sensitive to a Radar Cross Section (RCS) of $O(1m^2)$. Deployment of the overhauled stations took place in February 2016, with a planned data-taking duration of 3-6 weeks.

Acknowledgements

The author is indebted to Dave Besson for unwavering support and continual guidance, KC Kong for consistent encouragement, Sam Kunwar for inspiration and assistance, Rob Young and Ken Ratzlaff for expertise, Jeff Engelstad, Alexander Novikov, Marina Smirnova, and Gopi Moholabeng for their hands-on assistance in the field, Antonis Stylianou for edits and sanity checks, and the TARA collaboration for remote aid.

Contents

List of Figures

Chapter 1

Introduction

Ultra-High-Energy Cosmic Rays (UHECR) interact with atoms in the upper atmosphere causing cascades of charged particles; these cascades are known as Extensive Air Showers (EAS)(Gorham, 2001). For high enough incident energies, the plasma formed in this EAS becomes dense enough to reflect radio, which may be used to interrogate and classify the primary particle (Abbasi et al., 2014). TARA transmits continuous wave (CW) at 54.1 MHz. When this CW signal reaches the moving plasma core, a red-shifted signal is reflected to the receiving stations. This signal is known colloquially as a “chirp", analogous to a quick, frequency varying audio signal. Our receivers are co-located with the Telescope Array, so that received events may be correlated with the TA ground detector events. The area around TARA is very remote and relatively radio frequency (RF) quiet, so anthropogenic backgrounds in the RF spectrum are minimized.

The first iteration of the RS was plagued by one critical issue that severely compromised the data taken during the interval of June 2014 to September 2015. A faulty piece of hardware in the trigger path resulted in only very high SNR events triggering the device. Based on previous theoretical and subsequent experimental limits placed on the radar cross-section of EAS, the chance of a primary event with sufficient energy to trigger the stations during this period is statistically highly unfavored.

This issue was discovered and isolated during the trip to retrieve the stations at the culmination

of their initial deployment. It was decided in the subsequent few days that rather than attempt a preliminary data analysis on data that was likely devoid of events, an attempt at a re-design and re-deployment would be a better use of resources. As the diagnosed problem, which will be elucidated in this paper, was a hardware issue, it was decided that the entire hardware-based trigger should be migrated to firmware. This decision was very fruitful. It led to a drastic reduction in systematic error within the trigger, as well as total control over parameters that were once invisible or very difficult to probe. Most significantly, it led to an order of magnitude improvement in SNR sensitivity.

In this paper we will outline the construction of the stations, the new trigger and timing systems, some benchmarking of signal sensitivity and timing resolution, and an explanation of the problem that precipitated this revision.

Chapter 2

Remote Station Construction Overview

The two remote stations are fully autonomous. Photovoltaic (PV) panels power the stations during sunlight hours and charge batteries for the night. GPS provides ± 10 ns timing resolution, and microwave communications allow remote access to monitor the system health and retrieve data. Each station brain is a Xylinx Spartan-6 FPGA, controlled by a Raspberry-Pi single board computer (SBC) with 32 GB of on board storage. A high-speed analog to digital converter (ADC) is controlled by the FPGA, which supplies samples for the trigger and data aquisition firmware modules. A system health monitor, developed by the Instrumentation Design Laboratory at the University of Kansas, gives real-time system statistics, such as battery and PV voltage and current, ambient temperature, and approximate RF background levels for each station. Our antennas are custom log-periodic dipole antennas with good forward directivity. The two stations are positioned with their antennas separated along a baseline of 65 meters, giving us azimuthal information for event reconstruction. Before data acquisition, the incoming signals are bandpass-filtered from 55-80 MHz, tightly bandstop-filtered at our carrier frequency of 54.1 MHz, and amplified by 60 dB. Filtration in conjunction with our trigger method, described below, eliminates a good deal of background. Amplification ensures that we are sensitive to the galactic noise floor.

A detailed overview of the station components is given in (Kunwar et al., 2015), with only one significant hardware change: the first iteration of the RS had a separate trigger board, which housed

the faulty components. This has been removed and replaced by the aforementioned firmware trigger.

Chapter 3

Updated Trigger

The characteristic chirp signal described in the introduction is exploited for use in our trigger using a heterodyne method, similar to that employed in FM radio reception. We then filter and envelope detect on this heterodyned signal.

3.1 Heterodyne theory

In heterodyning, a signal modulation is extracted from a carrier by way of a simple trigonometric identity,

$$2cos\theta cos\phi = cos(\theta - \phi) + cos(\theta + \phi). \tag{3.1}$$

In FM radio, for example, the local oscillator in your stereo changes frequency as you tune to the station you want to hear. Then for equation 3.1, $\theta = \omega_{lo}t$ and $\phi = (\omega_c + \omega_{mod})t$, where ω_{lo} is the frequency of the local oscillator, ω_c is the carrier frequency, and ω_{mod} is some modulation, like the audio-frequency voices and instruments in the desired radio broadcast. When $\omega_{lo} = \omega_c$, the difference term leaves only the modulation after mixing, and the sum term is shifted up and may be easily filtered.

Similarly, a Doppler-shifting signal, such as our expected chirp, mixed with a time-delayed

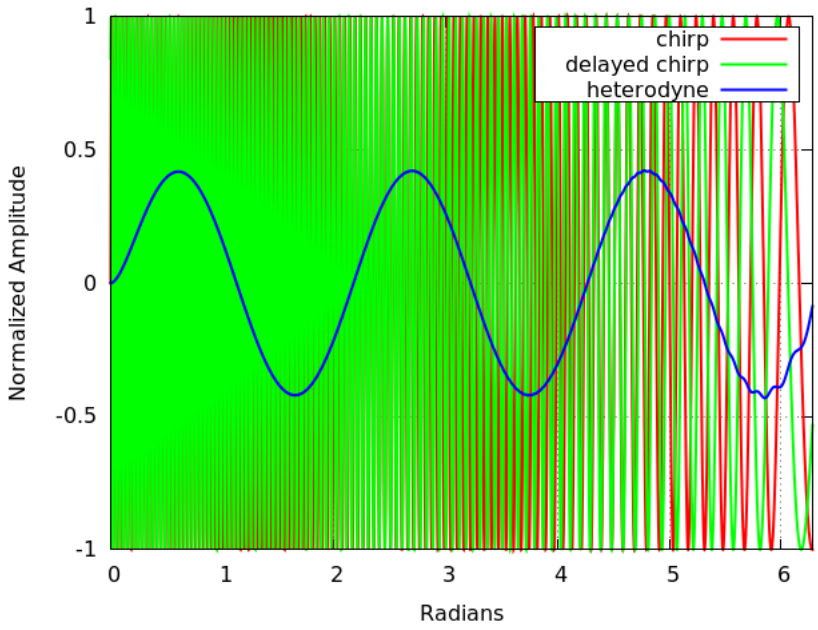

Figure 3.1: An example of the heterodyne method. A chirp with a starting angular frequency of 200 Hz and chirp rate κ of -15 Hz/s is mixed with a copy of itself delayed by a δt of 1/10 of a second. The resultant monotone is 3 Hz.

copy of itself, results in a monotone. I.e. we take the incoming signal, split it, and delay one copy by δt. Then for a chirp,

$$\theta = \omega t + \kappa t^2 \quad and \quad \phi = \omega \tau + \kappa \tau^2, \tag{3.2}$$

where $\tau = t + \delta t$ and κ is the slope of the frequency shift, or chirp rate. We then mix these two together, and the resultant sum term is easily filtered, with the difference term giving a monochromatic signal at a frequency of $2|\kappa|\delta t$. With this method, both continuous wave (CW) signals and noise are suppressed, leaving Doppler-shifting signals. We can, by altering δt, tune the monotone to a desired frequency range for ease of filtration. This method is diagrammed in Figure 3.1.

3.2 Heterodyne implementation

To implement this heterodyne method, we start by sampling the incoming signal at 200 MS/s. These samples are read into two paths, one a simple circular RAM buffer from which data are read

in the event of a trigger, and the other the trigger path. In the trigger path the samples are split, and one half is sent into a small circular RAM buffer to act as our delay, δt. The current sample is then mixed with the delayed sample to produce our heterodyne output signal. We can vary the read point in our circular buffer to adjust the amount of delay.

The sample rate sets the Nyquist frequency at 100 MHz, so any primary CW signals above 50 MHz will have their sum terms pushed up and out of band after mixing. We filter everything below 50 MHz prior to acquisition, so CW below 50 MHz is not present in the primary signal. Therefore, we effectively eliminate CW, which is the largest source of background in the trigger path, effectively lowering our background floor.

3.3 Filtering, envelope detecting, and trigger logic

After the heterodyne, the resultant signal is squared and passed to a time-domain first-order infinite-inpulse-response (IIR) filter module that acts in this capacity as both a filter and an envelope detector. Details of the design of this filter may be found in Appendix A. In hardware, an envelope detector can most easily be described as a series rectifier and resistor followed by a capacitor to ground. The rectifier makes the signal positive-definite, and the time constant of the RC tank sets both the frequency response and the transient response time. In firmware, perfect rectification is possible by way of squaring. The filter coefficients act as the RC tank. We then average over several samples to smooth the envelope, and trigger on this smoothed signal.

The trigger logic includes implementation of a 2 part time-over-threshold (TOT) requirement. In order to trigger, the envelope must rise above a high threshold and then remain above a low threshold for a designated interval, as diagrammed in Figure 3.2. The level thresholds and TOT are adjusted based on environmental and operational factors. Threshold information is recorded at trigger time.

Upon satisfying the trigger criteria, the FPGA timing counters are latched and an interrupt is sent from the FPGA to the SBC, resulting in data transfer. The trigger latch point is 20 μs from the

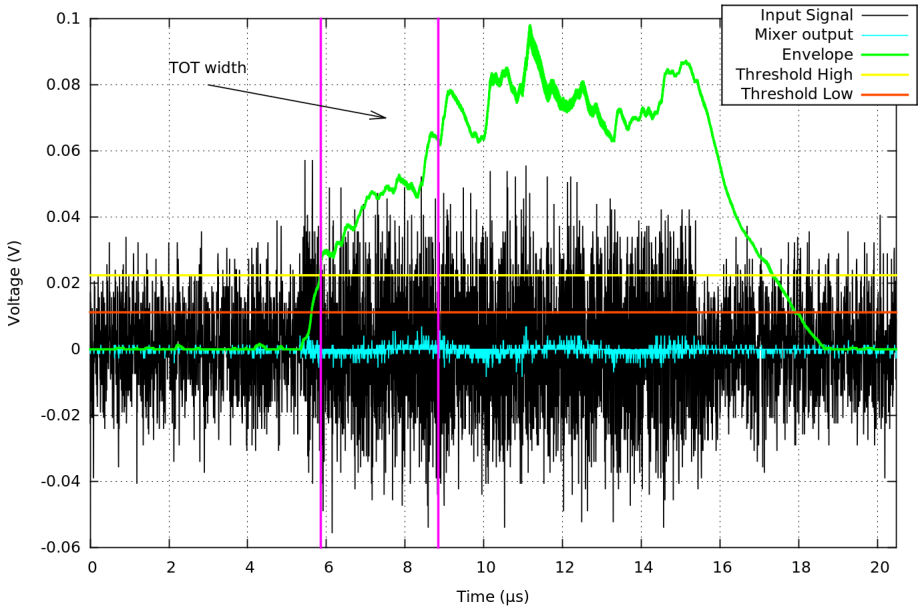

Figure 3.2: Actual positive trigger using FPGA ChipScope-captured traces from field signal. A chirp signal is embedded in noise at SNR 1.5 and sent through the trigger path. The resultant traces are recorded here, with added bands denoting TOT width. Trigger logic evaluates true at the second TOT band if the envelope has not dropped below the low threshold.

start of the circular data buffer, so that 20 μs of data prior to the trigger is recorded as well as 21 μs after it. Upon successful SPI transfer, the FPGA and SBC return to a ready state. Transfer of data to the FPGA is done by way of a first-in-first-out (FIFO) buffer, so that clock speeds between transferrer and transferee may vary, which they do in this case. The SBC has a 16 MHz clock, so transfer of the 16,384 byte buffer per event takes $\sim$1.02 ms.

The envelope detector, working on the square of the heterodyne, has good noise and transient rejection. However, due of the nature of IIR filters, very high amplitude transients will “activate" the envelope for a short time, with a duration that is commensurate with the magnitude of the transient. The TOT width is the critical factor in rejecting time transients of $O(1ns)$ up to $O(1\mu s)$. This is shown in Figure 3.3. High amplitude transients that would trip a simple edge trigger are well rejected.

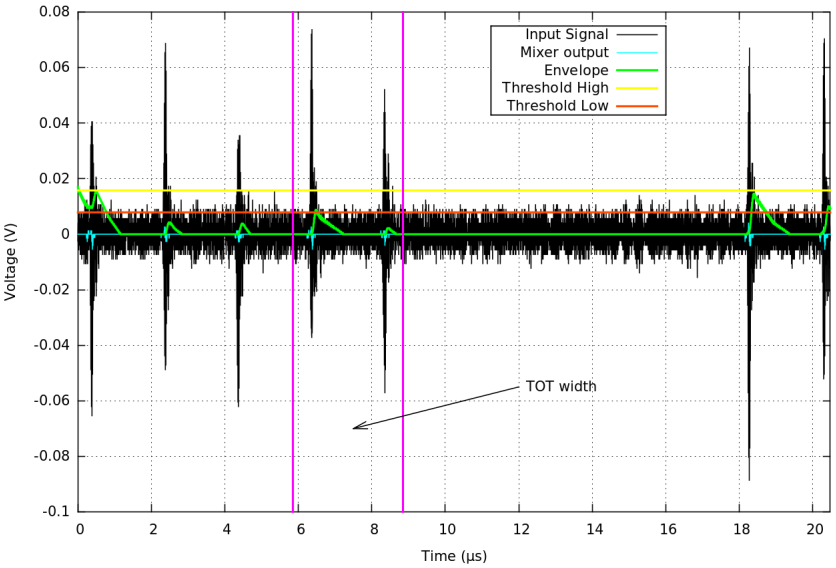

Figure 3.3: Actual noise that does not satisfy trigger logic, showing the response of the filter to transients at high relative amplitude. Such transients may "turn on" the envelope, but a very strong signal is required to both rise above the threshold and persist long enough for the TOT criteria to be satisfied.

Chapter 4

Updated Timing

4.1 System description

Nanoscale timing resolution is essential, both for establishing coincidence between the two RS and with the co-located TA ground detector. Due to the design of our signal path and the GPS units used, a queried timestamp with nanosecond or tens of nanosecond resolution was not possible. The Raspberry-Pi, chosen for it's low power consumption, ease of use, small size, and various other features, has only a single core, and high priority software interrupts such as would be used to query a GPS unit, have variable CPU response time of up to several clock cycles. Instead of using this inaccurate query method, a timing solution was devised that utilizes the high-speed stability of the FPGA units.

Both stations are clocked by a single high-precision 200 MHz crystal oscillator. This oscillator clocks both the ADC and the FPGA in each station. Running both stations off of the same clock has the benefit of synchronizing system functions, and uniform cable lengths minimize phase offsets.

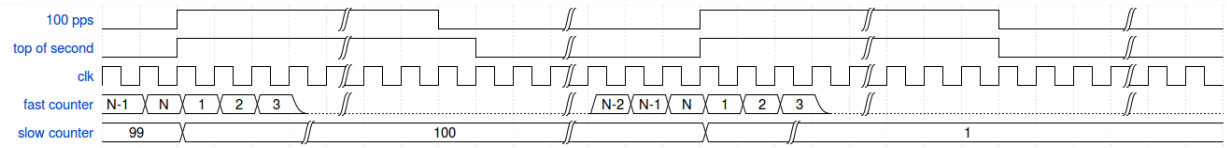

Figure 4.1: Timing schematic. The pps line is shown twice, once for reference and once for the top-of-the-second, where the imminent start of the second is indicated by a longer pulse.

The timing system is broken down into two parts: single-second precision, and sub-second precision, as follows:

1. Single-second precision: Each station is equipped with an iLotus GPS receiver and antenna. This antenna provides query-able timestamps with $\pm$20 nS error to the GPS time. To achieve our down-to-the-second timing precision, the GPS unit is queried by the SBC once every 20 seconds for a timestamp, which is used to update the system clock on the SBC. In the event of a trigger, this timestamp is recorded to the second in the event header. In this way, we obtain accurate universal timing to the second, but must rely on a more stable system for higher precision.

2. Sub-second. The GPS unit also provides a 100 pulse-per-second (pps) output line with the same error characteristics. The 100 pps line from one of the iLotus boards is split and sent to both FPGAs where each reads that signal locally and uses it to update two counters, as diagrammed in Figure 4.1. A slow counter counts the 100pps pulses. A fast counter counts the number of FPGA system clock rising edges between successive increments of the slow counter. On the rising edge of each incoming pulse the slow counter increments and the fast counter is reset. The slow counter resets every second, as the iLotus chip sends out a slightly longer pulse to indicate the top of the second. When a trigger is registered, both counter values are latched and sent as a nanosecond timestamp, which is recorded along with the second-scale stamp on the SBC.

Benchtop tests with a coincident, single rising edge trigger demonstrate a σ of 2.66 ns drift between timestamps recorded by the two stations for $\sim$5000 causal triggers registered over a total time of 1 hour, as shown in Figure 4.2.

4.2 Angular Resolution

During a field test in January 2016 the angular resolution of the stations was tested at KU. In the field, the antennas are separated along a 65 m baseline, giving a coincidence window of roughly

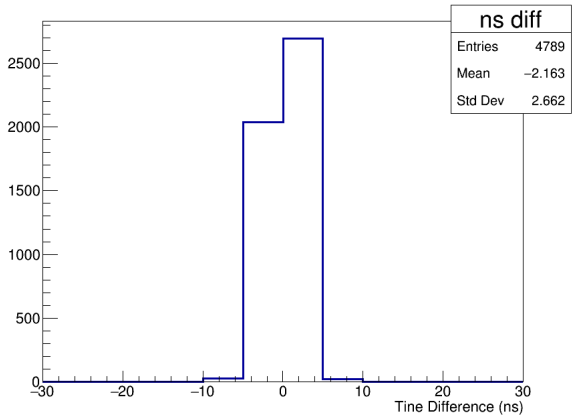

Figure 4.2: Time stamp difference between stations for a logic level external trigger.

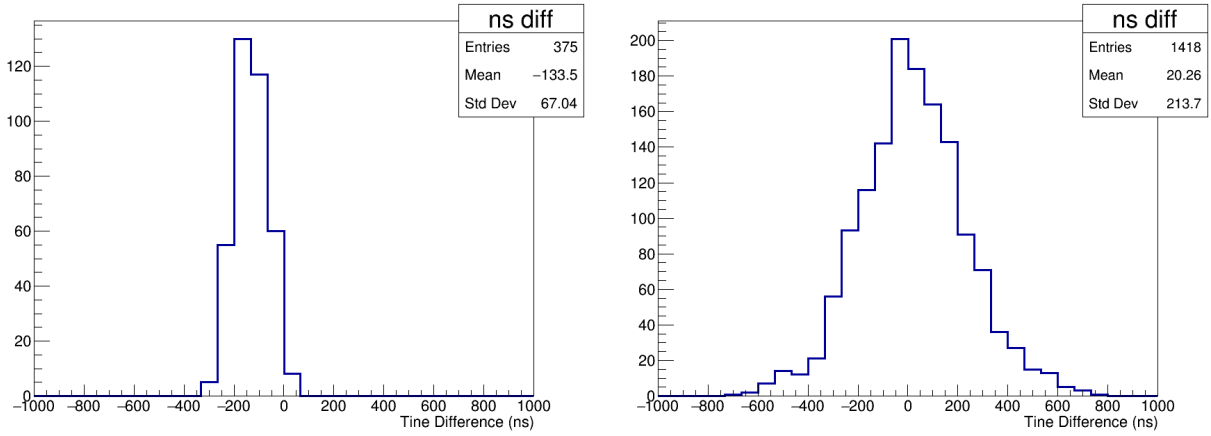

Figure 4.3: SNR=4 and SNR=1 signals, respectively, showing the effect of signal amplitude on the width of the timestamp difference distribution.

$\pm$200 ns. Thus, if a signal were to occur at one end of a horizontal line connecting the two antennas, one station would see it roughly 200 ns before the other. Though our timing system, as described, has nominal resolution well within this window, the phenomenon of trigger-point slewing between the two stations smears out the fine timing. This is illustrated in Figure 4.3, where a SNR 4 signal and a SNR 1 signal, respectively, show the effect of trigger point slewing. The distribution of timestamp differences recorded for a coincident signal smears out as the SNR decreases.

This effect of trigger point slewing appears to have the effect of reducing our angular resolution for low SNR signals. However, as further tests indicated, it actually does the reverse. A slight difference between the two stations in the received amplitudes of a coincident trigger provides

angular resolution proportional to the signal amplitude ratio. This is independent of the absolute amplitude of the received signal. We explain this as follows.

Consider a triangle formed by the chirp source and the two antennas. $\vec{r_1}$ is a vector from the source to antenna 1, and $\vec{r_2}$ is a vector from the source to antenna 2. Assuming uniform atmosphere between the source and the antennas, as well as nominal gains in the two stations, the received amplitude at each antenna is simply a function of distance. Then, the relative amplitude between the two stations is proportional to the ratio $|\vec{r_2}|/|\vec{r_1}|$. This can be simulated by feeding a coincident chirp into the two stations and attenuating one signal path more than the other. Figure 4.4 shows the result of this test. In the figure, the time difference in nanoseconds is plotted, and each peak corresponds to a different incoming signal amplitude ratio. From left to right in the figure, the ratio in voltage between RS2 and RS1 is .5, .75, 1, 1.5, 2. The width of the peaks, as before, is proportional to the SNR. Here, a SNR of 1.5 was used. The recorded time difference between the stations is directly proportional to the received amplitude, which is directly proportional to the distance from the source, so this provides a constraint on the position of our source signal.

Note that in Figure 4.4, while there are distinct peaks spaced correctly for the relative input amplitudes, the actual measured time difference is on the order of microseconds, while the true offset in time is zero, as we have identical cable lengths going from source to each station. The relative difference in trigger point is responsible for this large offset. To correct for this, we cross-correlate the associated events responsible for the distribution of Figure 4.4 with one another, and find the maximum correlation value. We subtract this offset from the recorded timestamp difference. This procedure is diagrammed in Figure 4.5, and the result is given in Figure 4.6. The correction has the effect of bringing the events into the range expected for causal coincidence by eliminating the artificial offset introduced in trigger point slewing.

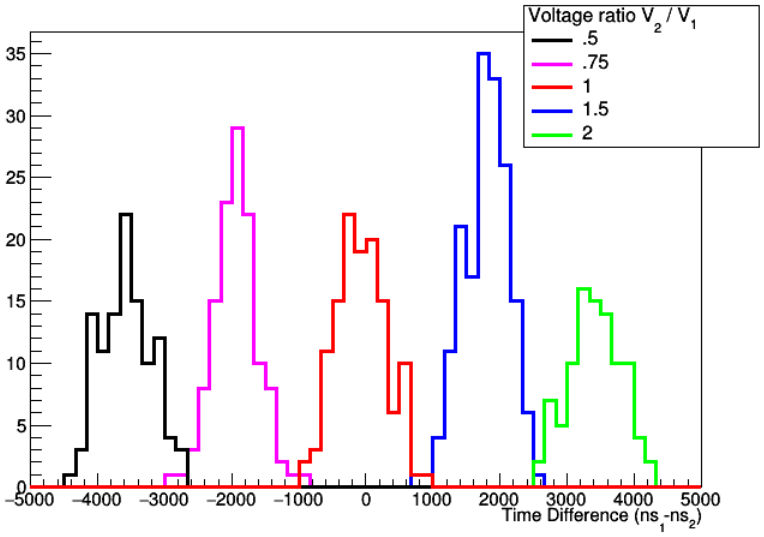

Figure 4.4: Relative difference in received voltage of coincident signal gives angular resolution via trigger point slewing.

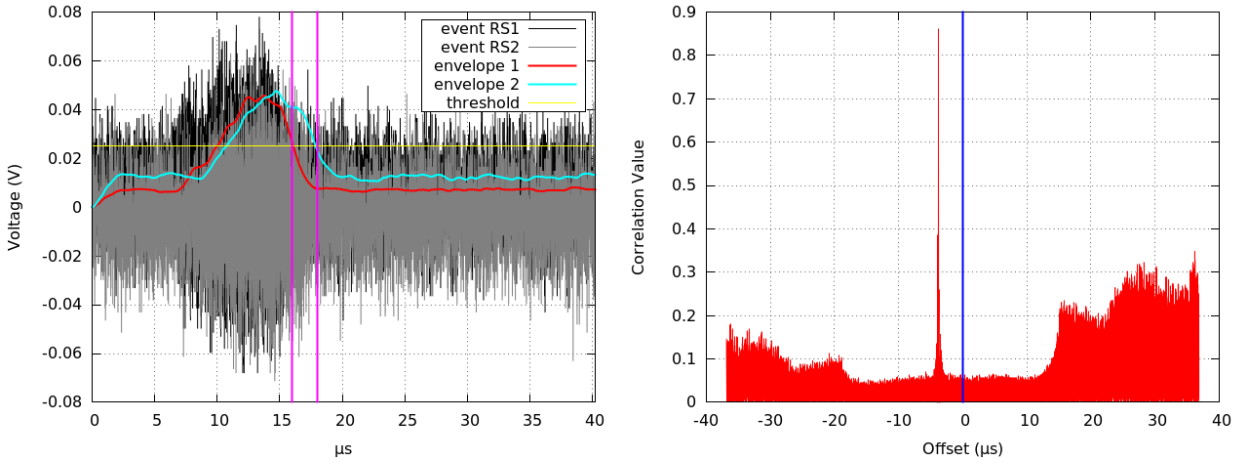

Figure 4.5: Waveform of the same event captured by two stations, left, and cross correlation of the same waveforms, right. The point of maximum correlation is identified by the cross correlation procedure, and the time offset is subtracted from the recorded GPS timestamp difference between the stations.

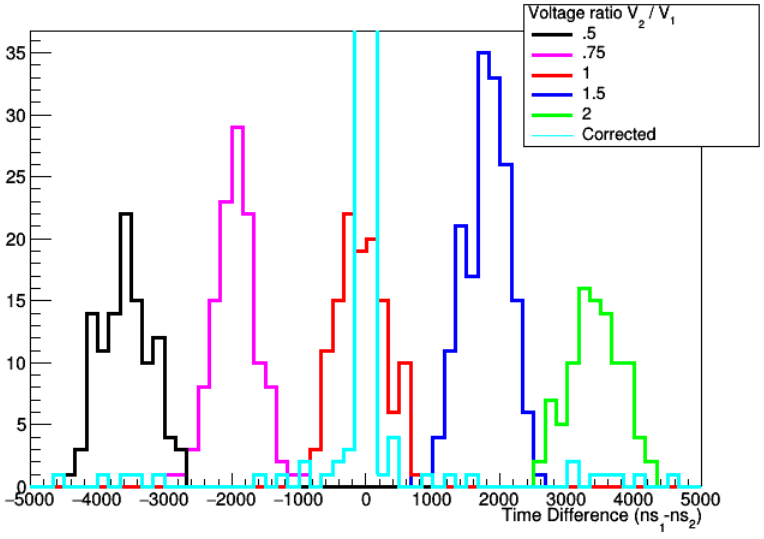

Figure 4.6: Trigger point correction via cross-correlation. Causal timestamps migrate toward zero after correction.

Chapter 5

System Sensitivity

The new system has high trigger efficiency down to SNR $\sim$1, and depending upon the character of the local noise on Long Ridge during data-taking, perhaps lower. Signal generator tests with Gaussian noise and sub-noise chirps show efficient sensitivity down to -4 dB SNR, but this optimistic scenario is unlikely in the field. Obviously, the RF noise condition of our lab, being in the center of town, is not ideal for tests, and is far noisier than the conditions on Long Ridge. Even still, we are able to capture transmitted events at SNR of roughly 1, as shown in Figure 5.1. The chirp is visible in frequency space; in time space, the signal amplitude is noticeably below the noise spikes in the trace.

5.1 Threshold Scan

To perform a scan for reliable trigger sensitivity, a threshold was set just above the level that would trigger on noise. This threshold was empirically determined for our laboratory, and differs significantly from the condition on Long Ridge. We then sent test chirps at various signal levels into the DAQ and monitored the number of triggers registered per chirps sent.

A reasonable approximation for the received signal power of a scattered radio signal is given by the Friis Equation,

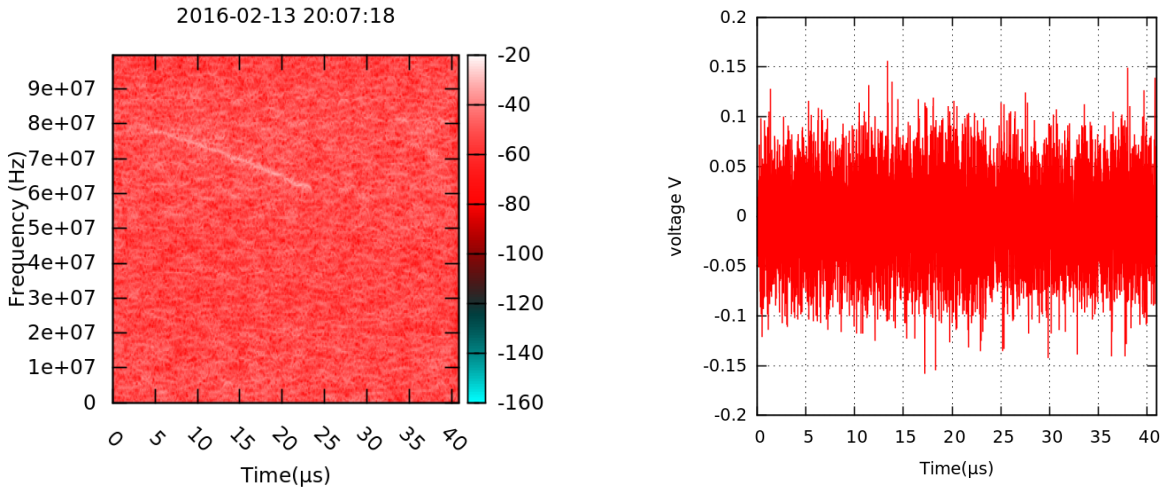

Figure 5.1: SNR $\sim$1 transmitted event, in frequency and time space. The "head" of the chirp is at $\sim 20\mu$s.

$$P_r = P_t \frac{G_r}{4\pi R_r^2} \sigma_{eas} \frac{G_t \lambda^2}{16\pi^2 R_t^2}, \tag{5.1}$$

where subsrcipts r and t refer to receiver and transmitter, G is system gain, λ is the wavelength of the received signal, R is the distance from the EAS, and σ_{eas} is the radar cross section (RCS). The RCS is the cross sectional area of the EAS from which radio can be reflected, and it is the primary unknown in our experiment. If we treat our test signal as the received reflection from an event, then we can plot the trigger efficiency as a function of the RCS, having determined the remainder of the terms in Equation 5.1 with parameters from the field. This is shown in Figure 5.2.

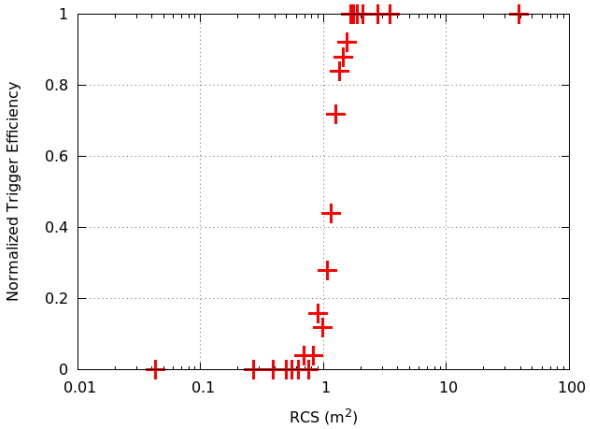

Figure 5.2: Trigger efficiency vs. cross-sectional area of the EAS ionized core, or radar-cross-section, as measured in the lab at KU.

Chapter 6

Problem Identification and Solution

The anomaly that precipitated the RS re-design was discovered while performing a threshold scan on the station recovery trip in October 2015. Using a transmitted calibration chirp signal, we sought to quantify the lowest reliably discernible signal. This test had been performed previously with mixed results-a series of hardware issues had kept us from having an adequate full-system threshold scan with both stations before this trip.

Our results were surprising. It was discovered that there was a very small dynamic range of effective threshold values, and that small threshold changes of O(1 mV) were enough to pass from constant noise triggers to no triggers at all. Furthermore, it was discovered that the same threshold would allow triggers with vastly different rms and signal characteristics. For example, the two events of Figure 6.1 triggered on the same threshold setting, which was very high, about 0.4 V. Most worrying, a calibration chirp event, like the high-amplitude signal in Figure 6.1, would not trigger the station if its output amplitude were any smaller than what is shown here. Note the order of magnitude difference in the y axis values.

It should be noted here that the waveforms of Figure 6.1 are the recorded data, not the trigger signal. Figure 6.2 shows a field-captured calibration chirp that has been, for purposes of demonstration, heterodyned and envelope detected, mimicking what the trigger path should have been seeing to a reasonable approximation. Shown there are three thresholds that were set with the

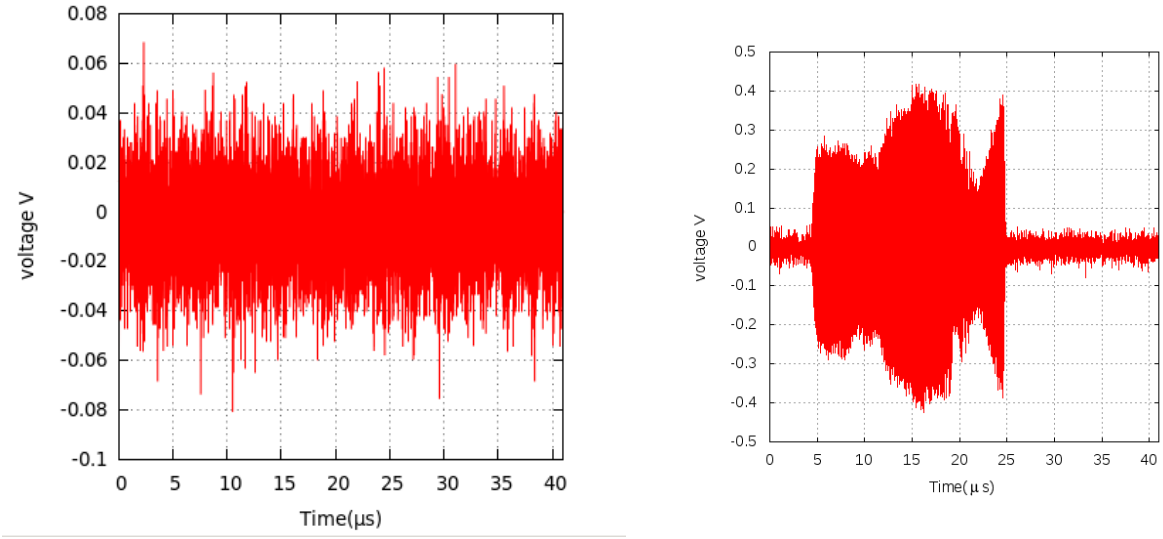

Figure 6.1: Two events that satisfied the trigger for the same high ($\sim$.4 V) threshold setting.

following results.

1. The lowest threshold shown resulted in constant noise triggers.

2. The middle threshold shown resulted in some noise triggers, and would reliably trigger on calibration signals of the pictured amplitude, but no lower.

3. The top threshold shown would result in no triggers of any kind.

Yet it is clear from Figure 6.2 that all three threshold levels should have triggered on the chirp, and only on the chirp. Such small variations in threshold level should not have resulted in the observed behavior in a properly functioning system.

Subsequent tests traced the problem to a low-bandwidth diode in the trigger path that attenuated incoming signals down to nearly indiscernible levels before they reached the point of digitization, so that all signals big or small were truncated to a similar amplitude. This explained the conundrum of Figure 6.2, and indicated that our experimental sensitivity was not where we had hoped.

Therefore, a full redesign was initiated in order to bring our experimental sensitivity up to adequate levels. The solution was to migrate the entire trigger path to firmware. The trigger theory remains the same, but the implementation is vastly different. There are many advantages to

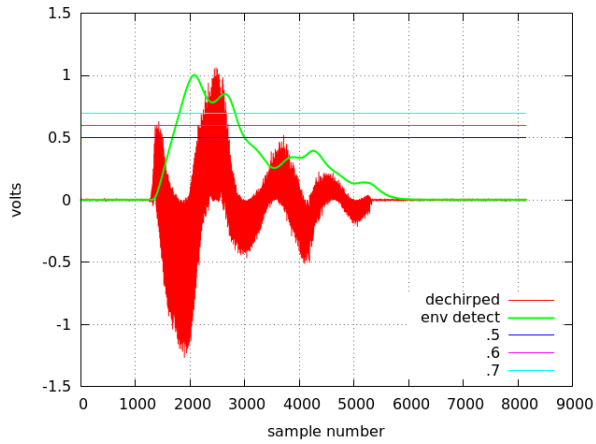

Figure 6.2: A heterodyned field-captured chirp, and the associated filter and envelope detection, along with three thresholds.

a firmware trigger. The greatest of these is the overall reduction of systematics. The trigger path in hardware had 14 circuit elements that the incoming signal needed to pass through prior to digitization. These were splitters, amplifiers, mixers, resistors, capacitors-all things that introduce systematic error. With a firmware-based design, the incoming signal is digitized and then manipulated only algorithmically, so systematic error is greatly minimized, and where it exists it is highly quantifiable. Firmware is also highly portable, so there is greater cohesion between trigger systems in different stations using the same hardware. Since each step of the trigger path is a firmware module, each can be probed, stored, and scrutinized in software.

References

Abbasi, R. et al. (2014). Telescope array radar (tara) observatory for ultra-high energy cosmic rays. *Nuclear Instruments and Methods A*.

Arfken, G. (1966). *Mathematical Methods for Physicists*. Academic Press, New York, 1 edition.

Boas, M. (2006). *Mathematical Methods in the Physical Sciences*. Wiley and Sons, New Jersey, 3 edition.

Gorham, P. (2001). On the possibility of radar echo detection of ultra-high energy cosmic ray- and neutrino-induced extensive air showers. *Astroparticle Physics*, 15, 117–202.

Kunwar, S. et al. (2015). Design, construction and operation of a low-power, autonomous radio-frequency data-acquisition station for the tara experiment. *Nuclear Instruments and Methods A*.

Lyons, R. G. (1997). *Understanding Digital Signal Processing*. Addison Wesley, Massachusetts.

Smith, J. O. (2014). *Introduction to Digital Filters with Audio Applications*. https://ccrma.stanford.edu/ jos/filters/.

Appendix A

Infinite Impulse Response Filters

The firmware implementation of a lowpass filter and envelope detector requires optimization of a time-domain filter. This filter has to be both fast and stable if it is to act as a trigger signal. Broadly speaking, simple time domain discrete filters fall into two categories: finite impulse response (FIR) and infinite impulse response (IIR) (Lyons, 1997). The former refers to filters that only act on incoming samples, and the latter are recursive, and act both on incoming samples and previous filter outputs. They are so named because the output of an FIR is only a function of incoming samples, and so if an incoming signal amplitude goes to zero, so too will the output. But an IIR is recursive, so even if the incoming signal goes to zero, the output of the filter can only asymptotically approach zero. FIR filters are generally considered stable due to this lack of recursion, but to achieve good filtration results, dozens to hundreds of filter coefficients (calculations) are needed, which translates to a corresponding time delay in filtration Alternatively, if one can design a stable IIR, only one or two calculations must be performed between filter input and output, and the transient response is excellent. Below we describe IIR filters, and their implementation in a firmware setting.

A.1 Sampling, Transfer Functions, and the z-Domain

Consider a radio signal, an antenna, and a computer. The radio signal exists as some time varying electric field in the space around the antenna, and we can imagine that the antenna measures an

electric potential at a point in this space corresponding to the electric field of the signal. In order to translate the external analog potential into a stream of discrete values, the computer takes 'snapshots' of the voltage at regular intervals. This process is called sampling. The shorter the time interval between samples and the greater the resolution of possible values for the measured voltage, the closer the discretized signal resembles the analog signal it should represent. Mathematically, the process of sampling is best described by using the Dirac delta on a continuous function.

$$f(n) = \int_{-\infty}^{\infty} f(t)\delta(t-n)dt \tag{A.1}$$

Here the Dirac delta picks out the value of $f(t)$ at n, where n is the index of an array, or element in a vector, representing the discretized signal f. We can imagine that the Dirac delta is some circuit device, such as an analog-to-digital converter (ADC) that gives no output except for when some instruction is sent to it (like a clock pulse) that it should measure and read out the voltage, $f(t)$, at its input, giving the result $f(n)$.

We can then write the output values, $y[n]$ in terms of the input values, $x[n]$ as

$$y[n] = x[n-1] \tag{A.2}$$

This implies that the output value is the previous input value, as causality demands. Furthermore, if we assume that the samples do not pass through the computer "unscathed", meaning they are acted upon in some fashion, we may rewrite this equation with a coefficient as below.

$$y[n] = cx[n-1] \tag{A.3}$$

In an analog filter, the individual circuit elements between input and output alter the character of the signal in some predictable way. In digital filters, there must exist some function that maps input values to output values, taking the place of these circuit elements. This function is called a transfer function, and will be designated $H(x)$, where x is the domain of the transfer function.

$$H(t) = \frac{Y(t)}{X(t)} \tag{A.4}$$

Here the transfer function of output to input signals, Y to X, is given in the time domain. We can write the transfer function in any domain in which the signal may be described, and in this paper we will consider several, including the z-domain.

The z-transform is the discrete valued cousin of the Laplace transform. Its formal expression is given in Eq. A.5.

$$F(z) = \sum_{0}^{\infty} f[n] z^{-n} \tag{A.5}$$

The best way to describe the function of the z-transform is by giving a value to z. Since z is a complex number it may be represented as $z = re^{i\theta}$, or, calling r unity and replacing θ by ω, $z = e^{i\omega}$. Plugging this into Eq. A.5 returns the discrete Fourier transform (DFT) of $f(t)$. Essentially, the z-transform maps a discrete function in time into a function in the complex plane, which more conveniently describes quantities such as frequency and phase.

$$\sum_{0}^{\infty} f(n) z^{-n} \Rightarrow \sum_{0}^{\infty} f[n] e^{-i\omega n} \tag{A.6}$$

So our procedure is as follows:

1. Define the time-domain transfer function of the analog filter we want to represent. (This step is often simplified by using Laplace transforms, as will be demonstrated.)

2. Take the z-transform of this function.

3. Map the solution of this transform to sample coefficients for a simple time domain filter.

4. Write software to implement the filter, and plot the filter response.

The next sections demonstrate this procedure for the RC low-pass filter.

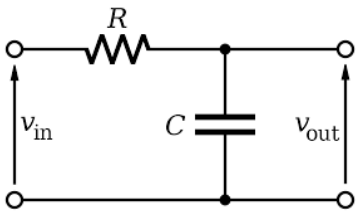

Figure A.1: Circuit diagram representation of an analog RC lowpass filter. (Wikipedia)

A.2 The Analog Filter Transfer Function

The simple RC lowpass circuit of Figure A.1 is one in which the output voltage is determined by the behavior of the two circuit elements, according to the differential equation of (A.7). The impedance of a capacitor goes as the inverse of frequency, so the path to ground looks blocked for low frequencies and favorable for high frequencies. Assuming the circuit is closed, then the current through the capacitor and resistor are equal, according to Kirchoff's circuit rules.

$$\frac{v_i - v_o}{R} = C\frac{dv_o}{dt} \tag{A.7}$$

Here v_i is the input voltage and v_o is the output voltage. $v_i - v_o$ is the voltage drop across the resistor. In keeping with digital signal processing standards, we rewrite Eq. A.7 with $x(t)$ for the input voltage and $y(t)$ as the output voltage.

$$\frac{x(t) - y(t)}{R} = C\frac{dy(t)}{dt} \tag{A.8}$$

Now we take the Laplace transform, $\pounds = F(p)$, of both sides of the equation, and move R, denoting $\pounds(y)$ as Y and $\pounds(x)$ as X.

$$RC(pY - y_o) = X - Y \tag{A.9}$$

We want to find a transfer function in the form $H(p) = Y(p)/X(p)$, so we distribute the RC term and collect terms in Y.

$$Y(RCp+1) = X + RCy_o \tag{A.10}$$

And

$$Y = \frac{X + RCy_o}{(RCp+1)} \tag{A.11}$$

As y_o is an initial constant for the output voltage we are free to call it 0, so we now have our desired function.

$$H(p) = \frac{Y(p)}{X(p)} = \frac{1}{1+RCp} \tag{A.12}$$

We want (A.12) in terms of Laplace transforms that may be simply used. Rewriting $1/RC$ as ω_{rc} and multiplying through we find a form that maps via an inverse Laplace transform into the time domain.

$$H(p) = \frac{Y(p)}{X(p)} = \frac{\omega_{rc}}{\omega_{rc}+p} = \omega_{rc}\frac{1}{\omega_{rc}+p} \tag{A.13}$$

This gives a time domain transfer function in terms of ω_{rc}.

$$H(p) = \omega_{rc}\frac{1}{\omega_{rc}+p} \Rightarrow H(t) = \omega_{rc}e^{-\omega_{rc}t} \tag{A.14}$$

We have thus derived an expression in (A.12) that compares input to output values of a given analog circuit. If we call p a complex number with frequency $i\omega$, and set p equal to $1/RC$ then the denominator will approach 2 and result in a value of 1/2. This is called the *characteristic* or *cutoff* frequency of the circuit and is a convenient measure for the frequency response of a filter. If we consider that (A.12) gives us the output voltage of a first order filter, we might recursively use this value as the input voltage for a second, identical filter in series, and increase our stopband attenuation; this is standard in analog filters. Solving for $X(p)$ in (A.12) gives $X(p) = Y(p)(1 + RCp)$, and if we take that as the $X_2(p)$, or input value, of the second filter, we obtain a transfer

function with the following form:

$$H_2(p) = \frac{Y_2(p)}{X_2(p)} = \frac{1}{(1+RCp)^2} \tag{A.15}$$

Here we have denoted the second order transfer function by the subscript 2. The Laplace transform of this may also be computed, having multiplied through to recast it in a recognizable form, which we find in a lookup table.

$$H_2(p) = (\omega_{rc})^2 \frac{1}{(\omega_{rc}+p)^2} \Rightarrow H_2(t) = (\omega_{rc})^2 t e^{-\omega_{rc}t} \tag{A.16}$$

Equations (A.14) and (A.16) will be the basis for first and second-order filters in what follows.

A.3 The z-Transform of a Time Domain Transfer Function

Now that we have a transfer function in the time domain, we use Eq. A.5 to transform it into the z domain, for reasons which we now detail.

From Eq. A.6, when the magnitude of z is 1 with phase $e^{i\omega}$, the z-transform reduces to the Discrete Fourier Transform. The z-transform takes a time domain input function, transforms it into a discrete valued expression in z (complex space), and then allows us to directly calculate time domain filter coefficients due to a property of the z-transform known as shift, or time delay. From the theory of the z-transform (Arfken, 1966) we can show how these equations lead us to time-domain coefficients.

Note that this system is classified as both linear, meaning the output value is a linear function of the input value, and time-invariant, meaning that the system through which the signal passes (the filter) does not change with time. An example of a time-varying system would be something like $y(n) = sin(n)x(n-1)$, where the coefficient would vary depending on depth into the sampled signal. First, it is useful to restate the equation relating our discrete input values to output values.

$$y[n] = cx[n-1] \tag{A.17}$$

We then present the general form for the z-transform, Equation A.5, using this relation.

$$Y(z) = \sum_{n=-\infty}^{\infty} y(n)z^{-n} = c\sum_{n=-\infty}^{\infty} x(n-1)z^{-n} \tag{A.18}$$

Setting m = n-1, we can rewrite (A.18) as:

$$Y(z) = c\sum_{m=-\infty}^{\infty} x(m)z^{-(m+1)} = c\sum_{m=-\infty}^{\infty} x(m)z^{-m}z^{-1} \tag{A.19}$$

Comparing (A.18) to (A.19) we see that the z^{-1} operator acting on a sample is the same as taking the z transform of the previous sample. It essentially is an operator that moves backwards one sample in the series.

Now we take the z-transform of our two Laplace-transformed transfer functions, Equations (A.14) and (A.16). Using a lookup table, we obtain the z-transform of our first order filter.

$$H(t) = \omega_{rc}e^{-\omega_{rc}t} \Rightarrow H(z) = \omega_{rc}\frac{z}{z - e^{-\omega_{rc}T}} \tag{A.20}$$

multiplying through by z^{-1} we arrive at our z-transformed transfer function.

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\omega_{rc}}{1 - e^{-\omega_{rc}T}z^{-1}} \tag{A.21}$$

The ω_{rc} term has carried through the equation and can be used as a gain value. The T symbol represents the sampling period, or $1/f_s$ where f_s denotes the sampling rate of our computer. Customarily, the equation will also include a scaling value along with ω_{rc} so that our gain at DC is = 1, or whatever gain value is requisite for a particular application. In our case we use T as our scaling factor.

Rearranging terms gives us the following expression, which we immediately put in terms of the discrete sample values, as per the delay property of the z-transform. We have thus found our

time domain filter coefficients.

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\omega_{rc}}{1 - e^{-\omega_{rc}T} z^{-1}} \Rightarrow Y(z)(1 - e^{-\omega_{rc}T} z^{-1}) = X(z)\omega_{rc} \tag{A.22}$$

$$y[n] = \omega_{rc} x[n] + e^{-\omega_{rc}T} y[n-1] \tag{A.23}$$

The same is done for the second order equation.

$$H(t) = \omega_{rc} t e^{-\omega_{rc}t} \Rightarrow \frac{Y(z)}{X(z)} = \omega_{rc}^2 \frac{T e^{-\omega_{rc}} z^{-1}}{1 - 2e^{-\omega_{rc}T} z^{-1} + e^{-2\omega_{rc}T} z^{-2}} \tag{A.24}$$

$$y[n] = T\omega_{rc}^2 e^{-2\omega_{rc}T} x[n-1] + 2e^{-\omega_{rc}T} y[n-1] - e^{-2\omega_{rc}T} y[n-2] \tag{A.25}$$

A.4 The Infinite-Impulse Response Low-Pass Filter

Equations (A.23) and (A.25) provide the time domain representations of the initial analog RC filter and its second order cousin. We can now use the sampling period and cutoff frequency of our desired filter, remembering that the sampling period sets the bandwidth of our data, and obtain numerical values for our coefficients.

As an example, we now implement a low pass filter with a sampling period of 1 μs, and cutoff frequency of 3000 Hz for use in a heterodyne application. Here we wish to remove high frequency terms and retain low frequency modulations. We find the filter coefficients for the first and second order filters using our expressions from the previous section.

$$y_1[n] = .01884955x[n] + 0.98132698y[n-1] \tag{A.26}$$

$$y_2[n] = 0.0003553x[n-1] + 1.962654y[n-1] - 0.96300265y[n-2] \tag{A.27}$$

In each case, the first coefficient (that for the input sample) has been scaled by a factor of T,

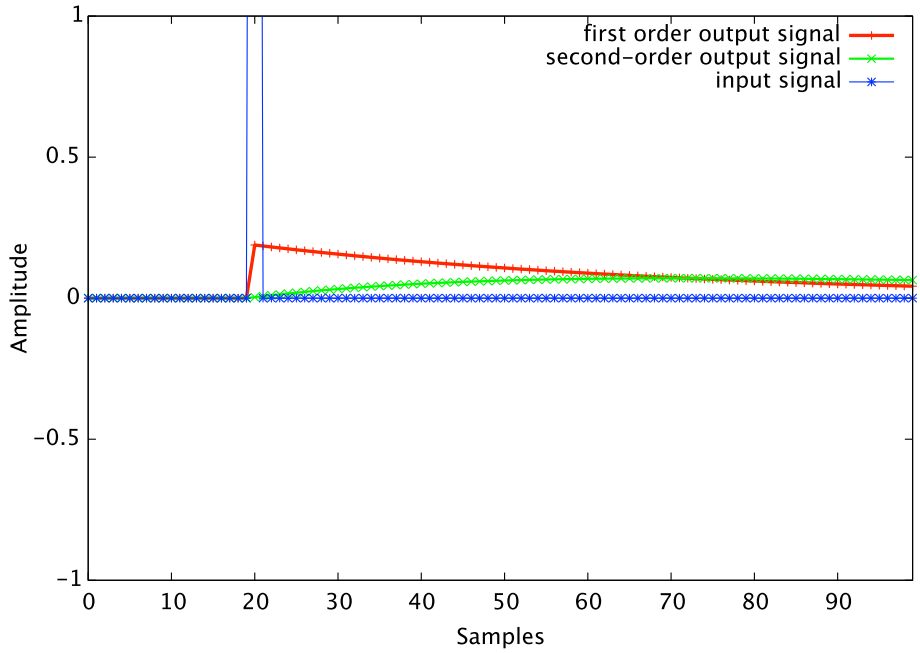

Figure A.2: Time domain representation of the impulse response of the first order (red) and second order (green) digital low-pass filters to an amplitude=10 narrow impulse. Both filters attenuate this signal drastically, as it has very high frequency components.

which brings the DC gain to unity. These values can immediately be implemented in software (here written in C) and tested with various input signals. Figure A.2 shows the response of these filters to an impulse, or a delta function. The z-domain representation of the filter allows us to calculate the poles and zeroes of a filter, and if they have certain values, i.e. if $|z| < 1$ for poles, then the filter will converge and be stable (Boas, 2006). Figure A.2 shows that the impulse, a very high frequency signal, is highly attenuated by our filters, as it should be.

Figure A.3 shows the frequency response of these filters on a log-log (Bode) plot. The -3 db cutoff point, where the first order filter has an amplitude of 70% of the input value, is at the desired frequency, and the second order filter shows-6 dB atteniation at the cutoff frequency, which is n (filter order) times -3 db, in agreement with circuit theory. The slope of the attenuation goes as the order of the filter, with a higher order filter having a sharper cutoff. Fitting lines to these curves, using (A.12) and (A.15) we find cutoff frequencies of 3030.65Hz and 3002.51Hz for first and second order, respectively, in agreement with the expected analog filter response.

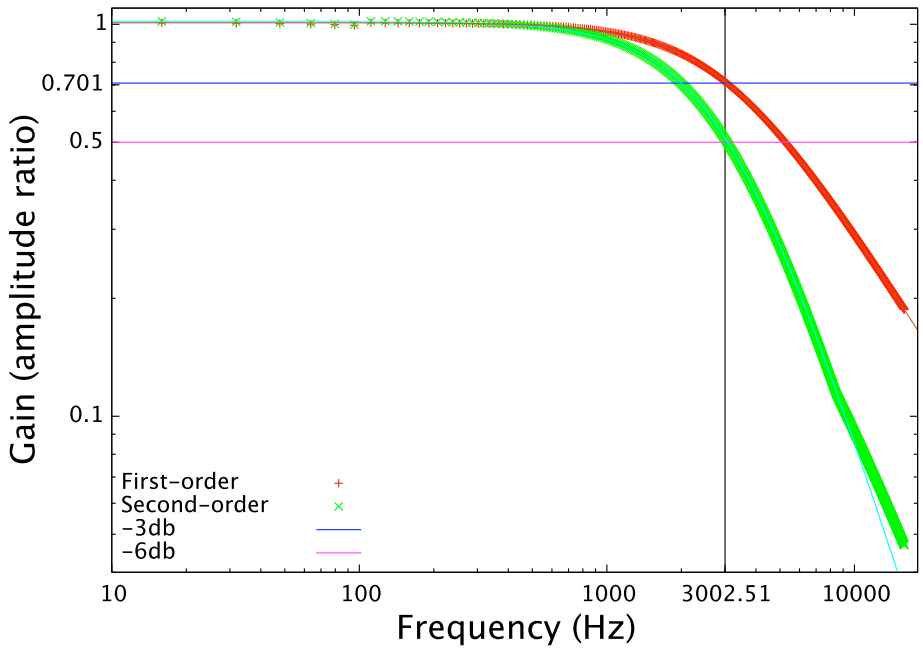

Figure A.3: Bode plot of first order (red) and second-order (green) digital low-pass filter output amplitudes vs frequency. Notice the agreement with theory at the calculated cutoff frequency. The best fit lines for each filter give cutoff frequencies of 3030.65Hz and 3002.51Hz for first and second order, respectively.

A.5 Filter Stability and Conclusions

We note that (A.21) and (A.24) are functions of a complex variable z and that both have poles in the denominator. (A.21) can be multiplied through by z to give $z - e^{-\omega_{rc}T}$ in the denominator, which gives a pole (infinite value) at $z = e^{-\omega_{rc}T}$. Since this value, specifically the value that we have used in this filter, is < 1, this filter is guaranteed to converge, which is essential for software or firmware implementation. The pole falls inside of the unit circle in the complex plane, and will therefore converge to zero after some time (Smith, 2014). The second filter can be factored into two of these terms with the same poles. This explains why the second order filter attenuates better than the first order-the pole; the second order filter is a second order pole.

The first and second order lowpass filters behave as theoretically predicted, and their implementation is straightforward. By contrast, to take the input samples to the frequency domain, i.e. using an FFT algorithm to perform filtration, would require $Nlog(N)$ calculations, where N is the number of samples in the FFT. The IIR process requires only 2 calculations for the first order and

3 for the second order filter, so $(n+1)$ calculations for an n-order filter. The cousin of the IIR, the FIR filter, takes as many calculations as there are filter coefficients, which tends to be O(100) for any reliable filter.